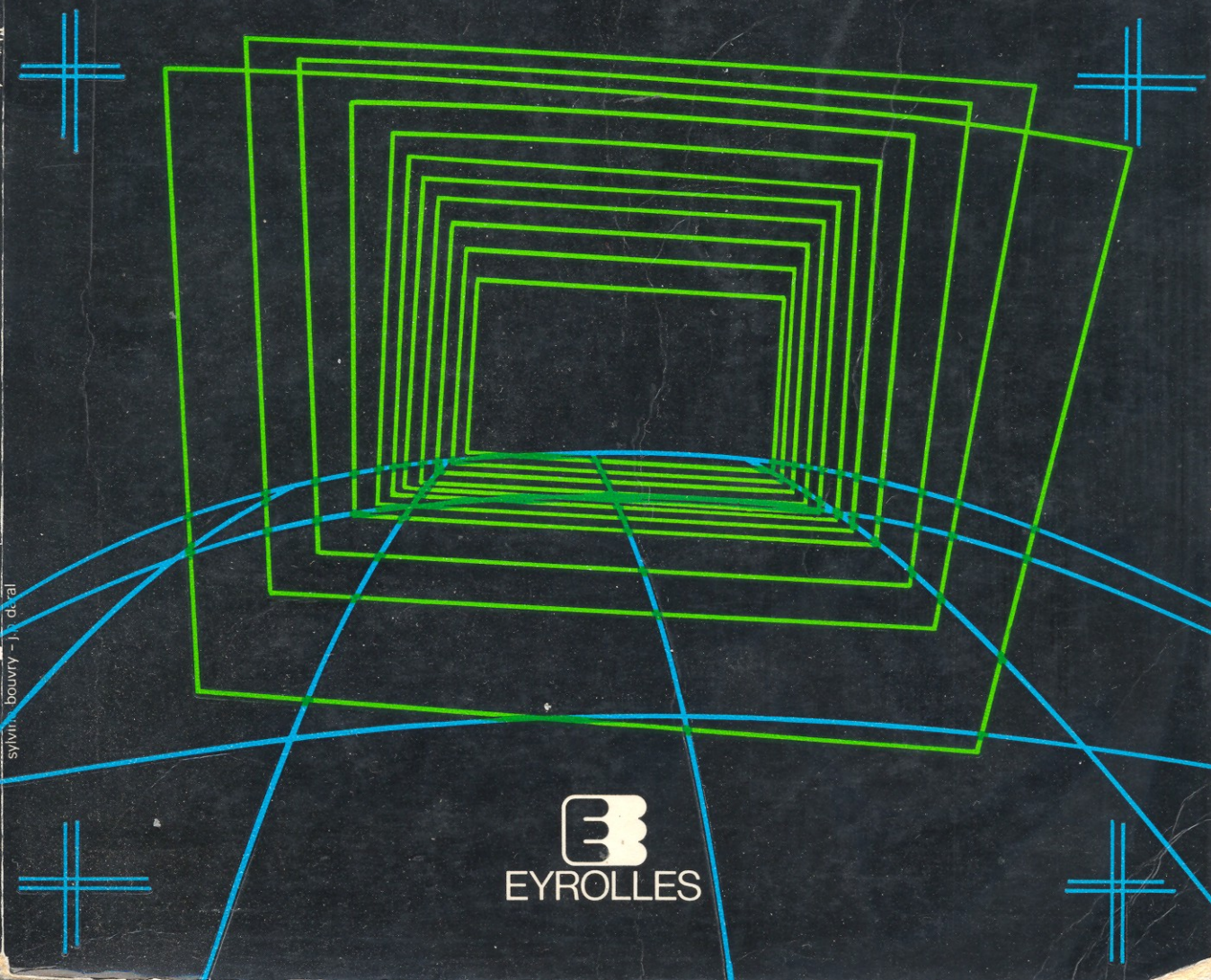
MATHÉMATIQUES
SUR MICRO-ORDINATEUR

1. ANALYSE

Alain REVERCHON
Marc DUCAMP



EYROLLES

MATHÉMATIQUES
SUR MICRO-ORDINATEUR

1. ANALYSE

CHEZ LE MEME EDITEUR

BILLES	- *Exercices d'application du L.S.E.* - 152 p. ; 1982.
BRAUNSCHWEIG	- *La simulation sur micro-ordinateur* - Les modèles de dynamique des systèmes - 184 p. ; 1984, (coll. Informatique et Entreprise).
CANAL	- *Parler L.S.E. et apprendre à l'utiliser* - 160 p. ; 1983, (coll. Micro-ordinateurs).
DUCAMP et SCHAEFFER	- *Faites vos jeux avec Commodore 64* - 192 p. ; 1984, (coll. Microplus).
DE GEETER	- *Forth pour micros* - 192 p. ; 1984, (coll. Micro-ordinateurs).
GONDRAN	- *Introduction aux systèmes experts* - 104 p. ; 1985.
KRUTCH	- *Expériences d'intelligence artificielle en Basic* - 128 p. ; 1984, (coll. Micro-ordinateurs).
LUCAS, MARTIN et De SABLET	- *UNIX* - Mécanismes de base. Langage de commande. Utilisation - 204 p. ; 1984, (coll. Informatique et Entreprise).
MIRANDA et BUSTA	- *L'art des bases de données* - tome 1 - 248 p. ; 1984.
POLITIS et VANRYB	- *Le système d'exploitation MS-DOS* - Versions 1 et 2 - 216 p. ; 1984.
SAGUEZ	- *Télécommande avec votre micro-ordinateur* - 136 p. ; 1983, (coll. Micro-ordinateurs).
SAGUEZ et ANDRIEUX	- *Maîtrisez les interfaces de votre micro-ordinateur* - 144 p. ; 1984, (coll. Micro-ordinateurs).
TRIO	- *Microprocesseurs 8086-8088* - Architecture et programmation. Coprocesseur 8087 - 240 p. ; 1984.
VULDY	- *Graphisme 3D sur votre micro-ordinateur* - 128 p. ; 1984, (coll. Micro-ordinateurs).
WEIDENFELD	- *LOGO* - 160 p. ; 1984.

MATHÉMATIQUES
SUR MICRO-ORDINATEUR

1. ANALYSE

par

Alain REVERCHON
Marc DUCAMP

EYROLLES
61, boulevard Saint-Germain — 75005 Paris
1984

Si vous désirez être tenu au courant de nos publications, il vous suffit d'adresser votre carte de visite au :

Service «Presse», Éditions EYROLLES
61, Boulevard Saint-Germain,
75240 PARIS CEDEX 05,

en précisant les domaines qui vous intéressent. Vous recevrez régulièrement un avis de parution des nouveautés en vente chez votre libraire habituel.

Avant-propos

La recherche scientifique, utilisant de plus en plus le calcul numérique, a été à l'origine du développement des ordinateurs; elle est aujourd'hui la meilleure cliente des constructeurs de super-ordinateurs.

Les micro-ordinateurs semblent avoir oublié la noble vocation de leurs parents, et leur utilisation est aujourd'hui principalement ludique. Cependant, ils sont la plupart du temps capables d'effectuer avec précision et rapidité toutes sortes de calculs numériques.

Dans le but d'exploiter ces capacités oubliées, cet ouvrage s'adresse à tous ceux qui désirent utiliser leur micro-ordinateur dans un but scientifique, et poursuit un triple objectif:

- *familiariser aux mathématiques les non-spécialistes de façon originale, grâce à des programmes complets, d'une utilisation aisée, permettant une compréhension* par l'exemple *des résultats théoriques;*
- *permettre aux étudiants d'appliquer sur l'ordinateur les théories qui leur sont exposées en cours;*
- *fournir des* outils *en vue de la réalisation de programmes spécialisés (en mécanique, en électronique ou en astronomie, par exemple).*

Nous avons donc adopté la démarche suivante: la théorie reste qualitative autant que possible, de manière à privilégier la compréhension intuitive plutôt que les démonstrations fastidieuses, ce qui explique la place importante consacrée aux exemples d'application, et la présence de deux programmes utilisant les possibilités graphiques de l'ordinateur.

Tous les chapitres sont indépendants, et peuvent donc être abordés dans un ordre quelconque. Chaque chapitre traite d'un sujet particulier, et les programmes qu'il comporte ont été étudiés pour fonctionner simultanément; il est donc possible de comparer simplement et rapidement différentes méthodes. Pour chaque sujet, les résultats théoriques indispensables et la méthode de résolution sont tout d'abord exposés. Viennent ensuite le programme et son utilisation. Enfin, de nombreux exemples commentés illustrent les avantages et les limitations des méthodes étudiées.

Les programmes sont écrits en BASIC standard, ce qui permet dans la plupart des cas une introduction sans modifications; seuls les programmes graphiques nécessitent une légère adaptation: les trois fonctions concernées ne portent pas toujours le même nom d'un BASIC à l'autre.

Nous espérons que cet ouvrage atteindra ses objectifs, et vous souhaitons de nombreuses heures passionnantes avec votre ordinateur.

Table des matières

1 Suites - Séries

1. Introduction

Le but du chapitre est le calcul de limites de suites et de séries.

Nous verrons d'abord deux programmes recherchant les limites des suites récurrentes (simples et doubles), puis nous examinerons le cas des séries.

Pour utiliser les trois programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu :

```
100 CLS
110  PRINT  TAB( 5);"SUITES-SERIES": PRINT : PRINT
120  PRINT : PRINT "1- SUITE RECURRENTE"
130  PRINT : PRINT "2- SUITE RECURRENTE DOUBLE"
140  PRINT : PRINT "3- SERIE"
150  PRINT : PRINT "4- FIN"
190  PRINT : PRINT : INPUT "VOTRE CHOIX:";E
200  ON E GOTO 1000,2000,3000,4000
210  GOTO 100
4000  END
```

L'ensemble nécessite environ 2 Koctets de mémoire.

2. *Suite récurrente*

a. Principe

On se propose de chercher la limite d'une suite définie par récurrence par:

$$\begin{cases} u_0 \\ u_{k+1} = f(u_k) \end{cases}$$

Pour cela, l'ordinateur va calculer les n premiers termes de la suite.

Mais l'examen de la seule valeur u_n ne permet aucune conclusion sur la valeur de la limite; c'est pour cette raison que le programme affiche dix valeurs réparties entre u_1 et u_n; ces dix valeurs permettent d'apprécier la convergence de la suite.

La relation de récurrence doit être définie dans le programme à la ligne 111Ø sous la forme U = COS(U) pour la relation $u_{k+1} = \cos(u_k)$.

Le programme demande ensuite les deux premiers termes u_0 et u_1 et le nombre de termes à calculer.

Pour pouvoir afficher dix valeurs de la suite, le programme arrondit n à la dizaine supérieure (ainsi, si on demande 937 termes, le programme en calculera 94Ø).

b. Programme

```
1000 CLS
1010  PRINT  TAB( 5);"SUITE RECURRENTE": PRINT : PRINT

1020  PRINT "1- PROGRAMMATION DE LA SUITE": PRINT
1030  PRINT "2- CALCUL": PRINT
1040  PRINT : INPUT "VOTRE CHOIX:";E: PRINT : PRINT
1050  IF E = 1 THEN  LIST 1110: END
1060  INPUT "TERME INITIAL U0:";U
1070  INPUT "NOMBRE DE TERMES A CALCULER:";N: PRINT
1080 N =  INT ((N + 9) / 10)
1090  FOR I = 1 TO 10
1100  FOR J = 1 TO N
1110 U =  COS (U)
```

```
1120  NEXT J
1130  PRINT U
1140  NEXT I
1150  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
1160  IF Z$ = "O" THEN 1000
1170  END
```

c. Exemples commentés

- ***Exemple 1*** :

```
1110 U=(U/3)+4


   SUITE RECURRENTE


TERME INITIAL UO:-3
NOMBRE DE TERMES A CALCULER:30

5.66666667
5.98765432
5.99954275
5.99998306
5.99999937
5.99999998
6
6
6
6

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La limite de la suite :

$$\begin{cases} u_0 = -3 \\ u_{k+1} = \dfrac{u_k}{3} + 4 \end{cases}$$

est trouvée de façon exacte : $l = 6$.

• ***Exemple 2*** :

```
1110 U=COS(U)


   SUITE RECURRENTE


TERME INITIAL U0:2
NOMBRE DE TERMES A CALCULER:30

.610065299
.775994613
.727634792
.74256755
.738019141
.739410808
.738985575
.739115562
.739075833
.739087977

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
   SUITE RECURRENTE


TERME INITIAL U0:2
NOMBRE DE TERMES A CALCULER:100

.746749602
.739232918
.739087977
.739085188
.739085134
.739085134
.739085134
.739085134
.739085134
.739085134

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Dans le premier cas où seulement trente valeurs sont calculées, l'examen des résultats ne fournit que la valeur très approximative $l \simeq 0{,}739$.

Dans le second cas où cent termes sont calculés, on peut affirmer que la limite de la suite est obtenue avec la précision maximale autorisée par l'ordinateur (à savoir 10^{-8}, et non 10^{-9} car le dernier chiffre est sujet aux erreurs numériques).

3. Suite récurrente double

a. Principe

Le programme permet de chercher la limite d'une suite définie par:

$$\begin{cases} u_0, u_1 \\ u_{k+2} = f(u_{k+1}, u_k) \end{cases}$$

Le principe du calcul est identique à celui du programme précédent.

La relation de récurrence doit être programmée à la ligne 211Ø. Par exemple:

$$u_{k+2} = \frac{1}{2}(3u_{k+1} - u_k)$$

se programmera:

```
211Ø W = (3*V-U)/2
```

b. Programme

```
2000 CLS
2010  PRINT  TAB( 5);"SUITE RECURRENTE DOUBLE": PRINT :
     PRINT
2020  PRINT "1- PROGRAMMATION DE LA SUITE": PRINT
2030  PRINT "2- CALCUL": PRINT
2040  PRINT : INPUT "VOTRE CHOIX:";E: PRINT : PRINT
2050  IF E = 1 THEN  LIST 2110: END
2060  INPUT "TERMES INITIAUX U0,U1:";U,V
2070  INPUT "NOMBRE DE TERMES A CALCULER:";N: PRINT : PRINT

2080 N =  INT ((N + 9) / 10)
2090  FOR I = 1 TO 10
2100  FOR J = 1 TO N
2110 W = (3 * V - U) / 2
2120 U = V:V = W
```

```
2130  NEXT J
2140  PRINT V
2150  NEXT I
2160  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
2170  IF Z$ = "O" THEN 2000
2180  END
```

c. Exemple commenté

```
2110 W=(3*V-U)/2
```

```
    SUITE RECURRENTE DOUBLE


TERMES INITIAUX U0,U1:1,2
NOMBRE DE TERMES A CALCULER:50


2.96875
2.99902344
2.99996948
2.99999905
2.99999997
3
3
3
3
3

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La suite définie par:

$$\begin{cases} u_0 = 1, u_1 = 2 \\ u_{k+2} = \frac{1}{2}(3u_{k+1} - u_k) \end{cases}$$

admet donc $l = 3$ pour limite.

4. Série

a. Principe

Le but du programme est la détermination d'une valeur approchée de la somme d'une série (dans le cas où celle-ci est convergente), c'est-à-dire une valeur approchée de :

$$\lim_{N\to\infty} \sum_{n=n_0}^{N} u_n = \sum_{n=n_0}^{\infty} u_n$$

La méthode consiste à calculer $\sum_{n=n_0}^{m} u_n$ avec m suffisamment grand.

La programmation de l'expression de u_n se fera en ligne 311Ø. Par exemple, pour :

$$u_n = \frac{1}{n}$$

on introduira :

```
311Ø U = 1/N
```

Il faut ensuite indiquer au programme le nombre m de termes à calculer (qui sera arrondi à la dizaine supérieure), ainsi que l'indice n_0 du premier terme.

b. Programme

```
3000 CLS
3010  PRINT  TAB( 5);"SERIE": PRINT : PRINT
3020  PRINT "1- PROGRAMMATION DE LA SERIE": PRINT
3030  PRINT "2- CALCUL": PRINT
3040  PRINT : INPUT "VOTRE CHOIX:";E: PRINT : PRINT
3050  IF E = 1 THEN  LIST 3110: END
3060  INPUT "NOMBRE DE TERMES A CALCULER:";M: PRINT
3070  INPUT "INDICE DU PREMIER TERME:";N
3080 M =  INT ((M + 9) / 10):S = 0
3090  FOR I = 1 TO 10
3100  FOR J = 1 TO M
3110 U = 1 / N / N
```

```
3120 S = S + U:N = N + 1
3130  NEXT J
3140  PRINT S
3150  NEXT I
3160  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
3170  IF Z$ = "O" THEN 3000
3180  END
```

c. Exemples commentés

- ***Exemple 1*** :

```
3110 U=(2^(-N)+3^(-N))/N


   SERIE


NOMBRE DE TERMES A CALCULER:50

INDICE DU PREMIER TERME:1
1.0936857
1.09852922
1.09861048
1.09861225
1.09861229
1.09861229
1.09861229
1.09861229
1.09861229
1.09861229

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La somme de la série $\sum_{n=1}^{\infty} \frac{2^{-n} + 3^{-n}}{n}$ est ln(3) soit approximativement 1,Ø9861229, ce qui est la valeur donnée par l'ordinateur ; la série converge donc rapidement.

- ***Exemple 2*** :

```
3110 U=1/N/N


    SERIE


NOMBRE DE TERMES A CALCULER:100

INDICE DU PREMIER TERME:1
1.54976773
1.59616324
1.61215012
1.62024396
1.62513273
1.62840552
1.63074991
1.63251187
1.63388445
1.6349839

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    SERIE


NOMBRE DE TERMES A CALCULER:5000

INDICE DU PREMIER TERME:1
1.64293607
1.64393456
1.64426762
1.64443419
1.64453415
1.64460079
1.64464839
1.6446841
1.64471187
1.64473409

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La série étudiée est:

$$\sum_{n=1}^{\infty} \frac{1}{n^2}$$

Avec 1ØØ termes, il semble que l'on ait trois chiffres significatifs: 1,63.

Avec 5 ØØØ termes, le "3" apparaît faux: il faut donc rester très prudent sur le nombre de chiffres significatifs à retenir.

La conclusion la plus audacieuse que l'on puisse faire est:

$$\sum_{n=1}^{\infty} \frac{1}{n^2} \simeq 1{,}644$$

La limite exacte est $\frac{\pi^2}{6} = 1{,}644934067...$

Contrairement à l'exemple précédent, cette série converge très lentement.

On pourra vérifier avec le programme que:

$$\sum_{n=1}^{\infty} \frac{1}{n^4} = \frac{\pi^4}{90}$$

$$\sum_{n=1}^{\infty} \frac{1}{n^6} = \frac{\pi^6}{945}$$

Ces deux séries convergent beaucoup plus rapidement.

d. Les erreurs

Lorsque l'on se trouve confronté à une suite qui converge lentement, le premier réflexe consiste à choisir un nombre élevé de termes, 1ØØ ØØØ par exemple.

Malheureusement, les erreurs numériques se cumulent et peuvent fausser le résultat de façon significative.

En effet, l'incertitude de calcul est en général de 10^{-9}: si l'on calcule 1ØØ ØØØ termes, les erreurs s'ajoutent et l'incertitude totale devient 10^{-4}!

Le choix du nombre de termes m résulte donc d'un compromis entre précision et vitesse de convergence.

Il peut arriver, mais c'est un cas extrême, que les erreurs numériques faussent totalement le résultat. Prenons l'exemple de la série $\sum_{n=1}^{\infty} \frac{1}{n}$ qui est divergente: elle tend vers $+\infty$, mais extrêmement lentement.

Si on calculait 1Ø ØØØ ØØØ termes (calcul en une semaine environ), on obtiendrait un nombre auquel les termes suivants ne s'ajouteraient pas : en effet, ceux-ci seraient trop petits par rapport à la somme pour que l'ordinateur puisse les prendre en compte.

Ainsi, si l'on recalculait la série avec 1ØØ ØØØ ØØØ termes, on obtiendrait le même résultat, ce qui tendrait à faire croire que la série $\sum_{n=1}^{\infty} \frac{1}{n}$ converge !

2 Équations

1. Introduction

La résolution d'équations est d'une importance capitale en mathématiques.

Cependant, on ne peut obtenir le plus souvent que des valeurs approchées des racines.

Il existe des formules algébriques exactes donnant les racines des équations polynômiales de degré inférieur à 5 et le mathématicien Evariste Gallois a démontré que cette résolution exacte n'était plus possible pour les polynômes de degré supérieur ou égal à 5.

La recherche systématique des racines de ces polynômes peut toutefois être effectuée de manière approchée par la méthode de Bairstow.

Les équations non polynômiales peuvent également être résolues grâce à plusieurs méthodes approchées. Nous en examinerons trois: la méthode des dichotomies, la méthode des itérations et la méthode de Newton.

Nous verrons enfin une méthode de résolution des systèmes d'équations à deux inconnues.

Pour utiliser les huit programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu :

```
100 CLS
110  PRINT  TAB( 5);"RESOLUTION D'EQUATIONS": PRINT
120  PRINT : PRINT "1- SECOND DEGRE"
130  PRINT : PRINT "2- TROISIEME DEGRE"
140  PRINT : PRINT "3- QUATRIEME DEGRE"
150  PRINT : PRINT "4- POLYNOME QUELCONQUE"
160  PRINT : PRINT "5- METHODE DES DICHOTOMIES"
170  PRINT : PRINT "6- METHODE DES ITERATIONS"
180  PRINT : PRINT "7- METHODE DE NEWTON"
190  PRINT : PRINT "8- SYSTEMES A 2 INCONNUES"
200  PRINT : PRINT "9- FIN"
210  PRINT : INPUT "VOTRE CHOIX:";E
220  ON E GOTO 1000,2000,3000,4000,5000,6000,7000,8000,
    9000
230  GOTO 100
9000  END
```

L'ensemble nécessite environ 7 Koctets de mémoire.

2. *Équations du second degré*

L'équation à résoudre est :

$ax^2 + bx + c = 0$ avec $a \neq 0$.

a. Résolution théorique

On calcule le discriminant Δ donné par :

$$\Delta = b^2 - 4ac$$

Il faut alors distinguer trois cas :

→ $\Delta < 0$: l'équation n'a pas de racines.

→ $\Delta = 0$: l'équation admet une racine double :

$$x_1 = x_2 = -\frac{b}{2a}$$

→ $\Delta > 0$: l'équation admet deux racines:

$$x_1 = \frac{-b - \sqrt{\Delta}}{2a} \qquad x_2 = \frac{-b + \sqrt{\Delta}}{2a}$$

b. Programme

L'utilisation du programme est très simple: il suffit d'introduire les trois coefficients a, b et c.

Le programme cherche alors les racines et affiche les résultats.

```
1000 CLS
1010  PRINT  TAB( 5);"RESOLUTION DE AX^2+BX+C=0": PRINT
    : PRINT
1020  INPUT "A=";A
1030  INPUT "B=";B
1040  INPUT "C=";C
1050  PRINT
1060 D = B * B - 4 * A * C
1070  IF D < 0 THEN  PRINT "PAS DE SOLUTIONS"
1080  IF D = 0 THEN  PRINT "UNE SOLUTION DOUBLE: "; - B
     / 2 / A
1090  IF D > 0 THEN  PRINT "X1=";( - B -  SQR (D)) / 2 /
    A: PRINT "X2=";( - B +  SQR (D)) / 2 / A
1100  PRINT
1110  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME:";Z$
1120  IF Z$ = "O" THEN 1000
1130  END
```

c. Exemples commentés

```
    RESOLUTION DE AX^2+BX+C=0


A=1
B=1
C=1

PAS DE SOLUTIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
    RESOLUTION DE AX^2+BX+C=0
```

```
A=1
B=2
C=1

UNE SOLUTION DOUBLE: -1

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
    RESOLUTION DE AX^2+BX+C=O


A=1
B=2
C=-15

X1=-5
X2=3

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Les trois équations proposées font apparaître les trois résultats possibles.

3. *Équations du troisième degré*

L'équation à résoudre est:

$$ax^3 + bx^2 + cx + d = 0 \qquad \text{avec } a \neq 0 \tag{1}$$

a. Résolution théorique

On se ramène tout d'abord à l'équation:

$$x^3 + b'x^2 + c'x + d' = 0 \tag{2}$$

en posant:

$$b' = \frac{b}{a} \qquad c' = \frac{c}{a} \qquad d' = \frac{d}{a}$$

On pose ensuite $X = x + \frac{b'}{3}$, ce qui s'écrit encore $x = X - \frac{b'}{3}$. En remplaçant dans l'équation (2), puis en développant, on aboutit à:

$$X^3 + pX + q = 0 \tag{3}$$

avec :

$$p = c' - \frac{b'^2}{3} \quad \text{et} \quad q = d' - \frac{c'b'}{3} + 2\left(\frac{b'}{3}\right)^3.$$

On recherche alors X sous la forme $\sqrt[3]{\alpha} + \sqrt[3]{\beta}$.

En posant arbitrairement $\alpha + \beta = -q$, et en développant l'expression $X^3 + pX + q = 0$, on obtient la relation supplémentaire :

$$\alpha\beta = \left(-\frac{p}{3}\right)^3$$

α et β sont donc racines de l'équation :

$$u^2 + qu - \left(\frac{p}{3}\right)^3 = 0 \qquad (4)$$

dont le discriminant vaut $\Delta = \dfrac{4p^3 + 27q^2}{27}$

→ *Cas où Δ est positif ou nul :*

Les solutions sont alors :

$$\alpha = \frac{-q + \sqrt{\Delta}}{2} \qquad \text{et} \qquad \beta = \frac{-q - \sqrt{\Delta}}{2}$$

L'équation (3) admet donc la racine réelle :

$$X_1 = \sqrt[3]{\alpha} + \sqrt[3]{\beta}$$

On peut alors montrer, en divisant le polynôme $X^3 + pX + q$ par $X - (\sqrt[3]{\alpha} + \sqrt[3]{\beta})$, que les deux autres racines de (3) sont complexes et sont données par :

$$X_2 = j\sqrt[3]{\alpha} + j^2\sqrt[3]{\beta}$$
$$X_3 = j^2\sqrt[3]{\alpha} + j\sqrt[3]{\beta} = \overline{X_2}$$

j étant le nombre complexe tel que $j^3 = 1$ $\left(j = -\frac{1}{2} + \frac{\sqrt{3}}{2}\,i\right)$.

Dans le cas où Δ est nul, ces deux racines sont confondues et valent :

$$X_2 = X_3 = -\frac{X_1}{2}$$

→ *Cas où Δ est négatif :*

On cherche maintenant X sous la forme :

$$X = r \cos \theta$$

En remplaçant dans l'équation (3) et en utilisant la formule :

$$\cos^3\theta = \frac{1}{4}\cos 3\theta + \frac{3}{4}\cos\theta$$

il vient, en posant $r = \sqrt{\frac{-4p}{3}}$:

$$(-\frac{p}{3})\sqrt{\frac{-4p}{3}}\cos 3\theta + q = 0 \qquad (5)$$

Cette équation n'admet de solutions que si :

$$\frac{3q}{2p}\sqrt{\frac{3}{-p}} \leqslant 1$$

ce qui est équivalent à $\Delta \leqslant 0$.

Un angle solution de l'équation (5) est alors :

$$\theta_1 = \frac{1}{3}\text{Arc cos}\,[\,\frac{3q}{2p}\sqrt{\frac{3}{-p}}\,]$$

Les deux autres solutions sont :

$$\theta_2 = \theta_1 + \frac{2\pi}{3}$$

$$\theta_3 = \theta_1 + \frac{4\pi}{3}$$

Les trois solutions de l'équation (3) sont alors :

$$X_1 = \sqrt{\frac{-4p}{3}}\cos\theta_1$$

$$X_2 = \sqrt{\frac{-4p}{3}}\cos\theta_2$$

$$X_3 = \sqrt{\frac{-4p}{3}}\cos\theta_3$$

Récapitulation de la méthode :

On calcule :

$$b' = \frac{b}{a} \qquad c' = \frac{c}{a} \qquad d' = \frac{d}{a}$$

puis :

$$p = c' - \frac{b'^2}{3} \qquad q = d' - \frac{c'b'}{3} + 2(\frac{b'}{3})^3$$

et :

$$\Delta = \frac{4p^3}{27} + q^2.$$

→ *Si* $\Delta > 0$*:*

L'équation admet une solution réelle :

$$x = \sqrt[3]{\frac{-q + \sqrt{\Delta}}{2}} + \sqrt[3]{\frac{-q - \sqrt{\Delta}}{2}} - \frac{b'}{3}$$

→ *Si* $\Delta = 0$*:*

L'équation admet deux solutions :

$$x_1 = 2\sqrt[3]{\frac{-q}{2}} - \frac{b'}{3}$$

$$x_2 = -\sqrt[3]{\frac{-q}{2}} - \frac{b'}{3}$$ (solution double)

Ces deux racines peuvent éventuellement être égales (cas d'une racine triple).

→ *Si* $\Delta < 0$*:*

On calcule :

$$\theta_1 = \frac{1}{3} \text{ Arc cos } [\ \frac{3q}{2p}\sqrt{\frac{3}{-p}}\ ]$$

Il y a alors trois solutions :

$$x_1 = \sqrt{\frac{-4p}{3}} \cos\theta_1 - \frac{b'}{3}$$

$$x_2 = \sqrt{\frac{-4p}{3}} \cos(\theta_1 + \frac{2\pi}{3}) - \frac{b'}{3}$$

$$X_3 = \sqrt{\frac{-4p}{3}} \cos(\theta_1 + \frac{4\pi}{3}) - \frac{b'}{3}$$

On remarque que l'équation admet toujours au moins une racine ; c'est le cas de toutes les équations polynomiales de degré impair.

b. Programme

Le programme est ici la traduction exacte de la méthode. Il présente cependant quelques particularités :

→ Nous avons écrit P*P*P au lieu de P ↑ 3 car le calcul est alors plus précis et plus rapide.

→ La fonction Arc cos est absente sur la plupart des micro-ordinateurs. On utilise alors la formule :

$$\text{Arc cos } x = \frac{\pi}{2} - \text{Arc tg } \frac{x}{\sqrt{1-x^2}}$$

Le calcul de θ_1 se simplifie en :

$$\theta_1 = \frac{1}{3}\left(\frac{\pi}{2} + \text{Arc tg}\left(\frac{q}{\sqrt{-\Delta}}\right)\right)$$

→ Prendre la racine cubique d'un nombre revient à l'élever à la puissance $\frac{1}{3}$.

Cependant, tous les ordinateurs n'admettent pas que l'on élève un nombre négatif à une puissance fractionnaire.

On procède alors de la manière suivante : la valeur absolue du nombre est élevée à la puissance $\frac{1}{3}$, et le signe est restitué ensuite au résultat.

On écrira donc : SGN(X)*(ABS(X)↑(1/3))

à la place de : X↑(1/3)

```
10 PI = 3.1415926536
2000 CLS
2010  PRINT "RESOLUTION DE AX^3+BX^2+CX+D=0": PRINT : PRINT

2020  INPUT "A=";A
2030  INPUT "B=";B
2040  INPUT "C=";C
2050  INPUT "D=";D
2060  PRINT
2070 B = B / A / 3:C = C / A:D = D / A
2080 P = C - 3 * B * B
2090 Q = D + B * (2 * B * B - C)
2100 T = Q * Q + 4 / 27 * P * P * P
2110  IF T > 0 THEN 2250
2120  IF T = 0 THEN 2210
2130  REM  *** CAS T<0 ***
2140 T =  SQR ( - T)
2150 X = PI / 2 +  ATN (Q / T)
2160  FOR K = 1 TO 3
2170  PRINT "X";K;"="; SQR ( - 4 * P / 3) *  COS ((X +
    2 * K * PI) / 3) - B
```

```
2180  NEXT K
2190  GOTO 2290
2200  REM  *** CAS T=0 ***
2210  PRINT "X1="; -  SGN (Q) * (4 *  ABS (Q)) ^ (1 / 3
    ) - B
2220  PRINT "X2="; SGN (Q) * ( ABS (Q) / 2) ^ (1 / 3) -
    B;" (DOUBLE)"
2230  GOTO 2290
2240  REM  *** CAS T>0 ***
2250 T =  SQR (T)
2260 X =  SGN (T - Q) * ( ABS (T - Q) / 2) ^ (1 / 3)
2270 X = X -  SGN (T + Q) * ( ABS (T + Q) / 2) ^ (1 / 3
    ) - B
2280  PRINT "X=";X
2290  PRINT
2300  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME:";Z$
2310  IF Z$ = "O" THEN 2000
2320  END
```

c. Exemples commentés

Nous avons volontairement choisi des équations admettant des racines entières afin de montrer l'excellente précision des résultats.

```
RESOLUTION DE AX^3+BX^2+CX+D=0


A=1
B=4
C=4
D=3

X=-3

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^3+BX^2+CX+D=0


A=1
B=3
C=-10
D=-24
```

```
X1=-4
X2=-2
X3=3

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^3+BX^2+CX+D=0


A=1
B=9
C=15
D=-25

X1=1
X2=-5 (DOUBLE)

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^3+BX^2+CX+D=0


A=1
B=-3
C=3
D=-1

X1=1
X2=1 (DOUBLE)

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

4. Équations du quatrième degré

L'équation à résoudre est:

$$ax^4 + bx^3 + cx^2 + dx + e = 0 \qquad \text{avec } a \neq 0 \tag{1}$$

a. Résolution théorique

On se ramène tout d'abord à l'équation:

$$x^4 + 4b'x^3 + c'x^2 + d'x + e' = 0 \tag{2}$$

en posant :

$$b' = \frac{b}{4a} \qquad c' = \frac{c}{a} \qquad d' = \frac{d}{a} \qquad e' = \frac{e}{a}$$

On pose ensuite $X = x + b'$, et on remplace x par $(X - b')$ dans l'équation (2).

On aboutit alors à :

$$X^4 + pX^2 + qX + r = 0 \tag{3}$$

avec :

$$p = c' - 6b'^2$$
$$q = d' + 8b'^3 - 2b'c'$$
$$r = e' - d'b' + c'b'^2 + 3b'^4$$

On recherche alors t, u et v de sorte que :

$$X^4 + pX^2 + qX + r = (X^2 + tX + u)(X^2 - tX + v)$$

Posons :

$$S = u + v \qquad \text{et} \qquad P = uv$$

L'identification des deux polynômes conduit aux résultats suivants :

$$P = r$$

$$t = \frac{q}{v - u}$$

$$S^3 - pS^2 - 4rS + (4rp - q^2) = 0 \tag{4}$$

Nous examinerons plus tard le cas $u = v$ qui correspond encore à $q = 0$.

La recherche d'une solution S_0 de cette équation s'effectue selon la méthode exposée précédemment ; u et v sont alors racines de l'équation :

$$y^2 - S_0 y + r = 0$$

Le discriminant de cette équation est $\Delta = S_0^2 - 4r$.

→ Si $\Delta < 0$, l'équation (1) n'a pas de solutions.

→ Si $\Delta > 0$, on obtient alors :

$$u = \frac{S_0 - \sqrt{\Delta}}{2} \qquad v = \frac{S_0 + \sqrt{\Delta}}{2} \qquad t = \frac{q}{\sqrt{\Delta}}$$

(Δ ne peut pas être nul puisqu'on a supposé u différent de v.)

Les solutions de l'équation (3) sont alors celles des équations :

$$X^2 + tX + u = 0$$
$$X^2 - tX + v = 0$$

On peut donc obtenir 0, 2 ou 4 solutions. Étudions maintenant le cas où q est nul. L'équation (3) devient :

$$X^4 + pX^2 + r = 0$$

qui est une équation bicarrée. Posons $\Delta = p^2 - 4r$:

→ Si $\Delta < 0$, il n'y a pas de solutions.

→ Si $\Delta \geqslant 0$, les racines sont celles de :

$$X^2 = \frac{-p + \sqrt{\Delta}}{2} \qquad \text{et} \qquad X^2 = \frac{-p - \sqrt{\Delta}}{2}$$

Suivant le signe de ces quantités, on peut encore obtenir 0, 2 ou 4 solutions.

b. Programme

On retrouve ici les particularités du programme précédent.

```
10 PI = 3.1415926536
3000 CLS
3010  PRINT "RESOLUTION DE AX^4+BX^3+CX^2+DX+E=0": PRINT
    : PRINT
3020  INPUT "A=";A
3030  INPUT "B=";B
3040  INPUT "C=";C
3050  INPUT "D=";D
3060  INPUT "E=";E
3070  PRINT
3080 B = B / A / 4:C = C / A:D = D / A:E = E / A
3090 P = C - 6 * B * B
3100 Q = D + 2 * B * (4 * B * B - C)
3110 R = E - D * B + C * B * B - 3 * B ^ 4
3120  IF Q = 0 THEN 3450
3130 F =  - 4 * R - P * P / 3
3140 G = 8 * R * P / 3 - Q * Q - 2 / 27 * P ^ 3
3150  IF G = 0 THEN S = 0: GOTO 3210
3160 DL = 4 * (F ^ 3) / 27 + G * G
3170  IF DL < 0 THEN S =  SQR ( - 4 * F / 3) *  COS ((P
    I / 2 +  ATN (G /  SQR ( - DL))) / 3): GOTO 3210
3180 DL =  SQR (DL)
3190 S =  SGN (DL - G) * (( ABS (DL - G) / 2) ^ (1 / 3)
    )
3200 S = S -  SGN (DL + G) * (( ABS (DL + G) / 2) ^ (1 /
    3))
```

```
3210 S = S + P / 3
3220 D2 = S * S - 4 * R
3230  IF D2 < 0 THEN 3430
3240 U = (S -  SQR (D2)) / 2
3250 V = (S +  SQR (D2)) / 2
3260 T = Q /  SQR (D2)
3270 D3 = T * T - 4 * U
3280  IF  ABS (D3) < 1E - 6 THEN D3 = 0
3290  IF D3 < 0 THEN 3320
3300  PRINT ( - T +  SQR (D3)) / 2 - B
3310  PRINT ( - T -  SQR (D3)) / 2 - B
3320 D4 = Q * Q / D2 - 2 * S - 2 *  SQR (D2)
3330  IF  ABS (D4) < 1E - 6 THEN D4 = 0
3340  IF D4 >  = 0 THEN 3370
3350  IF D3 < 0 THEN 3430
3360  GOTO 3390
3370  PRINT (T +  SQR (D4)) / 2 - B
3380  PRINT (T -  SQR (D4)) / 2 - B
3390  PRINT
3400  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME:";Z$
3410  IF Z$ = "O" THEN 3000
3420  END
3430  PRINT "PAS DE SOLUTIONS"
3440  GOTO 3390
3450 T = P * P - 4 * R
3460  IF T < 0 THEN 3430
3470  IF (R > 0) AND (P > 0) THEN 3430
3480  IF R < 0 THEN 3520
3490 Z =  SQR (( - P -  SQR (T)) / 2)
3500  PRINT Z - B
3510  PRINT  - Z - B
3520 Z =  SQR (( - P +  SQR (T)) / 2)
3530  PRINT Z - B
3540  PRINT  - Z - B
3550  GOTO 3390
```

c. Exemples commentés

```
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=0


A=-2
B=4
```

```
C=90
D=68
E=-160

8
1
-2
-5

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=0


A=1
B=13
C=15
D=-73
E=44

-4
-11
1
1

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=0


A=1
B=4
C=6
D=4
E=1

-1
-1
-1
-1

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=0


A=-3
B=2
```

```
C=-6
D=-3
E=-8

PAS DE SOLUTIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=O


A=1
B=9
C=-39
D=-40
E=-48

4
-12

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION DE AX^4+BX^3+CX^2+DX+E=O


A=1
B=3
C=3
D=1
E=O

2.32830644E-10
-1.00046767
-.999766164
-.999766164

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

La précision des résultats est excellente, sauf dans le dernier exemple.

Cela provient du fait que (—1) est racine multiple (d'ordre 3) de l'équation, et les racines multiples posent toujours en calcul numérique des problèmes de précision : nous en étudierons les causes lors de l'étude de la méthode de Bairstow.

5. Méthode de Bairstow

Cette méthode permet le calcul approché de toutes les racines d'un polynôme de degré quelconque n :

$$P(x) = a_0x^n + a_1x^{n-1} + ... + a_{n-1}x + a_n = 0$$

a. Principe

La méthode consiste à décomposer le polynôme P en produit de polynômes plus simples, de degré 2 et éventuellement de degré 1.

La première étape consiste donc à déterminer deux valeurs p et q telles que :

$$P(x) = (x^2 + px + q) \,.\, Q(x)$$

Les racines de P(x) sont alors d'une part celles de $x^2 + px + q = 0$ que l'on sait parfaitement déterminer, et d'autre part celles de Q(x), qui est un polynôme de degré n—2.

On applique de nouveau le processus à Q(x), et ainsi de suite jusqu'à ce que toutes les racines soient déterminées.

La difficulté réside donc dans la détermination des coefficients p et q : on utilise alors une méthode itérative.

Soient p_0 et q_0 donnés. Il n'y a pas de raison particulière pour que $x^2 + p_0x + q_0$ divise le polynôme P(x) ; on a donc la relation générale :

$$P(x) = (x^2 + p_0x + q_0) \,.\, B(x) + (R_0x + S_0)$$

où R_0 et S_0 sont deux réels.

La méthode de Bairstow va vous permettre de calculer deux valeurs p_1 et q_1 telles que :

$$P(x) = (x^2 + p_1x + q_1) \,.\, B(x) + (R_1x + S_1)$$

avec :

$$|R_1| < |R_0| \qquad \text{et} \qquad |S_1| < |S_0|$$

Les valeurs (p_1,q_1) obtenues sont ainsi "meilleures" que (p_0,q_0).

On réitère le procédé jusqu'à ce que les différences $p_{n+1} - p_n$ et $q_{n+1} - q_n$ soient inférieures à une valeur fixée.

Nous nous contenterons de donner la transformation permettant de passer des valeurs (p_n,q_n) aux valeurs (p_{n+1},q_{n+1}) ; sa démonstration sort du cadre de cet ouvrage.

A partir des coefficients (a_i) du polynôme P, on calcule :

→ *les coefficients (b_i) :*

$$b_0 = a_0$$
$$b_1 = a_1 - pb_0$$
$$b_2 = a_2 - pb_1 - qb_0$$
$$b_3 = a_3 - pb_2 - qb_1$$
.....
$$b_{n-1} = a_{n-1} - pb_{n-2} - qb_{n-3}$$
$$b_n = a_n - pb_{n-1} - qb_{n-2}$$

Ces coefficients sont ceux du polynôme B, quotient de la division de P par $(x^2 + px + q)$.

→ *le tableau des (c_i) :*

$$c_0 = b_0$$
$$c_1 = b_1 - p.c_0$$
$$c_2 = b_2 - p.c_1 - q.c_0$$
.....
$$c_{n-2} = b_{n-2} - p.c_{n-3} - q.c_{n-4}$$
$$c_{n-1} = 0 - p.c_{n-2} - q.c_{n-3}$$

On calcule alors :

$$D = c_{n-2}^2 - c_{n-1} \cdot c_{n-3}$$
$$U = b_{n-1} \cdot c_{n-2} - b_n \cdot c_{n-3}$$
$$V = b_n \cdot c_{n-2} - b_{n-1} \cdot c_{n-1}$$

et alors :

$$p_{n+1} = p_n + \frac{U}{D}$$

$$q_{n+1} = q_n + \frac{V}{D}$$

b. Programme

Après avoir fourni au programme le degré du polynôme ainsi que tous les coefficients, il faut introduire la précision p souhaitée : 1 correspond à une précision médiocre, alors que 8 correspond à la précision maximale.

Le programme calcule alors l'erreur maximale $E = 10^{-p}$ tolérée pour les différences :

$|p_{n+1} - p_n|$ et $|q_{n+1} - q_n|$

REMARQUE : Le coefficient A(Ø) est celui du terme de degré n.

Le programme demande enfin les valeurs initiales de p et q. Le couple (0,0) conduit en général au résultat ; cependant, si beaucoup de coefficients sont nuls, ces valeurs s'avèrent inadaptées, il convient alors de relancer le programme avec deux nouvelles valeurs.

```
20  DIM A(40),B(40),C(40)
4000 CLS
4010  PRINT "RACINES D'UN POLYNOME DE DEGRE N": PRINT :
     PRINT
4020  INPUT "N=";N
4030  IF N < 3 THEN 4020
4040  PRINT : FOR I = 0 TO N: PRINT "A(";I;: INPUT ")="
    ;A(I): NEXT I: PRINT
4050  FOR I = 1 TO N:A(I) = A(I) / A(0): NEXT I:A(0) =
    1
4060  PRINT : INPUT "PRECISION (1 A 8) :";E:E = 10 ^ ( -
    E)
4070  INPUT "P,Q:";P,Q
4080  PRINT :S = 0
4090 J = 0
4100 B(0) = A(0):B(1) = A(1) - P * B(0)
4110  FOR K = 2 TO N
4120 B(K) = A(K) - P * B(K - 1) - Q * B(K - 2)
4130  NEXT K
4140 C(0) = B(0):C(1) = B(1) - P * C(0)
4150  IF N = 3 THEN 4190
4160  FOR K = 2 TO N - 2
4170 C(K) = B(K) - P * C(K - 1) - Q * C(K - 2)
4180  NEXT K
4190 C(N - 1) =  - P * C(N - 2) - Q * C(N - 3)
4200 D = C(N - 2) * C(N - 2) - C(N - 1) * C(N - 3)
4210 U = B(N - 1) * C(N - 2) - B(N) * C(N - 3)
4220 V = B(N) * C(N - 2) - B(N - 1) * C(N - 1)
4230  IF D = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    4420
4240 Y = U / D:Z = V / D
4250 P = P + Y:Q = Q + Z
4260  IF ( ABS (Y) +  ABS (Z)) < E THEN 4300
4270  IF J < 100 THEN J = J + 1: GOTO 4100
4280  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS..."
4290  GOTO 4420
4300 T = P * P - 4 * Q
4310  IF T < 0 THEN 4350
```

```
4320  PRINT ( - P +  SQR (T)) / 2
4330 S = 1
4340  PRINT ( - P -  SQR (T)) / 2
4350 N = N - 2
4360  FOR I = 1 TO N
4370 A(I) = B(I)
4380  NEXT I
4390  IF N > 2 THEN 4090
4400  IF N = 2 THEN P = B(1):Q = B(2): GOTO 4300
4410  IF N = 1 THEN  PRINT  - B(1) / B(0):S = 1
4420  PRINT : IF S = 0 THEN  PRINT "PAS DE SOLUTIONS"
4430  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
4440  IF Z$ = "O" THEN 4000
4450  END
```

c. Exemples commentés

- ***Exemple 1*** :

```
RACINES D'UN POLYNOME DE DEGRE N


N=4

A(0)=1
A(1)=4
A(2)=-164
A(3)=-336
A(4)=5760


PRECISION (1 A 8) :7
P,Q:0,0

6
-8
10
-12

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

• ***Exemple 2*** :

```
 VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
RACINES D'UN POLYNOME DE DEGRE N


N=7

A(0)=2
A(1)=-13
A(2)=-49
A(3)=385
A(4)=-77
A(5)=-1652
A(6)=684
A(7)=720


PRECISION (1 A 8) :9
P,Q:0,0

1
-.5
3
-2
6
-5
4

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

Les racines des deux équations :

$x^4 + 4x^3 - 164x^2 - 336x + 5760 = 0$

$2x^7 - 13x^6 - 49x^5 + 385x^4 - 77x^3 - 1652x^2 + 684x + 720 = 0$

sont trouvées avec une excellente précision.

• ***Exemple 3*** :

```
RACINES D'UN POLYNOME DE DEGRE N


N=6
```

```
A(0)=1
A(1)=-7
A(2)=-21
A(3)=203
A(4)=-140
A(5)=-756
A(6)=720


PRECISION (1 A 8) :8
P,Q:0,0

3
-2
6
4
.999999994
-5

VOULEZ-VOUS REUTILISER LE PROGRAMME?:0
```

Les racines de :

$$x^6 - 7x^5 - 21x^4 + 203x^3 - 140x^2 - 756x + 720 = 0$$

sont obtenues de manière très satisfaisante.

- ***Exemple 4*** :

```
RACINES D'UN POLYNOME DE DEGRE N


N=4

A(0)=-3
A(1)=2
A(2)=-6
A(3)=-3
A(4)=-8


PRECISION (1 A 8) :4
P,Q:0,0


PAS DE SOLUTIONS
VOULEZ-VOUS REUTILISER LE PROGRAMME?:0
```

Le polynôme :

$$-3x^4 + 2x^3 - 6x^2 - 3x - 8$$

ne possédant pas de racines réelles, le programme affiche : "pas de solutions".

- ***Exemple 5*** :

Reprenons l'équation de l'exemple 3, en demandant cette fois une précision de 9.

```
RACINES D'UN POLYNOME DE DEGRE N

N=6

A(0)=1
A(1)=-7
A(2)=-21
A(3)=203
A(4)=-140
A(5)=-756
A(6)=720


PRECISION (1 A 8) :9
P,Q:0,0

3
-2
LE CALCUL NECESSITE TROP D'ITERATIONS...

VOULEZ-VOUS REUTILISER LE PROGRAMME?:0
```

La précision demandée étant trop grande, le programme ne peut trouver toutes les racines.

Cette remarque est très importante, et elle s'applique à tous les programmes où il faut choisir une précision : de façon presque systématique, le programme ne pourra fonctionner normalement lorsqu'une trop grande précision sera demandée.

On peut donc adopter la démarche suivante : effectuer une première recherche avec une précision médiocre (1 ou 2) de manière à localiser les racines, puis reprendre le calcul avec une meilleure précision (7 ou 8).

• ***Exemple 6*** :

Cas de racines multiples

Considérons l'équation suivante :

$$x^6 - 4x^5 - 6x^4 + 32x^3 + x^2 - 60x + 36 = 0$$

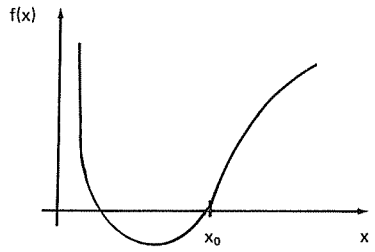
qui admet 1, −2 et 3 comme racines doubles.

```
RACINES D'UN POLYNOME DE DEGRE N

N=6

A(0)=1
A(1)=-4
A(2)=-6
A(3)=32
A(4)=1
A(5)=-60
A(6)=36


PRECISION (1 A 8) :8
P,Q:0,0

3.00000668
.999995369
2.99999332
1.00000463
-2
-2

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

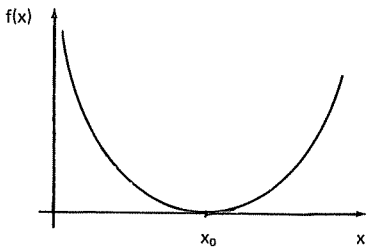
La précision demandée était pourtant 8.

Expliquons ce phénomène de manière intuitive

Lorsqu'une fonction possède une racine multiple x_0, sa représentation graphique au voisinage de x_0 présente une forme "aplatie".



racine simple



racine multiple

Ainsi, lorsqu'on se déplace sur la courbe, une faible variation de y correspond à une variation importante de x.

Plus la multiplicité de la racine est grande, et plus le phénomène s'accentue.

De manière plus rigoureuse, considérons le développement en série de Taylor de la fonction f au voisinage de x_0:

$$f(x_0 + h) = f(x_0) + \frac{h}{1!} f'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots + \frac{h^n}{n!} f^{(n)}(x_0) + h^n \varepsilon(h)$$

Si x_0 est racine de f, alors $f(x_0) = 0$.

Si x_0 est racine double de f, alors $f(x_0) = 0$ et $f'(x_0) = 0$.

Plus généralement, si x_0 est une racine de multiplicité m, toutes les dérivées d'ordre inférieur ou égal à $(m - 1)$ sont nulles en x_0; ainsi, le développement d'ordre m de f se réduit à:

$$f(x_0 + h) = \frac{h^m}{m!} f^{(m)}(x_0) + h^m \varepsilon(h)$$

Supposons par exemple une fonction admettant une racine de multiplicité 4, et prenons $h = 0{,}01$.

$$f(x_0 + 0{,}01) \simeq \frac{10^{-8}}{24} f^{(m)}(x_0) < 4.10^{-10} f^{(m)}(x_0)$$

Si la dérivée $f^{(m)}(x_0)$ n'est pas trop grande, une variation importante de x (0,01) n'a qu'une très faible incidence sur la valeur de la fonction.

• ***Exemple 7***:

Considérons l'équation suivante:

$$x^7 + 7x^6 + 21x^5 + 35x^4 + 35x^3 + 21x^2 + 7x + 1 = 0$$

dont (—1) est l'unique racine, de multiplicité 7.

```
RACINES D'UN POLYNOME DE DEGRE N


N=7

A(0)=1
A(1)=7
A(2)=21
A(3)=35
A(4)=35
A(5)=21
A(6)=7
A(7)=1


PRECISION (1 A 8) :2
P,Q:0,0

-1.07326042

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
RACINES D'UN POLYNOME DE DEGRE N


N=7

A(0)=1
A(1)=7
A(2)=21
A(3)=35
A(4)=35
A(5)=21
A(6)=7
A(7)=1


PRECISION (1 A 8) :2
P,Q:0,1

-.982286899
-1.05281435
-.946642653

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Bien que la précision demandée soit faible, le programme n'affiche qu'une valeur dans le premier cas, et trois valeurs dans le second : ce n'est pas vraiment satisfaisant !

Il convient donc de rester très critique vis-à-vis des résultats fournis par le programme dans le cas de racines multiples et de se contenter d'une valeur approchée.

6. *Méthode des dichotomies*

Laissons maintenant de côté la résolution des équations polynômiales et abordons le problème plus général de la recherche approchée des zéros d'une fonction quelconque.

Commençons par la méthode la plus simple, la méthode dichotomique.

a. Principe

Soit une fonction f donnée ; on se propose de déterminer ses zéros sur un intervalle $[x_{min}, x_{max}]$ fixé.

Supposons que l'on connaisse un intervalle [a,b] sur lequel la fonction change de signe, c'est-à-dire tel que $f(a) \cdot f(b) < 0$.

Si la fonction f est continue, nous savons que cet intervalle contient alors une racine de f.

Nous allons couper cet intervalle en deux en posant :

$$c = \frac{a + b}{2}$$

Deux cas sont alors possibles :

→ si f(c) est du signe de f(a), la racine appartient à l'intervalle [c,b].

→ si f(c) est de signe opposé à f(a), la racine appartient à l'intervalle [a,b].

Nous avons donc encadré la racine par un intervalle deux fois plus petit.

Il suffit de répéter le processus jusqu'à ce que la précision demandée soit atteinte.

b. Programme

La méthode dichotomique présente deux inconvénients majeurs : il faut connaître au départ un intervalle [a,b] sur lequel la fonction change de signe, et une seule racine est trouvée sur cet intervalle.

Le programme s'affranchit de ces deux inconvénients : avec l'intervalle $[x_{min}, x_{max}]$ qui peut être quelconque, on choisit un incrément D.

L'intervalle $[x_{min}, x_{max}]$ est ensuite découpé en sous-intervalles de longueur D. Sur chacun de ces sous-intervalles, le programme examine si la fonction change de signe ; si c'est le cas, la méthode dichotomique y est appliquée.

En choisissant D suffisamment petit, toutes les racines seront obtenues.

L'utilisation du programme ne pose pas de problèmes particuliers : il faut introduire les bornes x_{min} et x_{max} de l'intervalle d'étude, l'incrément de recherche D et la précision désirée sous la forme d'un nombre compris entre 1 et 8.

Il faut également définir la fonction à la ligne 1ØØØØ du programme sous la forme :

```
1ØØØØ  F = 4*SIN(X) + 1 - X
```

```
5000 CLS
5010  PRINT  TAB( 5);"METHODE DES DICHOTOMIES": PRINT :
     PRINT
5020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
5030  PRINT "2- RESOLUTION": PRINT : PRINT
5040  INPUT "VOTRE CHOIX:";E
5050  IF E = 1 THEN  LIST 10000 - 10999: END
5060  INPUT "XMIN=";A
5070  INPUT "XMAX=";B: PRINT
5080  INPUT "INCREMENT:";D: PRINT
5090  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

5100 U = 0
5110  FOR I = A TO B - D STEP D
5120 X = I
5130  GOSUB 10000
5140  IF F = 0 THEN 5280
5150 Z = F
5160 X = I + D: GOSUB 10000
5170  IF X = B AND F = 0 THEN 5280
5180  IF F = 0 THEN 5300
5190  IF Z * F > 0 THEN 5300
5200 P = I:Q = I + D
5210 X = (P + Q) / 2
5220  IF  ABS (Q - P) < E THEN 5280
5230  GOSUB 10000
5240  IF F = 0 THEN 5280
5250  IF F * Z > 0 THEN P = X: GOTO 5210
```

```
5260 Q = X
5270  GOTO 5210
5280  PRINT X
5290 U = 1
5300  NEXT I
5310  IF U = 0 THEN  PRINT "PAS DE SOLUTIONS"
5320  PRINT
5330  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
5340  IF Z$ = "O" THEN 5000
5350  END
10000 F = 4 *  SIN (X) - X + 1
10010  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

```
10000 F=X+4*LOG(X)-5


   METHODE DES DICHOTOMIES


XMIN=1
XMAX=10

INCREMENT:1

PRECISION (1 A 8):8

2.07676138

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

- ***Exemple 2*** :

```
10000 F=4*SIN(X)-X+1


   METHODE DES DICHOTOMIES
```

```
XMIN=-3
XMAX=3

INCREMENT:1

PRECISION (1 A 8):8

-2.21008394
-.342185054
2.70206138

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les racines de l'équation $4 \sin x + 1 - x = 0$ sont trouvées en quelques secondes à 10^{-8} près. (La précision est effectivement de 10^{-8} lorsque la racine n'est pas multiple.)

- ***Exemple 3*** :

```
 10000 F=-160+X*(68+X*(90+X*(4-2*X)))


    METHODE DES DICHOTOMIES


XMIN=-10
XMAX=10

INCREMENT:5

PRECISION (1 A 8):1

-5
.9765625
8.0078125

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DES DICHOTOMIES


XMIN=-10
XMAX=10

INCREMENT:1
```

```
PRECISION (1 A 8):8

-5
-2
1
8

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Avec un incrément de 5, le programme ne trouve que trois racines, alors que les quatre racines sont obtenues avec un incrément de 1.

Lorsqu'on ne connaît ni le nombre ni la position approximative des racines, on peut adopter la démarche suivante :

→ une première recherche avec faible précision et faible incrément permet une localisation rapide de toutes les racines.

→ une deuxième recherche avec précision et incrément important permet alors d'obtenir des valeurs précises.

7. Méthode des itérations

a. Principe

La forme de l'équation à résoudre n'est plus $f(x) = 0$ mais $x = \varphi(x)$.

Cette hypothèse n'est pas restrictive ; par exemple l'équation :

$$4 \sin x - x + 1 = 0$$

peut se mettre sous la forme :

$$x = 4 \sin x + 1$$

La méthode consiste, à partir d'un point x_0 qu'il faut choisir le plus proche possible de la racine, à calculer successivement :

$x_1 = \varphi(x_0)$
$x_2 = \varphi(x_1)$
.....
$x_{n+1} = \varphi(x_n)$

Lorsque le processus converge, la différence $|x_{n+1} - x_n|$ finit par devenir inférieure à l'erreur maximale fixée à l'avance : x_n est alors une solution approchée de l'équation.

Malheureusement, la méthode présente le grave défaut de ne converger que pour les fonctions φ lipschitziennes de rapport K avec $K \leqslant 1$.

Ceci signifie, si l'on prend deux points quelconques a et b, que la fonction φ doit vérifier:

$$|\varphi(a) - \varphi(b)| \leqslant K|a - b|$$

Si la fonction φ est dérivable, il revient au même de dire que φ' doit être strictement majorée par 1.

b. Programme

La fonction φ doit être définie à la ligne 1ØØØØ du programme sous la forme:

```
1ØØØØ F = COS(X)
```

Il faut ensuite fournir au programme le point de départ x_0 et la précision désirée, toujours sous la forme d'un nombre compris entre 1 et 8.

Si le processus ne converge pas ou si la précision demandée est trop grande, le programme s'arrête en affichant:

```
LE CALCUL NECESSITE TROP D'ITERATIONS
```

Nous retrouverons cette particularité dans les derniers programmes de ce chapitre, ainsi que dans ceux du chapitre "Recherche d'extrémums".

```
6000 CLS
6010  PRINT  TAB( 5);"METHODE DES ITERATIONS": PRINT : PRINT

6020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
6030  PRINT "2- RESOLUTION": PRINT : PRINT
6040  INPUT "VOTRE CHOIX:";E
6050  IF E = 1 THEN  LIST 10000 - 10999: END
6060  INPUT "XO=";X: PRINT
6070  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

6080  FOR I = 1 TO 200
6090  GOSUB 10000
6100  IF  ABS (F - X) < E THEN 6140
6110 X = F
6120  NEXT I
6130  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS": GOTO
    6150
```

```
6140  PRINT "X=";X
6150  PRINT
6160  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
6170  IF Z$ = "O" THEN 6000
6180  END
10000 F =  COS (X)
10010  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

On se propose de résoudre l'équation $x = \cos x$.

```
 10000 F=COS(X)


    METHODE DES ITERATIONS


XO=1

PRECISION (1 A 8):8

X=.739085128

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DES ITERATIONS


XO=1

PRECISION (1 A 8):9

LE CALCUL NECESSITE TROP D'ITERATIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Dans le premier cas, la racine est trouvée avec une bonne précision (à 10^{-8} près).

Dans le deuxième cas, le fait de demander une précision trop forte empêche le programme de fonctionner normalement.

• ***Exemple 2*** :

Soit à résoudre l'équation :

$x = 4 \sin x + 1$

```
10000 F=4*SIN(X)+1


   METHODE DES ITERATIONS


XO=1

PRECISION (1 A 8):1

X=-2.20428408

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
   METHODE DES ITERATIONS

XO=1

PRECISION (1 A 8):2

LE CALCUL NECESSITE TROP D'ITERATIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Une première recherche à faible précision (1) fournit une valeur approchée voisine de $-2{,}2$.

Dans le but d'obtenir un meilleur résultat, le programme est relancé avec une précision légèrement supérieure (2). Le programme ne trouve alors plus la racine.

Ce comportement est dû au fait que la fonction ne vérifie pas le critère de convergence ; en effet $\varphi'(x) = 4 \cos x$ n'est pas majorée par 1 mais par 4.

La méthode est donc inapplicable.

• ***Exemple 3*** :

Nous sommes ici encore dans un cas de non-application de la méthode : la fonction $\varphi(x) = \dfrac{x^2 + 1}{2}$ admet pour dérivée $\varphi'(x) = x$. Or la racine de l'équation $x = \varphi(x)$ est $x_0 = 1$.

Au voisinage de x_0, la dérivée de φ tend vers 1, donc vers la valeur limite assurant la convergence.

```
 10000 F=(X*X+1)/2


    METHODE DES ITERATIONS


XO=1

PRECISION (1 A 8):4

X=1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DES ITERATIONS


XO=O

PRECISION (1 A 8):4

X=.985887497

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DES ITERATIONS


XO=O

PRECISION (1 A 8):5

LE CALCUL NECESSITE TROP D'ITERATIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Ici le processus converge, mais le programme s'arrête parce que la convergence est trop lente.

d. Conclusion

L'application de la méthode doit se faire avec prudence, après avoir vérifié la condition de Lipschitz.

Le seul avantage de cette méthode est d'être plus rapide que la méthode des dichotomies.

8. Méthode de Newton

a. Principe

Nous considérons de nouveau une équation du type :

$$f(x) = 0$$

La méthode de Newton, tout comme la méthode des itérations, consiste à calculer une suite de valeurs (x_n) convergeant vers la racine α de l'équation.

Partant d'un point x_0, on calcule successivement :

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

.....

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

Le processus s'arrête lorsque la différence $|x_n - x_{n-1}|$ devient inférieure à l'erreur maximale fixée : x_n est alors une valeur approchée de α.

La méthode est applicable à la double condition suivante : la fonction doit être dérivable et la dérivée ne doit pas s'annuler au voisinage de α.

En particulier, la méthode ne peut être utilisée pour la détermination de racines multiples.

b. Programme

La fonction doit être définie à la ligne 1ØØØØ du programme sous la forme :

```
1ØØØØ F = 4★SIN(X)−X+1
```

Il faut ensuite introduire le point de départ x_0 et la précision sous la forme d'un nombre compris entre 1 et 8.

La dérivée de la fonction est calculée de manière approchée par le programme grâce à la formule :

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} \text{ avec } h = 0{,}001.$$

```
7000 CLS
7010  PRINT  TAB( 5);"METHODE DE NEWTON": PRINT : PRINT

7020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
7030  PRINT "2- RESOLUTION": PRINT : PRINT
7040  INPUT "VOTRE CHOIX:";E
7050  IF E = 1 THEN  LIST 10000 - 10999: END
7060  INPUT "X0=";X: PRINT
7070  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

7080  FOR I = 1 TO 200
7090 X = X + .001: GOSUB 10000:Z = F
7100 X = X - .002: GOSUB 10000:Z = (Z - F) / .002
7110 X = X + .001: GOSUB 10000
7120  IF Z = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    7180
7130 X = X - F / Z
7140  IF  ABS (F / Z) < E THEN 7170
7150  NEXT I
7160  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS": GOTO
    7180
7170  PRINT "X=";X
7180  PRINT
7190  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
7200  IF Z$ = "O" THEN 7000
7210  END
10000 F = 4 *  SIN (X) - X + 1
10010  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

```
10000 F=4*SIN(X)-X+1
```

```
   METHODE DE NEWTON
```

```
XO=-3

PRECISION (1 A 8):8

X=-2.21008394

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DE NEWTON


XO=O

PRECISION (1 A 8):8

X=-.342185053

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

Selon la valeur initiale x_0, les racines détectées ne sont pas les mêmes.

La précision des valeurs obtenues est de 10^{-8}; on pourra comparer ces valeurs aux résultats fournis par la méthode des dichotomies.

- ***Exemple 2*** :

```
 10000 F=X+4*LOG(X)-5


    METHODE DE NEWTON


XO=1

PRECISION (1 A 8):8

X=2.07676139

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La racine obtenue est ici encore égale à 10^{-8} près à la valeur fournie par la méthode des dichotomies.

• ***Exemple 3*** :

```
10000 F=X*X

   METHODE DE NEWTON

XO=0

PRECISION (1 A 8):1

LA METHODE EST INADAPTEE

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La fonction $f(x) = x^2$ admet 0 comme racine double ; le programme indique donc que la méthode de Newton est inapplicable.

d. Quelle méthode utiliser?

La condition d'application de la méthode de Newton est nettement moins contraignante que celle de la méthode des itérations : les fonctions usuelles sont bien plus souvent dérivables que lipschitziennes de rapport inférieur à 1.

La méthode de Newton est très rapide ; en général, quelques itérations suffisent pour obtenir la racine avec la précision maximale.

Cependant notre préférence va à la méthode dichotomique car cette dernière permet la recherche simultanée de plusieurs racines, et possède une condition d'application très large (il suffit que la fonction soit continue) ; de plus le gain de temps apporté par la méthode de Newton n'est pas vraiment sensible dans la majorité des cas, le BASIC étant suffisamment rapide dans ce genre d'applications.

9. Résolution des systèmes de deux équations à deux inconnues

Le problème est le suivant ; il faut déterminer une solution (α, β) du système :

$$\begin{cases} f(x,y) = 0 \\ g(x,y) = 0 \end{cases}$$

a. Principe

Nous allons utiliser ici une généralisation de la méthode de Newton, appelée méthode de linéarisation.

Nous allons donc former, à partir d'un point initial (x_0,y_0), deux suites (x_n) et (y_n) convergeant vers (α,β).

La relation de définition des suites est la relation de récurrence suivante :

$$x_{k+1} = x_k - \frac{f.g'_y - g.f'_y}{\Delta}$$

$$y_{k+1} = y_k - \frac{g.f'_x - f.g'_x}{\Delta}$$

où :

$$\Delta = f'_x.g'_y - f'_y.g'_x$$

Les notations adoptées pour les dérivées partielles sont les suivantes

$$f'_x = \frac{\partial f}{\partial x}$$

$$f'_y = \frac{\partial f}{\partial y}$$

$$g'_x = \frac{\partial g}{\partial x}$$

$$g'_y = \frac{\partial g}{\partial y}$$

Le programme s'arrête lorsque la quantité $|x_n - x_{n-1}| + |y_n - y_{n-1}|$ est inférieure à l'erreur maximale fixée ; (x_n,y_n) constitue alors une solution approchée du système.

b. Programme

Les fonctions f et g doivent être programmées à partir de la ligne 11ØØØ sous la forme :

```
11ØØØ F = X*X-Y*Y+X+1
11Ø1Ø G = 2*X*Y+Y
```

Il faut ensuite introduire le point initial (x_0,y_0) ainsi que la précision désirée.

Le programme calcule les valeurs approchées des différentes dérivées partielles grâce aux formules suivantes:

$$f'_x(x,y) = \frac{f(x+h,y) - f(x-h,y)}{2h}$$

$$f'_y(x,y) = \frac{f(x,y+h) - f(x,y-h)}{2h}$$

$$g'_x(x,y) = \frac{g(x+h,y) - g(x-h,y)}{2h}$$

$$g'_y(x,y) = \frac{g(x,y+h) - g(x,y-h)}{2h}$$

```
8000 CLS
8010  PRINT "RESOLUTION D'UN SYSTEME NON LINEAIRE": PRINT
    : PRINT
8020  PRINT "1- PROGRAMMATION DU SYSTEME": PRINT
8030  PRINT "2- RESOLUTION": PRINT : PRINT
8040  INPUT "VOTRE CHOIX:";E
8050  IF E = 1 THEN  LIST 11000 - 11999: END
8060  INPUT "X0=";X
8070  INPUT "Y0=";Y
8080  INPUT "PRECISION (1 A 7):";E:E = 10 ^ ( - E)
8090 J = 0:H = .001: PRINT
8100 X = X + H: GOSUB 11000:A = F:B = G
8110 X = X - 2 * H: GOSUB 11000:A = (A - F) / 2 / H:B =
    (B - G) / 2 / H
8120 X = X + H:Y = Y + H
8130  GOSUB 11000:C = F:D = G
8140 Y = Y - 2 * H: GOSUB 11000:C = (C - F) / 2 / H:D =
    (D - G) / 2 / H
8150 Y = Y + H: GOSUB 11000
8160 T = A * D - B * C
8170  IF T = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    8250
8180 P = (F * D - G * C) / T
8190 Q = (G * A - F * B) / T
8200 X = X - P:Y = Y - Q:J = J + 1
8210  IF J = 100 THEN  PRINT "LE CALCUL NECESSITE TROP
    D'ITERATIONS": GOTO 8250
8220  IF ( ABS (P) +  ABS (Q)) > E THEN 8100
8230  PRINT "SOLUTION APPROCHEE : "
8240  PRINT  TAB( 5);"X=";X: PRINT  TAB( 5);"Y=";Y
8250  PRINT
```

```
8260  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME:";Z$
8270  IF Z$ = "O" THEN 8000
8280  END
11000 F = X * X - Y * Y + X + 1
11010 G = 2 * X * Y + Y
11020  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

```
 11000 F=X*X+Y*Y+2*(X+Y)-23

 11010 G=X*X+Y*Y+X*Y-19


RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=2
YO=2
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
    X=3
    Y=2

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=1
YO=-1
PRECISION (1 A 7):1

SOLUTION APPROCHEE :
    X=2.0000908
    Y=-5.00009144

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION D'UN SYSTEME NON LINEAIRE
```

```
XO=1
YO=-1
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
    X=2
    Y=-5

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=0
YO=0
PRECISION (1 A 7):1

LA METHODE EST INADAPTEE

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Selon le point de départ (x_0,y_0), le programme fournit la racine (3,2) ou la racine (2,—5).

Si l'on choisit le point (0,0), le programme indique que la méthode est inadaptée : ceci est dû ici au fait que toutes les dérivées sont nulles à l'origine.

- ***Exemple 2*** :

```
11000 F=8*X-Y-3

11010 G=X*X+4*X*Y-5*Y*Y+12*X+92


RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=0
YO=0
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
    X=-.163763066
    Y=-4.31010453
```

```
VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=1
YO=1
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
     X=1
     Y=5

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Les deux solutions du système:

$$\begin{cases} 8x - y - 3 = 0 \\ x^2 + 4xy - 5y^2 + 12x + 92 = 0 \end{cases}$$

sont ici aussi trouvées de façon exacte.

Le premier couple est:

$$\left(\frac{-47}{287}, \frac{-1237}{287}\right).$$

Cette méthode permet également de déterminer les zéros complexes d'une fonction complexe.

Cherchons par exemple les racines de l'équation:

$$z^2 + z + 1 = 0$$

Posons:

$$z = x + iy$$

L'équation devient en remplaçant:

$$(x + iy)^2 + (x + iy) + 1 = 0$$

soit:

$$(x^2 - y^2 + x + 1) + i\,(2xy + y) = 0$$

Cette équation est vérifiée si la partie réelle et la partie imaginaire sont nulles ; x et y sont donc solutions du système :

$$\begin{cases} x^2 - y^2 + x + 1 = 0 \\ 2xy + y = 0 \end{cases}$$

```
11000 F=X*X-Y*Y+X+1

11010 G=2*X*Y+Y
```

```
RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=1
YO=1
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
    X=-.5
    Y=.866025404

VOULEZ-VOUS REUTILISER LE PROGRAMME:O
RESOLUTION D'UN SYSTEME NON LINEAIRE


XO=-1
YO=-1
PRECISION (1 A 7):7

SOLUTION APPROCHEE :
    X=-.5
    Y=-.866025404

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Le programme trouve effectivement les deux racines de $z^2 + z + 1 = 0$, à savoir :

$$j = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$$

$$j^2 = \bar{j} = -\frac{1}{2} - \frac{\sqrt{3}}{2} i$$

3 Recherche d'extrémums

1. Introduction

Nous allons nous intéresser dans ce chapitre à diverses méthodes permettant la détermination de maximums et de minimums de fonctions.

Nous diviserons notre étude en deux parties : l'une sera consacrée aux fonctions d'une variable, l'autre aux fonctions de deux variables.

Comme pour la recherche de zéros de fonctions, deux types de méthodes existent :

→ les méthodes de diminution et d'augmentation systématiques, qui sont le pendant de la méthode des dichotomies,

→ les méthodes faisant appel aux dérivées, plus élaborées, qui sont le pendant de la méthode de Newton.

Pour utiliser les sept programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu :

```
100 CLS
110  PRINT  TAB( 5);"OPTIMISATION DE FONCTIONS": PRINT
    : PRINT
120  PRINT : PRINT "1- F(X):METHODE DE NEWTON"
130  PRINT : PRINT "2- F(X):DIMINUTION SYSTEMATIQUE"
140  PRINT : PRINT "3- F(X):AUGMENTATION SYSTEMATIQUE"
150  PRINT : PRINT "4- F(X,Y):METHODE DE RELAXATION"
160  PRINT : PRINT "5- F(X,Y):METHODE DU GRADIENT"
170  PRINT : PRINT "6- F(X,Y):DIMINUTION SYSTEMATIQUE"
180  PRINT : PRINT "7- F(X,Y):AUGMENTATION SYSTEMATIQUE
    "
190  PRINT : PRINT "8- FIN"
200  PRINT : INPUT "VOTRE CHOIX:";E
210  ON E GOTO 1000,2000,6000,3000,4000,5000,7000,8000
220  GOTO 100
8000  END
```

L'ensemble nécessite environ 6 Koctets de mémoire.

2. *Méthode de Newton*

a. Principe

Soit f une fonction d'une variable.

Nous savons que f admet un extrémum en α si la propriété $f'(\alpha) = 0$ est vérifiée.

Il suffit donc de déterminer un zéro de la dérivée de f: cette recherche est effectuée par la méthode de Newton.

Condition d'application

La fonction f doit être deux fois dérivable en α, et de plus la valeur de sa dérivée seconde en α ne doit pas être nulle. Dans le cas contraire, α serait racine double de f', et nous avons vu que la méthode de Newton était inapplicable dans le cas de racines multiples.

b. Programme

La fonction étudiée doit être programmée à la ligne 1ØØØØ sous la forme:

```
1ØØØØ F = X*SIN(X)-COS(X)/X
```

La recherche débute à partir d'un point x_0 que l'on introduit dans le programme et qui doit se rapprocher le plus possible de l'extrémum cherché.

Il faut également choisir la précision sous forme d'un nombre compris entre 1 et 8.

Le calcul de la dérivée seconde est effectué grâce à la formule:

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

Lorsque le programme détecte un extrémum, il précise s'il s'agit d'un maximum ou d'un minimum:

→ si $f''(\alpha) < 0$, il s'agit d'un maximum.

→ si $f''(\alpha) > 0$, il s'agit d'un minimum.

```
1000 CLS
1010  PRINT "F(X):METHODE DE NEWTON": PRINT : PRINT
1020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
1030  PRINT "2- RECHERCHE DES EXTREMA": PRINT : PRINT
1040  INPUT "VOTRE CHOIX:";E: PRINT
1050  IF E = 1 THEN  LIST 10000 - 10999: END
1060  INPUT "XO=";X
1070  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

1080 H = .001
1090  FOR K = 1 TO 100
1100  GOSUB 10000:B =  - 2 * F
1110 X = X + H: GOSUB 10000:A = F:B = B + F
1120 X = X - 2 * H: GOSUB 10000:A = A - F:B = B + F
1130 A = A / 2 / H:B = B / H / H:X = X + H
1140  IF B = 0 THEN  PRINT "LA METHODE EST INADAPTEE":
      GOTO 1230
1150 X = X - A / B
1160  IF  ABS (A / B) > E THEN 1210
1170  GOSUB 10000
1180  PRINT : PRINT "X=";X: PRINT
1190  IF B < 0 THEN  PRINT "F(X)=";F;"  (MAXIMUM)": GOTO
     1230
1200  IF B > 0 THEN  PRINT "F(X)=";F;"  (MINIMUM)": GOTO
     1230
1210  NEXT K
1220  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
1230  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
     E?:";Z$
```

```
1240  IF Z$ = "O" THEN 1000
1250  END
10000 F = X * X * (45 + X * (2 - X))
10010  RETURN
```

c. Exemples commentés

• ***Exemple 1*** :

Nous avons cherché ici un extrémum de la fonction $x \sin x - \frac{\cos x}{x}$ au voisinage du point 1.

```
 10000 F=X*SIN(X)-COS(X)/X


F(X):METHODE DE NEWTON


XO=1
PRECISION (1 A 8):1


X=2.12962472

F(X)=2.0546204   (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):METHODE DE NEWTON


XO=1
PRECISION (1 A 8):6


X=2.12962307

F(X)=2.0546204   (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):METHODE DE NEWTON
```

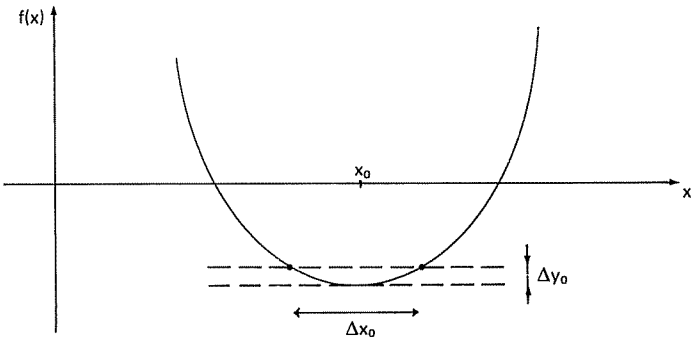
```
XO=1
PRECISION (1 A 8):8

LE CALCUL NECESSITE TROP D'ITERATIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Un premier calcul effectué avec une faible précision permet la localisation de la racine ; les deux suivants ont pour but d'améliorer cette précision.

Nous remarquons ici encore que le fait de demander une précision trop forte peut empêcher le programme de fonctionner normalement.

D'autre part, à l'issue de la première recherche, la valeur de α n'est connue qu'avec cinq décimales exactes ($\alpha = 2{,}12962$) alors que $f(\alpha)$ est connue à 10^{-7} près.

Ceci rappelle la mauvaise précision obtenue lors de la recherche de zéros multiples de fonctions et l'explication est identique : au voisinage d'un extrémum, la représentation graphique de la fonction s'aplatit :



et à une petite variation de y_0 correspond une grande variation de x_0.

L'extrémum de f sera donc en général connu avec une bonne précision, alors que la valeur α correspondante sera entachée d'une erreur plus importante : ceci sera valable pour tous les programmes de recherche d'extrémums.

- ***Exemple 2*** :

```
 10000 F=X*X*(90+X*(4-2*X))

F(X):METHODE DE NEWTON


XO=1
PRECISION (1 A 8):8
```

```
X=-2.22222651E-08

F(X)=4.4444616E-14  (MINIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):METHODE DE NEWTON


XO=10
PRECISION (1 A 8):8


X=5.55234276

F(X)=1558.45366  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):METHODE DE NEWTON


XO=-10
PRECISION (1 A 8):8


X=-4.05234305

F(X)=672.421342  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On peut vérifier que le polynôme $-2x^4 + 4x^3 + 90x^2$ est effectivement :

→ minimal pour :

$x_1 = 0$

→ maximal pour :

$$x_2 = \frac{3 + 3\sqrt{41}}{4} = 5{,}5523432$$

et pour :

$$x_3 = \frac{3 - 3\sqrt{41}}{4} = -4{,}0523432$$

les valeurs correspondantes de f étant :

$$f(x_1) = 0$$

$$f(x_2) = \frac{17\,847 + 1\,107\sqrt{41}}{16} = 1558{,}45366$$

$$f(x_3) = \frac{17\,847 - 1\,107\sqrt{41}}{16} = 672{,}421342$$

Ici encore, les maximums de f sont obtenus avec la précision maximale alors que les valeurs des x_i correspondants ne sont pas exactes qu'à 10^{-6} près.

3. Diminution et augmentation systématiques de la fonction

a. Introduction

La méthode de Newton présente deux inconvénients :

→ elle ne s'applique ni aux fonctions dont la dérivée seconde s'annule au voisinage de l'extrémum cherché, ni aux fonctions non dérivables,

→ il peut être gênant d'obtenir tous les extrémums alors qu'on ne cherche que les minimums (ou les maximums).

Nous allons donc examiner une méthode dont le champ d'application est plus vaste et qui ne cherche qu'une sorte d'extrémums.

Nous aurons deux programmes, l'un pour la recherche des minimums, l'autre pour la recherche des maximums.

b. Principe

Nous n'expliquerons que la recherche des minimums.

On choisit au départ un point x_0, le plus proche possible du point cherché, et un pas initial de découpage h.

On calcule la valeur de la fonction aux points $x_0 + h$ et $x_0 - h$; si en l'un de ces deux points la fonction est plus petite qu'en x_0, on remplace x_0 par ce point, et on recommence. Par exemple, si $f(x_0 + h) < f(x_0)$, on examine $f(x_0 + 2h)$.

Lorsqu'on se rapproche suffisamment du minimum, les deux relations :

$$f(x + h) > f(x)$$
$$f(x - h) > f(x)$$

sont vérifiées: le pas de découpage est alors divisé par 2, et l'on poursuit l'encadrement.

Le calcul se termine lorsque h est inférieur à une valeur fixée à l'avance, caractérisant la précision du résultat.

c. Programmes

La fonction doit ici encore être définie à la ligne 1ØØØØ du programme sous la forme:

```
1ØØØØ F = X*X*(90+X*(4-2*X))
```

Il faut également fournir au programme le point de départ x_0, qui doit se situer le plus près possible de l'extrémum, le pas initial h (que l'on doit choisir suffisamment petit lors de la recherche d'extrémums relatifs) et la précision désirée, toujours sous la forme d'un nombre compris entre 1 et 8.

Programme de recherche des minimums:

```
2000 CLS
2010  PRINT "F(X):DIMINUTION SYSTEMATIQUE": PRINT : PRINT

2020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
2030  PRINT "2- RECHERCHE DES MINIMA": PRINT : PRINT
2040  INPUT "VOTRE CHOIX:";E: PRINT
2050  IF E = 1 THEN  LIST 10000 - 10999: END
2060  INPUT "XO=";X
2070  INPUT "H=";H
2080  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

2090  FOR K = 1 TO 200
2100  GOSUB 10000:A = F
2110 X = X - H: GOSUB 10000
2120  IF A > F THEN 2180
2130 X = X + 2 * H: GOSUB 10000
2140  IF A > F THEN 2180
2150 X = X - H
2160  IF H < E THEN  PRINT : PRINT "X=";X: PRINT : PRINT
    "F(X)=";A: GOTO 2200
2170 H = H / 2
2180  NEXT K
2190  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
2200  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
```

```
2210  IF Z$ = "O" THEN 2000
2220  END
10000 F = X * X * (45 + X * (2 - X))
10010  RETURN
```

Programme de recherche des maximums:

```
6000 CLS
6010  PRINT "F(X):AUGMENTATION SYSTEMATIQUE": PRINT : PRINT

6020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
6030  PRINT "2- RECHERCHE DES MAXIMA": PRINT : PRINT
6040  INPUT "VOTRE CHOIX:";E: PRINT
6050  IF E = 1 THEN  LIST 10000 - 10999: END
6060  INPUT "XO=";X
6070  INPUT "H=";H
6080  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

6090  FOR K = 1 TO 200
6100  GOSUB 10000:A = F
6110 X = X - H: GOSUB 10000
6120  IF A < F THEN 6180
6130 X = X + H + H: GOSUB 10000
6140  IF A < F THEN 6180
6150 X = X - H
6160  IF H < E THEN  PRINT : PRINT "X=";X: PRINT : PRINT
    "F(X)=";A: GOTO 6200
6170 H = H / 2
6180  NEXT K
6190  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
6200  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
6210  IF Z$ = "O" THEN 6000
6220  END
10000 F = X * X * (45 + X * (2 - X))
10010  RETURN
```

d. Exemples commentés

- ***Exemple 1*** :

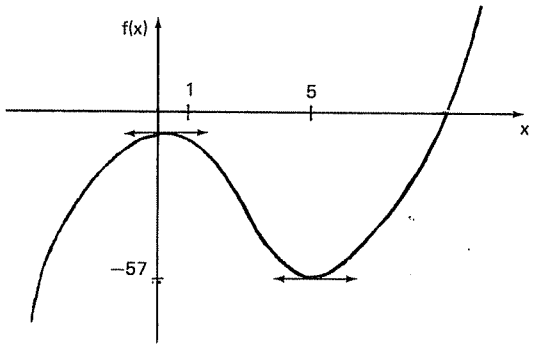
Le minimum du polynôme $x^3 - 8x^2 + 5x - 7$ est obtenu en $x = 5$.

```
 10000 F=-7+X*(5+X*(-8+X))


F(X):DIMINUTION SYSTEMATIQUE


XO=1
H=1
PRECISION (1 A 8):1


X=5

F(X)=-57

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):DIMINUTION SYSTEMATIQUE


XO=1
H=.22
PRECISION (1 A 8):7


X=5.00001465

F(X)=-57

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):DIMINUTION SYSTEMATIQUE


XO=1
H=5
PRECISION (1 A 8):1

LE CALCUL NECESSITE TROP D'ITERATIONS

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Premier cas: La valeur exacte du minimum est trouvée car $x_0 + 4h = 5$, et le programme est tombé juste dessus.

Deuxième cas: Avec une autre valeur de h, le programme ne trouve cette fois qu'une valeur approchée.

Troisième cas: Considérons la représentation graphique du polynôme:



Avec h = 5, les deux premiers points examinés par le programme sont −4 et 6; on remarque grâce à la courbe que f(−4) < f(1).

Le programme va donc ensuite examiner les points −9, −14, −19, etc... et le processus diverge car $\lim_{x \to -\infty} f(x) = -\infty$.

Il convient donc, lors de la recherche de minimums relatifs, de choisir un pas initial suffisamment petit pour éviter ce phénomène. Dans l'exemple, la valeur h = 1 fournit le résultat attendu.

Recherchons maintenant la valeur du maximum:

```
10000 F=-7+X*(5+X*(-8+X))

F(X):AUGMENTATION SYSTEMATIQUE

XO=0
H=1
PRECISION (1 A 8):8

X=.333328247

F(X)=-6.18518519

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le programme fournit le résultat $\alpha' = 0{,}333328247$ alors que la valeur exacte est $\alpha = \frac{1}{3}$; la précision est donc de 10^{-5}. La valeur de $f(\alpha)$ est par contre exacte : $f(\alpha) = -\frac{167}{27}$

- ***Exemple 2*** :

```
10000 F=X*X*(90+X*(4-2*X))

F(X):AUGMENTATION SYSTEMATIQUE

XO=2
H=1
PRECISION (1 A 8):8

X=5.55236816

F(X)=1558.45366

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X):AUGMENTATION SYSTEMATIQUE

XO=-2
H=1
PRECISION (1 A 8):8

X=-4.05233765

F(X)=672.421342

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les deux maximums du polynôme $-2x^4 + 4x^3 + 90x^2$ sont trouvés à 2.10^{-5} près ; on pourra comparer ces résultats aux valeurs obtenues par la méthode de Newton.

4. *Méthode de relaxation*

Cette méthode, ainsi que la méthode du gradient, sont deux extensions de la méthode de Newton, et permettent la recherche des extrémums locaux des fonctions de deux variables f(x,y).

Condition d'application :

Les dérivées partielles d'ordre 1 et 2 de f doivent exister, et les dérivées secondes ne doivent pas s'annuler au voisinage de l'extrémum.

a. Principe

On notera :

$f'_x = \dfrac{\partial f}{\partial x}$ dérivée partielle de f par rapport à x

$f'_y = \dfrac{\partial f}{\partial y}$ dérivée partielle de f par rapport à y

$f''_x = \dfrac{\partial^2 f}{\partial x^2}$ dérivée seconde de f par rapport à x

$f''_y = \dfrac{\partial^2 f}{\partial y^2}$ dérivée seconde de f par rapport à y

La méthode consiste, à partir d'un point initial (x_0,y_0) qui doit se situer le plus près possible de l'extrémum cherché, à calculer les suites (x_n) et (y_n) suivantes

$$\begin{cases} x_1 = x_0 - \dfrac{f'_x(x_0,y_0)}{f''_x(x_0,y_0)} \\ y_1 = y_0 \end{cases}$$

$$\begin{cases} x_2 = x_1 \\ y_2 = y_1 - \dfrac{f'_y(x_1,y_1)}{f''_y(x_1,y_1)} \end{cases}$$

.....

$$\begin{cases} x_{n+1} = x_n - \dfrac{f'_x(x_n,y_n)}{f''_x(x_n,y_n)} \\ y_{n+1} = y_n \end{cases}$$

$$\begin{cases} x_{n+2} = x_{n+1} \\ y_{n+2} = y_{n+1} - \dfrac{f'_y(x_{n+1},y_{n+1})}{f''_y(x_{n+1},y_{n+1})} \end{cases}$$

Lorsque ces deux suites convergent, c'est nécessairement vers un extrémum de la fonction f.

Cette méthode revient à optimiser la fonction d'abord en x, à y fixé, ensuite en y, à x fixé, et à réitérer le processus jusqu'à approcher suffisamment l'extrémum cherché.

b. Programme

La fonction doit être définie à la ligne 11ØØØ du programme sous la forme:

```
11ØØØ F = X*X+(Y-1)*Y
```

Le programme demande ensuite le point de départ (x_0,y_0) ainsi que la précision souhaitée.

Les dérivées partielles de f sont calculées grâce aux formules approchées:

$$f'_x(x,y) = \frac{f(x+h,y) - f(x-h,y)}{2h}$$

$$f'_y(x,y) = \frac{f(x,y+h) - f(x,y-h)}{2h}$$

$$f''_x(x,y) = \frac{f(x+h,y) - 2f(x,y) + f(x-h,y)}{h^2}$$

$$f''_y(x,y) = \frac{f(x,y+h) - 2f(x,y) + f(x,y-h)}{h^2}$$

Lorsqu'il trouve un extrémum, le programme précise s'il s'agit d'un maximum ou d'un minimum.

```
3000 CLS
3010  PRINT "F(X,Y):METHODE DE RELAXATION": PRINT : PRINT

3020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
3030  PRINT "2- RECHERCHE DES EXTREMA": PRINT : PRINT
3040  INPUT "VOTRE CHOIX:";E: PRINT
3050  IF E = 1 THEN  LIST 11000 - 11999: END
3060  INPUT "XO=";X
3070  INPUT "YO=";Y
3080  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

3090 H = .001
3100  FOR K = 1 TO 100
3110  GOSUB 11000:B =  - 2 * F
3120 X = X + H: GOSUB 11000:A = F:B = B + F
3130 X = X - 2 * H: GOSUB 11000:A = A - F:B = B + F
```

```
3140 A = A / 2 / H:B = B / H / H:X = X + H
3150  IF B = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    3320
3160 P = A / B:X = X - P
3170  GOSUB 11000:D =  - 2 * F
3180 Y = Y + H: GOSUB 11000:C = F:D = D + F
3190 Y = Y - 2 * H: GOSUB 11000:C = C - F:D = D + F
3200 C = C / 2 / H:D = D / H / H:Y = Y + H
3210  IF D = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    3320
3220 Q = C / D:Y = Y - Q
3230  IF ( ABS (P) +  ABS (Q)) > E THEN 3300
3240 X = X - .01:Y = Y - .01: GOSUB 11000:M = F
3250 X = X + .01:Y = Y + .01: GOSUB 11000
3260  PRINT : PRINT "X=";X: PRINT "Y=";Y: PRINT
3270  IF F > M THEN  PRINT "F(X,Y)=";F;"  (MAXIMUM)"
3280  IF F < M THEN  PRINT "F(X,Y)=";F;"  (MINIMUM)"
3290  GOTO 3320
3300  NEXT K
3310  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
3320  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
3330  IF Z$ = "O" THEN 3000
3340  END
11000 F = X * X * (45 + X * (2 - X)) + Y * Y * (45 + Y *
    (2 - Y))
11010  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

```
 11000 F=(X*X-4*X-1)/(X*Y-Y*Y+16-2*ABS(X))+(Y/(2*X-5))


F(X,Y):METHODE DE RELAXATION


XO=1
YO=-2
PRECISION (1 A 8):1
```

```
X=1.10504354
Y=-1.97761219

F(X,Y)=.163059598  (MINIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DE RELAXATION


XO=1
YO=-2
PRECISION (1 A 8):7


X=1.11488821
Y=-1.97135164

F(X,Y)=.163028363  (MINIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Un minimum relatif de la fonction :

$$\frac{x^2 - 4x - 1}{xy - y^2 + 16 - 2|x|} + \frac{y}{2x - 5}$$

est trouvé vers (1,11 ; −1,97).

Ici encore la précision est meilleure pour la valeur de la fonction que pour les valeurs de x et y puisque, avec la précision minimale, quatre chiffres significatifs sont obtenus pour f(x,y), contre deux pour x et y.

- ***Exemple 2*** :

```
11000 F=X*X*(90+X*(4-2*X))+Y*Y*(90+Y*(4-2*Y))


F(X,Y):METHODE DE RELAXATION


XO=1
YO=1
PRECISION (1 A 8):5
```

```
X=-2.22225935E-08
Y=-2.22225935E-08

F(X,Y)=8.88918594E-14  (MINIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DE RELAXATION


XO=10
YO=10
PRECISION (1 A 8):5
X=5.55234305
Y=5.55234323

F(X,Y)=3116.90732  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DE RELAXATION


XO=-10
YO=-10
PRECISION (1 A 8):5
X=-4.05234293
Y=-4.05234288

F(X,Y)=1344.84268  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DE RELAXATION


XO=10
YO=-10
PRECISION (1 A 8):5


X=5.55234354
Y=-4.05234244

F(X,Y)=2230.875  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La fonction $f(x,y) = (-2x^4 + 4x^3 + 90x^2) + (-2y^4 + 4y^3 + 90y^2)$ possède la particularité de s'écrire également :

$$f(x,y) = g(x) + h(y)$$

La recherche des extrémums d'une telle fonction peut être effectuée en calculant indépendamment les extrémums de g et ceux de h.

Dans l'exemple, nous avons en plus la relation $g = h$, et nous connaissons les extrémums de cette fonction, déjà étudiée dans le second paragraphe : ils sont obtenus pour les valeurs :

$\alpha = 5{,}5523432$
$\beta = -4{,}0523432$
$\gamma = 0$

La fonction f aura donc cinq extrémums locaux, aux points :

$(0,0)$; (α,α) ; (α,β) ; (β,α) ; (β,β)

Les couples $(0,\alpha)$; $(\alpha,0)$; $(\beta,0)$; $(0,\beta)$ ne conviennent pas car les extrémums obtenus en 0 et en α ou β sont de natures différentes ; 0 correspond à un minimum de la fonction, α et β à des maximums.

Selon le point de départ, le programme trouve l'un ou l'autre de ces extrémums.

On peut également vérifier que les valeurs de f(x,y) sont trouvées avec la précision maximale.

5. *Méthode du gradient*

a. Principe

Avec la méthode de relaxation, la fonction est optimisée alternativement dans la direction de l'axe Ox puis dans celle de l'axe Oy.

Avec la méthode du gradient, la fonction est systématiquement optimisée dans la direction du gradient de la fonction.

A partir d'un point de départ (x_0,y_0) on forme la suite :

$$x_{n+1} = x_n + \mu\, f'_x(x_n,y_n)$$
$$y_{n+1} = y_n + \mu\, f'_y(x_n,y_n)$$

avec :

$$\mu = \frac{f'^2_x(x_n,y_n) + f'^2_y(x_n,y_n)}{f'^2_x(x_n,y_n).f''_x(x_n,y_n)+2f'_x(x_n,y_n).f'_y(x_n,y_n).f''_{xy}(x_n,y_n)+f'^2_y(x_n,y_n).f''_y(x_n,y_n)}$$

La notation $f''_{xy} = \frac{\partial^2 f}{\partial x\, \partial y}$ représente la dérivée mixte de f par rapport à x et y.

Si cette suite (x_n, y_n) converge, elle converge vers un extrémum de la fonction.

b. Programme

L'utilisation est identique à celle du programme utilisant la méthode de relaxation.

La dérivée mixte est calculée par la formule approchée :

$$f''_{xy}(x,y) = \frac{f(x+h,y+h) + f(x-h,y-h) - f(x-h,y+h) - f(x+h,y-h)}{4h^2}$$

```
4000 CLS
4010  PRINT "F(X,Y):METHODE DU GRADIENT": PRINT : PRINT

4020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
4030  PRINT "2- RECHERCHE DES EXTREMA": PRINT : PRINT
4040  INPUT "VOTRE CHOIX:";E: PRINT
4050  IF E = 1 THEN  LIST 11000 - 11999: END
4060  INPUT "XO=";X
4070  INPUT "YO=";Y
4080  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

4090 H = .001
4100  FOR K = 1 TO 100
4110  GOSUB 11000:B =  - 2 * F:D =  - 2 * F
4120 X = X + H: GOSUB 11000:A = F:B = B + F
4130 X = X - 2 * H: GOSUB 11000:A = A - F:B = B + F
4140 A = A / 2 / H:B = B / H / H:X = X + H
4150 Y = Y + H: GOSUB 11000:C = F:D = D + F
4160 Y = Y - 2 * H: GOSUB 11000:C = C - F:D = D + F
4170 C = C / 2 / H:D = D / H / H
4180 X = X - H: GOSUB 11000:L = F
4190 X = X + 2 * H: GOSUB 11000:L = L - F
4200 Y = Y + 2 * H: GOSUB 11000:L = L + F
4210 X = X - 2 * H: GOSUB 11000:L = L - F
4220 X = X + H:Y = Y - H
4230 M = A * A * B + 2 * A * C * L + C * C * D
4240  IF M = 0 THEN  PRINT "LA METHODE EST INADAPTEE": GOTO
    4360
4250 M = (A * A + C * C) / M
4260 X = X - M * A:Y = Y - M * C
```

```
4270  IF  ABS (M) * ( ABS (A) +  ABS (C)) > E THEN 4340

4280 X = X + .01:Y = Y + .01: GOSUB 11000:N = F
4290 X = X - .01:Y = Y - .01: GOSUB 11000
4300  PRINT : PRINT "X=";X: PRINT "Y=";Y: PRINT
4310  IF F > N THEN  PRINT "F(X,Y)=";F;"  (MAXIMUM)"
4320  IF F < N THEN  PRINT "F(X,Y)=";F;"  (MINIMUM)"
4330  GOTO 4360
4340  NEXT K
4350  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
4360  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
4370  IF Z$ = "O" THEN 4000
4380  END
11000 F = X * X * (45 + X * (2 - X)) + Y * Y * (45 + Y *
    (2 - Y))
11010  RETURN
```

c. Exemples commentés

◦ *Exemple 1* :

Reprenons l'étude de la fonction :

$$\frac{x^2 - 4x - 1}{xy - y^2 + 16 - 2|x|} + \frac{y}{2x - 5}$$:

```
11000F=(X*X-4*X-1)/(X*Y-Y*Y+16-2*ABS(X))+(Y/(2*X-5))


F(X,Y):METHODE DU GRADIENT


XO=1
YO=-2
PRECISION (1 A 8):2


X=1.11815993
Y=-1.9697479

F(X,Y)=.163031856  (MINIMUM)
```

```
VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DU GRADIENT


XO=1
YO=-2
PRECISION (1 A 8):5


X=1.11488373
Y=-1.97135385

F(X,Y)=.163028363  (MINIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Nous retrouvons la même valeur de f, mais les valeurs de x et y ne coïncident qu'à 10^{-5} près: ceci reste toutefois conforme aux remarques précédentes sur la précision.

- ***Exemple 2*** :

```
 11000 F=X*X*(90+X*(4-2*X))+Y*Y*(90+Y*(4-2*Y))


F(X,Y):METHODE DU GRADIENT


XO=10
YO=10
PRECISION (1 A 8):5


X=5.55234225
Y=5.55234332

F(X,Y)=3116.90732  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):METHODE DU GRADIENT


XO=-3
YO=3
PRECISION (1 A 8):5
```

```
X=5.55234345
Y=-4.05234325

F(X,Y)=2230.875  (MAXIMUM)

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

De la même manière qu'avec la méthode de la relaxation, un maximum différent est obtenu selon le point de départ choisi.

6. *Diminution et augmentation systématiques de la fonction*

a. Principe

De même que pour les fonctions d'une variable, il peut être utile, voire indispensable, de ne pouvoir effectuer que la recherche des minimums (ou des maximums) d'une fonction de deux variables.

Examinons le cas d'un minimum.

On fixe au départ un point (x_0,y_0) et un pas initial de découpage h.

On examine alors les valeurs prises par la fonction aux différents points:

$$(x_0 + h,y_0 + h)\,;\,(x_0 + h,y_0 - h)\,;\,(x_0 - h,y_0 + h)\,;\,(x_0 - h,y_0 - h)$$

Si en l'un de ces points, la fonction est plus petite qu'en (x_0,y_0), on remplace (x_0,y_0) par ce point et on recommence.

Lorsque les quatre valeurs de la fonction sont supérieures, le pas h est divisé par 2, et l'encadrement se poursuit jusqu'à ce que h soit inférieur à une valeur fixée à l'avance caractérisant la précision.

b. Programme

La fonction doit ici encore être définie à la ligne 11ØØØ du programme sous la forme:

```
11ØØØ F = COS(X*X-SIN(Y)/Y+3)
```

Il faut ensuite introduire au programme le point de départ (x_0,y_0), le pas initial h ainsi que la précision.

Programme de recherche des minimums:

```
5000 CLS
5010  PRINT "F(X,Y):DIMINUTION SYSTEMATIQUE": PRINT : PRINT

5020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
5030  PRINT "2- RECHERCHE DES MINIMA": PRINT : PRINT
5040  INPUT "VOTRE CHOIX:";E: PRINT
5050  IF E = 1 THEN  LIST 11000 - 11999: END
5060  INPUT "XO=";X
5070  INPUT "YO=";Y
5080  INPUT "H=";H
5090  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

5100  FOR K = 1 TO 200
5110  GOSUB 11000:A = F
5120 X = X - H:Y = Y - H: GOSUB 11000
5130  IF A > F THEN 5230
5140 X = X + 2 * H: GOSUB 11000
5150  IF A > F THEN 5230
5160 Y = Y + 2 * H: GOSUB 11000
5170  IF A > F THEN 5230
5180 X = X - 2 * H: GOSUB 11000
5190  IF A > F THEN 5230
5200 X = X + H:Y = Y - H
5210  IF H < E THEN  PRINT : PRINT "X=";X: PRINT "Y=";Y
    : PRINT : PRINT "F(X,Y)=";A: GOTO 5250
5220 H = H / 2
5230  NEXT K
5240  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
5250  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
5260  IF Z$ = "O" THEN 5000
5270  END
11000 F = X * X * (45 + X * (2 - X)) + Y * Y * (45 + Y *
    (2 - Y))
11010  RETURN
```

Programme de recherche des maximums:

```
7000 CLS
7010  PRINT "F(X,Y):AUGMENTATION SYSTEMATIQUE": PRINT :
     PRINT
7020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
7030  PRINT "2- RECHERCHE DES MAXIMA": PRINT : PRINT
7040  INPUT "VOTRE CHOIX:";E: PRINT
7050  IF E = 1 THEN  LIST 11000 - 11999: END
7060  INPUT "XO=";X
7070  INPUT "YO=";Y
7080  INPUT "H=";H
7090  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E): PRINT

7100  FOR K = 1 TO 200
7110  GOSUB 11000:A = F
7120 X = X - H:Y = Y - H: GOSUB 11000
7130  IF A < F THEN 7230
7140 X = X + H + H: GOSUB 11000
7150  IF A < F THEN 7230
7160 Y = Y + H + H: GOSUB 11000
7170  IF A < F THEN 7230
7180 X = X - H - H: GOSUB 11000
7190  IF A < F THEN 7230
7200 X = X + H:Y = Y - H
7210  IF H < E THEN  PRINT : PRINT "X=";X: PRINT "Y=";Y
    : PRINT : PRINT "F(X,Y)=";A: GOTO 7250
7220 H = H / 2
7230  NEXT K
7240  PRINT "LE CALCUL NECESSITE TROP D'ITERATIONS"
7250  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
7260  IF Z$ = "O" THEN 7000
7270  END
11000 F = X * X * (45 + X * (2 - X)) + Y * Y * (45 + Y *
    (2 - Y))
11010  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

Considérons la fonction :

$$f(x,y) = \cos(x^2 - \frac{\sin y}{y} + 3)$$

```
11000 F=COS(X*X-SIN(Y)/Y+3)
F(X,Y):DIMINUTION SYSTEMATIQUE


XO=2
YO=2
H=.23
PRECISION (1 A 8):2


X=.957812501
Y=1.1878125

F(X,Y)=-.999987095

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):DIMINUTION SYSTEMATIQUE


XO=2
YO=2
H=.23
PRECISION (1 A 8):8


X=.96133606
Y=1.18288513

F(X,Y)=-1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):DIMINUTION SYSTEMATIQUE


XO=2
```

```
YO=2
H=.22
PRECISION (1 A 8):8


X=1.05108887
Y=.47262207

F(X,Y)=-1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Cette fonction possède une infinité de minimums correspondant tous à la valeur (−1). Les valeurs initiales x_0, y_0 et h influent alors énormément sur le minimum obtenu : dans l'exemple, à (x_0,y_0) fixé, les deux valeurs $h = 0{,}23$ et $h = 0{,}22$ conduisent à des minimums différents.

- ***Exemple 2*** :

Reprenons un des problèmes du chapitre précédent, à savoir la détermination d'une racine du système

$$\begin{cases} f(x,y) = 0 \\ g(x,y) = 0 \end{cases}$$

Posons :

$$h(x,y) = f^2(x,y) + g^2(x,y)$$

La fonction h est toujours positive ; 0 en est donc un minimum.

On peut donc chercher les minimums de h, et ceux qui correspondront à une valeur nulle de la fonction seront racines du système initial.

Considérons l'équation complexe $z^2 + z + 1 = 0$, qui peut être résolue en étudiant le système :

$$\begin{cases} x^2 - y^2 + x + 1 = 0 \\ 2xy + y = 0 \end{cases}$$

Minimisons alors la fonction :

$$h(x,y) = (x^2 - y^2 + x + 1)^2 + (2xy + y)^2$$

```
 11000  F=(X*X-Y*Y+X+1)^2+(2*X*Y+Y)^2


F(X,Y):DIMINUTION SYSTEMATIQUE
```

```
XO=5
YO=5
H=2
PRECISION (1 A 8):7


X=-.5
Y=-.866025448

F(X,Y)=5.83214032E-15

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):DIMINUTION SYSTEMATIQUE


XO=-3
YO=3
H=1
PRECISION (1 A 8):7


X=-.5
Y=.866025448

F(X,Y)=5.83214032E-15

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les deux racines $(-\frac{1}{2}, \frac{\sqrt{3}}{2})$ et $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$ sont trouvées. (Du fait des erreurs numériques, la valeur de f, de l'ordre de 10^{-14}, peut être considérée comme nulle.)

- ***Exemple 3*** :

Cherchons maintenant les maximums de la fonction :

$$f(x,y) = \cos(x^2 - \frac{\sin y}{y} + 3):$$

```
11000 F=COS(X*X-SIN(Y)/Y+3)


F(X,Y):AUGMENTATION SYSTEMATIQUE
```

```
XO=2
YO=2
H=.23
PRECISION (1 A 8):8


X=1.96854767
Y=1.6792508

F(X,Y)=1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
F(X,Y):AUGMENTATION SYSTEMATIQUE


XO=2
YO=2
H=.3
PRECISION (1 A 8):8


X=1.92490845
Y=2.07472534

F(X,Y)=1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La fonction possède une infinité de maximums correspondant à la valeur 1. Ici encore, le choix de x_0, y_0 et h conditionne le résultat.

7. Conclusion sur les méthodes d'optimisation

La recherche d'extrémums des fonctions de deux variables constitue un problème délicat.

Les difficultés proviennent plus souvent de la fonction elle-même que des méthodes de résolution : les fonctions ont fréquemment des branches asymptotiques vers l'infini qui font diverger la méthode, ou encore une infinité d'extrémums.

Pour ces raisons, il est préférable de connaître un ordre de grandeur de l'extrémum cherché, de manière à pouvoir écarter les résultats erronés.

4 Représentations graphiques

1. Introduction

Nous allons maintenant nous tourner vers les possibilités graphiques des micro-ordinateurs, et utiliser l'écran pour afficher en haute-résolution les représentations de plusieurs types de fonctions.

La plupart des micro-ordinateurs possède un mode graphique haute-résolution ainsi que de puissantes instructions BASIC permettant de l'utiliser facilement.

Nous pourrons donc tracer les courbes d'équation $y = f(x)$, les courbes d'équations paramétrées et les courbes d'équation polaire.

Nous verrons ensuite un programme de représentation de surfaces en perspective cavalière, qui nous permettra de mieux appréhender les fonctions de deux variables.

Nous nous intéresserons enfin à titre d'application à la simulation du jeu Spirographe, qui permet à l'aide de roues dentées de dessiner toutes sortes de rosaces.

Il est cette fois indispensable d'introduire le menu suivant avec le premier programme; en effet la présence de sous-programmes entre les lignes 5ØØ et 9ØØ empêche un fonctionnement autonome des programmes:

```
100  TEXT :CLS
110  PRINT  TAB( 5);"REPRESENTATIONS GRAPHIQUES": PRINT

120  PRINT : PRINT "1- COURBE                Y=F(X)"
130  PRINT : PRINT "2- COURBE PARAMETREE   X=F(T) Y=G(T)
    "
140  PRINT : PRINT "3- COURBE POLAIRE       R=F(T)"
150  PRINT : PRINT "4- SURFACE              Z=F(X,Y)"
160  PRINT : PRINT "5- SIMULATION DU SPIROGRAPH"
170  PRINT : PRINT "6- FIN"
180  PRINT : INPUT "VOTRE CHOIX:";E
190  ON E GOTO 1000,2000,3000,4000,5000,6000
200  GOTO 180
6000  END
```

L'ensemble nécessite environ 8 Koctets de mémoire.

2. *Courbes d'équation* $y = f(x)$

a. Programme

La méthode utilisée est très simple; l'axe horizontal étant celui des abscisses x, on associe à chaque x la valeur f(x) et on trace un segment de droite entre le point (x,f(x)) et le point précédent.

Il faut définir la fonction à représenter à partir de la ligne 5ØØ, sous la forme:

```
5ØØ Y=1
51Ø IF X<> 0 THEN Y=SIN(X)/X
599 RETURN
```

Après avoir lancé le programme, il faut introduire les bornes du domaine de définition.

Il est possible de choisir l'échelle sur l'axe vertical en introduisant les valeurs minimale et maximale désirées; dans le cas d'une réponse négative, le programme calcule les extrémums de la fonction, et représente la fonction en utilisant toute la hauteur de l'écran.

Les deux possibilités présentent des avantages : le fait de choisir l'échelle permet par exemple de centrer les axes sur l'écran et de maintenir une échelle constante, ce qui autorise une comparaison directe entre différentes fonctions. L'échelle automatique permet de tracer la courbe sans en connaître les extrémums, et utilise tout l'écran pour le tracé.

Il est également possible de représenter des axes gradués. Les valeurs des graduations sont des puissances entières de 10 (par exemple 0,01 ; 1 ; 10...) ; la valeur est calculée par le programme de façon à optimiser la lisibilité. Il faut donc avoir un ordre de grandeur des valeurs prises par la fonction pour savoir exactement la longueur comprise entre deux graduations.

Le programme utilisant la haute-résolution graphique de l'ordinateur, voici la signification des commandes graphiques employées :

HIRES : place l'ordinateur en mode haute-résolution.

TEXT : place l'ordinateur en mode texte. (Certains ordinateurs ne distinguent pas mode texte et mode haute-résolution.)

LINE(A,B)-(C,D) : trace un segment de droite entre les points de coordonnées (A,B) et (C,D).

Ces instructions (ou leur équivalent) sont présentes sur la plupart des micro-ordinateurs ; il ne devrait donc pas se présenter de difficultés d'adaptation.

Les valeurs C et L définies à la ligne 1Ø du programme définissent la résolution de l'ordinateur (C nombre de colonnes, L nombre de lignes) ; il faut donc adapter ces valeurs à chaque machine. Sur Apple II par exemple, la résolution est de 280 x 160 ; on programmera donc :

```
1Ø C=279: L=159
```

```
10 C = 279:L = 159
500 Y = 1
510  IF X <  > 0 THEN Y =  SIN (X) / X
599  RETURN
1000  TEXT :CLS
1010  PRINT  TAB( 5);"COURBE  Y=F(X)": PRINT
1020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
1030  PRINT "2- TRACE DE LA COURBE": PRINT : PRINT
1040  INPUT "VOTRE CHOIX ";E
1050  IF E = 1 THEN  LIST 500 - 599: END
1060  PRINT : PRINT
1070  INPUT "X MIN:";XN
1080  INPUT "X MAX:";XM
1090  IF XM = XN THEN 1070
```

```
1100  PRINT : PRINT : INPUT "VOULEZ-VOUS CHOISIR L'ECHE
    LLE?:";Z$
1110  IF Z$ <  > "O" THEN 1150
1120  PRINT : PRINT
1130  INPUT "Y MIN ";YN
1140  INPUT "Y MAX ";YM
1150  PRINT : INPUT "VOULEZ-VOUS LES AXES:";Y$
1160  IF Z$ = "O" GOTO 1260
1170 X = XN
1180  GOSUB 500
1190 YM = Y:YN = Y
1200 IN = (XM - XN) / 100
1210 X = X + IN
1220  GOSUB 500
1230  IF Y > YM THEN YM = Y
1240  IF Y < YN THEN YN = Y
1250  IF X < XM THEN 1210
1260  TEXT :HIRES
1270  IF Y$ = "O" THEN  GOSUB 10000
1280 I = 2
1290  GOSUB 1400
1300 XP = 2:YP = YY
1310  FOR I = 3 TO C - 2
1320  GOSUB 1400
1330  IF YP > L OR YP < 0 OR YY > L OR YY < 0 THEN 1350

1340 LINE(XP,YP) - (XP + 1,YY)
1350 XP = I:YP = YY
1360  NEXT I
1370  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
1380  IF Z$ = "O" THEN 1000
1390  GOTO 100
1400 X = XN + (I - 2) * (XM - XN) / (C - 4)
1410  GOSUB 500
1420 YY =  INT ((Y - YM) * (L - 4) / (YN - YM) + 0.5) +
    2
1430  RETURN
```

Le sous-programme 1ØØØØ se charge de représenter les axes et les graduations ; il est commun aux trois programmes de tracé de courbes, il suffira donc de l'introduire en même temps que le premier programme :

```
10000 XA =  INT (XN / (XN - XM) * (C - 4)) + 2:YA =  IN
    (YM / (YM - YN) * (L - 4)) + 2
10010  IF XA >  = 0 AND XA <  = C THEN LINE(XA,0) - (XA
    ,L)
10020  IF YA >  = 0 AND YA <  = L THEN LINE(0,YA) - (C,
    YA)
10030 PX = 10 ^ ( INT ( LOG ((XM - XN) / 4) /  LOG (10)
    ))
10040 PY = 10 ^ ( INT ( LOG ((YM - YN) / 4) /  LOG (10)
    ))
10050 HX = PX / (XM - XN) * (C - 4)
10060 HY = PY / (YM - YN) * (L - 4)
10070 X = XA:Y = YA
10080  IF YA < 0 OR YA > L THEN 10200
10090 DG =  - 2 + (2 - YA +  ABS (2 - YA)) / 2
10100 LG = 2 - (YA - L + 2 +  ABS (YA - L + 2)) / 2 - D
    G
10110  IF XA > C THEN 10160
10120 X = X + HX
10130  IF X < 0 THEN 10120
10140  IF X <  = C THEN LINE(X,Y + DG) - (X,Y + DG + LG
    ): GOTO 10120
10150 X = XA
10160  IF XA < 0 THEN 10200
10170 X = X - HX
10180  IF X > C THEN 10170
10190  IF X >  = 0 THEN LINE(X,Y + DG) - (X,Y + DG + LG
    ): GOTO 10170
10200  IF XA < 0 OR XA > C THEN  RETURN
10210 DG =  - 2 + (2 - XA +  ABS (2 - XA)) / 2
10220 LG = 2 - (XA - C + 2 +  ABS (XA - C + 2)) / 2 - D
    G
10230 X = XA
10240  IF YA > L THEN 10290
10250 Y = Y + HY
10260  IF Y < 0 THEN 10250
10270  IF Y <  = L THEN LINE(X + DG,Y) - (X + DG + LG,Y
    ): GOTO 10250
10280 Y = YA
10290  IF YA < 0 THEN  RETURN
10300 Y = Y - HY
10310  IF Y > L THEN 10300
10320  IF Y >  = 0 THEN LINE(X + DG,Y) - (X + DG + LG,Y
    ): GOTO 10300
10330  RETURN
```

b. Exemples

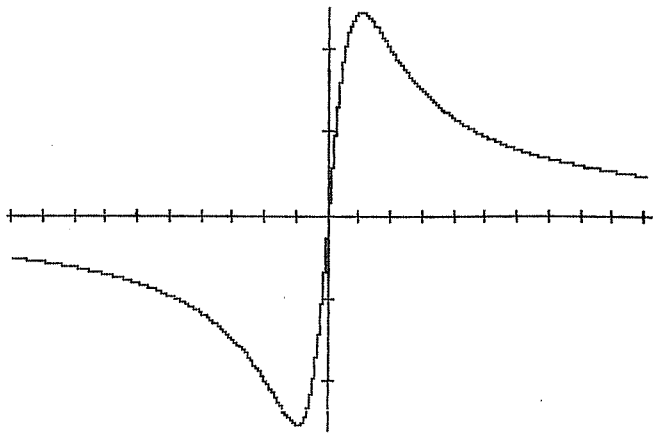
- ***Exemple 1*** :

Représentons la courbe d'équation :

$$y = \frac{5x}{1 + x^2}$$ (cubique serpentine)

pour :

$x \in [-10;10]$



Le tracé fait apparaître deux extrémums de la fonction situés approximativement aux points (−1 ;−2,5) qui correspond au minimum, et (1 ;2,5) qui correspond au maximum.

Le calcul des zéros de la dérivée de f confirme ce résultat.

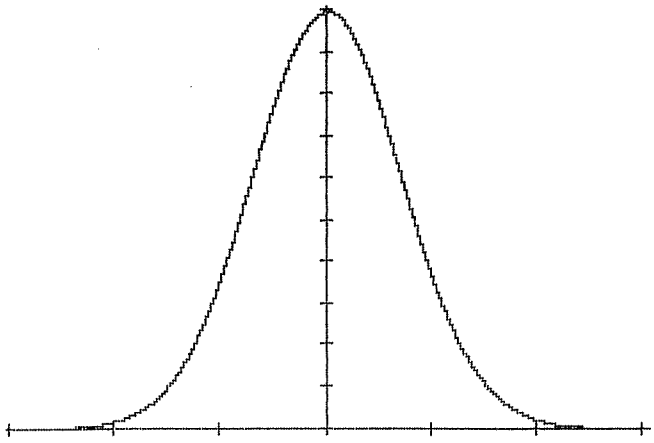
- ***Exemple 2*** :

Considérons l'équation de la gausienne.

$y = e^{-x^2}$

et traçons la courbe pour :

$x \in [-3,3]$



Cette courbe décroit très rapidement de part et d'autre de l'axe Oy et vient se confondre avec l'axe Ox.

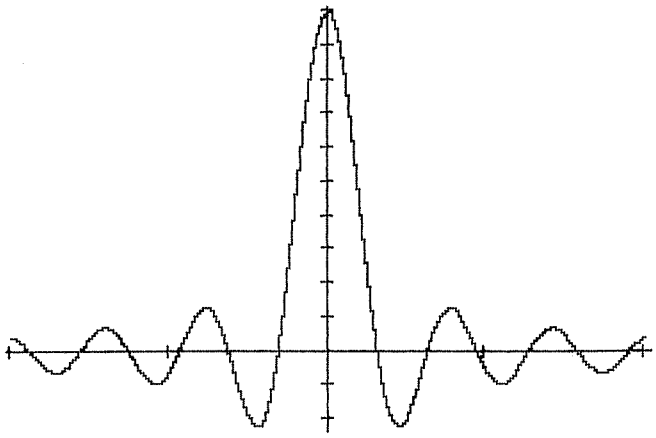
- ***Exemple 3*** :

Représentons la courbe d'équation

$$y = \frac{\sin x}{x}$$

pour :

$x \in [-20,20]$.



Les graduations valent dix unités sur l'axe des abscisses, un dixième d'unité sur l'axe des ordonnées.

3. Courbes d'équations paramétrées

a. Principe

Les courbes paramétrées sont définies par deux équations donnant les coordonnées x et y d'un point de la courbe en fonction d'un paramètre t que l'on fait varier :

$$\begin{cases} x = f(t) \\ y = g(t) \end{cases}$$

On remarque que les courbes d'équation $y = f(x)$ sont un cas particulier de courbes paramétrées ; en effet, si l'on pose $x = t$, il vient :

$$\begin{cases} x = t \\ y = g(t) = g(x) \end{cases}$$

La représentation est plus délicate que celle d'une courbe $y = f(x)$; pour celle-ci, il suffisait en effet de calculer des points correspondants à des valeurs équidistantes de x pour obtenir une excellente représentation.

Dans le cas de courbes paramétrées, il n'est pas possible de procéder de la même manière avec le paramètre t ; si on fixe un incrément Δt constant, il pourra arriver qu'entre deux valeurs de t séparées de Δt, les points soient très éloignés, ou alors confondus ; la seule solution pour obtenir un tracé précis consisterait alors à choisir un incrément Δt très faible ; le tracé serait cependant très lent.

Nous avons donc décidé d'utiliser un incrément variable. Nous fixons au départ une valeur de l'incrément, nous calculons deux points consécutifs et la distance les séparant ; si celle-ci est supérieure à une valeur fixée, nous divisons l'incrément par 2 ; si au contraire la distance est plus faible qu'une autre valeur fixée, nous multiplions le pas par 2.

Cette solution permet donc de tracer des segments de longueur à peu près constante, ce qui optimise finalement le temps de calcul et la précision du tracé.

b. Programme

Il faut définir les deux fonctions donnant X et Y en fonction de T à partir de la ligne 8ØØ du programme sous la forme :

```
8ØØ X=COS(T)
81Ø Y=SIN(T)
899 RETURN
```

Il faut ensuite introduire les valeurs extrêmes prises par le paramètre.

Il est ici encore possible de fixer les échelles cette fois sur les deux axes, ou de laisser le programme calculer les extrémums des deux fonctions X et Y et tracer la courbe sur tout l'écran.

```
10 C = 279:L = 159
700  IF PN = 1 THEN 800
800 U =  COS (T):V =  SIN (T)
810 X = U * U * U:Y = V * V * V
899  RETURN
2000  TEXT :CLS
2010  PRINT  TAB( 5);"COURBE PARAMETREE  X=F(T) Y=G(T)"
    : PRINT : PRINT
2020  PRINT "1- PROGRAMMATION DES FONCTIONS": PRINT
2030  PRINT "2- TRACE DE LA COURBE": PRINT
2040  INPUT "VOTRE CHOIX:";E
2050  IF E = 1 THEN  LIST 800 - 899: END
2060  PRINT : PRINT
2070  INPUT "T MIN ";A
2080  INPUT "T MAX ";B
2090  IF A = B THEN 2070
2100 PN = 1
2110  GOSUB 11000
2120  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
2130  IF Z$ = "O" THEN 2000
2140  GOTO 100
```

Le sous-programme 11ØØØ constitue en fait la partie active du programme puisque c'est lui qui trace la courbe ; il est commun aux programmes de courbes paramétrées et polaires, il ne sera donc pas nécessaire de l'introduire deux fois.

```
11000  PRINT : INPUT "VOULEZ-VOUS CHOISIR L'ECHELLE ";Z
    $
11010  IF Z$ <  > "O" THEN 11080
11020  PRINT : PRINT
11030  INPUT "X MIN ";XN
11040  INPUT "X MAX ";XM
11050  INPUT "Y MIN ";YN
11060  INPUT "Y MAX ";YM
11070  IF XN = XM OR YN = YM THEN 11030
11080  PRINT : INPUT "VOULEZ-VOUS LES AXES:";Y$
11090  IF Z$ = "O" THEN 11220
11100 T = A:F2 = 0
```

```
11110  GOSUB 700
11120 XM = X:XN = X
11130 YM = Y:YN = Y
11140 H = (B - A) / 100
11150 T = T + H
11160  GOSUB 700
11170  IF X > XM THEN XM = X
11180  IF X < XN THEN XN = X
11190  IF Y > YM THEN YM = Y
11200  IF Y < YN THEN YN = Y
11210  IF T < B THEN 11150
11220  TEXT :HIRES
11230  IF Y$ = "O" THEN  GOSUB 10000
11240 T = A:F2 = 0
11250  GOSUB 700
11260 XP =  INT ((X - XN) / (XM - XN) * (C - 4)) + 2
11270 YP =  INT ((Y - YM) / (YN - YM) * (L - 4)) + 2
11280 IN = (B - A) / 100
11290 T = T + IN
11300  GOSUB 700
11310  IF F1 = 1 THEN F1 = 0: GOTO 11260
11320 XX =  INT ((X - XN) / (XM - XN) * (C - 4)) + 2
11330 YY =  INT ((Y - YM) / (YN - YM) * (L - 4)) + 2
11340 D = (XX - XP) * (XX - XP) + (YY - YP) * (YY - YP)

11350 KK = KK + 1: IF KK = 10 THEN T = T + IN / 2:KK =
    0
11360  IF D > 15 THEN T = T - IN:IN = IN / 2: GOTO 1143
    0
11370  IF D < 2 THEN T = T - IN:IN = IN * 2: GOTO 11430

11380  IF XP > C OR XP < 0 OR YP > L OR YP < 0 THEN 114
    10
11390  IF XX > C OR XX < 0 OR YY > L OR YY < 0 THEN 114
    10
11400 LINE(XP,YP) - (XX,YY)
11410 KK = 0
11420 XP = XX:YP = YY
11430  IF T + IN < B THEN 11290
11440  RETURN
```

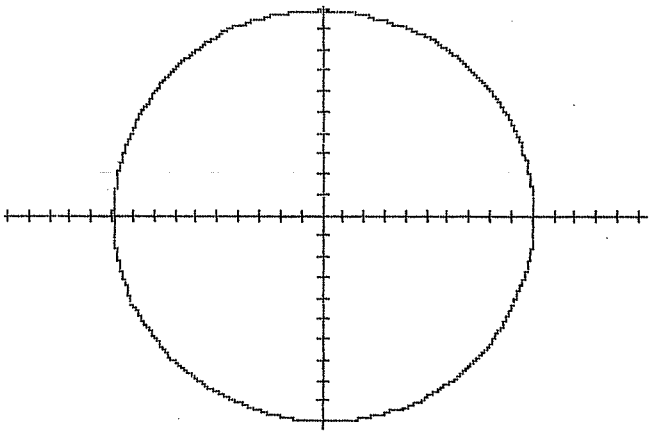
c. Exemples

- ***Exemple 1*** :

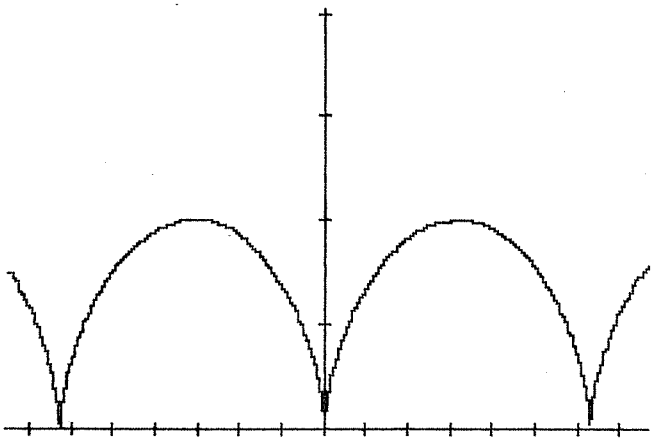
Soit le cercle d'équations :

$$\begin{cases} x = \cos t \\ y = \sin t \end{cases}$$

Le cercle est entièrement obtenu avec $t \in [0,2\pi]$.



- ***Exemple 2*** :



La trochoïde admet les équations paramétrées:

$$\begin{cases} x = 2t - 3\sin t \\ y = 2 - 3\cos t \end{cases}$$

Nous avons choisi cette fois l'échelle:

$x_{min} = -17$ $\qquad$ $x_{max} = 17$

$y_{min} = -1$ $\qquad$ $y_{max} = 8$

et $t \in [-9,9]$

• ***Exemple 3***:

La trajectoire d'un point appartenant à la circonférence d'une roue en mouvement est appelée cycloïde; ses équations sont:

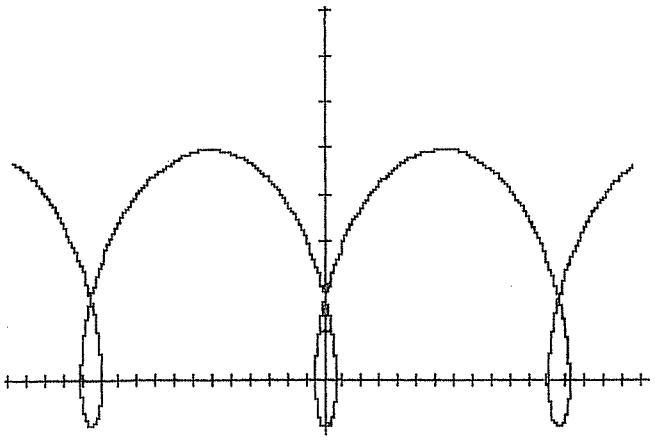
$$\begin{cases} x = t - \sin t \\ y = 1 - \cos t \end{cases}$$

Nous avons choisi:

$x_{min} = -7,5$ $\qquad$ $x_{max} = 7,5$

$y_{min} = 0$ $\qquad$ $y_{max} = 4$

et $t \in [-9,9]$

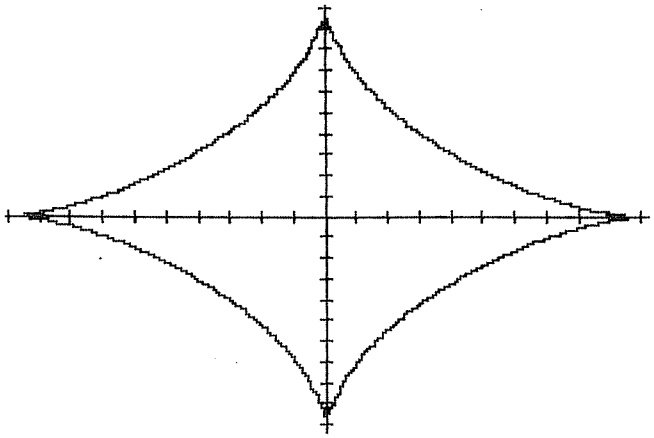


• ***Exemple 4***:

Les équations de l'hypocycloïde à quatre points de rebroussement sont:

$$\begin{cases} x = \cos^3 t \\ y = \sin^3 t \end{cases}$$

Représentons-la pour $t \in [0,2\pi]$



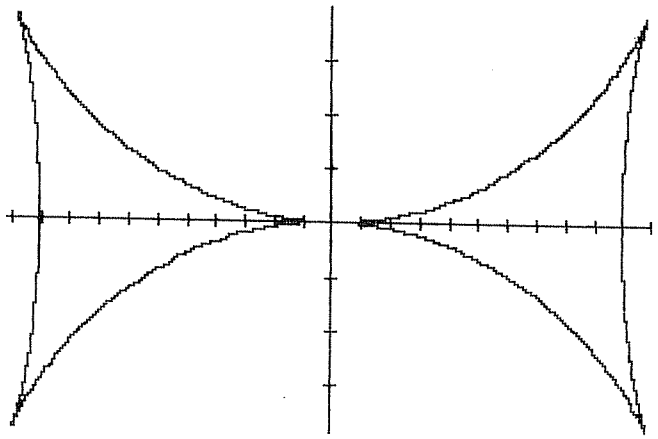
- ***Exemple 5*** :

Soit la courbe d'équations :

$$\begin{cases} x = (1 + \cos^2 t) \sin t \\ y = \sin^2 t \cos t \end{cases}$$

Traçons-la pour :

$t \in [0,2\pi]$



• ***Exemple 6*** :

Nous allons maintenant représenter le Folium de Descartes dont les équations sont :

$$\begin{cases} x = \dfrac{3t}{1+t^3} \\ y = \dfrac{3t^2}{1+t^3} \end{cases}$$

pour :

$t \in [-10,30]$

Les deux fonctions ne sont pas définies au point de paramètre $t = -1$; on a en effet :

$$\begin{cases} \lim\limits_{t \to -1^-} x(t) = +\infty \\ \lim\limits_{t \to -1^-} y(t) = -\infty \end{cases} \quad \text{et} \quad \begin{cases} \lim\limits_{t \to -1^+} x(t) = -\infty \\ \lim\limits_{t \to -1^+} y(t) = +\infty \end{cases}$$

Le programme tel que nous l'avons utilisé jusqu'à présent ne permettait que le tracé de courbes continues.

Nous avons cependant prévu un moyen de contourner les points de discontinuité (qui correspondent le plus souvent à des points où la fonction n'est pas définie).

Programmons alors :

```
800 IF T > - 1.2 AND F2=0 THEN T=-0.8:F1=1:F2=1
810 X=3*T/(1+T*T*T)
820 Y=3*T*T/(1+T*T*T)
899 RETURN
```

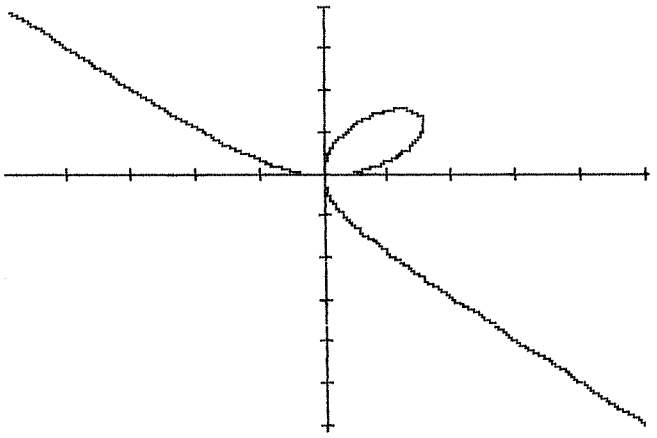
Nous avons introduit deux drapeaux F1 et F2.

Le fait de placer le drapeau F1 à 1 provoque une interruption du tracé : en effet, le programme procède habituellement en traçant un segment de droite entre le point courant et le point précédent. Le fait de placer F1 à 1 redéfinit le point de départ après avoir changé la valeur du paramètre, ce qui réalise le saut recherché.

Le drapeau F2 empêche simplement que le même changement de paramètre soit effectué plusieurs fois ; en effet, dès que T devient supérieur à -1.2 pour la première fois, le test est positif et le saut de paramètre est effectué ($T = -0.8$), le programme est prévenu du saut par $F1 = 1$, et le fait de placer F2 à 1 empêchera le test d'être positif à l'itération suivante, pour laquelle T sera toujours supérieur à $-1,2$.

Il est parfaitement possible de représenter des courbes possédant plusieurs discontinuités ; le programme s'écrirait alors par exemple :

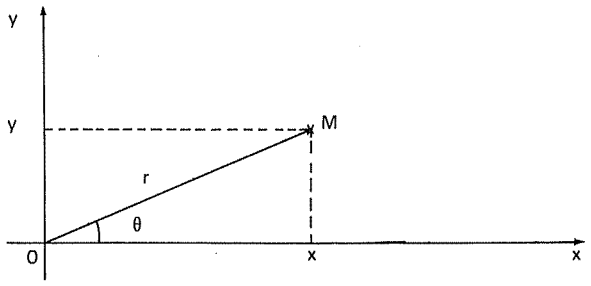
```
8ØØ IF T>0 AND F2=0 THEN T=1:F1=1:F2=1
81Ø IF T>10 AND F2=1 THEN T=11:F1=1:F2=2
.....
```



4. Courbes d'équation polaire

a. Principe

Une courbe polaire est une courbe dont l'équation est définie à partir des coordonnées polaires des points qui la constituent :



Les coordonnées polaires sont alors:

→ la distance r du point M à l'origine O,

→ l'angle θ défini par les deux demi-droite Ox et [OM).

Les formules de passage des coordonnées polaires aux coordonnées rectangulaires sont:

$$\begin{cases} x = r \cos\theta \\ y = r \sin\theta \end{cases}$$

Une équation polaire sera du type:

$$r = f(\theta)$$

b. Programme

Pour représenter les courbes polaires, le programme utilise les formules:

$$x = r \cos\theta$$
$$y = r \sin\theta$$

et se ramène au tracé d'une courbe d'équations paramétrées.

Pour cette raison, il n'est pas nécessaire de réintroduire le sous-programme 11ØØØ.

L'utilisation est donc identique à celle du programme précédent, seule diffère la définition de la fonction qui doit être effectuée à partir de la ligne 9ØØ:

```
9ØØ R=1
91Ø IF T<>0 THEN R=SIN(T)/T
999 RETURN
```

```
10 C = 279:L = 159
700  IF PN = 1 THEN 800
710  GOSUB 900
720 X = R *  COS (T)
730 Y = R *  SIN (T)
740  RETURN
900 R = 1
910  IF T <  > 0 THEN R =  SIN (T) / T
999  RETURN
3000  TEXT :CLS
3010  PRINT  TAB( 5);"COURBE POLAIRE  R=F(THETA)": PRINT
    : PRINT
3020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
```

```
3030  PRINT "2- TRACE DE LA COURBE": PRINT : PRINT
3040  INPUT "VOTRE CHOIX ";E
3050  IF E = 1 THEN  LIST 900 - 999: END
3060  PRINT : PRINT
3070  INPUT "TMIN ";A
3080  INPUT "TMAX ";B
3090  IF A = B THEN 3070
3100 PN = 2
3110  GOSUB 11000
3120  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
3130  IF Z$ = "O" THEN 3000
3140  GOTO 100
```

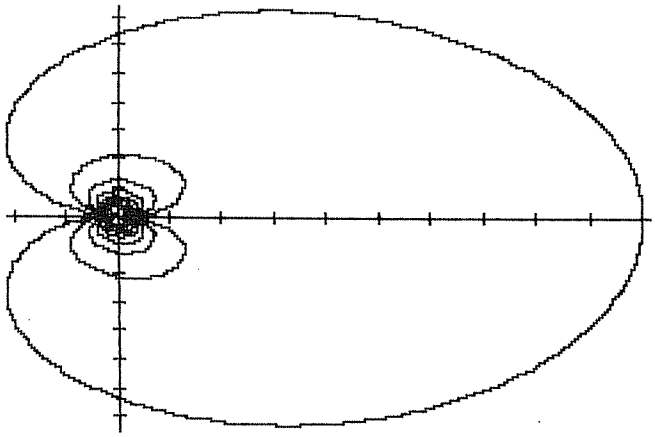
c. Exemples

• ***Exemple 1*** :

Représentons la cochléoïde d'équation :

$$r = \frac{\sin \theta}{\theta} \qquad \text{avec } \theta \in [-30;30]$$

Lorsque $|\theta|$ augmente, la courbe tend vers le point O en effectuant des boucles de plus en plus serrées.

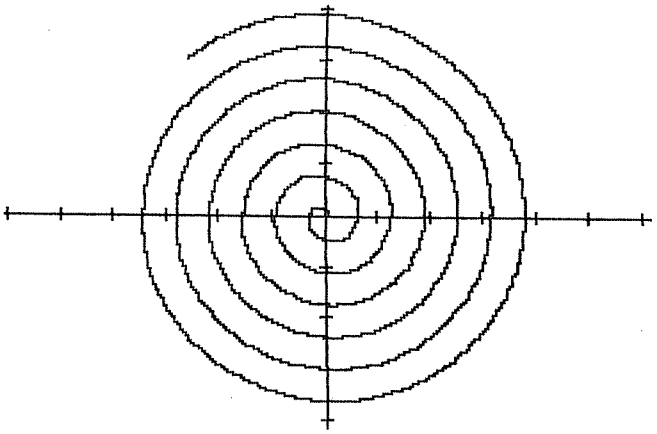


• ***Exemple 2*** :

La spirale d'Archimède possède une équation très simple :

$r = \theta$

Représentons-la pour $\theta \in [0,40]$.



• ***Exemple 3*** :

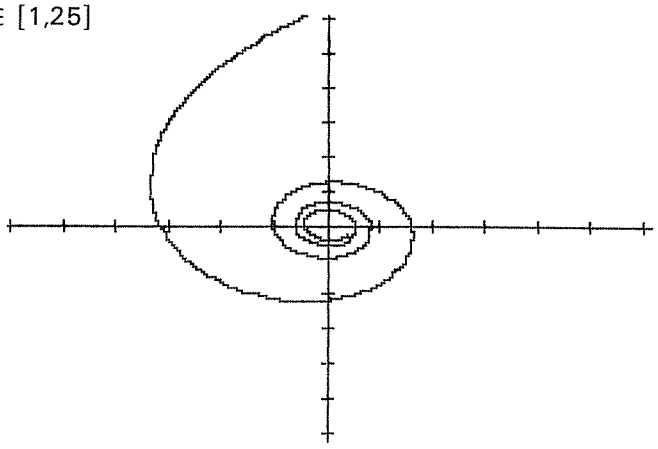
L'équation $r = \frac{1}{\theta}$ est celle de la spirale hyperbolique.

Nous choisirons cette fois l'échelle :

$x_{min} = -0,6$ $\quad x_{max} = 0,6$

$y_{min} = -0,6$ $\quad y_{max} = 0,6$

et $\theta \in [1,25]$



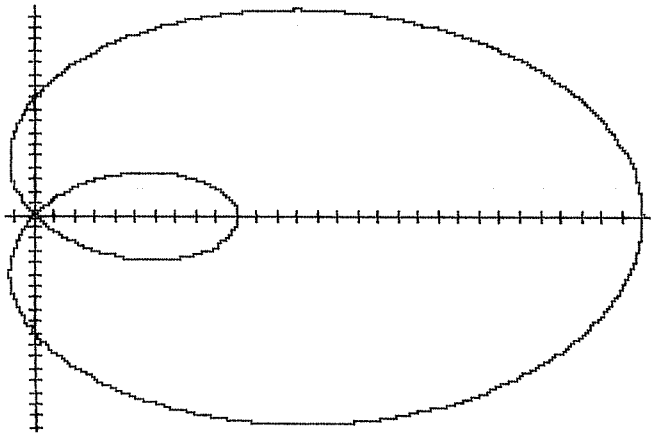
La courbe s'enroule autour du point O tout en s'en rapprochant : en effet $\lim_{\theta \to +\infty} r(\theta) = 0$, on dit alors que le point origine est point asymptote de la spirale.

- ***Exemple 4*** :

Le limaçon de Pascal a pour équation :

$$r = 1 + 2 \cos\theta$$

Nous le tracerons pour $\theta \in [0,2\pi]$.

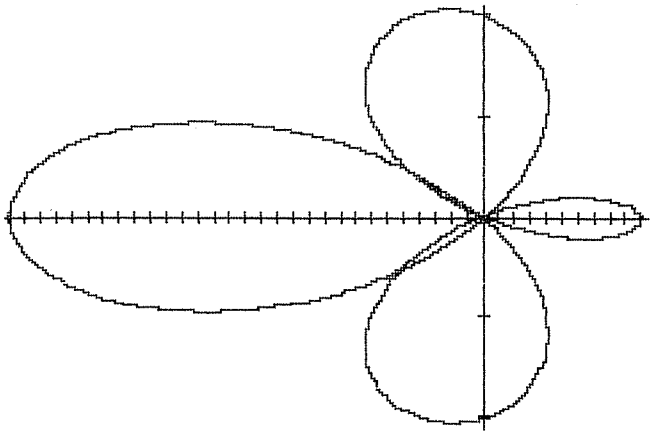


- ***Exemple 5*** :

La courbe d'équation :

$$r = 2 \cos 2\theta - \cos\theta$$

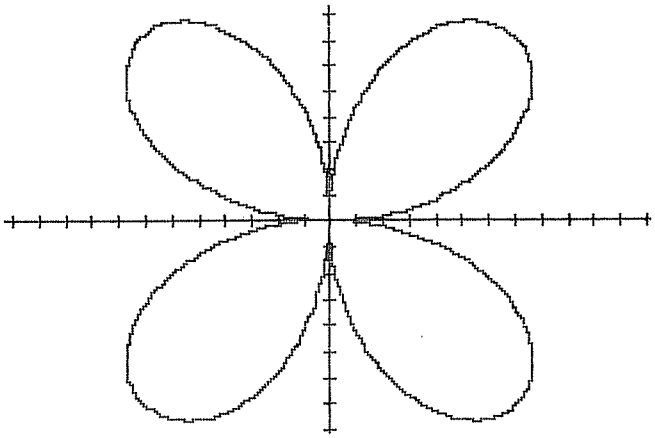
est appelée scarabée ; représentons-la pour $\theta \in [0,2\pi]$.



• ***Exemple 6*** :

Nous terminerons par la rosace à quatre branches :

$r = \sin 2\theta$ avec $\theta \in [0,2\pi]$.



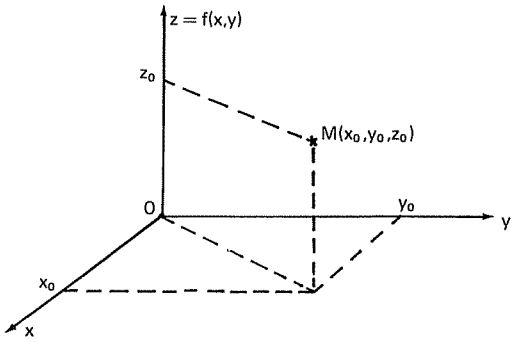
5. Représentation de surfaces

a. Principe

Nous nous intéressons maintenant aux fonctions de deux variables.

La représentation des fonctions d'une variable s'effectue dans un espace à deux dimensions ; une pour la fonction, une pour la variable. Il est donc logique que les fonctions de deux variables nécessitent trois dimensions ; toujours une pour la fonction, mais deux pour les variables. La représentation graphique est cette fois une surface.

Nous avons choisi de représenter ces surfaces en perspective cavalière, le choix des axes étant le suivant :



La fuyante (axe des x) est à 45° et le coefficient de réduction adopté est 0,5.

La méthode consiste alors à choisir deux valeurs x et y, à calculer f(x,y), et à placer le point M de coordonnées (x, y, z = f(x,y)) dans le repère (0, x, y, z) ; il ne reste plus ensuite qu'à effectuer la projection nécessitée par la perspective cavalière.

b. Programme

Le programme procède par quadrillage de la surface de définition (0, x, y) ; pour chaque point ainsi obtenu, des segments de droite sont tracés, le reliant aux points précédemment placés.

Plus le quadrillage effectué est serré, meilleure est la définition ; cependant la représentation peut devenir confuse. Nous avons donc laissé la possibilité de choisir la résolution du tracé, en fixant ce que nous avons appelé pas de représentation.

La résolution prise par défaut est de 12 pas sur l'axe Ox et de 16 pas sur l'axe Oy. La résolution maximale est 30 × 30 ; il est cependant possible de l'augmenter en redimensionnant les tableaux X% et Y% à la ligne 2Ø.

Pour utiliser le programme, il faut bien sûr commencer par définir la fonction : ceci se fait à partir de la ligne 49ØØ du programme, sous la forme :

```
49ØØ R=SQR(X*X+Y*Y)
491Ø Z=1
492Ø IF R<>0 THEN Z=SIN(R)/R
493Ø RETURN
```

Il faut ensuite introduire le domaine de définition de la fonction sous la forme des quatre bornes X_{min}, X_{max}, Y_{min}, Y_{max} et fixer également l'échelle sur l'axe Oz (Z_{min},Z_{max}).

Enfin, on peut éventuellement choisir la résolution.

```
10 C = 279:L = 159
20  DIM X%(30),Y%(30)
4000  TEXT :CLS
4010  PRINT  TAB( 5);"SURFACE  Z=F(X,Y)": PRINT : PRINT

4020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
4030  PRINT "2- TRACE DE LA SURFACE": PRINT : PRINT
4040  INPUT "VOTRE CHOIX ";E
4050  IF E = 1 THEN  LIST 4900 - 4999: END
4060  PRINT : PRINT
4070  INPUT "X MIN ";XN
4080  INPUT "X MAX ";XM
4090  INPUT "Y MIN ";YN
4100  INPUT "Y MAX ";YM
4110  INPUT "Z MIN ";ZN
4120  INPUT "Z MAX ";ZM
4130  IF XN = XM OR YN = YM OR ZN = ZM GOTO 4070
4140 K = 2 *  SQR (2)
4150 C1 =  INT (C / (1 +  SQR (2) / 4))
4160 C2 =  INT (C1 / K)
4170 LX = XM - XN
4180 LY = YM - YN
4190 LZ = ZM - ZN
4200 X1 = C - C1
4210 Y1 = ZM / LZ * (L - C2)
4220 MX = 12:MY = 16
4230  PRINT : PRINT : INPUT "VOULEZ-VOUS CHOISIR LA RES
    OLUTION ";Z$
4240  IF Z$ <  > "O" GOTO 4290
4250  PRINT : PRINT
4260  INPUT "PAS SUR L'AXE DES X ";MX
4270  INPUT "PAS SUR L'AXE DES Y ";MY
4280  IF MX > 30 OR MY > 30 THEN 4260
4290  TEXT :HIRES
4300 X = XN:Y = YN
4310  GOSUB 4900
4320 XX = X1
4330 YY =  INT (Y1 - (L - C2) * Z / LZ + 0.5)
4340  IF YY < 0 OR YY > L THEN YY = 2 * L
4350 X%(0) = XX:Y%(0) = YY
4360 XP = XX:YP = YY
4370  FOR RY = 1 TO MY
4380 Y = YN + RY * LY / MY
4390  GOSUB 4900
4400 XX =  INT (X1 + RY / MY * C1 + 0.5)
```

```
4410 YY =  INT (Y1 - (L - C2) * Z / LZ + 0.5)
4420  IF YY < 0 OR YY > L THEN YY = 2 * L
4430  IF YP < 0 OR YP > L OR YY < 0 OR YY > L GOTO 4450

4440 LINE(XP,YP) - (XX,YY)
4450 XP = XX:YP = YY
4460 X%(RY) = XX:Y%(RY) = YY
4470  NEXT
4480  FOR RX = 1 TO MX
4490 X = XN + RX * LX / MX:Y = YN
4500  GOSUB 4900
4510 XX =  INT (X1 - RX / K / MX * C1 + 0.5)
4520 YY =  INT (Y1 + RX / K / MX * C1 - (L - C2) * Z /
    LZ + 0.5)
4530  IF YY < 0 OR YY > L THEN YY = 2 * L
4540 KX = X%(0):KY = Y%(0)
4550 X%(0) = XX:Y%(0) = YY
4560 XP = XX:YP = YY
4570  IF KY < 0 OR KY > L OR YY < 0 OR YY > L GOTO 4590

4580 LINE(KX,KY) - (XX,YY)
4590  FOR RY = 1 TO MY
4600 Y = YN + RY * LY / MY
4610  GOSUB 4900
4620 XX =  INT (X1 + (RY / MY - RX / K / MX) * C1 + 0.5
    )
4630 YY =  INT (Y1 + RX / K / MX * C1 - (L - C2) * Z /
    LZ + 0.5)
4640  IF YY < 0 OR YY > L THEN YY = 2 * L
4650 KX = X%(RY):KY = Y%(RY)
4660  IF YP < 0 OR YP > L OR YY < 0 OR YY > L GOTO 4700

4670 LINE(XP,YP) - (XX,YY)
4680  IF KY < 0 OR KY > L GOTO 4700
4690 LINE(KX,KY) - (XX,YY)
4700 X%(RY) = XX:Y%(RY) = YY
4710 XP = XX:YP = YY
4720  NEXT RY
4730  NEXT RX
4740  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
4750  IF Z$ = "O" GOTO 4000
4760  GOTO 100
4900 R =  SQR (X * X + Y * Y)
4910 Z = 1
4920  IF R <  > 0 THEN Z =  SIN (R) / R
4930  RETURN
```

c. Exemples commentés

Les quatre premières surfaces que nous représenterons posséderont la symétrie de révolution autour de l'axe Oz ; elles s'appuieront de plus sur des courbes génératrices que nous avons déjà rencontrées.

- ***Exemple 1*** :

Posons :

$$r = \sqrt{x^2 + y^2}$$

Représentons alors la gaussienne de révolution dont l'équation est :

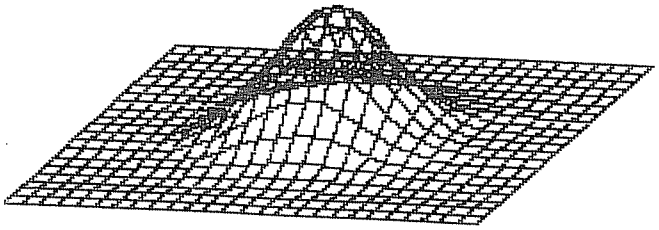
$$z = e^{-r^2}$$

Nous la programmerons ainsi :

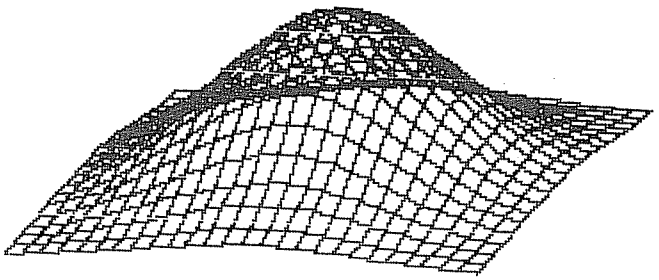
```
4900 R =  SQR (X * X + Y * Y)
4910 Z =  EXP ( - R * R)
4920  RETURN
```

et nous la tracerons avec :

$X_{min} = -3$ $\quad$ $X_{max} = 3$
$Y_{min} = -3$ $\quad$ $Y_{max} = 3$
$Z_{min} = 0$ $\quad$ $Z_{max} = 2$
résolution 20 x 26



$X_{min} = -1{,}5$ $\quad$ $X_{max} = 1{,}5$
$Y_{min} = -1{,}5$ $\quad$ $Y_{max} = 1{,}5$
$Z_{min} = 0$ $\quad$ $Z_{max} = 1{,}5$
résolution 20 x 26



Les représentations présentent donc l'aspect d'un grillage qui aurait été déformé pour épouser la forme de la surface.

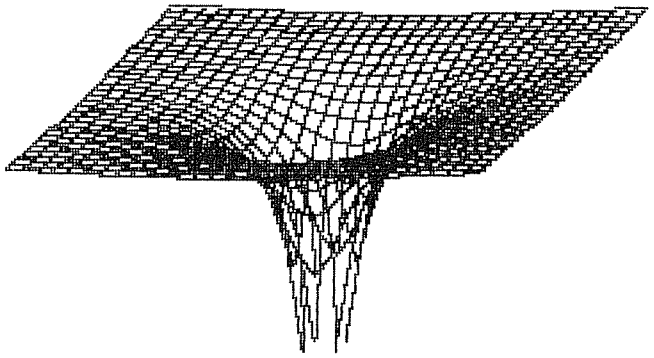
• ***Exemple 2*** :

Soit l'équation :

$$z = \frac{1}{r}$$

Traçons la surface pour :

$X_{min} = -1$ $\quad$ $X_{max} = 1$
$Y_{min} = -1$ $\quad$ $Y_{max} = 1$
$Z_{min} = -10$ $\quad$ $Z_{max} = 0$
résolution 30 x 30



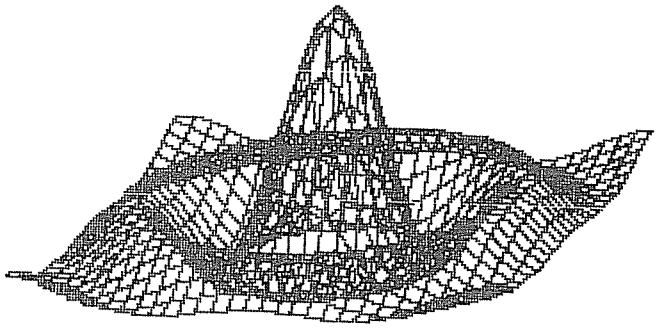
• ***Exemple 3*** :

Soit l'équation :

$$z = \frac{\sin(r)}{r}$$

On obtient avec :

$X_{min} = -10$ $\quad$ $X_{max} = 10$
$Y_{min} = -10$ $\quad$ $Y_{max} = 10$
$Z_{min} = -0{,}22$ $\quad$ $Z_{max} = 1$
résolution 30 x 30

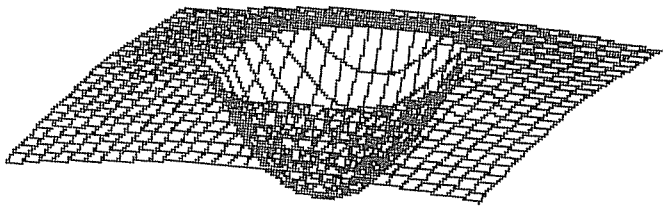


- ***Exemple 4*** :

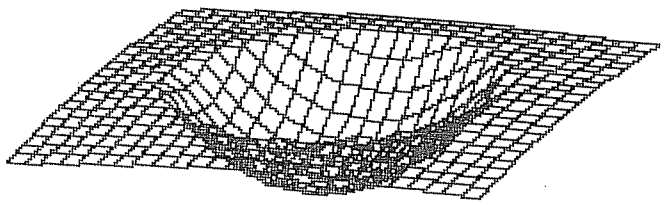
Pour terminer avec les surfaces de révolution, traçons la représentation de la fonction définie par :

```
4900 R =  SQR (X * X + Y * Y)
4910  IF R < 1 THEN Z = R * R
4920  IF R >  = 1 THEN Z =  EXP ((1 - R) / 5)
4930  RETURN
```

$X_{min} = -2$ $X_{max} = 2$
$Y_{min} = -2$ $Y_{max} = 2$
$Z_{min} = 0$ $Z_{max} = 2$
résolution 24 x 30



$X_{min} = -1{,}5$ $X_{max} = 1{,}5$
$Y_{min} = -1{,}5$ $Y_{max} = 1{,}5$
$Z_{min} = 0$ $Z_{max} = 2{,}5$
résolution 20 x 26



• ***Exemple 5*** :

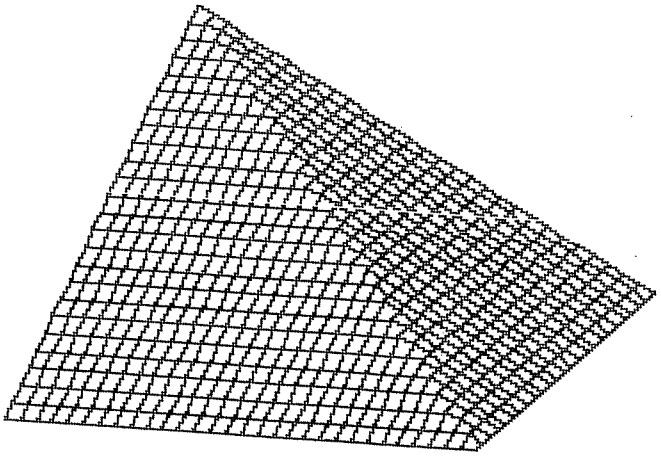
Nous allons représenter des intersections de plans, puis une intersection plans-cylindre.

Pour toutes ces surfaces, nous avons pris :

$X_{min} = 0$ $X_{max} = 1$
$X_{min} = 0$ $Y_{max} = 1$
$Z_{min} = 0$ $Z_{max} = 1$
résolution 24 x 30

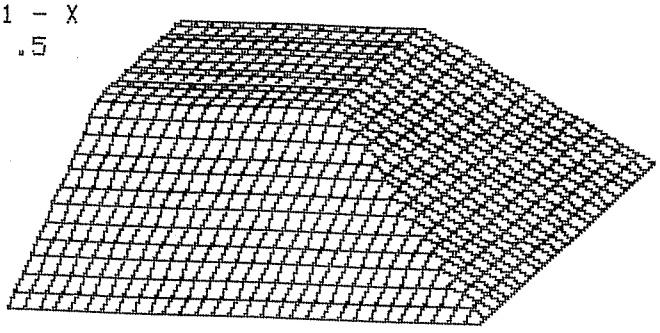
→ intersection de deux plans :

```
4900 Z = 1 - Y
4910  IF Y < X THEN Z = 1 - X
4920  RETURN
```



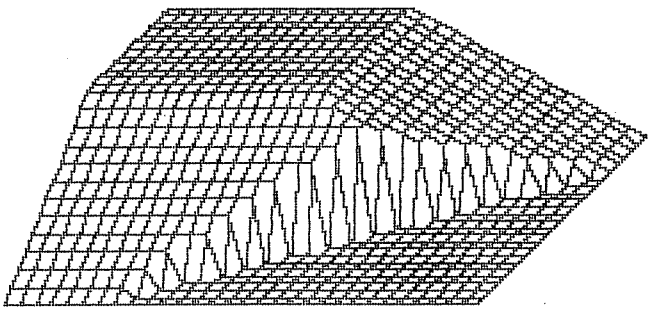
→ intersection de trois plans :

```
4900 Z = 1 - Y
4910  IF Y < X THEN Z = 1 - X
4920  IF Z > .5 THEN Z = .5
4930  RETURN
```



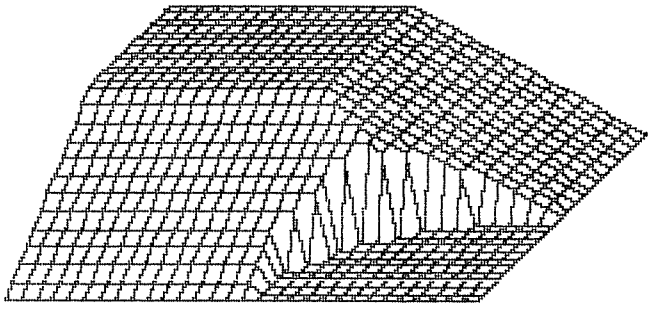
→ intersection de quatre plans :

```
4900 Z = 1 - Y
4910  IF Y < X THEN Z = 1 - X
4920  IF Z > .5 THEN Z = .5
4930 R = X + Y
4940  IF R > 1.2 THEN Z = 0
4950  RETURN
```



→ intersection de trois plans et d'un cylindre :

```
4900 Z = 1 - Y
4910  IF Y < X THEN Z = 1 - X
4920  IF Z > .5 THEN Z = .5
4930 R =  SQR ((1 - X) * (1 - X) + (1 - Y) * (1 - Y))
4940  IF R < .5 THEN Z = 0
4950  RETURN
```

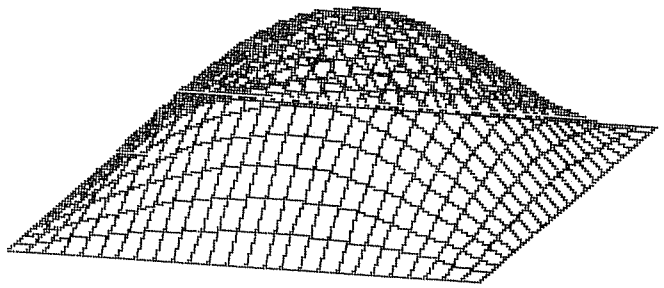


• ***Exemple 6*** :

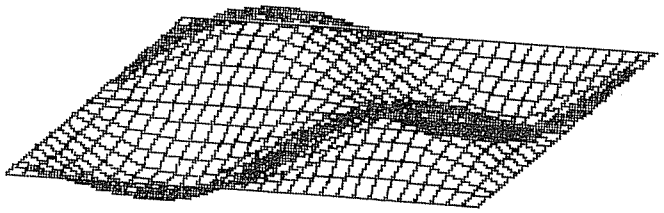
Soit la fonction :

$z = \sin x \cdot \sin y$

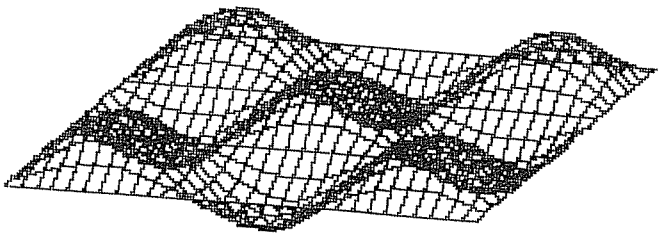
$X_{min} = 0$ $X_{max} = 3,14$
$Y_{min} = 0$ $Y_{max} = 3,14$
$Z_{min} = 0$ $Z_{max} = 1,5$
résolution 20 x 26



$X_{min} = -3,14$ $X_{max} = 3,14$
$Y_{min} = -3,14$ $Y_{max} = 3,14$
$Z_{min} = -2,5$ $Z_{max} = 2,5$
résolution 24 x 30



$X_{min} = 0$ $X_{max} = 9,42$
$Y_{min} = 0$ $Y_{max} = 9,42$
$Z_{min} = -2,5$ $Z_{max} = 2,5$
résolution 24 x 30

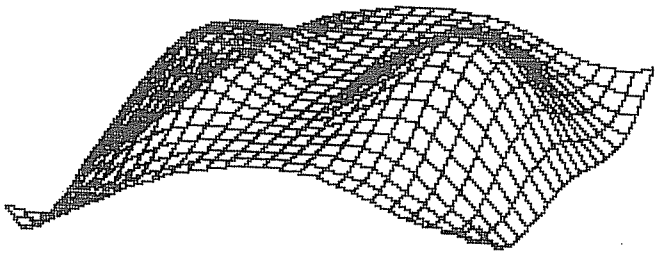


• ***Exemple 7*** :

Pour finir, traçons la courbe d'équation :

$$z = y^2 \frac{\sin r}{r}$$

$X_{min} = -4$ $X_{max} = 4$
$Y_{min} = -4$ $Y_{max} = 4$
$Z_{max} = -4$ $Z_{max} = 5$
résolution 20 x 26



6. Simulation du spirographe

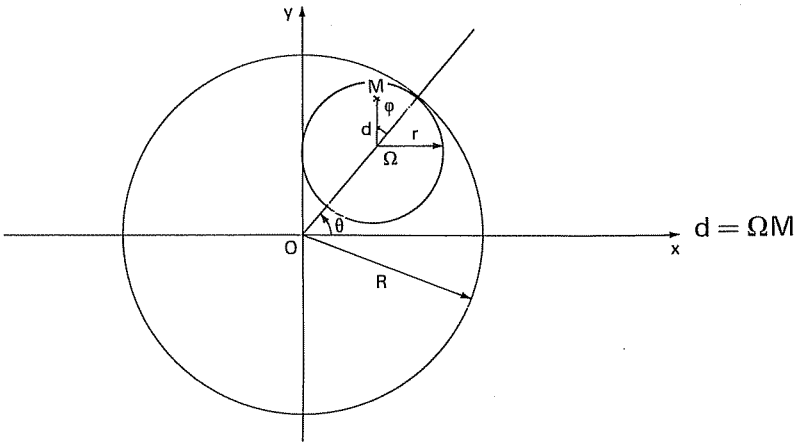
a. Principe

Le spirographe est un jeu consistant à faire tourner une roue dentée à l'intérieur d'une autre de diamètre supérieur.

On place un crayon dans un des trous pratiqués dans la petite roue, et on obtient diverses courbes hypocycloïdales, en faisant varier le nombre de dents des roues et la position du crayon.

b. Equations des courbes

Nous représenterons par M la position du stylo, par O le centre de la grande roue et par Ω le centre de la petite roue, selon la figure suivante :



Le stylo est à la distance d du centre de la petite roue, nous pouvons écrire :

$$d = k \frac{r}{10}$$

Ainsi, lorsque k varie entre 0 et 10, le point M passe du centre du cercle à un point de la circonférence.

Plus k est proche de 10, plus les courbes sont anguleuses ; inversement, si k se rapproche de 0, les courbes s'arrondissent.

On a alors :

$$\overrightarrow{OM} = \overrightarrow{O\Omega} + \overrightarrow{\Omega M}$$

soit :

$$\begin{cases} x = (R - r) \cos\theta + d \cos(\theta + \varphi) \\ y = (R - r) \sin\theta + d \sin(\theta + \varphi) \end{cases}$$

Puisque la petite roue tourne à l'intérieur de la grande, on a la relation supplémentaire :

$$r\varphi = -R\theta$$

En éliminant φ, on obtient :

$$x = (R - r) \cos\theta + \frac{k.r}{10} \cos\left[\left(1 - \frac{R}{r}\right)\theta\right]$$

$$y = (R - r) \sin\theta + \frac{k.r}{10} \sin\left[\left(1 - \frac{R}{r}\right)\theta\right]$$

Nous allons plutôt utiliser les nombres de dents des roues, soient N et n respectivement pour la grande et la petite roue. Nous choisissons de plus le facteur d'échelle R = 1.

Les équations sont donc finalement:

$$\begin{cases} x = (1 - \frac{n}{N}) \cos\theta + k \frac{n}{10.N} \cos[(1 - \frac{N}{n})\theta] \\ y = (1 - \frac{n}{N}) \sin\theta + k \frac{n}{10.N} \sin[(1 - \frac{N}{n})\theta] \end{cases}$$

c. Programme

Nous pouvons bien sûr utiliser le programme de tracé de courbes paramétrées pour tracer nos rosaces.

Cependant, nous avons écrit un programme particulier qui présente plusieurs avantages:

→ la période, qui est égale à $\frac{2\pi n}{PGCD(n,N)}$ est calculée automatiquement.

→ nous pouvons superposer plusieurs courbes correspondant à différentes valeurs de k, ce qui permet d'obtenir des figures complexes.

De plus, les axes ne sont plus représentés, le pas n'est plus variable car l'évolution des courbes est régulière, et l'échelle est fixée une fois pour toutes.

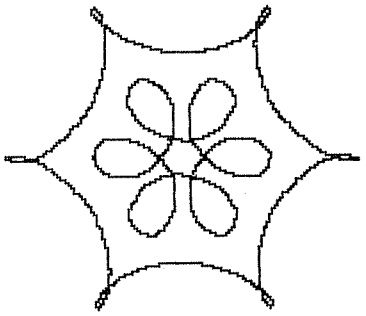
Cependant, pour obtenir des figures bien rondes, il convient d'ajuster selon l'ordinateur la valeur du paramètre XM à la ligne 5ØØØ du programme.

```
10 C = 279:L = 159
5000 XM = 1.7:IN = .1
5010  TEXT :CLS
5020  PRINT  TAB( 5);"SIMULATION DU SPIROGRAPH": PRINT
    : PRINT
5030  INPUT "GRANDE ROUE:";N2
5040  INPUT "PETITE ROUE:";N1
5050 R = N1 / N2:R1 = 1 - R:R2 = 1 - 1 / R
5060  TEXT :HIRES
5070  INPUT "K=";K
5080  IF K > 10 OR K < 0 THEN 5070
5090 K = K * R / 10
5100 A = N2:B = N1
5110 D = A - B *  INT (A / B)
5120  IF D <  > 0 THEN A = B:B = D: GOTO 5110
5130 B = N1 / B * 6.3
5140 T = 0
5150 XP =  INT (((1 - R + K) + XM) * C / 2 / XM)
```

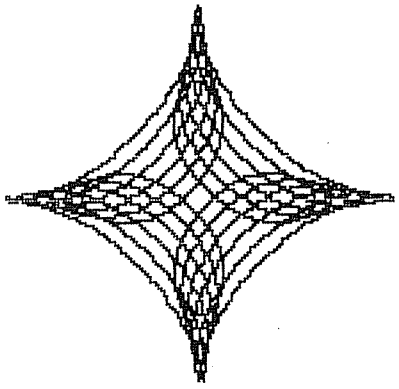
```
5160 YP =  INT (L / 2)
5170 T = T + IN:T2 = R2 * T
5180 X = R1 *  COS (T) + K *  COS (T2)
5190 Y = R1 *  SIN (T) + K *  SIN (T2)
5200 XX =  INT ((X + XM) * C / 2 / XM)
5210 YY =  INT ((1 - Y) / 2 * L)
5220 LINE(XP,YP) - (XX,YY)
5230 XP = XX:YP = YY
5240  IF T < B THEN 5170
5250  INPUT "AUTRE VALEUR DE K?:";Z$
5260  IF Z$ = "O" THEN 5070
5270  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
5280  IF Z$ = "O" THEN 5010
5290  GOTO 100
```

d. Exemples commentés

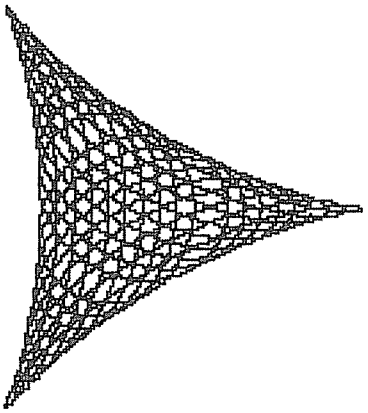
Nous donnons ici quelques rosaces ainsi que les valeurs des paramètres ayant permis de les obtenir.



N = 96 n = 80
K = 8 K = 3

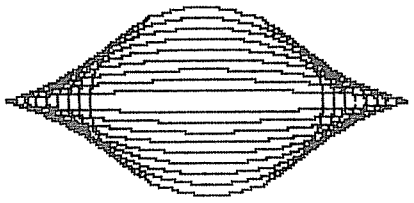


N = 96 n = 72
K = 9 à 4



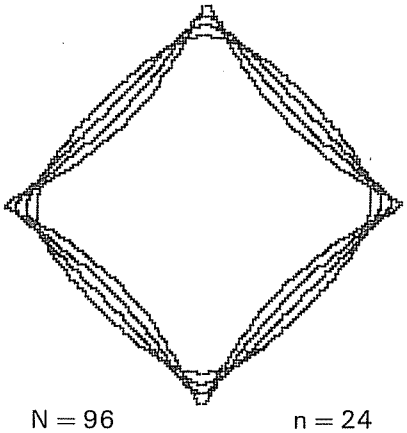
N = 96 n = 64

K = 9 à 1



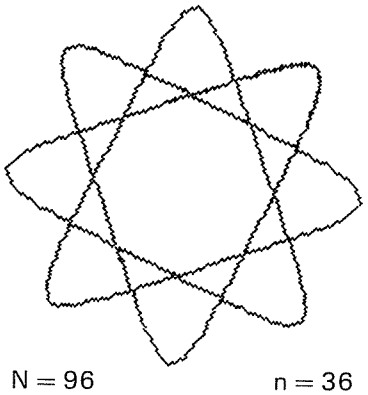
N = 96 n = 48

K = 9 à 1



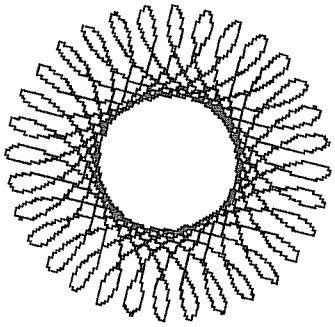
N = 96 n = 24

K = 8, 6, 4, 2



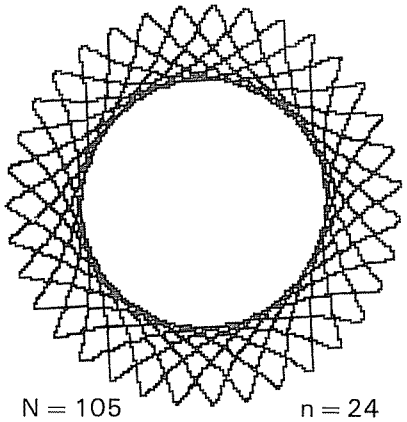
N = 96 n = 36

K = 6



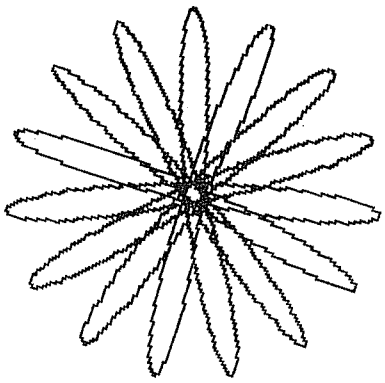
N = 96 n = 75

K = 7

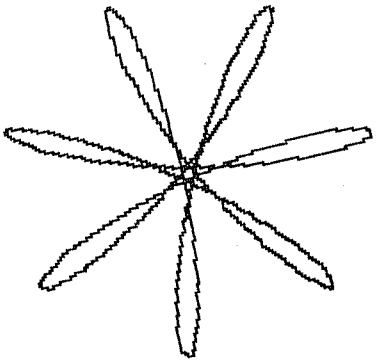


N = 105 n = 24

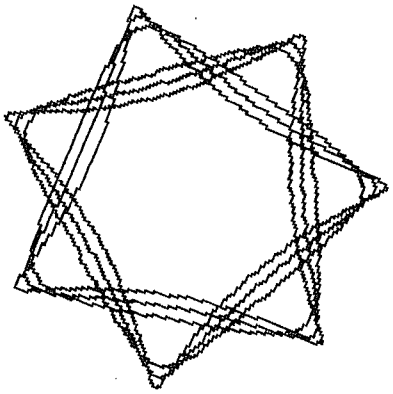
K = 8



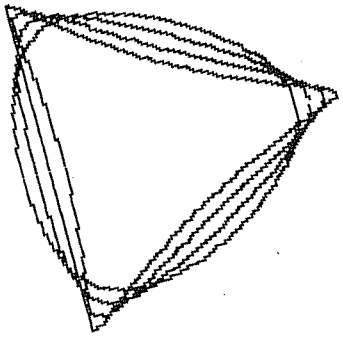
N = 105 n = 56
K = 8



N = 105 n = 60
K = 8



N = 105 n = 30
K = 8, 6, 4



N = 96 n = 32
K = 8, 6, 4, 2

5 Dérivées - développements limités

1. Introduction

Ce chapitre comporte deux parties :

→ une première partie avec deux programmes s'intéresse au calcul des dérivées successives et du développement limité d'une fonction f donnée jusqu'à l'ordre 5, en un point quelconque x_0 du domaine de définition de f.

→ une deuxième partie permet avec trois programmes le calcul des développements limités des fonctions f.g, f/g et gof à partir des développements limités des fonctions f et g.

Pour utiliser les cinq programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu :

```
100 CLS
110  PRINT "DERIVEES - DEVELOPPEMENTS LIMITES": PRINT
120  PRINT : PRINT "1- CALCUL DE DERIVEE N-IEME"
130  PRINT : PRINT "2- DEVELOPPEMENT LIMITE EN XO"
140  PRINT : PRINT "3- DEVELOPPEMENT LIMITE DE F.G"
```

```
150  PRINT : PRINT "4- DEVELOPPEMENT LIMITE DE F/G"
160  PRINT : PRINT "5- DEVELOPPEMENT LIMITE DE GOF"
170  PRINT : PRINT "6- FIN": PRINT
180  PRINT : INPUT "VOTRE CHOIX:";E
190  ON E GOTO 1000,2000,3000,4000,5000,6000
200  GOTO 100
6000  END
```

L'ensemble nécessite environ 8 Koctets de mémoire.

2. *Dérivées successives d'une fonction*

a. Principe

La dérivée d'une fonction f en x_0 est donnée par la formule :

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0 - h)}{2h}$$

On peut montrer que pour h donné, l'expression :

$$\frac{f(x_0 + h) - f(x_0 - h)}{2h}$$

est une approximation d'ordre 2 en h de $f'(x_0)$.

Cela signifie que la différence entre valeur calculée et valeur exacte est de l'ordre de grandeur de h^2, ce que l'on note :

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} + O(h^2)$$

Il existe d'autres formules, d'ordre 4, 6, 8, ... qui fournissent de meilleures approximations de $f'(x_0)$.

Ainsi, une approximation d'ordre 4 de $f'(x_0)$ est :

$$f'(x_0) = \frac{1}{12h}[8(f(x_0 + h) - f(x_0 - h)) - (f(x_0 + 2h) - f(x_0 - 2h))] + O(h^4)$$

De même, pour le calcul des dérivées d'ordre supérieur à 1, il existe des formules d'approximation d'ordre 2, 4, 6, ...

Par exemple, une approximation d'ordre 4 de $f'''(x_0)$ est :

$$f'''(x_0) = \frac{1}{8h^3}\,[-13\,(f(x_0+h) - f(x_0-h)) + 8\,(f(x_0+2h) - f(x_0-2h)) - (f(x_0+3h) - f(x_0-3h))] + O(h^4)$$

On constate donc sur les exemples précédents que les formules sont de la forme :

$$f^{(n)}(x_0) = \frac{1}{A_0\,h^n} \sum_{k=1}^{m} A_k(f(x_0+kh) - f(x_0-kh)) + O(h^p)$$

Dans cette expression, m est un nombre entier et A_0, A_1, ..., A_m sont des constantes.

Il est toujours possible d'obtenir une formule de ce type pour le calcul de dérivées d'ordre impair.

Le nombre de termes m de la somme est alors lié à l'ordre n de la dérivée ainsi qu'à l'ordre p de l'approximation par la relation :

$$m = \frac{n+p-1}{2}$$

Ce résultat est conforme à l'intuition : l'approximation est d'autant plus précise que le nombre de termes de la formule est important.

Pour le calcul des dérivées d'ordre pair, les formules sont du type :

$$f^{(n)}(x_0) = \frac{1}{A_0\,h^n} \sum_{k=1}^{m} A_k[f(x_0+kh) + f(x_0-kh) - 2f(x_0)] + O(h^p)$$

Cette fois, le nombre de termes de la somme est lié à l'ordre n de la dérivation et à l'ordre p de l'approximation par la relation :

$$m = \frac{n+p-2}{2}$$

b. Calcul des coefficients des formules d'approximation

L'élaboration de ces formules repose sur l'utilisation de la formule de Taylor :

$$f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + \ldots + \frac{h^r}{r!}f^{(r)}(x_0) + O(h^{r+1})$$

que l'on peut également écrire sous forme condensée :

$$f(x_0+h) = f(x_0) + \sum_{i=1}^{r} \frac{h^i}{i!} f^{(i)}(x_0) + O(h^{r+1})$$

• **Cas des dérivées d'ordre impair**

On choisit alors $r = 2m$ et on effectue la différence :

$$f(x_0 + h) - f(x_0 - h) = \sum_{i=1}^{m} 2 \frac{h^{2i-1}}{(2i-1)!} f^{(2i-1)}(x_0) + O(h^{2m+1})$$

(Les puissances paires se détruisent dans la somme.)

On obtient de la même manière la relation plus générale :

$$f(x_0 + kh) - f(x_0 - kh) = \sum_{i=1}^{m} 2k^{2i-1} \frac{h^{2i-1}}{(2i-1)!} f^{(2i-1)}(x_0) + O(h^{2m+1})$$

En écrivant cette relation m fois avec k variant de 1 à m, on obtient un système de m équations à m inconnues :

$f'(x_0), f'''(x_0), \ldots, f^{(2m-1)}(x_0)$

Ce système peut s'écrire :

$$\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 2 & 2^3 & 2^5 & \ldots & 2^{2m-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ m & m^3 & m^5 & \ldots & m^{2m-1} \end{pmatrix} \begin{pmatrix} h f'(x_0) \\ \frac{h^3}{3!} f'''(x_0) \\ \ldots \\ \frac{h^{2m-1}}{(2m-1)!} f^{(2m-1)}(x_0) \end{pmatrix} = \frac{1}{2} \begin{pmatrix} f(x_0 + h) - f(x_0 - h) \\ f(x_0 + 2h) - f(x_0 - 2h) \\ \ldots \\ f(x_0 + mh) - f(x_0 - mh) \end{pmatrix}$$

Il suffit alors d'inverser la matrice et de résoudre le système pour obtenir m formules donnant chacune l'expression d'une dérivée $f^{(n)}(x_0)$.

REMARQUE : Les termes négligés dans le développement de Taylor sont d'ordre $2m+1$; la formule obtenue pour $f^{(n)}(x_0)$ est donc d'ordre $p = (2m+1) - n$. On retrouve ainsi le résultat $m = \dfrac{n + p - 1}{2}$

• **Cas des dérivées d'ordre pair**

Le calcul est semblable au précédent.

On choisit maintenant $r = 2m+1$.

On obtient alors :

$$f(x_0 + h) + f(x_0 - h) - 2f(x_0) = \sum_{i=1}^{m} 2 \frac{h^{2i}}{(2i)!} f^{(2i)}(x_0) + O(h^{2m+2})$$

(Les puissances impaires se détruisent dans la somme.)

$$f(x_0+kh)+f(x_0-kh)-2f(x_0)=\sum_{i=1}^{m} 2k^{2i}\,\frac{h^{2i}}{(2i)!}\,f^{(2i)}(x_0)+0(h^{2m+2})$$

On obtient également un système de m équations à m inconnues en écrivant m fois cette relation k variant de 1 à m :

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 2^2 & 2^4 & 2^6 & \dots & 2^{2m} \\ \dots & \dots & \dots & \dots & \dots \\ m^2 & m^4 & m^6 & \dots & m^{2m} \end{pmatrix} \begin{pmatrix} \frac{h^2}{2!}f''(x_0) \\ \frac{h^4}{4!}f^{(4)}(x_0) \\ \dots \\ \frac{h^{2m}}{(2m)!}f^{(2m)}(x_0) \end{pmatrix} = \frac{1}{2}\begin{pmatrix} f(x_0+h)+f(x_0-h)-2f(x_0) \\ f(x_0+2h)+f(x_0-2h)-2f(x_0) \\ \dots \\ f(x_0+mh)+f(x_0-mh)-2f(x_0) \end{pmatrix}$$

La résolution de ce système conduit à m formules donnant chacune une expression approchée d'une dérivée $f^{(n)}(x_0)$.

REMARQUE : Les termes négligés dans le développement de Taylor étaient d'ordre 2m+2. La formule obtenue pour $f^{(n)}(x_0)$ est donc d'ordre $p = (2m+2) - n$.

Par exemple, la formule donnant $f''(x_0)$ sera d'ordre 2m.

On retrouve bien le résultat :

$$m = \frac{p+n-2}{2}$$

c. Programme

Les coefficients des formules d'approximation que nous venons d'étudier ne peuvent pas être calculés simplement et rapidement par le programme.

Nous nous sommes donc limités aux dérivées d'ordre 5, et nous avons inclus au programme les valeurs des coefficients des formules d'ordre 6.

La valeur adoptée pour h est 0,1 : il s'agit d'un compromis résultant de nombreux essais et assurant une précision acceptable pour un grand nombre de fonctions.

Quoi qu'il en soit, le calcul reste relativement imprécis et cette imprécision augmente avec l'ordre de la dérivation ; elle peut devenir inacceptable au-delà de l'ordre 5.

La fonction à dériver doit être définie à la ligne 5Ø du programme :

```
5Ø DEF FN F(X) = 1/(1-X)
```

Il faut ensuite introduire la valeur x_0 en laquelle on désire calculer la dérivée ainsi que l'ordre n de la dérivée.

```
50  DEF  FN F(X) = 1 / (1 - X)
1000 CLS
1010  PRINT "CALCUL DE LA DERIVEE N-IEME DE F EN XO": PRINT
    : PRINT
1020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
1030  PRINT "2- CALCUL DE DERIVEE N-IEME": PRINT : PRINT

1040  INPUT "VOTRE CHOIX:";E
1050  IF E = 1 THEN  LIST 50: END
1060  INPUT "XO=";X
1070  INPUT "N=";N: IF N > 5 THEN 1070
1080  GOSUB 2500
1090  PRINT : PRINT "LA DERIVEE D'ORDRE ";N;" EN ";X;"
    VAUT:"
1100  PRINT : PRINT  TAB( 5);T
1110  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
1120  IF Z$ = "O" THEN 1000
1130  END
2500 H = .1
2510  RESTORE : FOR K = 1 TO N: FOR J = 0 TO 6: READ U(
    J): NEXT J: NEXT K
2520 T = 0:P = ( - 1) ^ N
2530  FOR K = 1 TO U(0)
2540 T = T + U(K) * (( FN F(X + K * H) -  FN F(X)) + P *
    ( FN F(X - K * H) -  FN F(X)))
2550  NEXT K
2560 T = T / U(6) / (H ^ N)
2570  RETURN
2600  DATA   3,45,-9,1,0,0,60
2610  DATA   3,270,-27,2,0,0,180
2620  DATA   4,-488,338,-72,7,0,240
2630  DATA   4,-1952,676,-96,7,0,240
2640  DATA   5,1938,-1872,783,-152,13,288
```

d. Exemples commentés

- ***Exemple 1*** :

```
50 DEF FN F(X)=EXP(X)


CALCUL DE LA DERIVEE N-IEME DE F EN XO
```

```
XO=0
N=1

LA DERIVEE D'ORDRE 1 EN 0 VAUT:

    1.00000001

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
CALCUL DE LA DERIVEE N-IEME DE F EN XO


XO=0
N=2

LA DERIVEE D'ORDRE 2 EN 0 VAUT:

    1.00000002

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
CALCUL DE LA DERIVEE N-IEME DE F EN XO


XO=0
N=3

LA DERIVEE D'ORDRE 3 EN 0 VAUT:

    .999999918

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
CALCUL DE LA DERIVEE N-IEME DE F EN XO


XO=0
N=4

LA DERIVEE D'ORDRE 4 EN 0 VAUT:

    .9999905

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
CALCUL DE LA DERIVEE N-IEME DE F EN XO
```

```
XO=0
N=5

LA DERIVEE D'ORDRE 5 EN 0 VAUT:

    1.00007019

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Toutes les dérivées successives de e^x en 0 sont égales à 1. On constate que l'erreur sur le résultat est d'autant plus importante que l'ordre de la dérivation est élevé ; ce résultat est général.

- ***Exemple 2*** :

```
 50 DEF FN F(X)=LOG(X)


CALCUL DE LA DERIVEE N-IEME DE F EN XO


XO=1
N=2

LA DERIVEE D'ORDRE 2 EN 1 VAUT:

    -1.00001003

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
CALCUL DE LA DERIVEE N-IEME DE F EN XO


XO=1
N=5

LA DERIVEE D'ORDRE 5 EN 1 VAUT:

    24.1360323

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La dérivée seconde de Log x en 1 vaut (—1) : la précision est acceptable.

La dérivée cinquième de Log x en 1 vaut 24 : la précision n'est déjà plus que de 5 pour 1 000.

Remarquons cependant que si la précision n'est pas très bonne pour les dérivées d'ordre 4 et 5, les calculs de dérivées d'ordre 1 ou 2 (les plus rencontrés en pratique) sont eux suffisamment précis.

e. Erreur de méthode et erreurs numériques

• Erreur de méthode

Les formules d'approximation utilisées par le programme sont d'ordre 6 ; on pourrait penser que les erreurs sont de l'ordre de 10^{-6}.

En fait, une approximation en h^6 signifie que l'erreur est de la forme $k.h^6$ où k est un paramètre dépendant de la fonction. Les valeurs prises par k peuvent être faibles comme elles peuvent être très importantes. Ainsi, avec $h = 0,1$, si k est de l'ordre de 1 000, l'erreur sur le résultat est 10^{-3} (ceci se produit pour les dérivées d'ordre élevé).

Toujours avec $k = 1\ 000$, si l'on choisit $h = 0,5$, l'erreur est voisine de 15 ! Le résultat n'a alors aucune signification.

Dans le but de minimiser cette erreur, on peut décider d'adopter des valeurs de h les plus faibles possibles, mais on se heurte alors à une deuxième cause d'erreur, les erreurs numériques.

• Erreurs numériques

Lorsqu'on effectue la différence de deux nombres très voisins (par exemple 125,000002 et 124,999998), le résultat est entaché d'une erreur relative très importante. Ce phénomène est connu sous le nom de *différences évanescentes*, et provient du fait que l'ordinateur ne mémorise pas suffisamment de chiffres significatifs.

Dans le cas du calcul de dérivées, ce phénomène est d'autant plus sensible que la valeur de h est faible.

Par exemple, si l'on choisit $h = 0,01$ lors d'un calcul de dérivée d'ordre 5, les chiffres significatifs du résultat sont dans les calculs intermédiaires de l'ordre de 10^{-10}; ainsi, avec un ordinateur ne calculant qu'avec 8 ou 9 chiffres après la virgule, le résultat fourni par le programme est complètement faux.

• Il est cependant possible de jouer sur la valeur de h pour augmenter la précision (la valeur de h est fixée à la ligne 25ØØ du programme).

Pour certaines fonctions, telle ln(x) en $x_0 = 1$, la diminution de h ($h = 0,05$) conduit à de meilleurs résultats.

Par contre, avec d'autres fonctions comme les fonctions polynômes, le résultat est plus proche de la valeur exacte si on augmente h (par exemple $h = 0,5$).

Dans le cas de dérivées d'ordre 1 et 2, il est possible d'adopter $h = 0{,}01$ ou même $h = 0{,}001$ mais l'utilisation du programme doit se faire avec prudence : par exemple, si un premier calcul avec $h = 0{,}1$ fournit un résultat proche d'une valeur remarquable, on pourra reprendre le calcul avec une valeur de h plus faible dans le but de confirmer la présomption.

3. Développement limité d'une fonction en un point

a. Principe

Le développement limité à l'ordre n d'une fonction f autour d'un point x_0 est donné par la formule de Taylor :

$$f(x) = f(x_0) + (x-x_0)\frac{f'(x_0)}{1!} + (x-x_0)^2\frac{f''(x_0)}{2!} + (x-x_0)^3\frac{f'''(x_0)}{3!} + \ldots (x-x_0)^n\frac{f^{(n)}(x_0)}{n!} + O((x-x_0)^{n+1})$$

ou encore :

$$f(x) = a_0 + a_1(x-x_0) + a_2(x-x_0)^2 + a_3(x-x_0)^3 + \ldots + a_n(x-x_0)^n + O((x-x_0)^{n+1})$$

avec :

$$a_i = \frac{f^{(i)}(x_0)}{i!}$$

Dans le cas particulier où $x_0 = 0$, la formule devient :

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_nx^n + O(x^{n+1})$$

b. Programme

Le programme calcule les coefficients a_i du développement limité d'une fonction f en un point x_0 quelconque.

Pour évaluer les dérivées successives de la fonction en x_0, le programme précédent est utilisé, ce qui explique que l'on obtienne des développements limités d'un ordre au plus égal à 5.

Il faut encore programmer la fonction à la ligne 5Ø du programme:

5Ø DEF FN F(X) = 1/(1–X)

On introduit ensuite le point x_0 et l'ordre n du développement limité.

```
50  DEF  FN F(X) = 1 / (1 - X)
2000 CLS
2010  PRINT "CALCUL DU DEVELOPPEMENT LIMITE EN XO": PRINT
    : PRINT
2020  PRINT "1- PROGRAMMATION DE LA FONCTION": PRINT
2030  PRINT "2- CALCUL DU DEVELOPPEMENT": PRINT : PRINT

2040  INPUT "VOTRE CHOIX:";E
2050  IF E = 1 THEN  LIST 50: END
2060  INPUT "XO=";X
2070  INPUT "N=";M: IF M > 5 THEN 2070
2080  PRINT : PRINT "AO="; FN F(X)
2090  FOR N = 1 TO M
2100  GOSUB 2500
2110  FOR K = 1 TO N:T = T / K: NEXT K
2120  PRINT "A";N;"=";T
2130  NEXT N
2140  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
2150  IF Z$ = "O" THEN 2000
2160  END
2500 H = .1
2510  RESTORE : FOR K = 1 TO N: FOR J = 0 TO 6: READ U(
    J): NEXT J: NEXT K
2520 T = 0:P = ( - 1) ^ N
2530  FOR K = 1 TO U(0)
2540 T = T + U(K) * (( FN F(X + K * H) -  FN F(X)) + P *
    ( FN F(X - K * H) -  FN F(X)))
2550  NEXT K
2560 T = T / U(6) / (H ^ N)
2570  RETURN
2600  DATA   3,45,-9,1,0,0,60
2610  DATA   3,270,-27,2,0,0,180
2620  DATA   4,-488,338,-72,7,0,240
2630  DATA   4,-1952,676,-96,7,0,240
2640  DATA   5,1938,-1872,783,-152,13,288
```

(Ce programme comporte une partie commune avec le précédent: les lignes 25ØØ à 264Ø. Dans le cas d'une utilisation simultanée, il n'est pas nécessaire d'entrer deux fois ces lignes.)

c. Exemples commentés

• ***Exemple 1*** :

Le développement limité à l'ordre 5 en $x_0 = 0$ de la fonction $\frac{1}{1-x}$ est :

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + x^4 + x^5 + 0(x^6)$$

```
 50 DEF FN F(X)=1/(1-X)

CALCUL DU DEVELOPPEMENT LIMITE EN XO

XO=0
N=5

AO=1
A1=1.00004163
A2=1.00004161
A3=1.00112078
A4=1.00112157
A5=1.01364753

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Nous retrouvons ici encore que les coefficients des termes de plus haut degré sont entachés de l'erreur la plus importante.

• ***Exemple 2*** :

Le développement limité à l'ordre 5 en $x_0 = 1$ de la fonction ln x est :

$$\ln x = (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \frac{1}{4}(x-1)^4 + \frac{1}{5}(x-1)^5 + 0((x-1)^6)$$

```
50 DEF FN F(X)=LOG(X)


CALCUL DU DEVELOPPEMENT LIMITE EN XO


XO=1
N=5

AO=0
A1=1.00000576
A2=-.500005017
A3=.33345057
A4=-.250107246
A5=.201133603

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

L'erreur reste toujours inférieure à 1 %.

d. Conclusion

Il apparaît clairement que la méthode directe que nous venons d'étudier ne permet pas le calcul du développement limité d'une fonction quelconque de manière précise.

Nous sommes de plus limités à l'ordre 5.

Les trois programmes suivants vont nous affranchir partiellement de ces limitations.

4. Développement limité de f.g

a. Introduction

Les développements limités que nous calculerons désormais seront des développements limités en $x_0 = 0$.

Les fonctions usuelles admettent des développements limités connus (donnés en annexe).

Nous allons donc utiliser ces résultats pour trouver les développements limités d'autres fonctions.

Les trois programmes suivants vont nous permettre de calculer précisément et à un ordre élevé les développements limités des fonctions f.g, f/g et g o f, connaissant les développements limités de f et g.

b. Principe

Nous étudions ici le cas du produit de deux fonctions f et g.

Supposons connus les développements limités de f et g à l'ordre n:

$$f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n + O(x^{n+1})$$
$$= P(x) + O(x^{n+1})$$
$$g(x) = b_0 + b_1x + b_2x^2 + \ldots + b_nx^n + O(x^{n+1})$$
$$= Q(x) + O(x^{n+1})$$

Alors, le développement limité à l'ordre n de la fonction f.g est égal à:

$$(f.g)(x) = (a_0 + a_1x + \ldots + a_nx^n) \cdot (b_0 + b_1x + \ldots + b_nx^n) + O(x^{n+1})$$
$$= c_0 + c_1x + \ldots + c_nx^n + O(x^{n+1})$$

Les coefficients c_i peuvent être calculés en multipliant le polynôme P par le polynôme Q:

$$c_i = \sum_{k=0}^{i} a_k \cdot b_{i-k}$$

c. Programme

Il faut introduire les coefficients des développements limités de f et g (il y a deux façons d'introduire les coefficients; nous les détaillerons dans les exemples commentés).

Le programme calcule ensuite les coefficients du développement limité de f.g.

```
10  DIM A(100),B(100),C(100)
3000 CLS
3010  PRINT  TAB( 5);"DEVELOPPEMENT LIMITE DE F.G": PRINT
    : PRINT
3020  INPUT "ORDRE DU D.L.:";N
3030  PRINT : PRINT "COEFFICIENTS ENTRES MANUELLEMENT:
    M"
3040  INPUT "COEFFICIENTS CALCULES (EN 8000) : C ";Y$
3050  IF Y$ = "C" THEN  FOR I = 0 TO N: GOSUB 8000: NEXT
    I: GOTO 3100
```

```
3060  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
      F:"
3070  FOR I = 0 TO N: PRINT "A";I;: INPUT "=";A(I): NEXT
     I
3080  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
      G:"
3090  FOR I = 0 TO N: PRINT "B";I;: INPUT "=";B(I): NEXT
     I
3100  PRINT : INPUT "DESIREZ-VOUS LES APPROXIMATIONS?:"
     ;Y$
3110  FOR M = 0 TO N
3120 T = 0
3130  FOR I = 0 TO M:T = T + A(I) * B(M - I): NEXT I
3140  PRINT "C";M;"=";T;
3150  IF Y$ = "O" THEN X = T: GOSUB 9000: PRINT "  :";P
     ;"/";Q;
3160  PRINT " "
3170  NEXT M
3180  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
     E?:";Z$
3190  IF Z$ = "O" THEN 3000
3200  END
8000 A(I) = 1
8010  FOR H = 1 TO I:A(I) = A(I) / H: NEXT H
8020 B(I) = ( - 1) ^ I
8030  RETURN
9000 AB = X:AC =  INT (X):P = AC
9010 Q = 1:AD = 1:AF = 0: GOTO 9050
9020 AB = 1 / (AB - AC):AC =  INT (AB)
9030 AY = AC * P + AD:AD = P:P = AY
9040 AY = AC * Q + AF:AF = Q:Q = AY
9050  IF  ABS ((X - P / Q)) > 1E - 7 THEN 9020
9060  RETURN
```

d. Exemples commentés

● *Exemple 1* :

Soit à calculer le développement limité de la fonction:

$$h(x) = \frac{e^x}{1 + x}$$

en la considérant comme le produit de:

$$f(x) = e^x \qquad \text{par} \qquad g(x) = \frac{1}{1 + x}$$

dont les développements limités sont:

$$f(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!} + O(x^{n+1})$$

$$g(x) = 1 - x + x^2 - x^3 + \ldots + (-1)^n x^n + O(x^{n+1})$$

Le programme demande tout d'abord l'ordre du développement à calculer; répondons 5.

Nous avons ensuite le choix entre:

→ introduire les coefficients manuellement,

→ les faire calculer par le programme.

Répondons "M" pour l'instant; nous examinerons le calcul automatique des coefficients ultérieurement.

Il faut alors introduire les six coefficients du développement de e^x, puis les six coefficients du développement de $\frac{1}{1+x}$.

A la dernière question "Désirez-vous les approximations?", répondons "N" pour l'instant.

Le programme affiche alors les six coefficients du développement de h(x).

```
DEVELOPPEMENT LIMITE DE F.G
ORDRE DU D.L.:5

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C M

ENTREZ LES COEFFICIENTS DU D.L. DE F:
A0=1
A1=1
A2=.5
A3=.16666666
A4=.04166666
A5=.00833333

ENTREZ LES COEFFICIENTS DU D.L. DE G:
B0=1
B1=-1
B2=1
B3=-1
B4=1
B5=-1
```

```
DESIREZ-VOUS LES APPROXIMATIONS?:N
C0=1
C1=0
C2=.5
C3=-.33333334
C4=.375
C5=-.36666667

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

Remarquons que s'il paraît évident que les quatre premiers coefficients sont égaux à 1, 0, $\frac{1}{2}$ et $-\frac{1}{3}$, en revanche il est plus difficile de reconnaître la forme rationnelle des deux derniers termes (qui sont égaux à $\frac{3}{8}$ et $-\frac{11}{30}$ respectivement).

De plus, la plupart des développements limités obtenus soit par produit, soit par quotient, soit par composée de fonctions ont des coefficients rationnels.

C'est pourquoi nous avons ajouté le sous-programme 9ØØØ, qui recherche à partir d'un nombre décimal la fraction rationnelle irréductible s'en rapprochant à 10^{-7} près.

Reprenons l'exécution en répondant "O" à la question "Désirez-vous les approximations?".

```
    DEVELOPPEMENT LIMITE DE F.G


ORDRE DU D.L.:5

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C M

ENTREZ LES COEFFICIENTS DU D.L. DE F:
A0=1
A1=1
A2=.5
A3=.16666666
A4=.04166666
A5=.00833333

ENTREZ LES COEFFICIENTS DU D.L. DE G:
B0=1
```

```
B1=-1
B2=1
B3=-1
B4=1
B5=-1

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1   :1/1
C1=0   :0/1
C2=.5  :1/2
C3=-.33333334  :-1/3
C4=.375  :3/8
C5=-.36666667  :-11/30

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le programme fournit donc en plus des valeurs décimales les approximations rationnelles, qui ici coïncident avec les valeurs exactes.

Intéressons-nous maintenant au calcul automatique des coefficients.

Pour les fonctions usuelles, le coefficient a_i du développement s'exprime en fonction de i. Dans l'exemple, nous avons:

$$a_i = \frac{1}{i!} \qquad \text{et} \qquad b_i = (-1)^i.$$

Nous allons donc programmer ces formules entre les lignes 8ØØØ et 8Ø3Ø, et l'option "coefficients calculés" se chargera d'évaluer et de stocker tous les coefficients nécessaires grâce à ces formules.

Dans l'exemple, les lignes 8ØØØ et 8Ø1Ø calculent $\frac{1}{i!}$, la ligne 8Ø2Ø calcule $(-1)^i$ et la ligne 8Ø3Ø (à ne pas oublier) termine le sous-programme.

```
8000 A(I)=1

8010 FOR H=1 TO I:A(I)=A(I)/H:NEXT H

8020 B(I)=(-1)^I

8030 RETURN
```

```
   DEVELOPPEMENT LIMITE DE F.G
```

```
ORDRE DU D.L.:5

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1   :1/1
C1=0   :0/1
C2=.5  :1/2
C3=-.333333333  :-1/3
C4=.375  :3/8
C5=-.366666667  :-11/30
```

On retrouve bien les mêmes résultats.

Un examen des résultats sous leur forme décimale montre que ceux-ci sont exacts; la précision est donc bien meilleure que celle obtenue avec le calcul direct.

Nous allons donc pouvoir chercher des développements à un ordre supérieur à 5.

Demandons cette fois un calcul à l'ordre 15:

```
    DEVELOPPEMENT LIMITE DE F.G


ORDRE DU D.L.:15

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1   :1/1
C1=0   :0/1
C2=.5  :1/2
C3=-.333333333  :-1/3
C4=.375  :3/8
C5=-.366666667  :-11/30
C6=.368055556  :53/144
C7=-.367857143  :-103/280
C8=.367881945  :898/2441
C9=-.367879189  :-536/1457
C10=.367879464  :1001/2721
C11=-.367879439  :-1001/2721
```

```
C12=.367879441   :1001/2721
C13=-.367879441  :-1001/2721
C14=.367879441   :1001/2721
C15=-.367879441  :-1001/2721

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La précision est maximale pour tous les coefficients (10^{-9}), ce qui confirme la possibilité d'obtenir des développements à un ordre très élevé.

Cet ordre est limité à 1ØØ par le programme, mais cela ne constitue pas une limitation en pratique.

Le développement exact à l'ordre 8 de la fonction $\frac{e^x}{1+x}$ est

$$\frac{e^x}{1+x} = 1 + \frac{x^2}{2} - \frac{x^3}{3} + \frac{3}{8}x^4 - \frac{11}{30}x^5 + \frac{53}{144}x^6 - \frac{103}{280}x^7 + \frac{2119}{5760}x^8$$

Remarquons que les approximations rationnelles coïncident avec les valeurs exactes jusqu'au terme d'ordre 7.

5. *Développement limité de f/g*

a. Principe

Supposons connus les développements de f et g à l'ordre n :

$$\begin{aligned} f(x) &= a_0 + a_1x + a_2x^2 + \ldots + a_nx^n + O(x^{n+1}) \\ &= P(x) + O(x^{n+1}) \\ g(x) &= b_0 + b_1x + b_2x^2 + \ldots + b_nx^n + O(x^{n+1}) \\ &= Q(x) + O(x^{n+1}) \end{aligned}$$

avec $b_0 \neq 0$, puisque dans le cas contraire, la fonction f/g n'est pas définie en $x = 0$.

Le développement limité de la fonction f/g est alors obtenu en effectuant la division suivant les puissances croissantes à l'ordre n du polynôme P par le polynôme Q.

On peut montrer que les coefficients c_i satisfont la relation de récurrence :

$$c_0 = \frac{a_0}{b_0}$$

$$c_i = a_i - \sum_{k=1}^{i} b_k \cdot c_{i-k}$$

b. Programme

La structure générale du programme est semblable à celle du programme précédent, seule la partie de calcul est modifiée.

L'utilisation est donc identique.

```
10  DIM A(100),B(100),C(100)
4000 CLS
4010  PRINT  TAB( 5);"DEVELOPPEMENT LIMITE DE F/G": PRINT
    : PRINT
4020  INPUT "ORDRE DU D.L.:";N
4030  PRINT : PRINT "COEFFICIENTS ENTRES MANUELLEMENT:
    M"
4040  INPUT "COEFFICIENTS CALCULES (EN 8000) : C ";Y$
4050  IF Y$ = "M" THEN 4090
4060  FOR I = 0 TO N: GOSUB 8000: NEXT I
4070  IF B(0) = 0 THEN  PRINT "B0 NE DOIT PAS ETRE NUL"
    : GOTO 4250
4080  GOTO 4140
4090  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
     F:"
4100  FOR I = 0 TO N: PRINT "A";I;: INPUT "=";A(I): NEXT
    I
4110  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
     G:"
4120  INPUT "B0=";B(0): IF B(0) = 0 THEN  PRINT "B0 NE
    DOIT PAS ETRE NUL": GOTO 4120
4130  FOR I = 1 TO N: PRINT "B";I;: INPUT "=";B(I): NEXT
    I
4140  PRINT : INPUT "DESIREZ-VOUS LES APPROXIMATIONS?:"
    ;Y$
4150 C(0) = A(0) / B(0): PRINT "C0=";C(0);
4160  IF Y$ = "O" THEN X = C(0): GOSUB 9000: PRINT "  :
    ";P;"/";Q;
4170  PRINT " "
4180  FOR I = 1 TO N
4190 C(I) = A(I)
4200  FOR K = 1 TO I:C(I) = C(I) - B(K) * C(I - K): NEXT
    K
```

```
4210  PRINT "C";I;"=";C(I);
4220  IF Y$ = "O" THEN X = C(I): GOSUB 9000: PRINT "  :
    ";P;"/";Q;
4230  PRINT " "
4240  NEXT I
4250  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
4260  IF Z$ = "O" THEN 4000
4270  END
8000 A(I) = O
8010 M =  INT (I / 2)
8020  IF I = 2 * M THEN 8050
8030 A(I) = ( - 1) ^ M
8040  FOR H = 1 TO I:A(I) = A(I) / H: NEXT H
8050 B(I) = O
8060  IF I <  > 2 * M THEN 8090
8070 B(I) = ( - 1) ^ M
8080  FOR H = 1 TO I:B(I) = B(I) / H: NEXT H
8090  RETURN
9000 AB = X:AC =  INT (X):P = AC
9010 Q = 1:AD = 1:AF = O: GOTO 9050
9020 AB = 1 / (AB - AC):AC =  INT (AB)
9030 AY = AC * P + AD:AD = P:P = AY
9040 AY = AC * Q + AF:AF = Q:Q = AY
9050  IF  ABS ((X - P / Q)) > 1E - 7 THEN 9020
9060  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

Nous avons repris la fonction :

$$h(x) = \frac{e^x}{1 + x}$$

en la considérant cette fois comme le rapport des fonctions :

$f(x) = e^x$ et $g(x) = 1 + x$

Les lignes 8ØØØ à 8Ø3Ø définissent les coefficients des développements de f et g.

```
8000 A(I)=1

8010 FOR H=1 TO I:A(I)=A(I)/H:NEXT H

8020 B(I)=0:B(0)=1:B(1)=1

8030 RETURN
```

```
   DEVELOPPEMENT LIMITE DE F/G


ORDRE DU D.L.:6

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1   :1/1
C1=0   :0/1
C2=.5  :1/2
C3=-.333333333  :-1/3
C4=.375  :3/8
C5=-.366666667  :-11/30
C6=.368055556  :53/144

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les résultats sont identiques à ceux fournis par le programme précédent: la précision est ici encore maximale, et les approximations rationnelles sont les valeurs exactes.

- ***Exemple 2*** :

Soit à calculer le développement à l'ordre 6 de la fonction:

$$h(x) = \frac{1}{1 + shx}$$

```
8000 A(I)=0:A(0)=1

8010 B(I)=0

3020 M=INT(I/2)
```

```
8030 IF I=2*M  THEN 8060

8040 B(I)=1

8050 FOR H=1 TO I:B(I)=B(I)/H:NEXT H

8060 B(0)=1

8070 RETURN
```

```
   DEVELOPPEMENT LIMITE DE F/G


ORDRE DU D.L.:6

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1   :1/1
C1=-1  :-1/1
C2=1   :1/1
C3=-1.16666667  :-7/6
C4=1.33333333   :4/3
C5=-1.50833333  :-181/120
C6=1.71111111   :77/45

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les approximations sont ici encore exactes.

• ***Exemple 3*** :

Calculons le développement de tg x, en posant :

$f(x) = \sin x$ et $g(x) = \cos x$

```
8000 A(I)=0

8010 M=INT(I/2)

8020 IF I=2*M THEN 8050

8030 A(I)=(-1)^M
```

```
8040 FOR H=1 TO I:A(I)=A(I)/H:NEXT H

8050 B(I)=0

8060 IF I<>2*M THEN 8090

8070 B(I)=(-1)^M

8080 FOR H=1 TO I:B(I)=B(I)/H:NEXT H

8090 RETURN
```

```
   DEVELOPPEMENT LIMITE DE F/G


ORDRE DU D.L.:10

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=0  :0/1
C1=1  :1/1
C2=0  :0/1
C3=.333333333  :1/3
C4=0  :0/1
C5=.133333333  :2/15
C6=0  :0/1
C7=.053968254  :17/315
C8=0  :0/1
C9=.0218694885  :62/2835
C10=0  :0/1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le développement obtenu, à l'ordre 10, est exact.

6. *Développement limité de gof*

a. Principe

Supposons connus à l'ordre n les développements de g et f :

$$g(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n + 0(x^{n+1})$$
$$= P(x) + 0(x^{n+1})$$
$$f(x) = 0 + b_1x + b_2x^2 + \ldots + b_nx^n + 0(x^{n+1})$$
$$= Q(x) + 0(x^{n+1})$$

Le terme b_0 doit nécessairement être nul, pour que f(x) tende vers 0 lorsque x tend vers 0 ; autrement, on ne pourrait pas parler du développement de g o f en 0.

Le développement de g o f est alors :

$$(g \circ f)(x) = P[Q(x)] + 0(x^{n+1})$$

ou encore, sous une forme plus explicite :

$$(g \circ f)(x) = a_0 + a_1[b_1x + b_2x^2 + \ldots + b_nx^n]$$
$$+ a_2[b_1x + b_2x^2 + \ldots + b_nx^n]^2$$
$$+ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$+ a_n[b_1x + b_2x^2 + \ldots + b_nx^n]^n + 0(x^{n+1})$$

Ce développement contient des termes de degré 0 à n^2.

Pour obtenir le développement limité à l'ordre n, il convient de ne conserver dans ce développement que les termes de degré inférieur ou égal à n.

On montre que le coefficient en x^m d'un terme $[b_1x + b_2x^2 + \ldots + b_nx^n]^i$ est donné par :

$$\gamma_{i,m} = \sum_{\alpha_1,\alpha_2,\ldots\alpha_i} b_{\alpha_1} . b_{\alpha_2} \ldots b_{\alpha_i}$$

la sommation se faisant pour tous les i-uplets $(\alpha_1,\ldots\alpha_i)$ tels que :

$$\sum_{j=1}^{i} \alpha_j = m$$

En faisant la somme de ces coefficients partiels $\gamma_{i,m}$ pour toutes les valeurs de i, on obtient le coefficient du terme en x^m du développement de g o f :

$$c_m = \sum_{i=1}^{n} \gamma_{i,m}$$

b. Programme

Les calculs effectués par le programme sont beaucoup plus nombreux que ceux effectués par les deux programmes précédents.

De plus, dans le but d'améliorer la vitesse d'exécution, le programme stocke au début de son exécution les coefficients du binôme C_i^j, ce qui nécessite de l'espace mémoire.

Pour ces deux raisons, nous avons décidé d'adopter 20 comme valeur maximale de l'ordre du développement limité de g o f.

Le temps de calcul est d'environ une minute pour un développement de l'ordre 10.

L'utilisation du programme reste identique à celle des deux programmes précédents.

```
10  DIM A(100),B(100),C(100)
20  DIM D(20,20),ID(20)
5000 CLS
5010  PRINT  TAB( 5);"DEVELOPPEMENT LIMITE DE GOF": PRINT
    : PRINT
5020  INPUT "ORDRE DU D.L.:";N
5030  PRINT : PRINT "COEFFICIENTS ENTRES MANUELLEMENT:
    M"
5040  INPUT "COEFFICIENTS CALCULES (EN 8000) : C ";Y$
5050  IF Y$ = "M" THEN 5090
5060  FOR I = 0 TO N: GOSUB 8000: NEXT I
5070  IF B(0) <  > 0 THEN  PRINT "B0 DOIT ETRE NUL": GOTO
    5490
5080  GOTO 5140
5090  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
     G:"
5100  FOR I = 0 TO N: PRINT "A";I;: INPUT "=";A(I): NEXT
    I
5110  PRINT : PRINT "ENTREZ LES COEFFICIENTS DU D.L. DE
     F:"
5120  INPUT "B0=";B(0): IF B(0) <  > 0 THEN  PRINT "B0
    DOIT ETRE NUL": GOTO 5120
5130  FOR I = 1 TO N: PRINT "B";I;: INPUT "=";B(I): NEXT
    I
5140  PRINT : INPUT "DESIREZ-VOUS LES APPROXIMATIONS?:"
    ;Y$
5150  PRINT "C0=";A(0);: IF Y$ = "O" THEN X = A(0): GOSUB
    9000: PRINT " :";P;"/";Q;
5160  PRINT " "
```

```
5170 D(1,0) = 1:D(1,1) = 1
5180  FOR I = 2 TO N
5190 D(I,0) = 1:D(I,I) = 1
5200  FOR J = 1 TO I - 1:D(I,J) = D(I - 1,J - 1) + D(I -
    1,J): NEXT J
5210  NEXT I
5220  FOR M = 1 TO N
5230 T = A(1) * B(M)
5240  IF M = 1 THEN 5450
5250  FOR I = 2 TO M
5260 H = 0
5270  FOR J = 1 TO I - 1:ID(J) = 1: NEXT J
5280 ID(I) = M: FOR J = 1 TO I - 1:ID(I) = ID(I) - ID(J
    ): NEXT J
5290  IF ID(I) < ID(I - 1) THEN 5420
5300 L = B(ID(1)):P = 1:Q = I
5310  FOR K = 2 TO I
5320 L = L * B(ID(K))
5330  IF ID(K) = ID(K - 1) THEN P = P + 1: GOTO 5360
5340 L = L * D(Q,P)
5350 Q = Q - P:P = 1
5360  NEXT K
5370 H = H + L
5380  IF ID(I) - ID(I - 1) > 1 THEN ID(I - 1) = ID(I -
    1) + 1: GOTO 5280
5390  IF I = 2 THEN 5430
5400 K = 1
5410  IF ID(I - K - 1) < ID(I - K) THEN ID(I - K - 1) =
    ID(I - K - 1) + 1: FOR J = I - K TO I - 1:ID(J) = I
    D(J - 1): NEXT J: GOTO 5280
5420  IF K < I - 2 THEN K = K + 1: GOTO 5410
5430 T = T + H * A(I)
5440  NEXT I
5450  PRINT "C";M;"=";T;
5460  IF Y$ = "O" THEN X = T: GOSUB 9000: PRINT "  :";P
    ;"/";Q;
5470  PRINT " "
5480  NEXT M
5490  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
5500  IF Z$ = "O" THEN 5000
5510  END
8000 A(I) = ( - 1) ^ I
8010 B(I) = 0
8020 M =  INT (I / 2)
```

```
8030  IF I = 2 * M THEN 8060
8040 B(I) = 1
8050  FOR H = 1 TO I:B(I) = B(I) / H: NEXT H
8060  RETURN
9000 AB = X:AC =  INT (X):P = AC
9010 Q = 1:AD = 1:AF = 0: GOTO 9050
9020 AB = 1 / (AB - AC):AC =  INT (AB)
9030 AY = AC * P + AD:AD = P:P = AY
9040 AY = AC * Q + AF:AF = Q:Q = AY
9050  IF  ABS ((X - P / Q)) > 1E - 7 THEN 9020
9060  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

Soit à calculer le développement limité à l'ordre 10 de la fonction :

$h(x) = e^{\sin x}$

```
8000 A(I)=1

8010 FOR H=1 TO I:A(I)=A(I)/H:NEXT H

8020 B(I)=0

8030 M=INT(I/2)

8040 IF I=2*M THEN 8070

8050 B(I)=(-1)^M

8060 FOR H=1 TO I:B(I)=B(I)/H:NEXT H

8070 RETURN
```

```
   DEVELOPPEMENT LIMITE DE GOF


ORDRE DU D.L.:10

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C
```

```
DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1 :1/1
C1=1  :1/1
C2=.5  :1/2
C3=0  :0/1
C4=-.125  :-1/8
C5=-.0666666667  :-1/15
C6=-4.16666667E-03  :-1/240
C7=.0111111111  :1/90
C8=5.38194445E-03  :31/5760
C9=1.76366843E-04  :1/5670
C10=-8.1321649E-04  :-3/3689

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On peut vérifier par le calcul l'exactitude de chacun des coefficients.

Les approximations sont elles exactes jusqu'à l'ordre 9, le coefficient C_{10} étant :

$$C_{10} = -\frac{2\,951}{3\,628\,800}$$

- ***Exemple 2*** :

Nous avons ici cherché à retrouver les résultats du programme précédent concernant la fonction :

$$h(x) = \frac{1}{1 + \text{sh}x}$$

en la considérant cette fois comme la composée de :

$$f(x) = \text{sh}x$$

et de :

$$g(x) = \frac{1}{1 + x}$$

```
 8000 A(I)=(-1)^I

 8010 B(I)=0

 8020 M=INT(I/2)

 8030 IF I=2*M THEN 8060

 8040 B(I)=1
```

```
8050 FOR H=1 TO I:B(I)=B(I)/H:NEXT H

8060 RETURN


   DEVELOPPEMENT LIMITE DE GOF


ORDRE DU D.L.:6

COEFFICIENTS ENTRES MANUELLEMENT: M
COEFFICIENTS CALCULES (EN 8000) : C C

DESIREZ-VOUS LES APPROXIMATIONS?:O
C0=1 :1/1
C1=-1  :-1/1
C2=1  :1/1
C3=-1.16666667  :-7/6
C4=1.33333333  :4/3
C5=-1.50833333  :-181/120
C6=1.71111111  :77/45

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On retrouve effectivement les mêmes résultats.

7. Conclusion

Les trois derniers programmes permettent, comme nous l'avons vu, de calculer des développements limités d'ordre important avec une précision excellente.

On utilisera donc systématiquement l'un de ces trois programmes lorsque cela sera possible, et on aura recours au calcul direct uniquement si la fonction étudiée ne peut s'exprimer au moyen de fonctions plus simples.

6 Intégration

1. Introduction

Nous allons dans ce chapitre nous intéresser à différentes méthodes permettant le calcul de l'intégrale :

$$\int_a^b f(x)\,dx$$

où f est la fonction à intégrer, et [a,b] un intervalle donné du domaine de définition de f :

$$[a,b] \subset D_f$$

Nous commencerons par les formules de Cotes, qui consistent à découper l'intervalle d'intégration en sous-intervalles, et à substituer un polynôme à la fonction à intégrer sur chacun de ces sous-intervalles.

Nous verrons ensuite la méthode de Gauss, plus particulièrement adaptée à l'intégration des polynômes, puis la méthode de Romberg, qui permet d'obtenir la valeur de l'intégrale avec une précision donnée.

Nous terminerons par deux méthodes permettant l'intégration des fonctions discrètes (fonctions dont les valeurs ne sont connues qu'en certains points), la méthode des trapèzes et la méthode de Simpson.

Pour utiliser les sept programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu:

```
100 CLS
110  PRINT  TAB( 5);"CALCUL D'INTEGRALES": PRINT : PRINT

120  PRINT : PRINT "1- METHODE DES TRAPEZES"
130  PRINT : PRINT "2- METHODE DE SIMPSON"
140  PRINT : PRINT "3- METHODE DE VILLARCEAU"
150  PRINT : PRINT "4- METHODE DE GAUSS"
160  PRINT : PRINT "5- METHODE DE ROMBERG"
170  PRINT : PRINT "6- METHODE DISCRETE DES TRAPEZES"
180  PRINT : PRINT "7- METHODE DISCRETE DE SIMPSON"
200  PRINT : PRINT "8- FIN"
210  PRINT : INPUT "  VOTRE CHOIX:";E
220  ON E GOTO 1000,2000,3000,4000,5000,6000,7000,8000
230  GOTO 100
8000  END
```

L'ensemble nécessite environ 5 Koctets de mémoire.

2. *Méthode des trapèzes*

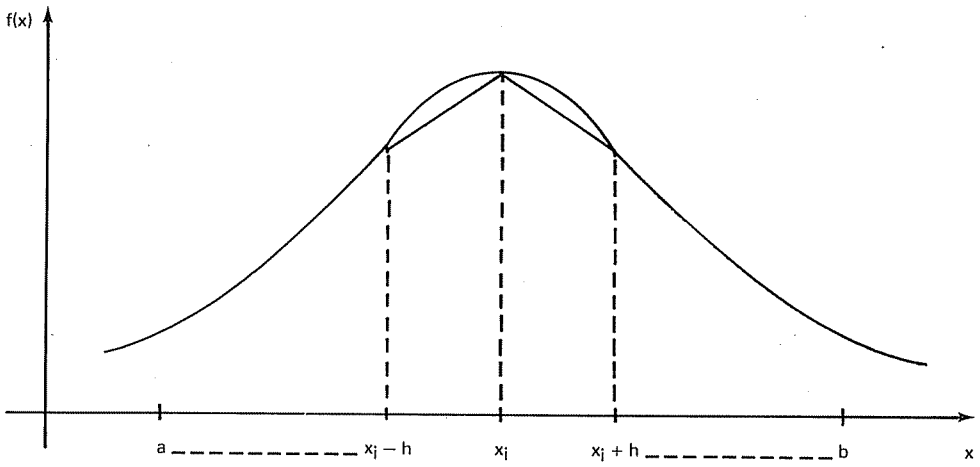
a. Principe

C'est la plus simple des méthodes de Cotes.

L'intervalle d'intégration [a,b] est découpé en n sous-intervalles de longueur:

$$h = \frac{b - a}{n}$$

Sur chacun de ces sous-intervalles, la courbe réelle est remplacée par un segment de droite.



Sur chaque intervalle $[x_i, x_i + h]$ on calcule :

$$T_i = h \ \frac{f(x_i) + f(x_i + h)}{2}$$

La valeur approchée de l'intégrale est alors :

$$I = \sum_{i=0}^{n-1} T_i$$

On démontre que l'approximation des trapèzes est d'ordre h^2, ce qui signifie que la différence entre la valeur exacte de l'intégrale et la valeur calculée est de l'ordre de grandeur de h^2.

Il est possible d'améliorer la précision en utilisant la formule :

$$I' = I + \frac{h^2}{12} (f'(a) - f'(b))$$

où I est la première approximation, f'(a) et f'(b) les valeurs de la fonction dérivée de f aux points a et b.

On démontre alors que la nouvelle approximation I' est d'ordre h^4.

b. Programme

Le programme calcule l'approximation d'ordre h^4 de l'intégrale.

Les dérivées sont évaluées grâce aux formules :

$$f'(a) = \frac{f(a + k) - f(a)}{k} \qquad \text{et} \qquad f'(b) = \frac{f(b) - f(b - k)}{k}$$

où :

$k = 0{,}001$

Cette approximation est tout à fait satisfaisante lorsque la longueur de l'intervalle (b—a) est de l'ordre de l'unité.

La fonction à intégrer doit être programmée à la ligne 5Ø :

```
5Ø DEF FN F(X) = X * X
```

Il faut ensuite introduire au programme les deux bornes a et b de l'intervalle d'intégration, puis le nombre n de pas d'intégration (nombre de sous-intervalles).

La valeur calculée est d'autant plus précise que n est grand.

```
50  DEF  FN F(X) = X ^ 14 - 8 * X ^ 11 + 3 * X ^ 3
1000 CLS
1010  PRINT  TAB( 5);"METHODE DES TRAPEZES": PRINT : PRINT

1020  INPUT "A=";A
1030  INPUT "B=";B
1040  INPUT "N=";N
1050 H = (B - A) / N
1060 T =  FN F(B)
1070 T = (T +  FN F(A)) / 2
1080 X = A
1090  FOR I = 1 TO N - 1
1100 X = X + H:T = T +  FN F(X)
1110  NEXT I
1120 K = .001
1130 D =  FN F(A + K) -  FN F(A)
1140 D = D -  FN F(B) +  FN F(B - K)
1150 D = D * H / K / 12
1160 T = (T + D) * H
1170  PRINT : PRINT "L'INTEGRALE VAUT:";T
1180  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
1190  IF Z$ = "O" THEN 1000
1200  END
```

c. Exemples commentés

• ***Exemple 1*** :

Soit à calculer l'intégrale :

$$\int_0^{1/2} \sqrt{1-x^2}\,dx$$

```
50 DEF FN F(X)=SQR(1-X*X)

   METHODE DES TRAPEZES


A=0
B=0.5
N=100

L'INTEGRALE VAUT:.478305736

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On obtient avec 100 pas d'intégration :

$I = 0{,}478305736$

La valeur exacte est :

$$\int_0^{1/2} \sqrt{1-x^2}\,dx = \frac{\pi}{12} + \frac{\sqrt{3}}{8} = 0{,}478305739$$

• ***Exemple 2*** :

Soit à calculer l'intégrale :

$$\int_1^2 (x^{14} - 8x^{11} + 3x^3)\,dx$$

```
50 DEF FN F(X)=X^14-8*X^11+3*X^3

   METHODE DES TRAPEZES
```

```
A=1
B=2
N=500

L'INTEGRALE VAUT:-534.283321

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La valeur exacte est:

$$\int_1^2 (x^{14} - 8x^{11} + 3x^3)\,dx = -\frac{32057}{60} = -534{,}2833...$$

Le résultat obtenu est assez précis, mais a été calculé avec 500 pas d'intégration, ce qui est déjà important, et nécessite un temps d'exécution relativement long (entre une et trois minutes selon l'ordinateur).

- ***Exemple 3***:

Soit à calculer:

$$\int_1^{12} x \sin^2 x\,dx$$

```
 50 DEF FN F(X)=X*SIN(X)^2


    METHODE DES TRAPEZES


A=1
B=12
N=100

L'INTEGRALE VAUT:38.5890128

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DES TRAPEZES


A=1
B=12
N=1000
```

```
L'INTEGRALE VAUT:38.5890188

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La valeur exacte est :

$$\int_1^{12} x \sin^2 x \, dx = \frac{1}{8} [288 - 24 \sin(24) - \cos(24) - 2 + 2 \sin(2) + \cos(2)]$$

$$= 38{,}5890187$$

La valeur obtenue avec 100 pas d'intégration est déjà très précise.

Pour obtenir la précision maximale, il faut cependant 1 000 pas d'intégration, ce qui nécessite un temps de calcul de plusieurs minutes.

3. *Méthode de Simpson*

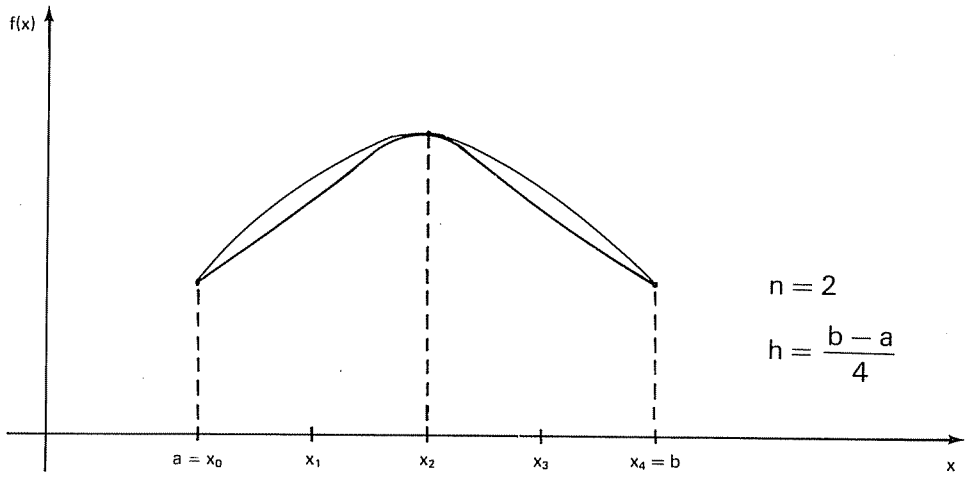
a. Principe

Sur chacun des n sous-intervalles de [a,b], la fonction f est remplacée par un arc de parabole.

On pose $h = \frac{b-a}{2n}$ et $x_i = a + ih$

Il y a donc $(2n + 1)$ points x_i, avec :

$x_0 = a$ et $x_{2n} = b$



On peut alors montrer que sur l'intervalle $[x_{2i}, x_{2i+2}]$, l'intégrale :

$$\int_{x_{2i}}^{x_{2i+2}} f(x)\,dx$$

est approchée par :

$$T_{2i} = \frac{h}{3}\left[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})\right]$$

L'intégrale I vaut alors :

$$I = T_0 + T_2 + T_4 + \ldots + T_{2(n-1)} = \sum_{i=0}^{n-1} T_{2i}$$

L'approximation obtenue est d'ordre h^4.

b. Programme

L'utilisation est identique à celle du programme précédent : la fonction doit être programmée à la ligne 5Ø, il faut introduire les bornes et le nombre de pas d'intégration.

```
50  DEF  FN F(X) = X ^ 14 - 8 * X ^ 11 + 3 * X ^ 3
2000 CLS
2010  PRINT  TAB( 5);"METHODE DE SIMPSON": PRINT : PRINT

2020  INPUT "A=";A
2030  INPUT "B=";B
2040  INPUT "N=";N
2050 H = (B - A) / 2 / N
2060 T = ( FN F(A) +  FN F(B)) / 2
2070 X = A
2080  FOR I = 1 TO N - 1
2090 X = X + H:T = T + 2 *  FN F(X)
2100 X = X + H:T = T +  FN F(X)
2110  NEXT I
2120 X = X + H:T = T + 2 *  FN F(X)
2130 T = 2 * H * T / 3
2140  PRINT : PRINT "L'INTEGRALE VAUT:";T
2150  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
2160  IF Z$ = "O" THEN 2000
2170  END
```

c. Exemples commentés

• ***Exemple 1*** :

Soit à calculer :

$$I = \int_5^{11} \frac{1}{\sqrt{x^2 - 6x + 8}}\, dx$$

```
50 DEF FN F(X)=1/SQR(X*X-6*X+8)

          METHODE DE SIMPSON

A=5
B=11
N=100

L'INTEGRALE VAUT:1.45170151

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La valeur exacte est :

$$I = \text{Log}\ \frac{8 + \sqrt{63}}{2 + \sqrt{3}} = 1{,}45170149$$

La valeur obtenue avec 100 pas d'intégration présente une erreur très faible.

• ***Exemple 2*** :

Reprenons le calcul de :

$$\int_1^2 (x^{14} - 8x^{11} + 3x^3)\, dx$$

```
50 DEF FN F(X)=X^14-8*X^11+3*X^3

          METHODE DE SIMPSON

A=1
B=2
N=100

L'INTEGRALE VAUT:-534.283337

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

L'erreur sur le résultat est comparable à celle obtenue avec la méthode des trapèzes.

- ***Exemple 3*** :

```
50 DEF FN F(X)=X*SIN(X)^2


   METHODE DE SIMPSON


A=1
B=12
N=100

L'INTEGRALE VAUT: 38.5890214

VOULEZ-VOUS REUTILISER LE PROGRAMME?: N
```

De même, la précision est approximativement celle donnée par la méthode des trapèzes.

La méthode de Simpson est cependant meilleure que la méthode des trapèzes lorsque la longueur de l'intervalle d'intégration est inférieure à 0,01 ; l'évaluation de la dérivée faite par le programme de la méthode des trapèzes est alors trop grossière, et cette méthode redevient d'ordre h^2.

4. Méthode de Villarceau

a. Principe

Cette méthode est la dernière des méthodes de Cotes que nous présenterons.

Elle repose toujours sur un découpage en n sous-intervalles de [a,b], mais la fonction est remplacée cette fois par un polynôme de degré 4.

On pose alors :

$$h = \frac{b-a}{4n} \qquad \text{et} \qquad x_i = a + ih$$

On démontre alors que sur chaque intervalle $[x_{4i}, x_{4i+4}]$, l'intégrale :

$$\int_{x_{4i}}^{x_{4i+4}} f(x)\,dx$$

est approchée par :

$$T_{4i} = \frac{2h}{45}\,[7f(x_{4i}) + 32f(x_{4i+1}) + 12f(x_{4i+2}) + 32f(x_{4i+3}) + 7f(x_{4i+4})]$$

L'intégrale I vaut alors :

$$I = \sum_{i=0}^{n-1} T_{4i}$$

Cette approximation de $\int_a^b f(x)\,dx$ est d'ordre h^6.

b. Programme

L'utilisation du programme est identique à celle des programmes précédents.

```
50  DEF  FN F(X) = X ^ 14 - 8 * X ^ 11 + 3 * X ^ 3
3000 CLS
3010  PRINT  TAB( 5);"METHODE DE VILLARCEAU": PRINT : PRINT

3020  INPUT "A=";A
3030  INPUT "B=";B
3040  INPUT "N=";N
3050 H = (B - A) / 4 / N
3060 X = A:T = 7 *  FN F(A)
3070  FOR I = 1 TO N
3080 X = X + H:T = T + 32 *  FN F(X)
3090 X = X + H:T = T + 12 *  FN F(X)
3100 X = X + H:T = T + 32 *  FN F(X)
3110 X = X + H:T = T + 14 *  FN F(X)
3120  NEXT I
3130 T = T - 7 *  FN F(X)
3140 T = 2 * H * T / 45
3150  PRINT : PRINT "L'INTEGRALE VAUT:";T
3160  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
3170  IF Z$ = "O" THEN 3000
3180  END
```

c. Exemples commentés

- ***Exemple 1*** :

Reprenons le calcul de l'intégrale :

$$\int_1^{12} x \sin^2 x \, dx$$

```
 50 DEF FN F(X)=X*SIN(X)^2


    METHODE DE VILLARCEAU


A=1
B=12
N=50

L'INTEGRALE VAUT:38.5890188

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le résultat est trouvé à 10^{-7} près avec 50 intervalles alors qu'il en fallait 1 000 avec la méthode des trapèzes.

- ***Exemple 2*** :

```
 50 DEF FN F(X)=X^14-8*X^11+3*X^3


    METHODE DE VILLARCEAU


A=1
B=2
N=15

L'INTEGRALE VAUT:-534.283331

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DE VILLARCEAU
```

```
A=1
B=2
N=25

L'INTEGRALE VAUT:-534.283346

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

L'intégrale est trouvée à 2.10^{-6} près avec seulement 15 pas d'intégration, alors que la précision tombe à 13.10^{-6} avec 25 pas : les erreurs numériques l'emportent ici sur l'erreur de méthode.

d. Conclusion sur les méthodes de Cotes

Il existe beaucoup d'autres formules de Cotes (en fait une infinité) donnant des approximations d'ordre h^n avec $n > 6$.

Cependant, les erreurs numériques dues à ces méthodes peuvent annuler la précision supplémentaire qu'elles apportent. On utilisera en général une vingtaine d'intervalles avec la méthode de Villarceau, et une centaine d'intervalles avec les deux autres méthodes ; ces valeurs représentent un bon compromis entre erreur de méthode, erreurs numériques et rapidité.

Les formules d'ordre supérieur à 6 sont généralement peu utilisées, et on leur préfère la méthode de Romberg.

5. Méthode de Gauss

a. Principe

Dans les méthodes de Cotes, les n points x_i intervenant dans le calcul de la valeur approchée étaient régulièrement espacés.

Dans la méthode de Gauss, ces points ne seront plus équidistants, mais seront calculés de sorte que l'approximation soit *exacte* pour les polynômes de degré inférieur à 2n.

La formule est alors :

$$I = \sum_i A_i f(x_i)$$

où les coefficients A_i sont également à calculer.

b. Programme

Le calcul des points x_i et des coefficients A_i ne peut pas être effectué dans le cas général de façon simple par le programme.

C'est pourquoi le programme proposé est un cas particulier de la méthode de Gauss utilisant $n = 8$ points.

Il faut tout d'abord effectuer un changement de variable de manière à ramener l'intervalle d'intégration à $[-1,1]$. On pose donc:

$$u = \frac{2x-(b+a)}{b-a} \quad (\Leftrightarrow \quad x = \left(\frac{b-a}{2}\right) u + \frac{b+a}{2})$$

L'intégrale s'écrit alors:

$$\int_a^b f(x)\,dx = \frac{b-a}{2} \int_{-1}^{+1} f\left[\left(\frac{b-a}{2}\right) u + \frac{b+a}{2}\right] du$$

Les huit valeurs u_i et les huit coefficients A_i sont alors indépendants de l'intervalle d'intégration choisi, et sont stockés dans le programme

L'approximation de l'intégrale est alors:

$$I = \frac{b-a}{2} \sum_{i=1}^{8} A_i f\left[\left(\frac{b-a}{2}\right) u + \frac{b+a}{2}\right]$$

Le programme ne demande donc plus que les bornes d'intégration a et b, la fonction étant toujours programmée à la ligne 5Ø:

```
5Ø DEF FN F(X) = X↑14-8*X↑11+3*X↑3
```

```
50  DEF  FN F(X) = X ^ 14 - 8 * X ^ 11 + 3 * X ^ 3
4000 CLS
4010  PRINT  TAB( 5);"METHODE DE GAUSS": PRINT : PRINT

4020  INPUT "A=";A
4030  INPUT "B=";B
4040  RESTORE
4050  FOR I = 1 TO 4
4060  READ U(I),C(I)
4070 U(9 - I) =  - U(I):C(9 - I) = C(I)
4080  NEXT I
4090  DATA  -.9602898564,0.1012285362,-0.7966664774,0.2
    223810344
4100  DATA  -0.5255324099,0.3137066458,-0.1834346424,0.
    3626837833
```

```
4110 T = 0
4120  FOR I = 1 TO 8
4130 T = T + C(I) *  FN F(((B - A) * U(I) + (B + A)) /
    2)
4140  NEXT I
4150 T = T * (B - A) / 2
4160  PRINT : PRINT "L'INTEGRALE VAUT:";T
4170  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
4180  IF Z$ = "O" THEN 4000
4190  END
```

c. Exemples commentés

- ***Exemple 1*** :

```
50 DEF FN F(X)=X^14-8*X^11+3*X^3


   METHODE DE GAUSS


A=1
B=2

L'INTEGRALE VAUT:-534.283333

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La valeur *exacte* de :

$$\int_1^2 (x^{14} - 8x^{11} + 3x^3)\,dx$$

est trouvée très rapidement.

Ceci est conforme au fait que la formule programmée est exacte pour les polynômes de degré inférieur ou égal à 15.

- ***Exemple 2*** :

```
50 DEF FN F(X)=X^4+6*X^2-11


   METHODE DE GAUSS
```

```
A=3
B=7

L'INTEGRALE VAUT:3900.80001

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le calcul théorique montre que :

$$\int_3^7 (x^4 + 6x^2 - 11)\,dx = \frac{19504}{5} = 3900{,}8$$

La légère erreur est une erreur numérique, et non une erreur de méthode.

- ***Exemple 3*** :

Reprenons le calcul de l'intégrale :

$$\int_1^{12} x \sin^2 x\,dx$$

```
50 DEF FN F(X)=X*SIN(X)^2


    METHODE DE GAUSS


A=1
B=12

L'INTEGRALE VAUT:38.9999328

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le résultat n'est obtenu qu'à 1 % près environ.

d. Conclusion

A nombre d'intervalles n égal, la méthode de Gauss est plus précise que les méthodes des trapèzes et de Simpson.

Malheureusement, avec n = 8 intervalles, la précision est inacceptable pour les fonctions autres que les polynômes.

L'usage de la méthode de Gauss sera donc réservé à ces derniers ; les résultats seront alors fournis très rapidement et avec une grande précision.

6. Méthode de Romberg

a. Introduction

Le principal inconvénient des méthodes précédentes réside dans le fait que l'on ne connaît pas la précision du résultat.

La méthode de Romberg permet de choisir à priori cette précision ; les itérations se font alors jusqu'à obtention de la précision demandée.

b. Principe

Cette méthode repose sur une utilisation originale de la méthode des trapèzes.

Étant donné un pas h, une estimation $T_0(h)$ de l'intégrale I cherchée peut être obtenue par cette méthode.

Si l'on divise le pas h par 2, l'estimation $T_0(\frac{h}{2})$ sera plus précise et on peut de même calculer $T_0(\frac{h}{4})$, $T_0(\frac{h}{8})$,... et $T_0(\frac{h}{2^n})$.

Cette suite converge vers I.

On démontre que l'on peut obtenir une deuxième suite $T_1(\frac{h}{2^{n-1}})$ pour laquelle $T_1(\frac{h}{2^k})$ est une meilleure approximation de l'intégrale que $T_0(\frac{h}{2^k})$. On obtient $T_1(\frac{h}{2^k})$ par la formule :

$$T_1(\frac{h}{2^k}) = \frac{4\,T_0(\frac{h}{2^{k+1}}) - T_0(\frac{h}{2^k})}{4-1} \qquad k \in [0,n-1]$$

On peut de la même manière obtenir une troisième suite :

$$T_2\ (\frac{h}{2^k}) = \frac{4^2 T_1\ (\frac{h}{2^{k+1}}) - T_1\ (\frac{h}{2^k})}{4^2-1} \qquad k \in [0,n-2]$$

et d'une manière générale :

$$T_p\left(\frac{h}{2^k}\right) = \frac{4^p T_{p-1}\left(\frac{h}{2^{k+1}}\right) - T_{p-1}\left(\frac{h}{2^k}\right)}{4^p - 1} \qquad k \in [0, n-p]$$

On peut alors constituer le tableau suivant :

$T_0(h)$ $\quad T_0\left(\frac{h}{2}\right)$ $\quad \ldots\ldots T_0\left(\frac{h}{2^{n-2}}\right)$ $\quad T_0\left(\frac{h}{2^{n-1}}\right)$ $\quad T_0\left(\frac{h}{2^n}\right)$

$T_1(h)$ $\quad T_1\left(\frac{h}{2}\right)$ $\quad \ldots\ldots T_1\left(\frac{h}{2^{n-2}}\right)$ $\quad T_1\left(\frac{h}{2^{n-1}}\right)$

$T_2(h)$ $\quad T_2\left(\frac{h}{2}\right)$ $\quad \ldots\ldots T_2\left(\frac{h}{2^{n-2}}\right)$

$\ldots\ldots$

$T_{n-1}(h)$ $\quad T_{n-1}\left(\frac{h}{2}\right)$

$T_n(h)$

On démontre que la suite $T_k(h)$ converge bien plus rapidement vers I que la suite $T_0\left(\frac{h}{2^k}\right)$.

Utilisation de la méthode

On commence donc par calculer $T_0(h)$ (par la méthode des trapèzes).

On calcule ensuite $T_0\left(\frac{h}{2}\right)$ et on en déduit $T_1(h)$, puis $T_0\left(\frac{h}{4}\right)$, $T_1\left(\frac{h}{2}\right)$ et $T_2(h)$...

Plus généralement, on obtient le tableau d'ordre $(n+1)$ à partir du tableau d'ordre n en calculant $T_0\left(\frac{h}{2^{n+1}}\right)$ et en en déduisant les $T_k\left(\frac{h}{2^{n+1-k}}\right)$ où k appartient à l'intervalle $[1,n+1]$.

Le calcul est terminé lorsque la différence entre $T_{n+1}(h)$ et $T_n(h)$ est inférieure à une limite fixée.

c. Programme

La fonction à intégrer doit être définie à la ligne 5Ø du programme :

```
5Ø DEF FN F(X) = LOG(1+TAN(X))
```

On fournit ensuite au programme les valeurs a et b des bornes de l'intervalle d'intégration, ainsi que la précision désirée, sous forme d'un nombre p compris entre 1 et 8 (la précision correspondante est alors de l'ordre de 10^{-p}).

```
50  DEF  FN F(X) = X ^ 14 - 8 * X ^ 11 + 3 * X ^ 3
5000 CLS
5010  PRINT  TAB( 5);"METHODE DE ROMBERG": PRINT : PRINT

5020  INPUT "A=";A
5030  INPUT "B=";B
5040  INPUT "PRECISION (1 A 8):";E:E = 10 ^ ( - E)
5050 N = 1:T(0,0) = (B - A) * ( FN F(A) +  FN F(B)) / 2

5060 H = (B - A) / (2 ^ N):X = A
5070 T(0,N) = ( FN F(B) +  FN F(A)) / 2
5080  FOR I = 1 TO (2 ^ N) - 1
5090 X = X + H:T(0,N) = T(0,N) +  FN F(X)
5100  NEXT I
5110 T(0,N) = T(0,N) * H
5120  FOR P = 1 TO N
5130 K = N - P
5140 T(P,K) = ((4 ^ P) * T(P - 1,K + 1) - T(P - 1,K)) /
    ((4 ^ P) - 1)
5150  NEXT P
5160  IF  ABS (T(N,0) - T(N - 1,0)) < E THEN 5190
5170  IF N < 9 THEN N = N + 1: GOTO 5060
5180  PRINT : PRINT "LA PRECISION NE PEUT ETRE QUE 1E";
    1 +  INT ( LOG ( ABS (T(N,0) - T(N - 1,0))) /  LOG
    (10))
5190  PRINT : PRINT "L'INTEGRALE VAUT:";T(N,0)
5200  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
5210  IF Z$ = "O" THEN 5000
5220  END
```

d. Exemples commentés

- ***Exemple 1*** :

```
50 DEF FN F(X)=X*SIN(X)^2


   METHODE DE ROMBERG
```

```
A=1
B=12
PRECISION (1 A 8):7

L'INTEGRALE VAUT:38.5890187

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La valeur à 10^{-7} près de l'intégrale :

$$\int_1^{12} x \sin^2 x \, dx$$

est obtenue assez rapidement, alors qu'il fallait 1 000 intervalles par la méthode des trapèzes.

- ***Exemple 2*** :

La valeur exacte de l'intégrale :

$$I = \int_0^{\pi/4} \ln (1 + \operatorname{tg} x) \, dx$$

est :

$$I = \frac{\pi}{8} \operatorname{Log} 2 = 0{,}272198261$$

```
50 DEF FN F(X)=LOG(1+TAN(X))

    METHODE DE ROMBERG


A=0
B=.785398163
PRECISION (1 A 8):1

L'INTEGRALE VAUT:.272198261

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DE ROMBERG


A=0
B=.785398163
PRECISION (1 A 8):8

L'INTEGRALE VAUT:.272198261

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Avec la précision la plus faible (1), le programme trouve déjà la valeur exacte (et ceci très rapidement) ; un second calcul avec la précision maximale permet de le vérifier.

• ***Exemple 3*** :

```
50 DEF FN F(X)=X^14-8*X^11+3*X^3


   METHODE DE ROMBERG


A=1
B=2
PRECISION (1 A 8):6

L'INTEGRALE VAUT:-534.283335

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
   METHODE DE ROMBERG


A=1
B=2
PRECISION (1 A 8):7

LA PRECISION NE PEUT ETRE QUE 1E-6

L'INTEGRALE VAUT:-534.283333

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

L'erreur obtenue est de 2.10^{-6} alors qu'une précision de 6 avait été demandée.

Ceci provient du fait que le test d'erreur se fait sur la différence entre deux valeurs successives des approximations, et non entre l'approximation et la valeur exacte, puisque cette dernière n'est pas connue.

En demandant une précision n, le programme ne fournit donc pas toujours le résultat avec la précision 10^{-n} ; cependant une précision de $10^{-(n-1)}$ est la plupart du temps obtenue.

Lorsqu'une précision demandée ne peut être atteinte, le programme le signale, et indique également l'ordre de grandeur de la précision obtenue.

Dans ce cas, le programme établit le tableau des approximations à l'ordre 10 et le temps de calcul est de l'ordre de quelques minutes (environ trois minutes pour l'exemple).

7. Méthode discrète des trapèzes

a. Introduction

Tous les programmes précédents permettent l'intégration de fonctions dont on connaît l'expression f(x), valable en tout point x de leur domaine de définition.

Mais il existe des fonctions dont on ne connaît la valeur qu'en un nombre fini de points: ce sont les fonctions *discrètes*.

Nous allons donc examiner l'adaptation de la méthode des trapèzes et de la méthode de Simpson à ce type de fonctions.

b. Principe

On connaît les valeurs $f(x_i)$ prises par la fonction en n points x_i non nécessairement équidistants.

L'intégrale:

$$\int_{x_i}^{x_{i+1}} f(x)\,dx$$

est alors approchée par:

$$T_i = \frac{1}{2}(x_{i+1} - x_i)(f(x_i) + f(x_{i+1}))$$

et l'on a donc:

$$\int_{x_1}^{x_n} f(x)\,dx = \sum_{i=1}^{n-1} T_i$$

c. Programme

Il faut fournir au programme le nombre n de points, les valeurs x_i et $f(x_i)$ correspondantes.

Le programme peut traiter jusqu'à cent points. Il est possible d'augmenter ce nombre en modifiant la ligne 1Ø. L'adjonction de cent points supplémentaires nécessite cependant environ 1 Koctets de mémoire.

```
10  DIM X(100),Y(100)
6000 CLS
6010  PRINT  TAB( 5);"METHODE DISCRETE DES TRAPEZES": PRINT
    : PRINT
6020  INPUT "N=";N
6030  FOR I = 1 TO N
6040  PRINT "X";I;: INPUT "=";X(I)
6050  PRINT "F(X";I;: INPUT ")=";Y(I)
6060  NEXT I
6070 T = 0
6080  FOR I = 1 TO N - 1
6090 T = T + (X(I + 1) - X(I)) * (Y(I) + Y(I + 1))
6100  NEXT I
6110  PRINT : PRINT "L'INTEGRALE VAUT:";T / 2
6120  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
   E?:";Z$
6130  IF Z$ = "O" THEN 6000
6140  END
```

d. Exemples commentés

- ***Exemple 1*** :

```
 METHODE DISCRETE DES TRAPEZES

N=5
X1=1
F(X1)=1
X2=2
F(X2)=4
X3=5
F(X3)=25
X4=7
F(X4)=49
X5=10
F(X5)=100
```

```
L'INTEGRALE VAUT:343.5

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DISCRETE DES TRAPEZES


N=6
X1=1
F(X1)=1
X2=2
F(X2)=4
X3=5
F(X3)=25
X4=7
F(X4)=49
X5=10
F(X5)=100
X6=20
F(X6)=400

L'INTEGRALE VAUT:2843.5

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Nous avons tout d'abord cherché une estimation de $\int_1^{10} x^2\,dx$, qui vaut 333, en prenant les valeurs de la fonction aux points 1, 2, 5, 7 et 10.

Le résultat obtenu est 343,5 : l'approximation est grossière car les points sont peu nombreux.

Avec six points, le programme donne 2843,5 pour $\int_1^{20} x^2\,dx$ qui vaut 2666,33...

- ***Exemple 2*** :

Nous allons montrer dans cet exemple une application de la méthode à un exemple concret : la charge d'une batterie.

L'expérience est donc la suivante : une batterie est chargée pendant quinze heures, l'intensité i la traversant étant mesurée à certains instants t_k. On se propose alors de déterminer la charge q emmagasinée par la batterie (en négligeant les pertes).

La charge totale de la batterie est donnée par :

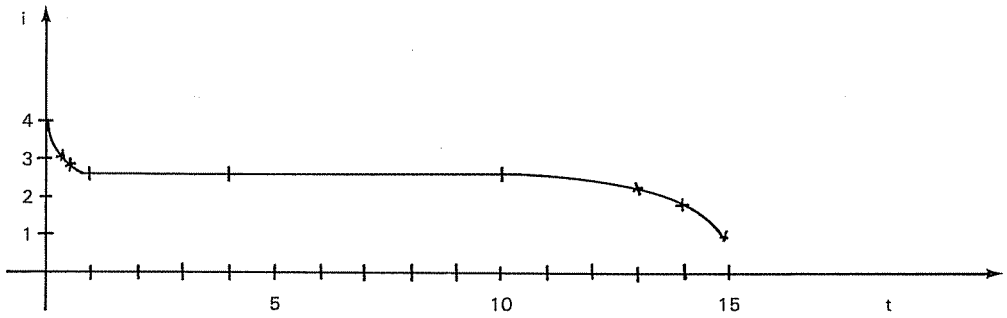
$$q = \int_{t_0}^{t_f} i(t)\,dt$$

Il suffit donc d'introduire au programme la date des instants t_k en heures, les intensités correspondantes i_k en ampères, et il fournira une estimation de la charge en A.h.

Voici le relevé expérimental :

t (en h)	0	0,25	0,5	1	4	10	13	14	14,5	15
i (en A)	4	3	2,7	2,5	2,4	2,3	2,1	1,8	1,5	1

Ce qui correspond à la courbe suivante :




```
METHODE DISCRETE DES TRAPEZES

N=10
X1=0
F(X1)=4
X2=.25
F(X2)=3
X3=.5
F(X3)=2.7
X4=1
F(X4)=2.5
X5=4
F(X5)=2.4
X6=10
F(X6)=2.3
X7=13
F(X7)=2.1
X8=14
```

```
F(X8)=1.8
X9=14.5
F(X9)=1.5
X10=15
F(X10)=1

L'INTEGRALE VAUT:34.3375

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Le programme fournit le résultat q = 34,3 Ah.

8. *Méthode discrète de Simpson*

a. Principe

Cette méthode est la généralisation de la méthode de Simpson aux fonctions discrètes.

Il faut ici que le nombre de valeurs x_i soit impair : 2m+1.

On démontre alors que :

$$T_{2i} = \int_{x_{2i-1}}^{x_{2i+1}} f(x)\,dx$$

$$= \frac{h_1 + h_2}{6h_1}\left[(2h_1 - h_2)\,f(x_{2i-1}) + \frac{(h_1 + h_2)^2}{h_2}\,f(x_{2i}) + \frac{h_1}{h_2}(2h_2 - h_1)\,f(x_{2i+1})\right]$$

avec :

$$h_1 = x_{2i} - x_{2i-1} \qquad \text{et} \qquad h_2 = x_{2i+1} - x_{2i}$$

et par suite :

$$I = \int_{x_1}^{x_{2m+1}} f(x)\,dx = \sum_{i=1}^{m} T_{2i}$$

Cette formule est exacte pour les polynômes de degré 2.

b. Programme

L'utilisation du programme reste en tout point identique à celle du programme précédent.

Si le nombre n de points est pair, la méthode de Simpson est appliquée sur les (n — 1) premiers points ; la méthode des trapèzes est alors utilisée sur le dernier intervalle, ce qui en général affaiblit la précision.

On aura donc intérêt à utiliser autant que possible le programme avec un nombre impair de points.

D'autre part, plus le nombre de points est important, moins la perturbation causée par la méthode des trapèzes est sensible.

```
10  DIM X(100),Y(100)
7000 CLS
7010  PRINT  TAB( 5);"METHODE DISCRETE DE SIMPSON": PRINT
    : PRINT
7020  INPUT "N=";N: PRINT : IF N < 3 THEN 7020
7030  FOR I = 1 TO N
7040  PRINT "X";I;: INPUT "=";X(I)
7050  PRINT "F(X";I;: INPUT ")=";Y(I)
7060  NEXT I
7070 T = 0
7080  FOR I = 1 TO N - 2 STEP 2
7090 H = X(I + 1) - X(I):K = X(I + 2) - X(I + 1)
7100 T = T + ((2 * H - K) * Y(I) + ((H + K) ^ 2) * Y(I +
    1) / K + H / K * (2 * K - H) * Y(I + 2)) * (H + K) /
    6 / H
7110  NEXT I
7120  IF  INT (N / 2) = N / 2 THEN T = T + (X(N) - X(N -
    1)) * (Y(N) + Y(N - 1)) / 2
7130  PRINT : PRINT "L'INTEGRALE VAUT:";T
7140  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
7150  IF Z$ = "O" THEN 7000
7160  END
```

c. Exemples commentés

```
    METHODE DISCRETE DE SIMPSON


N=5

X1=1
F(X1)=1
```

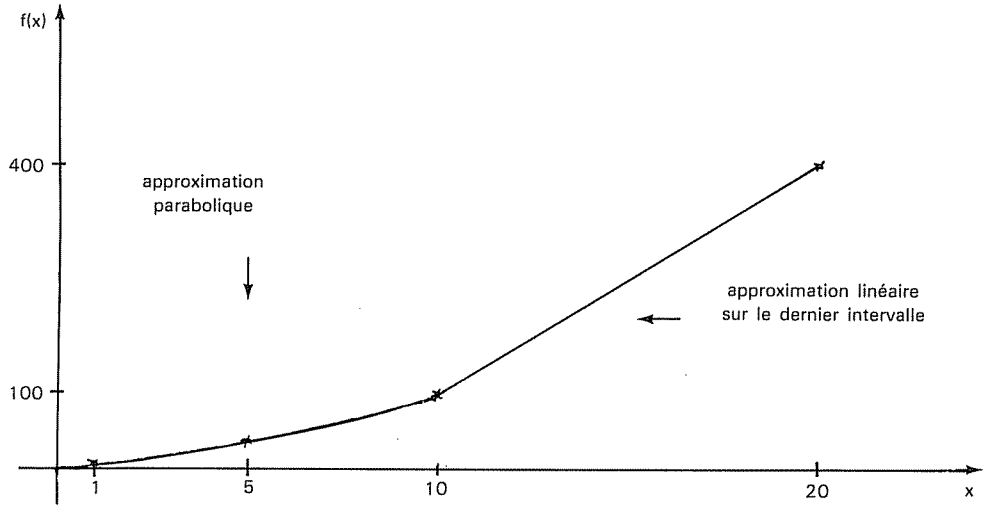
```
X2=2
F(X2)=4
X3=5
F(X3)=25
X4=7
F(X4)=49
X5=10
F(X5)=100

L'INTEGRALE VAUT:333

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    METHODE DISCRETE DE SIMPSON


N=6

X1=1
F(X1)=1
X2=2
F(X2)=4
X3=5
F(X3)=25
X4=7
F(X4)=49
X5=10
F(X5)=100
X6=20
F(X6)=400

L'INTEGRALE VAUT:2833

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N


    METHODE DISCRETE DE SIMPSON


N=10

X1=0
F(X1)=4
X2=.25
F(X2)=3
```

```
X3=.5
F(X3)=2.7
X4=1
F(X4)=2.5
X5=4
F(X5)=2.4
X6=10
F(X6)=2.3
X7=13
F(X7)=2.1
X8=14
F(X8)=1.8
X9=14.5
F(X9)=1.5
X10=15
F(X10)=1

L'INTEGRALE VAUT:34.0972223

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Nous avons repris le calcul des intégrales étudiés avec la méthode des trapèzes.

Dans le premier cas, le résultat est exact : en effet le nombre de points utilisés est impair et la méthode de Simpson donne toujours un résultat exact pour les polynômes du second degré.

Dans le deuxième cas, le résultat, meilleur que celui obtenu par la méthode des trapèzes, reste relativement imprécis, car le nombre de points étant pair, la méthode de Simpson n'est appliquée qu'entre 1 et 10.



9. Conclusion

Des cinq méthodes que nous avons étudiées pour l'intégration des fonctions continues, notre préférence va à la méthode de Romberg.

Celle-ci permet de s'affranchir du nombre d'intervalles à utiliser et surtout de connaître un ordre de grandeur de la précision du résultat.

Nous retrouverons d'ailleurs cette méthode dans le chapitre suivant consacré aux séries de Fourier.

En ce qui concerne les fonctions discrètes, la meilleure méthode est bien entendu celle de Simpson.

7 Séries de Fourier

1. Introduction

Le fait qu'une fonction périodique puisse s'exprimer sous forme d'une somme de fonctions sinusoïdales n'est pas un résultat qui laisse indifférent : comment peut-on obtenir de cette manière un signal carré, triangulaire ou en dent de scie ?

Et pourtant, la plupart des fonctions périodiques (de période T) peuvent se décomposer selon leurs harmoniques fondamentales, c'est-à-dire se développer en série de la forme :

$$f(x) = a_0 + \sum_{n=1}^{+\infty} \left(a_n \cos n \frac{2\pi}{T} x + b_n \sin n \frac{2\pi}{T} x\right)$$

De telles séries sont appelées séries de Fourier.

Nous commencerons par nous intéresser au calcul des coefficients a_n et b_n de la série de Fourier d'une fonction f.

Nous représenterons ensuite le spectre de la fonction, qui permet d'apprécier entre autre la rapidité de convergence de la série de Fourier.

Le troisième programme représentera le signal défini par les coefficients de sa série de Fourier; nous pourrons alors comparer directement le signal reconstitué avec le signal initial.

Nous terminerons cette étude avec l'examen de divers exemples.

Pour utiliser les trois programmes simultanément, ce qui est ici particulièrement intéressant puisqu'ils sont interactifs, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu:

```
100  TEXT :CLS
110  PRINT  TAB( 5);"SERIE DE FOURIER": PRINT
120  PRINT : PRINT "1- CALCUL DES COEFFICIENTS"
130  PRINT : PRINT "2- REPRESENTATION DU SPECTRE"
140  PRINT : PRINT "3- SYNTHESE DU SIGNAL"
150  PRINT : PRINT "4- FIN"
160  PRINT : INPUT "VOTRE CHOIX:";E
170  ON E GOTO 1000,2000,3000,4000
180  GOTO 100
4000  END
```

L'ensemble nécessite environ 6 Koctets de mémoire.

2. *Calcul des coefficients*

a. Principe

Lorsqu'une fonction périodique f de période T admet un développement en série de Fourier, celui-ci est unique. Les coefficients sont alors donnés par:

$$f(x) = a_0 + \sum_{n=1}^{+\infty} \left(a_n \cos n \frac{2\pi}{T} x + b_n \sin n \frac{2\pi}{T} x\right)$$

avec:

$$\begin{cases} a_0 = \dfrac{1}{T} \displaystyle\int_{-\frac{T}{2}}^{\frac{T}{2}} f(x)\, dx \\ a_n = \dfrac{2}{T} \displaystyle\int_{-\frac{T}{2}}^{\frac{T}{2}} f(x) \cos\left(n \frac{2\pi}{T} x\right) dx \\ b_n = \dfrac{2}{T} \displaystyle\int_{-\frac{T}{2}}^{\frac{T}{2}} f(x) \sin\left(n \frac{2\pi}{T} x\right) dx \end{cases}$$

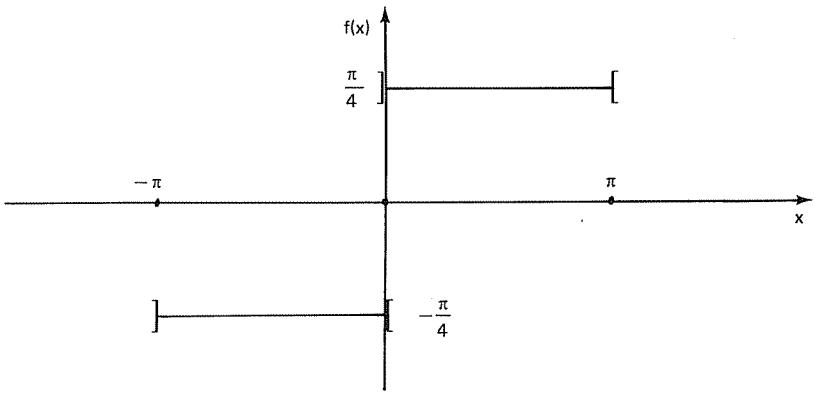
Le calcul des intégrales sera effectué par la méthode de Romberg, étudiée dans le chapitre précédent. Rappelons simplement que cette méthode permet de choisir la précision des résultats.

b. Programme

La fonction f doit être définie à partir de la ligne 19ØØ du programme sous la forme :

```
19ØØ Y=0
191Ø IF X>-PI AND X<0 THEN Y=-PI/4
192Ø IF X>0 AND X<PI THEN Y=PI/4
193Ø RETURN
```

Ceci correspond au signal carré :



Il est important que les points de discontinuité de la fonction soient affectés de la valeur 0 ; les calculs des intégrales sont dans ce cas les plus précis.

Il faut également préciser la période de la fonction à la ligne 195Ø du progamme :

```
195Ø T=2*PI
```

REMARQUE : Certains interpréteurs BASIC ne possèdent pas la valeur de la constante PI ; il faut alors la définir au début du programme, par exemple :

```
5 PI=3.1415926536
```

Le programme demande ensuite le nombre de termes à calculer (égal au plus à 20) ainsi que la précision désirée.

```
10 C = 279:L = 159:PI = 3.1415926536
20  DIM A(20),B(20),C(20),P(C),AF(20)
1000 CLS
1010  PRINT  TAB( 5);"CALCUL DES COEFFICIENTS": PRINT
1020  PRINT : PRINT "1- DEFINITION DE LA FONCTION"
1030  PRINT : PRINT "2- CALCUL DES COEFFICIENTS"
1040  PRINT : INPUT "VOTRE CHOIX:";E
1050  IF E = 1 THEN  LIST 1900 - 1999: END
1060  GOSUB 1950
1070  PRINT : INPUT "NOMBRE DE TERMES A CALCULER:";NB: PRINT

1080  IF NB < 1 OR NB > 20 THEN 1070
1090  INPUT "PRECISION (1 A 8):";E: PRINT
1100  FOR Q = 0 TO NB
1110  PRINT
1120 N = 0:A =  - T / 2:T2 = T / 2000
1130 X = A: GOSUB 1900:YA = Y *  COS (2 * PI * Q * X /
    T)
1140 X =  - A: GOSUB 1900:YB = Y *  COS (2 * PI * Q * X
     / T)
1150 T(Q,0) = T * (YA + YB) / 2
1160 N = N + 1:H = T / (2 ^ N)
1170 T(Q,N) = (YA + YB) / 2
1180  FOR I = 1 TO (2 ^ N) - 1
1190 X = A + I * H
1200  IF  ABS (X) < T2 THEN X = 0
1210  GOSUB 1900
1220 T(Q,N) = T(Q,N) + Y *  COS (2 * PI * Q * X / T)
1230  NEXT I
1240 T(Q,N) = T(Q,N) * H
1250  FOR P = 1 TO N
1260 K = N - P
1270 T(P,K) = ((4 ^ P) * T(P - 1,K + 1) - T(P - 1,K)) /
    ((4 ^ P) - 1)
1280  NEXT P
1290  IF (N < 4) OR (Q = 8 AND N < 5) OR (Q = 16 AND N <
    6) THEN 1160
1300  IF  ABS (T(N,0) - T(N - 1,0)) < 10 ^ ( - E) THEN
    1340
1310  IF N < 10 THEN 1160
1320 PR = 1 +  INT ( LOG ( ABS (T(N,0) - T(N - 1,0)) *
    2 / T) /  LOG (10))
1330  IF ( - PR <  > E) THEN  PRINT "LA PRECISION NE PE
    UT ETRE QUE 1E";PR
```

```
1340 A(Q) = T(N,0) * 2 / T
1350  IF  ABS (A(Q)) < 1E - 8 THEN A(Q) = 0
1360  IF Q = 0 THEN A(0) = A(0) / 2: PRINT "A0=";A(0): GOTO
    1630
1370  PRINT "A";Q;"=";A(Q)
1380 N = 0
1390 X = A: GOSUB 1900:YA = Y *  SIN (2 * PI * Q * X /
    T)
1400 X =  - A: GOSUB 1900:YB = Y *  SIN (2 * PI * Q * X
     / T)
1410 T(0,0) = T * (YA + YB) / 2
1420 N = N + 1:H = T / (2 ^ N)
1430 T(0,N) = (YA + YB) / 2
1440  FOR I = 1 TO (2 ^ N) - 1
1450 X = A + I * H
1460  IF  ABS (X) < T2 THEN X = 0
1470  GOSUB 1900
1480 T(0,N) = T(0,N) + Y *  SIN (2 * PI * Q * X / T)
1490  NEXT I
1500 T(0,N) = T(0,N) * H
1510  FOR P = 1 TO N
1520 K = N - P
1530 T(P,K) = ((4 ^ P) * T(P - 1,K + 1) - T(P - 1,K)) /
    ((4 ^ P) - 1)
1540  NEXT P
1550  IF (N < 4) OR (Q = 8 AND N < 5) OR (Q = 16 AND N <
    6) THEN 1420
1560  IF  ABS (T(N,0) - T(N - 1,0)) < 10 ^ ( - E) THEN
    1600
1570  IF N < 10 THEN 1420
1580 PR = 1 +  INT ( LOG ( ABS (T(N,0) - T(N - 1,0)) *
    2 / T) /  LOG (10))
1590  IF ( - PR <  > E) THEN  PRINT "LA PRECISION NE PE
    UT ETRE QUE 1E";PR
1600 B(Q) = T(N,0) * 2 / T
1610  IF  ABS (B(Q)) < 1E - 8 THEN B(Q) = 0
1620  PRINT "B";Q;"=";B(Q)
1630  NEXT Q
1640 CLS
1650  FOR Q = 0 TO NB
1660  PRINT "A";Q;"=";A(Q);
1670  IF Q = 0 THEN  PRINT : GOTO 1690
1680  PRINT  TAB( 20);"B";Q;"=";B(Q)
1690  IF NB < 11 THEN  PRINT
1700  NEXT Q
```

```
1710  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E:";Z$
1720  IF Z$ = "O" THEN 1000
1730  END
1900 Y = 0
1910  IF X >  - PI AND X < 0 THEN Y =  - PI / 4
1920  IF X > 0 AND X < PI THEN Y = PI / 4
1930  RETURN
1950 T = 2 * PI
1960  RETURN
```

3. Représentation du spectre

a. Principe

Nous avons vu que la série de Fourier d'une fonction f s'écrivait sous la forme :

$$f(x) = a_0 + \sum_{n=1}^{+\infty} \left(a_n \cos n \frac{2\pi}{T} x + b_n \sin n \frac{2\pi}{T} x\right)$$

Elle peut encore s'exprimer ainsi :

$$f(x) = \sum_{k=-\infty}^{+\infty} c_k e^{ik\frac{2\pi}{T}x}$$

où les coefficients c_k sont complexes et donnés par :

$$\begin{cases} c_0 = a_0 \\ c_k = \dfrac{a_k - ib_k}{2} \\ c_{-k} = \dfrac{a_k + ib_k}{2} \end{cases} \Leftrightarrow \begin{cases} a_0 = c_0 \\ a_k = c_k + c_{-k} \\ b_k = i(c_k - c_{-k}) \end{cases}$$

avec $k > 0$

Le module du coefficient c_k représente en quelque sorte le "poids" de l'harmonique k dans le développement de f en série de Fourier.

Portons alors sur un graphe les modules des coefficients c_k en fonction de la fréquence : nous obtenons le spectre de la fonction f.

Un spectre se présente donc sous la forme d'une suite de segments parallèles (que l'on appelle raies) ; chaque segment correspond à un harmonique et sa longueur est proportionnelle au module du coefficient c_k.

Il est possible d'apprécier la convergence de la série en considérant la décroissance des raies ; plus celle-ci est rapide, plus la convergence de la série vers la fonction l'est également.

b. Programme

Le programme trace le spectre sur l'écran à partir soit des coefficients calculés par le programme précédent, soit de coefficients qui lui sont fournis.

Il faut ici encore préciser le nombre de raies à représenter (au plus égal à 20).

Le programme utilise la haute résolution graphique de l'ordinateur, et voici la signification des commandes graphiques employées :

→ HIRES : place l'ordinateur en mode haute résolution.

→ TEXT : place l'ordinateur en mode texte. (Certains ordinateurs ne distinguent pas mode texte et mode haute résolution.)

→ LINE(A,B)-(C,D) : trace un segment de droite entre les points de coordonnées (A,B) et (C,D).

Ces instructions (ou leur équivalent) sont présentes sur la plupart des micro-ordinateurs ; il ne devrait donc pas y avoir de difficultés d'adaptation.

Pour conserver un facteur d'échelle constant et permettre des comparaisons directes entre différents spectres, nous avons systématiquement affecté la longueur maximale des segments (soit dix unités) à l'harmonique de coefficient $|c_k|$ le plus élevé ; celui-ci est en général le terme fondamental (ordre 1).

```
10 C = 279:L = 159:PI = 3.1415926536
20  DIM A(20),B(20),C(20),P(C),AF(20)
2000 CLS
2010  PRINT  TAB( 5);"REPRESENTATION DU SPECTRE": PRINT

2020  PRINT : PRINT "1- UTILISATION DES COEFFICIENTS DE
    JA"
2030  PRINT "   CALCULES"
2040  PRINT : PRINT "2- INTRODUCTION DES COEFFICIENTS"
2050  PRINT : INPUT "VOTRE CHOIX:";E
2060  IF E = 1 THEN 2210
2070  PRINT
2080  INPUT "NOMBRE DE TERMES:";NB
```

```
2090  IF NB < 1 OR NB > 20 THEN 2070
2100  PRINT
2110  INPUT "A0:";A(0)
2120 C(0) =  ABS (A(0))
2130  PRINT
2140  FOR I = 1 TO NB
2150  PRINT "A";I;
2160  INPUT A(I)
2170  PRINT "B";I;
2180  INPUT B(I)
2190  PRINT
2200  NEXT I
2210 MX = 0
2220  FOR I = 1 TO NB
2230 C(I) =  SQR (A(I) * A(I) + B(I) * B(I))
2240  IF C(I) > MX THEN MX = C(I)
2250  NEXT I
2260  IF MX = 0 THEN 2070
2270  FOR I = 1 TO NB
2280 C(I) = C(I) / MX * (L - 20)
2290  NEXT I
2300 E =  INT ((C - 20) / (NB + 1))
2310  TEXT :HIRES
2320 XA = 10:YA = L - 10
2330 LINE(XA,0) - (XA,L)
2340 LINE(XA + (NB + 1) * E,0) - (XA + (NB + 1) * E,L)
2350 LINE(0,YA) - (2 * XA + (NB + 1) * E,YA)
2360  FOR I = 1 TO NB
2370 LG = 3
2380  IF (I / 5) =  INT (I / 5) THEN LG = 6
2390 LINE(XA + I * E,YA) - (XA + I * E,YA + LG)
2400  NEXT I
2410  FOR I = 1 TO 10
2420 LG = 3
2430  IF (I / 5) =  INT (I / 5) THEN LG = 6
2440 LINE(XA,YA - I * (L - 20) / 10) - (XA - LG,YA - I *
    (L - 20) / 10)
2450 LINE(XA + (NB + 1) * E,YA - I * (L - 20) / 10) - (
    LG + XA + (NB + 1) * E,YA - I * (L - 20) / 10)
2460  NEXT I
2470  FOR I = 1 TO NB
2480 LINE(XA + I * E,YA) - (XA + I * E,YA - C(I))
2490  NEXT I
2500  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
2510  TEXT
2520  IF Z$ = "O" THEN 2000
2530  END
```

4. *Synthèse du signal*

a. Principe

Nous nous sommes intéressés jusqu'à présent à la transformation de la fonction f en sa série de Fourier; nous allons maintenant adopter la démarche inverse: à partir des coefficients de Fourier, nous allons calculer puis représenter la fonction f.

Ceci nous permettra en particulier de vérifier les résultats des calculs et d'apprécier la rapidité de convergence de la série de Fourier vers la fonction.

b. Programme

Le programme peut ici encore utiliser des coefficients qui lui sont fournis ou les coefficients déjà introduits ou calculés par l'intermédiaire de l'un des programmes précédents.

Les harmoniques sont additionnés les uns après les autres, il est donc possible de visualiser tous les intermédiaires de la reconstitution; cependant, pour éviter une saturation de l'écran, nous avons prévu la possibilité de choisir les courbes que l'on désire superposer sur l'écran.

Afficher la courbe 7 signifie donc tracer la fonction obtenue en sommant les sept premiers harmoniques (avec éventuellement la composante continue correspondant au terme a_0).

```
10 C = 279:L = 159:PI = 3.1415926536
20  DIM A(20),B(20),C(20),P(C),AF(20)
3000 CLS
3010  PRINT  TAB( 5);"SYNTHESE DU SIGNAL": PRINT
3020  PRINT : PRINT "1- UTILISATION DES COEFFICIENTS"
3030  PRINT "   PRECEDENTS"
3040  PRINT : PRINT "2- INTRODUCTION DES COEFFICIENTS"
3050  PRINT : INPUT "VOTRE CHOIX:";E
3060  IF E = 1 THEN 3200
3070  PRINT
3080  INPUT "NOMBRE DE TERMES:";NB
3090  IF NB < 1 OR NB > 20 THEN 3070
3100  PRINT
3110  INPUT "A0:";A(0)
3120  PRINT
```

```
3130  FOR I = 1 TO NB
3140  PRINT "A";I;
3150  INPUT A(I)
3160  PRINT "B";I;
3170  INPUT B(I)
3180  PRINT
3190  NEXT I
3200  PRINT : PRINT "COURBES A AFFICHER :": PRINT
3210  FOR K = 0 TO NB
3220  PRINT "COURBE NUMERO ";K;
3230  INPUT K$
3240 AF(K) = 0
3250  IF K$ = "O" THEN AF(K) = 1
3260  NEXT K
3270  TEXT :HIRES
3280 XA =  INT (C / 2)
3290 YA =  INT (L / 2)
3300 LINE(XA,0) - (XA,L)
3310 LINE(0,YA) - (C,YA)
3320 DG =  INT ((C - 5) / 24)
3330 D = 1
3340  FOR I = 1 TO 12
3350 LG = 0:PG = 0
3360  IF I / 3 =  INT (I / 3) THEN LG = 3:PG = 1
3370  IF I / 2 =  INT (I / 2) THEN LG = 5:PG = 2
3380  IF I / 6 =  INT (I / 6) THEN LG = 7:PG = 3
3390 LINE(XA + D * I * DG,YA - PG) - (XA + D * I * DG,Y
    A - PG + LG)
3400  NEXT I
3410  IF D = 1 THEN D =  - 1: GOTO 3340
3420  FOR I = 1 TO 5
3430 LG = 3:PG = 1
3440  IF (I / 5) =  INT (I / 5) THEN LG = 5:PG = 2
3450 LINE(XA - PG,YA - D * I * (L - 20) / 10) - (XA - P
    G + LG,YA - D * I * (L - 20) / 10)
3460  NEXT I
3470  IF D =  - 1 THEN D = 1: GOTO 3420
3480  FOR I = 0 TO C
3490 P(I) = YA - A(0) * (L - 20) / 2
3500  NEXT I
3510 XD = XA - 12 * DG
3520 XF = XA + 12 * DG
3530 CX = 24 * DG
3540  IF AF(0) = 0 THEN 3560
3550  IF P(0) >  = 0 AND P(0) <  = L THEN LINE(XD, INT
    (P(0))) - (XF, INT (P(0)))
```

```
3560  FOR I = 1 TO NB
3570 FL = 0
3580  IF AF(I) = 1 THEN FL = 1
3590  IF A(I) = 0 AND B(I) = 0 AND FL = 0 THEN 3690
3600 K = I * 2 * PI / CX:K1 =  - PI * I
3610 P(XD) = P(XD) - (L - 20) / 2 * (A(I) *  COS (K1) -
    B(I) *  SIN (K1))
3620  FOR J = XD + 1 TO XF
3630 K2 = (J - XD) * K - K1
3640 P(J) = P(J) - (L - 20) / 2 * (A(I) *  COS (K2) + B
    (I) *  SIN (K2))
3650  IF FL = 0 THEN 3680
3660  IF P(J) < 0 OR P(J) > L OR P(J - 1) < 0 OR P(J -
    1) > L THEN 3680
3670 LINE(J - 1, INT (P(J - 1))) - (J, INT (P(J)))
3680  NEXT J
3690  NEXT I
3700  INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMME?:";Z$
3710  TEXT
3720  IF Z$ = "O" THEN 3000
3730  END
```

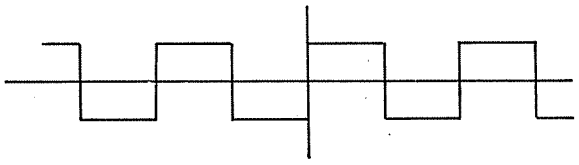
5. *Exemples commentés*

Nous allons maintenant appliquer les programmes précédents à l'étude de différents signaux fréquemment rencontrés en électricité.

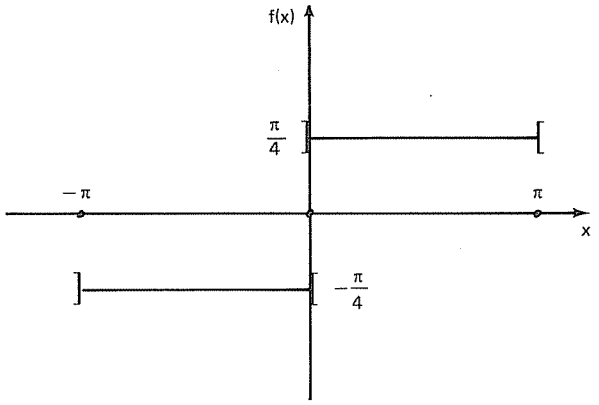
Pour chaque exemple, nous comparerons les coefficients de Fourier calculés par le programme avec les valeurs exactes, nous représenterons le spectre, et quatre étapes de la reconstitution nous permettront d'apprécier la convergence des séries vers les fonctions (n représentera le nombre d'harmoniques utilisés pour chaque courbe).

a. Signal carré

Voici tout d'abord la représentation graphique d'un tel signal, tel que l'on pourrait l'observer sur un oscilloscope qui serait connecté à un générateur de signaux:



Nous limiterons notre étude à une période de ce signal, et nous programmerons en fait la fonction suivante :




```
1900 Y=0

1910 IF X>-PI AND X<0 THEN Y=-PI/4

1920 IF X>0 AND X<PI THEN Y=PI/4

1950 T=2*PI
```

Rappelons que les valeurs adoptées aux points de discontinuité jouent un rôle essentiel lors du calcul des coefficients.

Les valeurs des coefficients de Fourier d'un tel signal sont :

$a_n = 0$

$$b_n = \begin{cases} \frac{1}{n} \text{ si n est impair} \\ 0 \text{ si n est pair} \end{cases}$$

Voici les résultats fournis par le programme avec une précision demandée de 7 :

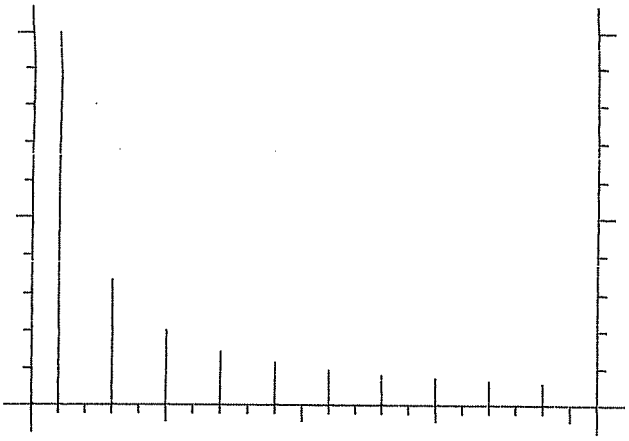
```
        CALCUL DES COEFFICIENTS


NOMBRE DE TERMES A CALCULER:20

PRECISION (1 A 8):7

A0=0
A1=0                    B1=1
A2=0                    B2=0
```

```
A3=0                    B3=.333333334
A4=0                    B4=0
A5=0                    B5=.199999999
A6=0                    B6=0
A7=0                    B7=.142857143
A8=0                    B8=0
A9=0                    B9=.111111111
A10=0                   B10=0
A11=0                   B11=.090909091
A12=0                   B12=0
A13=0                   B13=.0769230768
A14=0                   B14=0
A15=0                   B15=.0666666667
A16=0                   B16=0
A17=0                   B17=.0588235293
A18=0                   B18=0
A19=0                   B19=.0526315788
A20=0                   B20=0

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Les valeurs obtenues sont en fait exactes à 10^{-9} près.

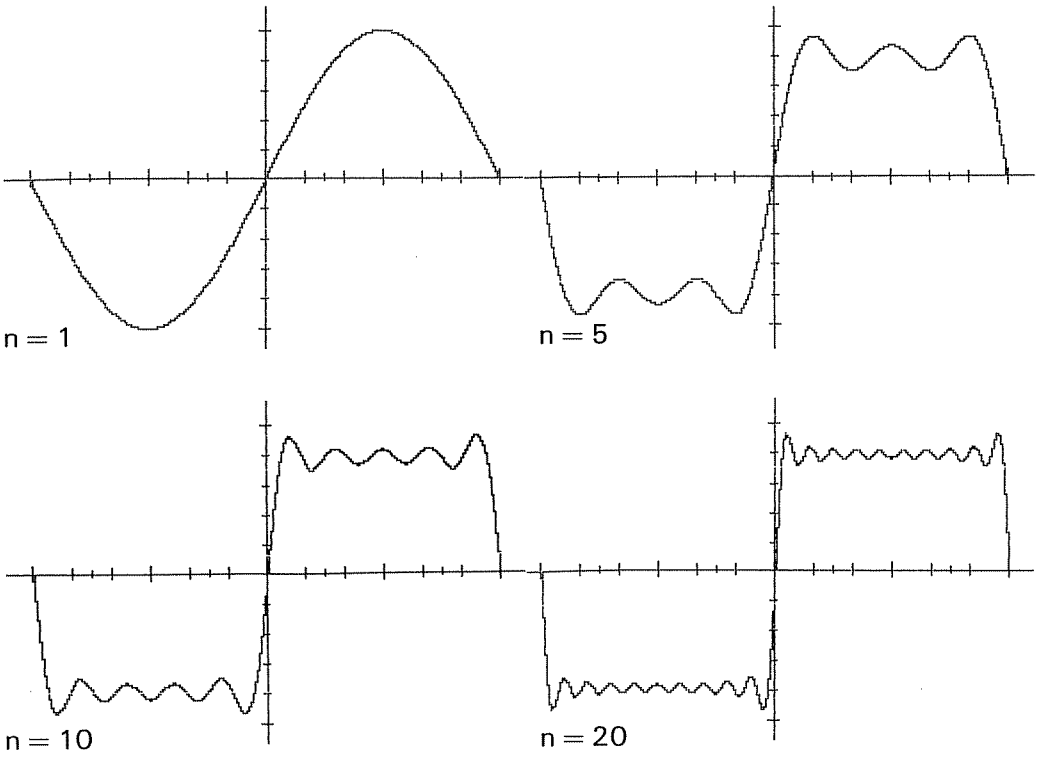
Il n'est pas surprenant que les coefficients a_n soient nuls, puisque la fonction f est impaire.

Représentation du spectre :



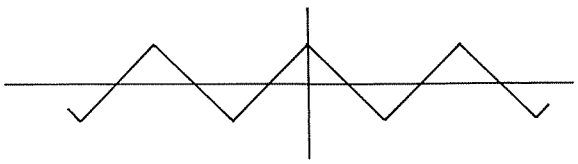
Nous constatons immédiatement que le signal carré ne possède que des harmoniques impaires.

Reconstitution du signal :

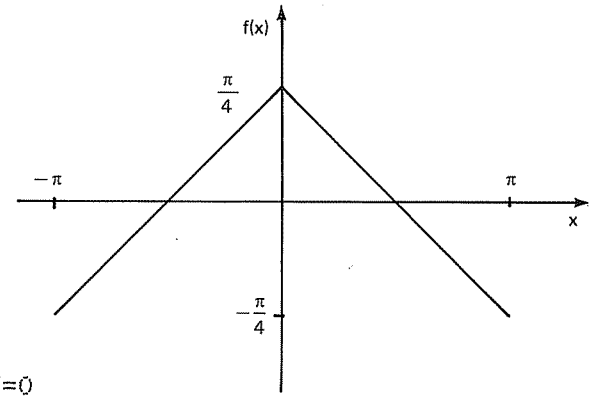


b. Signal triangulaire

Nous aurions sur l'oscilloscope avec un tel signal :



Pour une période, nous considérerons donc:




```
1900 Y=0

1910 IF X<=0 THEN Y=(PI/2+X)/2

1920 IF X>0 THEN Y=(PI/2-X)/2
```

Les valeurs des coefficients de Fourier sont:

$$a_n = \begin{cases} \dfrac{2}{\pi n^2} & \text{si n est impair} \\ 0 & \text{si n est pair} \end{cases}$$

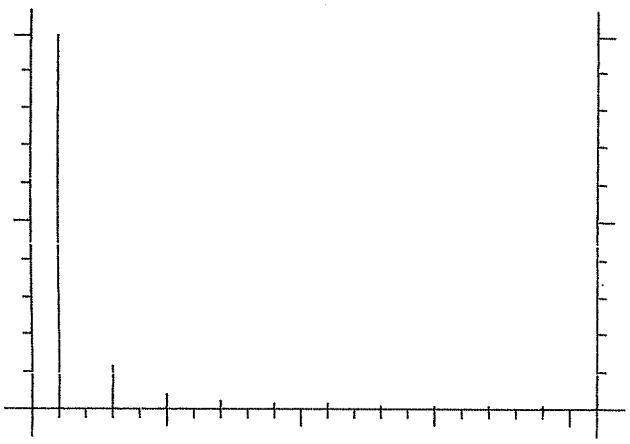
$b_n = 0$

Les résultats fournis par le programme avec une précision demandée de 4 sont en fait exacts à 10^{-6} près.

```
          CALCUL DES COEFFICIENTS


NOMBRE DE TERMES A CALCULER:20

PRECISION (1 A 8):7

A0=.636619772
A1=0                    B1=0
A2=-.424413181          B2=0
A3=0                    B3=0
A4=-.0848826366         B4=0
A5=0                    B5=0
A6=-.0363782725         B6=0
A7=0                    B7=0
A8=-.0202101515         B8=0
```

```
A9=0                    B9=0
A10=-.0128610055        B10=0
A11=0                   B11=0
A12=-8.90377308E-03B12=0
A13=0                   B13=0
A14=-6.52943359E-03B14=0
A15=0                   B15=0
A16=-4.99309658E-03B16=0
A17=0                   B17=0
A18=-3.9419186E-03 B18=0
A19=0                   B19=0
A20=-3.19107666E-03B20=0

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

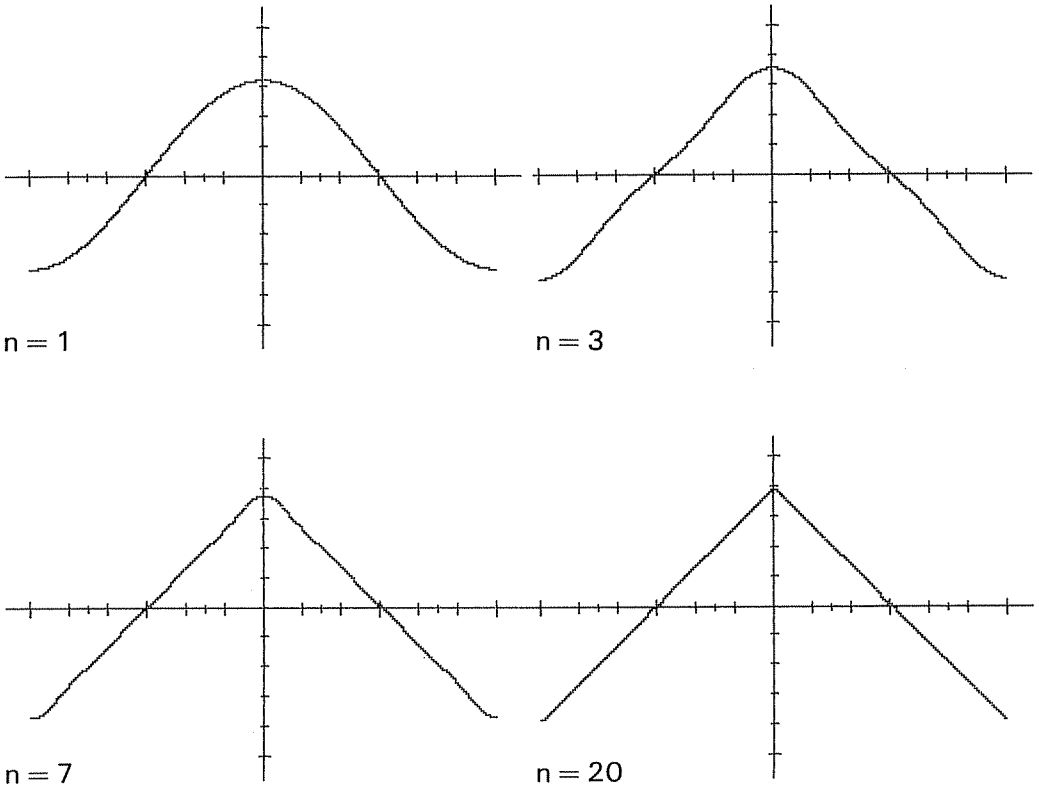
Représentation du spectre :



Comparons ce spectre avec celui du signal carré : la décroissance des raies est ici plus marquée et la série converge beaucoup plus rapidement vers la fonction (ceci s'explique par le fait que les coefficients du triangle sont en $\frac{1}{n^2}$ alors que ceux du carré ne sont qu'en $\frac{1}{n}$).

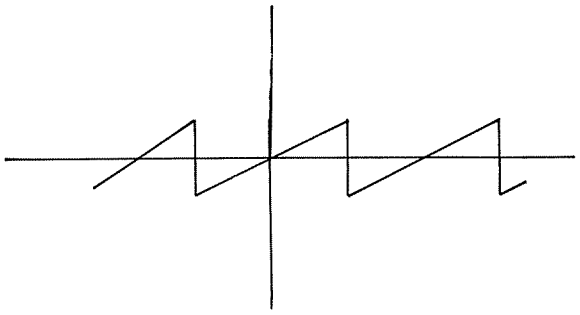
Reconstitution du signal :

Nous constatons la rapide convergence de la série :

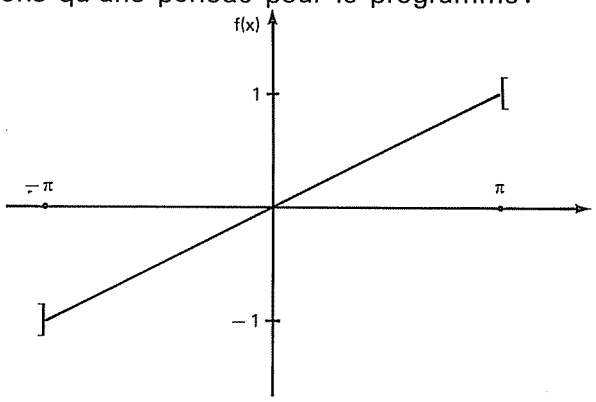


c. Signal en dent de scie

Nous observerions sur l'oscilloscope :



Nous ne retiendrons qu'une période pour le programme:




```
1900 Y=0

1910 IF X>-PI AND X<PI THEN Y=X/PI

1920 REM
```

Les coefficients de Fourier sont:

$a_n = 0$

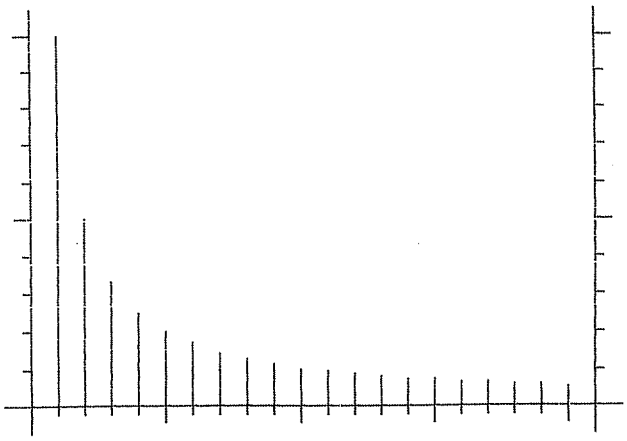
$b_n = (-1)^{n+1} \cdot \frac{2}{\pi n}$

Les résultats fournis par le programme avec une précision demandée de 6 sont exacts à 10^{-9} près.

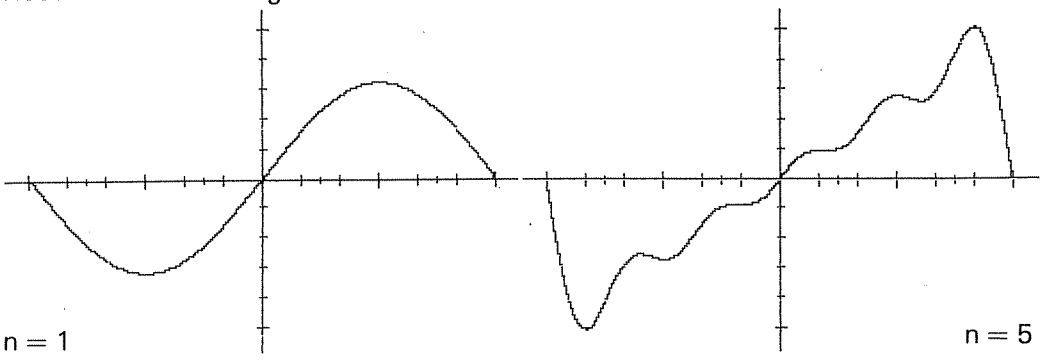
```
    CALCUL DES COEFFICIENTS

NOMBRE DE TERMES A CALCULER:20

PRECISION (1 A 8):6

A0=0
A1=0                B1=.636619772
A2=0                B2=-.318309886
A3=0                B3=.212206591
A4=0                B4=-.159154943
A5=0                B5=.127323955
A6=0                B6=-.106103296
A7=0                B7=.0909456819
```

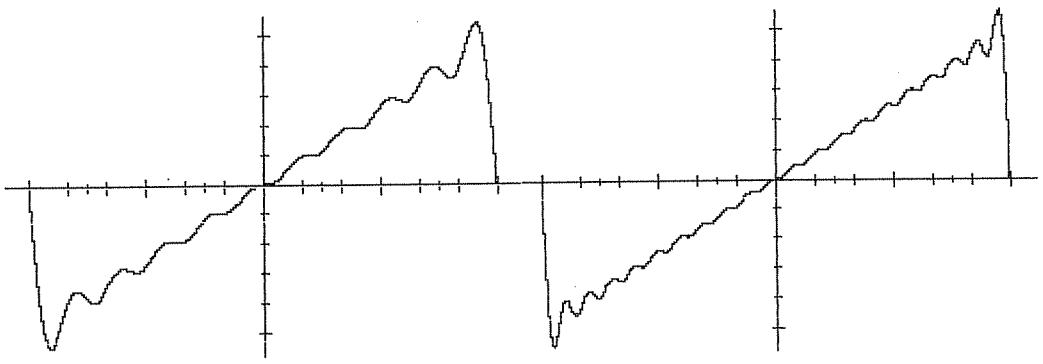
```
A8=0                    B8=-.0795774712
A9=0                    B9=.0707355302
A10=0                   B10=-.0636619773
A11=0                   B11=.057874525
A12=0                   B12=-.0530516479
A13=0                   B13=.0489707524
A14=0                   B14=-.0454728411
A15=0                   B15=.0424413182
A16=0                   B16=-.0397887357
A17=0                   B17=.0374482222
A18=0                   B18=-.035367765
A19=0                   B19=.0335063039
A20=0                   B20=-.0318309885

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Représentation du spectre :



Reconstitution du signal :

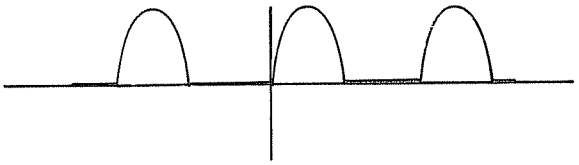




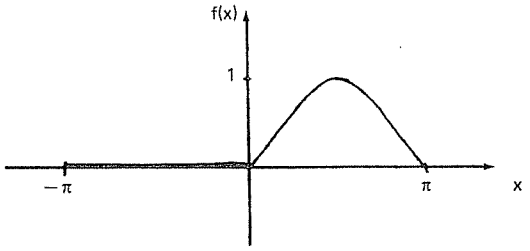
n = 10 n = 20

d. Sinusoïde redressée simple alternance

Ce signal pourrait s'observer en visualisant la tension présente aux bornes d'une diode utilisée en régime sinusoïdal :



Nous ne retiendrons qu'une période de ce signal :




```
1900 Y=0

1910IF X>0 THEN Y=SIN(X)
```

Les calculs théoriques fournissent les valeurs suivantes:

$$a_0 = \frac{1}{\pi}$$

$$a_n = \begin{cases} 0 & \text{si n est impair} \\ -\dfrac{2}{\pi(n^2-1)} & \text{si n est pair} \end{cases}$$

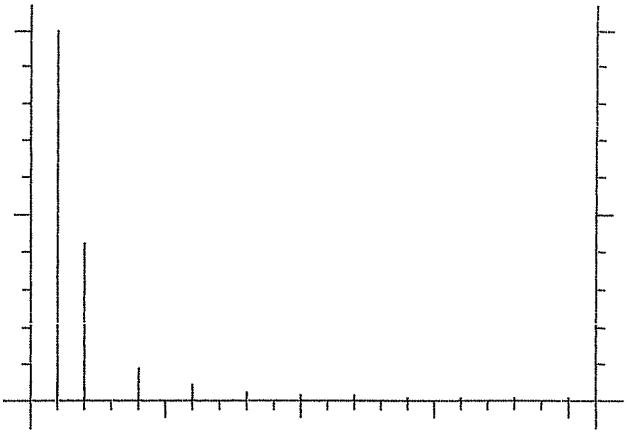
$$b_1 = \frac{1}{2}$$

$$b_n = 0 \qquad \text{pour } n \geqslant 2$$

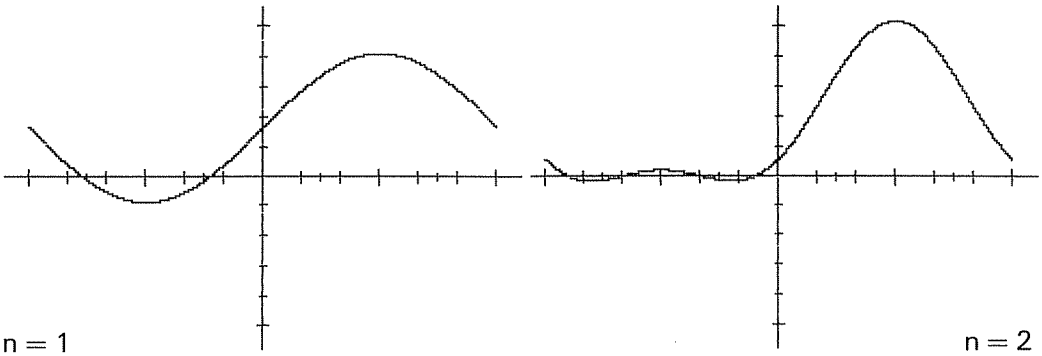
Avec une précision demandée de 4, le programme donne les valeurs exactes à 10^{-7} près, sauf le coefficient a_{10} qui n'est juste qu'à 2.10^{-4} près.

```
                CALCUL DES COEFFICIENTS


 NOMBRE DE TERMES A CALCULER:20

 PRECISION (1 A 8):4

 A0=.31830988
 A1=0                    B1=.5
 A2=-.212206655          B2=0
 A3=0                    B3=0
 A4=-.0424413253         B4=0
 A5=0                    B5=0
 A6=-.0181891363         B6=0
 A7=0                    B7=0
 A8=-.0101050781         B8=0
 A9=0                    B9=0
 A10=-6.60951968E-03B10=0
 A11=0                   B11=0
 A12=-4.4519192E-03 B12=0
 A13=0                   B13=0
 A14=-3.26471638E-03B14=0
 A15=0                   B15=0
 A16=-2.49654934E-03B16=0
 A17=0                   B17=0
 A18=-1.9709596E-03 B18=0
 A19=0                   B19=0
 A20=-1.59553972E-03B20=0

 VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Représentation du spectre :

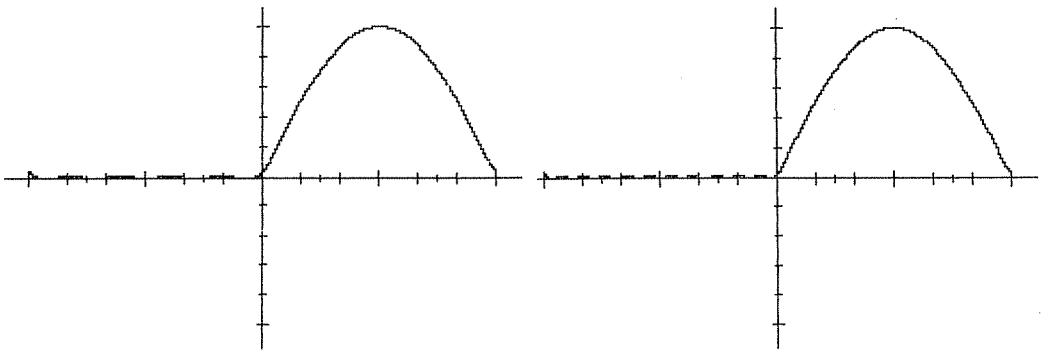


Reconstitution du signal :



n = 1

n = 2

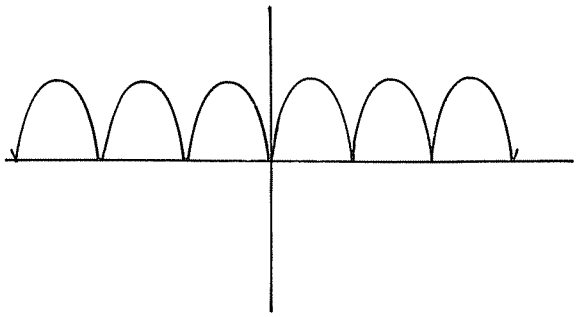


n = 8

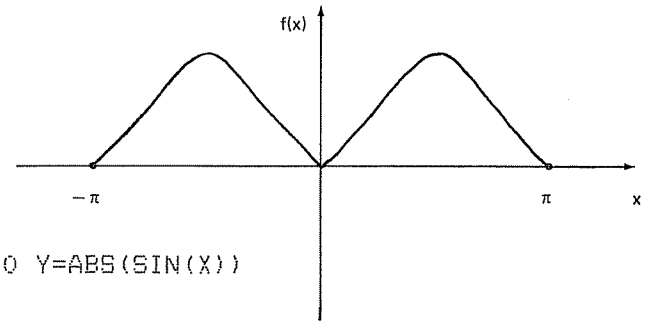
n = 20

e. Sinusoïde redressée double alternance

Ce signal pourrait s'observer en sortie d'un pont de diodes:



Sur une période nous obtenons:




```
1910 Y=ABS(SIN(X))
```

Les calculs théoriques fournissent les valeurs:

$$a_0 = \frac{2}{\pi}$$

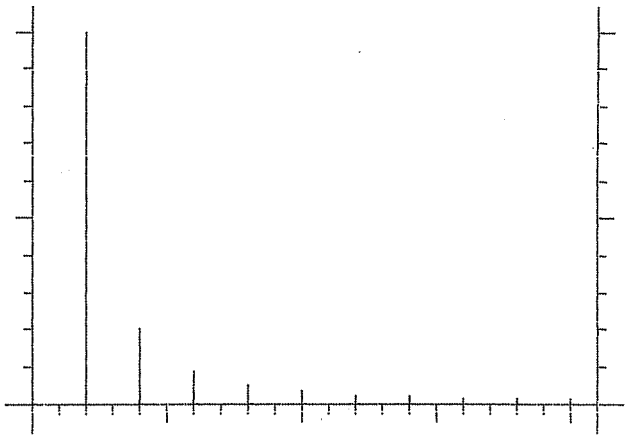
$$a_n = \begin{cases} 0 & \text{si n est impair} \\ -\dfrac{4}{\pi(n^2-1)} & \text{si n est pair} \end{cases}$$

Avec une précision demandée de 7, les résultats sont obtenus à 10^{-9} près:

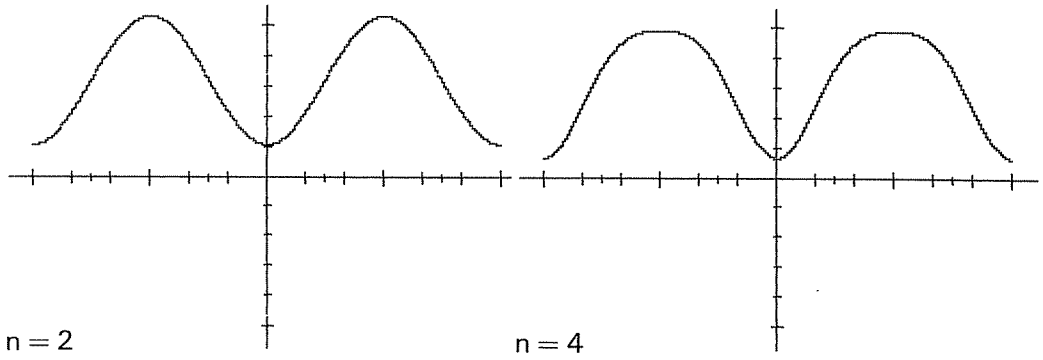
```
CALCUL DES COEFFICIENTS

NOMBRE DE TERMES A CALCULER:20

PRECISION (1 A 8):4
```

```
A0=0
A1=.636619769          B1=0
A2=0                   B2=0
A3=.0707355301         B3=0
A4=0                   B4=0
A5=.0254648096         B5=0
A6=0                   B6=0
A7=.0129922395         B7=0
A8=0                   B8=0
A9=7.85950436E-03      B9=0
A10=0                  B10=0
A11=5.26133163E-03     B11=0
A12=0                  B12=0
A13=3.76705407E-03     B13=0
A14=0                  B14=0
A15=2.82942069E-03     B15=0
A16=0                  B16=0
A17=2.20283577E-03     B17=0
A18=0                  B18=0
A19=1.76319098E-03     B19=0
A20=0                  B20=0

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

Représentation du spectre :

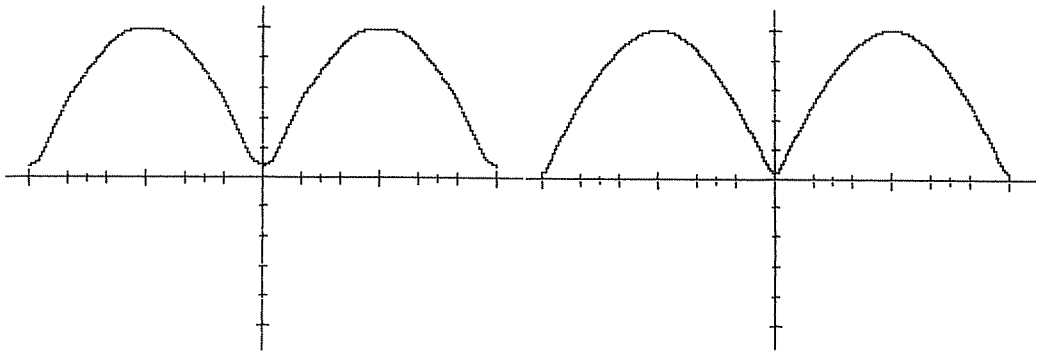


Reconstitution du signal :



n = 2

n = 4



n = 8

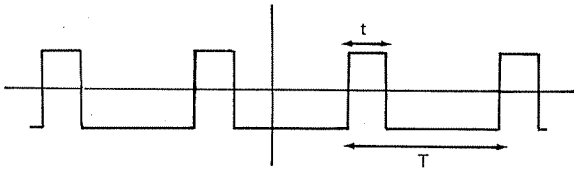
n = 20

f. Signal rectangulaire de rapport cyclique 1/4

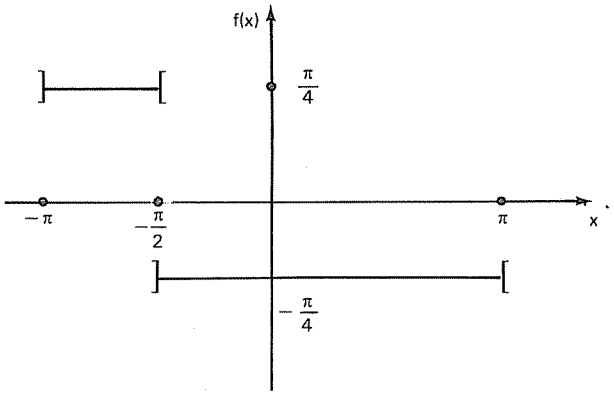
On observerait un tel signal grâce à un générateur de signaux :

rapport cyclique :

$\eta = \frac{t}{T}$



Nous retiendrons donc:




```
1900 Y=0

1910 IF X>-PI AND X<-PI/2 THEN Y=PI/4

1920 IF X>-PI/2 AND X<PI THEN Y=-PI/4
```

Les coefficients de série sont:

$$a_0 = -\frac{\pi}{8}$$

$$a_n = \begin{cases} -\frac{1}{2n} & \text{si } n = 4p+1 \\ \frac{1}{2n} & \text{si } n = 4p+3 \\ 0 & \text{si } n \text{ est pair} \end{cases}$$

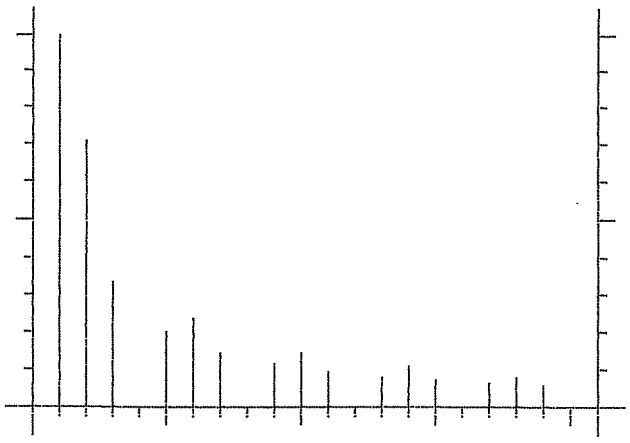
$$b_n = \begin{cases} 0 & \text{si } n = 4p \\ -\frac{1}{2n} & \text{si } n = 4p+1 \\ \frac{1}{n} & \text{si } n = 4p+2 \\ -\frac{1}{2n} & \text{si } n = 4p+3 \end{cases}$$

Les résultats obtenus avec une précision demandée de 4 sont exacts à 10^{-7} près:

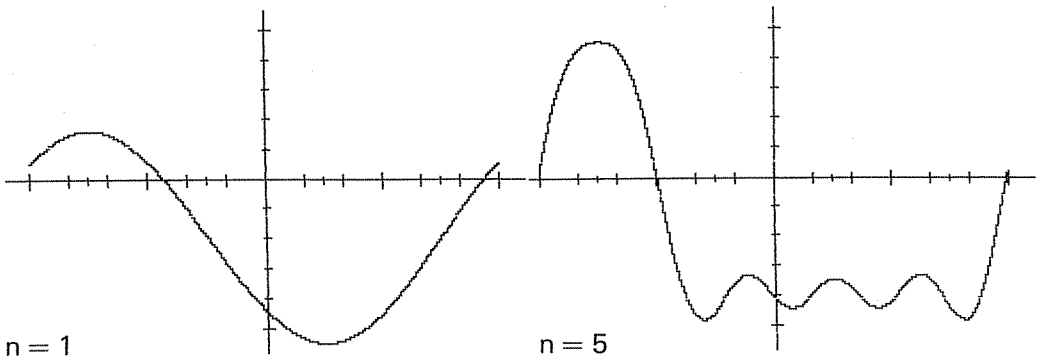
```
CALCUL DES COEFFICIENTS


NOMBRE DE TERMES A CALCULER:20
```

```
PRECISION (1 A 8):4

A0=-.392699082
A1=-.499999998      B1=-.49999993
A2=0                B2=.49999999
A3=.166666667       B3=-.166666667
A4=0                B4=0
A5=-.0999999885     B5=-.0999999894
A6=0                B6=.166666667
A7=.0714285716      B7=-.0714285716
A8=0                B8=0
A9=-.0555555521     B9=-.0555555519
A10=0               B10=.0999999893
A11=.0454545339     B11=-.0454545337
A12=0               B12=0
A13=-.0384614755    B13=-.0384614765
A14=0               B14=.0714285715
A15=.0333333339     B15=-.0333333342
A16=0               B16=0
A17=-.0294117628    B17=-.0294117634
A18=0               B18=.0555555527
A19=.0263157874     B19=-.0263157872
A20=0               B20=0

VOULEZ-VOUS REUTILISER LE PROGRAMME:N
```

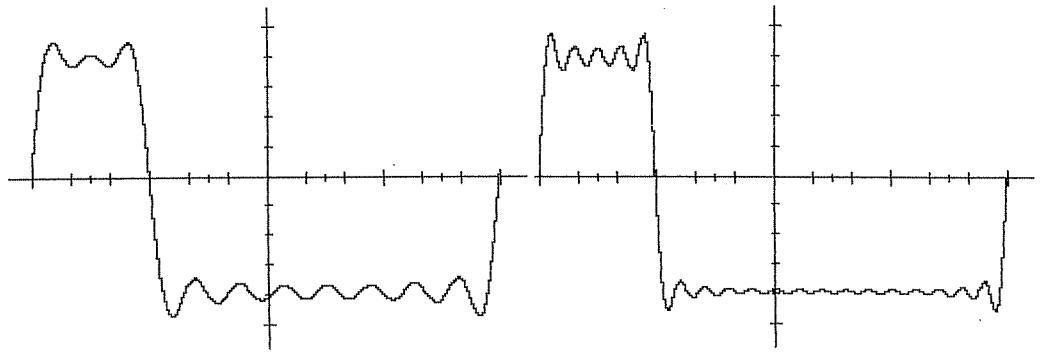
Représentation du spectre :



Reconstitution du signal :



n = 1 n = 5



n = 10 n = 20

8 Équations différentielles

1. Introduction

Une équation différentielle est une équation dans laquelle interviennent, en plus de la variable x et de la fonction y, les dérivées successives de la fonction y, jusqu'à un ordre n, appelé ordre de l'équation différentielle. Ainsi, une équation différentielle se présente sous la forme générale :

$$G(x,y,y',y'',\ldots,y^{(n)}) = 0$$

Résoudre l'équation différentielle, c'est trouver l'ensemble des fonctions y vérifiant l'équation ci-dessus.

On ne sait actuellement résoudre les équations différentielles de manière exacte que sur des cas particuliers, et il faut le plus souvent recourir aux méthodes numériques pour obtenir les solutions d'une équation quelconque.

Nous présenterons ici des méthodes et des programmes permettant la résolution des équations les plus fréquemment rencontrées.

Les deux premières méthodes, celle de Runge-Kutta et celle dite prédicteur-correcteur concernent les équations du premier ordre.

Le troisième programme résoud les systèmes d'équations différentielles (non nécessairement linéaires).

Nous verrons enfin l'extension de la méthode de Runge-Kutta aux équations d'ordre quelconque.

Pour utiliser les quatre programmes simultanément, il faut remplacer chaque instruction END par l'instruction GOTO 1ØØ et ajouter le menu :

```
100 CLS
110  PRINT  TAB( 5);"EQUATIONS DIFFERENTIELLES": PRINT

120  PRINT : PRINT "1- Y'=F(X,Y) METHODE DE RUNGE-KUTTA
     "
130  PRINT : PRINT "2- Y'=F(X,Y) PREDICTEUR-CORRECTEUR"

140  PRINT : PRINT "3- EQUATION D'ORDRE N"
150  PRINT : PRINT "4- SYSTEMES DE N EQUATIONS DE DEGRE
     1"
160  PRINT : PRINT "5- FIN"
170  PRINT : INPUT "VOTRE CHOIX:";E
180  ON E GOTO 1000,2000,3000,4000,5000
190  GOTO 100
5000  END
```

L'ensemble nécessite environ 5 Koctets de mémoire.

2. *Rappels théoriques*

a. Les conditions initiales

Une équation différentielle n'admet généralement pas une solution unique, mais une infinité de solutions.

La détermination d'*une* fonction solution nécessite donc l'apport d'informations supplémentaires sur la fonction, que l'on appelle le plus souvent *conditions initiales*.

Soit x_0 un point quelconque. Les conditions initiales sont alors :

- pour une équation du premier ordre, la valeur de la fonction en x_0, soit $y(x_0)$,
- pour une équation d'ordre n, $y(x_0)$ et les valeurs des $(n-1)$ premières dérivées de y au point x_0, soit $y'(x_0)$, $y''(x_0)$, ..., $y^{(n-1)}(x_0)$.

Les conditions initiales étant fixées, il ne reste généralement plus qu'une solution.

En fait, les problèmes physiques faisant intervenir des équations différentielles font apparaître naturellement ces conditions initiales.

Par exemple, un objet en chute libre dans l'air obéit à laloi :

$$mz'' + kz' - g = 0$$

Les conditions initiales sont alors la position $z_0 = z(t_0)$ et la vitesse $z'_0 = z'(t_0)$ du mobile à l'instant origine t_0, valeurs indispensables à la détermination complète de la loi du mouvement.

b. Equations résolues

Toutes les équations auxquelles nous allons nous intéresser sont des équations dites *résolues*. Cela signifie que la dérivée d'ordre n peut s'exprimer au moyen des $(n-1)$ autres dérivées, de la fonction et de la variable.

Une équation résolue se présente donc sous la forme :

$$y^{(n)} = F(y^{(n-1)}, y^{(n-2)}, \ldots, y', y, x)$$

Une équation telle que :

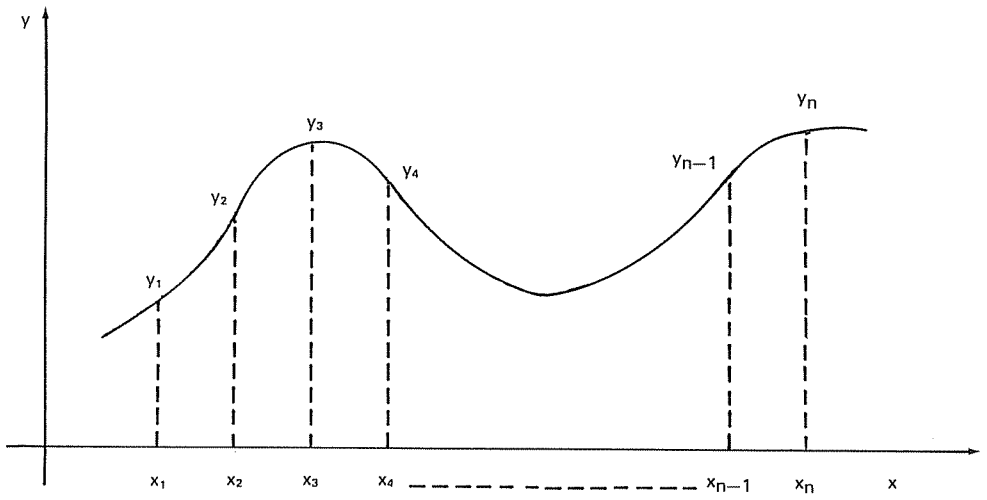
$$y'' + e^{y''} = y' - y$$

n'est pas une équation résolue et ne pourra donc pas être traitée par les méthodes exposées dans ce chapitre.

c. Forme analytique et forme discrète

La résolution théorique d'une équation différentielle fournit la solution sous sa forme *analytique*.

La solution fournie par le programme sera sous forme *discrète*, c'est-à-dire sous forme d'une suite de valeurs prises par la fonction pour des valeurs équidistantes de la variable x.



La solution sera la suite des valeurs $y_1, y_2, ..., y_n$.

3. *Méthode de Runge-Kutta appliquée aux équations du premier ordre*

a. Introduction

L'équation à résoudre se présente donc sous la forme :

$$y' = f(y,x)$$

avec la condition initiale en un point x_0, soit :

$$y(x_0) = y_0$$

Nous allons de plus fixer la valeur finale x_f de la variable.

Il s'agit donc de déterminer la solution pour des valeurs de la variable comprises entre x_0 et x_f.

Nous savons que la réponse de l'ordinateur est une fonction discrète, définie pour un certain nombre de valeurs de la variable.

Il faut donc fixer le nombre n de points calculés entre x_0 et x_f. On peut alors définir un "pas de calcul" h égal à la différence entre deux valeurs successives de x.

En résumé, les données sont :

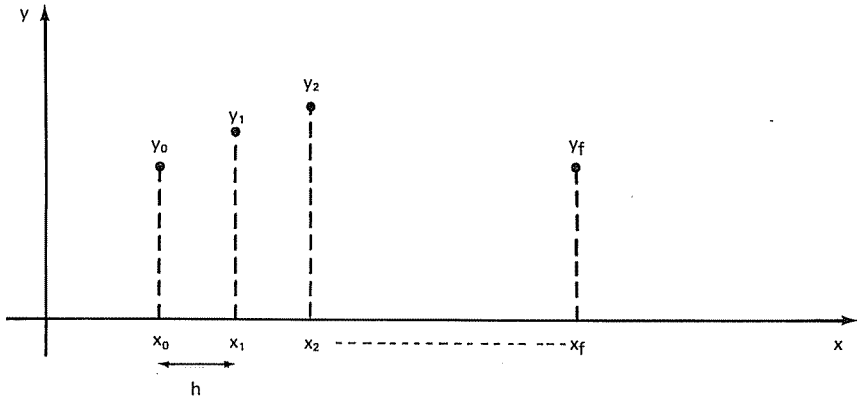
→ l'équation $y' = f(x,y)$,

→ le point initial (x_0,y_0),

→ l'abscisse finale x_f,

→ le nombre de points à calculer,

et le résultat est l'ensemble des valeurs y_i.



Avant d'étudier la méthode de Runge-Kutta, nous verrons le principe de la méthode d'Euler, d'une interprétation plus facile, sans toutefois donner un programme, car les résultats fournis par cette méthode sont médiocres.

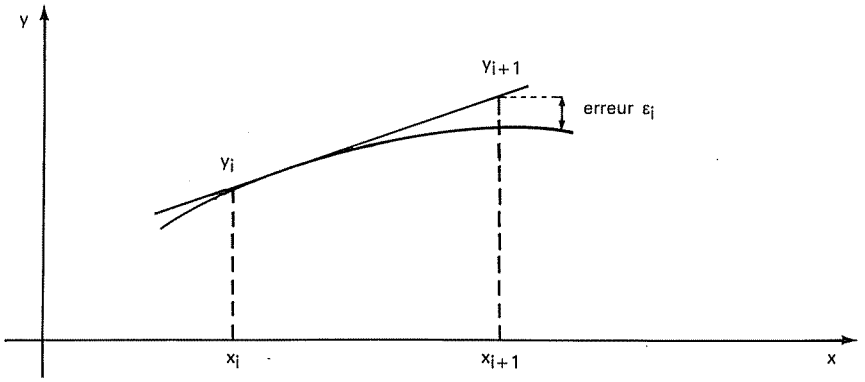
b. Méthode d'Euler

Supposons un point (x_0,y_0) connu.

La méthode d'Euler consiste alors à poser :

$$y_1 = y_0 + h\, f(y_0,x_0)$$
$$y_2 = y_1 + h\, f(y_1,x_1)$$
.....
$$y_n = y_{n-1} + h\, f(y_{n-1},x_{n-1})$$

Le principe consiste donc à assimiler la courbe à sa tangente en chaque point x_i, puisque $f(y_i,x_i)$ est le coefficient directeur de cette tangente.



Cette méthode est simple et, contrairement aux méthodes suivantes, présente l'avantage d'une interprétation graphique évidente. Cependant, les erreurs commises à chaque pas se cumulent et elle n'est jamais utilisée en pratique du fait de son imprécision.

c. Méthode de Runge-Kutta

La méthode présentée ici est une méthode d'ordre 4, ce qui signifie que le développement de y autour de x_n coïncide avec son développement de Taylor à l'ordre 4 :

$$y_{n+1} = y_n + h\,y'_n + \frac{h^2}{2}\,y''_n + \frac{h^3}{6}\,y'''_n + \frac{h^4}{24}\,y''''_n$$

Bien entendu, il existe d'autres méthodes selon l'ordre de coïncidence avec la formule de Taylor ; en particulier, on peut remarquer que la méthode d'Euler n'est rien d'autre qu'une méthode de Runge-Kutta d'ordre 1.

On utilise habituellement la formule d'ordre 4 car elle représente un bon compromis entre complexité, rapidité de calcul et précision.

A partir d'un point connu (x_n, y_n), le point suivant est déterminé par les formules suivantes :

$$K_0 = h\,f(x_n, y_n)$$

$$K_1 = h\,f\left(x_n + \frac{h}{2}\,,\, y_n + \frac{K_0}{2}\right)$$

$$K_2 = h\,f\left(x_n + \frac{h}{2}\,,\, y_n + \frac{K_1}{2}\right)$$

$$K_3 = h\,f(x_n + h,\, y_n + K_2)$$

alors :

$$y_{n+1} = y_n + \frac{1}{6}\,(K_0 + 2K_1 + 2K_2 + K_3)$$

La signification des formules n'est plus du tout intuitive comme dans le cas de la méthode d'Euler.

Par contre, la précision est bien meilleure. On peut montrer que l'erreur sur un pas, pour la méthode d'Euler, est de l'ordre de h^2 ; cette erreur est de l'ordre de h^5 pour la méthode de Runge-Kutta.

Le principal inconvénient de cette dernière est la longueur du temps de calcul. Quatre évaluations de la fonction sont en effet nécessaires pour chaque pas, et prennent d'autant plus de temps que la fonction est compliquée.

d. Programme

- La fonction doit être programmée à partir de la ligne 28ØØ, sous la forme :

  ```
  28ØØ F=4*Y+X
  ```

  où Y désigne la fonction et X la variable.

- Il faut ensuite lancer le programme qui demande :
  - → le point de départ (x_0, y_0),
  - → la deuxième borne de l'intervalle d'étude x_f,
  - → le nombre de points à afficher n,
  - → la finesse m.

- La *finesse* est un moyen d'augmenter la précision du calcul. La précision est bien sûr directement liée au nombre de termes calculés.

  Pour bénéficier d'une meilleure précision sans toutefois saturer l'écran de résultats inutiles, une partie seulement des points calculés est affichée.

  La finesse représente alors le nombre de points calculés entre deux points affichés. Pour une finesse de 5, par exemple, le programme calculera cinq fois plus de points qu'il n'en affichera. Si par contre on désire que tous les points calculés soient affichés, il suffit de demander une finesse de 1.

```
1000 CLS
1010  PRINT  TAB( 5);"Y'=F(X,Y) METHODE DE RUNGE-KUTTA"
    : PRINT : PRINT
1020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
1030  PRINT "2- RESOLUTION": PRINT : PRINT
1040  INPUT "VOTRE CHOIX:";E: PRINT
1050  IF E = 1 THEN  LIST 2800 - 2890: END
1060  INPUT "X0=";X
1070  INPUT "Y0=";Z
1080  INPUT "DEUXIEME BORNE XF:";XF
1090  INPUT "NOMBRE DE POINTS:";M
1100  INPUT "FINESSE:";FI: PRINT
1110 X1 = X
1120 H = (XF - X) / FI / M
1130  FOR I = 1 TO M
1140  FOR J = 1 TO FI
1150 X = X1 + ((I - 1) * FI + J - 1) * H
1160  GOSUB 2500
1170  NEXT J
```

```
1180  PRINT "X=";X; TAB( 20);"Y=";Z
1190  NEXT I
1200  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
1210  IF Z$ = "O" THEN 1000
1220  END
2500 XX = X:Y = Z: GOSUB 2800:A = H * F
2510 X = X + H / 2:Y = Z + A / 2: GOSUB 2800:B = H * F
2520 Y = Z + B / 2: GOSUB 2800:C = H * F
2530 X = XX + H:Y = Z + C: GOSUB 2800
2540 Z = Z + (A + 2 * B + 2 * C + H * F) / 6
2550  RETURN
2800 F =  - Y * Y
2890  RETURN
```

e. Exemples commentés

• *Exemple 1* :

Nous allons chercher la solution d'une équation de Bernoulli :

$$y' = \frac{4xy + 4x\sqrt{y}}{1 + x^2} \qquad \text{avec } y(0) = 1$$

La solution exacte de cette équation est :

$$y = (2x^2 + 1)^2$$

Ainsi :

$$y(10) = 40\,401$$

Effectuons une première recherche entre 0 et 10, avec une finesse de 1 :

```
2800 F=4*X*(Y+SQR(Y))/(1+X*X)


   Y'=F(X,Y) METHODE DE RUNGE-KUTTA


XO=0
YO=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:10
FINESSE:1
```

```
X=1             Y=7.82063886
X=2             Y=67.5300668
X=3             Y=298.711912
X=4             Y=900.02937
X=5             Y=2149.79618
X=6             Y=4405.83497
X=7             Y=8105.4573
X=8             Y=13765.4663
X=9             Y=21982.1609
X=10            Y=33431.3388

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
```

On remarque que l'ordre de grandeur est correct mais la précision médiocre.

Effectuons une deuxième recherche avec une finesse de 10:

```
Y'=F(X,Y) METHODE DE RUNGE-KUTTA


XO=0
YO=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:10
FINESSE:10

X=1             Y=8.99965055
X=2             Y=80.9951207
X=3             Y=360.977528
X=4             Y=1088.93232
X=5             Y=2600.83903
X=6             Y=5328.67132
X=7             Y=9800.39699
X=8             Y=16639.978
X=9             Y=26567.3704
X=10            Y=40398.5245

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La précision est cette fois correcte.

Le temps de calcul dépend bien sûr de la fonction et de l'ordinateur, mais reste généralement de l'ordre de grandeur de la seconde par point *calculé*.

Ainsi, une finesse de 10 représente un bon compromis entre précision et temps de calcul puisque l'ordinateur affiche un point toutes les dix secondes environ.

Une finesse de 100 ou plus donnera bien entendu un résultat extrêmement précis, mais nécessitera plusieurs minutes de calcul par point affiché.

- ***Exemple 2*** :

L'équation étudiée est la suivante :

$y' = -y^2$ avec $y(1) = 1$

La solution exacte est :

$$y = \frac{1}{x}$$

```
 2800 F=-Y*Y


    Y'=F(X,Y) METHODE DE RUNGE-KUTTA


XO=1
YO=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:9
FINESSE:10

X=2             Y=.500000298
X=3             Y=.333333479
X=4             Y=.250000084
X=5             Y=.200000054
X=6             Y=.166666704
X=7             Y=.142857171
X=8             Y=.125000021
X=9             Y=.111111128
X=10            Y=.100000014

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    Y'=F(X,Y) METHODE DE RUNGE-KUTTA


XO=1
YO=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:9
FINESSE:50
```

```
X=2             Y=.5
X=3             Y=.333333334
X=4             Y=.25
X=5             Y=.2
X=6             Y=.166666666
X=7             Y=.142857143
X=8             Y=.125
X=9             Y=.111111111
X=10            Y=.1

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Les résultats sont ici encore satisfaisants avec une finesse de 10, et sont exacts avec une finesse de 50.

4. Méthode prédicteur-correcteur

a. Principe

Dans la méthode de Runge-Kutta, le terme y_{n+1} est calculé à partir du seul point (x_n,y_n).

Ainsi, de tous les points calculés auparavant, seul le dernier est utilisé.

Ce type de méthode est appelé méthode à "pas unique" ou à "pas séparés".

Les méthodes que nous allons examiner maintenant sont des méthodes à "pas multiples" ou à "pas liés", c'est-à-dire que le calcul de y_{n+1} fait intervenir non seulement y_n, mais aussi y_{n-1}, y_{n-2},..., le nombre de points utilisés dépendant de la formule considérée.

On distingue deux types de formules:

→ les formules explicites, de la forme:

$$y_{n+1} = y_n + h[a_0y'_n + a_1y'_{n-1} + \ldots + a_ky'_{n-k}]$$

→ les formules implicites, de la forme:

$$y_{n+1} = y_n + h[b_{-1}y'_{n+1} + b_0y'_n + \ldots + b_ky'_{n-k}]$$

Dans les formules implicites, la valeur $y'_{n+1} = f(y_{n+1},x_{n+1})$ qui intervient n'est pas connue, puisque y_{n+1} n'est pas encore calculée.

Il faut alors injecter une première valeur estimée ${}^0y'_{n+1}$ dans la formule, ce qui permet de calculer ${}^1y_{n+1}$; on réinjecte cette valeur dans l'équation pour calculer

${}^{1}y'_{n+1}$, et ainsi de suite jusqu'à obtenir deux valeurs ${}^{i-1}y_{n+1}$ et ${}^{i}y_{n+1}$ suffisamment proches.

L'intérêt des méthodes implicites n'est pas évident à première vue, puisqu'elles nécessitent plusieurs itérations, au lieu d'une seule pour les méthodes explicites, donc un temps de calcul plus important.

En fait, ces méthodes sont bien plus précises que les méthodes explicites du même ordre ; de plus, elles sont en général stables, alors que les méthodes explicites sont souvent instables.

Les formules que nous retiendrons seront d'ordre 4 (comme pour la méthode de Runge-Kutta) :

→ formule explicite :

$$y_{n+1} = y_n + \frac{h}{24}[55y'_n - 59y'_{n-1} + 37y'_{n-2} - 9y'_{n-3}]$$

→ formule implicite :

$$y_{n+1} = y_n + \frac{h}{24}\ [9y'_{n+1} + 19y'_n - 5y'_{n-1} + y'_{n-2}]$$

Nous n'utiliserons en fait ni la formule explicite, ni la formule implicite, mais une combinaison des deux, de manière à bénéficier des avantages de l'une et de l'autre : c'est la méthode prédicteur-correcteur.

En effet, nous avons vu que l'inconvénient des formules implicites était la nécessité d'effectuer plusieurs itérations. Mais, si la valeur injectée ${}^{0}y'_{n+1}$ est proche de la valeur réelle, le nombre d'itérations est faible (de l'ordre de 2 ou 3).

Ainsi, nous procéderons en deux étapes :

→ l'estimation initiale ${}^{0}y'_{n+1}$ sera calculée par la méthode explicite ;

→ cette valeur sera injectée dans la formule implicite, qui ne nécessitera alors que quelques itérations.

Les formules à pas liés présentent l'inconvénient de ne pas "démarrer" seules. En effet, seule la valeur y_0 est connue initialement, alors que le calcul de y_1 nécessiterait la connaissance de $y_0, y_{-1}, y_{-2}, y_{-3}$. Les premières valeurs y_1, y_2, y_3 seront donc calculées par la méthode de Runge-Kutta, ce qui permettra d'initialiser le processus.

En résumé, l'algorithme utilisé est le suivant :

→ calcul de y_1, y_2 et y_3 par la méthode de Runge-Kutta pour initialiser le processus,

→ à partir du point (x_n, y_n), calcul du point suivant par les formules :

a) $^{0}y_{n+1} = y_n + \frac{h}{24}\ (55y'_n - 59y'_{n-1} + 37y'_{n-2} - 9y'_{n-3})$

calcul du prédicteur

b) $^{1}y_{n+1} = y_n + \frac{h}{24}(9\ ^{0}y'_{n+1} + 19y'_n - 5y'_{n-1} + y'_{n-2})$

puis jusqu'à convergence :

$$^{r+1}y_{n+1} = y_n + \frac{h}{24}\ (9\ ^{r}y'_{n+1} + 19y'_n - 5y'_{n-1} + y'_{n-2})$$

b. Programme

L'utilisation est en tout point similaire à celle du programme précédent.

```
2000 CLS
2010  PRINT  TAB( 5);"PREDICTEUR-CORRECTEUR": PRINT : PRINT

2020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
2030  PRINT "2- RESOLUTION": PRINT : PRINT
2040  INPUT "VOTRE CHOIX:";E: PRINT
2050  IF E = 1 THEN  LIST 2800 - 2890: END
2060  INPUT "X0=";X
2070  INPUT "Y0=";Z
2080  INPUT "DEUXIEME BORNE XF:";XF
2090  INPUT "NOMBRE DE POINTS:";M
2100  INPUT "FINESSE:";FI: PRINT
2110 H = (XF - X) / FI / M
2120 K = 0:X1 = X
2130 Y = Z: GOSUB 2800:P(3) = F
2140  FOR I = 1 TO M
2150  FOR J = 1 TO FI
2160 K = K + 1:X = X1 + ((I - 1) * FI + J - 1) * H
2170  IF K < 4 THEN  GOSUB 2500:Y = Z: GOSUB 2800:P(3 -
    K) = F: GOTO 2250
2180 X = X + H
2190 Y = Z + H / 24 * (55 * P(0) - 59 * P(1) + 37 * P(2
    ) - 9 * P(3))
```

```
2200  GOSUB 2800
2210 W = Z + H / 24 * (9 * F + 19 * P(0) - 5 * P(1) + P
    (2))
2220  IF  ABS (Y - W) > 1E - 8 THEN Y = W: GOTO 2200
2230 Z = W:P(3) = P(2):P(2) = P(1):P(1) = P(0)
2240 Y = Z: GOSUB 2800:P(0) = F
2250  NEXT J
2260  PRINT "X=";X; TAB( 20);"Y=";Z
2270  NEXT I
2280  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
   E?:";Z$
2290  IF Z$ = "O" THEN 2000
2300  END
2500 XX = X:Y = Z: GOSUB 2800:A = H * F
2510 X = X + H / 2:Y = Z + A / 2: GOSUB 2800:B = H * F
2520 Y = Z + B / 2: GOSUB 2800:C = H * F
2530 X = XX + H:Y = Z + C: GOSUB 2800
2540 Z = Z + (A + 2 * B + 2 * C + H * F) / 6
2550  RETURN
2800 F =  - Y * Y
2890  RETURN
```

c. Exemples commentés

▫ *Exemple 1* :

Reprenons l'équation de Bernoulli que nous avons résolue par la méthode de Runge-Kutta :

```
2800 F=4*X*(Y+SQR(Y))/(1+X*X)
```

```
   PREDICTEUR-CORRECTEUR


X0=0
Y0=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:10
FINESSE:10

X=1              Y=8.9999841
X=2              Y=80.9998808
```

```
X=3                 Y=360.999497
X=4                 Y=1088.99851
X=5                 Y=2600.99649
X=6                 Y=5328.99285
X=7                 Y=9800.9869
X=8                 Y=16640.9778
X=9                 Y=26568.9646
X=10                Y=40400.9463

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Avec le même nombre de points calculés, la précision est cinquante fois meilleure sur la dernière valeur.

- ***Exemple 2*** :

Dans certains cas, en particulier dans l'exemple précédent, la méthode prédicteur-correcteur donne de meilleurs résultats que la méthode de Runge-Kutta.

Par contre, dans d'autres cas, le phénomène inverse se produit. Reprenons l'équation :

$y' = -y^2$ avec $y(1) = 1$

dont la solution exacte est $y = \frac{1}{x}$.

```
2800 F=-Y*Y
```

```
    PREDICTEUR-CORRECTEUR


XO=1
YO=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:9
FINESSE:10

X=2                 Y=.499988002
X=3                 Y=.333326726
X=4                 Y=.249996121
X=5                 Y=.199997481
X=6                 Y=.166664907
X=7                 Y=.142855845
```

```
X=8                Y=.124999005
X=9                Y=.111110324
X=10               Y=.0999993618

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
    PREDICTEUR-CORRECTEUR


X0=1
Y0=1
DEUXIEME BORNE XF:10
NOMBRE DE POINTS:9
FINESSE:50

X=2                Y=.499999967
X=3                Y=.333333317
X=4                Y=.24999999
X=5                Y=.199999994
X=6                Y=.166666662
X=7                Y=.14285714
X=8                Y=.124999997
X=9                Y=.111111109
X=10               Y=.0999999984

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

La précision est cette fois meilleure avec la méthode de Runge-Kutta.

Dans les programmes suivants, nous généraliserons la méthode de Runge-Kutta plutôt que la méthode prédicteur-correcteur, bien que cette dernière fournisse parfois de meilleurs résultats.

5. Système d'équations couplées

a. Principe

Il s'agit maintenant de déterminer non plus *une* mais *n* fonctions, solutions d'un système différentiel du premier ordre de la forme :

$$y'_1 = f_1(x, y_1, y_2, \ldots, y_n)$$
$$y'_2 = f_2(x, y_1, y_2, \ldots, y_n)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$y'_n = f_n(x, y_1, y_2, \ldots, y_n)$$

Les équations sont ici encore sous leur forme résolue.

Le système peut s'écrire plus simplement en utilisant le formalisme vectoriel :

posons : On a alors :

$$\vec{y} = \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{pmatrix} \qquad \vec{y'} = \begin{pmatrix} y'_1 \\ y'_2 \\ . \\ . \\ . \\ y'_n \end{pmatrix}$$

Le système s'écrit alors :

$$\vec{y'} = f(\vec{x},\vec{y})$$

La résolution se fait de la même manière que pour une équation différentielle du premier ordre, la seule différence étant le caractère vectoriel de l'inconnue $\vec{y}$ et de la fonction $\vec{f}$.

b. Programme

Le programme est semblable à celui de résolution d'une équation du premier ordre par la méthode de Runge-Kutta.

Les seules différences sont :

→ la programmation du système. Les inconnues sont représentées par Y(1), Y(2), Y(3), ..., et le système doit être programmé à partir de la ligne 45ØØ de la manière suivante :

```
45ØØ F(1) = Y(1) – Y(3) + X
451Ø F(2) = –Y(2) + SIN(X)
.....
```

Le nombre d'équations (et donc d'inconnues) doit être indiqué à la ligne 46ØØ. ligne 46ØØ.

→ il faut fournir au programme les valeurs initiales de toutes les inconnues, soit :

$y_1(x_0), y_2(x_0), ..., y_n(x_0)$

```
4000 CLS
4010  PRINT  TAB( 3);"SYSTEME DE N EQUATIONS DE DEGRE 1
    ": PRINT : PRINT
4020  PRINT "1- PROGRAMMATION DU SYSTEME": PRINT
4030  PRINT "2- RESOLUTION": PRINT : PRINT
```

```
4040  INPUT "VOTRE CHOIX:";E: PRINT
4050  IF E = 1 THEN  LIST 4500 - 4600: END
4060  INPUT "X0=";X: PRINT
4070  GOSUB 4600
4080  FOR L = 1 TO N
4090  PRINT "Y";L;"(";X;: INPUT ")=";Z(L)
4100  NEXT L: PRINT
4110  INPUT "DEUXIEME BORNE XF:";XF
4120  INPUT "NOMBRE DE POINTS:";M
4130  INPUT "FINESSE:";FI
4140 H = (XF - X) / FI / M
4150  FOR I = 1 TO M
4160  FOR J = 1 TO FI
4170  FOR L = 1 TO N:Y(L) = Z(L): NEXT L
4180  GOSUB 4500
4190  FOR L = 1 TO N:A(L) = H * F(L):Y(L) = Z(L) + A(L)
     / 2: NEXT L
4200 X = X + H / 2
4210  GOSUB 4500
4220  FOR L = 1 TO N:B(L) = H * F(L):Y(L) = Z(L) + B(L)
     / 2: NEXT L
4230  GOSUB 4500
4240  FOR L = 1 TO N:C(L) = H * F(L):Y(L) = Z(L) + C(L)
    : NEXT L
4250 X = X1 + ((I - 1) * FI + J) * H
4260  GOSUB 4500
4270  FOR L = 1 TO N:Z(L) = Z(L) + (A(L) + 2 * B(L) + 2
     * C(L) + H * F(L)) / 6: NEXT L
4280  NEXT J
4290  PRINT : PRINT "X=";X
4300  FOR L = 1 TO N: PRINT "Y";L;"=";Z(L): NEXT L
4310  NEXT I
4320  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
4330  IF Z$ = "O" THEN 4000
4340  END
4500 F(1) = Y(1) + Y(2) +  SIN (X)
4510 F(2) =  - Y(1) + 3 * Y(2)
4590  RETURN
4600 N = 2
4610  RETURN
```

c. Exemples commentés

• *Exemple 1* :

Soit le système :

$$\begin{cases} y'_1 = y_2 + y_3 - 3y_1 \\ y'_2 = y_1 + y_3 - 3y_2 \\ y'_3 = y_1 + y_2 - 3y_3 \end{cases} \quad \text{avec} \quad \begin{cases} y_1(0) = 1 \\ y_2(0) = 2 \\ y_3(0) = -1 \end{cases}$$

La solution exacte est :

$$y_1 = \frac{1}{3}(e^{-4x} + 2e^{-x})$$

$$y_2 = \frac{1}{3}(4e^{-4x} + 2e^{-x})$$

$$y_3 = \frac{1}{3}(-5e^{-4x} + 2e^{-x})$$

```
4500 F(1)=Y(2)+Y(3)-3*Y(1)

4510 F(2)=Y(1)+Y(3)-3*Y(2)

4520 F(3)=Y(1)+Y(2)-3*Y(3)

4600 N=3
```

```
 SYSTEME DE N EQUATIONS DE DEGRE 1


X0=0

Y1(0)=1
Y2(0)=2
Y3(0)=-1

DEUXIEME BORNE XF:3
NOMBRE DE POINTS:6
FINESSE:10

X=.5
Y1=.449466967
Y2=.584806516
Y3=.17878787
```

```
X=1
Y1=.251358572
Y2=.269675365
Y3=.214724985

X=1.5
Y1=.149579781
Y2=.152058768
Y3=.144621808

X=2
Y1=.090335367
Y2=.0906708719
Y3=.0896643571

X=2.5
Y1=.0547384755
Y2=.0547838826
Y3=.0546476614

X=3
Y1=.0331934328
Y2=.0331995782
Y3=.033181142

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
  SYSTEME DE N EQUATIONS DE DEGRE 1


XO=0

Y1(0)=1
Y2(0)=2
Y3(0)=-1

DEUXIEME BORNE XF:3
NOMBRE DE POINTS:6
FINESSE:100

X=.5
Y1=.449465534
Y2=.584800817
Y3=.178794967

X=1
Y1=.251358173
```

```
Y2=.269673812
Y3=.214726896

X=1.5
Y1=.14957969
Y2=.152058443
Y3=.144622186

X=2
Y1=.0903353428
Y2=.0906708055
Y3=.0896644176

X=2.5
Y1=.0547384656
Y2=.0547838655
Y3=.0546476658

X=3
Y1=.0331934269
Y2=.0331995711
Y3=.0331811385

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On peut vérifier que les résultats obtenus avec une finesse de 100 sont exacts à 10^{-10} près et remarquer que les résultats obtenus avec une finesse de 10 sont déjà précis.

- ***Exemple 2*** :

Phénomène de divergence

Considérons le système suivant :

$$y'_1 = y_1 + y_2 + \sin x$$
$$y'_2 = -y_1 + 3y_2$$

Les solutions sont du type :

$$y_1 = (\lambda + \mu - \lambda x)\, e^{2x} - \frac{1}{25}\,(13 \sin x + 9 \cos x)$$

$$y_2 = (\mu - \lambda x)\, e^{2x} - \frac{1}{25}\,(3 \sin x + 4 \cos x)$$

où λ et μ sont des constantes arbitraires déterminant les conditions initiales.

Ainsi, si nous imposons comme conditions initiales :

$$y_1(0) = -\frac{9}{25} \qquad \text{et} \qquad y_2(0) = -\frac{4}{25},$$

nous devons obtenir la solution :

$$y_1 = -\frac{1}{25}(13 \sin x + 9 \cos x)$$

$$y_2 = -\frac{1}{25}(3 \sin x + 4 \cos x)$$

qui est une solution périodique.

Or, la solution fournie par le programme n'est pas périodique et diverge.

Ceci provient du fait que les résultats numériques ne sont pas exacts, et que les termes en $x\,e^{2x}$ qui devraient être inexistants deviennent peu à peu prépondérants, jusqu'à masquer totalement la solution recherchée.

Ce phénomène se manifeste relativement souvent avec les systèmes d'équations différentielles. Il faut alors pour obtenir un résultat correct reprendre le calcul avec une meilleure finesse.

```
4500 F(1)=Y(1)+Y(2)+SIN(X)

4510 F(2)=-Y(1)+3*Y(2)

4600 N=2


 SYSTEME DE N EQUATIONS DE DEGRE 1


X0=0

Y1(0)=-.36
Y2(0)=-.16

DEUXIEME BORNE XF:15.70796327
NOMBRE DE POINTS:10
FINESSE:20

X=1.57079633
Y1=-.519991942
Y2=-.119983531
```

```
X=3.14159266
Y1=.360507041
Y2=.160711859

X=4.71238898
Y1=.539169775
Y2=.143900923

X=6.28318531
Y1=.255548687
Y2=.565019039

X=7.85398164
Y1=17.7030622
Y2=20.636255

X=9.42477797
Y1=514.124937
Y2=572.543963

X=10.9955743
Y1=14019.9157
Y2=15375.9816

X=12.5663706
Y1=373718.477
Y2=405107.798

X=14.1371669
Y1=9788923.64
Y2=10515279.9

X=15.7079633
Y1=252920914
Y2=269729056

VOULEZ-VOUS REUTILISER LE PROGRAMME?:O
SYSTEME DE N EQUATIONS DE DEGRE 1

X0=0

Y1(0)=-.36
Y2(0)=-.16
```

```
DEUXIEME BORNE XF:15.70796327
NOMBRE DE POINTS:10
FINESSE:2000

X=1.57079633
Y1=-.519999998
Y2=-.119999998

X=3.14159265
Y1=.360000036
Y2=.160000029

X=4.71238898
Y1=.52000059
Y2=.120000457

X=6.28318531
Y1=-.359991183
Y2=-.15999425

X=7.85398164
Y1=-.519907459
Y2=-.119978408

X=9.42477797
Y1=.359562543
Y2=.157920752

X=10.9955743
Y1=.45019896
Y2=.0122067417

X=12.5663706
Y1=-3.35623554
Y2=-4.03540177

X=14.1371669
Y1=-101.812056
Y2=-121.756573

X=15.7079633
Y1=-3083.11758
Y2=-3554.10378

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

• ***Exemple 3*** :

Résolution d'un système d'équations d'ordre supérieur à 1.

Soit à résoudre le système suivant :

$$\begin{cases} z''_1 = -4z_1 - 3z_2 \\ z''_2 = -8z_1 - 2z_2 \end{cases} \quad \text{avec} \quad \begin{cases} z_1(0) = 3 \\ z_2(0) = 4 \end{cases} \quad \begin{matrix} z'_1(0) = 0 \\ z'_2(0) = 0 \end{matrix}$$

Ramenons-nous à un système d'équations d'ordre 1 en posant :

$$\begin{cases} y_1 = z_1 \\ y_2 = z'_1 \end{cases} \qquad \begin{cases} y_3 = z_2 \\ y_4 = z'_2 \end{cases}$$

Le système initial est alors équivalent au système suivant :

$$\begin{cases} y'_1 = y_2 \\ y'_2 = -4y_1 - 3y_3 \\ y'_3 = y_4 \\ y'_4 = -8y_1 - 2y_3 \end{cases} \quad \text{avec} \quad \begin{cases} y_1(0) = 3 \\ y_2(0) = 0 \\ y_3(0) = 4 \\ y_4(0) = 0 \end{cases}$$

Ce système, du premier ordre, peut alors être résolu par le programme.

La solution exacte est :

$$\begin{cases} y_1 = 3\cos(2\sqrt{2}\,x) \\ y_3 = 4\cos(2\sqrt{2}\,x) \end{cases}$$

La variable décrivant l'intervalle $[0, \frac{4\pi}{2\sqrt{2}}]$, les résultats pour y, et y_3 doivent être :

(0,0), (−3,−4), (0,0), (3,4), (0,0), ...

```
4500 F(1)=Y(2)
4510 F(2)=-4*Y(1)-3*Y(3)
4520 F(3)=Y(4)
4530 F(4)=-8*Y(1)-2*Y(3)
4600 N=4
```

SYSTEME DE N EQUATIONS DE DEGRE 1

X0=0

Y1(0)=3
Y2(0)=0
Y3(0)=4
Y4(0)=0

DEUXIEME BORNE XF:4.442882938
NOMBRE DE POINTS:8
FINESSE:100

X=.555360367
Y1=1.79716153E-09
Y2=-8.48528138
Y3=7.38509698E-09
Y4=-11.3137085

X=1.11072074
Y1=-3
Y2=-1.35914888E-08
Y3=-4
Y4=-2.19442882E-08

X=1.6660811
Y1=-9.55333235E-09
Y2=8.48528139
Y3=-9.48057278E-09
Y4=11.3137085

X=2.22144147
Y1=3.00000001
Y2=3.23634595E-08
Y3=4.00000001
Y4=4.04834282E-08

X=2.77680184
Y1=1.57742761E-08
Y2=-8.4852814
Y3=1.6647391E-08
Y4=-11.3137085

```
X=3.3321622
Y1=-3.00000001
Y2=-5.14555723E-08
Y3=-4.00000001
Y4=-4.57512215E-08

X=3.88752257
Y1=-2.54440238E-08
Y2=8.48528141
Y3=-1.13650458E-08
Y4=11.3137086

X=4.44288294
Y1=3.00000001
Y2=3.70491762E-08
Y3=4.00000006
Y4=1.21071935E-07

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

d. Conclusion

Les phénomènes de divergence constituent le seul problème particulier de la résolution numérique des systèmes d'équations différentielles.

Il convient donc de rester très critique vis-à-vis des résultats obtenus, et ne pas hésiter à reprendre le calcul avec une plus grande finesse en cas de doute.

La résolution des système d'équations différentielles constitue le principe de la résolution des équations différentielles d'ordre quelconque, que nous allons aborder maintenant.

6. Équation différentielle d'ordre quelconque

a. Principe

Nous nous intéressons donc maintenant aux équations de la forme :

$$y^{(n)} = f(y^{(n-1)}, y^{(n-2)}, \ldots, y', y, x)$$

Effectuons le changement de variables suivant :

posons :

$$\begin{cases} y_1 = y \\ y_2 = y' \\ \ldots\ldots\ldots\ldots \\ y_n = y^{(n-1)} \end{cases}$$

L'équation de départ est alors équivalente au système :

$$\begin{cases} y'_1 = y_2 \\ y'_2 = y_3 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y'_{n-1} = y_n \\ y'_n = f(y_n, y_{n-1}, \ldots, y_2, y_1, x) \end{cases} \quad \text{avec} \quad \begin{cases} y_1(0) = y(0) \\ y_2(0) = y'(0) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y_{n-1}(0) = y^{(n-2)}(0) \\ y_n(0) = y^{(n-1)}(0) \end{cases}$$

qui est un système de n équations couplées d'ordre 1 que nous savons résoudre.

b. Programme

L'ordre de l'équation doit être indiqué à la ligne 36ØØ :

```
36ØØ N=2
```

L'équation se programme à la ligne 35ØØ. La fonction inconnue y est représentée par Y(Ø), la dérivée première y' par Y(1), ..., jusqu'à la dérivée Kième représentée par Y(K).

Ainsi, l'équation :

$$y'' = -\sin y$$

se programmera :

```
35ØØ F = -SIN(Y(Ø))
36ØØ N=2
```

De même :

$$y^{(4)} = 3y^{(3)} + y' + 3y - x$$

se programmera :

```
35ØØ F = 3*Y(3)+Y(1)+3*Y(Ø)-X
36ØØ N=4
```

Les conditions initiales à fournir au programme sont ici les n valeurs:

$y(x_0), y'(x_0), ..., y^{(n-1)}(x_0),$

les autres paramètres demandés par le programme ayant la même signification que ceux des programmes précédents.

```
3000 CLS
3010  PRINT  TAB( 5);"EQUATION D'ORDRE N": PRINT : PRINT

3020  PRINT "1- PROGRAMMATION DE L'EQUATION": PRINT
3030  PRINT "2- RESOLUTION": PRINT : PRINT
3040  INPUT "VOTRE CHOIX:";E: PRINT
3050  IF E = 1 THEN  LIST 3500 - 3600: END
3060  GOSUB 3600
3070 N1 = N - 1
3080  INPUT "XO=";X:X1 = X: PRINT
3090  FOR L = 0 TO N1
3100  PRINT "Y";: IF L > 0 THEN  FOR L1 = 1 TO L: PRINT
    "'";: NEXT L1
3110  PRINT "(";X;: INPUT ")=";Z(L)
3120  NEXT L: PRINT
3130  INPUT "DEUXIEME BORNE XF:";XF
3140  INPUT "NOMBRE DE POINTS:";M
3150  INPUT "FINESSE:";FI: PRINT
3160 H = (XF - X) / FI / M
3170  FOR I = 1 TO M
3180  FOR J = 1 TO FI
3190  FOR L = 0 TO N1:Y(L) = Z(L): NEXT L
3200  GOSUB 3500
3210  FOR L = 0 TO N - 2:A(L) = H * Y(L + 1):Y(L) = Z(L
    ) + A(L) / 2: NEXT L
3220 A = H * F:Y(N1) = Z(N1) + A / 2:X = X + H / 2
3230  GOSUB 3500
3240  FOR L = 0 TO N - 2:B(L) = H * Y(L + 1):Y(L) = Z(L
    ) + B(L) / 2: NEXT L
3250 B = H * F:Y(N1) = Z(N1) + B / 2
3260  GOSUB 3500
3270  FOR L = 0 TO N - 2:C(L) = H * Y(L + 1):Y(L) = Z(L
    ) + C(L): NEXT L
3280 C = H * F:Y(N1) = Z(N1) + C:X = X1 + ((I - 1) * FI
     + J) * H
3290  GOSUB 3500
3300  FOR L = 0 TO N - 2:Z(L) = Z(L) + (A(L) + 2 * B(L)
     + 2 * C(L) + H * Y(L + 1)) / 6: NEXT L
```

```
3310 Z(N1) = Z(N1) + (A + 2 * B + 2 * C + H * F) / 6
3320  NEXT J
3330  PRINT "X=";X; TAB( 20);"Y=";Z(0)
3340  NEXT I
3350  PRINT : INPUT "VOULEZ-VOUS REUTILISER LE PROGRAMM
    E?:";Z$
3360  IF Z$ = "O" THEN 3000
3370  END
3500 F = (2 * Y(1) * Y(1) + Y(0) * Y(0)) / Y(0)
3590  RETURN
3600 N = 2
3610  RETURN
```

c. Exemples commentés

- ***Exemple 1*** :

Soit l'équation du second ordre suivante :

$$y'' = \frac{2y'^2 + y^2}{y} \qquad \text{avec} \qquad \begin{cases} y(4) = 2 \\ y'(4) = -2\text{tg } 1 \end{cases}$$

La solution exacte est :

$$y = \frac{2 \cos 1}{\cos (x - 5)}$$

```
3500 F=(2*Y(1)*Y(1)+Y(0)*Y(0))/Y(0)

3600 N=2


   EQUATION D'ORDRE N


XO=4

Y(4)=2
Y'(4)=-3.11481545

DEUXIEME BORNE XF:6
NOMBRE DE POINTS:10
FINESSE:20
```

```
X=4.2                Y=1.55101796
X=4.4                Y=1.30929115
X=4.6                Y=1.17321722
X=4.8                Y=1.10258287
X=5                  Y=1.08060462
X=5.2                Y=1.10258287
X=5.4                Y=1.17321722
X=5.6                Y=1.30929115
X=5.8                Y=1.55101795
X=6                  Y=1.99999999

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

On peut vérifier que les résultats sont exacts à 10^{-9} près.

- ***Exemple 2*** :

Considérons l'équation du troisième ordre :

$$y''' = y \qquad \text{avec} \qquad \begin{cases} y(0) = 2 \\ y'(0) = -1 \\ y''(0) = -1 \end{cases}$$

La solution exacte est :

$$y = 2\, e^{-\frac{x}{2}} \cos\left(\frac{\sqrt{3}}{2} x\right)$$

```
3500 F=Y(0)

3600 N=3

   EQUATION D'ORDRE N

X0=0

Y(0)=2
Y'(0)=-1
Y''(0)=-1

DEUXIEME BORNE XF:-5
NOMBRE DE POINTS:10
FINESSE:50
```

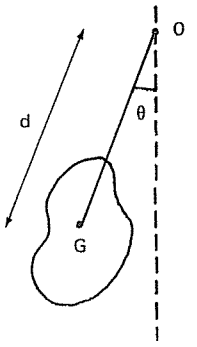
```
X=-.5           Y=2.33103443
X=-1            Y=2.13627896
X=-1.5          Y=1.13651377
X=-2            Y=-.872875848
X=-2.5          Y=-3.90849808
X=-3            Y=-7.67171459
X=-3.5          Y=-11.4390069
X=-4            Y=-14.0161999
X=-4.5          Y=-13.8125106
X=-5            Y=-9.08863072

VOULEZ-VOUS REUTILISER LE PROGRAMME?:N
```

Il est à noter sur cet exemple la possibilité d'introduire une abscisse finale x_f inférieure à l'abscisse initiale x_0.

- ***Exemple 3*** :

Nous allons nous intéresser maintenant à un exemple concret d'application du programme de résolution d'équation d'ordre quelconque.

Le pendule pesant est un système mécanique bien connu des lycéens.



L'élongation angulaire du pendule satisfait à l'équation :

$$\frac{d^2\theta}{dt^2} = -\frac{mgd}{J}\sin\theta = -K\sin\theta.$$

Cette équation ne peut être résolue de manière exacte.

Habituellement, on effectue une résolution approchée en assimilant $\sin\theta$ à θ, en supposant que l'élongation maximale θ_{max} reste suffisamment faible pour que l'approximation soit valable.

La solution approchée est alors une fonction sinusoïdale dont la période est indépendante de θ_{max} et vaut :

$$T_0 = \frac{2\pi}{\sqrt{K}}$$

On peut alors montrer, par une exploitation plus fine de l'équation de départ, que le mouvement possède une période qui dépend de l'angle maximal selon la relation :

$$T = T_0 (1 + \frac{\theta^2_{max}}{16})$$

(Cette formule est toujours approchée, et valable seulement pour de petits angles.)

Nous allons simuler le fonctionnement du pendule sur l'ordinateur, et chercher le domaine de validité de la formule précédente.

Nous prendrons comme conditions initiales :

$\theta(0) = \theta_{max}$ (variable)
$\theta'(\theta) = 0$ (le pendule est laché sans vitesse initiale)

Nous choisirons $K = \frac{\pi^2}{4}$, de sorte que T_0 soit égale à 4, et le premier passage à l'origine du pendule aura lieu pour des valeurs de t proches de l'unité, ce qui permettra une vérification directe de la formule.

L'équation à programmer est donc :

```
F = -PI*PI/4*SIN(Y(Ø))
```

La valeur finale est 1.5, le nombre de points à calculer 1 500 et la finesse 1, de manière à pouvoir déterminer une valeur suffisamment précise du quart de période.

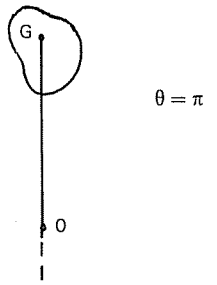
L'exécution doit être reprise plusieurs fois avec des amplitudes initiales différentes.

Voici quelques résultats que nous avons obtenus :

θ_{max}	T/4	$1 + \theta^2_{max}/16$
Ø.2	1.ØØ25	1.ØØ25
Ø.35	1.ØØ77	1.ØØ77
Ø.5	1.Ø158	1.Ø156
Ø.75	1.Ø363	1.Ø352
1	1.Ø663	1.Ø625
1.5	1.162Ø	1.14Ø6
2	1.329Ø	1.25ØØ

On constate que la formule est très bien vérifiée jusqu'à $\theta = 0{,}75$ rad soit 45° environ.

On peut également vérifier que si l'on introduit $\theta_{max} = \pi$, θ reste constant, ce qui correspond à la position d'équilibre instable du pendule :



De même, si l'on entre $\theta_{max} \geqslant \pi$, θ augmente au lieu de diminuer : le pendule retombe de l'autre côté.

Si au lieu de spécifier une dérivée initiale nulle, on entre $\theta'(0) = 10$ par exemple, ce qui revient à lancer le pendule, ce dernier n'a plus un mouvement périodique, mais tourne indéfiniment et θ tend vers l'infini.

Enfin, on peut tenir compte de la résistance de l'air, en modifiant l'équation différentielle :

$$\theta'' = -K \sin\theta - K' \theta'$$

Si on lance le pendule, il tournera d'abord autour du point 0 de moins en moins vite, puis entrera dans une deuxième phase de mouvement pseudo-périodique amorti qui tendra à immobiliser le pendule à sa position d'équilibre stable correspondant à $\theta = 0$.

7. Conclusion

Les méthodes que nous avons étudiées permettent la résolution numérique de la plupart des équations différentielles usuelles avec conditions initiales.

Leur utilisation est simple, mais nécessite souvent plusieurs minutes, voire plusieurs heures de calcul.

C'est pourquoi nous n'avons pas traité le cas des équations avec conditions aux limites, ou le cas des équations non résolues, qui nécessitent un temps de calcul nettement plus important : le BASIC s'avère alors insuffisant, et l'utilisation d'un compilateur s'impose.

Annexe 1
Équations de courbes classiques

1. Courbes d'équation $y = f(x)$

- $y = \dfrac{5x}{1 + x^2}$ (cubique serpentine)
- $y = \ln \dfrac{1}{\cos x}$ (chaînette)
- $y = e^{-x^2}$ (gaussienne)
- $y = e^{-\frac{x}{2}} \sin x$
- $y = \dfrac{\sin x}{x}$

2. Courbes paramétrées

- $\begin{cases} x = \cos t + \sin t \\ y = \sin t \end{cases}$ (ellipse)
- $\begin{cases} x = t - \sin t \\ y = 1 - \cos t \end{cases}$ (cycloïde)
- $\begin{cases} x = 2\cos^2 t \\ y = \cos^3 t \sin t \end{cases}$ (quartique piriforme)
- $\begin{cases} x = 2\cos t + \cos 2t \\ y = 2\sin t - \sin 2t \end{cases}$ (hypocycloïde à trois pointes)
- $\begin{cases} x = \cos^3 t \\ y = \sin^3 t \end{cases}$ (hypocycloïde à quatre pointes)
- $\begin{cases} x = e^{t-1} - t \\ y = t^3 - 3t \end{cases}$
- $\begin{cases} x = \operatorname{tg} t + \sin t \\ y = \dfrac{1}{\cos t} \end{cases}$
- $\begin{cases} x = 2t - 3\sin t \\ y = 2 - 3\cos t \end{cases}$ (trochoïde)
- $\begin{cases} x = \cos t \\ y = \sin t \end{cases}$ (cercle)
- $\begin{cases} x = \cos 3t \\ y = \sin 2t \end{cases}$
- $\begin{cases} x = \dfrac{3t^2}{1 + t^3} \\ y = \dfrac{3t^2}{1 + t^3} \end{cases}$ (Folium de Descartes)
- $\begin{cases} x = (1 + \cos^2 t)\sin t \\ y = \sin^2 t \,.\, \cos t \end{cases}$

3. Courbes polaires

- $r = \theta$ (spirale d'Archimède)
- $r = \dfrac{1}{\theta}$ (spirale hyperbolique)

- $r = 2\cos\theta + 1$ (limaçon de Pascal)
- $r = 1 + \cos\theta$ (cardioïde)
- $r = \cos^3\theta$ (Folium simple)
- $r = \dfrac{\sin^2\theta}{\cos\theta}$ (cissoïde de droite)
- $r = \dfrac{\sin\theta}{\theta}$ (cochléoïde)
- $r = \sin 2\theta$ (rosace à quatre branches)
- $r = 2\cos 2\theta - \cos\theta$ (scarabée)
- $r = \cos^2\theta\,(\cos\theta + \sin 4\theta)$ (Bifolium)
- $r = \sin\theta$
- $r = \cos 3\theta$
- $r = \sin 2\theta\,(\cos\theta + \sin\theta)$
- $r = \dfrac{\sin\theta}{1 + 2\cos\theta}$
- $r = \sin\dfrac{2}{3}\theta$
- $r = \sin^3\dfrac{\theta}{3}$
- $r = 1 + \operatorname{tg} 2\theta$

Annexe 2

Dérivées des fonctions usuelles

Fonction	*Dérivée*
x^n	$n\,x^{n-1}$
$\frac{1}{x}$	$-\frac{1}{x^2}$
$\sqrt{x}$	$\frac{1}{2\sqrt{x}}$
e^{mx}	$m\,e^{mx}$
$\ln\|x\|$	$\frac{1}{x}$
$\cos x$	$-\sin x$
$\sin x$	$\cos x$
$\operatorname{tg} x$	$\frac{1}{\cos^2 x} = 1 + \operatorname{tg}^2 x$
$\operatorname{cotg} x$	$-\frac{1}{\sin^2 x}$

Fonction	*Dérivée*
Arc cos x	$\frac{-1}{\sqrt{1-x^2}}$
Arc sin x	$\frac{1}{\sqrt{1-x^2}}$
Arc tg x	$\frac{1}{1+x^2}$
ch x	sh x
sh x	ch x
th x	$\frac{1}{\mathrm{ch}^2 x} = 1 - \mathrm{th}^2 x$
Arg ch x	$\frac{1}{\sqrt{x^2-1}}$
Arg sh x	$\frac{1}{\sqrt{x^2+1}}$
Arg th x	$\frac{1}{1-x^2}$

Annexe 3

Développements limités des fonctions usuelles

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^n \frac{x^{2n}}{(2n)!} + O(x^{2n+2})$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + O(x^{2n+3})$$

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + O(x^{n+1})$$

$$(1+x)^\alpha = 1 + \frac{\alpha}{1!}x + \frac{\alpha(\alpha-1)}{2!}x^2 + \ldots + \frac{\alpha(\alpha-1)\ldots(\alpha-n+1)}{n!}x^n + O(x^{n+1})$$

$$\frac{1}{1+x} = 1 - x + x^2 - x^3 + \ldots + (-1)^n x^n + O(x^{n+1})$$

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \ldots + x^n + O(x^{n+1})$$

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots + (-1)^{n-1}\frac{x^n}{n} + O(x^{n+1})$$

$$\operatorname{Arctg} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \ldots + (-1)^n \frac{x^{2n+1}}{2n+1} + O(x^{2n+3})$$

$$\operatorname{Arcsin} x = x + \frac{1}{2}\frac{x^3}{3} + \frac{1.3}{2.4}\frac{x^5}{5} + \ldots + \frac{1.3\ldots(2n-1)}{2.4\ldots 2n}\frac{x^{2n+1}}{2n+1} + O(x^{2n+3})$$

$$\operatorname{ch} x = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + \frac{x^{2n}}{(2n)!} + O(x^{2n+2})$$

$$\operatorname{sh} x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + \frac{x^{2n+1}}{(2n+1)!} + O(x^{2n+3})$$

$$\operatorname{Argth} x = x + \frac{x^3}{3} + \frac{x^5}{5} + \ldots + \frac{x^{2n+1}}{2n+1} + O(x^{2n+3})$$

$$\operatorname{Argsh} x = x - \frac{1}{2}\frac{x^3}{3} + \frac{1.3}{2.4}\frac{x^5}{5} - \ldots + (-1)^n \frac{1.3\ldots(2n-1)}{2.4\ldots 2n}\frac{x^{2n+1}}{2n+1} + O(x^{2n+3})$$

Annexe 4

Primitives des fonctions usuelles

Fonction	*Primitive*
$x^n \ (n \neq -1)$	$\frac{x^{n+1}}{n+1}$
$\frac{1}{x^2}$	$\frac{-1}{x}$
$\frac{1}{\sqrt{x}}$	$2\sqrt{x}$
$\frac{1}{x}$	$\ln \lvert x \rvert$
e^{mx}	$\frac{e^{mx}}{m}$
$\ln x$	$x \ln x - x$
$\cos x$	$\sin x$
$\cos (\omega x + \varphi)$	$\frac{1}{\omega} \sin (\omega x + \varphi)$
$\sin x$	$-\cos x$

Fonction	*Primitive*
$\sin(\omega x + \varphi)$	$-\frac{1}{\omega}\cos(\omega x + \varphi)$
$\operatorname{tg} x$	$-\ln\lvert\cos x\rvert$
$\operatorname{cotg} x$	$\ln\lvert\sin x\rvert$
$\frac{1}{\cos^2 x}$	$\operatorname{tg} x$
$\frac{1}{\sin^2 x}$	$-\operatorname{cotg} x$
$\frac{1}{\cos x}$	$\ln\lvert\operatorname{tg}(\frac{x}{2} + \frac{\pi}{4})\rvert$
$\frac{1}{\sin x}$	$\ln\lvert\operatorname{tg}(\frac{x}{2})\rvert$
$\frac{1}{1+\cos x}$	$\operatorname{tg}\frac{x}{2}$
$\frac{1}{1-\cos x}$	$\operatorname{cotg}\frac{x}{2}$
$\operatorname{ch} x$	$\operatorname{sh} x$
$\operatorname{sh} x$	$\operatorname{ch} x$
$\operatorname{th} x$	$\ln(\operatorname{ch} x)$
$\operatorname{coth} x$	$\ln\lvert\operatorname{sh} x\rvert$
$\frac{1}{\operatorname{ch}^2 x}$	$\operatorname{th} x$
$\frac{1}{\operatorname{sh}^2 x}$	$-\operatorname{coth} x$
$\frac{1}{\operatorname{sh} x}$	$\ln\lvert\operatorname{th}\frac{x}{2}\rvert$
$\frac{1}{\operatorname{ch} x}$	$\operatorname{Arctg}(\operatorname{sh} x)$
$\frac{1}{1+\operatorname{ch} x}$	$\operatorname{th}\frac{x}{2}$
$\frac{1}{1-\operatorname{ch} x}$	$\operatorname{coth}\frac{x}{2}$

Achevé d'imprimer sur les presses
de **CID** *éditions* — Nantes
Dépôt légal : octobre 1984
N° d'Éditeur : 4185

Bien qu'ils soient le plus souvent utilisés à des fins ludiques, les micro-ordinateurs n'en sont pas moins capables d'effectuer avec précision et rapidité toutes sortes de calculs numériques.

S'adressant à tous ceux qui désirent utiliser leur micro-ordinateur dans un but scientifique, cet ouvrage permet aussi bien l'initiation aux méthodes numériques que la réalisation de programmes spécialisés utilisant les programmes présentés comme des outils.

La théorie reste qualitative autant que possible, de manière à privilégier la compréhension intuitive, les exemples commentés jouant un rôle essentiel. Chaque chapitre traite d'un sujet particulier, et tous les programmes qu'il comporte peuvent être utilisés simultanément. De plus, tous les chapitres sont indépendants et peuvent être abordés dans un ordre quelconque. Deux programmes graphiques permettent en outre de bénéficier de l'affichage haute résolution de l'ordinateur pour représenter des courbes ou des surfaces.

Les programmes sont écrits en BASIC standard, ce qui en permet l'introduction sans modifications sur la majorité des ordinateurs.

EYROLLES

9605