領海画定等に用いる計算アルゴリズムについて

柴山信行：沿岸域海洋情報管理室（元領海確定調査室）
今井義隆：海図維持管理室（元領海確定調査室）

Calculation Algorithms for Drawing Outer Limits of Territorial Seas

Nobuyuki Shibayama : Coastal Information Management Office
(Territorial Sea Baseline Research Office 1997-1998)
Yoshitaka Imai : Chart Maintenance Office (Territorial Sea Baseline Research Office 1995-1998)

1．始めに

平成８年の領海法の改正（新たに「領海及び接続水域に関する法律」となった）で，直線基線の制度が新たに採用された．直線基線は低潮線等に加え領海の基線となるもので，両端の位置が定められたその間を最も短く結ぶ線（測地線，大圏線）である．

領海確定調査室（平成10年４月廃止）では，基点から外縁線，中間線等を計算するプログラムに加え，直線基線に基づく外縁線，中間線等を計算するプログラムが必要となった．当初は，直線基線上の点を一定間隔毎に前もって計算しておくことで基点からの計算プログラムを代用していた．直線基線に関してより広範な対応をするためには，直線基線の両端の位置情報から直接取り扱うプログラムが必要であった．

本稿では，プログラム作成に当たり，開発した計算アルゴリズム，検討した計算の最適化条件について報告する．計算アルゴリズムは，いずれも区間縮小法（収束計算法）を用いた数値解析法である．なお，領海画定等に用いる計算は本稿で述べる収束計算以外に，回転楕円体上の２点間の距離を求める計算（以下「距離計算」）が必要であるが，本稿では取扱わない．

2．領海画定等に必要な計算アルゴリズム

領海画定等に必要な主要な計算は，外縁線及び中間線を求めるもの各２通り，合わせて４通りである．いずれも，基点又は直線基線（以下「基点/線」）間の距離の関係を判定するものである．

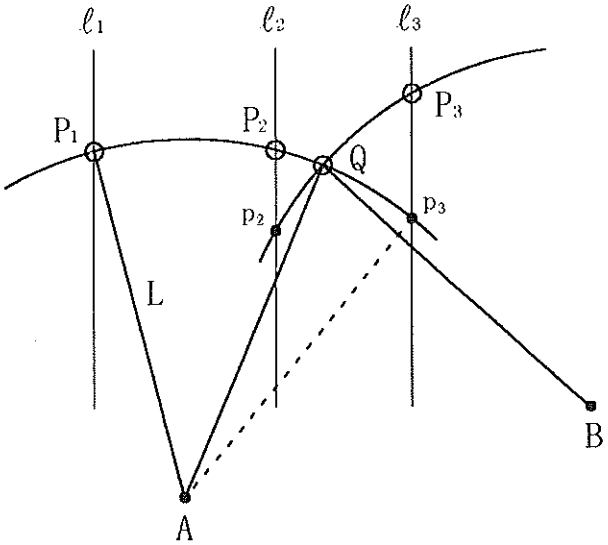
2.1　領海等の外縁線

平面座標で表現すれば(第１図)，基点を中心とする所定の距離の円弧と直線（ℓ_1，ℓ_2，ℓ_3）の交点（外縁点：P_1，P_2，P_3）及び２点からの円弧の交点（二点等距離点：Q)を求めるもので，点列 $P_1P_2QP_3$ が求める外縁線である．

外縁線の計算は，所定の距離（領海では12海里）との比較で行われる．

(1)　外縁点

所定の経度又は緯度（以下「経/緯線」）上の全ての基点/線からの一定距離にある点（ℓ_2上では P_2，



第１図　領海等の外縁線の求め方
Fig. 1　Drawing Outer limit of Territorial Seas.

p_2）で，最も外側にある点（P_2）と定義される．

計算アルゴリズムは，2段階からなる．

① 基点/線と求める経/緯線上の点との距離APと所定の距離Lとの差L−APの正負判定し，差を0（ゼロ）とする点P_1を求める．

② 各基点/線から求めた点P_1の内，最も外側の点Pを選ぶ．

点Pが求める経/緯線上の外縁点であり，その点に対応する基点/線が外縁線の基点/線となる．

(2) 二点等距離点

外縁点計算で求めた連続する2個の基点/線から所定の距離にある外側の点と定義される．

計算アルゴリズムは，3通りが考えられる．

① 外側の範囲の面上の点Pと2個の基点/線との距離（AP，BP）と所定の距離Lとの差の和 $|L-AP|+|L-BP|$の大小判定し，最小となる点Qを求める．

② 外縁線を求める手法で一方の点Aから所定の距離Lにある経/緯線上の点Pを求める．次いで点Pと他方の点Bとの距離と所定の距離との差L−BPの正負判定し，経/緯線を変化させ差を0とする点Qを求める．

③ 経/緯線上の点Pと2個の基点/線からの各距離の差AP−BPの正負判定し，差を0とする点Pを求める．次いでその距離と所定の距離と差L-APの正負判定し，経/緯線を変化させ差を0とする点Qを求める．

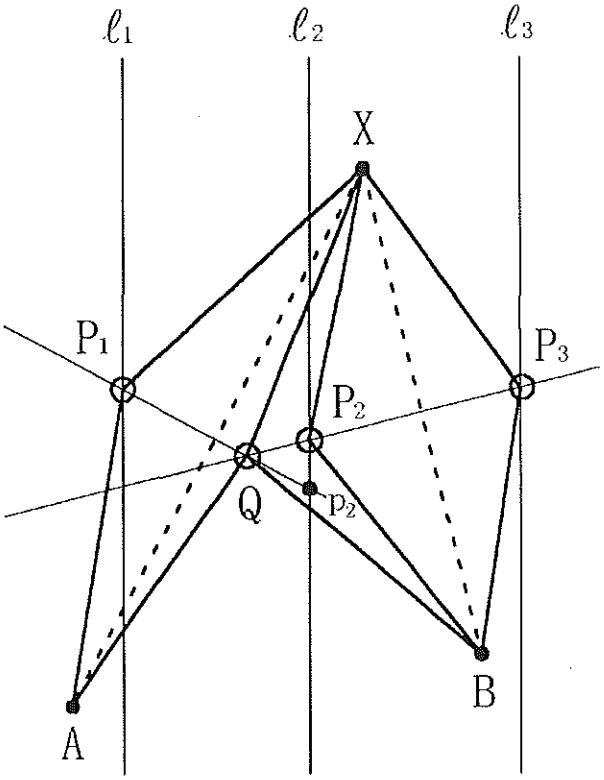
2.2 二国間の中間線

中間線計算は，隣接国との距離が所定の距離（領海では24海里）以下の場合に必要となる．中間線計算には，所定距離が存在しない．

平面座標で表現すれば(第2図)，自国と隣接国間の各基点を結ぶ線（AX，BX）の垂直二等分線と線（ℓ_1，ℓ_2，ℓ_3）の交点（中間点：P_1，P_2，P_3）及び2本の垂直二等分線の交点(三点等距離点：Q)を求めるもので，$P_1QP_2P_3$が求める中間線である．

(1) 中間点

自国の全ての基点/線と隣接国の全ての基点/線から等距離にある所定の経/緯線上の点で，その距離が最も短い点と定義される．



第2図　領海等の中間線の求め方
Fig. 2　Drawing Median line.

計算アルゴリズムは，2段階からなる．

① 自国と隣接国の各基点/線と所定の経/緯線上の点との距離の差AP_1-XP_1の正負判定し，差を0とする点P_1を求める．

② 求めた点P_1の内，最も距離が短い点Pを選ぶ．点Pが求める経/緯線上の中間点であり，その点に対応する各々の基点/線が両国の中間線の基点/線となる．

(2) 三点等距離点

自国の連続する2基点/線と隣接国の対応する1基点/線，又は隣接国の連続する2基点/線と自国の対応する1基点/線から等距離の点と定義される．

計算アルゴリズムは，2通りが考えられる．

① 面上の点と3個の基点/線との各距離の2組の距離の差の和 $|AP-BP|+|BP-XP|+|XP-AP|$（または，各距離の和AP+BP+XP)の大小判定し，最小となる点Qを求める．

② 経/緯線上の点と2基点/線からの両距離の差AP−BPの正負判定し，差を0とする点Pを求め

る．次いで点Pと残りの基点/線との距離との差AP－XPの正負判定し，差を0となるよう経/緯線を変化させ点Qを求める．

3．直線基線と基線外の点との距離を求めるアルゴリズム

直線基線と基線外の点の距離（以下「直線基線との距離」）とは，基線上の点と基線外の点を結ぶ最短の線の長さである．基線上の点とは，直線基線両端との距離の和と基線の長さが等しくなる点である．

平面座標で表現すれば(第3図)，直線ABに点Cから下ろした垂線CPの長さである．

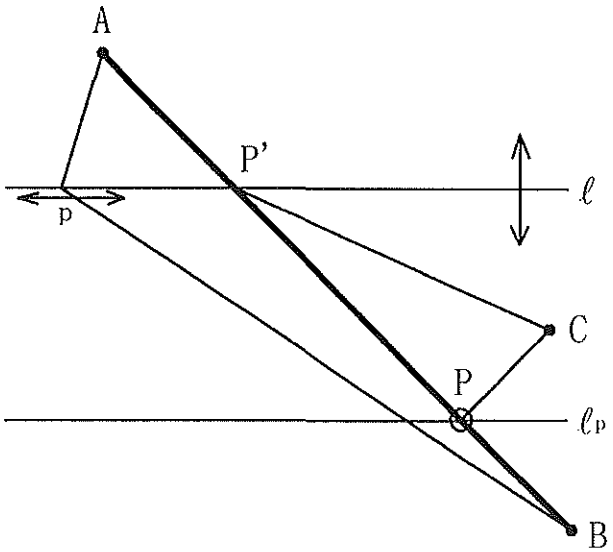
計算アルゴリズムは2段階に分けられる．

(1) 経/緯線と直線基線の交点を求めるアルゴリズム

経/緯線上（ℓ）の点（p）と直線基線の両端点との距離の和　Ap＋Bpの大小判定し，最小となる点P'を求める．距離計算は近似計算であるため直線基線の長さと直線基線の両端点との距離の和が等しくなる点を求めることは実際にはできない．

(2) 直線基線上の点で直線基線外の点との距離が最小になる点を求めるアルゴリズム

経/緯線と直線基線の交点P'と直線基線外の点Cとの距離　CP'の大小判定し，最小となるよう経/緯線を変化させ点Pを求める．最小距離CPが直線基線との距離である．



第3図　直線基線と基線外の点との距離の求め方
Fig. 3　Calculation Method of Distance between Straight Baseline and its Outer Point.

4．区間縮小法（収束計算法）

回転楕円体上の2点間の距離関数は近似計算にあっても複雑であり，求める条件に合致する解を解析的に解くことは困難であるので，数値解析法で解を求める必要がある．数値解析法として，関数に代入する変数の範囲を繰返し狭めて解を求める区間縮小法（収束計算法）を用いる．以下，正負判定，大小判定する値を得るために行う距離計算が組み合わされた計算を「組合せ計算」，得られる値を「評価値」と呼ぶことにする．

2．及び3．で示した領海画定等に用いられる計算アルゴリズムは，以下の3種類の点を求めるプログラムに集約される．

(1) 正負になる評価値を0にする線上の点(外縁線，中間線，他)

(2) 常に正である評価値を最小にする線上の点（経緯線と直線基線の交点，直線基線上の点で直線基線外の点との距離を最小にする点）

(3) 常に正である評価値を最小にする面上の点（二点等距離点，三点等距離点）

以上の3種類の点を求めるプログラムとして，線上（1次元）の点を求めるもの5通り，面上（2次元）の点を求めるもの2通りを掲げる．ここに掲げる7通りの収束計算法で一義的な解が得られるためには，評価値は計算区間内では単調増加（減少）又は極値が一つしかない単調な凹変化である必要があり，計算区間の初期値は適切に設定されなければならない．判定計算としては4.1(1)多分割順次比較法のみが正負判定で，他の方法はいずれも大小判定である．

各方法の名称は，二分法以外は本稿に限り使用されているもので，一般的名称ではない（英文名称も同様）．

4.1　線上の点

(1) 多分割順次比較法

初期収束範囲の両端で評価値の正負が異なり，収束範囲内で変化が単調であれば，収束範囲区間をN等分し，各区分点で組合せ計算を行い，正負が変わ

る点の前後を新たな収束範囲にする方法を用いることができる.

あらかじめ両端の正負が知られている計算の途中においては, $\frac{1}{2N}(N^2+N-2)$ 回*1の組合せ計算で, 収束範囲が$\frac{1}{N}$に縮小される.

(2) 単純多分割法

距離の最小値を求めるような場合は, 評価値は常に正であり正負判定を用いることはできない.

収束範囲区間をN等分し, 両端を除く各区分点で組合せ計算を行い, 最小値の点iの前後i−1からi+1を次の収束範囲にする方法で, N−1回の組合せ計算で, 収束範囲が$\frac{2}{N}$に縮小される.

第4図に示すようにiで最小値となった場合でも, 極値の存在するのは, i−1とi間, 又はiとi+1間の場合があるので, 収束範囲はi−1からi+1となる.

(3) 多分割順次比較法

単純多分割法では, 各区分点について順次組合せ計算し, 最小点を過ぎた後でも計算を継続しており無駄がある. 組合せ計算毎に前点の評価値と比較し増加（又は同値）に転じた点をもって1段階の収束計算を終了させるのが効率的である.

$\frac{1}{2(N-1)}(N^2+N-4)$ 回*2の組合せ計算で収束範囲は$\frac{2}{N}$に縮小される.

(4) 偶数分割法

単純多分割法で, 区分数が偶数の場合は, 繰返される収束計算で前の段階で行った評価値を中央の計算の代わりに使用することが可能であるので, 組合せ計算の回数を1回減らせる.

繰返し計算の途中においては, N−2回の組合せ計算で, 収束範囲が$\frac{2}{N}$に縮小される.

(5) 偶数分割順次比較法

多分割順次比較法も, (4)と同様に, 増加に転じるのが中心点より先まで進んだ場合には組合せ計算を1回省略することが可能である. 繰返し計算の途中においては, $\frac{1}{2(N-1)}(N^2-6)$ 回*3の組合せ計算で, 収束範囲が$\frac{2}{N}$に縮小される.

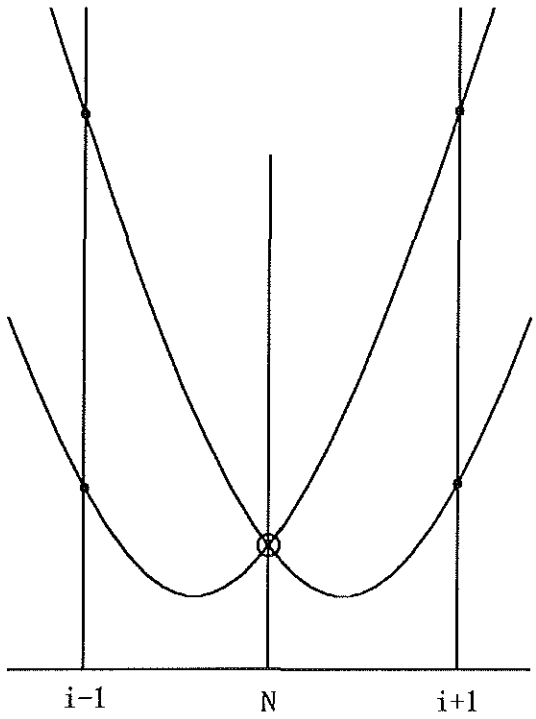
4.2 面上の点

(1) 二次元多分割法

単純多分割法と同様に2次元においても各次元をN等分し, 周辺を除く格子点で関数計算を行い, 最小値の点を中心に新たな収束範囲とする方法で $(N-1)^2$回の組合せ計算で, 各次元とも収束範囲は$\frac{2}{N}$に縮小される.

(2) 二次元偶数分割法

(4)と同様に, 中央の計算を省略が可能なので, N（N−2）回の組合せ計算で, 収束範囲が$\frac{2}{N}$に縮小される.



第4図　凹関数の最低値と極値
Fig. 4　Minimum Point and Smallest Point on Concave Curve.

5. 最適な区間縮小法（収束計算法）と区分数

組合せ計算が収束計算に必要な判定処理, 繰り返し制御に要する時間より十分に長い場合には, 計算に要する時間は組合せ関数計算の回数にのみ依存するとみなせる. 回転楕円体上の距離計算は, この条件に当てはまる.

目的とする精度Cを得るためには, 収束計算を繰返す（M回）必要がある. 縮小率Y（$\frac{1}{N}$又は$\frac{2}{N}$）との関係は,

$Y^M < C$

$M \times \mathrm{Log}(Y) < \mathrm{Log}(C) = c$

$M > \dfrac{1}{\mathrm{Log}(Y)} c$　　Log(Y) は 負

となる．この繰返し回数Mに各計算法毎の組合せ計算回数をかけることで収束計算に必要な組合せ計算の総回数を求めることができる．区分数Nが2から10の場合について各計算法毎の組み合わせ計算総回数の比較値を第1表に示す．表の数値は常用対数を用い，精度に関する共通な係数－cで割ってある．小さな数値ほど効率的であることを意味する．

比較し易くするために，各計算法で最も効率的な区分数に対して，初期収束範囲を緯度/経度で1度とし，100分の1秒の精度（36万分の1）が得られるのに必要な組合せ計算の回数を第2表に示す．表から，

(1) 線上の点を求める場合で，評価値が正負判定可能な場合は区分数2の順次比較法

(2) 線上の点を求める場合で，評価値が大小判定の場合は区分数4の偶数分割順次比較法

(3) 面上の点を求める場合は，区分数3の二次元多分割法

を用いるのが，効率的であることが知れる．

評価値が正負判定可能な場合の区分数2の順次比較法は，通常「二分法」と呼ばれる汎用な方法である．以下，その名に倣い区分数4の偶数分割順次比較法を「四分法」，区分数3の二次元多分割法を「三分法」と呼ぶことにする．

6．効率的な計算アルゴリズム

面上の点を求めるアルゴリズムには三分法以外に，二分法と二分法の組合せ法（以下「組合せ二分法」）が考えられる．異なるアルゴリズムでは組合わせ計算が異なることから，アルゴリズムの計算効率の比較は実際に行われる距離計算の回数で行わなければならない．

直線基線との距離計算は，二重に四分法を適用する必要があり，要求する計算条件を前と同じ（36万分の1）とすると，2211回（(33×2＋1)×33）の距離計算が必要である．したがって，組合せ計算に直線基線との距離計算が含まれる場合には，計算効率の比較のためには直線基線との距離計算回数のみを考慮すれば良い．

第3表に各アルゴリズム別に必要な距離計算回数，直線基線との距離計算回数を示す．表中の①②③は，二点等距離点，三点等距離点計算の各アルゴリズムの番号である．直線基線が1個組み合わされている計算の場合を除き，三分法が組合せ二分法よりも効率的である．

第1表　分割数による収束計算の効率

Table 1　Algorithms Efficiency with Divided Numbers.

線/面	線					面	
判定	正負判定	大小判定				大小判定	
分割数	多分割順次比較法	単純多分割法	多分割順次比較法	偶数多分割法	偶数順次比較法	二次元多分割法	二次元偶数比較法
2	3.322						
3	3.493	11.358	11.358			22.72	
4	3.737	9.966	8.858	6.644	5.537	29.90	26.58
5	4.006	10.052	8.167			40.21	
6	4.284	10.480	7.964	8.384	6.288	52.41	50.30
7	4.564	11.028	7.965			66.18	
8	4.844	11.627	8.068	9.966	6.881	81.39	79.72
9	5.123	12.247	8.229			97.98	
10	5.400	12.876	8.425	11.445	7.471	115.88	114.45

第2表　最適区分数収束計算の効率比較

Table 2　Efficiency with Algorithms using Most Effective Divided Number.

収束計算法	最適区分数	組合せ計算回数
多分割順次比較法(正負判定)	2	21　(19＋2)
単純多分割法	4	51　(19×3)
偶数分割法	4	39　(1×3+18×2)
多分割順次比較法(大小判定)	6	46　(12×3.8)
偶数分割順次比較法	4	33　(1×2.67+18×1.67)
二次元多分割法	3	128　(32×4)
二次元偶数分割法	4	286　(1×16+18×15)

第3表　面上の点を求めるアルゴリズムの計算効率比較

Table 3　Efficiency with Algorithms for Point Determined in Plane.

基点のみの場合の距離基点の回数		
	三分法①	組合せ二分法
二点等距離点	256(128×2)	②399(21×19) ③798(21×2×19)
三点等距離点	384(128×3)	②817((21×2+1)×19)
直線基線を含む場合の直線基線との距離を求める計算の回数		
	三分法①	組合せ二分法②
二点等距離点		
1直線基線, 1点	128	19
2直線基線	256(128×2)	380(20×19)
三点等距離点		
1直線基線, 2点	128	19
2直線基線, 1点	256(128×2)	399(21×19)
3直線基線	384(128×3)	817((21×2+1)×19)

7．プログラミング

二国間協議等の場において即座に対応できるようにするため，PC上ので動作するWindowsプログラムを開発することとし，言語にはDelphi3（Inprise社製Object Pascal）を使用した．

二分法，三分法，四分法の各プログラムの中心部分をプログラム1～3に掲げた．function FNは組合せ計算である．また，大文字の変数は掲載プログラム内では定義されていない．

実際には，収束条件としては収束計算回数ではなく，収束範囲の幅を判定に使用した．今回開発したプログラムで最も時間を要する3組の直線基線からの三点等距離点計算でも，Pentium（133MHz）搭載PCで5分以内に計算可能となっている．なお，距離計算には従来からのデッカの式を用いており，プログラム4に掲げた．

8．最後に

本稿は，日韓漁業協定（平成11年1月），日中漁業協定（平成12年6月）の発効を機に纏めたのである．日中，日韓の漁業交渉過程では様々な条件の境界線を作成する必要があった．本稿で紹介したアルゴリズムを反映させたプログラムを使用して，それら多くは作成された．

本稿作成中に新たな計算速度効率化のアイデアが生まれた．現在使用しているプログラムには，その成果が反映されている．漁業交渉過程で実際に使用していたプログラムの計算速度は，現在のものよりかなり遅いものであった．

また，直線基線が1個含まれる計算においては効率的である組合せ二分法は，三分法同様矩形の収束範囲を指定する必要がある．組合せ二分法の場合の矩形の指定は2段階の大小判定条件の両者とも満足するものでなくてはならないため，三分法に比べ容易ではなく，作成されたプログラムは業務に適用できなかった．

極小値の近傍で関数が左右対称となれば縮小率を$\frac{1}{N}$とすることが可能で，計算速度が更に上がることが期待される．

収束計算等の数値計算法について，浅学の著者にとってはオリジナルであるが，恐らくは既知なものと想像する．先学の方からのご教示をお待ちする．なお，基点からの外縁点，中間点を求めるアルゴリズムは，従前から使用されている既知のものである．

参　考　文　献

新野清志・船田哲男：だれでもわかる数値解析入門，近代科学社，34-40，(1995)

辰野忠夫：中間線・限界線の計算方法，水路部技報，11，9-18，(1992)

辰野忠夫：相異なる測地系間の中間線とジオイド，水路部技報，12，19-25，(1994)

プログラム1　二分法
Program 1　Bisection Method

```
function BiSectionMethod
            (X1, X2 : Extended) : Extended ;
var
  x, v, v2 : Extended ;
  k : byte ;
begin
  v  : =FN(X1) ;
  v2 : =FN(X2) ;
  if v1* v2>0 then begin
    Result : =IJYOTI ;
    Exit ;
  end ;
  if v>0 then H : =true
         else H : =false ;
  k : =0 ;
  repeat
    x : =(X1+X2)/2 ;
    v : =FN(x) ;
    if v>0 then if H then X1 : =x
                     else X2 : =x
           else if H then X2 : =x
                     else X1 : =x ;
    inc(k) ;
  until k>KAISUU ;
  Result : =x ;
end ;
```

プログラム2　三分法
Program 2　Trisection Method

```
function TriSectionMethod
            (X1, Y1, X2, Y2 : Extended) : TPoint ;
var
    x, y, SpanX, SpanY, V, MinV : Extended ;
    i, j, ii, jj, k : byte ;
begin
    SpanX : =(X1-X2)/3 ;
    SpanY : =(Y1-Y2)/3 ;
    k : =0 ;
repeat
      for i : =1 to 2 do
          for j : =1 to 2 do begin
       x : =X1+SpanX * i ;
       y : =Y1+SpanY * j ;
       V : =FN(x, y) ;
       if ((i=1) and (j=1))
                  or (MinV>V) then begin
        MinV : =v ;
        ii : =i ;
        jj : =j ;
       end ;
    end ;
    X1 : =X1+SpanX *(ii-1) ;
    Y1 : =Y1+SpanY *(jj-1) ;
    SpanX : =SpanX *2/3 ;
    SpanY : =SpanY *2/3 ;
    inc(k) ;
  until k>KAISUU ;
  Result.X : =Trunc(x * BAISUU+0.5) ;
  Result.Y : =Trunc(y * BAISUU+0.5) ;
end ;
```

プログラム3　四分法
Program 3　Tetrasection Method

```
function TetraSectionMethod
            (X1, X2 : Extended) : Extended ;
var
  x, Span, MinV : Extended ;
  v : array [1..3] : Extended ;
  i, ii, k : byte ;
  H : boolean ;
begin
  Span : =(X1-X2)/4 ;
  k : =0 ;
  repeat
    H : =false ;
    for i : =1 to 3 do begin
      ii : =i ;
      if (i<>2) or (k=0) then begin
        x : =X1+i * Span ;
        v[i] : =FN(x) ;
      end ;
      if i=1 then MinV : =v[1]
      else if MinV<=v[i] then begin
        H : =true ;
        break ;
      end else MinV : =v[i] ;
  end ;
  case ii of
     3 : if H then X1 : =X1+Span
              else begin
                     X1 : =X1+2* Span ;
                     v[2] : =v[3] ;
              end ;
    else v[2] : =v[1] ;
    end ;
    Span : =Span/2 ;
    inc(i) ;
  until k>KAISUU ;
  Result : =x ;
end ;
```

プログラム4　デッカ距離計算法
Program 4　Decca Distance Caluculation

```
function Decca
          (x1, y1, x2, y2 : Extended) : Extended ;
const
  RADD=3.14159265358979323846/180 ;
  A     =6377397.155 ;
  f     =1/299.152813 ;
var
  rx1, ry1, rx2, ry2 : Extended ;
  uy1, uy2 : Extended ;
  dx, dy, dz : Extended ;
  ss, s, r, p, q : Extended ;
  EE, SQEE : Extended ;
begin
  EE    : =f *(2-f) ;
  SQEE  : =Sqrt(1-EE) ;
  rx1   : =x1* RADD ;
  ry1   : =y1* RADD ;
  rx2   : =x2* RADD ;
  ry2   : =y2* RADD ;
  uy1   : =ArcTan(SQEE * Sin(ry1)/Cos(ry1)) ;
  uy2   : =ArcTan(SQEE * Sin(ry2)/Cos(ry2)) ;
  dx    : =Cos(uy2) * Cos(rx2)
                        -Cos(uy1) * Cos(rx1) ;
  dy    : =Cos(uy2) * Sin(rx2)
                        -Cos(uy1) * Sin(rx1) ;
  dz    : =SQEE *(Sin(uy2)-Sin(uy1)) ;
  ss    : =dx * dx+dy * dy+dz * dz ;
  s     : =Sqrt(ss) * A ;
  q     : =Sin((ry1+ry2)/2) ;
  r     : =A * SQEE/(1-EE * q * q) ;
  p     : =s/2/r ;
  Result : =2* r * ArcTan(p/Sqrt(1-p * p)) ;
end ;
```

＊1　多分割順次比較法

範囲（i−1〜i）になる確率$\frac{1}{N}$

計算を終了するまでに行う区分点での計算の回数は，N−1〜N以外は，i回，N−1〜Nは　N−1回であるので，平均回数は

$$\frac{1}{N}(1+2+\cdots+(N-1)+(N-1))$$

$$=\frac{1}{N}(1+2+\cdots+(N-1)+N-1)$$

$$=\frac{1}{N}\left(\frac{N(N+1)}{2}-1\right)$$

$$=\frac{1}{2N}(N^2+N-2)$$

$$=\frac{1}{2N}(N-1)(N+2)$$

＊2　多分割順次比較法

範囲（i−1〜i+1）になる確率$\frac{1}{N-1}$

計算を終了するまでに行う区分点での計算の回数は，N−2〜N以外は，i+1回，N−2〜NはN−1回であるので，平均回数は

$$\frac{1}{N-1}(2+\cdots+(N-2)+(N-1)+(N-1))$$

$$=\frac{1}{N-1}(1+2+\cdots+(N-2)+(N-1)+N-2)$$

$$=\frac{1}{N-1}\left(\frac{N(N+1)}{2}-2\right)$$

$$=\frac{1}{2(N-1)}(N^2+N-4)$$

＊3　偶数分割順次比較法

範囲（i−1〜i+1）になる確率$\frac{1}{N-1}$

計算を終了するまでに行う区分点での計算の回数は，$\frac{N}{2}$未満の点は多分割順次比較法に同じであり，それ以上では1回少ないので，平均回数は

$$\frac{1}{N-1}\left(2+\cdots+\left(\frac{N}{2}-1\right)+\left(\frac{N}{2}-1\right)\right.$$
$$\left.+\cdots+(N-3)+(N-2)+(N-2)\right)$$

$$=\frac{1}{N-1}(1+2+\cdots+(N-2)+(N-1)$$
$$+\left(\frac{N}{2}-1\right)-2)$$

$$=\frac{1}{N-1}\left(\frac{N(N-1)}{2}+\left(\frac{N}{2}-1\right)-2\right)$$

$$=\frac{1}{2(N-1)}(N^2-6)$$