OPTIMISATION NUMÉRIQUE

R. Coquidé 05/06/2021

Dans cet article, l'objectif est de calculer une optimisation sans utiliser de dérivation de fonction. En effet, s'il est souvent aisé de calculer une dérivée, il est le plus souvent difficile de calculer la valeur d'une variable ou d'un paramètre optimisant la grandeur souhaitée car la résolution des équations résultantes est souvent compliquée.

Il y a plusieurs possibilités de programmer, en « **J** », un pro-verbe représentant une fonction de plusieurs variables ou de plusieurs paramètres.

Dans cet article, une fonction de plusieurs variables (*qui seront nommées* **a,b,c,d**…) sera programmée ainsi (*exemple pour 4 variables*) sous forme **monadique** :

```
f =: {{instruction avec a,b,c,d  [ 'a b c d' =. y}}             ou
f =: {{)m
 'a b c d' =. y
 bloc d'instructions
 utilisant
 a,b,c,d
}}
```

et utilisée ainsi :

```
      R =.  f   (a,b,c,d)
```

Une fonction d'une variable et plusieurs paramètres notés **a,b,c,d**... sera programmée ainsi (*exemple avec 4 paramètres*) sous forme **dyadique** :

```
f =: {{instruction avec a,b,c,d,y  [ 'a b c d' =. x}}        ou
f =: {{)d
 'a b c d' =. x
 bloc d'instructions
 utilisant
 a,b,c,d,y
}}
```

et utilisée ainsi :

```
      R =. (a,b,c,d)  f  y
```

I) CALCUL DU MINI OU DU MAXI D'UNE FONCTION DE 1 à 4 VARIABLES

Les variables se nomment **a,b,c,d** et on les suppose situées respectivement dans les intervalles **(a0,a1) , (b0,b1) , (c0,c1) , (d0,d1)**.
Calcul du minimum (**fm**) ou du maximum (**fM**) d'une fonction **f** et des valeurs correspondantes **a,b,c,d.**

On utilise les pro-adverbes (programmés dans **III**)
FMIN1, FMIN2, FMIN3, FMIN4, FMAX1, FMAX2, FMAX3, FMAX4

Fonction **f** de 1 seule variable **a** :

```
'a fm' =. f  FMIN1 (a0,a1)
'a fM' =. f  FMAX1 (a0,a1)
```

Fonction **f** de 2 variables **a, b** :

```
'a b fm' =. f FMIN2 (a0,a1);(b0,b1)
'a b fM' =. f FMAX2 (a0,a1);(b0,b1)
```

Fonction **f** de 3 variables **a, b, c**

```
'a b c fm' =. f FMIN3 (a0,a1);(b0,b1);(c0,c1)
'a b c fM' =. f FMAX3 (a0,a1);(b0,b1);(c0,c1)
```

Fonction **f** de 4 variables **a, b, c, d**

```
'a b c d fm' =. f FMIN4 (a0,a1);(b0,b1);(c0,c1);(d0,d1)
'a b c d fM' =. f FMAX4 (a0,a1);(b0,b1);(c0,c1);(d0,d1)
```

Ex1 : Avec 3 variables **a,b,c**

FI =: {{(2*^*:a-1.2345)+(3.1*^*:b-2.134)+(1.6*^*:c-4.2587)['a b c'=.y}}

[`'a b c fm' =.` **FI FMIN3 (0.11 10.87);(0.11 10.87);(0.11 10.87)**

1.2345 2.134 4.2587 6.7 NB. Résultats vérifiables ici aisément de tête

Ex2 : Avec 3 variables **a,b,c**

```
MU =:{{(2.3*^*:a+b-3)+(4.2*^*:c+2*a-1)+(1.27*^*:a-4-3*c) [ 'a b c'=.y}}
['a b c fm' =. MU FMIN3 (_10.11 183.87);(_16.17 274.84);(_20.61 193.92)
```

0.4 2.6 1.2 7.77

On trouve donc a=0,4 b=2,6 c=1,2 MUm=7,77

Ex3 : Avec 4 variables **a,b,c,d**

```
   fonc=:{{)m
'a b c d'=.y
Z=.2.1*^-*:(2*a)+(3*b)+(2.5*c)+(_3.4*d)-6.023
Z=.Z+3.42*^-*:(3*a)+(1.65*b)+(2.13*c)+(1.12*d)-5.31
Z=.Z+5.034*^-*:(a)+(2.4*b)+(3.11*c)+(2.157*d)-6.215
Z+2.467*^-*:(2.6*a)+(1.68*b)+(2.35*c)+(1.468*d)-2.84
}}
   ['a b c d foncM' =. fonc FMAX4 (1.14 1.52);(23.85 24.21);(_20.51 _20.11);(4.9 5.3476)
1.2048 24.5812 _20.9702 5.20931 13.0127
```

Donc le Maxi de fonc est 13,0127 pour a=1,2048 b=24,5812, c=-20,9702   d=5,20931

II) CALCUL DES VALEURS OPTIMALES DES PARAMÈTRES D'UNE FONCTION PARAMÉTRIQUE PASSANT AU PLUS PRÈS D'UN NUAGE DE POINTS (X;Y) DU PLAN AU SENS DES MOINDRES CARRÉS.

(fonction de régression)

X = suite des valeurs de $\mathbf{x_i}$ mesurées

Y = suite des valeurs de $\mathbf{y_i}$ mesurées correspondantes

$$\mathbf{S^2 = \sum(y_i - f(x_i))^2} \quad \text{(somme des carrés des écarts)}$$

Si **f** est une fonction paramétrique d'au plus 4 paramètres **a,b,c,d,** $\mathbf{S^2}$ sera une fonction au plus des 4 variables **a,b,c,d.**

A) CALCUL DU S^2 (S² noté S2)

```
S2 =. f SDEUX0 (X;Y) ''          NB. 0 paramètre
S2 =. f SDEUX1 (X;Y) a           NB. 1 paramètre
S2 =. f SDEUX2 (X;Y) (a,b)       NB. 2 paramètres
S2 =. f SDEUX3 (X;Y) (a,b,c)     NB. 3 paramètres
S2 =. f SDEUX4 (X;Y) (a,b,c,d)   NB. 4 paramètres
```

(les pro-conjonctions **SDEUX0**, **SDEUX1**, **SDEUX2**, **SDEUX3** et **SDEUX4** sont programmées en **III**)

Ex1: 0 paramètre

```
   X
1 2 3 4 5 6
   Y
5.2  13.9  29.3  49.5  77  110.4
   fff =: {{2+3*y²}}                NB. 0 paramètre
[ S2 =. fff  SDEUX0  (X;Y)  ''
0.55
```

Ex2: 2 paramètres

```
    X
1 2 3 4 5 6
    Y
5.2 13.9 29.3 49.5 77 110.4
    g =: {{a+b*y^2 [ 'a b'=.x}}      NB. 2 paramètres
    [S2 =: g SDEUX2 (X;Y) (2 3)
0.55
```

Ex3: 4 paramètres

```
    X
0.002 0.003 0.004 0.006 0.007 0.009 0.01 0.02 0.057 0.12
    Y
8.33 8.334 8.34 8.357 8.36 8.404 8.379 8.434 8.71 9.222
    h =: {{)d
'a b c d'=.x
Z=.2.45+2*a^2
Z=.Z+3.1*^b*y
Z+(b+2.6*c)**:y
}}                                    NB. 4 paramètres
    [S2 =. h SDEUX4 (X;Y) (1.18 1.84 2.065 0.81)
0.0025612
```

Plus S2 est petit, plus la courbe de la fonction est « proche » du nuage de points au sens des moindres carrés. On dit alors que celle-ci « représente » bien le nuage de points de mesures. S2 est une mesure de cette représentation.
Si S2 = 0 la courbe passe exactement par tous les points du nuage. C'est l'idéal ! Mais il ne faut pas rèver car lorsque l'on fait des mesures, celles-ci sont généralement entachées d'erreurs. Heureusement on a souvent une idée a priori du type de fonction qui représentera le nuage de points de mesures. Cette fonction est souvent paramétrique et des études préalables permettent de savoir dans quel intervalle se trouve chaque paramètre.

Le problème à résoudre est alors de calculer les valeurs des paramètres (chacune étant située dans un intervalle plausible) minimisant le S^2 pour le nuage de points étudié (X;Y).

B) OPTIMISATION DES PARAMÈTRES D'UNE FONCTION CONNUE MINIMISANT LE S2 POUR UN NUAGE DE POINTS (X;Y) DONNÉ.

1) Moindres carrés **S2** pour des valeurs des paramètres **a,b,c,d** situées dans les intervalles **(a0 , a1), (b0 , b1), (c0 , c1), (d0 , d1).**

Si f a 1 paramètre **a** :
```
'a S2' =. (X;Y) f S2MIN1 (a0,a1)
```
Si f a 2 paramètres **a,b** :
```
'a b S2' =. (X;Y) f S2MIN2 (a0,a1);(b0,b1)
```
Si f a 3 paramètres **a,b,c** :
```
'a b c S2' =. (X;Y) f S2MIN3 (a0,a1);(b0,b1);(c0,c1)
```
Si f a 4 paramètres **a,b,c,d** :
```
'a b c d S2' =. (X;Y) f S2MIN4 (a0,a1);(b0,b1);(c0 c1);(d0 d1)
```

(Les pro-adverbes **S2MIN1, S2MIN2, S2MIN3, S2MIN4** sont programmés en III).

2) Représentations graphiques d’un nuage de points de mesures
Le monad **PlotNuage** et la pro-conjonction **PlotNuageFtReg** sont programmés en III

'' PlotNuage (X;Y)	(le monad PlotNuage est programmé en III)
'Titre' PlotNuage (X;Y)	Titre peut être vide

Avec une fonction de régression FtReg utilisons **PlotNuageFtReg**:

```
'Titre' FtReg PlotNuageFtReg 'SousTitre' (X;Y)
'Titre' FtReg PlotNuageFtReg '' (X;Y)
'' FtReg PlotNuageFtReg 'SousTitre' (X;Y)
'' FtReg PlotNuageFtReg '' (X;Y)
```

Titre et/ou SousTitre peuvent être vides

Exemple d’application avec 4 paramètres :

On se propose de chercher une fonction « **représentant bien** » le nuage de points de mesures suivant au sens des moindres carrés :

X

0 0.5 0.6 0.7 0.9 1 1.1 1.2 1.4 1.5 1.6 1.7 1.8 1.9 2.1 2.4 2.8 3 3.1 3.2 3.4 3.5 3.6 3.7 3.9

Y

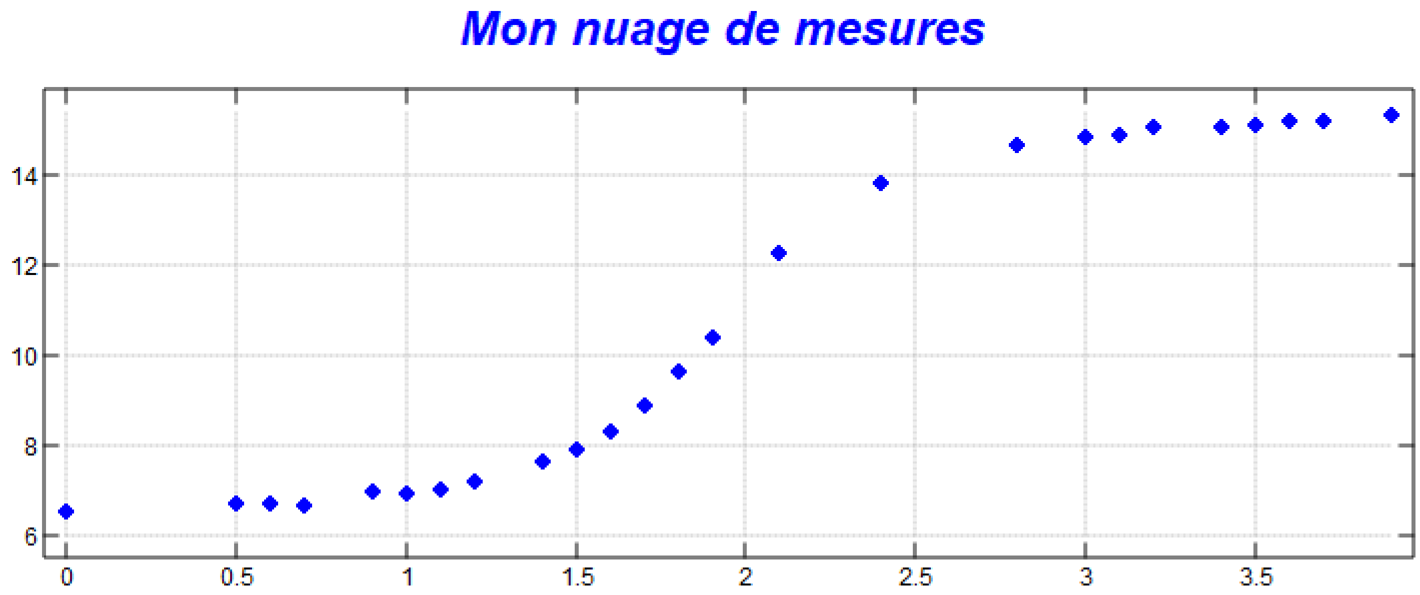
6.55 6.71 6.69987 6.65 6.96 6.94 7.003 7.20 7.62 7.91 8.31 8.90 9.62 10.40 12.26 13.82 14.63 14.83 14.88 15.06 15.04 15.09 15.20 15.18 15.32

Supposons qu'aucune étude préalable nous permette de savoir quel type de fonction on cherche.

Commençons par « **visualiser** » le nuage de points.

Utilisons **« PlotNuage »** programmé en III.

'Mon nuage de mesures' PlotNuage (X;Y)



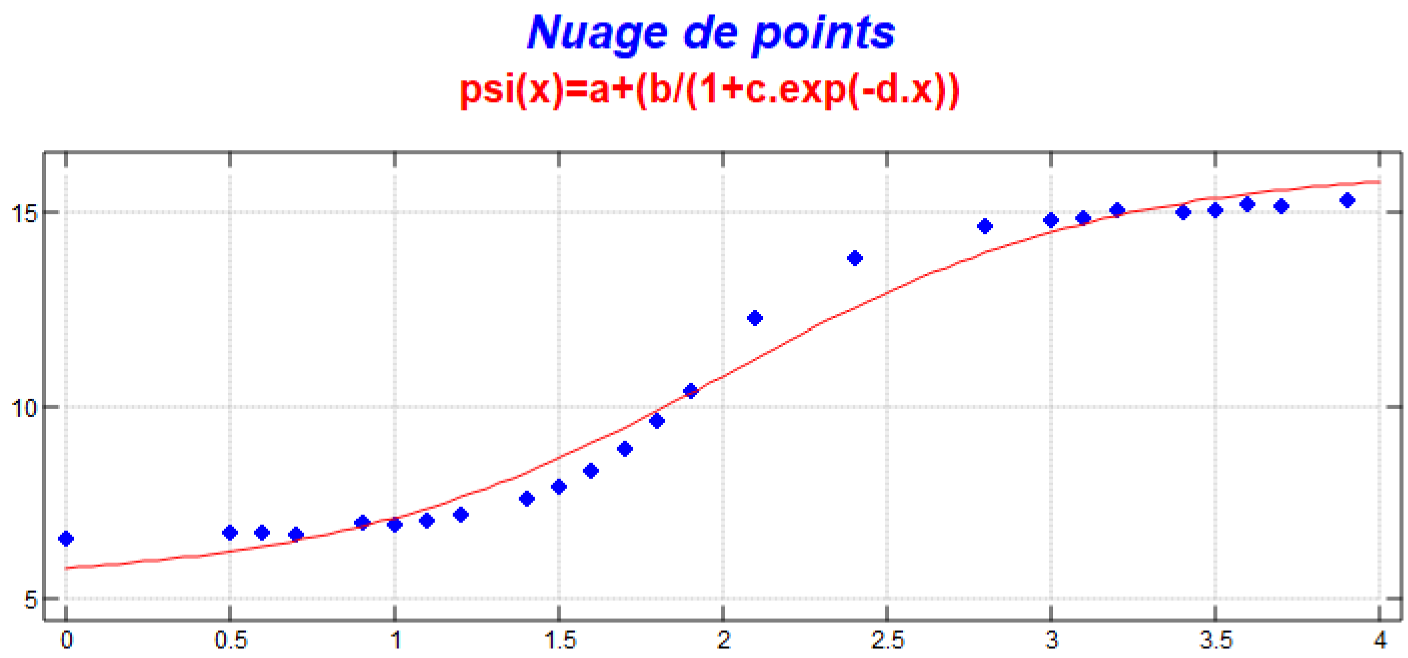
Ce graphique nous suggère les fonctions Sinus, Arctg, th, ou toute fonction de répartition d'une loi de probabilité (*à une translation et une anamorphose près*). Testons certaines de ces fonctions en faisant des essais comparatifs.Celle qui aura le plus petit S2 aura gagné...notre confiance.

1er essai

```
   psi=:{{a+b%1+c*^-d*y [ 'a b c d'=.x}}     NB. psi(x)=a+(b/(1+c.exp(-d.x))
   ['a b c d S2' =. (X;Y) psi S2MIN4 (4 8);(9 11);(30 35);(0.1 1.7)
5.45938 10.7 30.25 1.7 6.99411               NB. a b c d S2
```

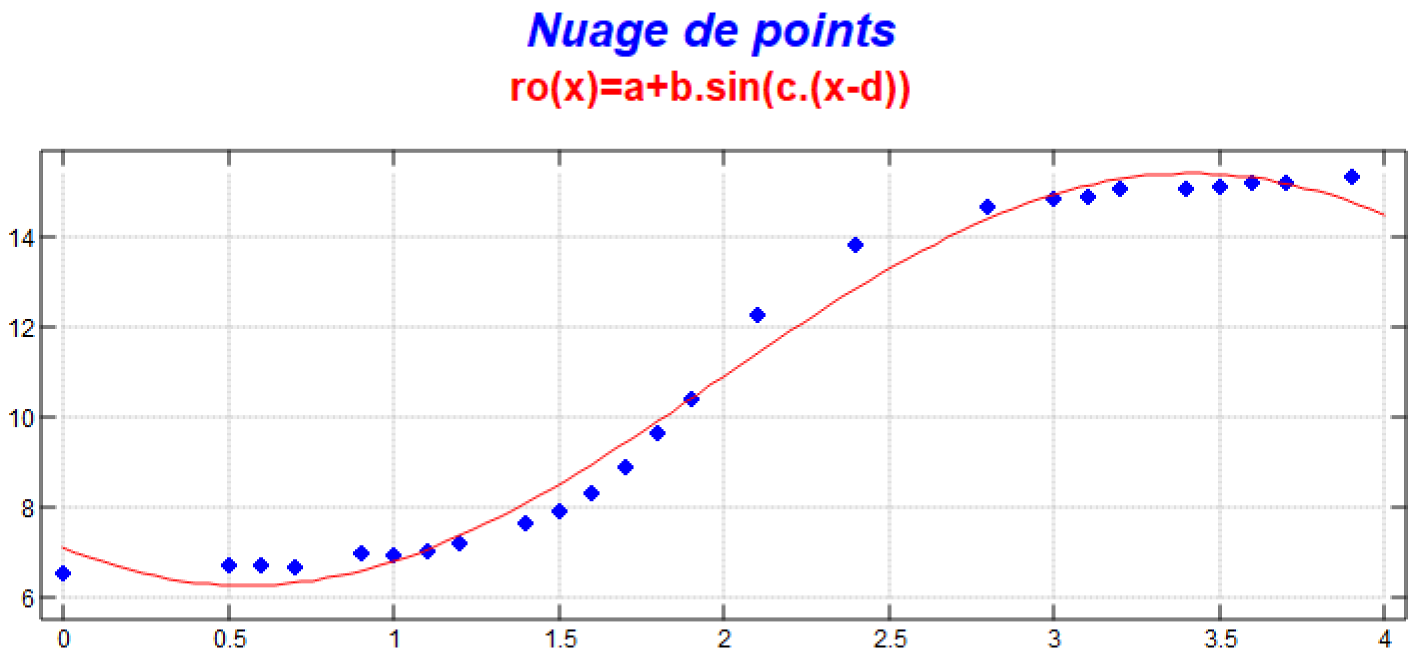
Ici le **s2 (6.99411)** est relativement important ce qui indique que la courbe s'éloigne des points de mesures pour certaines valeurs de **X.**
Utilisons **PlotNuageFtReg** pour les visualiser.

```
'Nuage de points'(a,b,c,d)&psi PlotNuageFtReg 'psi(x)=a+(b/(1+c.exp(-d.x))'(X;Y)
```



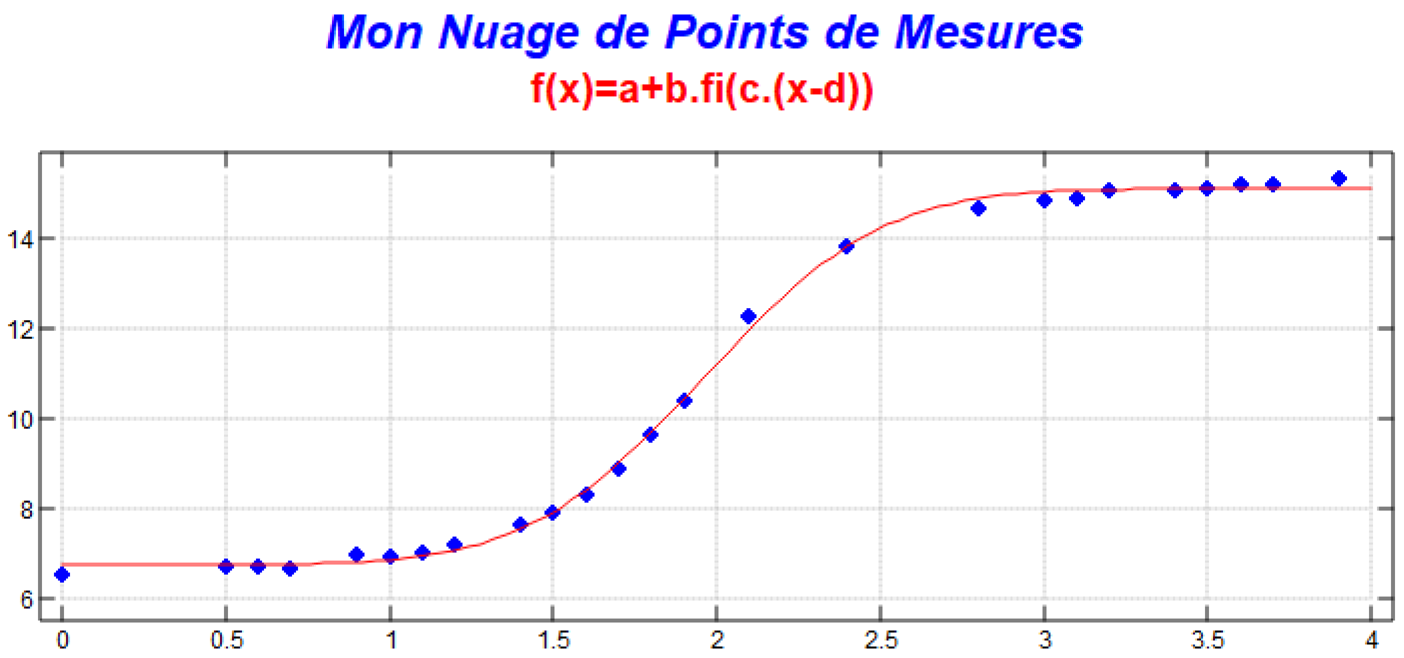
2e essai

```
   ro=:{{a+b*1 o. c*y-d [ 'a b c d'=.x}}    NB. ro(x)=a+b.sin(c.(x-d))
   ['a b c d S2' =. (X;Y) ro S2MIN4 (10 14);(4 10);(0.2 1.4);(1.9 2.5)
10.8241 4.56213 1.09887 1.987 4.70208       NB. a b c d S2
'Nuage de points' (a,b,c,d)&ro PlotNuageFtReg 'ro(x)=a+b.sin(c.(x-d))' (X;Y)
```



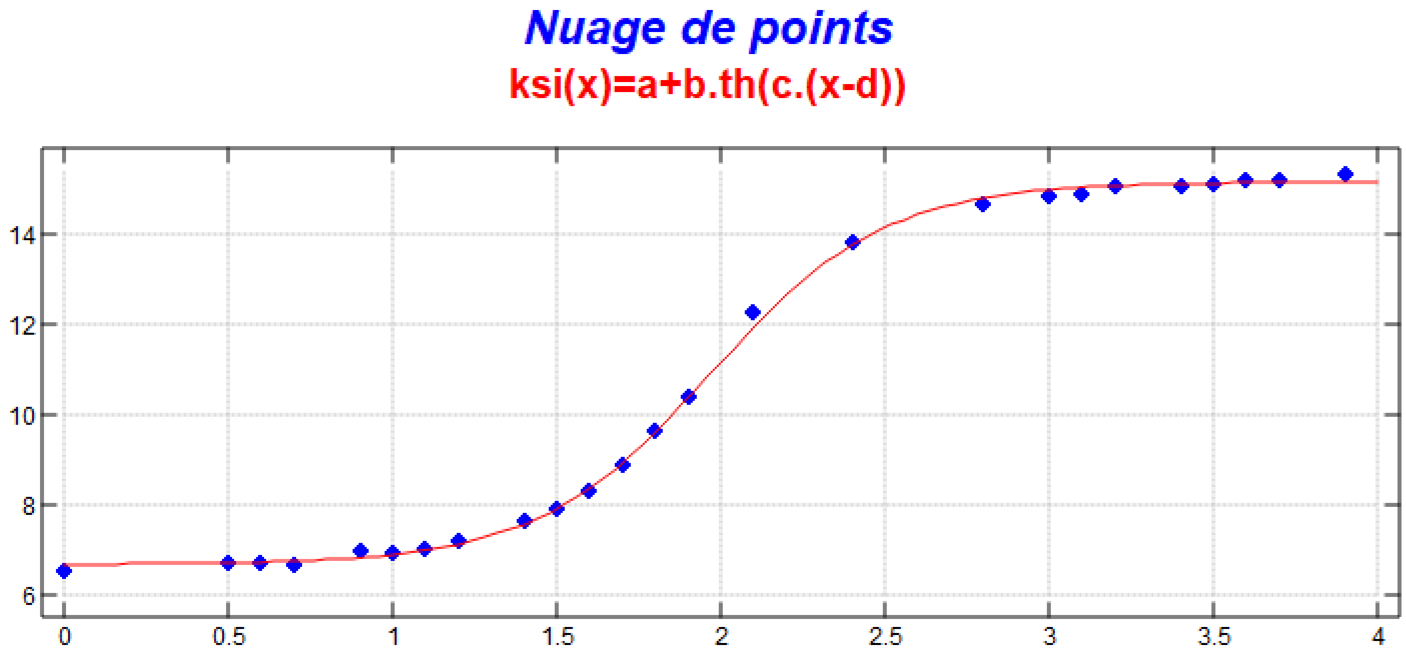
3e essai

```
   fi=:{{)m     NB. Intégrale de exp(-x²) entre 0 et y où y réel quelconque
if.y>:0 do.
1r2p1r2*-.%(1 0.0705230784 0.0422820123 0.0092705272 0.0001520143 0.000276572
0.0000430638 p.y)^16
else.-fi-y end.
}}
   eta=:{{a+b*fi c*(y-d) [ 'a b c d'=.x}}      NB. eta(x)=a+b.fi(c.(x-d))
   ['a b c d S2'=.(X;Y) eta S2MIN4 (7 12);(2 8.7);(1.1 3.21);(1 3.1)
10.9141 4.70035 1.64681 1.96285 0.436047       NB. a b c d S2
'Mon Nuage de Points de Mesures'(a,b,c,d)&eta PlotNuageFtRegr 'f(x)=a+b.fi(c.(x-
d))' (X;Y)
```



4^{e} essai

```
   ksi=:{{a+b*7 o. c*y-d [ 'a b c d'=.x}}   NB. a+b*th(c.(x-d))
   ['a b c d S2' =. (X;Y) ksi S2MIN4 (9.8 11.5);(4 5.63);(0.9 2.47);(0.8 2.7)
10.905 4.22851 1.88125 1.96898 0.289878          NB. a b c d S2
'Nuage de points' (a,b,c,d)&ksi PlotNuageFtReg 'ksi(x)=a+b.th(c.(x-d))' (X;Y)
```



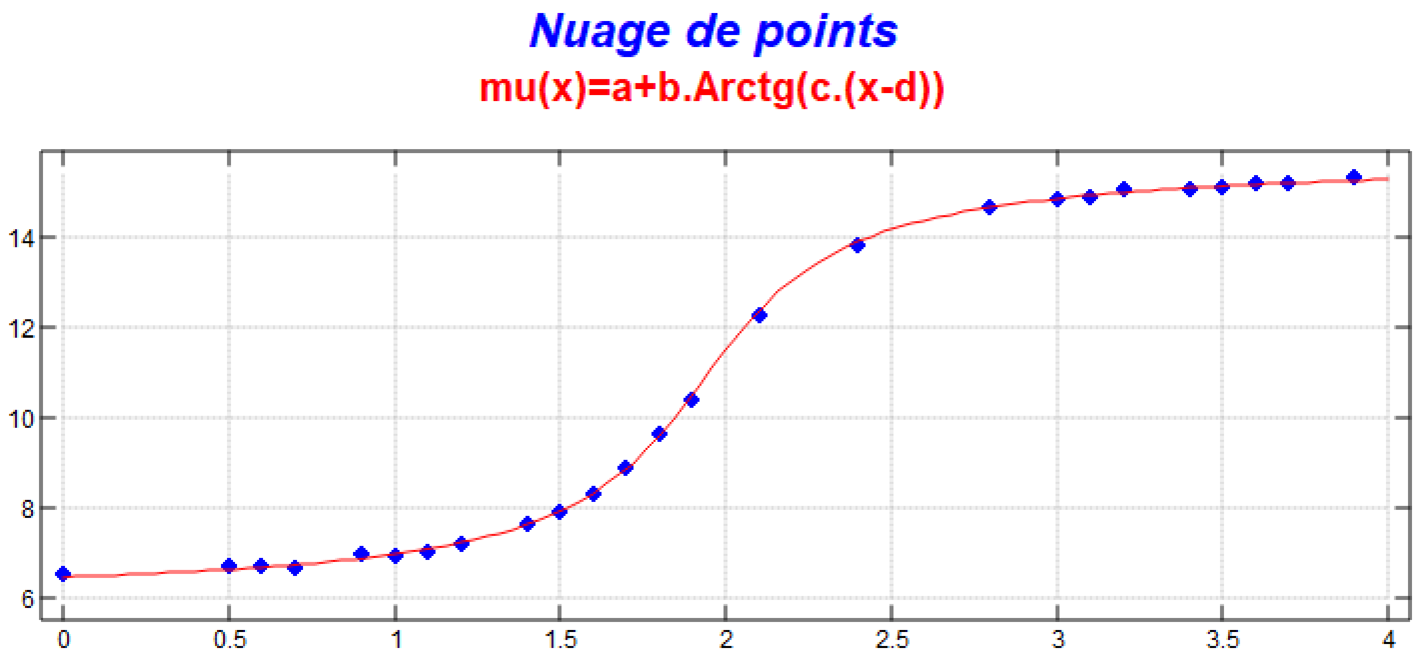
5e essai

mu=:{{a+b*_3 o.c*y-d ['a b c d'=.x}}   NB. a+b*Arctg(c.(x-d))

(X;Y) mu S2MIN4 (7 12);(1 4.2);(2.1 4.21);(1.07 3.65)

10.8431 3.11283 3.155 1.9343 0.0806053   NB. a b c d S2

'Nuage de points' (a,b,c,d)&mu PlotNuageFtReg 'mu(x)=a+b.Arctg(c.(x-d))' (X;Y)



Conclusion: ce dernier essai semble concluant car **S2=0,0806053** indique que la courbe passe presque parfaitement par chaque point du nuage de mesures. Cette fonction obtenue peut être utilisée avec une bonne fiabilité pour prévoir les valeurs obtenues entre les points des mesures effectuées. Mais en dehors de l'intervalle (0 4) il est risqué d'utiliser cette même fonction.

III) ESPACE DE TRAVAIL   NB. OPTIMISATION.ijs

```
    require 'plot'
E =: 1!:2&2
P =: 9!:11
sint =: {.@]+-/@|.@]*i.@>:@[%[
NB.   X =. N sint x1,x2    NB. Divise le segment (x1,x2) en N segments égaux

    NB. CALCUL DU MINI D'UNE FONCTION DE 1 à 4 VARIABLES
NB. (pro-adverbes)
    FMIN1 =: {{)a
I=.i.20 [ Fmin=._ [ 'A0 A1'=.y [ W=._1 1
for_i. I do. pA=.(A1-A0)%20 [ A=.20 sint A0,A1
for_a. A do. if. Fmin>z=.u a do. Fmin=.z [ am=.a end.end.
A1=.A1<.a1 [ A0=.A0>.a0 [ 'a0 a1'=.am+pA*W end.
am,Fmin
}}
    FMIN2 =: {{)a
I=.i.20 [ Fmin =. _ [ 'A0 A1'=.>0{y [ 'B0 B1'=.>1{y [ W=._1 1
for_i. I do. pA =. (A1-A0)%20 [ A =. 20 sint A0,A1
AB =.,{A;B [ pB =. (B1-B0)%20 [ B =. 20 sint B0,B1
for_ab. AB do. 'a b'=.>ab
if. Fmin>z=.u a,b do. am=.a [ bm=.b [ Fmin=.z end. end.
'a0 a1'=.am+pA*W [ 'b0 b1'=.bm+pB*W
A1=.A1<.a1 [ A0=.A0>.a0 [ B1=.B1<.b1 [ B0=.B0>.b0 end.
am,bm,Fmin
}}
    FMIN3 =: {{)a
I=.i.20 [ Fmin =. _ [ 'A0 A1'=.>0{y [ 'B0 B1'=.>1{y
'C0 C1'=.>2{y [ W=._1 1
for_i. I do. pA =. (A1-A0)%20 [ A =. 20 sint A0,A1
pB =. (B1-B0)%20 [ B =. 20 sint B0,B1
pC =. (C1-C0)%20 [ C =. 20 sint C0,C1
```

```
ABC =.,{A;B;C
for_abc. ABC do. 'a b c'=.>abc
if. Fmin>z=.u a,b,c do. am=.a [ bm=.b [ cm=.c [ Fmin=.z end. end.
'a0 a1'=.am+pA*W [ 'b0 b1'=.bm+pB*W [ 'c0 c1'=.cm+pC*W
A1=.A1<.a1 [ A0=.A0>.a0 [ B1=.B1<.b1 [ B0=.B0>.b0
C1=.C1<.c1 [ C0=.C0>.c0 end.
am,bm,cm,Fmin
}}
```

FMIN4 =: {{)a

```
I=.i.20 [ Fmin =. _ [ 'A0 A1'=.>0{y [ 'B0 B1'=.>1{y
'C0 C1'=.>2{y ['D0 D1'=.>3{y [ W=._1 1
for_i. I do. pA =. (A1-A0)%20 [ A =. 20 sint A0,A1
pB =. (B1-B0)%20 [ B =. 20 sint B0,B1
pC =. (C1-C0)%20 [ C =. 20 sint C0,C1
pD =. (D1-D0)%20 [ D =. 20 sint D0,D1
ABCD =.,{A;B;C;D
for_abcd. ABCD do. 'a b c d'=.>abcd
if. Fmin>z=.u a,b,c,d do. am=.a [ bm=.b [ cm=.c
dm=.d [ Fmin=.z end. end.
'a0 a1'=.am+pA*W [ 'b0 b1'=.bm+pB*W [ 'c0 c1'=.cm+pC*W
'd0 d1'=.dm+pD*W
A1=.A1<.a1 [ A0=.A0>.a0 [ B1=.B1<.b1 [ B0=.B0>.b0
C1=.C1<.c1 [ C0=.C0>.c0 [ D0=.D0>.d0 end.
am,bm,cm,dm,Fmin
}}
```

NB. CALCUL DU MAXI D’UNE FONCTION DE 1 à 4 VARIABLES

NB. (pro-adverbes)

FMAX1 =: {{1 _1*-@u FMIN1 y}}
FMAX2 =: {{1 1 _1*-@u FMIN2 y}}
FMAX3 =: {{1 1 1 _1*-@u FMIN3 y}}
FMAX4 =: {{1 1 1 1 _1*-@u FMIN4 y}}

NB. AVEC UN NUAGE DE POINTS DE MESURES (X;Y)
NB. CALCUL DU S2 = Σ(Y- f(X))²
NB. la fonction f ayant 0 à 4 paramètres
NB. (5 pro-conjonctions utilisant **n**=.**(X;Y) et u=.f**)

```
    NB. 0 paramètre :
SDEUX0 =: {{+/"1*:Y-u"0 X['X Y'=.n [z=.y}}
    NB. 1 paramètre :
SDEUX1 =: {{+/"1 *:Y-a&u"0 X['X Y'=.n [ a=.y}}
    NB. 2 paramètres :
SDEUX2 =: {{+/"1 *:Y-(a,b)&u"0 X['X Y'=.n [ 'a b'=.y}}
    NB. 3 paramètres :
SDEUX3 =: {{+/"1 *:Y-(a,b,c)&u"0 X['X Y'=.n [ 'a b c'=.y}}
    NB. 4 paramètres :
SDEUX4 =: {{+/"1 *:Y-(a,b,c,d)&u"0 X['X Y'=.n [ 'a b c d'=.y}}
```

NB. POUR DES FTS DE 1 À 4 PARAMÈTRES CALCUL DU MINI S2m PRÉCÉDÉ NB. PAR LES VALEURS OPTIMALES am,bm,cm,dm DES PARAMÈTRES
NB. (4 pro-adverbes)

```
    NB. 1 paramètre :
S2MIN1 =: {{u SDEUX1 x FMIN1 y}}
    NB. 2 paramètres :
S2MIN2 =: {{u SDEUX2 x FMIN2 y}}
    NB. 3 paramètres :
S2MIN3 =: {{u SDEUX3 x FMIN3 y}}
    NB. 4 paramètres :
S2MIN4 =: {{u SDEUX4 x FMIN4 y}}
```

PlotNuage =:{{)d
```
'X Y'=.y [ Titre=.x          NB. Titre éventuellement vide
pd 'new'
pd 'textfont arial 18 bold italic'
pd 'textcolor blue'
pd 'textc 500 _40x ',Titre
pd 'new 40x 20x _40x _70x'
pd 'color blue'
pd 'type marker'
pd X;Y
pd 'show'
}}
NB. Utilisation :
NB. '' PlotNuage (X;Y) ou
NB. 'Titre' PlotNuage (X;Y)
```

PlotNuageFtReg =:{{)c
```
Titre=.x [ SousTitre=.n
Y1=.u"0 X1=.100 sint (<.<./),(>.>./X) [ 'X Y'=.y
pd 'new'
pd 'textfont arial 18 bold italic'
pd 'textcolor blue'
pd 'textc 500 _15x ',Titre
pd 'textfont arial 15 bold'
pd 'textcolor red'
pd 'textc 500 _40x ',SousTitre
pd 'new 40x 20x _40x _70x'
pd 'color blue'
pd 'type marker'
pd X;Y
pd 'color red'
pd 'type line'
```

```
pd X1;Y1
pd 'show'
}}
NB. Utilisation :
NB. 'Titre' FtReg PlotNuageFtReg 'SousTitre' (X;Y)
NB. Titre et/ou SousTitre peuvent être vides
```

IV) REMARQUE

Les méthodes classiques d'optimisation utilisant des dérivées partielles et la résolution des équations résultantes, très rarement linéaires, pour obtenir les valeurs optimales des variables ou paramètres demandent généralement peu de « **temps machine** » mais beaucoup de « **temps bonhomme** ».

La méthode purement numérique utilisée ici demande peu de « **temps bonhomme** » mais… un important « **temps machine** ». Chaque optimisation demande le calcul d'une fonction en un nombre de points égal à:

20* 21^1 2 3 4

420 8820 185220 3889620

soit **420** pour **1** variable ou paramètre
8820 pour **2** variables ou paramètres
185220 pour **3** variables ou paramètres
3889620 pour **4** variables ou paramètres

C'est la raison pour laquelle cet article se limite à **4** variables ou paramètres.

Pour **5** variables ou paramètres il faudrait :

20*21^5

81682020 calculs de fonction.

C'est très faisable ! La programmation est aisée et peut se généraliser. Mais le « **temps machine** » (**20*21^n** calculs de fonction pour **n** variables ou paramètres) est dissuasif.

Rappelons toutefois que, en 1965 (*quand j'ai commencé à programmer !*), le **« temps machine »** coûtait très cher et le **« temps bonhomme »** beaucoup moins ; aujourd'hui, c'est l'inverse.