

極値探索－ 2

多変数関数：偏導関数を用いる場合

n 変数の関数 $y = f(x_1, \dots, x_n)$ の極小値を求めることを考える。

最急降下法

極小値を求めるための簡単な方法として、まず、適当な点 $\mathbf{x}_0 = (x_{01}, \dots, x_{0n})$ をとり、そこから関数値が減少する方向に \mathbf{x}_0 を変化させるということが考えられる。関数の増加が最大である方向は、次の傾き

$$\text{grad}(f) = \left[\frac{\partial f}{\partial x_1} \quad \dots \quad \frac{\partial f}{\partial x_n} \right]^t$$

で与えられる。したがって、減少が最大である方向は、その逆方向 $-\text{grad}(f)$ である。この $-\text{grad}(f)$ の方向に \mathbf{x}_0 を変化させて極小になる点 \mathbf{x}_1 を求める。このときの極小値は、 $-\text{grad}(f)$ の方向の直線で求めるので、1 変数の関数の最小値を求める方法が使える。 \mathbf{x}_1 が求まると、次にこの \mathbf{x}_1 における $-\text{grad}(f)$ を算出して、その方向に \mathbf{x}_1 を変化させ関数が極小になる点 \mathbf{x}_2 を求める。このように、 $-\text{grad}(f)$ の方向での極小値の探索を繰り返して、各探索での極小値を与える点を \dots 、 \mathbf{x}_i 、 \mathbf{x}_{i+1} 、 \dots と求めていく。この探索を繰り返して、 $-\text{grad}(f)$ が 0 とみなせる点 \mathbf{x}_k 、あるいは $\mathbf{x}_k \approx \mathbf{x}_{k+1}$ となって探索をすすめても次の点がほとんど変わらない点 \mathbf{x}_k に到達したとき、この点 \mathbf{x}_k は極小点であると考えられる。この、減少が最大である方向に探索を進める方法は、最急勾配法 (steepest descent method) (1)、(2) と呼ばれている。

最急勾配法で関数

$$y = f(x_1, x_2) = x_1^2 + 4 \cdot x_2^2$$

の極値探索を点 (1, 1) から行くと図 1 のようになる。探索はジグザグコースで進みながら最小値を与える極小点 (0, 0) に近づいていることが分かる。探索がジグザグコースでなくもっとスムーズに進むようにするためには、探索方向を $-\text{grad}(f)$ から少しずらす必要がある。そのような方法として、共役勾配法がある。

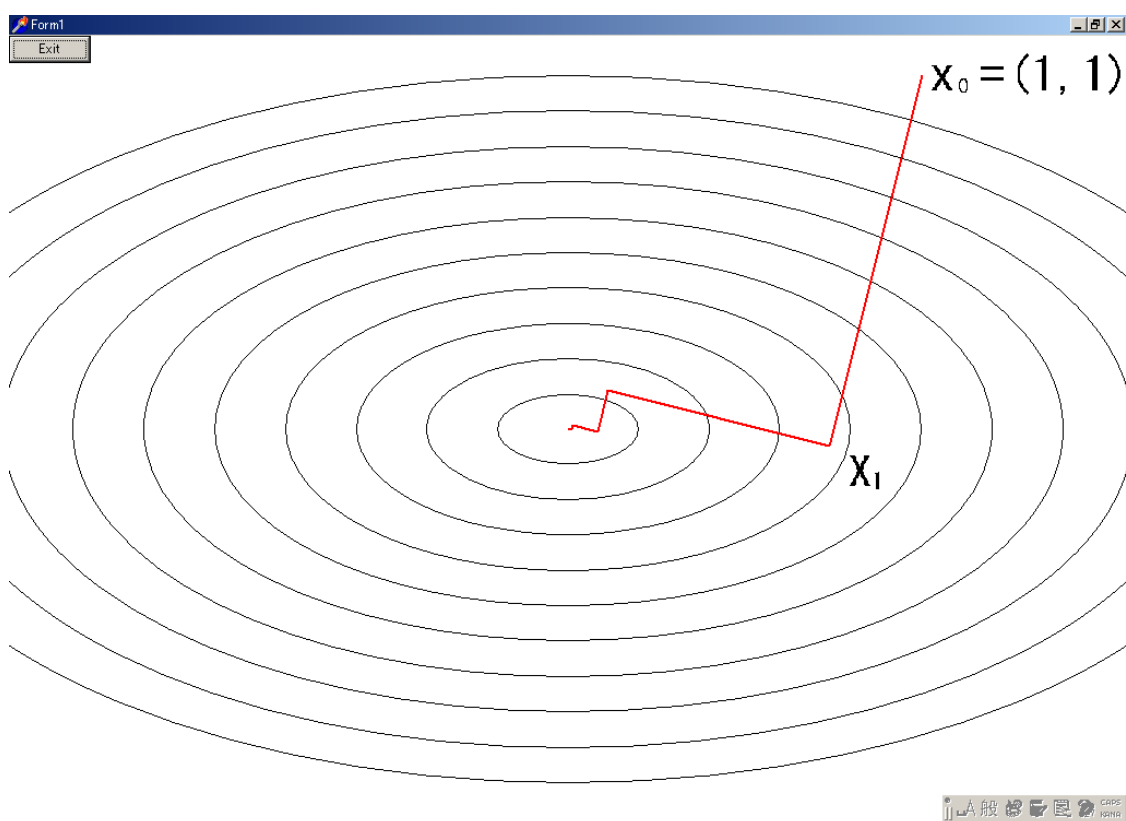


図 1 最急勾配法

共役勾配法 (Fletcher-Reeves method)

共役勾配法と呼ばれている方法では、探索方向を $-\text{grad}(f)$ からずらして探索の効率化を図っている。

$y = f(x)$ が 2 次関数 $\frac{1}{2}x'Ax + B'x + C$ である場合には、それぞれの探索ステップにおける探索方向 (例えば、 k 番目と j 番目のステップにおける探索方向 S_k と S_j) がお互いに行列 A に関して共役、すなわち $S_k'AS_j = 0$ 、であるようにすると効率よく探索が行える。一般の関数に対しては、Fletcher-Reeves の方法⁽¹⁾では、次のようにして探索方向を決めている。

(1) まず、 $i=1$ とおき、

$$\Delta_1 = -\text{grad}(f), \quad S_1 = -\text{grad}(f)$$

とする (初期設定)。

(2) $f(x + \lambda S_i)$ の値を最小にする λ を λ^* とおく。

- (3) $x + \lambda^* S_i$ を次の新しい x とする。この新しい x とその関数値 $f(x)$ の変化の様子を調べて、 x が極小点に収束したかどうかの判定をする。収束したと判定されたときは探索を終了する。収束したと判定できないときは (4) に移る。
- (4) $i = i + 1$ とおく。
- (3) $\Delta_i = -\text{grad}(f)$ を求める。

$$S_i = \Delta_i + \beta_i S_{i-1} \quad (1)$$

とおく。ただし、 $\beta_i = \Delta_i^t \Delta_i / \Delta_{i-1}^t \Delta_{i-1}$ である。

- (5) (2) に戻る。

上の (1) 式によって探索の方向を決めると、関数 $f(\mathbf{x})$ が 2 次関数のときは、各探索方向 S_i が共役になる⁽¹⁾。 $f(\mathbf{x})$ が 2 次関数でない場合でも、局所的に 2 次関数でよく近似できるときは、探索方向 S_i を共役にとることにより効率よく探索が行えることが期待できる。 $f(\mathbf{x})$ が n 個の変数の 2 次関数であるときは、高々 n 回の探索で極小値に達することができる⁽²⁾。2 次関数でない場合は、 n 回の探索で極小値に達するとは限らない。 n 回の探索で極小値に達することができなかった場合は、ステップ (1) に戻ってやり直す (restart)。

図 1 の場合の極値探索を、共役勾配法で行うと図 2 のようになる。

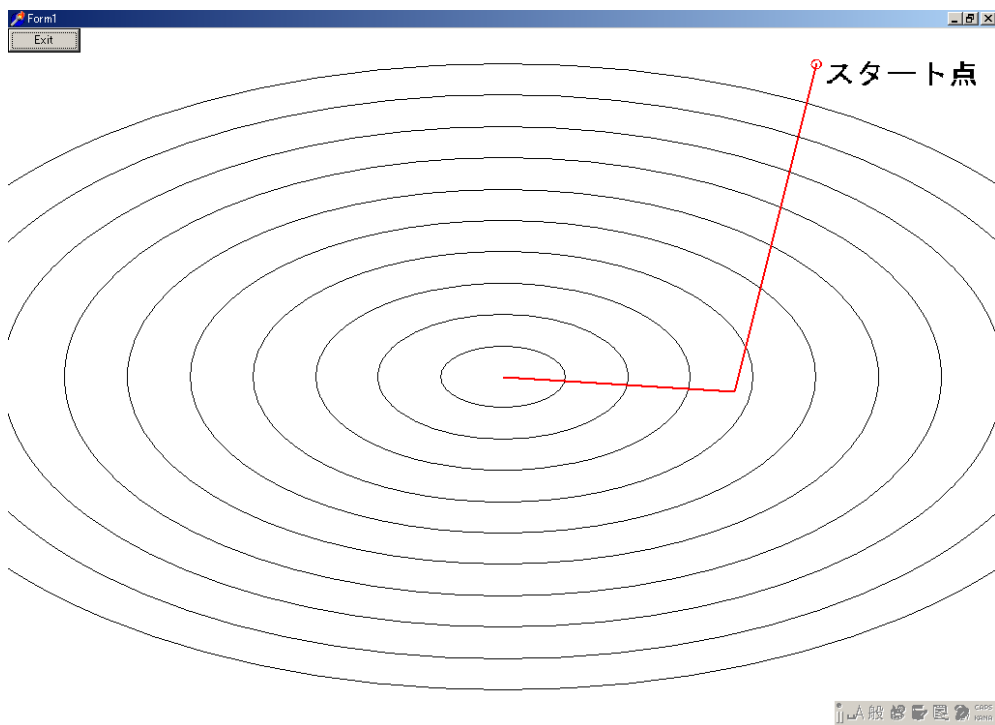


図 2 共役勾配法による極値探索

図 1 の場合と異なり、図 2 では 2 回目の探索の方向が中心（極小値の位置）に向かっていることが分かる。

図 1 はプログラム PDrawPathSteep.dpr により、図 2 はプログラム PDrawPathConj.dpr により描くことができる。それぞれのプログラムの実行開始時に表示されるフォームの GO ボタンをクリックすると、図 1 あるいは図 2 の描画が行われる。

Fletcher-Reeves method によって極小値を求める手続き MinByFR をユニットファイル UOptMultDim.pas に用意した。

手続き MinByFR のヘッダーは、次のようになっている。

```

procedure MinByFR( f : TOptFunc;
                  df : TOptDFunc;
                  n : Longint;
                  var x0 : TOptVector;
                  criterionX, // >= 1.0e-11
                  criterionF // >= 1.0e-17
                  : Extended );

```

第 1 パラメータ f に極小値を求める関数を、第 2 パラメータ df に f の偏導関数 $\partial f / \partial \mathbf{x}$ を設定する。それぞれの関数の型は以下のように宣言されている。

```

TOptVector = array[1..NDimOpt] of Extended;
TOptFunc   = function( x : TOptVector; n : Longint ) : Extended;
TOptDFunc  = function( x : TOptVector; n : Longint ) : TOptVector;

```

ここで、ユニット UOptMultDim では NDimOpt=10 と宣言されている。10 を超える変数を扱うときは、この値を変数の数以上のものに設定する。

第 3 パラメータ n には、変数の個数を設定する。第 4 パラメータには、探索における初期値を設定した変数を表す配列を置く。探索が終了すると、極小値を与える変数値がこのパラメータ（配列）に返される。

第 5、第 6 パラメータには、探索における収束の判定基準を設定する。CriterionX には、変数の変化量に対する基準値を設定する。1 変数の場合の基準と同じく、計算精度（Extended 型の場合は 19~20 桁）の半分くらいが精度の限界と予想される。詳しくは、1 変数の場合の極値探索を扱っているホームページにおいて解説している。CriterionF には、関数値の方の変化量の基準値を設定する。

例えば、次の関数

$$f(x_1, x_2) = (x_1 - 1)^2 + 8 \cdot (x_2 - 1)^2 + 1.0 \quad (1)$$

の極小値を **MinByFR** を用いて求めるときは次のようにする。

まず、関数宣言を

```
function FCheck( x : TOptVector; n : Longint ) : Extended;
begin
    FCheck:=sqr(x[1]-1) + 8*sqr(x[2]-1) + 1.0;
end;
```

と行う。偏導関数は次式

$$\frac{\partial f}{\partial x_1} = 2(x_1 - 1)$$

$$\frac{\partial f}{\partial x_2} = 16(x_2 - 1)$$

で与えられるので、偏導関数の計算を行う関数を次のように宣言する。

```
function DFCheck( x : TOptVector; n : Longint ) : TOptVector;
begin
    Result[1]:=2*(x[1]-1);
    Result[2]:=16*(x[2]-1);
end;
```

上の宣言のもとで、手続き **MinByFR** を次のように呼出す。

```
MinByFR( FCheck, DFCheck, 2, x, 1.0e-9, 1.0e-17 );
```

但し、**MinByFR** を呼出す前に、例えば

```
x[1]:= 5.0;    x[2]:= 5.0;
```

と配列 **x** に初期値を設定しておく。

プロジェクト **PCheckOpt.dpr** では、(1) 式の関数の極小値を上のような手順で求めている。このプロジェクトを実行すると、図3のようにフォームが表示される。

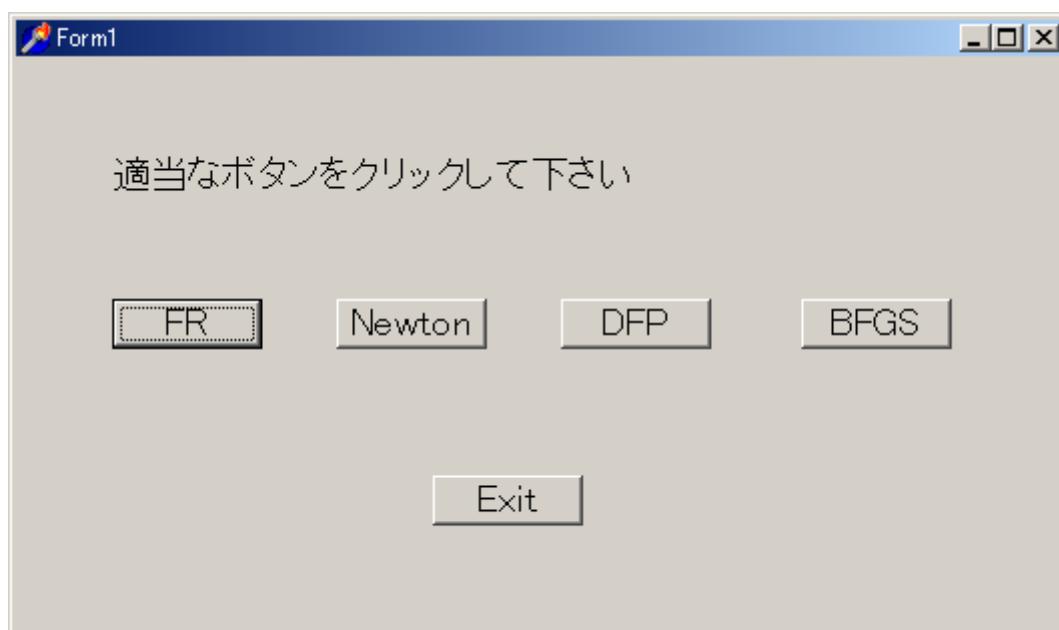


図3 プロジェクト PCCheck0pt. dpr の実行開始時のフォーム

「FR」ボタンのクリックで MinByFR による極値探索が始まり、計算結果が図4のように表示される。

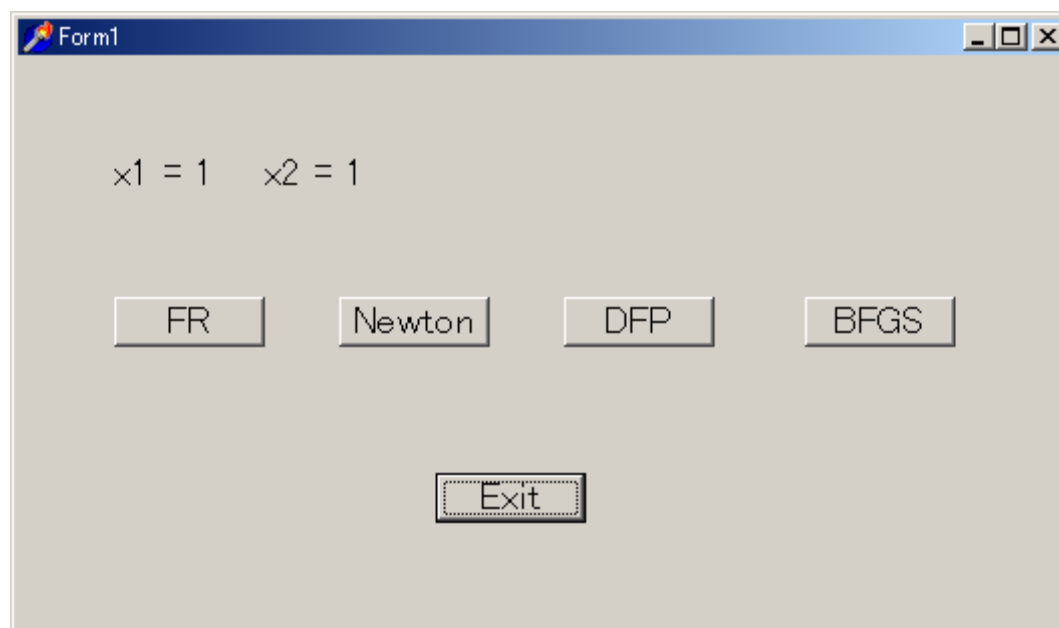


図4 実行結果の表示

(1) 式の関数の最小値 1.0 を与える変数値 $x_1 = x_2 = 1.0$ が表示されている。

図3において、「FR」ボタンをクリックすると画面が一瞬ちらつく。これは MinBy BR の実行時に、計算の途中結果を表示するフォームが表示されているからである。計算が一瞬で終了するため、画面のちらつきとなっている。探索が長く掛かる時は、このフォーム

上の Memo コンポーネントに表示される途中経過をみることによって、探索の進み具合がわかる。しかし、このフォームを表示したくない時は、MinByFR の呼び出し時に実行されるフォームの生成と、終了時に実行されるフォームの廃棄、および Memo コンポーネントに表示する手続きを、すべてコメント記号//などを使ってコメントにしておく。コメントとしておくと、また途中経過表示用のフォームを表示したくなったときは、そのコメント記号をはずすと表示できる。フォームの生成が行われる箇所は MinBy FR の実行開始における

```
FOptMultiDim:=TFOptMultiDim.Create(application);
FOptMultiDim.Visible:=true;
```

であり、フォームの廃棄を行っている箇所は実行終了における

```
FOptMultiDim.Close;
```

である。Memo コンポーネントへの表示は、

```
Display(表示文字列);
```

というように、手続き Display の呼び出しによって行っている。表示を行わないときは、これらの文の前に//を付けてコメントとする。

Newton 法

勾配法では、関数の減少する方向に探索が進められる。最急勾配法では、その点における減少が最大である方向、 $-\text{grad}(f)$ 、に探索が進められ、共役勾配法では探索を滑らかにするために各探索方向が共役になるようにとられた。勾配は接平面、すなわち 1 次関数で極小値を求めている関数を近似する考え方であるといえる。関数を 2 次関数で近似して極小値の探索を行うものとして Newton 法がある。関数 $f(\mathbf{x})$ の点 \mathbf{x}_0 における 2 次関数による近似は次式で与えられる⁽²⁾。

$$f(\mathbf{x}_0 + \mathbf{d}) \approx f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^t \cdot \mathbf{d} + \frac{1}{2} \cdot \mathbf{d}^t \cdot \nabla^2 f(\mathbf{x}_0) \cdot \mathbf{d} \quad (2)$$

ここで、 $\nabla f(\mathbf{x}_0)$ は勾配 (gradient) $\text{grad}(f)$ であり、 $\nabla^2 f(\mathbf{x})$ は 2 次偏導関数を要素とするヘッセ行列 (Hessian Matrix)

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2}, \dots, \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1}, \dots, \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_n} \end{pmatrix}$$

である。

2 次関数

$$g(\mathbf{x}) = \mathbf{a} + \mathbf{x}^t \mathbf{b} + \frac{1}{2} \mathbf{x}^t \mathbf{C} \mathbf{x}$$

の最小値は

$$\mathbf{x}_0 = -\mathbf{C}^{-1} \mathbf{b} \quad (3)$$

で与えられる。これは、次のようにして導くことができる。

$g(\mathbf{x})$ の勾配は、次式で与えられる。

$$\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{b} + \mathbf{C} \mathbf{x}^t$$

最小値を与える点 \mathbf{x}_0 では勾配は 0 なので、次式が成り立つ。

$$\mathbf{b} + \mathbf{C} \mathbf{x}_0^t = 0$$

上式より (3) 式が導かれる。

Newton 法では (3) 式によって極小値の探索を進める。このとき 2 次関数による近似 (2) 式を用いるので、

$$\mathbf{b} = \nabla f(\mathbf{x}), \quad \mathbf{C} = \nabla^2 f(\mathbf{x})$$

とおいた次の式

$$\mathbf{x}_{i+1} = -\{\nabla^2 f(\mathbf{x}_i)\}^{-1} \cdot \nabla f(\mathbf{x}_i) \quad (4)$$

によって、点 \mathbf{x}_i における 2 次関数 (2) 式による近似を用いた探索 ((3) 式、すなわち (4) 式による極小値の探索) を行う。(4) 式によって求めた 2 次関数の極小 (最小) 点 \mathbf{x}_{i+1} は、2 次関数による近似 (2) がよい場合には点 \mathbf{x}_i における関数値 $f(\mathbf{x}_i)$ より小さい値 $f(\mathbf{x}_{i+1}) < f(\mathbf{x}_i)$ を与えるが、近似が良くない場合には $f(\mathbf{x}_{i+1}) > f(\mathbf{x}_i)$ となることがある。また、近似 (2) がよくても、(2) 式の右辺の 2 次関数が上に凸 (お椀を伏せた形) のときは、(3) 式は最大値を与えるものとなり、(4) 式で与えられる点 \mathbf{x}_{i+1} の関数値は点 \mathbf{x}_i における関数値 $f(\mathbf{x}_i)$ より大きくなる可能性が高い。これらのことを考えて、手続き

MinByNewton では次のようにして (4) 式を用いた極小値の探索を行っている。

- (1) 探索の初期値を \mathbf{x}_0 とし、 $\mathbf{x}_i = \mathbf{x}_0$ とおく。
- (2) $\nabla f(\mathbf{x}_i)$ を求める。 $\nabla f(\mathbf{x}_i)$ の大きさが十分に小さければ極小値に達したとして終了する。
- (3) $\nabla^2 f(\mathbf{x}_i)$ を求め、その逆行列 $\{\nabla^2 f(\mathbf{x}_i)\}^{-1}$ を計算する。

(3 a) 逆行列 $\{\nabla^2 f(\mathbf{x}_i)\}^{-1}$ が求まったときは、

$$\mathbf{S} = -\{\nabla^2 f(\mathbf{x}_i)\}^{-1} \cdot \nabla f(\mathbf{x}_i)$$

を極小値の探索方向とする。

(3 b) 逆行列 $\{\nabla^2 f(\mathbf{x}_i)\}^{-1}$ が求まらなかったときは、

$$\mathbf{S} = -\nabla f(\mathbf{x}_i)$$

を探索方向とする。

(4) ステップサイズ s を 1.0 とおく。

(4 a) $f(\mathbf{x}_i + s \cdot \mathbf{S}) < f(\mathbf{x}_i)$ であれば、 $\mathbf{x}_i + s \cdot \mathbf{S}$ を新しく \mathbf{x}_i とおいて (2) に戻る。

$f(\mathbf{x}_i + s \cdot \mathbf{S}) < f(\mathbf{x}_i)$ でないときは (4 b) にすすむ。

(4 b) $f(\mathbf{x}_i - s \cdot \mathbf{S}) < f(\mathbf{x}_i)$ であれば、 $\mathbf{x}_i - s \cdot \mathbf{S}$ を新しく \mathbf{x}_i とおいて (2) に戻る。

$f(\mathbf{x}_i - s \cdot \mathbf{S}) < f(\mathbf{x}_i)$ でないときは (4 c) にすすむ。

(4 c) s の大きさを $1/2$ にする。 s の値が十分に小さくなっているときは極小値に達しているとみなして探索を終了する。 s がまだ十分に小さくないときは (4 a) に戻る。

上の手順で極小値を求める手続き MinByNewton をユニットファイル UOptMultDim.pas に宣言した。手続き MinByNewton のヘッダーは、次のようになっている。

```
MinByNewton( f    : TOptFunc;
              df   : TOptDFunc;
              ddf  : TOptDDFunc;
              n    : Longint;
              var x0 : TOptVector;
```

```

criterionX, // >= 1.0e-11
criterionF  // >= 1.0e-17
: Extended );

```

第1パラメータに極小値を求める関数 f を設定する。第2、第3パラメータには勾配 $\nabla f(\mathbf{x})$ とヘッセ行列 $\nabla^2 f(\mathbf{x})$ を与える関数を設定する。第4パラメータには変数の数を設定する。第5パラメータには変数を表わす配列を設定するが、この配列には極値探索の初期値を設定しておく。MinByNewton の終了時には、探索した極値を与える変数値が第5パラメータに設定した配列に返される。第6、第7パラメータには、極値探索における収束判定のための変数の変化量と関数値の変化量の基準値を設定する。

手続き MinByNewton によって式(1)の関数の極値探索を行うときは、手続き MinByFR のときと同じように関数 FCheck と勾配を与える関数 DFCheck を宣言しておく。さらに、ヘッセ行列を与える関数を

```

function DDFCheck( x : TOptVector; n : Longint ) : TOptMat;
begin
    Result[1,1]:=2.0;
    Result[1,2]:=0.0;
    Result[2,1]:=0.0;
    Result[2,2]:=16.0;
end;

```

と宣言しておく。ここで、型 TOptMat は、ユニット UOptMultDim に

```
TOptMat = array[1..NDimOpt, 1..NDimOpt] of Extended;
```

と宣言されている。

上の宣言のもとで、配列 x に適当な初期値を設定してから手続き MinByNewton を

```

MinByNewton( FCheck, DFCheck, DDFCheck, 2,
             x, 1.0e-9, 1.0e-15 );

```

と呼出すと、配列 x に極小値を与える変数値が返される。

プロジェクト PCheckOpt.dpr では、初期値を、

```
x[1] := 5.0;    x[2] := 5.0;
```

としている。このプロジェクトの実行開始時のフォームは図 3 のようになっているが、「Newton」ボタンをクリックすると手続き MinByNewton による極小値の探索が始まり、計算結果は図 4 のように表示される。MinByNewton 実行時の画面のちらつきをなくするためには、MinByNewton の実行時のフォームの生成と廃棄、およびそのフォーム上の Memo コンポーネントへの表示をコメントなどにして無効としておく。

準 Newton 法 (DFP 公式と BFGS 公式)

Newton 法においては、ヘッセ行列 $\nabla^2 f(\mathbf{x})$ が必要である。ヘッセ行列は、2 次の偏導関数を要素とするものである。探索方向の計算において、ヘッセ行列の逆行列 $\{\nabla^2 f(\mathbf{x})\}^{-1}$ が用いられている。この逆行列を、ヘッセ行列 $\nabla^2 f(\mathbf{x})$ の逆行列として直接与えずに、探索の各ステップで逐次近似していく方法がある。このような方法として、Davidon-Fletcher-Powell (DFP) 公式、および Broyden-Fletcher-Goldfarb-Shanno (BFGS) 公式がよく知られている⁽²⁾。これらの方法では、探索は基本的には Newton 法と同じ手順であるが、 $\{\nabla^2 f(\mathbf{x})\}^{-1}$ の逐次近似の計算と、この逐次近似の性質をよくするために各探索ステップにおける探索方向上での 1 次元の極値探索の精度を上げる必要がある。具体的には次のような手順になる。

DFP 公式による場合：

- (1) 変数の初期値を \mathbf{x}_0 、探索回数を $k = 0$ とおく。
- (2) $\{\nabla^2 f(\mathbf{x})\}^{-1}$ の近似の初期値 \mathbf{H}_0 を単位行列 \mathbf{I} とおく。
- (3) k 回目の探索における変数の出発点 \mathbf{x}_k での勾配ベクトル $\nabla f(\mathbf{x}_k)$ を求める。 $\nabla f(\mathbf{x}_k)$ の大きさが十分に小さいときは、極小値に達しているとみなして探索を終了する。 $\nabla f(\mathbf{x}_k)$ の大きさが十分に小さくないときは、探索方向 \mathbf{S}_k を次式で与える。

$$\mathbf{S}_k = -\mathbf{H}_k \cdot \nabla f(\mathbf{x}_k)$$
- (4) 探索方向 \mathbf{S}_k での 1 次元の探索を行う。すなわち、 $f(\mathbf{x}_k + \lambda \cdot \mathbf{S}_k)$ を最小にする λ を求め、 $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda \cdot \mathbf{S}_k$ とおく。
- (5) $k > n$ のときは $k \leftarrow 0$ とおいて (2) に戻る。ここで、 n は変数ベクトルの次元である。
- (6) 点 \mathbf{x}_{k+1} における勾配ベクトル $\nabla f(\mathbf{x}_{k+1})$ を求める。 $\nabla f(\mathbf{x}_{k+1})$ の大きさが十分に小さいときは、極小値に達しているとみなして探索を終了する。

$\nabla f(\mathbf{x}_{k+1})$ の大きさが十分に小さくないときは、 \mathbf{H}_{k+1} を次式 (DFP 公式) で与える。

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{s}_k \mathbf{s}_k^t}{\mathbf{s}_k^t \mathbf{y}_k} - \frac{(\mathbf{H}_k \mathbf{y}_k)(\mathbf{H}_k \mathbf{y}_k)^t}{\mathbf{y}_k^t (\mathbf{H}_k \mathbf{y}_k)}$$

ここで、 $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ 、 $\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$ である。

(7) $k \leftarrow k+1$ において (3) に戻る。

BFGS 公式による手順は、ステップ (6) における \mathbf{H}_k の更新を次式で行うことを除いて、上の DFP 公式によるものと同じである。

BFGS 公式：

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \left(1 + \frac{\mathbf{y}_k^t \mathbf{H}_k \mathbf{y}_k}{\mathbf{s}_k^t \mathbf{y}_k}\right) \cdot \frac{\mathbf{s}_k \mathbf{s}_k^t}{\mathbf{s}_k^t \mathbf{y}_k} - \left(\frac{\mathbf{s}_k (\mathbf{H}_k \mathbf{y}_k)^t + (\mathbf{H}_k \mathbf{y}_k) \mathbf{s}_k^t}{\mathbf{s}_k^t \mathbf{y}_k} \right)$$

DFP 公式による極小値の探索を行う手続き MinByDFP をユニットファイル UOptMultDim.pas に宣言した。MinByDFP のヘッダーは、次のようになっている。

```
procedure MinByDFP( f    : TOptFunc;
                   df   : TOptDFunc;
                   n    : Longint;
                   var x0 : TOptVector;
                   criterionX, // >= 1.0e-11
                   criterionF // >= 1.0e-17
                   : Extended );
```

つまり、Fletcher-Reeves 法による探索を行う手続き MinByFR と同じである。使用法も同じで、MinByFR のときと同じように関数と勾配の宣言を行ってから、次のように呼出す。

```
MinByDFP( FCheck, DFCheck, 2, x, 1.0e-9, 1.0e-15 );
```

BFGS 公式による手続き MinByBFGS もユニットファイル UOptMultDim.pas に宣言した。この手続きのヘッダーも

```
procedure MinByBFGS( f    : TOptFunc;
```

```

df : TOptDFunc;
n   : Longint;
var x0 : TOptVector;
criterionX, // >= 1.0e-11
criterionF  // >= 1.0e-17
           : Extended );

```

となっていて、MinByFR と同じである。使用法も同じように、例えば

```
MinByBFGS( FCheck, DFCheck, 2, x, 1.0e-9, 1.0e-15 );
```

と呼び出す。

手続き MinByDFP と MinByBFGS の使用例も、プロジェクト PCheckOpt.dpr にある。実行開始時のフォーム（図 3）において、「DFP」ボタン、あるいは「BFGS」ボタンをクリックすると、MinByDFP あるいは MinByBFGS による（1）式の関数の極小値の探索が始まり、計算結果が図 4 のように表示される。

感覚の尺度の作成

極値探索の例として、感覚の強さを表わす尺度の作成を行ってみる。重さや長さなど物理量を表わす場合には、グラム g やメートル m などの単位によって、100g の重りとか 1 m の棒というように表わしている。音の大きさとか物の重さの感じという感覚的なものも量として表わすことができると考え、これらを数値で表わしたものは（感覚の）尺度と呼ばれている。物理量と感覚の尺度の関係を表わす関数としては、べき関数がよく知られている⁽³⁾。ここでは、点の数とその数の多さの感じの関係を調べる簡単な実験について説明する。

n 個の点の数の多さの感じを表す量を ϕ で表す。点の数 n とその多さの感覚量 ϕ がべき関数の関係にあるとは、次の関係が成立することをいう。

$$\phi = \alpha(n + \beta)^\gamma$$

上式では、物理量が点の個数なので n で表わされているが、一般には物理量は重さや長さなどの連続量なので、物理量の方は φ で表わして、

$$\phi = \alpha(\varphi + \beta)^\gamma \quad (5)$$

というように書かれる。

(5) 式において、 α は感覚量の単位によって決まるものである。物理量の場合でいうと、mを単位としているものをcmを単位とするものに換えると数値は100倍になる。感覚量の場合も、どの量を数値1で表わすのか(単位の決定)によって α の値が決まる。式を簡単にするために、ここでは $\alpha=1$ であるように単位を決めるものとする。物理量と感覚量の本質的な関係は、 β と γ によって表されていると考えられる。 $-\beta$ は、感覚量 ϕ が0になる物理量 φ を表わす。 γ の値が大きい程、 φ の増加に伴う ϕ の増加は急激になる。 γ の値は、面の大きさ(広さ)の感覚の場合で0.7、電気ショックの場合で3.5となっている⁽³⁾。面の広がりを感じるは、物理的な面の大きさ(面積)の増加に対して、広さの感じの方は緩やかであるといえる。電気ショックの場合は、電流の増加に対してショックの大きさは急激に強くなるといえる。

β と γ の値を決めるための実験の1例として、以下のものがある。

幾つかの点がランダムに配置されたパターンをディスプレイの左側に表示する。右側にも同じようなランダムな点のパターンを表示するが、右側の点の数の大きさの感じが左側のものの半分であるように調節するものとする。左側の点の数が φ_L 、右側の点の数が φ_R のとき、右側の点の数の感じが左側の半分であると感じられたとする。左側および右側の点の数の感じを表わす感覚量を ϕ_L および ϕ_R で表わすと、右側の感じが左側の半分であるということは

$$\phi_L = 2\phi_R$$

が成り立つということである。

上式と(3)式から次式を得る。

$$(\varphi_L + \beta)^\gamma = 2(\varphi_R + \beta)^\gamma$$

これより、

$$\begin{aligned}\varphi_L + \beta &= 2^{1/\gamma}(\varphi_R + \beta) \\ \varphi_R &= 2^{-1/\gamma}(\varphi_L + \beta) - \beta \\ &= 0.5^{1/\gamma}\varphi_L + \beta(0.5^{1/\gamma} - 1)\end{aligned}\tag{6}$$

を得る。左側の点の数 φ_L に対して、数が半分であるという感じを与える右側の点の数 φ_R が理論的に(6)式で与えられる。この理論値 φ_R と実験データの値との差ができるだけ小さくなるように β と γ の値を決める。

左側の点の数として、 φ_{L1} 個、 \dots 、 φ_{Lk} 個のk種類のパターンを用いたとする。左側の点の数 φ_{Li} に対して、半分の感じと判断された右側の点の数を D_i で表わす。また、左側の点の数 φ_L に対して、(6)式で与えられる右側の数 φ_R を関数として

$$\varphi_R = p(\varphi_L) = 0.5^{1/\gamma} \varphi_L + \beta(0.5^{1/\gamma} - 1)$$

と表わす。このとき、データ D_i と理論値 $p(\varphi_L)$ との差の 2 乗和 SSE は、次式で与えられる。

$$SSE = \sum_{i=1}^k (D_i - p(\varphi_{Li}))^2 \quad (7)$$

SSE を β と γ の関数とみて、SSE を最小にする β と γ の値を極値探索法によって求める。ここでは手続き MinByFR (Fletcher-Reeves の方法) を用いてみる¹。この方法においては、勾配、すなわち、1 次の偏導関数が必要である。(7) 式の偏導関数は

$$\begin{aligned} \frac{\partial SSE}{\partial \beta} &= \sum_{i=1}^k 2(D_i - p(\varphi_{Li})) \cdot \left(-\frac{\partial p(\varphi_{Li})}{\partial \beta} \right) \\ \frac{\partial SSE}{\partial \gamma} &= \sum_{i=1}^k 2(D_i - p(\varphi_{Li})) \cdot \left(-\frac{\partial p(\varphi_{Li})}{\partial \gamma} \right) \end{aligned}$$

で与えられる。ここで、

$$\begin{aligned} \frac{\partial p(\varphi_{Li})}{\partial \beta} &= 0.5^{1/\gamma} - 1 \\ \frac{\partial p(\varphi_{Li})}{\partial \gamma} &= (\varphi_{Li} + \beta) \cdot 0.5^{1/\gamma} \cdot \log 0.5 \cdot (-\gamma^{-2}) \end{aligned}$$

である。

感覚尺度構成のための実験

φ_{Li} 個の点のパターンに対して、その半分の感じを与える点の数 D_i を求める実験を行うプログラム PExpPower.dpr を用意した。このプログラムを実行すると、図 5 のようなフォームが表示される。

¹ $a = 0.5^{1/\gamma}$ 、 $b = \beta(0.5^{1/\gamma} - 1)$ とおくと (6) 式は $\varphi_R = a\varphi_L + b$ となる。このとき、(7) 式で与えられる SSE を最小にするパラメータ値は回帰直線の係数として求めることができる。ここでは、極値探索法の使用例として SSE を最小にする γ と β の値を直接求めている。

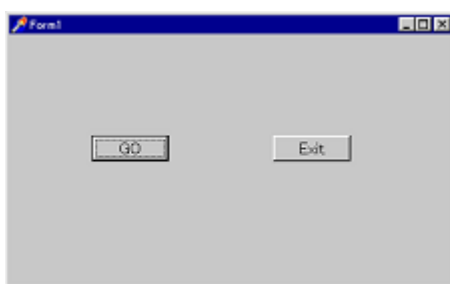


図 5 PExpPower.dpr の実行開始時のフォーム

「GO」 ボタンのクリックで、図 6 のフォームが表示される。



図 6 図 5 の「GO」 ボタンのクリックで表示されるフォーム

「ここをクリックして下さい」とあるところをクリックすると、図 7 の画面になる。

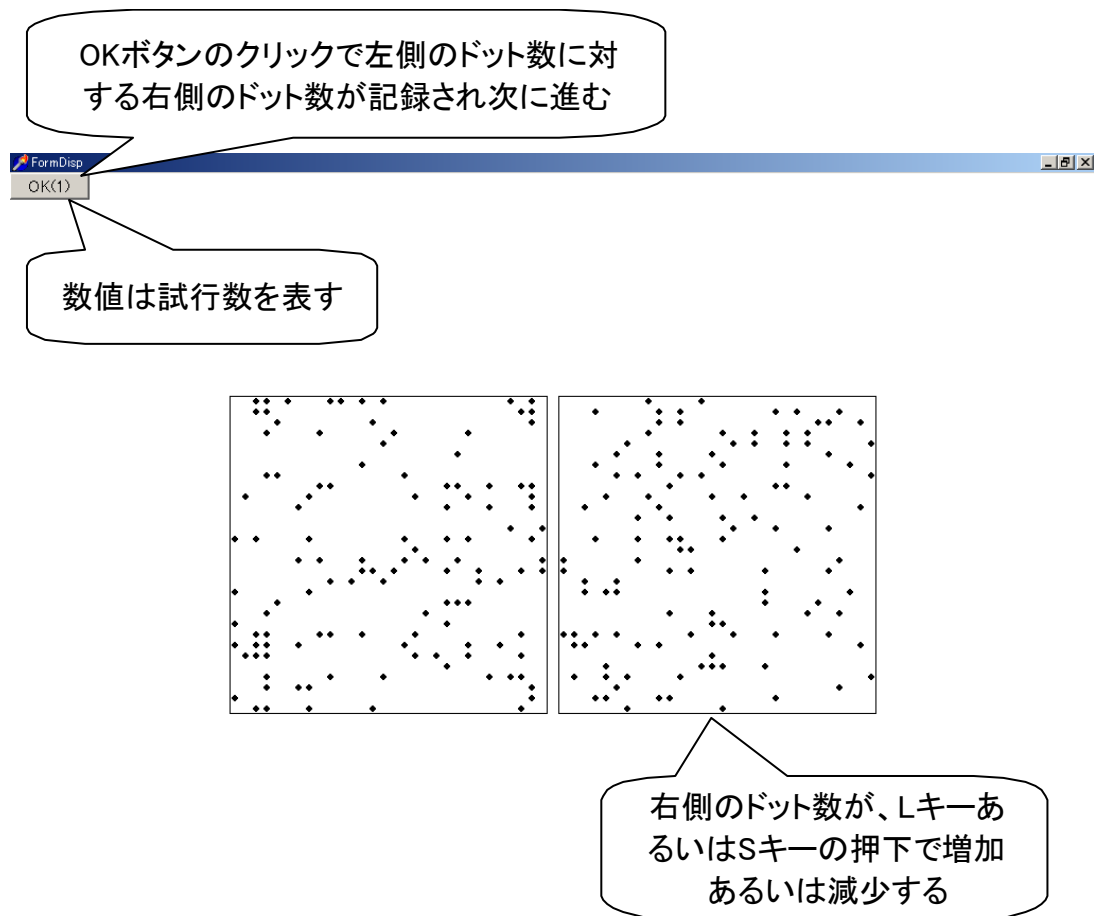


図7 図6の「ここをクリックして下さい」のクリックで表示された画面

この画面において、右側のドットの数が左側の半分の感じになるように調整する。客観的に個数がちょうど $1/2$ のなるようにするのではなく、主観的印象として右の点の数が左の半分ぐらいであるようにする。決して1つ1つ点を数えたりしてはならない。図7の場合は、右の点の数は左の半分より多いので点を減らすことになる。Lキーを押すと点が1つ1つ消えていく。減らし過ぎたときはSキーを押すと点が増える。LキーとSキーの操作で点の増減を行い、右側の点の数がちょうど左側の半分になったと感じたら画面左上の「OK(1)」と表示されているボタンをクリックする。() 内の数字1は、第1試行目(1回目の点の数の調整)であることを表わしている。このボタンをクリックすると、右側の点の数がデータとして保存されて、次のパターン(第2試行目)に移る。第2試行目も第1試行目と同じ要領であるが、提示される点のパターンは異なる。右側の点の数が左側のものの半分に感じられるように調整した後、「OK(2)」ボタンをクリックして第3試行に進む。試行数は、第20試行まで用意されている。

提示される点のパターンは毎回異なるが、左側に提示される点の数は、120個、150個、190個、240個、300個のうちのいずれかである。各試行において最初に提示される右側の

点の数は、左のものと同じ個数か0個である。左側に提示されるそれぞれの点の数に対して4試行（すなわち、4回）、点の配置を変えたものがランダムな順序で提示される。つまり、1つの ϕ_{Li} に対して4つの D_i がデータとして得られる。これら4つのデータ値 D_i の中央値を、それぞれの ϕ_{Li} に対して半分の感じを与える右側の点の数とする。これらの中央値を、(7)式における ϕ_{Li} に対する D_i としてSSEを算出する。 β と γ の値は、このSSE値を最小にするものとして求める。

左側の点の数が5種類であり、各個数が4回提示されるので、実験の総試行数は $5 \times 4 = 20$ 試行である。第20試行目において「OK(20)」ボタンをクリックすると実験は終了して、データの分析が始まる。

分析が始まると、まず、データと計算結果を書き出すテキストファイルの名前の設定を求めるダイアログボックスが表示される（図8）。

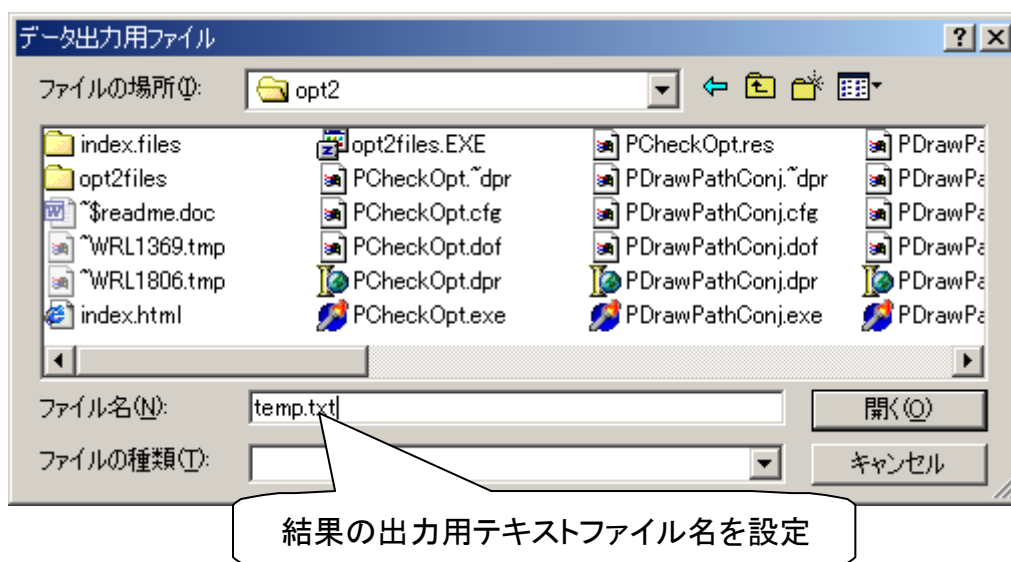


図8 実験データと分析結果の書き出し用テキストファイル名の設定

適当なファイル名を設定して「開く」ボタンをクリックすると計算が始まる。計算結果は図9のように表示される。

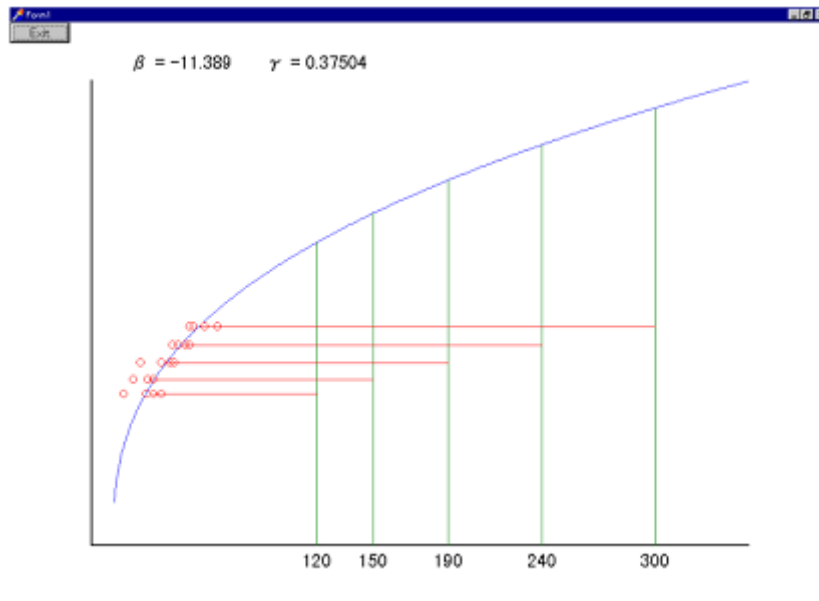


図9 データの分析結果の表示

β と γ の値は、画面左上に表示されている。図9の場合、 $\gamma < 1$ なので、 $\phi = (\varphi + \beta)^\gamma$ のグラフは上に凸になっている。このグラフは、画面では青の曲線で描かれている。横軸には、実験で用いた点の数 φ_{Li} の位置にドット数が表示され、緑の縦線が引かれている。縦軸の目盛りは表示されていない。これは、 ϕ の数値が(5)式の α に依存するという形で、単位に任意性があるからである。ここでは、 ϕ の相対的な大小の比率が意味をもっている。

赤の横線は、 φ_{Li} を表わす緑の縦線の $1/2$ のところからSSEの計算に用いた D_i の値の位置まで引かれている。赤の小円は、各 φ_{Li} に対して $1/2$ の感じを与える数として得られた4つのデータ値を表わしている。

変数変換

β と γ の値を極値探索によって求めるとき、(5)式において、 $\gamma > 0$ かつ $\varphi + \beta > 0$ の制約がある。 φ_{Li} の最小値が120なので、 $\beta > -120$ でなければならない。パラメータ β と γ がこれらの制約条件を満たすようにするために、次の変数変換

$$\begin{aligned}\beta &= -120 + b^2 \\ \gamma &= 0.01 + c^2\end{aligned}$$

を行っている。 γ が0.01を加える形になっているのは、 γ の値が0に近づいたとき実行時演算エラーが生起する可能性があるからである。(7)式を b と c の関数とみて最小化(極小化)を行う。このとき、 b と c には変域の制約がないので、手続きMinByFRによってSSEの最小値を与える b と c を求めることができる。

変数変換後の勾配は、次のように与えられる。

$$\frac{\partial SSE}{\partial b} = \frac{\partial SSE}{\partial \beta} \cdot \frac{d\beta}{db}$$

$$\frac{\partial SSE}{\partial c} = \frac{\partial SSE}{\partial \gamma} \cdot \frac{d\gamma}{dc}$$

ここで、

$$\frac{d\beta}{db} = 2b, \quad \frac{d\gamma}{dc} = 2c$$

である。

この変数変換は、 β と γ のようにパラメータのとりうる値の範囲に制約がある場合だけではなく、パラメータの値の絶対値が非常に小さい場合、あるいは非常に大きい場合にも必要である。ユニット UOptMultDim に宣言されている極値探索の手続き MinByFR などでは、パラメータの値が絶対値で 0.01~1000 ぐらいの大きさのものが想定されている。この範囲から大きく外れた値のときは、極値探索の精度が criterionX や criterionF で設定したものでうまくコントロールできない可能性がある。従って、例えば、パラメータ u が

$$u \approx 10^{-10} \quad \text{あるいは} \quad u \approx 10^{10}$$

ぐらいのときは、

$$v = 10^{10} \times u \quad \text{あるいは} \quad v = 10^{-10} \times u$$

と変数変換して、 v についての極値探索とする。上の変換により、 v の値は $v \approx 1.0$ となっている。

最初に極値探索を行うときは、極小値を与える変数値の大きさの見当が付かないことがある。このような場合は、取り敢えず変数変換を行わずにそのまま極値探索を行い、極小値を与えるパラメータ値を求めてみる。得られた値が 0.01~1000 の範囲から著しく逸脱していたときは、得られた値に合わせた変数変換を行い、極値探索をやり直せばよい。

参 考 文 献

- (1) S. S. Rao. *Optimization: Theory and Applications (Second Edition)*. Pp. 747, John Wiley & Sons, 1984.
- (2) 茨木俊秀・福島雅夫「最適化の手法」Pp. 218、共立出版株式会社、1993.
- (3) S. S. Stevens. *Psychophysics*. Pp. 329, John Wiley & Sons, 1975.