3C−7 方程式の全根を求める方法

井阪 秀高 (神戸日本電気ソフトウェア㈱)

1．はじめに

与えられた有限区間における方程式ｆ（ｘ）＝０の全ての根を求める一手法を提案する。

従来の方法は、区間内の根を分離してから解く必要があった。即ち、１区間に１根しか含まないような小部分区間を抽出し、後は区間内の根を求める方法を利用するという手法である。

今回提案する方法は、区間両端から１根ずつ探索して求めていく方法で根の分離の過程がない。従って方程式の評価回数は、従来より少ない関数評価回数で全根を求めることができた。

2．基本的考え方

方程式の根の反復列を $X_1, X_2, \cdots X_n, X_{n+1}$ とする。解の近くでは、$X_{n+1}=X_n+Q[f(X_n)]$ と近似することができ、この$Q[f(X_n)]$は、$f(X_n)$ が０に近づけばやはり０に近づくものである。

今、ここに$Q[f(X_n)]$を $R\cdot\sinh^{-1}[f(X_n)]$とする。Ｒは、根への修正量を制御するもので、$f(X_n)$ の符号が変わるたびに小さくなり、符号が一定で収束が遅いときに大きくなるようにする。また、$f(X_n)$ が正のとき X_n が大きくなるほうへ探策する（根の近くで導関数が負になる）ときは正にし、その逆は負にする。以上より、Ｒを次の関数とした。

$R=I\cdot 2^{(p/3-r-1)}$　　(文献(4)参照)

ここで、Ｉは探索方向を制御する変数で１または−１とする。ｒは、$f(X_n)$ の符号が変わるたびに１増加し、ｐは符号が一定のとき１増加する。

この算法の利点は、探索方向を制御することができることで初期値の点の導関数と根の付近の導関数の符号が異なっても解けることである。従って、これと従来の算法とを組み合わせれば大域的収束性をもつアルゴリズムを作ることもできる。

3．アルゴリズム概要

根を求める区間を［Ａ，Ｂ］とする。

①$X_{n+1}=B, X_n=X_{n+1}, f(X_{n+1})=0, r=2, p=0, I1=-1$
$I=-SIGN\{1, f(X_{n+1})\}$とする。

②$I2=SIGN\{1, I\cdot f(X_{n+1})\}$とする(I2:探索方向)。
$I1\cdot I2>0$のとき$p=p+1$, $I1\cdot I2<0$のとき$r=r+1$とする。
$X_{n-1}=X_n$,$f(X_{n-1})=f(X_n)$
$X_n=X_{n+1}$, $f(X_n)=f(X_{n+1})$ とする。
$X_{n+1}=X_n+I\cdot 2^{(p/3-r-1)}\cdot\sinh^{-1}(f(X_n))+I2\cdot\varepsilon'$
(ε':マシン上で$|X_n|$に対して加算できる最小値)
(X_nの更新量は、最大値を設けこれを越えないようにしている)

X_{n+1}がＢより大きくなれば、ｒを大きくし、ｐを０にして再度X_{n+1}を計算しなおす。

X_{n+1}がＡより小さくなれば、X_{n+1}をＡにする。

$|f(X_{n-1})| \gg |f(X_n)| > |f(X_{n+1})|$で$|f(X_{n+1})| < 0.1$で、夫々符号が等しいとき。

$X_n \sim X_{n+1}$ を初期値にセカント法反復を行う。根が見つかれば③へ移り、$|f(X_{n+1})|$が大きくなり、反復に失敗すればここに戻る。

$|f(X_{n-1})| > |f(X_n)| < |f(X_{n+1})|$のとき。

夫々符号が等しいとき。

$X_{n-1} \sim X_{n+1}$ で極値探索を行う。根が見つかれば③へ移り、根がなければ$f(X_{n+1})$, X_{n+1}を極値探索前の値に戻し、ここに戻る。

夫々符号が等しくないとき。

ｐを0.75小さくする。

関数値符号が常に変わらず$X_{n+1} \leqq A$なら根なしとみなす。

X_{n+1}の収束判定をし、根が見つかれば③へ移る。

The Method for Finding All Roots of an Equation
Hidetaka Isaka
NEC Kobe Software Corp.

I1=I2として②を反復する。

③②で求まった根をX、「根の分離幅」をδとする．B=X-δとして④に移る。

④X_{n+1}=A,X_n=X_{n+1},f(X_{n+1})=0,r=2,p=0,I1=1
I=SIGN(1,f(X_{n+1}))とする。

⑤②と同様の計算でX_{n+1}を得る。

X_{n+1}がAより小さくなれば、rを大きくし、pを0にして再度X_{n+1}を計算しなおす。

X_{n+1}がBより大きくなれば、X_{n+1}をBにする。

|f(X_{n-1})|≫|f(X_n)|>|f(X_{n+1})|で|f(X_{n+1})|<0.1で、夫々符号が等しいとき。

X_n〜X_{n+1} を初期値にセカント法反復を行う。根が見つかれば⑥へ移り、|f(X_{n+1})|が大きくなり、反復に失敗すればここに戻る。

|f(X_{n-1})|>|f(X_n)|<|f(X_{n+1})|のとき。

夫々符号が等しいとき。

X_{n-1}〜X_{n+1} で極値探索を行う。根が見つかれば⑥へ移り、根がなければf(X_{n+1}),X_{n+1}を極値探索前の値に戻し、ここに戻る。

夫々符号が等しくないとき。

pを0.75小さくする。

関数値符号が常に変わらずX_{n+1}≧Bなら終了する。

X_{n+1}の収束判定をし、根が見つかれば⑥へ移る。

I1=I2として⑤を反復する。

⑥⑤で求まった根をXとしてA=X+δとして④に移る．

以上

ここで、極値探索の方法は、参考文献(3)にあるBrent の方法を基本とする”ＦＭＩＮ”を参考にした。この計算中で関数値の符号の逆転するところが見つかればそこを挟む２区間に分割し，夫々の区間に対し区間内の根を求める。このときの方法も、同文献のBrent の方法を基本とする”ＺＥＲＯＩＮ”を参考にした。極値探索で符号の逆転がない場合は、重根がある可能性があるので区間が十分小さくなるまで関数値が十分小さいかを調べて根を判定した．

収束判定方法は、ｅを要求精度、$\bar{\varepsilon}$をマシンイプシロン、Δｘを|X_{n+1}-X_n| または探索縮小区間とし；

{ Δｘ<e·max(1,|X_{n+1}|) and
|f(X_{n+1})|<e+64·$\bar{\varepsilon}$·|X_{n+1}| } or
Δｘ≦$\bar{\varepsilon}$·max(1,|X_{n+1}|)

のとき収束したものとする。

なお、極値探索では、

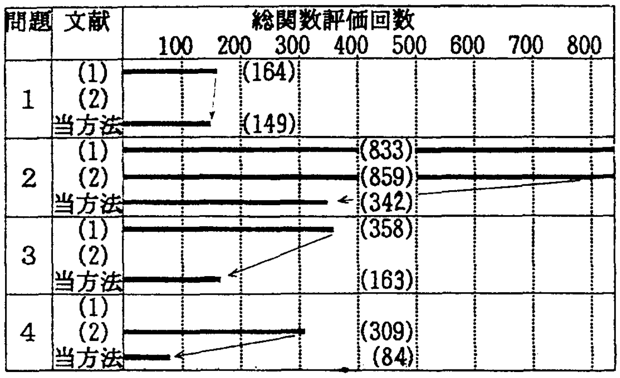
max{2·$\bar{\varepsilon}$·max(1,|X|),e/80}<Δｘ<e·max(1,|X|)
の範囲で　|f(X)|<e+64·$\bar{\varepsilon}$·|X|
のとき収束したものとする。

３．性能評価

参考文献(1),(2) との比較を試みる。当方法のプログラムは、PC9800 BASIC倍精度で計算し、要求精度1.D-10,δ=.001とした。文献(1),(2) は、１区間に１根しか含まない部分区間を分離した後個々の根を求める方法で、文献(1) の部分区間抽出幅εは、以下のいずれの問題も全根解ける最小限の条件であるε=.001 とし、部分区間の根の存在判定用の値K^2はこの条件で全根が求まる最小の値とした。文献(2)でもε=.001とした。いずれも全根が確実に求まっている（当方法では、δを0.1 のような大きな値としても,より少ない回数で全根が求まっている）。

問題：

1) f(x)=exp(.01x)+3-(x-231)(x-597) [-200,800]
2) f(x)=sin(πx/14)+sin(3πx/2) [0, 9]
3) f(x)=|exp(.01x)+3-(x-231)(x-597)| [0,1000]
4) f(x)=1000(x-373.2)(x-373.3) [-1000,1000]



最後に本発表を支援して下さったNEC情報処理システム技術本部の花村、萬、車氏に感謝いたします。

４．参考文献

(1)坂田,二宮,「方程式の根の分離プログラム」，情報処理学会全国大会講演論文集,Vol.32,pp.1709-1710,(1986)

(2)坂田,二宮,「方程式の根を求めるプログラム」，情報処理学会全国大会講演論文集,Vol.33,pp.1847-1848,(1986)

(3)G.E.Forsythe,M.A.Malcolm,C.B.Moler,森 訳,「計算機のための数値計算法」,日本コンピュータ協会,(1978)

(4)N.Kantaris,P.F.Howden,三井 訳,「方程式の数値解法」，啓学出版，(1987)