プログラム・ノート

科学計算用サブプログラム・ライブラリ（ＳＳＬ）の追加について

49年4月現在，下記のプログラムが当大型計算機センターのＳＳＬとして登録されました。これらは，京都大学大型計算機センターから譲り受けたものです。都合でディスクには登録されていませんので，**利用者はソースカードで入力しなければなりません。**御希望の方には，**ソースプログラムのリスト**をさしあげますので共同利用掛に申し出て下さい。

ライブラリィ名 LMINF

LINEAR MINIMIZATION WITHOUT USING DERIVATIVE

直線上の極小化（微係数を使用しない方法）

作成者：星野　聰

1．概　要

n 次元空間中のベクトルに沿って，関数の極小化を行なう。

関数計算回数を節約するために，内挿において，より小さい関数値を見出せば，RETURNする。

2．使用法

2.1　呼出し手順

CALL LMINF (TEST, N, X, F, YS, EPS, IER, STEP, INIT, X0, X1, X2)

引数	型と種類	内　容
TEST	サブルーチン名	極小化を行なう関数を定義するサブルーチン名である。 このサブルーチンは SUBROUTINE TEST(N, X, F, INIT) の形をもつこと。ここで N　：Xの次元数，整数型変数名，入力 X　：関数Fに対する位置ベクトル，実数型配列名，大きさはX(KK)，KK≧N。 F　：実数型変数名，関数値を入れること。 INIT　INIT＝0ならば関数の計算の他に，必要な initialization も行なようにすること。 INIT≠0のときは関数の計算のみ行なう。整数型変数名。 ㊟ TESTは呼ぶ側のプログラムでEXTERNAL宣言をする必要がある。
N	整数型	極小化を行なう関数の次元数。入力，保存される。

引数	型と種類	内容
X	実数型 1次元配列名	N次元の位置ベクトル。 linear searchの開始点の位置ベクトルをsetして，LMINFに入ること。 returnの際には，linear searchの結果として求められた，位置ベクトルが入っている。 入力および出力。大きさはX(KK), KK≧N。
F	実数型変数名	極小化しようとしている関数の値。 linear searchの開始点における関数値をsetして，LMINFに入ること。 returnの際には，linear searchの結果として求められた位置における関数値が入っている。 入力および出力。
YS	実数型 1次元配列名	linear searchの方向ベクトル。大きさはYS(KK), KK≧N。 サブルーチンLMINFをcallする際にsetしておくこと。 LMINF内で符号が変えられることがある。 LMINFで行なわれる最初のsearchの大きさは，LMINFがcallされたときの引数STEPの大きさと‖YS‖の積にひとしくとられる。入力および出力。
STEP	実数型変数名	LMINFで行なわれる最初のsearchの大きさを決める。従ってLMINFをcallする際にsetしておかねばならない。外挿が行なわれるたびに2倍され，内挿が行なわれるたびに½倍される。入力および出力。
EPS	実数型	内挿を行なった際のsearchによる変位の大きさがEPSの1/100以下になると，内挿を打切ってreturnする。この際IERは－2にsetされる。 この値の選び方は関数の性質により異なるが，テストにおいては，単精度の場合10^{-6}，倍精度の場合10^{-12}を使用した。入力，保存される。
IER	整数型変数名	return conditionを示す。出力。 IER＝－1：LMINFでzero divisionが起ったときにsetされる。 通常，極小点に十分近いときに発生する。 IER＝－2：LMINFで内挿が10回以上行なわれたとき，およびsearchによる変位の大きさがEPSの1/100以下になったときに，内挿を打切り，IER＝－2としてreturnする。 この時，STEPは1.0にsetされる。 IER＝0　：上記以外の時
INIT	整数型変数名	LMINF内でサブルーチンTESTのCALLにおいて引数として用いられる。(TEST参照)。入力，保存される。
X0 X1 X2	実数型 1次元配列名	作業用として用られる。大きさはX0(KK), X1(KK), X2(KK), KK≧N。

2.2 精　度

関数の性質により異なる。

3. 計算方法

n次元空間中のベクトルに沿ってminimizationを行なう。

まず，このベクトルに沿って2点（そのうち1点はsearchの開始点）をとる。新しくとられた点での関数値が開始点の関数値より大きければ，これと反対方向で同距離の点での関数値を求める。この点の関数値も，開始点での関数値より大きければ，第1のstep (extrapolation)

は終る。

もし，開始点の関数値より小さい関数値が現れると，stepを2倍にして，新たな点での関数値を求める。これが前の点より大きい関数値をもてば，extrapolationを終る。逆の場合には，さらにstepを2倍にしてすすむ。これをくり返し行なって，図1のような関係をもつ3点a, b, cを求める。

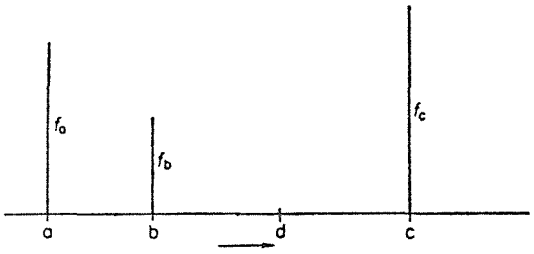

図1

$$f_a > f_b, \quad f_b < f_c$$

次にb, cの中央の点dで関数値f_aを計算して

$$f_a < f_d \text{ ならば 区間 } bc$$
$$f_d \geqq f_b \text{ ならば 区間 } ad$$

を考える。

次に内挿を行なう。これは関数を2次式で近似して微係数がゼロになる所を計算し，区間長を減少させて行く。

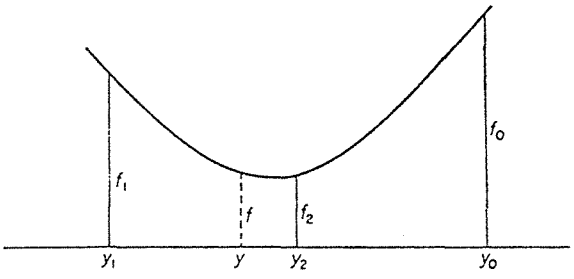

図2

とするとき，

$$y - y_2 = \frac{\dfrac{f_0 - f_2}{y_0 - y_2}(y_1 - y_2) \quad \dfrac{f_1 - f_2}{y_1 - y_2}(y_0 - y_2)}{2\left\{\dfrac{f_0 - f_2}{y_0 - y_2} - \dfrac{f_1 - f_2}{y_1 - y_2}\right\}}$$

で，微係数がゼロになる点の新しい近似とし，ここで関数fを計算する。

$$f > f_2 \quad \text{ならば} \quad \text{区間} \ (y, y_0)$$

$f \leqq f_2$ ならば 区間(y_1, y_2)

を新しい区間とする。

この操作をくり返して行なう。

内挿の打切りは

1．内挿の繰返し回数（LIM＝10回以上）

2．内挿による変位の大きさ（EPST＝EPS＊0.01以上）

によって行なっている。

〔備　　考〕

サブルーチン POW から LMINF を call することもある。

〔参考文献〕

W. H. Swann. Report on the development of a new direct searching method of optimization, C. I. L. Research Note 64/3.

ライブラリィ名 POW

MINIMIZATION OF FUNCTION WITHOUT USING DERIVATIVE

関数の極小化（微係数を使用しない方法）

作成者：星野　聰

1．概　要

関数の極小を求める。関数の偏微係数は必要としない。unconstrained の場合である。

n 次元空間内の点Xから出発して linear search を行ない，より小さい関数値を見出して return する。この subroutine POWをくりかえし call することによって極小を search することができる。linear search の方向は，このサブルーチンの内部で計算される。また，linear search 自身はサブルーチンを内部で call することによって行なわれる。

2．使用法

2.1　呼出し手順

CALL POW(TEST, N, X, F, YS, EPS, STEP, INIT, IER, NN, RITA, XS, FN, Y, X0, X1, X2, DD)

引数	型と種類	内　　容
TEST	サブルーチン名	極小化を行なう関数を定義するサブルーチン名である。 このサブルーチンは， SUBROUTINE TEST (N, X, F, INIT) の形をもつこと。ここで N　：Xの次元数，整数型。 X　：関数Fに対する位置ベクトル，1次元実数型配列名，大きさは X(K), K≧N。 F　：実数型変数名。関数値を入れること。 INIT：INITの項，参照。 ㊟TESTは呼ぶ側のプログラムで EXTERNAL 宣言をする必要ざある。
N	整　数　型	極小化を行なう関数の次元数。入力，保存される。
X	実　数　型 1次元配列名	N次元の位置ベクトル。 最初に，極小化を始める初期値ベクトルの値を set すること。POW から return すると1回の linear search 後の位置がXに入る。大きさは X(K), K≧N。 入力，出力。
F	実数型変数名	極小化しようとしている関数の値。入力，出力。 POWから return すると linear search 後の関数値が入る。
YS	実　数　型 1次元配列名	linear search の方向のベクトル。出力。 サブルーチンPOWの内部で計算されるものである。 大きさは YS(K), K≧N。

引数	型と種類	内　　容
EPS	実　数　型	1回の full powell step を行なった結果生じた displacement が EPS 以下になると converge したものと判定する。 このとき IER に 1 を set して return する。 EPS はまた linear search 自身を行なうサブルーチン LMINF において，内挿の打切りを行なうときの条件としても用いられる。これについては，LMINF の説明を参照のこと。この値の選び方は，関数の性質により異なるが，テストにおいては，単精度の場合，10^{-6} を，倍精度の場合には 10^{-12} を使用した。入力，保存される。
STEP	実数型変数名	サブルーチン LMINF による linear search で用いられる変数である。(LMINF 参照) また STEP は POW によって行なわれる最初の search の大きさにひとしい。
INIT	整　数　型	サブルーチン LMINF の実引数となる（LMINF 参照)。 最初 POW を call する前に INIT をゼロにセットしておく必要がある。
IER	整数型変数名	return codition を示す。出力。 IER＝1　：converge したことを示す。 すなわち，linear search による displacement が EPS 以下になった場合である。 JER＝0　：まだ converge していないことを示す。 つづいて POW を call すればよい。 IER＝－1：サブルーチン LMINF で zero division が起ったことを示す。 通常，充分，解に近いことが多い。 IER＝－2：サブルーチン LMINF で内挿の打切りが起ったことを示す。 （LMINF 参照） POW の計算はそのまま続ければよい。
NN	整　数　型	配列 RITA の DIMENSION の第 1 添字。入力，保存される。NN≧N。
RITA	実　数　型 （2次元配列名）	search の方向 YS の記憶に使用される実数型配列。作業用として用いられる。 大きさは RITA(NN, KK), KK≧N。
XS, Y, X0, X1, X2, DD	実　数　型 1 次元配列名	いずれも，作業用として用いられる一次元配列である。 大きさは各々 N 以上であること。
FN	実　数　型 1 次元配列名	作業用として用いられる一次元配列である。大きさは FN(K), K≧N＋1

2.2 精　度

関数の性質によって異なる。

2.3 使用ルーチン

サブルーチン LMINF，および極小化しようとする関数を定義するサブルーチンが必要である。

2.4 備　考

収束したと判定した場合でも，確かに所要の極小点であるかどうかは，利用者が判定する必要がある。

3．計算方法

POWELL の方法により，関数の極小を求める。すなわち，一次独立な方向 $\xi_1, \xi_2, \cdots, \xi_n$ と点 p_0 が与えられたとき

(1) $r=1, \cdots, n$ に対して，$f(p_{r-1}+\lambda_r \xi_r)$ が ξ_r 方向に対して極小になるように λ_r をきめ，

$$p_r = p_{r-1} + \lambda_r \xi_r$$

とする。

(2) $|f(p_{m-1})-f(p_m)|$ が最大であるような m を見出す。

ただし，$m=1, 2, \cdots, n$ である。

$\Delta = f(p_{m-1})-f(p_m)$ と定義する。

(3) $f_3 = f(2p_n - p_0)$ を計算する。

$f_1 = f(p_0)$, $f_2 = f(p_n)$ と定義する。

(4) $f_3 \geqq f_1$ と

$$(f_1 - 2f_2 + f_3)\cdot(f_1 - f_2 - \Delta)^2 \geqq \frac{1}{2}\Delta(f_1 - f_3)^2$$

の，少はくとも一方が成り立てば，前回の方向 $\xi_1, \xi_2, \cdots, \xi_n$ を次の iteration で再び用いる。また，p_n は次の iteration における p_0 とする。

その他の場合には，

(5) $\xi = (p_n - p_0)$ と定義し，λ を $f(p_n + \lambda\xi)$ が最小ななるようにきめる。次の iteration では $\xi_1, \xi_2, \cdots, \xi_{m-1}, \xi_{+1}\xi_{+1}$ ξ を新しい search の方向とし，また，$p_n + \lambda\xi$ を次回の iteration での starting point とする。

〔参考文献〕

M. J. D. Powell, An efficient methodsfor finding the minimum of a function of several variables without calculating derivatives, Computer Journal, Vol. 7, (1964)

ライブラリィ名 LMIN

LINEAR MINIMIZATION USING DERIVATIVE

直線上の極小化（微係数を使用）

作成者：星野　聰

1．概　要

n 次元空間中のベクトルに沿って関数の極小化を行なう。

関数計算回数を節約するために，内挿においてより小さい関数値を見出せばreturnする。

すなわち，n 次元空間中の一点Xよりベクトル YS 方向（関数が減少する向き）に linear search を行ない，より小さい関数値Fを与える点Xを求める。

2．使用法

2.1　呼出し手順

CALL LMIN(TEST, N, X, F, G, YS, EPS, IER, INIT)

引数	型と種類	内　　容
TEST	サブルーチン名	極小化を行なう関数を定義するサブルーチン名。 このサブルーチンは SUBROUTINE TEST(N, X, F, G, INIT) の形をもつこと。ここで N　：XおよびGの次元数。入力，保存される。 X　：関数Fに対する位置ベクトル。実数型1次元配列名。 大きさは X(KK), KK≧N。 F　：実数型変数名。関数値を入れること。 G　：関数Fに対する gradient ベクトル。実数型1次元配列名。 大きさは G(KK), KK≧N。 INIT：極小化を行なわせるとき，最初にゼロに set する必要がある整数型変数である。すなわち，サブルーチン TEST では，INIT≠0の場合には，関数の計算のみ行なうが，INIT＝0ならば，この他に必要な initialization も行なうようにしておくことが必要である。 TESTは callする側で EXTERNAL 宣言をする必要がある。
N	整　数　型	極小化を行なう関数の次元数。入力，保存される。
X	実　数　型 1次元配列名	N次元の位置ベクトル。 linear search の開始点の位置ベクトルを set して LMIN に入る。 return の際は，linear search の結果として求められた位置ベクトルが入っている。 大きさは X(KK), KK≧N。
F	実数型変数名	極小化しようとしている関数の値。入力，出力。 linear search の開始点における関数値を set しておいて，LMIN を call すること。return の際には linear search の結果として求められた位置における関数値が入っている。

引数	型と種類	内　　容
G	実　数　型 1次元配列名	極小化しようとしている関数のgradientの値。入力，出力。 大きさはG(KK)，KK≧N。 linear searchの開始点における関数のgradientをsetしておいてLMINをcallすること。 returnの際には，linear searchの結果として求められた位置における関数のgradientの値が入っている。
YS	実　数　型 1次元配列名	linear searchの方向のベクトル。 サブルーチンLMINをcallする際にsetする必要がある。 大きさはYS(KK)，KK≧N。 YSの長さは，このベクトルの方向でsearchの開始点から極小点までの距離の推定値にひとしくしておくこと。 ただし，LMINでYSの長さをかえることがある。
EPS	実　数　型	内挿を行なって生じた変位の大きさが，EPS＊0.01以下になれば内挿を打切ってreturnする。このときIERを－4にsetする。 EPSの選び方は，関数の性質により異なるが，テストにおいては，単精度の場合10^{-6}を，倍数度の場合には10^{-12}を選んだ。入力，保存される。
IER	整数型変数名	return conditionを示す。出力。 IER＝－2：searchの方向のgradient成分がゼロまたは正のときにsetされる。ただちにreturnする。 IER＝－3：内挿でzero divisionが発生したときにsetされる。極小点に近い場合が多い。 IER＝－4：linear searchのdisplacementがEPS＊0.01以下になったときにsetされる。 linear searchの開始点が極小点に充分近い場合である。 IER＝0　：上記以外のとき。
INIT	整数型変数名	LMIN内でのサブルーチンTESTのcallにおいて，引数として用いられる。

2.2 精　度

関数の性質によって異なる。

3．計算方法

search vector YSの方向に関数値fが減少していることを確かめてから，外挿を行なって，関数の極小が，区間内にはさまれるようにする。すなわち，より大きい関数値が現われるか，正のgradient（YS方向の成分）が現われるまでくり返す。

fをlinear searchの開始点での関数値，g_bをこの点でのgradientのsearch方向の成分とする。initial search vectorは$\eta \cdot YS(I)$で与えられる。ただし，

$$\eta = -\frac{2f}{g_b \| YS \|}$$

このηが1.0をこえるときは，$\eta=1.0$とする。このsearch vectorの長さは，外挿を行な

うたびに4倍される。

次にcubic interpolationを行なって，区間の両端の関数値よりもより小さい関数値をうるまで内挿をくりかえす。

すなわち，点$|x>$より，方向$|s>$にそって極小を求める。いいかえれば，関数fが極小になる$|x>+\alpha|s>$をきめる。このために一点

$$|y>=|x>+\lambda|s>$$

の情報を用いる。すなわち，x, y点での関数値とgradientをf_x，$|g_x>$，f_y，$|g_y>$とするとき，

$$\frac{\alpha}{\lambda}=1-\frac{<g_y|s>+w-z}{<g_y|s>-<g_x|s>+2w}$$

ここで，

$$w=(z^2-<g_x|s><g_y|s>)^{1/2}$$

$$z=\frac{3}{\lambda}(f_x-f_y)+<g_x|s>+<g_y|s>$$

〔参考文献〕

R. Fletcher, M. J. D. Powell, A rapidly convergent method for minimization, The Computer Journal Vol. 6 (1963)

ライブラリィ名 DAVID

MINIMIZATION OF FUNCTION USING DERIVATIVE

1．概　要

非負の関数の極小を求める。関数の偏微係数を必要とする。unconstrained の場合である。

n 次元空間の点Xから出発して linear search を行ない，より小さい関数値を見出して return する。このサブルーチン DAVID をくり返し call することによって，極小を search することができる。linear search の方向は，このサブルーチンの内部で計算される。また linear search 自身はサブルーチン LMIN を内部で call することによって行なわれる。

2．使用法

2.1　呼出し手順

CALL DAVID (TEST, N, X, F, G, Y, YS, H, NN, EPS, INIT, IER, SIGMA, TEMP1, TEMP2, TEMP3, DIAG,)

引数	型と種類	内　容
TEST	サブルーチン名	極小化を行なう関数を定義するサブルーチン名である（C7/LMIN の TEST の項参照）。
N	整　数　型	極小化を行なう関数の次元数。入力，保存される。
X	実　数　型 1 次元配列名	N次元の位置ベクトル。入力，出力。 最初は極小化を行なわせる初期値ベクトルの値を set すること。 DAVID から return すると，1回の linear search 後の位置が X に入る。 大きさは X(KK)，KK≧N。
F	実数型変数名	極小化しようとしている関数の値。入力，出力。 DAVID から return すると linear search 後の関数値が入る。
G	実　数　型 1 次元配列名	極小化しようとしている関数の gradient ベクトル。出力。 DAVID から return すると，linear search 後の gradient の値が入る。 大きさは G(KK)，KK≧N。
Y	実　数　型 1 次元配列名	linear search の前後における gradient の差が入れられる。出力。 大きさは Y(KK)，KK≧N。
YS	実　数　型 1 次元配列名	linear search の方向のベクトル。 サブルーチン DAVID の内部で計算されるものである。 大きさは YS(KK)，KK≧N。出力。
H	実　数　型 2 次元配列名	Jacobian の逆行列が入れられる。出力。 大きさは H(NN, KK)，KK≧N。
NN	整　数　型	配列Hの DIMENSION の第1添字を与える。入力，保存される。NN≧N。

引数	型と種類	内　　容
EPS	実　数　型	linear searchを行なった結果生じたdisplacementがEPS以下になると収束したものと判定する。このとき，IER＝1にsetしてreturnする。 EPSは，またlinear search自身を行なうサブルーチンLMINにおいて，内挿の打切りを行なうときの条件として用いられる（C7/LMINを参照のこと）。この値の選び方は，関数の性質により異なるが，テストにおいては，単精度の場合10^{-6}を，倍精度の場合は10^{-12}を使用した。入力，保存される。
INIT	整数型変数名	最初DAVIDをcallする前にINITをゼロにsetする必要がある。DAVIDでは，INITにゼロでない値をsetする。入力，出力。 INITはDAVIDの外部で以後，変化させてはならない。
IER	整数型変数名	return conditionを示す。出力。 IER＝1　：収束したことを示す。 IER＝0　：収束していないことを示す。 つづいてDAVIDをcallすればよい。 IER＝－1：行列Hのresetがひきつづいて発生したときである。充分解に近いことが多い。 これは次の場合におこる。 (1) $<\sigma \mid y>\leqq 0$ (2) $<y \mid H \mid y>\leqq 0$ 行列Hは，対角線上がDIAGにひとしく，それ以外の要素はすべてゼロにひとしい行列にresetして，linear searchを再開する。 IER＝－2：サブルーチンLMINにおいて，searchの方向のgradient成分がゼロまたは正のときにsetされるが，DAVIDと共に用いる場合には発生しない。 IER＝－3：サブルーチンLMINにおいて，内挿でzero divisionが発生したときにsetされる。 解に充分近い場合が多い。 IER＝－4：LMINでlinear searchによる変位がEPS＊0.01以下になったときsetされる。 DAVIDについてはそのまま計算をすすめればよい。 INIT＝0の状態で最初にDAVIDをcallすると，IER＝0とinitializeされる。
SIGMA TEMP1 TEMP2 TEMP3	実　数　型 1次元配列名	作業用として用いられる。 大きさは各々N以上であること。
DIAG	実　数　型	配列Hは，最初DAVIDがcallされるとき(INIT＝0のとき)に，対角要素は，すべてDIAGにひとしく，その他の要素は，ゼロにひとしい行列にsetされる。通常，DIAG＝1.0としておけばよい。入力，保存される。

2.2 精　度

関数の性質によって異なる。

2.3 使用ルーチン

サブルーチンLMINおよび極小化しようとする関数を定義するサブルーチンが必要である。

3. 計算方法

DAVIDONの方法により関数の極小を求める。

すなわち，Jacobianの逆行列の近似行列列Hをiterativeに求め，これを用いてNewton近似を行近似を行なう。

Jacobianの逆行列の近似のよりよいに求め，これを用　近似H_{i+1}は

$$H_{i+1}=H_i+\frac{|\sigma_i><\sigma_i|}{<\sigma_i|y_i>}-\frac{H_i|y_i><y_i|H_i}{<y_i|H_i|y_i>} \qquad \cdots\cdots(1)$$

により求められる。ここで，

H_i：Jacobianの逆行列の第i近似

$|\sigma_i>$：前のlinearrsearchによる変位ベクトル

$|y_i>:=|g_{i+1}>-|g_i>$

（現在の位置でのgradientと，linear search $|\sigma_i>$を行なわなかった以前の点でのgradientの差）

新しいsearchは

$$|\sigma_{i+1}>=-\alpha_{i+1}H_{i+1}|g_{i+1}> \qquad \cdots\cdots(2)$$

で与えられる。α_{i+1}は，$-H_{i+1}|g_{i+1}>$方向にlinear searchを行なって極小を求めた結果で決まる係数である。

なこのprocessによりえられるsearch $|\sigma_i>$は，もし関数fが位置$X=(x_1, \cdots, x_n)$のquadratic functionならば互いにconjugateになる。

行列Hは最初は，対角線上がすべて一定数DIAG（＞0）で，その他の要素がゼロになる行列にsetされる。(1)式により，行列H_{i+1}はpositive definiteである。しかし，演算は有限の桁数で行なわれるために$<\sigma_i|y_i>$と$<y_i|H_i|y_i>$が必ずしも正にはならないことがある。この場合にはH_iは，上記の行列（対角線上がDIAG，その他はゼロ）にresetしてから，linear searchをつづける。

〔参考文献〕

(1) W. C. Davidon, Variable metric method for minimization, A. E. C Research and Development Report, ANL-5990 (1959)

(2) R. Fletcher & M. J. D. Powell, A rapidly convergent descent method for minimization, The Computer Journal, Vol. 6 (1963)