ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ
Mühendislik Mimarlık Fakültesi
İnşaat Mühendisliği Bölümü
E-Posta: ogu.ahmet.topcu@gmail.com
Web: http://mmf2.ogu.edu.tr/atopcu

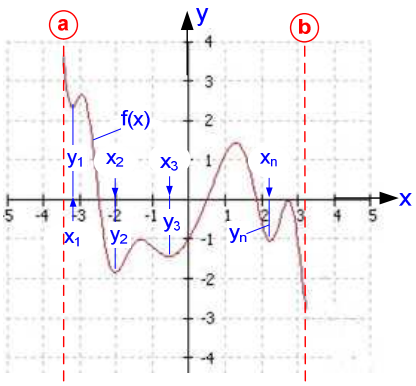
Bilgisayar Destekli

Nümerik Analiz

Ders notları 2014

Ahmet TOPÇU

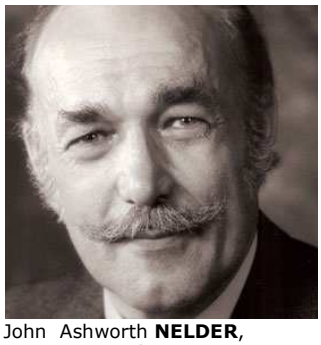
Min f(x)=?



37

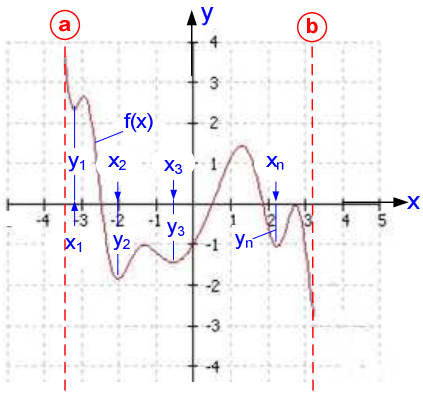
Min-Max

Bir fonksiyonun minimum-maksimum değerlerinin belirlenmesi



John Ashworth **NELDER**, (1924-2010), İngiliz

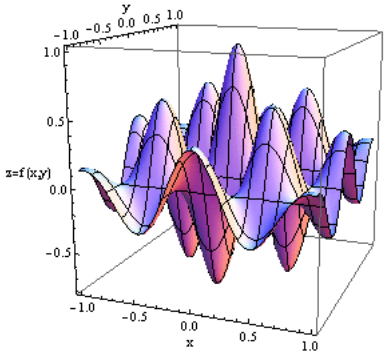
37. BİR FONKSİYONUN MİNUMUM-MAKSİMUM DEĞERLERİNİN BELİRLENMESİ



Tek veya çok değişkenli bir fonksiyonun, çoğu kez bazı ek koşulları da sağlayan, minimum veya maksimum değerinin aranması problemine optimizasyon(en iyileme) adı verilmektedir. Örneklersek: Min $z=f(x,y)=x^2+y$ problemi ek koşulsuz; $x>0$ olmak kaydıyla Min $z=f(x,y)=x^2+y$ problemi ise ek koşullu minimazasyon problemidir.

Bu bölümde minimizasyon teknikleri hakkında özet bilgiler verilecektir.

Sağda tek değişkenli bir $y=f(x)$ fonksiyonun grafiği [a,b] aralığında verilmiştir. Görüldüğü gibi, fonksiyon [a,b] aralığının bazı noktalarında büyük, bazılarında küçük değerler almaktadır; tepe(maksimum) ve vadi(minimum) noktaları vardır. x_1, x_2, x_n noktalarında ordinatlar y_1, y_2, …, y_n değerlerini almıştır, yani $y_1=f(x_1)$, $y_2=f(x_2)$, … , $y_n=f(x_n)$ dir. y_1, y_2, …, y_n değerleri hemen solundaki ve hemen sağındaki ordinatlardan daha küçüktür. Bu nedenle y_1, y_2, …, y_n ordinatları fonksiyonun [a,b] aralığındaki minimumlarıdır. Minimumlardan en küçüğüne global minimum, diğerlerine lokal minimum denir. Şekildeki fonksiyonun x_2 noktasındaki y_2 değeri global minimum, diğerleri lokal minimumdur.



Sağdaki iki değişkenli $z=f(x,y)$ yüzeyinin grafiğine bakıldığında $f(x,y)$ yüzeyinin çok sayıda minimum ve maksimum noktası olduğu görülür.

Bir değişkenli f(x), iki değişkenli f(x,y), ya da daha genel olarak, çok değişkenli f(x, y, z, v, …) fonksiyonunun belli bir bölgesinde minimum değerinin hesaplanması teknik ve endüstriyel problemlerde gerekli olur. Klasik matematikte verilen yöntemler çok değişkenli genel fonksiyonların minimum değerinin belirlenmesinde çoğu kez yetersiz kalır, nümerik metotlar çözüm için kullanılır.

Min-Max ve optimizasyon konusundaki yöntem ve programlar hakkında geniş bilgi için http://www.fing.edu.uy/if/cursos/fiscomp/extras/numrec/book/f10.pdf adresine bakılabilir. İzleyen sayfalarda minimum-maksimum bulma problemlerine yönelik bazı programlara ve örnek çözümlere yer verilmiştir. Bu programlar fonksiyonun analitik türevine gerek duymazlar.

1. GoldenRatioSearch[1]: tek değişkenli f(x) fonksiyonunun [a,b] aralığındaki minimum noktalarını ve minimum değerlerini Fibonacci golden ratio search metodu ile hesaplar.

2. Brent[2]: tek değişkenli f(x) fonksiyonunun [a,b] aralığındaki minimum noktalarını ve minimum değerlerini Brent metodu ile hesaplar.

3.NelderMead[3]: n değişkenli f(x,y,z,v, …) fonksiyonunun minimum olduğu noktanın koordinatlarını ve minimum değerini Nelder-Mead simplex(doğrusal programlama) adı verilen metod ile hesaplar. Minimum değerin arandığı bölgeyi tanımlayan n+1 noktanın başlangıç koordinatlarının verilmesi gerekir. Başlangıç koordinatlarına bağlı olarak hesaplanan değer lokal veya global minimum olabilir, global minimum olacağı garantisi yoktur. NelderMead, doğal olarak, tek değişkenli f(x) fonksiyonu için de kullanılabilir.

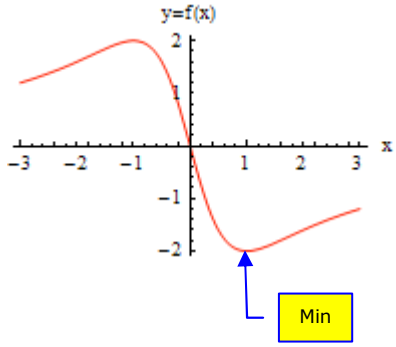
4.SteepestDescent: f(x,y,z) fonksiyonunun maksimum olduğu noktanın koordinatlarını ve maksimum değerini Steepest Descent metodu ile hesaplar. Maksimum değerin arandığı civarda x, y, z değişkenleri için başlangıç değerlerinin verilmesi gerekir. Başlangıç değerlerine bağlı olarak hesaplanan değer lokal veya global maksimum olabilir, global

[1] Teorik bilgi için bak: http://math.fullerton.edu/mathews/n2003/GoldenRatioSearchMod.html
[2] Richard Brent, Avusturalyalı, (1946-), tarafından 1973 yılında yayınlandı.
[3] Teorik bilgi için bak: http://math.fullerton.edu/mathews/n2003/NelderMeadMod.html. Nelder-Mead optimizasyon metodu John Ashworth **NELDER**, İngiliz(1924-2010) ve Roger **MEAD** tarafından 1965 yılında geliştirildi.

maksimum olacağı garantisi yoktur. SteepestDescent tek değişkenli f(x) veya iki değişkenli f(x,y) fonksiyonu için de kullanılabilir.

GoldenRatioSearch programı ve örnekleri:

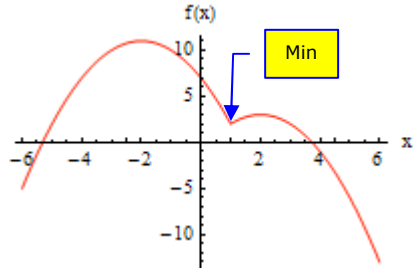
1. Sağda grafiği verilen $y = f(x) = \dfrac{-4x(x-2)}{(x^2+1)(x-2)}$ tek değişkenli fonksiyonunun [-3,3] aralığında bir minimum değeri olduğu görülmektedir. Bu aralıkta başka minimum olmadığından global minimumdur. Program x=1 noktasında f(1)=-2 global minimum değer olarak hesaplamıştır.



GoldenRatioSearch sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[-3 , 3 ] aralığında minimum değerler(GoldenRatioSearch):
x= 1.00000000410607          f(x)=-2          Global Minimum
```

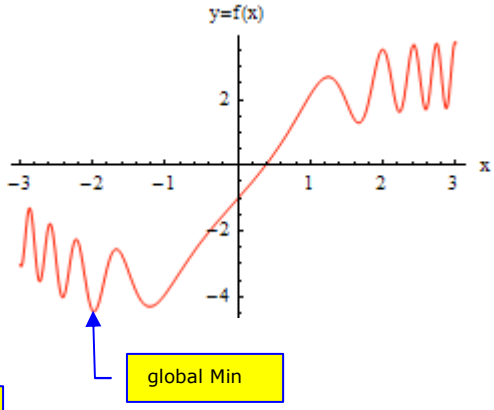
2. Sağda grafiği verilen $y = f(x) = -x^2 + 4|x-1| + 3$ tek değişkenli fonksiyonunun [-6,6] aralığında bir minimum değeri olduğu görülmektedir. Bu aralıkta başka minimum olmadığından global minimumdur. Program x=1 noktasında f(1)=2 global minimum değer olarak hesaplamıştır.



GoldenRatioSearch sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[-6 , 6 ] aralığında minimum değerler(GoldenRatioSearch):
x= 1.000000007014            f(x)= 2.00000001402799          Global Minimum
```

3. $y = f(x) = 2Sin(x) + Sin(x^3) + \sqrt{e^x} - 2$ tek değişkenli fonksiyonunun [-3,3] aralığında çok sayıda minimum değeri olduğu sağdaki grafikten görülmektedir. Bunlardan biri global, diğerleri lokal minimumdur. Program x=-1.98 noktasında f(-1.98)=-4.46 global minimum değer olarak hesaplamıştır.

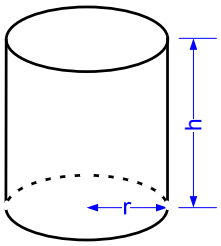


GoldenRatioSearch sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[-3 , 3 ] aralığında minimum değerler(GoldenRatioSearch):
x=-2.98637625063636          f(x)=-3.08210370712568          Lokal Minimum
x=-2.72989221515706          f(x)=-3.54202920688865          Lokal Minimum
x=-2.41359295145847          f(x)=-4.02864233984191          Lokal Minimum
x=-1.98333862609615          f(x)=-4.45988692760918          Global Minimum
x=-1.2145772814609           f(x)=-4.30530869162821          Lokal Minimum
x= 1.66259477179895          f(x)= 1.2946660780708           Lokal Minimum
x= 2.22228773883848          f(x)= 1.62840550206609          Lokal Minimum
x= 2.58495414302175          f(x)= 1.6984854798296           Lokal Minimum
x= 2.86655969907969          f(x)= 1.73560867026493          Lokal Minimum
```

Global Min

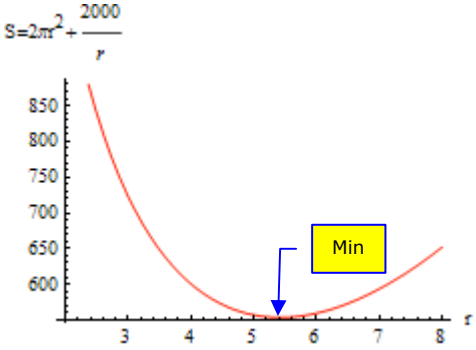
4. En az malzeme ile altı ve üstü kapalı silindir bir kutu imal edilecektir. Hacminin 1000 cm^3, toplam yüzeyinin en küçük (minimum) olması ve yarıçapın 4 cm den az 8 cm den büyük olmaması istenmektedir. Bu koşulları sağlayan r ve h var mıdır?



Toplam yüzey: $S = 2\pi r^2 + 2\pi rh$

Hacim: $V = \pi r^2 h$

$$S = 2\pi r^2 + 2\pi rh = 2\pi r^2 + 2\pi r\frac{1000}{\pi r^2} = 2\pi r^2 + \frac{2000}{r}$$



$S = 2\pi r^2 + \frac{2000}{r}$ sadece r nin fonksiyonudur. Bu durumda soru şuna dönüşmüştür: Bu fonksiyonun 4≤r≤8 aralığında global minimumu var mıdır? Fonksiyonun sağdaki grafiği incelendiğinde [5,6] aralığında minimum olduğu görülür. Aralık için a=5, b=6 ve r için x alınarak $S = f(x) = 2\pi x^2 + \frac{2000}{x}$ fonksiyonunun minimum olduğu yer ve minimum değer program ile hesaplanabilir:

GoldenRatioSearch sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[ 5 , 6 ] aralığında minimum değerler(GoldenRatioSearch):
x= 5.41926074280856          f(x)= 553.581044593209        Global Minimum
```

x=r=5.42 cm de S=f(x)=553.58 cm^2 global minimum değerini almıştır. h yüksekliği:

$V = \pi r^2 h = 1000 \rightarrow h = \frac{1000}{\pi \cdot 5.42^2} = 10.84$ cm bulunur.

GoldenRatioSearch ana programı

```
'--------------------Ana program GoldenRatioSearch -----------------
' f(x) fonksiyonunun [a,b] aralığındaki minimum değerlerini bulur
' Ahmet TOPÇU, Eskişehir, 2010
' Çağrılan alt program: GoldenRatioSearch
'-------------------------------------------------------------------------
DEFINT I-N
DEFDBL A-H, O-Z
DECLARE SUB GoldenRatioSearch (ai, bi, xMin, yMin, iHata)
   Pi = 4 * ATN(1)
'  a = -3: b = 3: DEF fnf (x) = -4 * x * (x - 2) / (x ^ 2 + 1) / (x - 2)
'  a = -6: b = 6: DEF fnf (x) = -x ^ 2 + 4 * ABS(x - 1) + 3
'   a = -3: b = 3: DEF fnf (x) = 2 * SIN(x) + SIN(x ^ 3) + SQR(EXP(x)) - 2
   a = 5: b = 6: DEF fnf (x) = 2 * Pi * x * x + 2000 / x
   CLS
   MaxAltAralik = 100
   DIM xi(MaxAltAralik, 2)
   iBulundu = 0
   Delta = (b - a) / MaxAltAralik
   bi = a
   FOR iAralik = 1 TO MaxAltAralik
    ai = bi
    bi = ai + Delta
    IF iAralik = MaxAltAralik THEN bi = b
    CALL GoldenRatioSearch(ai, bi, xMin, yMin, iHata)
    IF iHata = 0 THEN
     iBulundu = iBulundu + 1
     xi(iBulundu, 1) = xMin
     xi(iBulundu, 2) = yMin
    END IF
   NEXT iAralik
```

Devamı var

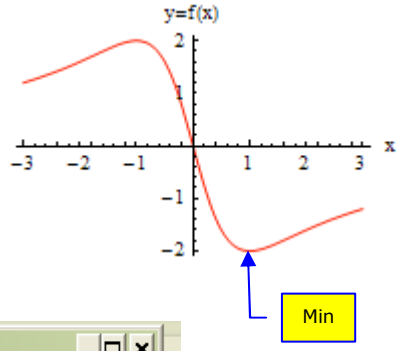
```
   IF iBulundu = 0 THEN
    PRINT "["; a; ","; b; "] aralığında minimum bulunamadı(GoldenRatioSearch)!":
    END
   END IF
' Global minimumu belirle
   j = 1: GlobalMin = xi(1, 2)
   FOR i = 1 TO iBulundu
    IF xi(i, 2) < GlobalMin THEN
     j = i
     GlobalMin = xi(i, 2)
    END IF
   NEXT i
   PRINT "["; a; ","; b; "] aralığında minimum değerler(GoldenRatioSearch):"
   FOR i = 1 TO iBulundu
    PRINT "x="; xi(i, 1), "f(x)="; xi(i, 2),
    IF i = j THEN PRINT " Global Minimum" ELSE PRINT " Lokal Minimum"
   NEXT i
END ' Ana program GoldenRatioSearch sonu
```

GoldenRatioSearch alt programı

```
SUB GoldenRatioSearch (ai, bi, xMin, yMin, iHata)
'-------------------------------------------------------------------------------
' f(x) fonksiyonunun minimum değerlerini bulur
' Veri:
'   f(x)    : çağıran programda DEF FNf(x)=... ile tanımlanmış olmalıdır
'   ai ve bi: minimum değer aranacak aralığın alt ve üst sınırı
' Çıktı:
'   xMin    : Minimum değerin bulunduğu absis
'   yMin    : xMin noktasında fonksiyonun değeri
'   iHata  =0 ise Minimum değer bulundu
'   iHata<>0  ise Minimum bulunamadı
' Metot: Fibonacci altın oran arama(Fibonacci golden ratio search)
' Fortran kodu: http://netlib.sandia.gov/textbook/mathews/index.html
' Kaynak:
'   NUMERICAL METHODS for Mathematics, Science and Engineering, 2nd Ed,
'   Prentice Hall, 1992. Section 8.1, Minimization of a Function, Page 413
'-------------------------------------------------------------------------------
   Epsilon = 1E-08
   Maxit = 100
   iHata = 0
   Rone = (SQR(5) - 1) / 2
   Rtwo = Rone * Rone

   a = ai
   b = bi
   h = b - a
   Ya = fnf(a)
   Yb = fnf(b)
   c = a + Rtwo * h
   d = a + Rone * h
   Yc = fnf(c)
   Yd = fnf(d)
   k = 1
   WHILE (ABS(Yb - Ya) > Epsilon OR h > Epsilon) AND k <= Maxit
    IF (Yc < Yd) THEN
     b = d
     Yb = Yd
     d = c
     Yd = Yc
     h = b - a
     c = a + Rtwo * h
     Yc = fnf(c)
    ELSE
     a = c
     Ya = Yc
     c = d
     Yc = Yd
     h = b - a
     d = a + Rone * h
     Yd = fnf(d)
    END IF
    k = k + 1
   WEND
   xMin = a
   yMin = Ya
   IF Yb < Ya THEN
    xMin = b
    yMin = Yb
   END IF
   ySol = fnf(xMin - Epsilon)
   ySag = fnf(xMin + Epsilon)
   IF k > Maxit OR ySol < yMin OR ySag < yMin THEN iHata = k
END SUB ' GoldenRatioSearch
```

Brent programı ve örnekleri:

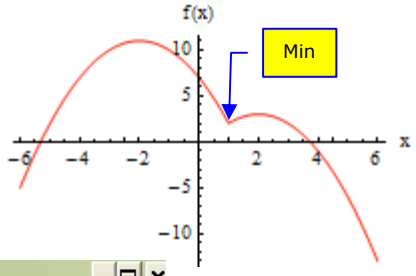
1. Sağda grafiği verilen $y = f(x) = \dfrac{-4x(x-2)}{(x^2+1)(x-2)}$ tek değişkenli fonksiyonunun [-2,2] aralığında minimumu Brent programı ile hesaplanmıştır.



Brent sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[ 0 , 1.5 ] aralığında minimum değerler(Brent):
x= .999999991058041          f(x)=-2          GLOBAL MİNİMUM
```

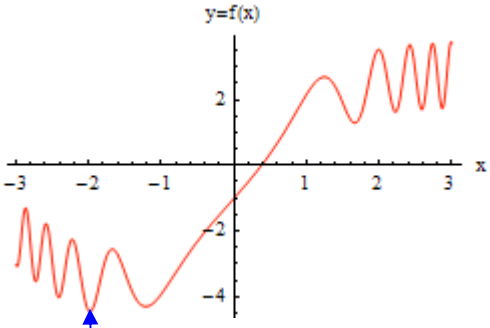
2. Sağda grafiği verilen $y = f(x) = -x^2 + 4|x-1| + 3$ tek değişkenli fonksiyonunun [-6,6] aralığında minimumu Brent programı ile hesaplanmıştır.



Brent sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[-6 , 6 ] aralığında minimum değerler(Brent):
x= 1.00000001060715          f(x)= 2.0000000212143          GLOBAL MİNİMUM
```

3. $y = f(x) = 2Sin(x) + Sin(x^3) + \sqrt{e^x} - 2$ tek değişkenli fonksiyonunun [-3,3] aralığındaki minimumları Brent programı ile hesaplanmıştır.



Brent sonucu

```
C:\NANALIZ\Basic\QBasic.EXE
[-3 , 3 ] aralığında minimum değerler(Brent):
x=-2.98637626149478          f(x)=-3.08210370712562          Lokal Minimum
x=-2.72989221265248          f(x)=-3.54202920688865          Lokal Minimum
x=-2.41359294800514          f(x)=-4.0286423398419           Lokal Minimum
x=-1.98333863616678          f(x)=-4.45988692760917          GLOBAL MİNİMUM
x=-1.21457726526441          f(x)=-4.30530869162821          Lokal Minimum
x= 1.66259479259537          f(x)= 1.29466607807081          Lokal Minimum
x= 2.22228774218603          f(x)= 1.62840550206609          Lokal Minimum
x= 2.58495413083523          f(x)= 1.69848547982962          Lokal Minimum
x= 2.86655966881121          f(x)= 1.73560867026526          Lokal Minimum
```

Global Min

Brent ana programı

```
'--------------------Ana program Brent-------------------------------
' f(x) fonksiyonunun [a,b] aralığındaki minimum değerlerini bulur
' Ahmet TOPÇU, Eskişehir, 2010
' Çağrılan alt program: Brent
'------------------------------------------------------------------

DEFINT I-N
DEFDBL A-H, O-Z

DECLARE FUNCTION Brent (Ax, Bx, Cx, xMin, iErr)
DECLARE FUNCTION Sign (a, b)

' [a,b] aralığı ve f(x) fonksiyonunun tanımlanması
'   a = 0: b = 1.5: DEF fnF (x) = -4 * x * (x - 2) / (x * x + 1) / (x - 2)
'   a = -6: b = 6: DEF fnF (x) = -x * x + 4 * ABS(x - 1) + 3
   a = -3: b = 3: DEF fnF (x) = 2 * SIN(x) + SIN(x ^ 3) + SQR(EXP(x)) - 2

   CLS
   MaxAltAralik = 100
   DIM xi(MaxAltAralik, 2)
   iBulundu = 0
   Delta = (b - a) / MaxAltAralik
   Cx = a
   FOR iAralik = 1 TO MaxAltAralik
    Ax = Cx
    Cx = Ax + Delta
    IF iAralik = MaxAltAralik THEN Cx = b
    Bx = Ax + Delta / 10
    yMin = Brent(Ax, Bx, Cx, xMin, iErr)
    IF iErr = 0 THEN
     iBulundu = iBulundu + 1
     xi(iBulundu, 1) = xMin
     xi(iBulundu, 2) = yMin
    END IF
   NEXT iAralik

   IF iBulundu = 0 THEN
    PRINT "["; a; ","; b; "] aralığında minimum bulunamadı(Brent)!":
    END
   END IF

' Global minimumu belirle
   j = 1: GlobalMin = xi(1, 2)
   FOR i = 1 TO iBulundu
    IF xi(i, 2) < GlobalMin THEN
     j = i
     GlobalMin = xi(i, 2)
    END IF
   NEXT i

   PRINT "["; a; ","; b; "] aralığında minimum değerler(Brent):"
   FOR i = 1 TO iBulundu
    PRINT "x="; xi(i, 1), "f(x)="; xi(i, 2),
    IF i = j THEN PRINT " GLOBAL MİNİMUM" ELSE PRINT " Lokal Minimum"
   NEXT i

END 'Brent ana sonu
```

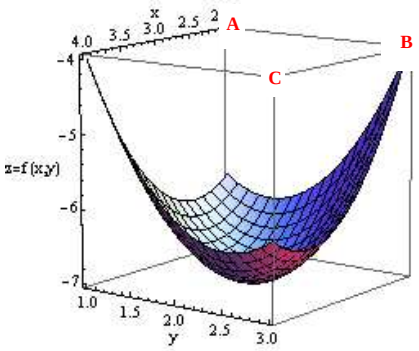
```
FUNCTION Sign (a, b)
  IF b >= 0 THEN Sign = ABS(a) ELSE Sign = -ABS(a)

END FUNCTION 'Sign
```

Brent alt programı

```
FUNCTION Brent (Ax, Bx, Cx, xMin, iErr)
'--------------------------------------------------------------------------
' Findes minimum of a function f(x)
' Given a function f(x), and a bracketing triplet of abscissas
' Ax,Bx,Cx (such that Bx is between AX and CX, and f(Bx) is less
' than both f(Ax) and f(Cx)), this routine isolates the minimum
' to a fractional precision of about TOL using Brent's method.
' The abscissa of the minimum is returned in Xmin, and the minimum
' function value is returned as Brent, the returned function value.
' Fortran code:
'  http://jean-pierre.moreau.pagesperso-orange.fr/Fortran/brent_f90.txt
'--------------------------------------------------------------------------
   EpsMach = 1E-15: ' machine epsilon
   Tol = SQR(EpsMach): ' Tolerance
   iTmax = 500: 'Maximum number of iteration
   Cgold = .381966: 'Golden Ratio
   Zeps = 1E-10: ' instead of zero
   iErr = 0:  ' error flag

   IF Ax < Cx THEN a = Ax ELSE a = Cx
   IF Ax > Cx THEN b = Ax ELSE b = Cx
   V = Bx
   W = V
   x = V
   E = 0
   Fx = fnF(x)
   Fv = Fx
   Fw = Fx
   FOR iTer = 1 TO iTmax:  'main loop
    Xm = .5 * (a + b)
    Tol1 = Tol * ABS(x) + Zeps
    Tol2 = 2 * Tol1
' Test for done here, if satisfiyed then exit section
    IF ABS(x - Xm) <= (Tol2 - .5 * (b - a)) THEN
     xMin = x
     Brent = Fx
     IF fnF(x - 1E-12) < Fx OR fnF(x + 1E-12) < Fx THEN iErr = 1
     EXIT FUNCTION
    END IF

    IF ABS(E) > Tol1 THEN  'Construct a trial parabolic fit
     R = (x - W) * (Fx - Fv)
     Q = (x - V) * (Fx - Fw)
     P = (x - V) * Q - (x - W) * R
     Q = 2 * (Q - R)  'bug corrected 07/24/2006 (0.2 instead of 2)
     IF Q > 0 THEN P = -P
     Q = ABS(Q)
     Etemp = E
     E = D
     IF ABS(P) > ABS(.5 * Q * Etemp) OR (P <= Q * (a - x)) OR (P >= Q * (b - x)) THEN GOTO 1
' The above conditions determine the acceptability of the
' parabolic fit. Here it is o.k.:
     D = P / Q
     U = x + D
     IF (U - a < Tol2) OR (b - U < Tol2) THEN D = Sign(Tol1, Xm - x)
     GOTO 2
    END IF
1   IF x >= Xm THEN
     E = a - x
    ELSE
     E = b - x
    END IF
    D = Cgold * E
2   IF ABS(D) >= Tol1 THEN
     U = x + D
    ELSE
     U = x + Sign(Tol1, D)
    END IF

    Fu = fnF(U) 'This the one function evaluation per iteration
    IF Fu <= Fx THEN
     IF U >= x THEN
      a = x
     ELSE
      b = x
     END IF
     V = W
     Fv = Fw
     W = x
     Fw = Fx
     x = U
     Fx = Fu
    ELSE
     IF U < x THEN
      a = U
     ELSE
      b = U
     END IF
     IF Fu <= Fw OR W = x THEN
      V = W
      Fv = Fw
      W = U
      Fw = Fu
     ELSE
      IF (Fu <= Fv) OR (V = x) OR (V = W) THEN
       V = U
       Fv = Fu
      END IF
     END IF
    END IF
   NEXT iTer
   iErr = 1: ' Brent exceed maximum iterations.'

END FUNCTION ' end of Brent
```

NelderMead programı ve örnekleri:

1. Sağda grafiği verilen $z = f(x,y) = x^2 + y^2 - 4x - y - xy$ çift değişkenli fonksiyonunun bir minimum değeri olduğu görülmektedir. Fonksiyonu minimum yapan x,y koordinatları ve Min Z aranmaktadır.



Nelder-Mead simplex metodu, iterasyona başlayabilmesi için, değişken sayısının bir fazlası kadar noktada değişkenlerin başlangıç değerinin verilmesini gerektirir. Bu örnekte değişkenler x ve y olduğundan değişken sayısı 2 dir. O halde 3 noktadaki x ve y başlangıç değeri verilmelidir. Bu noktalar

1.nokta x=2, y=1
2.nokta x=2, y=3
3.nokta x=4, y=3

olarak seçilmiştir (şekilde A(2,1), B(2,3) ve C(4,3) noktaları). Değişken sayısı ve başlangıç değerleri

```
DATA 2: 'n
'Başlangıç noktalarının koordinatları
 DATA 2,1
 DATA 2,3
 DATA 4,3
```

satırlarında programa verilmiş, $z = f(x,y) = x^2 + y^2 - 4x - y - xy$ fonksiyonu Func(P()) adı ile

```
FUNCTION Func (P())
' minimumu aranan fonksiyonun tanımı
  X = P(1): Y = P(2)
  Func = X * X + Y * Y - 4 * X - Y - X * Y
END FUNCTION
```

tanımlanmıştır. P() vektörü (n+1)xn boyutludur, bu örnekte P(3,2). Fonksiyonun başlangıç değerleri için aldığı z=f(x,y) ordinatları

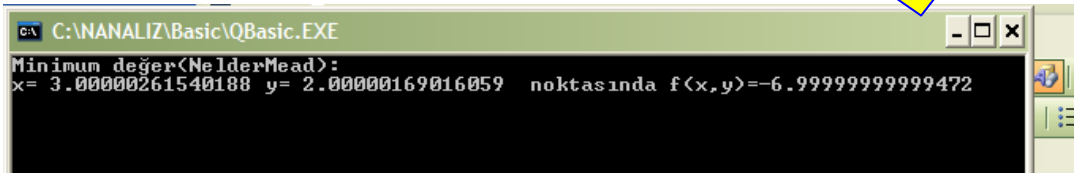
```
' Başlangıç noktalarında fonksiyonun aldığı değerleri hesapla
  FOR i = 1 TO n + 1
   PT(1) = P(i, 1)
   PT(2) = P(i, 2)
   Y(i) = Func(PT())
  NEXT i
```

satırlarında hesaplanıp Y() vektöründe depolanmakta, NelderMead alt programına aktarılarak minimum noktası ve değerinin aranması

```
' NelderMead alt programını çağır, Min değeri ara
  CALL NelderMead(P(), Y(), n, iErr)
```

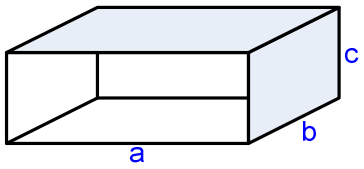
ile başlatılmaktadır. Programdan alınan sonuç:

NelderMead sonucu



Buna göre x=3, y=2 noktasında Min z=f(2,3)=-7 minimum değerini almıştır.

2. Kenarları a, b, c, ön tarafı açık olan 1000 cm^3 hacimli bir kutu imal edilecektir. Toplam yüzeyinin minimum olması için a, b ve c ne olmalıdır.

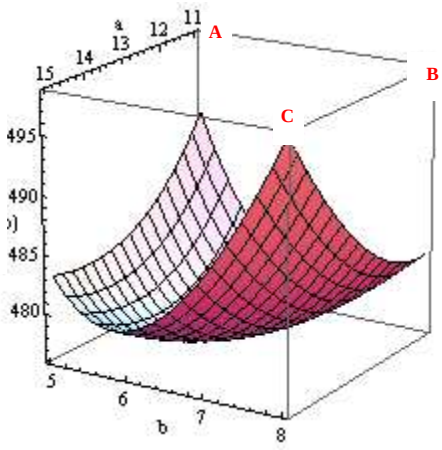


Toplam yüzey: $S = 2ab + 2bc + ac$

Hacim: $V = abc = 1000 \rightarrow c = \dfrac{1000}{ab}$

$$S = 2ab + 2b\frac{1000}{ab} + a\frac{1000}{ab}$$

$$S = 2ab + \frac{2000}{a} + \frac{1000}{b}$$



S sadece a ve b nin fonksiyonudur. Sağda verilen grafiği incelendiğinde bir minimumu olduğu görülür. Değişken sayısı 2 dir (a ve b), o halde 3 noktada a ve b için başlangıç değeri verilmelidir. Bu noktalar
1.nokta a=11, b=5 (A noktası)
2.nokta a=11, b=8 (B noktası)
3.nokta a=15, b=8 (C noktası)

seçilebilir. Programa

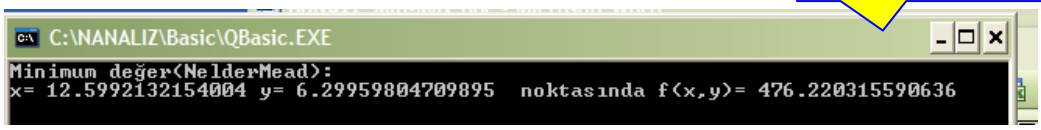
```
DATA 2: 'n
'Başlangıç noktalarının koordinatları
 DATA 11,5
 DATA 11,8
 DATA 15,8
```

satırları ile verilir. S fonksiyonu

```
FUNCTION Func (P())
' minimumu aranan fonksiyonun tanımı
  x = P(1): y = P(2)
  Func = 2 * x * y + 2000 / x + 1000 / y
END FUNCTION
```

şeklinde yazılır(programın yapısı gereği, S yerine Func, a yerine x, b yerine y yazıldığına dikkat ediniz). Program sonucu:

NelderMead sonucu



olur. a=x=12.60 cm, b=y=6.30 cm, Min S=f(x,y)=476.22 cm^2 olmuştur. c kenarı

$V = abc = 1000 \rightarrow c = \dfrac{1000}{ab}$ bağıntısında a ve b yerine yazılarak bulunur:

$$c = \frac{1000}{12.60 \cdot 6.30}$$

$c = 12.60$ cm.

```
'--------------------Ana program NelderMead--------------------------
' f(x,y,z,v, ...) gibi çok değişkenli fonksiyonunun minimum olduğu
' noktanın koordinatlarını, yani x, y, z,v, ... değerlerini ve
' f(x,y,z,v, ...) minimum değerini Nelder-Mead motodu ile hesaplar.
' Nelder-Mead metodunun yaygın kullanılan diğer adı amoeba dır.
' Hesaplanan minimum lokal veya global olabilir. Glabal minimum
' olacağı garantisi yoktur.
' Ahmet Topçu, Eskişehir, 2010
'--------------------------------------------------------------------------
 DEFINT I-N
 DEFDBL A-H, O-Z
 DECLARE SUB NelderMead (P(), Y(), n, iErr)
 DECLARE FUNCTION Func (P())

 DATA 2: 'n
'Başlangıç noktalarının koordinatları
 DATA 2,1
 DATA 2,3
 DATA 4,3

 CLS
 READ n
 DIM P(n + 1, n), Y(n + 1), PT(n + 1)

' Başlangıç koordinatlarını oku
  FOR i = 1 TO n + 1
   FOR j = 1 TO n
   READ P(i, j)
  NEXT j
  NEXT i

' Başlangıç noktalarında fonksiyonun aldığı değerleri hesapla
  FOR i = 1 TO n + 1
   PT(1) = P(i, 1)
   PT(2) = P(i, 2)
   Y(i) = Func(PT())
  NEXT i

' NelderMead alt programını çağır, Min değeri ara
  CALL NelderMead(P(), Y(), n, iErr)

  IF iErr <> 0 THEN
   PRINT iErr; " İterasyon sonunda minimum bulunamadı(NelderMead)!"
   END
  ELSE

' n+1 değerden minimum değeri belirle
   i = 1: yMin = Y(1)
   FOR j = 1 TO n + 1
    IF Y(j) < yMin THEN i = j: yMin = Y(j)
   NEXT j

   PRINT "Minimum değer(NelderMead):"
   PRINT "x="; P(i, 1); "y="; P(i, 2); " noktasında "; "f(x,y)="; Y(i)
  END IF

END ' Ana program NelderMead sonu
```

NelderMead ana programı.

Bu ana program iki değişkenli $f(x,y)= x^2 + y^2 - 4x - y - xy$ fonksiyonu için yazılmıştır.

Fonksiyonun değişken sayısı n=2

x, ve y değişkenlerinin 3 noktadaki başlangıç değerleri

Değişken sayısı okunuyor

x, ve y değişkenlerinin başlangıç değerleri okunuyor

Başlangıç değerleri için f(x,y) ordinatları hesaplanıyor

Fonksiyonun minimum noktasını ve değerini hesaplamak için NelderMead alt programı çağrılıyor

Sonuçlar yazdırılıyor

```
FUNCTION Func (P())
' minimumu aranan fonksiyonun tanımı
  X = P(1): Y = P(2)
  Func = X * X + Y * Y - 4 * X - Y - X * Y
END FUNCTION
```

$f(x,y)= x^2 + y^2 - 4x - y - xy$ fonksiyonunun programda tanımlanması

NelderMead alt programı.

Bu alt program geneldir. Herhangi bir n değişkenli f(x,y,z,v,….) fonksiyonu için yazılmıştır

```
SUB NelderMead (P(), Y(), n, iErr)
'-------------------------------------------------------------------------------
' Multidimensional minimization of the function FUNC(X) where X is
' an n dimensional vector, by the downhill simplex method of Nelder-Mead. Usially
' used other name: amoeba
' Input:
'   matrix P(n+1,n) which rows are the starting coordinates
'   vector Y(n+1) whose components must be preinitialized
'   to the values of FUNC evaluated at the n+1 vertices of P.
'   n is number of varibels of the function FUNC.
' output:
'   P and Y will have been reset to n+1 new points all within
'   Ftol of a minimum function value
'   iErr=0 a minimum found, otherwise not.
' Fortran kodu: http://jean-pierre.moreau.pagesperso-orange.fr/Fortran/tamoeba_f90.txt
'-------------------------------------------------------------------------------
  Ftol = 1E-12: ' Required tolerance
  iErr = 0: ' error flag
  Alpha = 1: Beta = .5: Gamma = 2: itMax = 1000
' Expected maximum number of dimensions, three parameters which define
' the expansions and contractions, and maximum allowed number of iterations.

  DIM Pr(n), Prr(n), Pbar(n)
  Mpts = n + 1
  iTer = 0
1 iLo = 1
  IF Y(1) > Y(2) THEN
   iHi = 1
   iNHi = 2
  ELSE
   iHi = 2
   iNHi = 1
  END IF

  FOR i = 1 TO Mpts
   IF Y(i) < Y(iLo) THEN iLo = i
   IF Y(i) > Y(iHi) THEN
    iNHi = iHi
    iHi = i
   ELSE
    IF Y(i) > Y(iNHi) THEN IF i <> iHi THEN iNHi = i
   END IF
  NEXT i

' Compute the fractional range from highest to lowest and return if
' satisfactory.
  Rtol = 2 * ABS(Y(iHi) - Y(iLo)) / (ABS(Y(iHi)) + ABS(Y(iLo)))
  IF Rtol < Ftol THEN EXIT SUB
  IF iTer = itMax THEN
   iErr = iTer: ' maximum iterations exceeded
   EXIT SUB
  END IF

  iTer = iTer + 1
  FOR j = 1 TO n
   Pbar(j) = 0
  NEXT j
  FOR i = 1 TO Mpts
   IF i <> iHi THEN
    FOR j = 1 TO n
     Pbar(j) = Pbar(j) + P(i, j)
    NEXT j
   END IF
  NEXT i

  FOR j = 1 TO n
    Pbar(j) = Pbar(j) / n
    Pr(j) = (1 + Alpha) * Pbar(j) - Alpha * P(iHi, j)
  NEXT j
  Ypr = Func(Pr())

  IF Ypr <= Y(iLo) THEN
   FOR j = 1 TO n
    Prr(j) = Gamma * Pr(j) + (1 - Gamma) * Pbar(j)
   NEXT j
   Yprr = Func(Prr())
   IF Yprr < Y(iLo) THEN
    FOR j = 1 TO n
     P(iHi, j) = Prr(j)
    NEXT j
    Y(iHi) = Yprr
   ELSE
    FOR j = 1 TO n
     P(iHi, j) = Pr(j)
    NEXT j
    Y(iHi) = Ypr
   END IF
  ELSE
   IF Ypr > Y(iNHi) THEN
    IF Ypr < Y(iHi) THEN
     FOR j = 1 TO n
      P(iHi, j) = Pr(j)
     NEXT j
     Y(iHi) = Ypr
    END IF

    FOR j = 1 TO n
     Prr(j) = Beta * P(iHi, j) + (1 - Beta) * Pbar(j)
    NEXT j
    Yprr = Func(Prr())
    IF Yprr < Y(iHi) THEN
     FOR j = 1 TO n
      P(iHi, j) = Prr(j)
     NEXT j
     Y(iHi) = Yprr
    ELSE
     FOR i = 1 TO Mpts
      IF i <> iLo THEN
       FOR j = 1 TO n
            Pr(j) = .5# * (P(i, j) + P(iLo, j))
            P(i, j) = Pr(j)
       NEXT j
       Y(i) = Func(Pr())
      END IF
     NEXT i
    END IF

   ELSE
    FOR j = 1 TO n
     P(iHi, j) = Pr(j)
    NEXT j
    Y(iHi) = Ypr
   END IF
  END IF
  GOTO 1

END SUB ' NelderMead
```

3. Üç değişkenli $u = f(x,y,z) = x^4 + y^3 + z^4 - 8xyz$ fonksiyonun minimum olduğu x, y, z değerleri ve Min u aranmaktadır. Fonksiyon 3 değişkenli olduğundan 4 noktada başlangıç değeri verilmek zorundadır. Minimumun aranacağı başlangıç değerleri:

1. nokta: x=1, y=0, z=0
2. nokta: x=0, y=1, z=0
3. nokta: x=0, y=0, z=1
4. nokta: x=1, y=1, z=1

olsun. NelderMead alt programı fonksiyon bağımlı değildir, değişmez, fakat ana program ve fonksiyonun tanımlanması aşağıdaki gibi değiştirilmelidir:

```
'--------------------Ana program NelderMead-------------------------
' f(x,y,z,v, ...) gibi çok değişkenli fonksiyonunun minimum olduğu
' noktanın koordinatlarını, yani x, y, z,v, ... değerlerini ve
' f(x,y,z,v, ...) minimum değerini Nelder-Mead motodu ile hesaplar.
' Hesaplanan minimum lokal veya global olabilir. Glabal minimum
' olacağı garantisi yoktur.
' Ahmet Topçu, Eskişehir, 2010
'-------------------------------------------------------------------
 DEFINT I-N
 DEFDBL A-H, O-Z
 DECLARE SUB NelderMead (p(), y(), n, iErr)
 DECLARE FUNCTION Func (p())

 DATA 3: 'n
'Başlangıç noktalarının koordinatları
 DATA 1,0,0
 DATA 0,1,0
 DATA 0,0,1
 DATA 1,1,1

 CLS
 READ n
 DIM p(n + 1, n), y(n + 1), Pt(n + 1)

' Başlangıç koordinatlarını oku
  FOR i = 1 TO n + 1
   FOR j = 1 TO n
   READ p(i, j)
  NEXT j
  NEXT i

' Başlangıç noktalarında fonksiyonun aldığı değerleri hesapla
  FOR i = 1 TO n + 1
   Pt(1) = p(i, 1)
   Pt(2) = p(i, 2)
   Pt(3) = p(i, 3)
   y(i) = Func(Pt())
  NEXT i

' NelderMead alt programını çağır, Min değeri ara
  CALL NelderMead(p(), y(), n, iErr)

  IF iErr <> 0 THEN
   PRINT iErr; " İterasyon sonunda minimum bulunamadı(NelderMead)!"
   END
  ELSE

' n+1 değerden minimum değeri belirle
   i = 1: yMin = y(1)
   FOR j = 1 TO n + 1
    IF y(j) < yMin THEN i = j: yMin = y(j)
   NEXT j

   PRINT "Minimum değer(NelderMead):"
   PRINT "x="; p(i, 1); "y="; p(i, 2); "z="; p(i, 3); " noktasında "; "f(x,y,z)="; y(i)
  END IF

END ' Ana program NelderMead sonu

FUNCTION Func (p())
' minimumu aranan fonksiyonun tanımı
  x = p(1): y = p(2): z = p(3)
  Func = x ^ 4 + y ^ 3 + z ^ 4 - 8 * x * y * z
END FUNCTION
```

NelderMead ana programı.
Bu ana program üç değişkenli $u=f(x,y,z)= x^4 + y^2 + z^4 - 8\ xyz$ fonksiyonu için yazılmıştır.

Fonksiyonun değişken sayısı n=3

x, y ve z değişkenlerinin 4 noktadaki başlangıç değerleri

Değişken sayısı okunuyor

x, y ve z değişkenlerinin başlangıç değerleri okunuyor

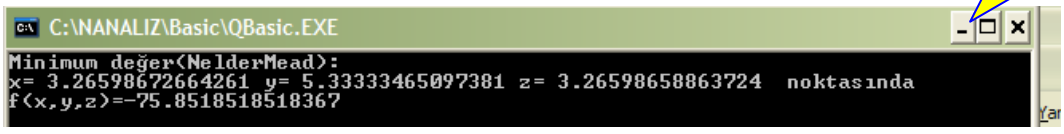
Başlangıç değerleri için u= f(x,y,z) ordinatları hesaplanıyor

Fonksiyonun minimum noktasını ve değerini hesaplamak için NelderMead alt programı çağrılıyor

Sonuçlar yazdırılıyor

$u=f(x,y,z)= x^4 + y^2 + z^4 - 8\ xyz$ fonksiyonunun programda tanımlanması

NelderMead sonucu



x=3.27, y=5.33, z=3.26 noktasında min u=f(x,y,z)=-75.85 bulunmuştur.

SteepestDescent programı ve örnekleri:

1. $f(x,y,z) = Sin(x) + 2Cos(y) - Sin(z)$ fonksiyonunun x=1, y=1 ve z=1 civarında maksimumu var mıdır? Fonksiyonun değişken sayısı n=3 tür DATA satırları

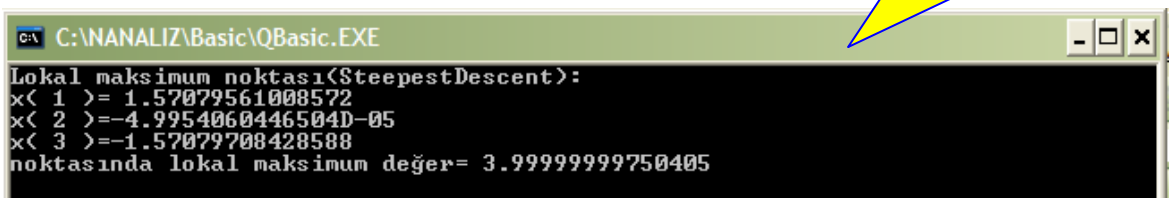
```
DATA 3: ' n değişken sayısı
DATA 1,1,1: ' değişkenlerin başlangıç değerleri
```

olarak verilir ve fonksiyonun tanımı

```
FUNCTION eval (x())
' Definition of Function f(x,y,z) to maximize.
' x(1), x(2),x(3) correspondes variables x,y,z
' Eval is the value of function f(x,y,z)
  x = x(1)
  y = x(2)
  z = x(3)
  eval = SIN(x) + 2 * COS(y) - SIN(z)

END FUNCTION ' eval
```

şeklinde yapılırsa program aşağıdaki sonucu verir:

SteepestDescent sonucu



x=1.57, y=0, z=-1.57 noktasında Max f(x,y,z)=4.0 bulunmuştur.

```
'--------------------Ana program SteepestDescent ------------------
' f(x) veya f(x,y) veya f(x,y,z) fonksiyonunun lokal maksimum
' olduğu noktanın koordinatlarını ve lokal maksimum değerini bulur.
' değişkenler için başlangıç değerleri verilmelidir. Başlangıç
' değerlerinin tanımladığı noktaya en yakın lokal maksimum bulunur.
' Metot: Steepest Descent
' Ahmet TOPÇU, Eskişehir, 2010
'-----------------------------------------------------------------

DEFINT I
DEFDBL A-H, O-Z
DECLARE SUB SteepestDescent (n, x(), iErr)
DECLARE SUB Utilit1 (n, dd, d())
DECLARE SUB Utilit2 (n, xk, dd, d(), y(), x(), x1())
DECLARE SUB Derivatives (n, dd, xk, d(), x(), y())
DECLARE FUNCTION eval (x())

  CLS

  DATA 3: ' n değişken sayısı
  DATA 1,1,1: ' değişkenlerin başlangıç değerleri

  READ n: ' değişken sayısını oku

  DIM x(n): ' değişkenlerin değerlerini içeren vektör

  FOR i = 1 TO n
   READ x(i): ' değişkenlerin başlangıç değerini oku
  NEXT i

  CALL SteepestDescent(n, x(), iErr)

  IF iErr <> 0 THEN
   PRINT "Lokal maksimum bulunamadı(SteepestDescent)"
   END
  END IF

  PRINT "Lokal maksimum noktası(SteepestDescent):"
  FOR i = 1 TO n
   PRINT "x("; i; ")="; x(i)
  NEXT i
  PRINT "noktasında lokal maksimum değer="; eval(x())

END ' ana program SteepestDescent sonu
```

SteepestDescent ana programı.

Bu ana program üç değişkenli f(x,y,z)= Sin(x)+2Cos(y)-Sin(z) fonksiyonu için yazılmıştır.

Değişken sayısı: fonksiyonda x,y,z değişkenleri var anlamında.

Değişkenlerin başlangıç değerleri: x=1, y=1, z=1 anlamındadır

Değişkenlerin başlangıç değerleri x vektöründe depolanıyor: x(1)=x, x(2)=y, x(3)=z anlamında

Lokal maksimumu ara

Eval alt programı.

f(x,y,z)= Sin(x)+2Cos(y)-Sin(z) fonksiyonu tanımlanması

```
FUNCTION eval (x())
' Definition of Function f(x,y,z) to maximize.
' x(1), x(2),x(3) correspondes variables x,y,z.
' Eval is the value of function f(x,y,z)
  x = x(1)
  y = x(2)
  z = x(3)
  eval = SIN(x) + 2 * COS(y) - SIN(z)

END FUNCTION ' eval
```

SteepestDescent alt programı.

```
SUB SteepestDescent (n, x(), iErr)
'------------------------------------------------------------------------
' Steepest descent optimization subroutine
' This routine finds the local maximum of function  f(x,y,z) using
' the method of steepest descent, or the gradient.
' The function must be defined in the function  Eval(X).
' In this version, finite differences are used to calculate the
' partial derivatives.

' INPUTS:
'    n  :number of variables of function
'    Eps : The convergence criteria
'    Maxit: The maximum number of iterations
'    xk : A starting constant
'    X(i): starting values of variables
' OUTPUTS:
'    X(i) : The locally optimum set: x=x(1),y=x(2),z=x(3)
'    iErr=0 : local maximum found found at the point x,y,z
'    iErr<>0: local maximum not found

' Subprograns called: Eval, Derivatives, Utilit1, Utilit2
' Fortran kodu:
'  http://jean-pierre.moreau.pagesperso-orange.fr/Fortran/steepda_f90.txt
'------------------------------------------------------------------------
  DIM x1(n), d(3), y(3): 'working vectors

  Eps = 1E-08: ' Tolerance
  Maxit = 500: ' max number of iteration
  xk = .1:  ' starting constant
  EpsMach = 1E-15: ' Machine epsilon
  it = 0: ' number of itaration
  iErr = 0: ' error flag

' The routine needs three values of Y to get started
' Generate starting D(i) values
' These are not even good guesses and slow the program a little
  dd = 1
  d(1) = 1 / SQR(n)
  FOR i = 2 TO n
    d(i) = d(i - 1)
  NEXT i

' Start initial probe
  FOR i = 1 TO n
' Obtain yy and D[i]
   y(i) = eval(x())
' Update X[i]
   CALL Utilit1(n, dd, d())
   CALL Utilit2(n, xk, dd, d(), y(), x(), x1())
  NEXT i

' We now have a history to base the subsequent search on
' Accelerate search if approach is monotonic
' start iteration
  WHILE it < Maxit
   IF ABS(y(2) - y(1)) >= EpsMach THEN
    IF (y(3) - y(2)) / (y(2) - y(1)) > 0 THEN xk = xk * 1.2
   END IF
' Decelerate if heading the wrong way
   IF y(3) < y(2) THEN xk = xk / 2
' Update the Y[i] if value has decreased
   IF y(3) > y(2) THEN
    y(1) = y(2)
    y(2) = y(3)
   ELSE
' Restore the X(i)
    FOR i = 1 TO n
     x(i) = x1(i)
    NEXT i
   END IF

' Obtain new values
   y(3) = eval(x())
   CALL Derivatives(n, dd, xk, d(), x(), y()): ' Get D(i)
'  if dd=0 then the precision limit of the computer has been reached
   IF dd < EpsMach THEN EXIT SUB
' Update X[i]
   CALL Utilit2(n, xk, dd, d(), y(), x(), x1())

' Check for convergence
   IF ABS(y(3) - y(2)) < Eps THEN EXIT SUB
' Try another iteration
   it = it + 1
  WEND

  iErr = 1: ' not converged

END SUB ' SteepestDescent
```

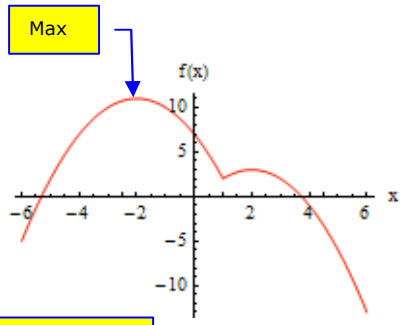
DİKKAT:
xk nın değeri sonucu çok etkiler. Değiştirilirse başka maximum bulunmasına veya bulunamamasına, neden olabilir!
Değiştirerek, örneğin, xk=1 alarak, deneyiniz.

```
SUB Utilit1 (n, dd, d())
' Find the magnitude of the gradient
' Functions called by SteepestDescent and Derivatives
  dd = 0
  FOR i = 1 TO n
   dd = dd + d(i) * d(i)
  NEXT i
  dd = SQR(dd)
END SUB ' Utilit1
```

```
SUB Derivatives (n, dd, xk, d(), x(), y())
' Find approximations of partial derivatives D(i) by finite differences
' Called by SteepestDescent
   FOR i = 1 TO n
    a = x(i): ' Save X(i)
    b = d(i) * xk / (2 * dd): ' Find increment
    x(i) = x(i) + b: ' Move increment in X(i)
    yy = eval(x()): ' Obtain yy
    IF ABS(b) < 1E-12 THEN b = 1E-12: ' Guard against divide by zero near maximum
    d(i) = (yy - y(3)) / b: ' Update D(i)
    IF ABS(d(i)) < .00001 THEN d(i) = .00001: ' Guard against locked up derivative
' Restore X(i) and yy
    x(i) = a
    yy = y(3)
   NEXT i
' Obtain dd
   CALL Utilit1(n, dd, d())

END SUB' Derivatives
```

```
SUB Utilit2 (n, xk, dd, d(), y(), x(), x1())
' Updates the X(i)
' called by SteepestDescent
  FOR i = 1 TO n
   x1(i) = x(i)
   x(i) = x(i) + xk * d(i) / dd
  NEXT i
  y(3) = eval(x())

END SUB ' Utilit2
```

2. Sağda grafiği verilen $y = f(x) = -x^2 + 4|x-1| + 3$ tek değişkenli fonksiyonunun [-6,6] aralığındaki maksimumu SteepestDescent ile aranacaktır.



a)Fonksiyonun tanımlanması için `eval` alt programı değiştirilir:

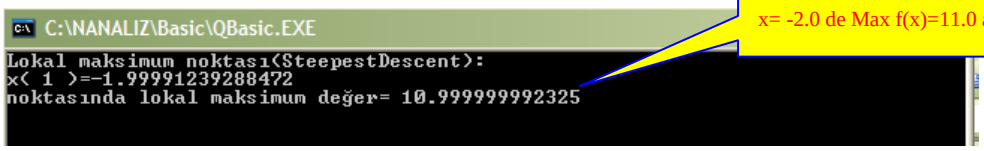
```
FUNCTION eval (x())
' Definition of Function f(x,y,z) to maximize.
' x(1), x(2),x(3) correspondes variables x,y,z.
' Eval is the value of function f(x,y,z)
  x = x(1)
  eval = -x * x + 4 * ABS(x - 1) + 3

END FUNCTION ' eval
```

f(x)= $-x^2+4|x-1|+3$ fonksiyonu

b) Soldaki tepe noktasını bulmak için ana programın DATA satırları aşağıdaki gibi değiştirilir:

```
  DATA 1: ' n değişken sayısı
  DATA -3: ' değişkenlerin başlangıç değerleri
```

c) Program çalıştırılarak sonuç alınır:

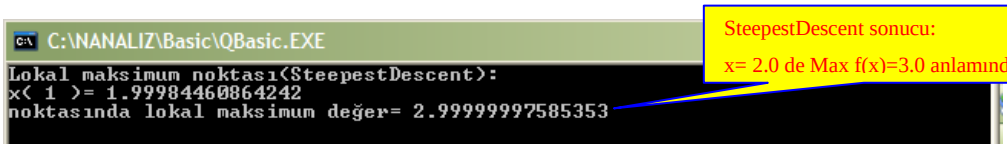


SteepestDescent sonucu:
x= -2.0 de Max f(x)=11.0 anlamında

d) Sağdaki tepe noktasını bulmak için DATA satırları aşağıdaki gibi değiştirilir

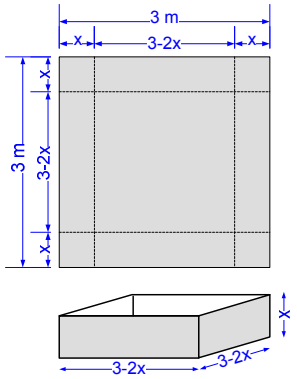
```
  DATA 1: ' n değişken sayısı
  DATA 3: ' değişkenlerin başlangıç değerleri
```

ve program çalıştırılırsa:



SteepestDescent sonucu:
x= 2.0 de Max f(x)=3.0 anlamında

3. 3x3 metre boyutlu sac levhanın kenarları kıvrılarak yüksekliği x olan sıvı tankı yapılacaktır. Tankın hacminin maksimum olması için x ne olmalıdır.



Hacim: $V=x(3-2x)^2$

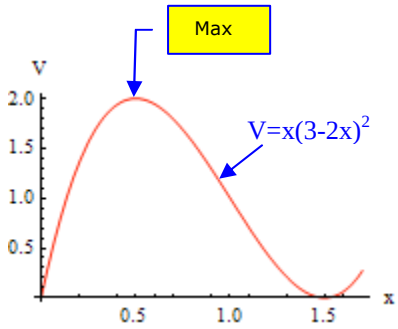
El çözümü:

V nin fonksiyonu çok basit olduğundan el çözümü yapılabilir. V nin x e göre türevinin sıfır olduğu noktalarda V maksimum veya minimum olur:

$V'=3(4x^2 - 8x+3)=0$

Denkleminin kökleri $x_1=0.5$ ve $x_2=1.5$ dir. İkinci türev kullanılarak bu köklerin V yi minimum mu yoksa maksimum mu yaptığı belirlenebilir. Sağdaki grafikten görüldüğü gibi $x=x_1=0.5$ için V(0.5) maksimum olmaktadır. Hacim:

$V=0.5(3-2\cdot0.5)=2\ m^3$ olur.



SteepestDescent Programı ile çözüm:

a)Fonksiyonun tanımlamak için `eval` alt programı değiştirilir:

```
FUNCTION eval (x())
' Definition of Function f(x,y,z) to maximize.
' x(1), x(2),x(3) correspondes variables x,y,z.
' Eval is the value of function f(x,y,z)
  x = x(1)
  eval = x * (3-2*x)^2
END FUNCTION ' eval
```

V= x(3-2x)² fonksiyonu

b) Ana programın `DATA` satırları aşağıdaki gibi değiştirilir:

```
  DATA 1: ' n değişken sayısı
  DATA 1: ' değişkenlerin başlangıç değerleri
```

c) SteepestDescent programı çalıştırılarak sonuç alınır:

SteepestDescent sonucu:
x= 0.5 m de Max V=2 m³ anlamında