数理解析研究所講究録
第1976巻 2015年 17-26

Asirでの3変数陰関数描画について
Drawing graphs of trivariate implicit functions on Asir

近藤祐史*
香川高等専門学校
YUJI KONDOH
NATIONAL INSTITUTE OF TECHNOLOGY, KAGAWA COLLEGE

大墨礼子†
サレジオ工業高等専門学校
NORIKO OSUMI
SALESIAN POLYTECHNIC

村尾裕一‡
電気通信大学
HIROKAZU MURAO
THE UNIVERSITY OF ELECTOR-COMMUNICATIONS

齋藤友克§
(株)アルファオメガ
TOMOKATSU SAITO
ALPHAOMEGA INC.

Abstract

We have long been working on developing algorithms for drawing graphs of implicit functions. We investigate methods for trivariate cases as a simple extension of the existing methods for bivariate cases. The basic strategy of our methods consists of division of the 3D space in the display area into a contiguous sequence of tiny cubes, and the check for every cube of the existence of the solutions to the polynomial equation defining a implicit function. We explain how we extend this method to trivariate cases. We also show some sample results obtained by our several experimental implementations on Asir for Windows.

1 はじめに

数式処理システム Risa/Asir には，2変数陰関数の実数上での零点の描画機能として関数 ifplot が実装されている．ifplot は，数学的に正確な描画を目的として，文献 [1][2] で提案されたアルゴリズムの一部により描画する．基本的なアイデアはは，描画する領域を Cell と呼ばれる矩形に区切り，Cell 内に零点が存在す

*kondoh@di.kagawa-nct.ac.jp
†n-osumi@salesio-sp.ac.jp
‡murao@cs.uec.ac.jp
§saito@a2z.co.jp

るかを判定する．判定には，Character と呼ばれる指標関数を用いている．指標関数は判定基準に応じて，Sign Weak Character，Boundary Character，Faithful Character がある．Sign Weak Character は矩形の端点の符号を判定し描画を行う指標関数であり，Boundary Character は Sturm の定理を利用して Cell の境界上に零点が存在するかを判定し描画を行う指標関数である．最も判定基準が厳格なグレブナー基底を用いて判定する Faithful Character による描画も，実現しているが，これは Asir 言語での実装で提供されている．ifplot を起動すると，デフォルトでは Sign Weak Character を用いたアルゴリズムで描画が実行される．

実装された当時の計算機環境の制約により，Sign Weak Character の実装には浮動小数点数演算が多用された。そのため，次数や項数の多い陰関数では，正確に表示できないなどの問題も指摘されている．そこで，近年 ifplot の改良 [4]，3D プリンタへの対応の検討 [5]，3 変数陰関数描画 [3][6] などを行っている．そこでは，実装において FreeBSD 上での Asir に対して試作を行っている．しかし，Asir の普及を考慮すれば，他の OS 版での動作の検証も必要である．Linux 等 X-window を用いているシステムでは，ソースコードは共通であるため，容易に移植可能であると思われる．そのため，本稿では，Windows 版への移植について報告を行う．

2　2 変数の陰関数描画

2 変数では，表示できる精度に分割した格子の 1 つの Cell 内に $f(x,y)=0$ の零点が含まれるかどうかを指標関数に基づき判定する方法を提案した [1][2].

関数の零点を描画するためには，描画装置が存在する．つまり，関数の零点描画とは，描画装置に依存していることを認識する必要がある．その装置により描画する画素の単位を Cell と呼ぶ．この単位は，装置の限界性能の必要性はなく，利用者の必要な精度であることからあえて装置解像度ではなく Cell と呼ぶ．単純に Cell とは何かといえば零点描画をする場合の描画最小単位のことである．

定義 1 (Cell の定義)
表示領域 D 上の $\mathcal{C}_k$ $(k=1,\ldots,m)$ が D 上定義された *Cell* であるとは，

1. $D=\bigcup_{k=1}^{m}\mathcal{C}_k, \qquad \mathcal{C}_k^i\bigcap\mathcal{C}_j^i=\phi, \qquad k\neq j,$
2. 各 $\mathcal{C}_k$ の *Jordan* 測度はゼロではない，

とする．ここで $\mathcal{C}_k^i$ は，$\mathcal{C}_k$ の内点の全体とする．

2.1　描画関数（Character）

描画空間の定義の Cell が定まった状態で描画を定める描画関数 (Character) を定義する．

定義 2 (Character の定義)
D 上定義されている関数 f の *Cell* の族 $\{\mathcal{C}_k\}$ による *Character* χ とは，

1. $\chi:\{\mathcal{C}_k\}\to\{0,1\}$
2. $\chi(\mathcal{C}_k)=0$ ならば $\mathcal{C}_k$ の任意の元 x に対し $f(x)\neq 0$

とする．

この描画関数の意味は，関数 $f(x)$ の零点を含む Cell に対する値は 1 である．つまり描画される．$\chi(\mathcal{C}_k)=1$ の場合は，零点を含む可能性があることである．この可能性の程度を定めることにより描画関数は，様々なものが考えられる．さらに，描画関数全体は，包含関係により束の代数構造を持つ．この束の極大元が存在しその元が Faithful Character であることが示されている [1].

定義 3 (Faithful Character)
D 上定義されている関数 f の *Cell* $\{\mathcal{C}_k\}$ による *Faithful Character* とは，

$$\chi(\mathcal{C}_k)=\begin{cases} 0 & f(x)\neq 0 \quad {}^{\forall}x\in\mathcal{C}_k \\ 1 & f(x)=0 \quad {}^{\exists}x\in\mathcal{C}_k \end{cases}$$

ここで Character の値が 1 の場合に点（Cell）を描画する．

実用のためには Character の概念を弱めた Weak Character と呼ばれるものを考えることは重要である．条件を弱めるとは，$\chi(\mathcal{C}_k)=0$ であっても解を含む場合がある，とすることである．この弱い Character の例として Sign Weak Character などがある．

以下，2 変数の場合の Character を示す．

定義 4 (Sign Weak Character)
D 上の *Cell* 族 $\{\mathcal{C}_k\}$ による関数 $f(x_1,\ldots,x_n)=0$ に対する *Sign Weak Character* σ とは，

1. $\sigma:\{\mathcal{C}_k\}\to\{0,1\}$
2. $\sigma(\mathcal{C}_k)=\begin{cases} 1 & \mathcal{C}_k\text{の } 2^n \text{ 個の端点の関数値の符号が異なっている} \\ 0 & otherwise \end{cases}$

定義 5 (Boundary Character)
D 上の *Cell* 族 $\{\mathcal{C}_k\}$ による関数 $f(x,y)=0$ に対する *Boundary Character* β とは，

1. $\beta:\{\mathcal{C}_k\}\to\{0,1\}$
2. $\beta(\mathcal{C}_k)=\begin{cases} 1 & \{(x,y)\in\mathcal{C}_k | f(x,y)=0\}\cap\partial\bar{\mathcal{C}}_k\neq\phi \\ 0 & otherwise \end{cases}$

ここで $\partial\bar{\mathcal{C}}_k$ は *Cell* $\mathcal{C}_k$ の閉包の境界をなす集合である．

大雑把にいえば，この Character は零点が Cell の境界上にある場合に 1 となる．この Character の実装としては，関数が 2 変数有理係数代数関数であれば，Sturm 列を求め Sturm の定理により解の判定が確実にできることを利用すれば計算できる．

定義 3 の Faithful Character は，Character の中で最も正しい Character である．現在この Character の実現可能な関数は，多変数多項式の場合のみである．

1. $f(x,y)$ を無平方化する．以下この無平方化した関数に関して実行する．
2. Boundary Character を実行する．
3. $\left\{\frac{\partial f}{\partial x},\frac{\partial f}{\partial y},f(x,y)\right\}$ の Gröbner 基底を求める．
4. Gröbner 基底の解の属する Cell を決定する．

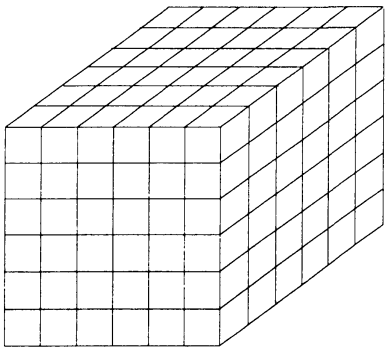

図 1: 表示空間

3 3変数の陰関数描画

必要な精度に分割した格子の1つのセル（ボクセル）内（図 1）に $f(x,y,z)=0$ の零点が含まれるかどうかを判定する。

2変数の場合と同様に考えると，必要な精度に分割した格子の1つのボクセル内に $f(x,y,z)=0$ の零点が含まれるかどうかを判定することになる．

- Sign Weak Character

  8個の格子点での $f(x,y,z)$ の符号により判定する．

- Boundary Character

  格子面上での2変数の陰関数と考え，2変数の Faithful Character を用いる．

- Faithful Character

  Boundary Character の結果に加え，ボクセル内に埋没する構造を判定する必要がある．構造が，孤立点や閉曲面であれば，2変数陰関数描画で行ったのと同様に，$\{f(x,y,z)=0, \partial f/\partial x=0, \partial f/\partial y=0, \partial f/\partial z=0,\}$ を解くことにより，判定できる．実際には，$\{f(x,y,z)=0, \partial f/\partial x=0\}$ を解くことで十分である．しかし，ボクセル内に埋没する構造が3次元空間での閉曲線の場合には，上記の方程式が0次元ではなくなるために計算できず，判定不能となる．この問題は，QE や CAD などの高度なアルゴリズムを用いることにより解決できる．そのため，実装に置いては，Sign Weak Character と Boundary Character のみを行う．

4 実装

文献 [3] では，FreeBSD を用いて実装を行った．

3変数陰関数描画の結果を表示するため，OpenGL を用いた．OpenGL を用いることにより，実装が容易になるうえ，3次元で表示されているものの認識を助けるための，視点の変更や回転，拡大縮小などの操作を実現することが容易にできる．また，他のプラットフォームへの移植性が高くなる．

本稿では，Windows 版への移植を行う．移植にあたり OpenGL が利用できるように，freeglut を利用できるようにインストールする必要がある．

文献 [3][6] と同様，Sign Weak Character においては，Character でのボクセルの判定に格子点上での関数値の計算に double 型を利用し，C 言語にて実装し，表示を OpenGL にて行う．

Boundary Character においては，2 変数の Faithful Character を用いるため，Asir 言語で実装し，表示に OpenGL を用いた．

Windows 版へ移植するにあたって，同様に実装を行った．また，表現も同様にボクセルとしての表示を行った．

5 実行例

実行例として，トーラス関数

$$Torus(x,y,z) = 16x^4 + (32y^2 + 32z^2 - 40)x^2 + 16y^4 + (32z^2 - 40)y^2 + 16z^4 + 24z^2 + 9$$

とハート型関数

$$Heart(x,y,z) = (x^2 + y^2 + 2z^2 - 1)^3 - x^2y^3$$

を用いる。また，表示空間は，$-2 \leqq x \leqq 2$, $-2 \leqq y \leqq 2$, $-2 \leqq z \leqq 2$, を $128 \times 128 \times 128$ 格子で計算した．

5.1 Sign Weak Character での結果

図 2 は，トーラス関数 図 3 は，ハート型関数 の実行結果であり，十分実用に耐えうるものであることが

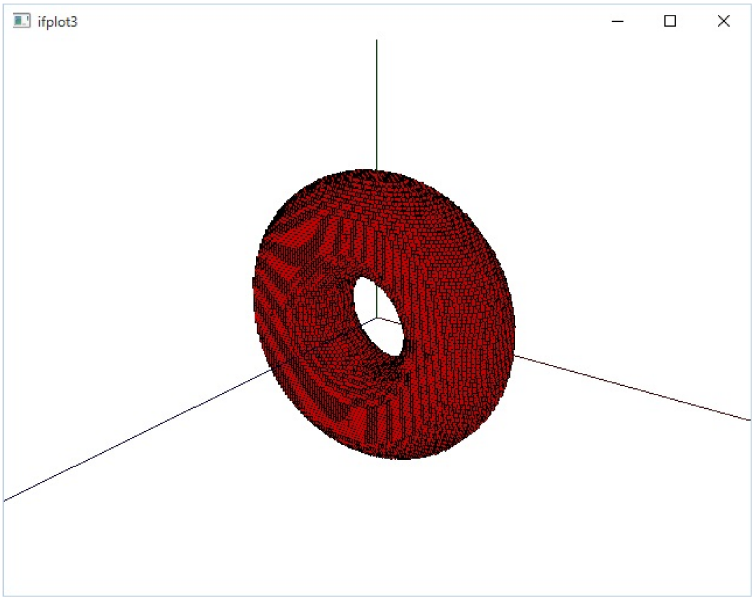

図 2: Sing Weak Character を用いたトーラス関数の表示

わかる．

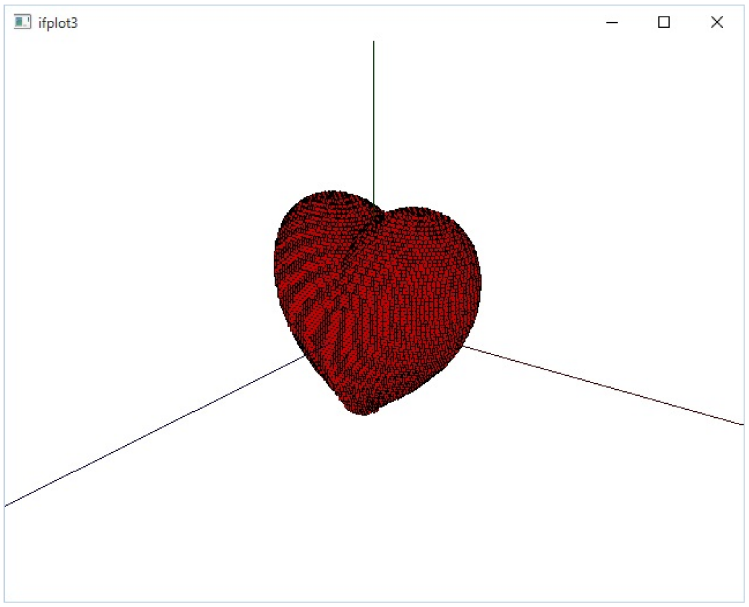

図 3: Sing Weak Character を用いたハート型関数の表示

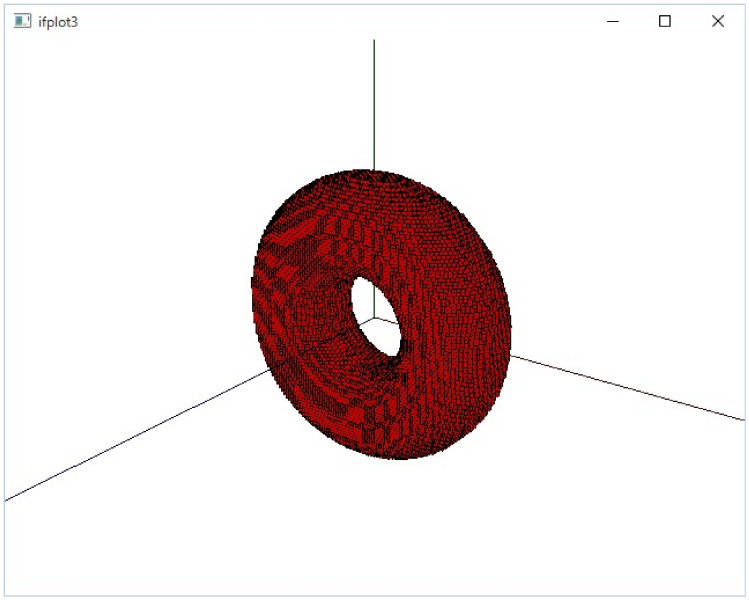

図 4: Boundary Character を用いたトーラス関数の表示

5.2 Boundary Character での結果

図 4 は，トーラス関数 図 5 は，ハート型関数 の実行結果である．どちらも Sign Weak Character との差はほとんどないが，より正確に描画できている．

図 6 は，変形した球と平面が交わるような 3 次元空間上の曲線

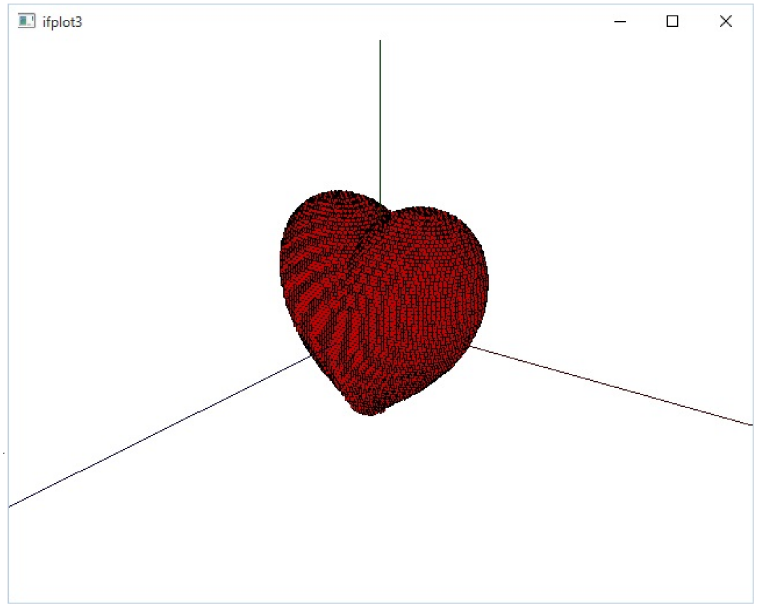

図 5: Boundary Character を用いたハート型関数の表示

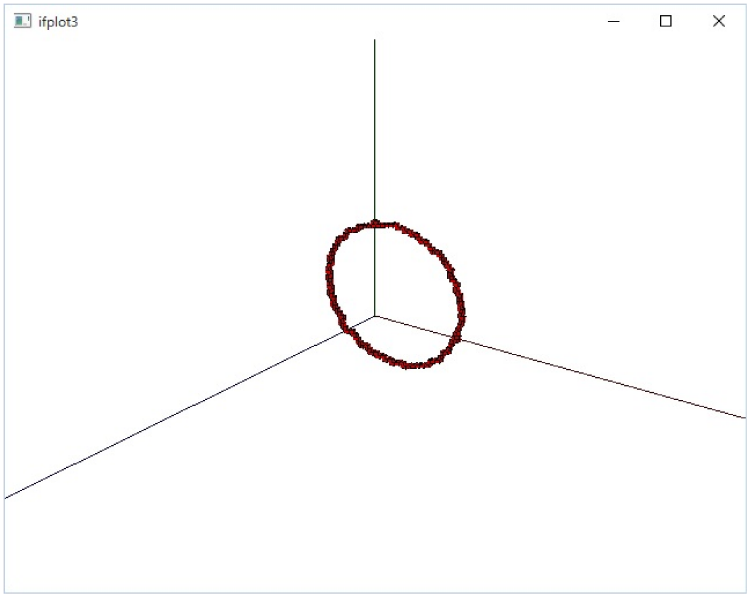

図 6: Boundary Character を用いた 3 次元空間上の曲線の表示

$$(x^2+y^2+2z^2-1)^2+(x+y+z-1)^2=0$$

を $128\times128\times128$ 格子で表示した結果である．このような空間曲線は，Sign Weak Character ではうまく表示できないが，Boundary Character では，表示できているのがわかる．

これらの実装により，実用上は問題なく 3 変数陰関数の描画を行えることを確認した．

6　各種描画関数の Windows 版での実行例

筆者らは，Asir に実現されている描画関係の実装の見直しを行っている [5]．文献 [5] では，主に FreeBSD 上の Asir への実装を行っている．そのため，Linux 等 X-window を用いる OS 上ではそのまま動作すると思われるが，Windows 系では移植を行う必要がある．そこで，本稿では Windows 版への移植を行った．その結果，ほぼ問題なく動作することを確認した．動作例を図に示す．例は，Asir のライブラリ ifplot に格納されているクローバ型の陰関数 C に対して，

$$-C > 0$$

を満たす領域を表示したものである．図は，それぞれ計算に double 型を用いている `ineqnD`，有理数を用いている `ineqnQ`，有理数計算に加えて Sturm 法（Boundary Character）を用いる `ineqnB` の結果である．

すべての関数についての検証はできていないが，Windows でも同様の結果を得ることができるようになった．

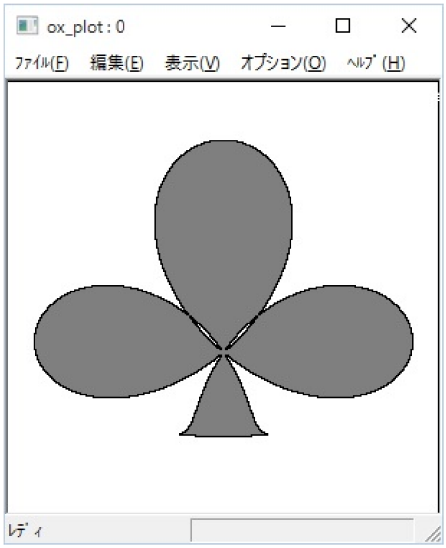

図 7: `ineqnD(-C,Gray50)` の出力結果

7　まとめ

3 変数陰関数描画 $f(x,y,z)=0$ について検討し，試作を行った．特に本稿では，Windows 版 Asir への実装について報告を行った．既に報告している Sign Weak Character と Boundary Character により OpenGL を用いて表示できることを実例により示した．3 変数の場合，Faithful Character は実現できていないが，Boundary Character で表示できないのは，ボクセル内に完全に入っている孤立点，閉曲面，閉曲線といった表示領域に比べ非常に微小な構造であるため，実用上は問題ないと思われる．

また，Asir に用意されている描画関数の見直しを行っており，本稿では Windows 版への実装を行った．

今後，なめらかに表示するためのボクセルの表現方法や簡便な方法は未解決でる Faithful Character の実現方法などの検討が必要である．また，実用に耐える実装を行い公開できるものを作っていきたい．

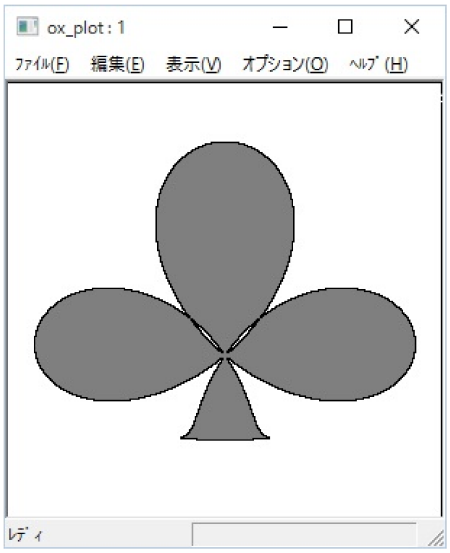

図 8: `ineqnQ(-C,Gray50)` の出力結果

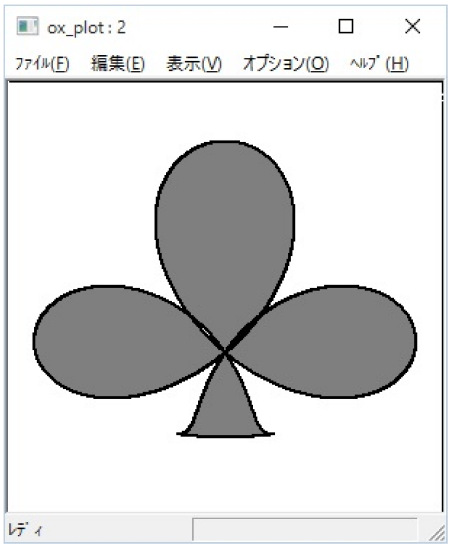

図 9: `ineqnB(-C,Gray50)` の出力結果

参 考 文 献

[1] 齋藤友克，近藤祐史，三好善彦，竹島卓，Displaying real solution of mathematical equations，数式処理，**6**(2), 1998, 2–21.

[2] T. Saito, Y. Kondoh, Y. Miyoshi and T. Takeshima, Faithful plotting of real curves defined by bivariate rational polynomials, 情報処理学会論文誌, **41**(4), 2000, 1009–1017.

[3] 近藤祐史，兵頭礼子，村尾裕一，齋藤友克，Asir での３変数陰関数描画，数理解析研究所講究録 1843, 2013, 140–145.

[4] 兵頭礼子，近藤祐史，村尾裕一，齋藤友克，ifplot の改良数式処理, **20**(2), 2014, 73–76.

[5] 兵頭礼子，近藤祐史，村尾裕一，齋藤友克，ifplot での 3 次元描画の拡張，数理解析研究所講究録 1907, 2014, 166–173.

[6] 近藤祐史，兵頭礼子，村尾裕一，齋藤友克，Asir における陰関数描画 ifplot の改良，数式処理，**21**(2), 2015, 76–79.