平成16年度卒業論文

Affine Arithmeticを利用した解の非存在領域の除去性能の評価

Evaluation of Removal Performance of Non-existence Region of Solution using Affine Arithmetic

平成17年2月2日

指導教授　柏木雅英　助教授

早稲田大学理工学部情報学科

1G01P098-7　水口義雄

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景

曲線や曲面等の方程式の解を求める方法の一つに、解の存在しない領域を判定し、その領域を除去していくというものがある。この時、区間演算を利用するのだが、区間演算には、得られる区間幅が極端に広くなってしまう場合が多いという欠点が存在する為、その判定性能は必ずしも良いとは言えなかった。Stolfi の提案した Affine Arithmetic は、区間演算の一手法で、与えられた区間を Affine 形式といわれる形式に変換することで、変数間の相関を考慮した計算を行うことが出来る。そのため、従来の様な区間幅の増大を抑えることが出来、その結果、解の非存在領域の除去性能を高めることが可能である。しかしこの Stolfi の方法も完璧とは言い難い。何故なら Affine 形式で得られた数式を、通常の区間表示に還元した後に、解の非存在を判定してしまうからである。つまり、数式間の持つ相関性を全く考慮せずに判定を行ってしまうからであり、これでは Affine Arithmetic が持ち得る筈の本来の性能を、十分には発揮出来ないということになる。従来の Stolfi の方法に加えて、最後の判定の前に数式間の相関性を考慮するための、更なる工程を付け加えることによって、Affine Arithmetic はその性能を真に発揮し得ることが可能になる筈である。

1.2　本論文の目的

解の非存在領域の判定の効率化をねらいとして、従来の Affine Arithmetic に新しい工程を加えることによって性能を向上できるかどうかを検証するために、具体的な例題に用いることで、非存在領域の除去性能を比較検証することが本論文の目的である。

1.3　本論文の構成

本論文の前半においては本論文のテーマである Affine Arithmetic を利用した非存在領域の除去を行うのに必要な事項について述べる。

まず、2 章において精度保証付き数値計算の基礎となる計算機による数値計算での数の扱いとその誤差の問題、区間演算による計算結果の精度保証の考え方について述べる。

3 章では区間演算のひとつである Affine Arithmetic について、その原

理と特徴を述べ、簡単な例題により通常の区間演算と Affine Arithmetic の比較を行う。

4 章においては Affine Arithmetic を利用した解の非存在領域を除去するアルゴリズムについて、従来の Stolfi の方法及び、新手法を加えた方法の原理を述べる。

続く 5 章において、具体的に非線形方程式を用いて、Affine Arithmetic を利用した非存在領域の除去性能の有効性を検証する。

第2章　区間解析

2.1　はじめに

本章では区間演算の考え方について述べる．まず計算機における数の表現について簡単に述べ, 計算機を用いて連続数学の問題を解く場合の数の扱いと誤差の問題, 数の拡張として考えられた区間演算により解を包み込むことで精度を保証しようという考え方について述べる.

2.2　計算機上の数

計算機では実数を浮動小数点数で近似して計算する．浮動小数点数システムについては IEEE 標準 754 が広く用いられているので, まずこの IEEE754 による 2 進浮動小数点数システムについて簡単に述べる．ここでは倍精度浮動小数点数のみを扱う.

2.2.1　データ形式

倍精度浮動小数点数は 64bit からなり,

s(符号部,1bit)	e(指数部,11bit)	m(仮数部,52bit)

のように表される.

s は符号を表し, 0 なら正, 1 なら負である．指数部 e は符号なし整数とみて $0 \leq e \leq 2047$ の値をとるが, $e = 0$ と $e = 2047$ を特別な場合の表現に使用することにし, $1 \leq e \leq 2046$ のとき（正規化数という）, 上の浮動小数点数は,

$$x = \pm 1.m \times 2^{e-1023} \tag{2.1}$$

という実数を表現している．正規化数の場合先頭は必ず 1 なのでそれはメモリに格納しない．これによって, 52bit で 53bit の精度をもつことができる．倍精度浮動小数点数は正規化されている場合 10 進でおよそ 15 桁の精度をもっている．正規化数以外に,

- 0　$(e = m = 0)$
- $\pm\infty$　$(e = 2047, m = 0)$
- NaN　$(e = 2047, m \neq 0)$

のような特別な数がある．$\pm\infty$ はオーバーフローやゼロでの割り算の結果など, NaN は $\sqrt{-5}$ や ∞/∞ のような不当な演算の結果などで得られる．

また, 正規化数よりも絶対値が小さい数は仮数部の先頭を 1 とせず, 非正規化数として表現される．このときは $e = 0$ とし,

$$x = \pm 0.m \times 2^{-1022} \tag{2.2}$$

という実数に対応する．この場合, 精度は 53bit より少なくなっている．

2.2.2 丸めの切り替え

倍精度浮動小数点数システムで表現できる数は 2^{64} 個以下であり有限個である．したがって, 任意の実数を表現することはできない．また, 浮動小数点数は四則演算について閉じていない．たとえば, ふたつの浮動小数点数の和が浮動小数点数で表せるとは限らない．これらのような場合にはその結果を本来の結果に近い浮動小数点数で近似する．この操作を丸めといい, この近似によって生じる誤差を丸め誤差という．いま, $c \in \mathbf{R}$ を丸めるとすると, その方法には

上方向への丸め ($\triangle$) c 以上の最小の浮動小数点数で近似する

下方向への丸め ($\bigtriangledown$) c 以下の最大の浮動小数点数で近似する

最近点への丸め ($\diamondsuit$) c に最も近い浮動小数点数で近似する．このような数が 2 つあるときは, 仮数部の最後のビットが偶数であるような浮動小数点数に近似する（偶数丸め方式）

0 方向への丸め 絶対値 $|c|$ 以下で c に最も近い浮動小数点数で近似する

の種類がある．IEEE754 にしたがった CPU ではこの丸めの方法を指定することができる．精度保証付き数値計算においてはこのうち上方向への丸め, 下方向への丸め, 最近点への丸めの利用が重要となっているが, IEEE754 では以下の性質を満たすことが要請されている．$\bigcirc \in \{\triangle, \bigtriangledown, \diamondsuit\}$, 任意の $x, y \in \mathbf{R}$ に対し,

$$\bigcirc x = x \tag{2.3}$$

$$x \le y \Rightarrow \bigcirc x \le \bigcirc y \tag{2.4}$$

$$\bigcirc(-x) = -(\bigcirc x) \tag{2.5}$$

さらに,

$$\cdot \in \{+, -, \times, /\}$$

に対し, IEEE754 では浮動小数点数演算 $\bigodot$ を,

$$x \bigodot y = \bigcirc(x \cdot y) \quad (x, y \in \mathbf{R}) \tag{2.6}$$

で定義する．式 (2.6) の右辺は実数での演算による数学的に正しい結果（これは実数）を丸めて得られる浮動小数点数であり, これに一致するように左辺が定義されている.

2.2.3　10 進小数と 2 進小数の変換

多く扱われる 10 進数は計算機においては 2 進数に変換される．このとき注意すべきことは

10 進の有限小数がかならずしも 2 進の有限小数で表せるとは限らない

ことである．10 進の 0.1 を 2 進で表現すると

$$0.000\dot{1}10\dot{0}$$

と循環小数になり, これは有限桁では表せない．10 進の 0.1 は浮動小数点数では正確に表現できず丸めにより近似が行われることになる．また, 逆に浮動小数点数を 10 進で表示する場合（C 言語の printf など）, 2 進小数を 10 進小数に変換するが, ここには丸めによる誤差が入りうる．したがって, 精度保証付きで計算結果を表示するには丸めの方向を制御できる表示関数を自分で作る必要がある.

2.3　区間演算

上に述べたように浮動小数点数は有限個であり, 厳密に実数を扱うことはできない．そこで, 浮動小数点数を両端にもつ区間で解を包み込むということが考えられた．この数の拡張としての区間演算について, ここではひとまず計算機のことを忘れ, 四則演算で閉じた実数体での区間演算を考える.

2.3.1 区間の定義

区間 $[x]$ は,

$$[\underline{x}, \overline{x}] = \{x \in \mathbf{R} | \underline{x} \leq x \leq \overline{x}\} \tag{2.7}$$

であらわされる閉区間とする．ここで $\underline{x} \leq \overline{x} \in \mathbf{R}$ であり, $\underline{x}$ を $[x]$ の下端, $\overline{x}$ を $[x]$ の上端という．$\underline{x} = \overline{x}$ のとき $[x]$ は点区間となり, これは実数となる．また, このような $[x]$ からなる閉区間の集合を $I\mathbf{R}$ とかく．

区間について以下を定義する．

$$\mathrm{mid}([x]) = \frac{\overline{x} + \underline{x}}{2} \tag{2.8}$$

$$\mathrm{rad}([x]) = \frac{\overline{x} - \underline{x}}{2} \tag{2.9}$$

$\mathrm{mid}([x]), \mathrm{rad}([x])$ をそれぞれ区間 $[x]$ の中心, 半径という．

2.3.2 四則演算

数の拡張として区間の四則演算を定義する．以下 $\cdot \in \{+, -, \times, /\}$ とする．区間 $[x] = [\underline{x}, \overline{x}], [y] = [\underline{y}, \overline{y}]$ に対して,

$$[x] \cdot [y] = \{x \cdot y \in \mathbf{R} | x \in [x], y \in [y]\} \tag{2.10}$$

と表される集合により定義する．ただしこのような集合を求めるには無限回の演算を行う必要はなく,

$$[x] + [y] = [\underline{x} + \underline{y}, \overline{x} + \overline{y}] \tag{2.11}$$

$$[x] - [y] = [\underline{x} - \overline{y}, \overline{x} - \underline{y}] \tag{2.12}$$

$$[x] \times [y] = [\min\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}] \tag{2.13}$$

$$[x]/[y] = [x] \times [1/\overline{y}, 1/\underline{y}] \tag{2.14}$$

とすることで有限回の演算で済む．ただし, 除算は $0 \notin [y]$ のときのみ定義される．また, 乗算と除算については表 2.1, 表 2.2 のように上端と下端による場合分けで演算回数を減らすことが可能である．

2.3.3 関数の値域評価

今度は初等関数を区間上の関数に拡張することを考える．そのためには次のように定義すればよい．

$$f([x]) = \{f(x) | x \in [x]\} \tag{2.15}$$

	$\underline{y} > 0$	$0 \in [y]$	$\overline{y} < 0$
$\underline{x} > 0$	$[\underline{x}\underline{y}, \overline{x}\overline{y}]$	$[\overline{x}\underline{y}, \overline{x}\overline{y}]$	$[\overline{x}\underline{y}, \underline{x}\overline{y}]$
$0 \in [x]$	$[\underline{x}\overline{y}, \overline{x}\overline{y}]$	$[\min\{\underline{x}\overline{y}, \overline{x}\underline{y}\}, \max\{\underline{x}\underline{y}, \overline{x}\overline{y}\}]$	$[\overline{x}\underline{y}, \underline{x}\underline{y}]$
$\overline{x} < 0$	$[\underline{x}\overline{y}, \overline{x}\underline{y}]$	$[\underline{x}\overline{y}, \underline{x}\underline{y}]$	$[\overline{x}\overline{y}, \underline{x}\underline{y}]$

表 2.1: 区間の乗算 $[x] \times [y]$

	$\underline{y} > 0$	$\overline{y} < 0$
$\underline{x} > 0$	$[\underline{x}/\overline{y}, \overline{x}/\underline{y}]$	$[\overline{x}/\overline{y}, \underline{x}/\underline{y}]$
$0 \in [x]$	$[\underline{x}/\underline{y}, \overline{x}/\underline{y}]$	$[\overline{x}/\overline{y}, \underline{x}/\overline{y}]$
$\overline{x} < 0$	$[\underline{x}/\underline{y}, \overline{x}/\overline{y}]$	$[\overline{x}/\underline{y}, \underline{x}/\overline{y}]$

表 2.2: 区間の除算 $[x]/[y]$

ただし, 関数 f は $\mathbf{R}$ の部分集合において定義され, その任意の閉区間において連続であるとする. 例えば,

$$\begin{aligned}\sqrt{[x]} &= [\sqrt{\underline{x}}, \sqrt{\overline{x}}] \quad (\underline{x} \geq 0) \\ \exp([x]) &= [\exp(\underline{x}), \exp(\overline{x})]\end{aligned}$$

さて, 次にこれらの初等関数の合成や四則演算からなる関数を区間上に拡張することを考えるのだが, このような関数の評価を厳密に行うのは困難なことがある. そこで, このような関数の演算規則を区間演算でおきかえて得られる関数を区間拡張とよび, $f_{[\,]}([x])$ とかく. このとき,

$$f([x]) \subseteq f_{[\,]}([x]) \tag{2.16}$$

が成り立つ. また, f を同値な式で書きかえたとき f の区間拡張は一般に異なる.

例 $f(x) = x^2 - 2x$ を区間 $[0.9, 1.1]$ で評価することを考える. このとき $f(x)$ を 3 通りに表現してその区間拡張を比べてみる. 真の評価 $f([0.9, 1.1]) = [-1, -0.99]$ である.

$f(x) = x^2 - 2x$

$$\begin{aligned}f_{[\,]}([0.9, 1.1]) &= [0.9, 1.1]^2 - 2 \times [0.9, 1.1] \\ &= [0.81, 1.21] - [1.8, 2.2] \\ &= [-1.39, -0.59]\end{aligned}$$

$f(x) = x(x-2)$

$$\begin{aligned} f_{[\,]}([0.9, 1.1]) &= [0.9, 1.1] \times ([0.9, 1.1] - 2) \\ &= [0.9, 1.1] \times [-1.1, -0.9] \\ &= [-1.21, -0.81] \end{aligned}$$

$f(x) = (x-1)^2 - 1$

$$\begin{aligned} f_{[\,]}([0.9, 1.1]) &= ([0.9, 1.1] - 1)^2 - 1 \\ &= [-0.1, 0.1]^2 - 1 \\ &= [0, 0.01] - 1 \\ &= [-1, -0.99] \end{aligned}$$

(2.16) で等式が成立するとき, 厳密に包み込んでいるという. 上の例では関数 f の表現を変えることで, 得られる区間の幅にして 80 倍もの差が出ている一方, 3 番目の区間拡張では厳密な包み込みができていることがわかる. どのように式を表現したら良い区間拡張が得られるかというのは難しい問題である.

なお, 関数 f の値域 $f([\underline{x}, \overline{x}])$ を $f([\underline{x}, \overline{x}]) \subseteq [\underline{y}, \overline{y}]$ なる区間 $[\underline{y}, \overline{y}]$ で評価するとき, この $[\underline{y}, \overline{y}]$ を $f([\underline{x}, \overline{x}])$ の区間包囲という. 区間拡張は区間包囲のひとつである.

2.4 計算機上の区間演算

今度は先に述べた実数体上の区間演算を有限の浮動小数点数システムに拡張した機械区間演算の実現を考える. 以下浮動小数点数の集合を F とおく. この区間演算によって, 求めたい実数解が幅の狭い区間の中に入ることがいえたならばそれは十分価値のあることといえる.

2.4.1 区間の定義

区間 $[x]$ を,

$$[\underline{x}, \overline{x}] = \{x \in \mathbf{R} | \underline{x} \le x \le \overline{x}\} \tag{2.17}$$

で定義する. ここで $\underline{x} \le \overline{x} \in F$ であり, $\underline{x}$ を $[x]$ の下端, $\overline{x}$ を $[x]$ の上端という. すなわち, 実数上の区間の定義で上端と下端が浮動小数点数であるものとなる. このような $[x]$ からなる閉区間の集合を IF とかく.

丸めのモード

$$\begin{aligned}\bigcirc &: I\mathbf{R} \to IF \\ \bigcirc &\in \{\triangle, \bigtriangledown, \Diamond\}\end{aligned}$$

とする．丸めによって $I\mathbf{R}$ の区間 $[\underline{x}, \overline{x}]$ を IF の区間に移すには,

$$[x] \subseteq \bigcirc[x] \quad ([x] \in I\mathbf{R}) \tag{2.18}$$

$$\bigcirc[x] = [x] \quad ([x] \in IF) \tag{2.19}$$

$$[x] \subseteq [y] \Rightarrow \bigcirc[x] \subseteq \bigcirc[y] \quad ([x], [y] \in I\mathbf{R}) \tag{2.20}$$

$$\bigcirc(-[x]) = -(\bigcirc[x]) \quad ([x] \in I\mathbf{R}) \tag{2.21}$$

をそれぞれ任意の $[x]$,$[y]$ についてみたすようにする．IEEE754 にしたがった浮動小数点数システムであれば,

$$\bigcirc[x] = [\bigtriangledown\underline{x}, \triangle\overline{x}] \tag{2.22}$$

とすれば上の条件は満たされる.

2.4.2 四則演算

$I\mathbf{R}$ での四則演算をもとに, IF 上の四則演算を定義する.

$$\begin{aligned}\cdot &\in \{+, -, \times, /\} \\ \bigcirc &\in \{\triangle, \bigtriangledown, \Diamond\}\end{aligned}$$

に対し, IF の演算を

$$[x] \odot [y] = \bigcirc([x] \cdot [y]) \tag{2.23}$$

で定義する．(2.22) から, 具体的には次のようにすればよい.

$$\begin{aligned}
[x] + [y] &= [\bigtriangledown(\underline{x} + \underline{y}), \triangle(\overline{x} + \overline{y})] \\
[x] - [y] &= [\bigtriangledown(\underline{x} - \overline{y}), \triangle(\overline{x} - \underline{y})] \\
[x] \times [y] &= [\min\{\bigtriangledown(\underline{x}\underline{y}), \bigtriangledown(\underline{x}\overline{y}), \bigtriangledown(\overline{x}\underline{y}), \bigtriangledown(\overline{x}\overline{y})\}, \\
&\quad \max\{\triangle(\underline{x}\underline{y}), \triangle(\underline{x}\overline{y}), \triangle(\overline{x}\underline{y}), \triangle(\overline{x}\overline{y})\}] \\
[x]/[y] &= [\min\{\bigtriangledown(\underline{x}/\underline{y}), \bigtriangledown(\underline{x}/\overline{y}), \bigtriangledown(\overline{x}/\underline{y}), \bigtriangledown(\overline{x}/\overline{y})\}, \\
&\quad \max\{\triangle(\underline{x}/\underline{y}), \triangle(\underline{x}/\overline{y}), \triangle(\overline{x}/\underline{y}), \triangle(\overline{x}/\overline{y})\}]
\end{aligned}$$

また, 乗算と除算の場合には実数の場合と同様に, 表 2.1, 表 2.2 を利用して演算回数を少なくすることができる. そのためには, 下端の計算を下向きで, 上端の計算を上向きで行えばよい.

2.5 おわりに

本章では計算機での数値の取り扱いと区間演算について述べた. 計算機を用いて数値計算を行う場合, 実数を浮動小数点数で近似しており, そこには必ず誤差が生じる. 一方, CPU の丸めの制御と区間演算の手法を利用することで浮動小数点数の四則演算での誤差を把握することが可能である. 区間演算は精度保証付き数値計算における非常に重要な技法のひとつである.

なお, 当たり前のことであるが, 本論文の数値実験における区間演算は機械区間演算のことをさす.

第3章　Affine Arithmetic

3.1　はじめに

Affine Arithmeticは区間演算の一手法で、1994年にStolfiらによって提案された。Affine Arithmeticにおける計算 は変数間の相関を考慮して行われるため、通常の区間演算で起こりやすい区間幅の爆発的な増大を抑える効果がある。本章ではAffine形式と区間表示との変換、Affine形式での演算について述べたあと、簡単な例題によって通常の区間演算とAffine Arithmeticの違いについて述べ、4章のAffine Arithmeticを応用した非存在判定テストのための足がかりとする。

3.2　Affine Arithmeticについて

Affine Arithmeticにおいては、変数xはAffine形式

$$x = x_0 + x_1\varepsilon_1 + \cdots + x_n\varepsilon_n \tag{3.1}$$

で表される。ここにε_iは$-1 \leq \varepsilon_i \leq 1$の値をとることがわかっているダミー変数とする。式(3.1)の右辺をAffine多項式という。

3.2.1　初期化

入力区間$[\underline{I}, \overline{I}]$はつぎのようにAffine形式に初期化する。

$$\frac{\overline{I} + \underline{I}}{2} + \frac{\overline{I} - \underline{I}}{2}\varepsilon \tag{3.2}$$

一般に、入力が区間ベクトル、区間行列などの場合は成分ごとに別のダミー変数を用意することになる。

3.2.2　Affine多項式の区間表示

Affine形式(3.1)を区間表示$[\underline{x}, \overline{x}]$にもどすには、

$$\underline{x} = x_0 - \Delta, \quad \overline{x} = x_0 + \Delta \tag{3.3}$$

とすればよい。ここで、

$$\Delta = \sum_{i=1}^{n} |x_i| \tag{3.4}$$

3.2.3　線形演算

Affine 多項式

$$
\begin{aligned}
x &= x_0 + x_1\varepsilon_1 + \cdots + x_n\varepsilon_n \\
y &= y_0 + y_1\varepsilon_1 + \cdots + y_n\varepsilon_n
\end{aligned}
$$

と実数 α に対して、

$$
\begin{aligned}
x \pm y &= (x_0 \pm y_0) + \sum_{i=1}^{n}(x_i \pm y_i)\varepsilon_i && (3.5)\\
x \pm \alpha &= (x_0 \pm \alpha) + \sum_{i=1}^{n} x_i\varepsilon_i && (3.6)\\
\alpha x &= (\alpha x_0) + \sum_{i=1}^{n}(\alpha x_i)\varepsilon_i && (3.7)
\end{aligned}
$$

で線形演算を定義する。複号同順。

3.2.4　非線形演算

非線形演算の像は一般には (3.1) のような Affine 多項式で表すことはできない。このため、Affine Arithmetic の非線形演算では像を線形な式で近似し、新たなダミー変数を設け、近似との誤差の項を追加して得られる領域をその結果とする。

非線形単項演算

$f(x)$ を非線形単項演算とする。$x \in I$ とすると、I における $f(x)$ の像 $f(I)$ は曲線を描くため Affine 形式では表現できない。Affine Arithmetic では、I における $f(x)$ の近似を

$$ax + b \tag{3.8}$$

とし、$f(x)$ と (3.8) との最大誤差 δ を、

$$\delta = \max_{x\in I} |f(x) - (ax + b)| \tag{3.9}$$

で求め、ダミー変数 ε_{n+1} を追加して、新たな Affine 多項式

$$(ax + b) + \delta\varepsilon_{n+1} \tag{3.10}$$

を得て、これを $f(I)$ の包含とする。Affine Arithmetic による非線形単項演算の評価のようすを 3.3 節の図 3.1 に示す。

非線形二項演算

$f(x,y)$ を非線形二項演算、$(x,y) \in I_x \times I_y = I$ とすると、I での $f(x,y)$ の像 $f(I)$ は曲面を描くため Affine 形式では表現できない。Affine Arithmetic では、I における $f(x,y)$ の近似を

$$ax + by + c \tag{3.11}$$

とし、$f(x,y)$ と (3.11) との最大誤差 δ を、

$$\delta = \max_{(x,y)\in I} |f(x,y) - (ax + by + c)| \tag{3.12}$$

で求め、ダミー変数 ε_{n+1} を追加して、新たな Affine 多項式

$$(ax + by + c) + \delta\varepsilon_{n+1} \tag{3.13}$$

を得て、これを $f(I)$ の包含とする。

非線形演算の包含

このように、Affine Arithmetic では非線形演算が行われるごとにダミー変数の個数が増える。また、Affine Arithmetic での非線形演算は、δ と、それを与える a, b, c を決定することに帰着し、δ を最小にすることにより得られた $f(I)$ の包含が演算の結果として与えられる。

3.3 計算例

ここで例題を取り上げて Affine Arithmetic と区間演算の違いをみることにする。そのまえに非線形単項演算のひとつである x^2 の包含の求め方 (a, b, δ の決定法) を図 3.1 に示す。

例題 3.1 $f(x) = 2x - x$ を区間 $I = [-2, 3]$ で評価する。$2x - x = x$ であるから、当然真の評価は $f(I) = [-2, 3]$ となる。

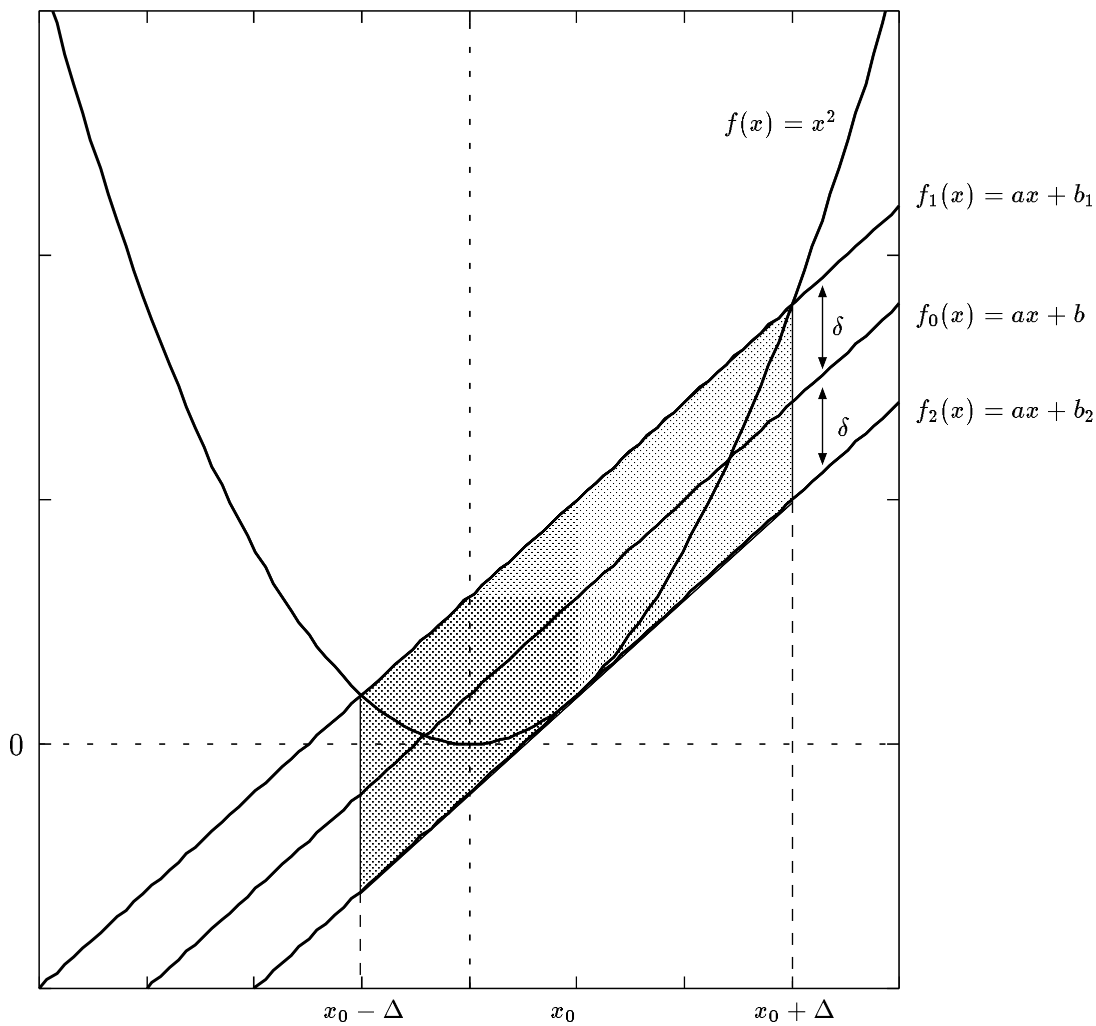

図 3.1: x^2 の包含-式 (3.4) により Δ を求める。このとき、$a = 2x_0, b_1 = -{x_0}^2 + \Delta^2, b_2 = -{x_0}^2$ であり、$b = \frac{1}{2}(b_1 + b_2) = -{x_0}^2 + \frac{1}{2}\Delta^2, \delta = \frac{1}{2}\Delta^2$ が得られる。

区間演算の場合

$$\begin{aligned}2\times[-2,3]-[-2,3] &= [-4,6]-[-2,3]\\ &= [-4-3,6-(-2)]\\ &= [-7,8]\end{aligned}$$

Affine Arithmetic の場合　$[-2,3]=0.5+2.5\varepsilon$ なので、

$$\begin{aligned}2\times(0.5+2.5\varepsilon)-(0.5+2.5\varepsilon) &= (1.0+5.0\varepsilon)-(0.5+2.5\varepsilon)\\ &= (0.5+2.5\varepsilon)\\ &\rightarrow [-2,3]\end{aligned}$$

例題 3.2 $f(x)=x^2-2x$ を区間 $I=[0.9,1.1]$ で評価する。真の評価は $f(I)=[-1,-0.99]$ となる。

区間演算の場合

$$\begin{aligned}[0.9,1.1]^2-2\times[0.9,1.1] &= [0.81,1.21]-[1.8,2.2]\\ &= [0.81-2.2,1.21-1.8]\\ &= [-1.39,-0.59]\end{aligned}$$

平均値形式を用いた場合　$c=\mathrm{mid}(I)=1.0$ となり、

$$\begin{aligned}f(c)+f'(I)(I-c) &= -1+(2\times[0.9,1.1]-2)\times([0.9,1.1]-1.0)\\ &= -1+[-0.2,0.2]\times[-0.1,0.1]\\ &= [-1.02,-0.98]\end{aligned}$$

Affine Arithmetic の場合　$[0.9,1.1]=1.0+0.1\varepsilon_1$、また、$I$ における x^2 の近似 $ax+b$ は $a=2,b=-0.995$ で得られ、このとき $\delta=0.005$ となる。Affine 形式 $x=1.0+0.1\varepsilon_1$ に対して

$$\begin{aligned}x^2-2x &= (2\times(1.0+0.1\varepsilon_1)-0.995+0.005\varepsilon_2)-2\times(1.0+0.1\varepsilon_1)\\ &= -0.995+0.005\varepsilon_2\\ &\rightarrow [-1,-0.99]\end{aligned}$$

例題 3.3 $f(x)=x^2+2x$ を区間 $I=[0,2]$ で評価する。真の評価は $f(I)=[0,8]$ となる。

区間演算の場合

$$\begin{aligned}[0,2]^2 + 2 \times [0,2] &= [0,4] + [0,4] \\ &= [0,8]\end{aligned}$$

Affine Arithmetic の場合　$[0,2] = 1 + \varepsilon_1$、また、$I$ における x^2 の近似 $ax+b$ は $a = 2, b = -0.5$ で得られ、このとき $\delta = 0.5$ となる。Affine 形式 $x = 1 + \varepsilon_1$ に対して

$$\begin{aligned}x^2 + 2x &= (2 \times (1 + \varepsilon_1) - 0.5 + 0.5\varepsilon_2) + 2 \times (1 + \varepsilon_1) \\ &= 3.5 + 4\varepsilon_1 + 0.5\varepsilon_2 \\ &\rightarrow [-1, 8]\end{aligned}$$

例題 3.1 においては $2x$ と x の自明な相関関係を、例題 3.2 においては x^2 と $2x$ という I において傾きの近い関数の相関関係を考慮しているために通常の区間演算より優れた評価が得られている。

例題 3.3 では得られた評価は通常の区間演算より悪いように見える。しかし Affine Arithmetic による評価では変数間の相関関係が考慮されており、得られた直方体領域のすべての値をとるわけではない。すなわち、$I = [0,2]$ において $f(I) = [-1,8]$ という評価が得られているが、ダミー変数 $\varepsilon_1, \varepsilon_2$ の変動により、例えば直方体（ここでは長方形）領域内の $(0,8)$ や $(2,0)$ といった点は Affine Arithmetic による評価で得られた領域内にはない。

これは Affine 多項式を直方体領域しか表現できない区間表示に変換した（例題では $\rightarrow$ で示されている部分）ことが原因であり、Affine Arithmetic では特に必要ない限り計算結果は Affine 形式のままにしておくのがよいとされる（図 3.2)。

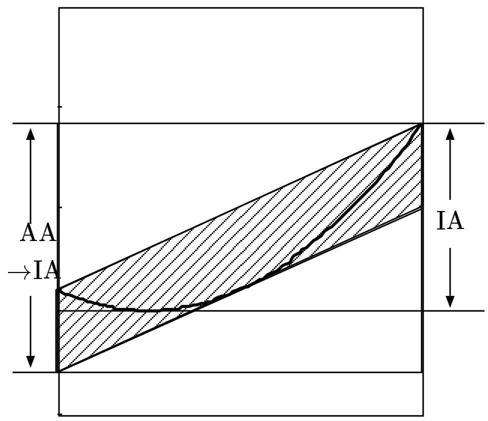

図 3.2: Affine 形式の区間演算への還元

3.4 おわりに

本章では Affine Arithmetic について、その方法と有効性を通常の区間演算との比較から簡単に述べた。何度もいうように Affine Arithmetic では変数間の相関関係を考慮した演算が行われ、非線形演算は線形な近似に誤差項を付加した領域をもってその結果とする。次章ではこのような Affine Arithemtic の特徴を解の非存在領域の判定に利用する方法についてその原理を述べる。

第4章　Affine Arithmeticによる解の非存在領域の除去

4.1 はじめに

本章では 3 章で述べた Affine Arithmetic を利用して、非線形方程式の解の非存在領域を判定、除去する方法について述べる。

4.2 Affine Arithmeticを利用した非存在テスト

探索領域 I において Affine Arithmetic を用いて $f(I)$ の評価を

$$\begin{aligned} f_1 &= a_{10} + a_{11}\varepsilon_1 + \cdots + a_{1n}\varepsilon_n \\ f_2 &= a_{20} + a_{21}\varepsilon_1 + \cdots + a_{2n}\varepsilon_n \\ &\vdots \\ f_m &= a_{m0} + a_{m1}\varepsilon_1 + \cdots + a_{mn}\varepsilon_n \end{aligned} \tag{4.1}$$

のように得ることを考える。ただし $m \leq n$ となる。この評価式をそのまま利用あるいは応用することによる解の非存在判定の方法が提案されている。以下にそれらを示す。

4.2.0 区間表示に還元する方法

従来の Stolfi の方法であるが、単純に $f_1, \cdots, f_m$ を式 (3.3)、式 (3.4) にしたがって通常の区間表示に変換する方法である。このとき、f_i を区間表示に戻したものを f_{Ii} とかくことにして、

$$0 \notin f_{Ii} \quad (1 \leq \exists i \leq m) \tag{4.2}$$

ならば、

$$f(x) = 0 \text{ の解が } I \text{ に存在しない}$$

ことがいえる。この方法は、解が存在しないための十分条件 $0 \notin F(I)$ の判定において、区間包囲 $F(I)$ の評価に Affine 形式の区間表示への還元を用いたものであるが、3.3 節に述べているように、この還元によって Affine 演算のもつ相関性が失われ、評価が甘くなることが考えられる。そのため、その点を考慮した新手法を以下に述べる。

4.2.1 絶対値の和を用いる方法

$$R = \sum_{i=1}^{m} |f_i| \tag{4.3}$$

を Affine Arithmetic により評価する。このとき、

$$R > 0 \Rightarrow I \text{ に } f(x) = 0 \text{ の解は存在しない。}$$

がいえる。この方法は、4.2.0 の方法ともとの考え方を同じくしており、別の角度から評価したものとみることができる。なお、Affine 多項式に対する絶対値関数

$$|f| = |a_0 + a_1\varepsilon_1 + \cdots + a_n\varepsilon_n| \tag{4.4}$$

はひとつの単項演算としてその結果を求めることができる（図 4.1）。Affine 形式 R に対する不等式 $R > 0$ の判定は、R の式 (3.3)、式 (3.4) によって得られる通常の区間表示の下端を 0 と比較する。

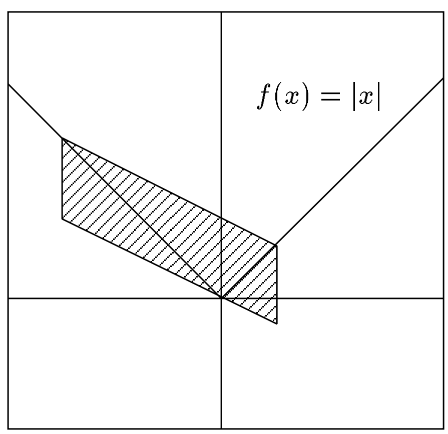

図 4.1: Affine 形式の絶対値関数 $|x|$

4.2.2 一般逆行列を用いる方法

式 (4.1) から $\varepsilon = (\varepsilon_1 \cdots \varepsilon_n)^t$ についての線形方程式

$$\begin{gathered} a_{10} + a_{11}\varepsilon_1 + \cdots + a_{1n}\varepsilon_n = 0 \\ a_{20} + a_{21}\varepsilon_1 + \cdots + a_{2n}\varepsilon_n = 0 \\ \vdots \\ a_{m0} + a_{m1}\varepsilon_1 + \cdots + a_{mn}\varepsilon_n = 0 \end{gathered} \tag{4.5}$$

を解くことを考える。

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \quad b = \begin{pmatrix} -a_{10} \\ \vdots \\ -a_{m0} \end{pmatrix}$$

により、式 (4.5) は、

$$A\varepsilon = b \tag{4.6}$$

とかける。ダミー変数の条件により、

$$-1 \leq \varepsilon_i \leq 1 \quad (i = 1, 2, \cdots, n) \tag{4.7}$$

で表される超立方体と方程式 (4.5) の解空間が交わりをもたない

$$\Rightarrow I \text{ に } f(x) = 0 \text{ の解が存在しない}$$

ことがいえる（図 4.2）。この交わりが空かどうかを判定するには線形計画問題を利用する。すなわち、式 (4.5)、式 (4.7) を制約条件として、単体法の Phase-1 を動かす。その結果、有限ステップで許容解が存在するか否かが判定される。許容解が存在しないときは (4.5) の解空間と (4.7) の超立方体が交わりをもたないことになる。

そこで (4.7) の超立方体の代わりに、それを包み込むような半径 $\sqrt{n}$ の超球を使う（図 4.3）。この方法では (4.5) の解でユークリッドノルム最小のものを得て、そのノルムが $\sqrt{n}$ より大きいならば (4.5) の解空間と (4.7) の超立方体が交わりをもたないことがいえる。この方法では、解の非存在領域の除去能力は劣るものの、実行時間は短くできると考えられる。

なお、方程式 (4.6) の最小ノルム解は

$$A^t(AA^t)^{-1}b \tag{4.8}$$

で得ることができる。

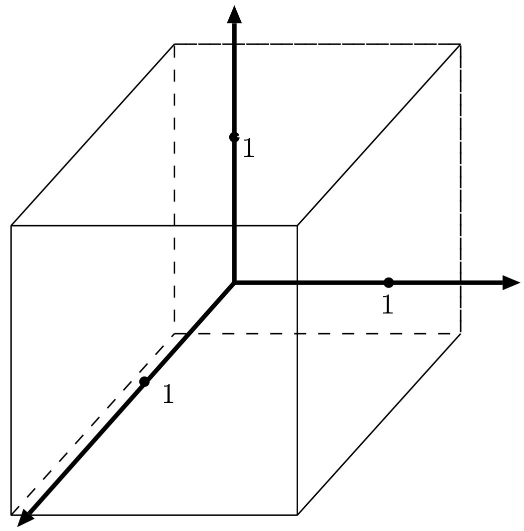

図 4.2: 超立方体 $-1 \leq \varepsilon_i \leq 1$

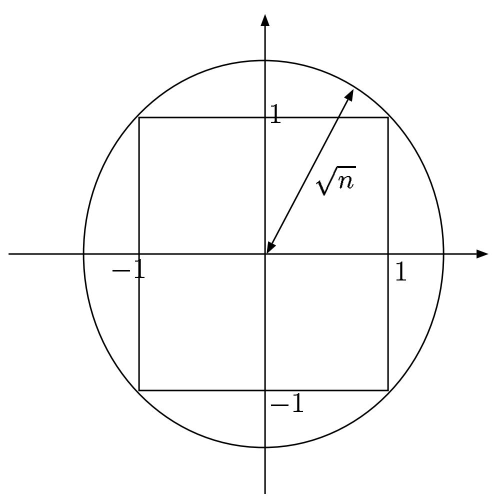

図 4.3: 超立方体と超球

4.3 おわりに

本章では Affine Arithmetic による評価を利用した解の非存在領域の除去に関して、従来の方法と、新手法について述べた。これらの新手法が性能面で優れているのかを 5 章で検証することにする。

第5章　数値実験

5.1 はじめに

本章では 4 章で述べたアルゴリズムを実装し、実際に曲線を可視化する際の計算に於いて、従来の非存在判定法と新手法との性能の差を比較し、Affine Arithmetic による非存在領域の除去性能を検証する。

5.2 実験に関して

区間演算と Affine Arithmetic による解の非存在領域の除去性能を比較するために、以下のようなアルゴリズムで実験を行うこととする。
また、非存在領域の判定が有効になされているのかを見る指標として、全解探索検索数を変化させながら、分割回数や実行時間を表示させることで、区間演算による方法や Affine Arithmetic による方法 (区間表示に還元する方法、絶対値の和を用いる方法、一般逆行列を用いる方法) に使用する。

1. 区間のリスト $\mathcal{L}$ を $\mathcal{L} = \{T\}$ で初期化する。
2. $\mathcal{L}$ が空ならば終了。そうでなければ、$\mathcal{L}$ の先頭の要素を取り出して、I とし、$\mathcal{L}$ から削除。
3. **テスト 1** 区間演算による非存在テストを行う。

   テスト 2 Affine Arithmetic の区間表示への還元による 4.2.0 の非存在テストを行う。

   テスト 3 絶対値の和を用いた 4.2.1 のテストを行う。

   テスト 4 一般逆行列を用いた 4.2.2 のテストを行う。

4. 解が存在しない場合、I を消去し、2 へ
5. 解が存在する場合、I をふたつの区間 I_1,I_2 に分割し、リスト $\mathcal{L}$ の末尾に追加する。2 へ。

5.3 プログラムについて

プログラミング言語に Visual C++を用いる。Visual C++を利用して区間演算、行列、Affine 演算を実装し、それらを用いて以下のような環境で行う。

5.3.1 実験環境

数値実験を行う計算機環境は以下のとおりである。

CPU PentiumII 995MHz

メモリ 240MB

OS Windows XP

5.4 実験

5.4.1 例題

二つの曲面の共有解を求める際に、区間演算や Affine Arithmtic を利用して、解の非存在領域の除去性能を検証する。
解の探索範囲を $[-10, 10] \times [-10, 10] \times [-10, 10]$ とする。

$$\begin{array}{l} f_1 : (x-2)^2 + (y-2)^2 + (z-2)^2 - 26 = 0 \\ f_2 : (x+2)^2 + (y+2)^2 + (z+2)^2 - 26 = 0 \end{array} \tag{5.1}$$

5.4.2 実行結果

全解探索検索	100	300	500	700
区間演算	32	132	174	206
Affine	12	56	94	131
絶対値の和	38	64	78	124
一般逆行列	60	58	98	146

表 5.1: 共有解を含む領域を絞る際に残った分割回数 (回数)

全解探索検索	100	300	500	700
区間演算	1.7	1.8	3.5	4.8
Affine	28	84	144	186
絶対値の和	31	94	159	227
一般逆行列	30	91	154	219

表 5.2: 共有解を含む領域を絞る際にかかる時間 (秒)

5.4.3　考察

今回の実験を考察する前に分割回数について述べる。分割回数とは、ある領域に共有解が存在する場合には分割が行われ、また分割された領域に共有解が存在しない場合には、その領域の消去が行われ、その結果、残った領域の個数でもある。したがって、分割回数が多い理由としては、共有解の存在領域を把握しているものの、非存在領域の消去ができていないためである。また分割回数が少ない理由としては、共有解の存在領域を把握しながら、非存在領域の消去も行われているためであり、少ない分割回数の方が、存在領域は、より限定されていることになる。このようなことから、共有解の存在領域の絞込みができているもの、つまり、非存在領域の除去性能があるのは、分割回数が少ない方である。
今回の実験（表 : 5-1）から、全解探索検索の増加にともなって区間演算と Affine Arithmetic との分割回数の間に大きな差が出ていたことがわかる。この結果から、区間演算では区間幅の増大により、分割回数が増加したと考えられる。また Affine Arithmetic による非存在テストでは変数間の相関を考慮して計算が行われることから、区間幅の増大を抑えることができたために、分割回数が区間演算よりも減少したと考えられる。したがって、Affine Arithmetic による非存在領域の除去性能が有効であることが示された。
また（表 : 5-2）には、全解探索検索にかかる実行時間を表したものだが、やはり区間幅の増大を抑えるための計算に時間がかかるために、Affine Arithmetic による非存在領域の判定の方が圧倒的に時間がかかってしまったようである。

5.5　まとめ

本章では実際に二つの非線形方程式の共有解を求める上で、区間演算や Affine Arithmetic による判定を用いて、解の非存在領域の除去を行い、共有解の存在領域を絞ることができた。また、区間演算や Affi ne Arithmetic による非存在領域の除去性能の比較においては、それぞれの演算の特徴が出た結果が得られ、Affine Arithmetic による判定に有効性が示された。

謝辞

本研究を行い, 論文を作成するにあたり, あらゆる面で適切かつ丁寧なご指導とご助言を幾度となくいただき, 日常のいろいろな会話からもたくさんの知識とものの考え方をいただいた柏木雅英助教授に心より深く感謝いたします.

また, 本研究と関わりの深い数値計算の様々な知識を講義などでご教授くださり, 自分に精度保証付き数値計算への興味をもたせてくれた大石進一教授に深く感謝いたします.

また, 本研究に非常に多くの機会で親身になってご助言くださった柏木研究室助手の宮島信也氏に深く感謝いたします.

また, 知識不十分な自分にたくさんの有用な知識と技術をくださり、終始丁寧なご指導をいただいた柏木研究室修士課程 2 年の円田直氏に深く感謝いたします.

また、本研究にご助言くださるとともに日常生活でもお世話になった柏木研究室の金子大樹氏、小糸彰氏、河野大祐氏、富永雄一氏、大和資治氏、麻生香雄氏、波多野亮氏、山下亮輔氏に深く感謝いたします。

また、時に励ましあい、ともに楽しい時間を過ごした柏木研究室 4 年の具志頭大輔氏、玉井純氏、辻裕介氏、野本侑希氏、花井正宏氏に深く感謝いたします。

最後に、研究含めすべての面でお世話になった柏木研究室の環境と、それを作り維持発展してきたすべての関係者の方々に深く感謝いたします。

参考文献

[1] 大石進一著"数値計算", 裳華房,1999 年

[2] 大石進一著,"精度保証付き数値計算", コロナ社,2000

[3] L.W.Swanson 著, 田畑吉雄訳,"線型計画法", 現代数学社,1983

[4] Marcus Vinícius A.Andrade, João L.D.Comba and Jorge Stolfi,"Affine Arithmetic",INTERVAL'94, St.petersburg(Russia),March 5-10,1994

[5] 神沢雄智, 柏木雅英, 大石進一, 中村晴幸,"有限ステップで停止する非線形方程式のすべての解を精度保証付きで求めるアルゴリズム", 電子情報通信学会論文誌:Vol.J80-A,No.7,pp.1130-1137,1997.7

[6] 柏木雅英,"Affine Arithmetic とその応用", 日本応用数理学会年会, 平成 11 年度

[7] 川上修,"Affine Arithmetic を利用した全解探索", 平成 11 年度卒業論文

[8] 菊池智行,"Affine Arithmetic を利用した非線形方程式の解の非存在の除去", 平成 12 年度卒業論文