数式処理システム Risa/Asir の開発と応用

竹島, 卓

(Degree)
博士（学術）

(Date of Degree)
2005-12-21

(Date of Publication)
2008-07-31

(Resource Type)
doctoral thesis

(Report Number)
乙2848

(URL)
https://hdl.handle.net/20.500.14094/D2002848

数式処理システムRisa/Asirの開発と応用

竹　島　卓

２００５年

前書き

Risa/Asir という回文名は「Research Instrument for Symbolic Algebra」に由来する．それは当初富士通研究所において開発された．現在は OpenXM committers team により維持され，日々発展しているオープンソースの計算機代数システム (いわゆる数式処理システム) である．

本論文は，著者がリーダとして関与した日本独自の数式処理システム Risa/Asir の開発に関わる 20 年間の研究をまとめたものである．その中で，Risa/Asir の開発過程と，その背景となるまたはそれを利用した数学アルゴリズム研究を紹介する．さらに，いくつかの応用事例を提示することで，数理科学問題の解決手段としての数式処理の可能性を示し，数式処理がさらに多くの応用分野に展開するための示唆を与える．

本論文の構成はつぎのとおりである．

第 1 章「研究の背景」では，数式処理とはどのような学問・技術であるかを概説した．すなわち，数式処理システムの開発や理論展開の歴史を概観するとともに，高度な IT 技術としての数式処理を可能ならしめる主要なアルゴリズムを説明した．併せてこの技術の応用分野について紹介した．Risa/Asir に先行してわが国で開発された，唯 2 つの数式処理システム AL および GAL の特徴および現状についても言及した．

第 2 章「日本における数式処理研究の経緯」では，Risa/Asir の開発に並行して著者が関与して実施された数式処理アルゴリズムの研究を紹介した．これらは，グレブナー基底の計算に関するもの，連立方程式解法に関するものおよび限定子除去に関するものの 3 件である．

第 3 章「数式処理システム Risa/Asir の開発」では，現在でも日々開発が継続され発展しているわが国唯一の数式処理システム Risa/Asir について，その開発に至った経緯や歴史，特徴について述べ，数式処理技術の今後の発展のための歴史的技術資料とした．第 2 章や第 4 章で紹介できなかった開発途上の成果についても文献を引用して紹介した．

第 4 章「Risa/Asir の応用」では，開発の各段階で Risa/Asir を用いて実施した応用について紹介した．とくに 4.1 節「多項式問題に関する High Quality Computing」では，方程式解法における数値解法と数式処理による解法とを対比させ，数式処理の優位性を論じた．すなわち，「近似」に基づき，「局所性」と「零次元性」に強く依存している数値計算では解決できないにも拘らず，看過されがちな重要な諸課題が，数式

処理によれば確実に解決されることを，文献に見る実際例の誤りを正すことにより論じた．そのほか Risa/Asir の高度な代数的計算能力の紹介の 1 例として，和算の問題の解法を紹介した．和算の問題が現代の数式処理を利用することで，どのようにまた人手と比べてどの程度にスマートに解かれるか，知ることができる．

第 5 章「平面図形の精密描画とその応用」では，Risa/Asir のユニークな特徴のひとつである強力な陰函数の描画技術をとりあげた．まず，この機能 (忠実な描画) の数学的定義とその定義を満足するアルゴリズムを紹介し，従来の方法では得られなかった正確な描画が安定して可能であることを，2 つの尖点，4 つの孤立特異点をもつハート型の陰函数グラフを例として示した．ついで，その描画機能を前提にして，指定した点を指定した順に通るような図形を表す陰関数 (正確には，その零点のグラフが指定の図形になるような 2 変数多項式) を構成することを議論し，数式処理の助け (グレブナ基底や終結式計算，因数分解など) を受けながら試行錯誤的に問題を解決することを論じた．忠実な描画の実例に用いたハートの形をグラフにもつ陰関数はこのようにして得られたものである．

第 6 章「Risa/Asir の教育への応用」では，本来は研究開発での利用を意図して開発された Risa/Asir の数学教育への応用の可能性を検討した．教育目的で開発された数式処理システム DERIVE で実施された教材を題材として比較しつつ検討を行った．結果として，Risa/Asir は，現状においても若干のプログラミングを前提とすることにより，十分教育に適用可能であることを確認し，今後の改良強化の方向を論じた．

第 7 章では数式処理が技術世界により広く活用され，一層の発展が可能なために，工学系の数学教育はいかにあるべきか現状に鑑みて提案した．最後に，本研究を総括して Risa/Asir の今後の発展への抱負を述べた．

付録として，第 4 章 1 節で文献記載の不適切な方程式から得られた 1 次元解の多項式と，第 5 章 2 節で作成したスペードとクラブの陰関数多項式，および本研究に関係する文献を記載した．

2005 年 12 月 22 日
著者誌す

目 次

図目次

表 目 次

第1章　研究の背景

1.1　数式処理概観

1.1.1　数式処理システム

数式処理と計算機代数

本論文の中心的役割を果たす Risa/Asir は数式処理システムのひとつである．「数式処理」は「formula manipulation」の訳語として津田塾大学の渡辺隼郎先生が命名した．わが国においては，より新しい「computer algebra」およびその訳語としての「計算機代数」よりも「数式処理」の用語が一般には親しまれて使用されている．「数式処理（formula manipulation）」は「数式（formula）」を人が紙と鉛筆で扱うように機械（計算機）で扱うことを意味し，記号としての数式の取り扱いを念頭に置けば「記号処理」の一種と考えることができる．他方，「計算機代数（computer algebra）」とは，単なる記号としての数式よりも，数式がまさに数式として意味があるように，数学構造としての代数構造やその特徴量を，計算機を利用することによってうまく取り扱うことを意識して用いられる．

理工学の研究や実践の現場で解決が求められる現実規模の問題に対しては，単に力づくで数式を取り扱うだけでは処理が困難であり，代数的な構造に着目してその構造を利用することなしには有効なシステムとはならない．この理由から，「数式処理/数式処理システム」よりも，「計算機代数/計算機代数システム」という用語の方が海外では好まれて広く使われている．本稿では，我が国の慣例に倣い「数式処理/数式処理システム」の方を専ら用いる．

数値処理と数式処理の特徴比較

良く知られているとおり，世界初の電子計算機は砲弾の弾道を精密に効率良く計算する目的で開発された．したがって当初は，数値の加減乗除が高速にできることが計算機の使命であった．6 桁あるいは 16 桁の精度範囲で実数値を近似的に扱う浮動小数点数の発明によって，数値シミュレーション技術が急速に発展普及し，理工学の数理的問題への計算機の応用が劇的に進んで来たことは疑いない．この目的からすれば，

数量化しにくい情報を扱うことは計算機の 2 次的な利用法であった．しかるに，アセンブラ，コンパイラに始まり，文書処理，データベース，電子メール，マルチメディア，Web とインターネットに至る，計算機の使途の拡大の歴史は，計算機利用における非数値処理の比重が圧倒的に大きいことを示している．これらが扱う情報の特徴は，情報の作成には情報自体を直接に手にする以外にない，という点にある．一方，数値計算の対象となる数値は，如何に大量であろうとも，（結果に比べれば）少量のデータと，単純な計算式（定義式）とから「計算」によって得られる[1)]．

「数式」は「非数値」の代表的な情報としてしばしば対比されるが，データベースやインターネットの情報とは異なり，「計算によっていつでも得られる」という点では，「数値」情報と何ら変わるところはない．「数値」との違いは結果の情報量が「数値」に比べて圧倒的に大きい[2)]ことである．「数式処理」では，「計算対象データ」そのものが「計算の指示（手続きや関数）」であったり，「対象間の関係を示す数式や論理式」[3)]であるということが，「数値」との対比では注目されるべきである．

この「結果の情報量の大きさ」という性質のゆえに，数式処理は「数値」処理では提供できない次元の情報サービスが提供できる．実際，現代の数式処理システムは，$(10 \div 3) \times 3 = 10$ ではなく，$(10 \div 3) \times 3 = 9.9997$ と誤るような，数学にとっての致命的な欠陥がないことや，初等的ではあるが，人手で得るには長い時間が掛かり誤りも生じやすい大きな式の導出を，間違いなく実行できるという信頼に足りる忠実さを持っている．しかもその上に，決して初等的とは言えない高度な式変形をも易々と実行できるという数学巧者の能力をも備えている．これらの有用な能力の故に，理工学の技術者・研究者がもはや手放せない道具となっている．

また，このような高度な能力の恩恵により教育の方法も変わる可能性がある．実際、従来の手計算ではごく小さないわゆる教科書モデルでしか例示できなかった問題が，日常実用レベルのモデルで例示可能になっている．このことを利用して，学習者の学習意欲向上が計れる現実感のある教材の提供など，数学教育においても新しい可能性が試みられるようになって来た．

次項では，本論文の第 2 章以降で紹介する研究の技術背景として，数式処理技術の歴史を概観する．

[1)]砲弾の弾道は，単純な微分方程式の初期値問題として定義され，今ではパソコンなどでも，必要な精度で比較的容易に計算できる．π の数値は何億桁も計算できるが，π 自体は単位円の円周の長さと簡潔に定義でき，いつでも（計算に時間はかかっても）同じ値が再現できる．しかし，インターネットで飛び交っている情報は計算で得られるものではなく，必要に応じて再現するには，情報それ自体を蓄積するしか方法はない．

[2)]「濃い」と言った方が適切かも知れない．

[3)]代数曲線を表す 2 変数多項式，semi-algebraic set を表す不等式の集合などがその一例である．

1.1.2 数式処理略史

黎明期（第 1 世代システム）

電子計算機の発明以来，人間の知的活動を計算機上に実現しようとの試み（人工知能の研究）が追求されて来た．数式を扱うことは人間の高度な知的能力に負うことは疑い無く，計算機による数式の処理（formula manipulation）は人工知能の研究者が最初に注目した研究対象であった．

最初の数式処理のプログラムは，数式の微分を行うプログラムで，1953 年に Kahrimanian[31] と Nolan[43] とがそれぞれ独立に作成した．微分演算は加減乗除や函数合成に関しても明確な手順が定義されており，計算機でも比較的扱いやすい問題であった．しかし，不定積分を行うとなると，必ずしも明確な手順（アルゴリズム）が確立しているとは言い難い．このため，積分を計算する作業は非常に高度な知的作業と考えられていた．この積分の数式処理に対して，大学の教養課程程度の不定積分問題の殆んどを解くことができる SAINT というプログラムが 1961 年に Slagle によって作成された．このプログラムは，いわゆる発見的方法に基づいたものであり，同時期の他の類似プログラムよりも高い能力を持ち，人工知能の最初の成功例と言われる．

この人工知能の実現に由来する数式処理とは別に，個々の演算は機械的に実行できるが，処理対象が大量の式であるため，人手ではとても処理しきれない問題を解決したいという要求からも，数式処理の研究とシステムの開発が始まっている．これらは，大量の数式を扱う天体力学に代表される物理の計算においてとくに要求が強く，FORMAC, ALPACK や PM，ALTRAN などの主に多項式/有理式を取り扱うシステムが開発された．

基礎アルゴリズムにおける 3 つの発明

第 1 世代のシステムでは，多項式有理式の加減乗除や微分などができる程度で，多項式の係数も浮動小数点数値かあるいは固定小数点数値であったが，扱える数式の範囲と演算機能を拡大して数式処理の発展を進めたものは，数式処理における 3 つの基礎アルゴリズムの発明である．それらは次の 3 者である．

1. 不定積分アルゴリズム（1969 年，Risch）[59],
2. 多項式因数分解（1967-1969 年，Berlekamp-Zassenhaus）[11, 100],
3. グレブナー基底を計算する Buchberger アルゴリズム（1965 年, Buchberger）[16]

有理式を扱う上では約分は必須であり, そのためには約数である最大公約多項式（GCD）が効率よく計算できなければならない．Collins[21] や Brown[14, 15] 等による多項式

の GCD 計算に関する基本アルゴリズム（たとえば，部分終結式アルゴリズム）も数式処理にとって重要な発明である．しかし，整数上の因数分解で開発された Hensel 構成に基づくアルゴリズム[4] が同様に適用できるため，筆者も上記 3 アルゴリズムを基礎アルゴリズムの 3 発明とすることに同意する．

不定積分　先に述べたように Slagle の SAINT では発見的手法に基づき不定積分が計算されており，SAINT が求められないからといって，不定積分が存在しないとは言えない．これに対し Risch は，有理函数と exp と log とを含み，これらに函数合成を許して構成される任意の初等函数の集合が，その集合の中に不定積分を持つか否かを決定することが可能で，持つ場合にはその不定積分を計算するアルゴリズムを提示した [59]．Risch のアルゴリズムは，代数函数による拡大をも許した初等函数の集合をも含むように拡張されている [23]．この決定アルゴリズムの発明により，人工知能の典型的問題と考えられていた不定積分は，数学アルゴリズムの対象となった [50]．Risch の方法は微分方程式の解法などへも応用されている．

因数分解　中学や高校で学ぶ多項式因数分解もまた発見的手法を要求する典型的な問題であった．整数係数の 1 変数多項式の因数分解は，この教科書的な方法で探索すべき因子の係数の組合わせが有限であることから，しらみ潰し的方法を取れば原理的には求めることは可能である（Kronecker の方法など）．しかし，そのような方法では計算機の能力をもってしても，次数が少し大きくなると計算量的破綻を来たすことは明らかである．

1967 年に Berlekamp が有限体（GF(p)，p は素数）上の 1 変数多項式を効率よく因数分解する方法を与えた．この Berlekamp[11] の結果を受けて，Zassenhaus[100] がその有限体上の因子から Hensel 構成によって整数上の因子を効率よく復元する方法を与えた．その方法は GCD や多変数多項式の因数分解，代数拡大体係数多項式の因数分解へと展開/拡張され，現在でも，より大きな困難な問題に対する計算効率の改善が研究され，新しい結果が報告されている．

グレブナー基底　グレブナー基底 [16] の概念は，数式処理の基本的な処理対象（多項式）をイデアルという明確な数学概念を用いて表現することで，多項式問題の簡潔な記述と見通し良い解決策を発見する手段を与えた．グレブナー基底を計算する Buchberger アルゴリズムは，多項式を代数的に扱う数式処理にとって，最終兵器とも言える強力なアルゴリズムとなっている．GCD は関与する多項式イデアルの生成元であるので，グレブナー基底を計算することで GCD が計算できる．これは，代数拡

[4] もちろん，Hensel 構成が使用できない環を係数とする GCD 計算も必要であり，その場合には部分終結式 GCD などが使用される．

大体（特に逐次拡大形式）上の GCD 計算に応用できる[5]．グレブナー基底の概念は可換環論や代数幾何といった現代数学と密接に関わっている．このために，数学者にも注目され代数幾何の諸量の計算などに利用されている．また，数学以外でも幾何の定理証明や代数方程式解法などを利用する，工学への応用も盛んに行われるようになってきた．工学への応用については後章で高品質計算（HQC）を実現する例を挙げて詳述する．

これらの 3 つの発明がすべて 1960 年代後半のほぼ同時期に行われたことは注目される．

システム開発（第 2 世代）

1960 年代後半になると，数式を扱うアルゴリズムの開発と相まって，数式処理システムが数多く開発されるようになった．代表的なものは，MIT の MAC プロジェクトで 100 人年を掛けて開発されたといわれている MACSYMA である．これは，対話的な使用を意図して開発されており，計算機の能力がまだ高くはなかった当時から，代数拡大体の計算や不定積分など（グレブナー基底計算は当初はなかったとはいえ,）ありとあらゆる数学アルゴリズムがインプリメントされた巨大システムであった．MACSYMA は開発母体の MAC プロジェクトが解散したのちも，いくつかの団体/会社に権利が移され，改良や機能追加などが続けられたが，現在ではフリーの Maxima[6] としても提供されている．

MACSYMA に対比される同時期に開発されたシステムとして REDUCE がある．REDUCE は MACSYMA に比べると高速性に重点をおいて作成された．開発者 A. C. Hearn の熱心な普及活動が多くの研究者の賛同を得て，世界中に普及し，現在なお改良や新機能の開発が続けられている．MACSYMA が各時代の大型機でなければ利用が困難であったのに対し，パソコンの能力が向上した時期にはパソコンでも利用することもできた．日本では後藤等の東大グループが HLISP を開発，REDUCE を移植して普及に努めた．さらに，後藤は理研において数式処理専用マシン FLATS を開発し，その上で REDUCE を動かして，電子線リソグラフィーの高次回折収差の計算に成功している．

MACSYMA や REDUCE と同時期，1963 年に開発が開始された SCHOONSHIP は素粒子物理学の計算のために Veltman によって開発されたシステムである．このシステムは高速性を追求するために，アセンブラーで開発された．開発者 M. J. G. Veltman は G. ’tHooft とともに 1999 年のノーベル物理学賞を受賞したが，この素粒子物理学の成果は SCHOONSHIP の開発とそれを利用した弱電理論の高次補正項の理論計算が

[5] 4.2 節に拡大体上の GCD 計算の例がある．
[6] http://maxima.sourceforge.net/

本質的な役割を果たしており，数式処理の輝かしい成功例のひとつとなった．

第五世代コンピュータと数式処理

第 5 世代コンピュータとして知られる国家プロジェクトは 1982 年に開始した．このプロジェクトは論理形言語をベースにした知識情報処理技術の開発を目指しており，1985 年に中期の開発期間を迎えた．このプロジェクトの推進機関である ICOT ではこの中期末までに完成される逐次型推論マシン（PSI）上での応用プログラムの実証システムを募集し，そのひとつとして，筆者の提案した「初等函数の不定積分が実行できる数式処理システムの開発」が採用された．これについては第 3 章で詳述する．

ICOT では，筆者等の数式処理システム以外にも数式処理に関連する研究を行っていた．たとえば，線形代数を対象とした証明システムの開発やグレブナー基底の研究（その応用や，並列処理を利用した高速計算など）も実施していた．グレブナー基底の発明者である Buchberger も客員研究者として ICOT に滞在し，ICOT の研究者達と交流しながら，論理形言語でグレブナー基底計算プログラムを試作していた，数式処理研究者の間でもようやく注目されはじめたグレブナー基底の創始者が，この時期に ICOT で研究していたことは歴史的にも興味深い．Risa/Asir の開発を含む数式処理研究において，筆者等富士通の数式処理グループと Buchberger とは後に緊密な協力関係を結ぶことになるが，当時の筆者等の目標は「不定積分」であったため，グレブナー基底については深入りできず，Buchberger との交流もとくにないままであった．

数式処理の研究分野で最も権威のある論文誌 Journal of Symbolic Computation は，丁度 1985 年に Buchberger によって創刊され，Buchberger は最初の 10 年間の主席編集者を勤めている．この時期にグレブナー基底計算アルゴリズムが大きく進歩した．

商用システム開発（第 3 世代）と特殊用途システムなど

Maple と Mathematica　1980 年代には，ICOT の第 5 世代コンピュータプロジェクトが始まるのと前後して商用の現代的数式処理システムがリリースされた．1983 年には Maple（Waterloo 大学）が発表され，1998 年には Mathematica（Wolfram Research Inc.）が登場した．

Maple は Waterloo 大学の数式処理研究者（K. O. Geddes 他）が中心になって開発したシステムで，数学アルゴリズムの教育や研究目的で使う上での信頼性という点では評価が高い．

Mathematica は SMP（1981 年，Caltech）の主要開発メンバーであった Wolfram が，SMP での経験を活かし，グラフィックス，GUI，文書（特に数式入り文書）処理などの機能を総合したデスクトップ環境を備えて提供した．このデスクトップ環境

は，既存/新規，商用/非商用を問わず，他の数式処理システムに大きな影響を与え，数式処理システムのユーザインタフェースは Mathematica に倣うものが多くなった．Mathematica は項書換えシステムを基礎とした関数型のプログラミング言語を採用しており，パターンマッチングを利用した式変形がプログラミング言語自体でサポートされているため，必要に応じた数式の簡単化プログラム作成は容易である．当初は数式処理機能が，Maple や Reduce に比べて信頼性が低く，見劣りがしたが，最近ではグレブナー基底についても機能強化され，性能も高くなってきた．

超小型システム muMath と教育用システム DERIVE　上記の Maple や Mathematica は当時のワークステーションやハイエンドパーソナルコンピュータ向きに作成されていたが，最初からごく初期のマイクロコンピュータで動作することを念頭に開発されたシステムとして，Stoutemeyer の muMath（1979 年，Hawaii 大学）がある [58]．最初の版はメモリ 48KB 程度で Intel 8080 や Zilog Z80 プロセッサ上で動作した．これは，数学教育で使用することを念頭に改版が重ねられ，現在の DERIVE（現在は TI が販売）へと進化していった．

その他のシステム　IBM では社内使用向けに ScratchPad （I と II とがある）が作成された．これはプログラム理論でいう抽象データ型の概念を使って，様々な数学対象を公理論的に定義することができる，汎用度の高いシステムであった．後の AXIOM は ScratchPad のサブセットを商用版としたものである．Axiom[7)]も現在ではオープンソースのソフトとなり，無料で使用できる．

この他，現在も使用可能な数式処理システムとして，汎用の MuPAD[8)]，代数幾何とくに特異点理論向けの Singular[9)]，可換代数，代数幾何向けの Macaulay2[10)], 群論用 GAP[11)]，微分作用素環用 kan/sm1[12)] などが挙げられる．

1.2 日本の数式処理システム

日本で開発された数式処理ソフトは，いくつかの先行的な個別機能の数式処理プログラム[13)]や，大学などで試作したものを考慮しても，大変少ない．とくに，汎用の数式処理システムとしては AL，GAL，Risa/Asir 以外にはないといって良い．前節 1.1.2

[7)] http://savannah.nongnu.org/projects/axiom/
[8)] http://www.mupad.de/
[9)] http://www.singular.uni-kl.de/
[10)] http://www.math.uiuc.edu/Macaulay2/
[11)] http://www.gap-system.org/
[12)] http://www.math.kobe-u.ac.jp/OpenXM/
[13)] 文献 [148] にいくつか引用がある．

項の最後に微分作用素環用としてあげた kan/sm1 （高山/神戸大学）は特殊用途システムであるが我が国で開発されたシステムである．

1.2.1 AL

AL（Algebraic Language）は日本電信電話公社 横須賀電気通信研究所 池原らのチームにより開発された [148, 101, 102]．当時（1975 年）は世界的にもシステム開発の盛んな時期であり，多くのシステムが Lisp 上に作成されていたが，AL はアセンブラで作成され，自前のガーベジコレクタを持っていた．処理系は「バッチ系」と「会話系」とが区別されて作成[14)]されていた点が興味深い．さらに，コアメモリ不足への対処として，数式データの 2 次記憶への自動待避機能も備えていた．システム開発規模は会話系とバッチ系とも約 40k ステップ，128 kbyte（会話系）または 256kbyte/512kbyte（バッチ系）のコアメモリで動作する設計であった．

Lisp をホスト言語とせずに会話的使用ができるようにシステムを設計することは，一般に開発工数が大きくなる（インタプリタやガーベジコレクタが必要）ため，Lisp をホストとしない場合には，バッチ処理形態をとるシステムが多かった．たとえば，IBM の FORMAC は Lisp ではなく FORTRAN（後に PL/I）をホスト言語としたプリコンパイル型のバッチ処理形態であった．これに対し AL は，Lisp を使わずに会話処理形式での数式処理を可能にしていた点が先進的である．当時の日本は TSS システムのビジネス展開時期にあたっており，会話処理機能の実現は，NTT が DIPS での TSS サービスを開始したことを反映していると考えられる．プログラミング言語も AL と呼ばれ，これは PL/I のサブセットに数式処理の拡張を加えたものであった．

扱える数式は，多項式，有理式といくつかの標準的な数学函数から成る数式であったが，多倍長整数はサポートされていない．不定積分は Risch アルゴリズムではなく，限定された数式[15)]のみをサポートしていた．因数分解は当時ではまだサポートしていないシステムが殆んどで[16)]，AL も因数分解機能は提供していない．

AL は多項式/有理式の問題にはかなり力を発揮できたと考えられるが，動作プラットフォームが NTT の大型計算機 DIPS-1 であったため，使用できる人が DIPS-1 の利用者に限られ，その普及には限度があったことが惜しまれる．

[14)] 現在の数式処理システムは対話的に使用されることが多いが，当時の一般的な計算機利用形態としては，単純な決まった処理で人間が介入する必要性の少ない処理が多く，プログラムの一括投入による処理，いわゆる「バッチ処理」が主体であり，対話的な処理（会話処理）は普及が始まったばかりの時期であった．AL の「会話系」の存在は，人間と対話を繰り返しながら問題解決を計るような計算機の使用法が次第に普及する時期に来ていたことを示している．

[15)] たとえば，分母は 2 次函数まで，指数部分は 1 次式のみなど．

[16)] この時期の代表的なシステム，MACSYMA, REDUCE, FORMAC のうち多項式の因数分解をサポートしていたシステムは MACSYMA のみであった．AL は多倍長整数が扱えないこともあり，Berlekamp-Zassenhaus 方式の多項式の因数分解は困難であったと考えられる．

1.2.2 GAL

GAL（General Algebraic Language/Laboratory）[123] は理化学研究所の佐々木が開発したシステムである．Lisp をホスト言語としており，大規模な有理式を高速に処理できることを目標に，データ構造をとくに注意深く設計している．多項式 GCD 計算 [63] や級数展開，筆者との共同研究であるグレブナ基底のモジュラー計算 [66, 125] などのアルゴリズムの研究に用いられた．佐々木の提唱する近似代数計算をサポートする機能も追加されている．また，データベースキュエリー言語と数式パターンマッチング機能 [124] を追加し，数学公式データベース [121] との連携 [121] も試みていた．さらに数値数式混合計算への発展システムとして ANS が検討されていた [70].

当初は，いわゆる大型汎用機（富士通 M200 など）上の Lisp で開発されていたが，後に，奈良女子大学の加古[17]が開発した加古 Lisp に移植され，一般のワークステーションでも動作するようになった． しかし，2005 年現在では GAL の開発活動は停止している．

1.3 アルゴリズム研究と応用分野

数式処置は，理論，システム，応用の 3 本の柱から成る．理論とは，効率性の追求と応用の拡大に不可欠のアルゴリズム構築の基礎を与え，システムとは，アルゴリズムを計算機にインプリメントし，実際の応用のために提供すること，応用とは，数式処理のさまざまな実践により適用可能性を追求し，新しい応用の開拓や，理論およびシステムへのフィードバックを行うことである．

本節では，数式処理の基礎アルゴリズムと応用分野とを説明する．

1.3.1 基礎アルゴリズム

アルゴリズムは一般に対象の表現に依存するので，ここではごく標準的であると考えられる対象の表現，すなわち Risa/Asir での表現に従って説明する．

基本データ

整数 数式処理で扱う計算対象の基本（プリミティブ）は整数である．演算単位の桁数が制限される数値計算の整数とは異なり，数式処理では（メモリの許す限り）任意の桁の整数（任意多倍長整数）を扱うことが基本原則である．

[17] 加古富志雄．奈良女子大学 理学部 教授.

有理数/多項式　整数を基礎にして，その商体である有理数（分母，分子を整数とする分数）が構成される．この有理数を係数とする 1 変数多項式が定義され，n 変数の多項式を係数とする新たな変数の 1 変数多項式として $n+1$ 変数の多項式が定義される．

有理式　有理式は有限個数の変数の多項式の商体として，つまり，多項式を分子，分母にもつものとして定義される．

代数的数　無理数の中で代数的数（$\sqrt{2}$ や $\frac{-1\pm\sqrt{-3}}{2}$ など）はそれを定義する既約多項式（最小多項式，定義多項式）の根として定義することにより，代数的な枠組の中で正確な数としての計算が可能になる．虚数単位（$\sqrt{-1}$）も原理的にはこの枠組で計算される．ただし，代数的数を大量に扱うとその計算には，定義多項式での除算が大量に必要となり，一般的には大変重い（時間計算量の大きい）演算となる．このため，数式処理システムの利用に当っては，代数的数の演算を避ける工夫が重要になる．

有理数の演算でさえ，（それは整数の GCD 計算を必然的に含むので）整数の四則演算に比べると重い演算となる．そのため，大きな問題を扱う際には，有理数を扱う必要がないよう計算を工夫することが推奨される．

その他一般の数式　これら以外の対象，例えば $\sin(x)$ や x^y などは四則演算以外の計算のルールがあり，多項式や有理式とは区別される．その特別の演算ルールが適用されない限り，$\sin(x)$ や x^y などはそれ自体が単一の変数と同様に扱われる．すなわち，$(\exp(x))^2$ と $\exp(2x)$ とはとくに変形規則が与えられない限り，別個の対象として扱われる．一般にこのような超越的函数を含む対象は標準型が存在せず，そればかりか，2 つの表現が同一の数学対象を表しているか否かも一般的には決定できないことが知られている．これは，多項式，有理式には標準型が存在し，2 つの表現が同じ数学対象を表すかどうかは差が 0 になるか否かで決定できることとは，大きな違いである．

このように，超越函数を含む対象については，簡約化規則を問題ごとに適切に用いて計算する必要があり，固定した簡約アルゴリズムをシステムに組み込んでおくことは適切でない．

ゆえに，数式処理では，有理式，多項式の演算が基礎になる．

多項式の GCD

数式処理の最初の実用的な応用は，天体力学の計算や，素粒子物理学の計算であったが，そこで必要とされたのは主に大量の有理式の計算であった．大量かつ大規模の有理式をあつかうために不可欠の演算は，多項式の GCD 計算である．少しでも数式処理を利用すれば容易に分かることであるが，有理式の分母分子の約分を行わなけれ

ば，式は演算の進行につれて爆発的に大きくなり，システムの資源（メモリ）は瞬く間に消費し尽くされてしまう．このために，分母分子の共通因子を取り出し，約分を可能にするために多項式 GCD を効率良く計算するアルゴリズムが熱心に研究された．

数学では GCD は Euclid の互除法（ユークリッド多項式剰余列）により原理的には計算できる．しかし，実際の例に適用すると係数（数値，もしくは多項式）サイズの爆発的な膨張により計算不能になる．この係数膨張を克服するために，さまざまなアルゴリズムが考案された．多項式 GCD の計算は終結式（resultant）計算と密接に関連しており，Collins[21] や Brown[14], Brown-Traub[15] の初期の研究により開発された縮小多項式剰余列計算ゴリズムや部分終結式アルゴリズムはその成功例である．現在では $f, g \in \mathbb{Z}[y_1, ..., y_n][x]$ の GCD の計算には，$\pmod{p, (y_1, ..., y_n)}$ での共通因子から Hensel 構成により $\pmod{p^k, (y_1, ..., y_n)^l}$ での因子を構成し，最終的には $\mathbb{Z}[y_1, ..., y_n][x]$ の因子を復元する方式の，モジュラーアルゴリズムを基礎としたアルゴリズムが用いられている．

高速乗算法

数式処理演算の一番底部に位置する基本演算は，正確な数としての任意桁の整数の計算である．このことから大きな整数を効率良く計算することは数式処理にとって根元的な課題になる．たとえば，グレブナー基底の計算においては，計算の途上で表れる多項式の係数は大変大きな整数になることが多い．グレブナ基底計算の全過程を GCD と同様なモジュラー計算として実施することは，Hensel 構成の停止条件が容易に計算できるならば良いが，必ずしもそうはできないために，大きな整数の四則演算が一般には避けられない．特に，通常の筆算と同様の桁繰り上がりを伴う乗算は bit 数の 2 乗オーダーの計算時間が掛かるため，bit 数 n が大きな 2 整数の乗算には，$O(n^{\log_2(3)}) = O(n^{1.58})$ の Karatsuba アルゴリズム，あるいは $O(n \log(n))$ の FFT 乗算法の適用が有効である．

Karatsuba 法や FFT 乗算法は大規模な多項式演算にも有効である．ただし，これらの高速乗算法は一定のオーバーヘッドを伴うため，対象の規模（bit 数や次数）に応じて通常演算，Karatsuba，FFT の各演算を切替えるための指標をシステムに設け，これらが有効な対象に対してのみ動作するようにアルゴリズムが設計される．

多項式因数分解

数式処理略史でも述べたように Berlekamp-Zassenhaus による多項式の因数分解アルゴリズムは（直接には式を小さくできることで，また，より高度な代数処理の基礎部品として.）数式処理の可用性を大きく広げることになった．有理数上の 1 変数多

項式の因数分解は次数および係数の表現長の多項式時間の決定性アルゴリズムで計算可能なことが Landau により見出されたが，これは次数の 6 乗オーダーの計算量であるため，実用領域では比例定数部分のオーバーヘッドが大きく実用とは言い難い．そのため，実際には最悪では指数時間計算量のアルゴリズムが，種々の改良を伴って使用されている．

多項式を既約因子に分解することは，より高度な代数演算の基本となる．例えば，有理函数の不定積分では，分母の代数拡大体上での多項式の因数分解が必要になる．Galois 群の計算 [3] やそれを利用した方程式の可解性の決定 [91, 5] では最小分解体での因数分解が必要である．効率的な準素イデアル分解はグレブナー基底計算法の進歩 [67] によって可能になったが，グレブナー基底だけでは分解はできず，多項式の因数分解が必要であり，因数分解の能力によっても計算の効率が左右される．

因数分解の効率化に使われた中国剰余アルゴリズムや p-adic および ideal-adic な Hensel 構成などのモジュラーアルゴリズムは，GCD 計算やその他の多くの数式処理アルゴリズム（例えば，多項式要素をもつ行列の逆行列 [138] や固有値，固有ベクトルの計算 [96, 145]）にも適用され，数式処理アルゴリズムの効率化に大きく貢献した．

グレブナ基底とイデアルの計算

Buchberger によって発明されたグレブナー基底の概念とそれを計算する Buchberger アルゴリズムは数多くの研究を経て飛躍的な効率化が達成された．代表的なものに，文献 [18] や [13, 29, 79, 30, 25] などがある．Risa/Asir での効率化達成は最先端の理論成果の取込による継続的な改良によることが本質的であるが，中でも，Traverso 等の Gröbner Trace アルゴリズム [79] と Homogenization を両立させるアルゴリズム [47, 74] を開発したことはこの分野への貢献のひとつである．筆者も中国剰余アルゴリズムによるモジュラーアルゴリズムの研究を行った [65, 66, 125, 133]．これについては第 2.1 節で詳述する．

また，微分方程式や，級数，作用素などへもグレブナー基底概念が拡張され，その応用は可換環論，代数幾何，D モジュールなど，数学の多くの領域に応用されるようになった．Risa/Asir が関与したものの代表例として，大阿久による **b**-function と D モジュールの計算 [52, 51]，野呂–McKay の replicable function の計算 [44] がある．最近では，NTT 白柳等 [33] によって，グレブナ基底の理論を用いて、スピンの交換のみを使った量子コンピュータ上で CNOT という量子回路が存在することの厳密証明に用いられた．

暗号の分野ではグレブナー基底の計算に基づく暗号攻撃法や暗号設計法が研究され注目された．Risa/Asir が関与した 1 例として，下山等の DES に対する改良された線形攻撃 [68] がある．

限定子除去と制御への応用

グレブナー基底ではイデアルの種々の操作が可能であるが，そのことはイデアルの複素零点である代数多様体を扱うことができるということである．しかし，一般に理工学の応用の場面では，不等式で表現される問題を扱うことも非常に多い．不等式が陽に表れていない場合でも，扱う変量や設計パラメータは実数であることが通例である．このことは，数学的には実数の semi-algebraic set を取り扱うことを要求する．

Tarski に始まる限定子除去（Quantifier Elimination, QE）の理論は，このような実数の問題の解決に適している．Tarski のアルゴリズム [78] は原理的なものであり，その後 Collins による Cylindrical Algeraic Decomposition（CAD）[22] によってアルゴリズムの効率が大きく改善された．しかしながら，CAD でさえ変数個数の 2 重指数的な計算量であり，理工学の実際の問題へ適用するには多くの現実的な工夫も必要である．Weispfenning らは除去する変数の次数を 1 次 [40] や 2 次 [85]，3 次 [86] などの低次に限定することで，効率的な QE アルゴリズムが得られることを示した．次数は限定されたとしても，十分大きな現実的な問題のクラスが限定された次数で定式化できることも強調されている．CAD は完全な QE を実行するが，特殊な形式の問題に対してはそれに特化することで，効率的な QE を目指すというアプローチもある．穴井は，制御工学の最適化設計問題などへの応用に対してこのような方向でのアプローチ [6, 7] を試みている．

このように理工学の現実問題では実数を扱うことが多く，数学教育においてもそのような現実を反映したモデル化により教育が行われることが望ましい．

このようなことを考え合わせれば，今後の数式処理の発展方向のひとつとして QE は大いに有望である．

1.3.2 応用分野例

古典的な応用 数式処理の応用分野は天文学や高エネルギー物理学，原子核物理学などの諸計算から始まったが，現在は科学技術のありとあらゆる分野で利用されるようになった．数学では群論，統計学，代数幾何学，代数的組み合せ論，整数論，環論など，情報系では符号理論や暗号に利用されている．工学では電気工学，電子工学，機械工学，宇宙工学，制御工学など数値計算が主流ではあるが，数値計算では困難な問題に対して数式処理の適用，あるいは併用が行われるようになった．

QE のシミュレーション技術への応用 QE の新しい応用として，生物やバイオ分野を含む実験やシミュレーションからの情報の抽出がある [53, 54]．これは数式処理と数値計算とが必然的に融合したアルゴリズムとなり，数値数式融合アルゴリズムの好例となることが期待される．

数式処理の現代暗号への応用　産業応用の 1 例として，Risa/Asir が関与した楕円曲線暗号の設計問題がある．暗号に使われる楕円曲線は種々のパラメータを持つが，攻撃に対する耐性をもつには少なくともその位数が小さな素因数をもたないこと，できれば位数が素数であること，が必要になる．このため，楕円曲線の位数の高速な計算や位数が素数であることが保証された楕円曲線の生成が必要であった．この計算を因数分解だけではなく Risa/Asir の数式処理の能力の総力をあげて実現し[18]，副産物として得られたモジュラー多項式のデータを公開[19]した． この成果は筆者等の Risa/Asir の成功例というばかりでなく，数式処理の工学応用の成功例のひとつと見なすことができる [143].

数式処理と数学教育　数式処理の初期のころからの応用のひとつとして教育がある．数式処理を用いれば，教材を作ることやグラフを書かせることが容易になり，教育の効果もあがると考えられる．単に数式処理で問題を数多く解かせるだけでも，豊富な実例から数学概念にたいするイメージがゆたかになり，学ぶ側の問題に対する理解を深める役割があることも指摘されている．

この分野では muMATH とその発展である DERIVE の貢献が注目される．また，数式処理を電卓に組み込んだ製品が数式処理電卓，グラフ電卓として入手可能になり，パソコンに比べて安価であるため教育現場での使用に適しているとされている．

数式処理を使った教育の支援については，黒板モデル（出口他）や Web 連携を用いたインタラクティブ Web 授業の試みもなされている．藤本等は，Risa/Asir を手書き入力システムとともに PDA（Portable Digital Assistant）に移植し（AsirPad[28]），それを使った授業を実践している [107, 160, 161, 162].

1.4　数式処理への期待

数式処理システムへの要望は，数学的により高度な機能を達成せよとするものと，理工学の現場からの問題（これは数式処理の問題としては規模が大きいものになることが通常である）に対処できる高速性，効率性を達成せよとするものとがともに存在する．これらは，必ずしも相反するものではなく，まず非効率でも高度な機能が実現される．ついで，その非効率性を克服するアルゴリズムが開発され，実際にインプリメントされて高速性や実効性を高める，また，非効率性を克服するアルゴリズム自体より高度な数学に基づき発明されるなど，むしろ互いに他を補完しあってシステムを進化させて行くものであると考えられる．

---

[18] FSEcParamGen. 1999 年富士通．楕円曲線暗号用楕円曲線パラメータ生成ソフトウェア.

[19] http://www.labs.fujitsu.com/jp/freesoft/modularpoly/

高度な数学機能や処理の効率性ではなく，ユーザにとっての使いやすさ，意図する結果を得る手間の少なさへの要望も多い．さらに，大学初年度程度までの解析学や数値計算を習得した大多数の利用者が，アルゴリズムの背景となる代数概念の多くを知らなくとも，容易に扱えることを重視する立場もある．

しかし，高度な代数を用いることなく，理工学で利用する数学をそのまま機械的に扱うことが，アルゴリズムとして易しいかというと，決してそんなことはない．たとえば，我々は普通実数を扱うが，計算機で実数を（数値計算して扱うのではなく，代数的に）扱う技術（たとえば，QE など）の応用は，長い基礎的な研究の上に，ようやく始まったばかりである．

第2章　日本における数式処理研究の経緯

2.1　グレブナ基底のモジュラー計算

本節はRisa/Asirにまだグレブナ基底がインプリメントされていなかった時期（1987年から1988年）に，グレブナ基底計算プログラムをGALにインプリメントされていた理化学研究所の佐々木博士とともに，そのアルゴリズムの効率化，とくに係数膨張を克服する方法について検討したものである．現在に比べれば計算機の能力は2桁以上劣っており，アルゴリズムもようやく効率化が始まろうとしていたころの研究であり，現在のタイミングデータに比べれば相当見劣りするが，この研究はグレブナー基底計算の効率化の先駆けのひとつである．

2.1.1　グレブナー基底と並列処理

1988年当時には，数式処理の分野でも並列算法に関する研究が活発になってきたが[56]，それらは多項式あるいは行列や行列式に関するものが大部分であり，これらに対しては非常に効率良い算法の構成が可能である（たとえば[62]を見よ）．多項式や線形代数の演算では，それ自体が高い並列性を有していることが多いからである．これに対し，本稿では，一見しただけでは並列性が見て取れない連立代数方程式の代数的解法に対して，非常に効率良い並列算法を提示する．

連立代数方程式の代数的解法としては，現在のところ，多項式イデアルのグレブナー基底を計算する方法[20, 17, 18, 10]が最も有効であるとされているが，この計算はしばしば猛烈な係数成長を起こすことが知られている．この係数成長は，連立代数方程式の低減化に通常用いられる辞書式順序において最も顕著に現れる．しかしながら，係数成長を観察すると，それは中間式に主に生じ，最終の式における係数の大きさはそれほどでないことが知られる．したがって，係数成長の問題はモジュラー算法により解決できると考えられる．非線形問題である連立代数方程式に対する我々の並列算法とは，このモジュラー算法の並列化であり，実は最も明白な並列性に着目したものである．

多項式イデアルのグレブナー基底のモジュラー算法としては，ヘンゼル構成を用いる方法が Trinks[80] と Winkler[99] により，中国剰余算法に基づく方法が筆者ら[66] と Traverso[79] により，それぞれ発表されているが，筆者らの方法のみが完全で，Traverso の方法は確率的，Trinks の方法は特殊な問題に対してのみ適用可能，Winkler の方法は一般問題に適用可能だが不完全である．2.1.2 節ではグレブナー基底と連立代数方程式の代数的解法を手短かに復習する．2.1.3 節では筆者らのモジュラー・グレブナー基底算法を簡単に説明し，モジュラー型・並列グレブナー基底算法を提示する．2.1.4 節では連立代数方程式に応用する場合の効率化について指摘し，実際的問題に対するタイミング・データで本算法の有効性を示す．2.1.5 節では逐次計算機上で実行しうる擬似並列算法を提示する．

2.1.2 グレブナー基底と連立代数方程式

本節では，多項式はすべて体あるいは環 K 上の多項式環 $K[x_1,\ldots,x_n]$ に属するものとする．K は，整数環 $\mathbb{Z}$，有理数体 $\mathbb{Q}$，素数 p_l を法とする剰余環 $\mathbb{Z}/(p_l)$（ガロア体），$\mathbb{Z}/(p_1\cdots p_k)$ のいずれかであるとする．$K[x_1,\ldots,x_n]$ を簡単のため $K[x]$ と表すことにする．

多項式 F_i, $F_i \in K[x]$, $i=1,\ldots,r$, に対し，

$$\{u_1F_1+\cdots+u_rF_r \mid u_i \in K[x],\ i=1,\ldots,r\} \tag{2.1}$$

なる多項式集合を $F_1,\ldots,F_r$ によって生成される多項式イデアルといい，$(F_1,\ldots,F_r)$ と表す．集合 $\{F_1,\ldots,F_r\}$ をイデアルの基底という．多項式 $F \in K[x]$ を単項式の和で表したとき，

$$F = c_1T_1 + c_2T_2 + \cdots,\ c_i \in K, T_i = x_1^{e_{i1}}\cdots x_n^{e_{in}} \tag{2.2}$$

とする．このとき，指数の組 $(e_{i1},\ldots,e_{in})$ に適当な全順序を導入することにより，任意の各項を一意的に順序づけることができる．この順序を項順序といい，T_1 の方が T_2 より高順位であるとき $T_1 \triangleright T_2$ で表す．$\triangleright$ には辞書式順序，全次数順序などがある．(2.2) 式において，$T_1 \triangleright T_2 \triangleright \cdots$ であるとき，c_1T_1 を頭項（head term），c_1 を頭係数（head coefficient），T_1 を頭べき積（head power product）といい，それぞれ $\mathrm{ht}(F)$, $\mathrm{hc}(F)$, $\mathrm{hpp}(F)$ と表す．さらに，二つの多項式の各項を項順序によって先頭から順次比較していくことにより，項順序 $\triangleright$ は任意の多項式の間の半順序に自然な形で拡張される．

多項式 $G \in K[x]$ を $G = c_1'T_1' + c_2'T_2' + \cdots$，ただし $\mathrm{ht}(G) = c_1'T_1'$，とする．式 (2.2) の多項式 F のある単項 cT に対し，T が T_1' の倍数であるとき，

$$F - (cT/\mathrm{ht}(G))\cdot G = F' \tag{2.3}$$

なる多項式 F' を構成すれば, $F \triangleright F'$ となる．これを F の G による M-簡約（M-reduction）といい，$F \to_G F'$ と表す．F を G で可能な限り M-簡約し，どの項ももはや G で M-簡約できなくなったときの式を $\widetilde{F}$ とする．$\widetilde{F}$ は G に関して M-既約（M-irreducible）であるといい,

$$F \to\!\!\to_G \widetilde{F} \tag{2.4}$$

と表す．同様に，F を多項式 $G_1, \ldots, G_s$ で M-簡約して M-既約にした式を $\widetilde{F}$ とするとき,

$$F \to\!\!\to_\Gamma \widetilde{F} \tag{2.5}$$

と表す．ただし，$\Gamma = \{G_1, \ldots, G_s\}$ である．多項式 F_1 と F_2 に対し,

$$\mathrm{Spol}(F_1, F_2) = \frac{lcm}{\mathrm{hpp}(F_1)} F_1 - \frac{\mathrm{hc}(F_1)}{\mathrm{hc}(F_2)} \frac{lcm}{\mathrm{hpp}(F_2)} F_2, \tag{2.6}$$

ここに，lcm は $\mathrm{hpp}(F_1)$ と $\mathrm{hpp}(F_2)$ の最小公倍式,

なる式で計算される $\mathrm{Spol}(F_1,F_2)$ は $K'[x]$ の多項式となる．ただし，K が体のとき $K' = K$, K が環 $\mathbb{Z}$ のとき K' は有理数体 $\mathbb{Q}$ である．この多項式を S-多項式（S-polynomial）という．多項式イデアル $(F_1, \ldots, F_r)$ に対し，多項式集合 $\Gamma = \{G_1, \ldots, G_s\}$ が次の二つの条件を満たすとき，Γ をグレブナー基底（Gröbner basis）という．

$$(F_1, \ldots, F_r) = (G_1, \ldots, G_s), \tag{2.7}$$

$$\text{任意の } F \in (F_1, \ldots, F_r) \text{ に対し，} F \to\!\!\to_\Gamma 0. \tag{2.8}$$

グレブナー基底 $\Gamma=\{G_1, \ldots, G_s\}$ が次の性質を満たすとき，正規グレブナー基底という．

$$i=1, \ldots, s \text{ に対し，} G_i \text{ は } \Gamma - \{G_i\} \text{ に関して M-既約で } \mathrm{hc}(G_i) = 1. \tag{2.9}$$

与えられた $\{F_1, \ldots, F_r\}$ からグレブナー基底を構成する算法として，有名なブッフバーガーの算法がある（たとえば [20] を見よ）が，それは以下のようなものである．初期設定として，$\Gamma = \{F_1, \ldots, F_r\}$ とおく; Γ の二つの要素，F_i と F_j, $i \neq j$, に対し，$\mathrm{Spol}(F_i, F_j) \to\!\!\to_\Gamma \widetilde{F}_{ij}$ を計算する; $\widetilde{F}_{ij} \neq 0$ ならば $\Gamma := \Gamma \cup \{\widetilde{F}_{ij}\}$ とする; これを繰り返し，Γ のあらゆる要素 F_i と F_j, $i \neq j$, に対して $\widetilde{F}_{ij} = 0$ となるとき，Γ はグレブナー基底である．この構成手順は停止性をもつことが証明されている．この手順において，$\widetilde{F}_{ij}$ が $F_1, \ldots, F_r$ の線形結合で表されることは容易に分かる．同様に，$i = 1, \ldots, s$ に対し

$$G_i = u_{i1} F_1 + \cdots + u_{ir} F_r, \; u_{ij} \in K'[x] \tag{2.10}$$

と表すことができ，u_{ij} はグレブナー基底の構成に付随して構成できる．

次式で与えられる連立代数方程式を考える．

$$\{F_1(x_1,\ldots,x_n)=0,\ldots,F_r(x_1,\ldots,x_n)=0\} \tag{2.11}$$

ただし，$F_i\in\mathbb{Q}[x], i=1,\ldots,r$，とし，方程式は（一般に複素数の）解をもち，その個数は有限個[1)]と仮定する．この方程式を代数的手法で解くのは以下のようになる．

代数的フェイズ: 連立方程式（2.11）を以下の形に変換する．

$$\begin{aligned}\{&G_1(x_1)=0,\ G_{21}(x_1,x_2)=0,\ \ldots,\ G_{31}(x_1,x_2,x_3)=0,\\ &\ldots,\ G_{n1}(x_1,x_2,\ldots,x_n)=0,\ \ldots\}\end{aligned} \tag{2.12}$$

ただし，$G_{ij}\in\mathbb{Q}[x_1,x_2,\ldots,x_i], i=2,\ldots,n,$ であり，$\mathrm{ht}(G_{ij})=c_{ij}x_i^{m_{ij}}, c_{ij}\in\mathbb{Q},$ である．なお，（2.12）式は以下の簡単な形式になることが多い．

$$\{G_1(x_1)=0, x_2-\widetilde{G}_2(x_1)=0,\ldots,x_n-\widetilde{G}_n(x_1)=0\}. \tag{2.13}$$

数値的フェイズ: 連立方程式（2.12）（あるいは（2.13））を数値的に解く．連立方程式（2.12）は数値解法に非常に好都合な形式であることに注意されたい．まず $G_1(x_1)=0$ を解く．この解のひとつ，それを α とする，を他の方程式に代入すれば，第 2 式は $G_{21}(\alpha,x_2)=0$ となり x_2 に関して数値的に解ける，以下等々．

したがって，問題は（2.11）式を（2.12）（あるいは（2.13））の形に変換することであるが，グレブナー基底の理論によると，$x_n\triangleright x_{n-1}\triangleright\cdots\triangleright x_1$ なる辞書式順序でイデアル $(F_1,\ldots,F_r)$ のグレブナー基底 Γ を計算すれば，$\Gamma=\{G_1,G_{21},\ldots,G_{n1},\ldots\}$ となることが証明されている．それゆえ，問題はグレブナー基底の計算に帰着される．

2.1.3　モジュラー・グレブナー基底算法とその並列化

前節で述べた数値的フェイズを並列化するのは極めて容易である．まず，$G_1(x_1)=0$ を解く．この根を $\alpha_1,\ldots,\alpha_\nu$ とするとき，各 α_i に対して $x_2,x_3,\ldots$ を順に解いていく計算は全く独立であり，並列化できる．ν は x_1 の解の個数に等しく，一般にかなり大きな値になるから，この並列計算の並列度はほぼ ν である．以下では，代数的フェイズの計算のみを論じる．

まず，代数的フェイズの計算の基礎となるモジュラー・グレブナー基底算法を大雑把に解説する．多項式 $F_1,\ldots,F_r$ は $\mathbb{Q}[x]$ の要素とする．係数の分母を払うことにより，これらは $\mathbb{Z}[x]$ の要素とみなせる．したがって，素数 p_l に対して，多項式の係数をガロ

[1)] 代数方程式系の解が有限個のとき，その方程式系を零次元方程式という．

ア体 $\mathbb{Z}/(p_l)$ に落としてグレブナー基底 $\Gamma^{(l)}$ を計算できる．この計算では係数成長が生じる余地はない. $\Gamma^{(l)}$ の構成は $\mathbb{Q}$ 上のグレブナー基底 Γ の構成と全く同じ手順によって行う．すなわち，以下の 3 条件を課す.

（a）多項式 F を $\{F_1, \ldots, F_r, \ldots\}$ で M-簡約するときは，M- 簡約に用いる多項式の選び方を同じとし，選び方は係数によらないものとする.

（b）グレブナー基底構成算法において，Spol(F_i,F_j) の F_i と F_j は係数によらない同じ選び方で選ぶとする.

（c）グレブナー基底の正規化を係数によらない同じ方法で行う.

これらの条件を満たすには，Γ と $\Gamma^{(l)}$ の構成を同じプログラムで係数演算だけを変えて行えばよい.

今，既約化された S-多項式 $F_{r+1}, F_{r+2}, \ldots$ を順に計算して Γ が得られるとし，$\Gamma^{(l)}$ は同様に $F^{(l)}_{r+1}, F^{(l)}_{r+2}, \ldots$ を順に計算して得られるとする.

$$p_l \not| \, \mathrm{hc}(F_j), \ j = 1, \ldots, r, r+1, \ldots \tag{2.14}$$

であるとき，素数 p_l はラッキー（lucky）であるといい，そうでないときアンラッキー（unlucky）であるという．M-簡約および S-多項式構成において，多項式の係数の分母に現れ得るのは $F_1, \ldots, F_r, F_{r+1}, \ldots$ の頭係数だけであるから，p_l がラッキーなとき

$$F_j \equiv F^{(l)}_j \pmod{p_l}, \ j = 1, \ldots, r, r+1, \ldots \tag{2.15}$$

が成立し，相異なるラッキーな素数 $p_1, \ldots, p_k$ に対するグレブナー基底 $\Gamma^{(1)}, \ldots, \Gamma^{(k)}$ から，$\mathbb{Z}/(p_1 \cdots p_k)$ 上での基底 $\Gamma^{(0)}$ が中国剰余算法で構成できる．このとき，基底を正規化しておけば

$$\Gamma \equiv \Gamma^{(0)} \pmod{p_1 \cdots p_k} \tag{2.16}$$

となる．$\Gamma^{(0)}$ の要素の係数は整数であるが，一定の条件下で有理数に変換する．すなわち，以下の定理に基づいて変換する.

[定理] 与えられた整数 u と正整数 m, ただし $-m < u < m$, に対し, $u \equiv a/b \pmod{m}$ を満たす整数 a と b は，$|a| < \sqrt{m/2}, 0 < b < \sqrt{m/2}$ なる条件下で高々ひとつだけ存在する.

したがって，こうして得られた $\mathbb{Q}$ 上でのイデアル基底 Γ' は，k が十分に大きいとき，Γ に一致するはずである．問題点は，p_l がラッキーでないときどうするか?，k をどのように決定するか?，整数の有理数化の効率的算法，などであるが，いずれも [6] で解決されている．本稿ではアンラッキーな素数 p_l についてだけ簡単に説明するに止める.

$\mathbb{Z}/(p_l)$ 上でのグレブナー基底 $\Gamma^{(l)}$ の構成において，j 番目に構成する S-多項式として $\mathrm{Spol}(F_{j1}, F_{j2}) \rightarrow\!\!\!\rightarrow F_j^{(l)}$ を計算したとする．このとき，j と $\mathrm{hpp}(F_j^{(l)})$ の対をインデックス j に関する増加リストとして残しておく．このリストを頭項ヒストリ（head term history）と呼ぶことにする．素数 p_l に対する頭項ヒストリと素数 $p_{l'}$ に対する頭項ヒストリが異なっているとき，少なくともどちらかの素数がアンラッキーであり，より小さな j に対して頭項が消えた方の素数に対する計算を捨てる．こうして，$\Gamma^{(1)}$,...,$\Gamma^{(k)}$ が同じ頭項ヒストリで計算できたとき，結果の Γ' が正しくないのは p_1,...,p_k がすべてアンラッキーな場合のみである．

$\mathbb{Z}/(p_1 \cdots p_k)$ 上での基底 $\Gamma^{(0)}$ を有理数変換して得られるイデアル基底を $\Gamma' = \{G'_1, \ldots, G'_{s'}\}$ とする．Γ' に対してまず次の二つのチェックを行う．

チェック I: Γ' の任意の要素 G'_i と G'_j に対し $\mathbb{Q}$ 上で $\mathrm{Spol}(G'_i, G'_j) \rightarrow\!\!\!\rightarrow_{\Gamma'} 0$.

チェック II: $\mathbb{Q}$ 上で $F_i \rightarrow\!\!\!\rightarrow_{\Gamma'} 0$, i=1, ..., r.

これらのチェックにパスすれば，Γ' は $(F_1, \ldots, F_r) \subseteq (G'_1, \ldots, G'_{s'})$ なる $\mathbb{Q}$ 上のグレブナー基底であることが分かる．つぎに，$\Gamma^{(l)}$ の計算に付随して，

$$G_i^{(l)} \equiv u_{i1}^{(l)} F_1 + \cdots + u_{ir}^{(l)} F_r \pmod{p_l} \tag{2.17}$$

なる $u_{ij}^{(l)} \in \mathbb{Z}/(p_l)[x]$, $j = 1, \ldots, r$, を計算する（前節（2.10）を参照）．$u_{ij}^{(l)}$, $l = 1, \ldots, k$, に対しても中国剰余算法と整数の有理数化を施して，

$$G'_i \equiv u'_{i1} F_1 + \cdots + u'_{ir} F_r \pmod{p_1 \cdots p_k} \tag{2.18}$$

なる $u'_{ij} \in \mathbb{Q}[x]$ を構成する．そして以下をチェックする．

チェック III: $\mathbb{Q}$ 上で

$$G'_i = u'_{i1} F_1 + \cdots + u'_{ir} F_r,\ i = 1, \ldots, s'. \tag{2.19}$$

(2.19) が成立するとき，$(G'_1, \ldots, G'_{s'}) \subseteq (F_1, \ldots, F_r)$ となり，チェック I,II と合わせれば $\mathrm{ideal}(\Gamma') = \mathrm{ideal}(\Gamma)$，すなわち Γ' は $(F_1, \ldots, F_r)$ の $\mathbb{Q}$ 上でのグレブナー基底であることが主張できる．逆にチェック I,II,III のどれかでも失敗すれば，k の値は十分大きくないとして，さらに別の素数を取り出して計算を続行する．アンラッキーな素数の個数は有限であるから，この手順は必ず停止する．

上記の計算手順を見れば，$\Gamma^{(1)}, \ldots, \Gamma^{(k)}$ の計算は並列的に実行することができ，しかもラッキーな素数に対する計算はほぼ同時間で終了するはずである．$u_{ij}^{(l)}$ の構成も同様である．また，チェック I,II,III の計算も，$\mathrm{Spol}(G'_i, G'_j)$, F_i, G'_i に対してそれぞれ

並列に実行できる．さらに中国剰余算法と有理数化演算も各 G'_i に対して並列に実行できる．すなわち，ほとんどの演算が並列に実行できることが分かる．

以上より，並列プロセッサの個数が N であるときの並列グレブナー基底算法は以下となる．なお，以下では，並列プロセッサの割り当ては $\Gamma^{(l)}$ の計算に対してのみ具体的に示すとし，他の部分の計算に対しては簡単のために省略する．また，“**for** $i = a, \ldots, b$, **do parallelly**（**serially**）$\cdots$” は指標 i に関して並列（逐次的）に $\cdots$ を実行することを意味する．

アルゴリズム 1 (Parallel Modular-Gröbner)

Input: a set of polynomials $\Phi = \{F_1, ..., F_r\} \in \mathbb{Z}[x_1, ..., x_n]$;
a set of distinct primes $\{p_1, p_2, ...\}$ of word-size;
a term order $\triangleright$;

Output: a normalized Gröbner basis $\Gamma = \{G_1, ..., G_s\}$ *w.r.t.* $\triangleright$;

Condition: number of parallel procedure is N;

Step 0 [initialize]: $P := 1$; $k := 0$;

Step 1 [N Gröbner bases over $\mathbb{Z}/(p_l), l = kN + 1, ..., kN + N$]:

for N distinct primes $p_l, l = kN + 1, ..., kN + N$, **calculate parallelly [**
$\Gamma^{(l)}$ = normalized Gröbner basis $\{G_1^{(l)}, ..., G_s^{(l)}\}$ over $\mathbb{Z}/(p_l)$;$^{(*1)}$
$\Upsilon^{(l)} = \{u_{i,j}^{(l)} | i = 1, ..., s, j = 1, ..., r\}$ satisfying （2.17）;
$H^{(l)}$ = head-term history for p_l;
]

Step 2 [check unlucky primes by using head-term history]:

for $l = kN + 1, ..., kN + N$, **do serially**
if $H =$ **nil then** $H := H^{(l)}$
else if$^{(*2)}$ $H \triangleright H^{(l)}$ **then** discard results for p_l;
else if $H^{(l)} \triangleright H$ **then [**
discard the results for all the previous primes;
$H := H^{(l)}$; $P := 1$
];

Step 3 [Chinese remainder and integer-rational conversion]:

if there is no surviving prime in $\{p_{kN+1}, ..., p_{kN+N}\}$ **then goto** Step 5;
for each surviving prime p_l in $\{p_{kN+1}, ..., p_{kN+N}\}$, **do serially [**
$P' := P$; $P := P \times p_l$;
for $i = 1, ..., s$, **do parallelly [**
if $P' = 1$ **then [** $G_i^{(0)} := G_i^{(l)}$ and $u_{i,j}^{(0)} := u_{i,j}^{(l)}, j = 1, ..., r$**]**
else [by applying Chinese remainder algorithm,
update $G_i^{(0)}$ in $\Gamma^{(0)} = \{G_1^{(0),...,G_s^{(0)}}\}$ and $u_{i,j}^{(0)}$ in $\Upsilon^{(0)}$ so that

$G_i^{(0)} \equiv G_i^{(l)}, u_{i,j}^{(0)} \equiv u_{i,j}^{(l)} \pmod{p_l}$;
G'_i := convert coefficients of $G_i^{(0)}$ to rationals;
$u'_{i,j}$:= convert coeefficients of $u_{i,j}^{(0)}$ to rationals
]
];
if conversion fails to some coeefficients
then $\Gamma_{(P)} := \Upsilon_{(P)} :=$ **nil** and **goto** Step 5;
$\Gamma_{(P)} := \{G'_1, ..., G'_s\}$; $\Upsilon_{(P)} := \{u'_{i,j} | i = 1, ..., s, j = 1, ..., r\}$
];
Step 4 [check the termination]:
if $P = 1$ **or** $\Gamma_{(P')} =$ **nil or** $\Gamma_P \neq \Gamma_{(P')}$ **or** $\Upsilon_{(P)} \neq \Upsilon_{(P')}$ **then goto** Step 5;
4.1: **for each** pair $\{G'_i, G'_j\}$ in $\Gamma_{(P)}$, **check parallelly**
$\mathrm{Spol}(G'_i, G'_j) \rightarrow\rightarrow_{\Gamma_P} 0$ over $\mathbb{Q}$;
if the check fails **then goto** Step 5;
4.2: **for** $i = 1, ..., r$ **check parallelly**
$F_i \rightarrow\rightarrow_{\Gamma_{(P)}} 0$ over $\mathbb{Q}$;
if the check fails **then goto** Step 5;
4.3: **for** $i = 1, ..., s$ **check parallelly**
$G'_i = u'_{i,1}F_1 + \cdots u'_{i,r}F_r$ over $\mathbb{Q}$;
if the check fails **then goto** Step 5;
return $\Gamma_{(P)}$;
Step 5 [continue the computation]:
$k := k + 1$ and **goto** Step 1.

(*1) 素数がラッキーであれば，異なる素数に対する計算はほぼ同時に終了する．したがって，どれかの素数に対する計算が長引いていればアンラッキーとして捨てればよい．

(*2) 頭項ヒストリに対しても記号 ▷ を用いるが，これはヒストリの先頭から順に，収納されている頭べき積を ▷ で比較して順序をつけるものとする．

2.1.4 計算効率に関する考察

アンラッキーな素数の個数は有限であり，一語長程度の素数がアンラッキーになる確率は小さいから，k 個の素数 p_1,...,p_k がすべてアンラッキーとなり，しかもそれらの頭項ヒストリが同じとなる確率は極度に小さいと言える．したがって，Step 4.2 をパスした段階でほとんどの場合に正しい基底が得られると言える．一方，以下のタイミング・データが示すように，計算時間の大部分は Step 1 で消費され，しかも Step 1

では$\Gamma^{(l)}$の計算よりも$\Upsilon^{(l)}$の計算の方により多くの時間がかかる．したがって，前節に与えた算法は極度に小さい確率でしか起こり得ない場合のために過半数の時間を消費するものである．

そこで，以下の確率的算法を考える．

モジュラー・グレブナー基底算法（確率的バージョン）

前節のモジュラー・グレブナー基底算法において，

1） $\Upsilon^{(l)}=\{u_{ij}^{(l)} \mid i=1,\ldots,\ s,\ j=1,\ldots,\ r\}$ の計算を省略;

2） Step 4.3 をスキップする．

確率的バージョンは，厳密に正しいグレブナー基底を計算するときにはもちろん使えないが，以下の理由により，連立代数方程式に対しては有用である．

（i） Step 4.1 および Step 4.2 のチェックをパスしたとき，$\Gamma_{(P)}=\{G'_1,\ldots,G'_s\}$ は $(F_1,\ldots,F_r)\subseteq(G'_1,\ldots,G'_s)$ を満たす．したがって，$\{G'_1=0,\ldots,G'_s=0\}$ の解はすべて $\{F_1=0,\ldots,F_r=0\}$ の解となる．ほとんどの場合 $\Gamma_{(P)}=\Gamma$ であるから，欲しい解はほとんどの場合 $\Gamma_{(P)}$ から得られる．

（ii）連立代数方程式においては，元の方程式から解の個数の上限を容易に決めることができる（いわゆるベズーの上限．実例については [37] を見よ）．もしもこの上限が $\{G'_1=0,\ldots,G'_s=0\}$ の根の個数に等しいならば，方程式（2.11）の解はすべて $\Gamma_{(P)}$ から得られる．なお，重根がある場合，根の個数は多重度を含めて勘定しなければならないからやっかいだが，重根がない場合，（必要ならば適当に線形変数変換をすることにより）連立方程式（2.11）は（2.13）の形に変換できることが示されており（証明については [37] を見よ），根の個数とは $G_1(x_1)$ の次数に他ならない．

連立方程式（2.11）が（2.13）の形に変換できる場合には，さらに以下の三つの効率的手法が適用できる．

（a）項順序 $\triangleright$ として，辞書式順序でなく，

$$x_2,\ldots,\ x_n \triangleright x_1, \text{および} \{x_2,\ldots,\ x_n\} \text{に対しては全次数順序} \tag{2.20}$$

を選ぶことができる（証明については [64] を見よ）．理論解析 [19] および経験によれば，辞書式順序に比べて全次数順序は計算量が一般にはるかに少ない．

（b）Step 4.1 は実行しなくてよい．これはつぎの定理による．

[定理] $\mathrm{hpp}(F_i)$ と $\mathrm{hpp}(F_j)$ が共通因子をもたないとき，$\mathrm{Spol}(F_i,F_j)\rightarrow\!\!\rightarrow_{F_i,F_j} 0$.

（c）Step 4.2 は代入操作で置き換えることができる．M-簡約を繰り返すより，代入の方がより効率的である．

実際的問題で並列化が極めて有効であることを示そう．前節に述べた並列算法は並列計算機上でテストされた訳ではないが，我々はモジュラー・グレブナー基底算法を

数式処理システム GAL にインプリメントし，逐次計算機上でテストした．我々の並列算法はモジュラー型なので，その有効性は逐次計算機上のタイミング・データから十分に立証できる．テスト問題は，既にベンチマーク問題となったといえる，桂のスピングラス代数方程式である [32]

例 1 (Katsura's equation)
Katsura's equation は Katsura(n) とパラメータを持つ $n+1$ 変数の連立方程式系である．全次数逆辞書式グレブナー基底の計算量が n に対して指数的となることが知られている．ここで例としたものは Katsura(4)[2)]である．

$$\begin{aligned}
P_1 &= 2(x_5^2 + x_4^2 + x_3^2 + x_2^2) + x_1^2 - x_1 = 0,\\
P_2 &= 2(x_5x_4 + x_4x_3 + x_3x_2 + x_2x_1) - x_2 = 0,\\
P_3 &= 2(x_5x_3 + x_4x_2 + x_3x_1) + x_2^2 - x_3 = 0,\\
P_4 &= 2(x_5x_2 + x_4x_1 + x_3x_2) - x_4 = 0,\\
P_5 &= 2(x_5 + x_4 + x_3 + x_2) + x_1 - 1 = 0.
\end{aligned}$$

この連立方程式では，ベズーの上限は $2\times2\times2\times2\times1 = 16$ であり，辞書式順序に関するグレブナー基底は（2.13）の形で，degree(G_1)= 16 となる．また，結果の式の係数は，分子と分母がそれぞれ 40 桁程度以下の有理数となる．上記の例に対し，項順序 $\triangleright$ として（2.20）をとり，$\Upsilon^{(l)}$ の計算を省略し，上記（b）と（c）の手法を用いてグレブナー基底を計算した．計算には大型機で約 26 秒強（逐次計算機であることに注意）を要したが，その内訳は以下の通りであった．

Step 1: 大きさ〜5×10^5 の素数を 16 個使用し，全体で約 26 秒.
Step 3: 全体で約 252 ミリ秒.
その他: 無視できる程度.

Step 3 の計算量が少ないのは [66] で与えられた効率的方法のせいであり，2.1.3 節の算法を各多項式に対して逐次的に素直に計算すると約 2 秒を要する．Step 1 においては，用いた 16 個の素数はすべてラッキーであり，各素数に対するイデアル基底の計算は約 1.63 秒で終了した．したがって，並列化の効果は絶大であるといえる．実際，並列プロセッサ 10 個の計算機を用いれば，上記問題は約 3.3 秒で答えが得られるはずである．

[2)]2005 年現在ではこの問題は $n = 10$ 以上に対して計算されているが，モジュラー計算を利用しようと試みた 1988 年当時では $n = 5$ でもまだ難しい問題のひとつであった．

2.1.5 擬似並列算法

2.1.4 節で与えた例題をモジュラー算でなく有理数算で，同じ条件で計算したところ，10 分の制限時間内には計算が終了しなかった．このことから，係数成長のすさまじさが理解できよう．したがって，我々のモジュラー算法は，たとえ逐次的に計算したとしても，大きな問題に対しては実際的に非常に有効な方法であるといえる．しかも，並列算法の実行に必要な計算機は最も簡単な構造— いわゆるモジュラー型並列計算機—でよい．

残念ながら，我々は現在，上記の並列算法を実行しうる並列計算機をもたない．そこで以下では，並列計算機によらないで，逐次計算機上でのグレブナー基底の計算を効率化する擬似並列算法を提示する．

モジュラー・グレブナー基底算法を逐次計算機上で実行する場合に，ネックとなるのは多数個の素数に対しグレブナー基底を計算することである．ところで，素数 p_i と p_j が共にラッキーであるとき，$\Gamma^{(i)}$ と $\Gamma^{(j)}$ の計算は全く同じ手順となり，異なるのは多項式の係数の値だけである．しかも，一語長程度の素数がアンラッキーとなることは稀である．そこで，多項式 F の係数を長さ ν のリストとし，

$$F = (c_{11}\ c_{21}\ \ldots\ c_{\nu 1})T_1 + (c_{12}\ c_{22}\ \ldots\ c_{\nu 2})T_2 + \cdots \tag{2.21}$$

と表現する．ここに，$T_1, T_2, \ldots$ は係数が 1 の単項式，c_{ij} は素数 p_i に対する T_j の係数である．そして，S-多項式は M-既約にしたあと，頭係数を常に 1 に規格化する．こうしておけば，M-簡約において係数演算が複雑になることはなく，異なる素数に対する M-簡約が全く同じ手順となる．さらに計算途上で，ある素数 p_j が他の素数に比べてアンラッキーであることが判明すれば，その時点で p_j に対する係数を **nil** とおき，**nil** とされた係数に対する演算は以後スキップする．このことにより，異なる素数に対する計算の同期終了性が保証される．以上より，表現（2.21）を用いて，ひとつのグレブナー基底の計算で同時に（アンラッキーの場合も含めて）ν 個分のグレブナー基底が計算できる．素数 p を法とするモジュラー演算では，グレブナー基底の計算に占める係数演算の割合は少ないから，上記の工夫は大きな効率向上をもたらすと期待できる．

2.2 実線形分離写像による零次元方程式の解法

n 個の変数 $\{x_1, x_2, ..., x_n\}$ に関する s 個の連立代数方程式 $f_1 = 0, ..., f_s = 0$ の実解を所望の絶対誤差で求めることを目的とする．一般に連立方程式の実解を求めることは $\mathbf{x} = \{x_1, ..., x_n\}$ を n 個の不定元の集合とし，$f_1, ..., f_s \in \mathbb{Q}[\mathbf{x}]$ としたとき，$\mathrm{ideal}(\{f_1, ..., f_s\})$ の $\mathbb{R}^n$ での零点を求めることである．ここで，$\mathrm{ideal}(\{f_1, ..., f_s\})$ は零次元，すなわち解の個数は有限個であることを仮定する．

2.2.1　従来の解法

代数的方法を基礎とした連立代数方程式の解法としては，消去法を用いて与方程式を $g_n(x_n) = 0, ..., g_2(x_2, ..., x_n) = 0, g_1(x_1, x_2, ..., x_n) = 0$ の形に三角化し，$g_n(x_n) = 0$, $g_{n-1}(x_{n-1}, x_n) = 0$, ..., $g_1(x_1, x_2, ..., x_n) = 0$ を逆代入の方法で，順次 x_n, x_{n-1}, ..., x_1 について解くという方法は自然である．

しかし，この方法では，誤差の累積が生じることが問題となる．誤差の累積を避ける方法として，shape lemma[9] や generalized shape lemma （rational univariate representation）による方法 [2, 49] が開発された．これらは，零次元かつ根基イデアルを生成する方程式の場合に，各変数をいわゆる一般の位置にある元の多項式あるいは有理式で表現し，一般の位置にある元の値を各表現に代入することによって，解を定めるものである．誤差の累積は避けられるが，誤差を定められた範囲に押えようとするとき，代数的数の符号判定問題と同様の処理が必要になる．

これとは異なる方法として，各変数の満たすべき最小多項式から各変数についての解を求め，その解の組合せが全体の解となるか否かを判定して解を得る方法 [126] がある．これは累積誤差の問題はないが，解の組合せが多くなるという欠点をもっていた．また，解の組合せから真の解を判定する方法も問題であった．すなわち，方程式の係数や次数が大きい場合，元の方程式に解の候補値を代入して 0 に近いか否かにより真の解を判定するという単純な方法は無力である．その理由は，解の候補値は近似値，もしくは解を含む区間として扱うしかなく，代入による零判定問題が代数的数の符号判定問題としての処理が必要になるためである．浮動小数点数により近似する場合には，必要となる精度に見合った大きな表現長のデータを処理するという負担が避けられない．

本論文では，後者の枠組の中で組合せ爆発を避け，かつ少ない計算量で精度の保証も容易な方法を提示する．

2.2.2　分離写像

用語と諸定義

$I = [l, r] \subset \mathbb{R}$ を区間，すなわち, $I = \{a \mid l \leq a \leq r\} \subset \mathbb{R}$ とする． $L(I)$ と $R(I)$ でそれぞれ $l = \min I$ と $r = \max I$，すなわち区間の左端と右端とを表す． 区間 I の幅を $\mathrm{Width}(I) = r - l$ と表す．

k 個の区間の集合 $\mathbf{I} = \{I_1, I_2, ..., I_k\}, k \in \mathbb{N}$ に対して，その左端と右端とを $L(\mathbf{I}) = \min\{\bigcup_{1 \leq i \leq k} I_i\}$，と $R(\mathbf{I}) = \max\{\bigcup_{1 \leq i \leq k} I_i\}$ によって定義する．

さらに， 区間集合 $\mathbf{I}$ の *covering interval* $\bar{\mathbf{I}} = \overline{\{I_1, I_2, ..., I_k\}}$ を $\bar{\mathbf{I}} = [L(\mathbf{I}), R(\mathbf{I})]$ により定義する．また， $\mathbf{I}$ の *span* を $\mathrm{Span}(\mathbf{I}) = \mathrm{Width}(\bar{\mathbf{I}}) = R(\mathbf{I}) - L(\mathbf{I})$ により定義する．

また，記号の簡略のために，$\mathbf{I}$，$\mathbf{J}$ を区間の有限集合とするとき，その「組合せ直積（pairing direct product）」（と仮に呼ぶ）を $\mathbf{I} \bowtie \mathbf{J} = \{I \times J | I \in \mathbf{I}, J \in \mathbf{J}\}$ と書くことにする．これは，$\mathbf{I}$ と $\mathbf{J}$ の各要素区間のペアごとに直積，すなわち長方形の領域，を作ったとき，それらの長方形領域の全てからなる集合である．

ふたつの区間 I と J とは $I \cap J = \emptyset$ のとき *disjoint* であると言い，そうでないとき *overlapped* であると言う．

ある区間 I は $R(I) < L(J)$ であるとき 区間 J の 左にあるという． I が J の左にあるとき J は I の右にあるという．この状況を $I < J$ あるいは $J > I$ と書く．

この関係 $<$ はつぎのように区間の集合に対しても拡張して用いる．すなわち，区間のふたつの集合 $\mathbf{I} = \{I_1, ..., I_r\}$ と $\mathbf{J} = \{J_1, ..., J_r\}$ とに対して，「$\mathbf{I} < \mathbf{J}$ とは，任意の $I \in \mathbf{I}$ と $J \in \mathbf{J}$ とに対して $I < J$ となること」と定義する．

ふたつの disjoint な区間 I と J とについてその，間隙 $\mathrm{Gap}(I, J)$ を

$$\begin{aligned} \mathrm{Gap}(I, J) &= L(J) - R(I) \text{ if } I < J \\ &= L(I) - R(J) \text{ if } J < I \\ &= \text{undefined otherwise} \end{aligned}$$

により定義する．

2 次元の場合

まず，2 変数の場合を考察する．

パラメータ $0 < \alpha \in \mathbb{R}$ をもつ線形写像 $\psi(x, y) = x + \alpha y$，$\psi : \mathbb{R}^2 \to \mathbb{R}$ を考える．

$X \subset \mathbb{R}$ と $Y \subset \mathbb{R}$ とを区間とするとき, これらの直積 $X \times Y \subset \mathbb{R}^2$ は長方形領域となる．ここで α が正であることから ψ は各変数について単調増加な写像である．よって，ψ による長方形領域の像は左端と右端とがそれぞれ $\psi(L(X), L(Y))$ および $\psi(R(X), R(Y))$ となる区間となる．すなわち，

$$\psi(X \times Y) = [\psi(L(X), L(Y)), \psi(R(X), R(Y))] \subset \mathbb{R}$$

は 1 次元の区間である．

$\mathbf{X} = \{X_1, ..., X_m\}$ と $\mathbf{Y} = \{Y_1, ..., Y_n\}$ とを disjoint なふたつの区間とし， $X_1 < X_2 < \cdots < X_m$ および $Y_1 < Y_2 < \cdots < Y_n$ のように配列されているものとしよう．

このとき，

$$\begin{aligned} S_x &= \mathrm{Span}(\mathbf{X}), \\ W_y &= \max_{Y \in \mathbf{Y}} \mathrm{Width}(Y), \\ G_x &= \min_{X, X' \in \mathbf{X}} \mathrm{Gap}(X, X'), \\ G_y &= \min_{Y, Y' \in \mathbf{Y}} \mathrm{Gap}(Y, Y') \end{aligned}$$

とおくとつぎの命題が成り立つ.

定理 1
パラメータ $\alpha > 0$ が条件

$$\frac{S_x}{G_y} < \alpha < \frac{G_x}{W_y} \tag{2.22}$$

を満たすとき，すべての長方形領域は $m \times n$ 個の disjoint な区間に写像され，しかもつぎのように配列されている.

$$\begin{array}{ccccccccc} & \psi(X_1 \times Y_1) & < & \psi(X_2 \times Y_1) & < & \cdots & < & \psi(X_m \times Y_1) \\ < & \psi(X_1 \times Y_2) & < & \psi(X_2 \times Y_2) & < & \cdots & < & \psi(X_m \times Y_2) \\ & \cdots & & \cdots & & \cdots & & \cdots \\ < & \psi(X_1 \times Y_n) & < & \psi(X_2 \times Y_n) & < & \cdots & < & \psi(X_m \times Y_n). \end{array} \tag{2.23}$$

証明
最初に，各 $j\,(1 \le j \le n)$ に対して，パラメータ $\alpha > 0$ が条件 （2.22）の右側の不等式を満たす，すなわち，

$$0 < \alpha < \frac{G_x}{W_y} \tag{2.24}$$

ならば

$$\psi(X_1 \times Y_j) < \psi(X_2 \times Y_j) < \cdots < \psi(X_m \times Y_j) \tag{2.25}$$

となることを示す.

不等式（2.24）が成立しているものとする．一対の水平に相隣り合う長方形領域がその位置に関する順序関係を保存すること，すなわち，

$$R(\psi(X_i \times Y_j)) < L(\psi(X_{i+1} \times Y_j)) \quad (1 \le i < m) \tag{2.26}$$

であることを示したい．これを示すために目的の不等式（2.26）の右辺から左辺を引いて差を計算する.

$$\begin{aligned} G &= L(\psi(X_{i+1} \times Y_j)) - R(\psi(X_i \times Y_j)) && (2.27)\\ &= \psi(L(X_{i+1}), L(Y_j)) - \psi(R(X_i), R(Y_j)) && (2.28)\\ &= (L(X_{i+1}) + \alpha \cdot L(Y_j)) - R(X_i) + \alpha \cdot R(Y_j) && (2.29)\\ &= (L(X_{i+1}) - R(X_i)) - \alpha \cdot (R(Y_j) - L(Y_j)) && (2.30)\\ &= \mathrm{Gap}(X_i, X_{i+1}) - \alpha \cdot \mathrm{Width}(Y_j). && (2.31) \end{aligned}$$

ここで $\mathrm{Gap}(X_i, X_{i+1}) \ge G_x > 0$ と $\mathrm{Width}(Y_j) \le W_y$ とから

$$\begin{aligned} G &\ge G_x - \alpha \cdot W_y && (2.32)\\ &> 0 && (2.33) \end{aligned}$$

を得る．最後の不等号は仮定した条件（2.24）による．これにより，水平方向に配列された長方形領域の像について目的とする不等式が成立することは証明された．

命題の証明を完成するには，ある水平方向に配列された長方形領域の行の最も右側の領域が，その直上に配列された長方形領域の行の最も左側よりも左に写像されることを示せばよい．

パラメータ $\alpha > 0$ が条件（2.22）の左側の不等式を満たす，すなわち，

$$\frac{S_x}{G_y} < \alpha \tag{2.34}$$

とする．すると任意の $1 \leq j < n$ となる j について，$R(\psi(X_m \times Y_j)) < L(\psi(X_1 \times Y_{j+1}))$ を示せばよい．

前と同様右辺から左辺を引いて差を計算する．

$$\begin{aligned} G' &= L(\psi(X_1 \times Y_{j+1})) - R(\psi(X_m \times Y_j)) && (2.35)\\ &= \psi(L(X), L(Y_{j+1})) - \psi(R(X), R(Y_j)) && (2.36)\\ &= (L(X) + \alpha \cdot L(Y_{j+1})) - R(X) + \alpha \cdot R(Y_j) && (2.37)\\ &= \alpha \cdot (L(Y_{j+1}) - R(Y_j)) - (R(X) - L(X)) && (2.38)\\ &= \alpha \cdot \mathrm{Gap}(Y_j, Y_{j+1}) - S_x. && (2.39) \end{aligned}$$

ここで，$\mathrm{Gap}(Y_j, Y_{j+1}) \geq G_y > 0$ であるから，

$$\begin{aligned} G' &\geq \alpha \cdot G_y - S_x && (2.40)\\ &> 0. && (2.41) \end{aligned}$$

最後の不等号は仮定した条件（2.34）による．

これにより命題は証明された． ∎

注意 1

証明の中に現れた G および G' はいずれも，像の空間での区間の Gap であることを注意しておく．

3 次元以上の場合

3 変数以上の場合にはつぎのように，再帰的に分離写像を定義する．

変数の個数を N とし，各変数を順に $x_1, x_2, ..., x_N$ とする．各変数についての区間の集合をそれぞれ順に $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N$ とする．$\mathbf{X}_i = \{X_i^{(1)} < X_i^{(2)} < \cdots < X_i^{(n_i)}\}$ $(n_i \in \mathbb{N}, i = 1, 2, ..., N)$ である．

まず，$i = 2, 3, ..., N$ について，パラメータ $\alpha_i > 0$ $(i = 2, 3, ..., N)$ をもつ線形写像 ψ_i を

$$\psi_i(x, y) = x + \alpha_i y \tag{2.42}$$

とする．つぎに，$i = 1, 2, ..., N$ に対する新しい変数 u_i $(i = 1, 2, ..., N)$ をつぎのように導入する．

$$u_1 = x_1, \tag{2.43}$$

$$u_i = \psi_i(u_{i-1}, x_i). \tag{2.44}$$

新しい変数 u_i についての区間集合はつぎのように定義する．

$$\mathbf{U}_1 = \mathbf{X}_1, \tag{2.45}$$

$$\mathbf{U}_i = \psi_i(\mathbf{U}_{i-1} \bowtie \mathbf{X}_i). \tag{2.46}$$

ここで，組合せ直積 $\mathbf{X} \bowtie \mathbf{Y}$ に ψ を作用させた結果は

$$\psi(\mathbf{X} \bowtie \mathbf{Y}) = \{\psi(X \times Y) | X \in \mathbf{X}, Y \in \mathbf{Y}\}$$

である．

この区間集合をもつ新しい各変数 u_i について区間集合の Span および最小の Gap を

$$S_{u_i} = \mathrm{Span}(\mathbf{U}_i), \tag{2.47}$$

$$G_{u_i} = \min_{U, U' \in \mathbf{U}_i} \mathrm{Gap}(U, U') \tag{2.48}$$

と書くことにする．

ここで，各 α_i $(i = 2, 3, ..., N)$ について

$$\frac{S_{u_{i-i}}}{G_{x_i}} < \alpha_i < \frac{G_{u_{i-1}}}{W_{x_i}} \tag{2.49}$$

が満たされるならば，2 次元の場合の**命題** 1 により，上記，S_{u_i} および G_{u_i} は，それぞれ次の式により逐次計算できることが容易に分かる．（G_{u_i} については，**注意** 1 を参照.）

$$S_{u_i} = S_{u_{i-1}} + \alpha_i S_{x_i}, \tag{2.50}$$

$$G_{u_i} = \min\{G_{u_{i-1}} - \alpha_i W_{x_i}, \alpha_i G_{x_i} - S_{u_{i-1}}\}. \tag{2.51}$$

ここに，$i = 1, 2, ..., N$ について，

$$G_{x_i} = \min_{X, X' \in \mathbf{X}_i} \mathrm{Gap}(X, X'), \tag{2.52}$$

$$W_{x_i} = \max_{X \in \mathbf{X}_i} \mathrm{Width}(X) \tag{2.53}$$

である．

全ての i $(i = 2, 3, ..., N)$ について α_i についての条件 （2.49）が満たされているとき，$(x_1, x_2, ..., x_N) \mapsto U_N = \Psi(x_1, x_2, ..., x_N) = x_1 + \alpha_2 x_2 + \cdots \alpha_N x_N$ によって線形写像 $\Psi : \mathbb{R}^N \to \mathbb{R}$ が構成できる．

この写像 Ψ を N 次元の分離写像，条件式 （2.49）を分離係数条件，と呼ぶことにする．

条件式（2.49）が成立するような α_i がとれるとき, 分離写像 Ψ はその構成法から, R^N に格子状に配列された $n_1 \times n_2 \times \cdots \times n_N$ 個の超直方体領域 $X_1^{(j_1)} \times X_2^{(j_2)} \times \cdots \times X_N^{(j_N)}$ を, その添字の辞書式順序で配列される重なりのない 1 次元の区間達 $\Psi(X_1^{(j_1)} \times X_2^{(j_2)} \times \cdots \times X_N^{(j_N)})$ に写像する．

2.2.3 連立方程式の解

連立方程式の解は各変数の満たす最小多項式の解の組合せの中に存在する．特に，実解の組合せのうち，どの組み合わせが真の実解を与えるかを判定するために前項の分離写像が利用できる．

基本的事項

零次元イデアルを生成する連立方程式の（有限個の）解の個数がそのイデアルの $\mathbb{Q}$-線形次元に一致することは良く知られている．

特に零次元根基イデアルについては次の Shape Lemma [9] が成立する．

定理 2 (Shape Lemma)
$\mathcal{I} \subset \mathbb{Q}[x_1, ..., x_N]$ を根基イデアルとする．有限個の例外を除く整数の組み $(a_1, a_2, ..., a_N)$ に対して，$\mathcal{I} \cup \{u - (a_1 x_1 + a_2 x_2 + \cdots + a_N x_N)\}$ の変数順序を $\{x_1, ..., x_N\} > u$ とした項順序でのグレブナ基底はつぎの形をしている．

$$\{x_1 - g_1, x_2 - g_2, ..., x_N - g_N, h\}.$$

ここに，$h, g_i \in \mathbb{Q}[u]$ $(i = 1, 2, ..., N)$，すなわちこれらは u の一変数多項式である．

この定理により，零次元根基イデアルについてはその零点 $(\xi_1^{(j_1)}, \xi_2^{(j_2)}, ..., \xi_N^{(j_N)}) \in \mathbb{C}^N$ が h の零点 $v^{(j_1, j_2, ..., j_N)} \in \mathbb{C}$ に 1 対 1 対応する．この定理を満たすような元 $a_1 x_1 + a_2 x_2 + \cdots + a_N x_N$ あるいは u を「複素数上一般の位置にある元」，あるいは，「複素分離元」と通例とは違うが「複素」を付して呼ぶ．h は u の最小多項式である．

$u = a_1 x_1 + a_2 x_2 + \cdots + a_N x_N$ が複素分離元であるか否かはつぎのよく知られた命題により判別できる．

命題 3 (複素分離元の判定)
定理 2 と同様の記号と条件の下で，u が複素分離元 $\Leftrightarrow \deg h = \dim_{\mathbb{Q}} \mathcal{I}$.

実解の判定

方程式を構成する多項式の生成するイデアルを $\mathcal{I}$ とする．今, 変数 x_i $(i = 1, 2, ..., N)$ の各々についての実解の区間のすべてが $\Xi_i^{(1)}, ..., \Xi_i^{(t_i)}$ と計算されており，分離係数条件が成立して分離写像 Ψ が構成され，u_N の最小多項式 $h_N(u_N)$ も計算されているものとする．

繁雑さを避けるため，区間の組 $(\Xi_1^{(j_1)}, \Xi_2^{(j_2)}, ..., \Xi_N^{(j_N)}) \in (\mathbb{R}^2)^N$ を指定するのに，単に添字の組 $(j_1, j_2, ..., j_N)$ を用いることにする．

イデアル $\mathcal{I}$ が零次元かつ根基である場合には，$\mathbb{C}^N$ における $\mathcal{I}$ の真の零点はすべて単純（重複なし) である．さらに，定理 2 により，真の零点は $h_N(u_N)$ の零点と 1 対 1 に対応するし，しかも，容易に分かるように，分離係数 α_i が複素分離元を与える，すなわち（命題 3）が成立している場合には，$\mathcal{I}$ の実零点も u の実零点に 1 対 1 に対応する[3)].

これにより次の命題を得る．

命題 4
$\mathcal{I}$ が零次元かつ根基であるとする．さらに，分離係数 α_i 達が複素分離元を与えるものとする．すると，組 $(j_1, j_2, ..., j_N)$ が $\mathcal{I}$ の実零点であることの必要十分条件は，$h_N(u_N)$ のある零点が区間 $\Psi(j_1, j_2, ..., j_N)$ すなわち，$\Psi(\Xi_1^{(j_1)}, \Xi_2^{(j_2)}, ..., \Xi_N^{(j_N)})$ に落ちる（含まれる）ことである．

$h_N(u_N)$ の実零点は精度保証さえあれば数値的に求めても良いが，必ずしも計算量的に有利とは即断できない．本論文では， $h_N(u_N)$ の零点も区間で求めることになるのでつぎの命題のほうがより実際的である．

命題 5
命題 4 と同様 $\mathcal{I}$ が零次元かつ根基であるとする．さらに，分離係数 α_i 達が複素分離元を与えるものとする．すると，組 $(j_1, j_2, ..., j_N)$ が $\mathcal{I}$ の実零点であることの必要十分条件は，$h_N(u_N)$ の零点を唯一含むある区間 v が存在し，$\Psi(j_1, j_2, ..., j_N)$ と共通部分をもち，かつ，これ以外の組み $(j'_1, j'_2, ..., j'_N)$ の像 $\Psi(j'_1, j'_2, ..., j'_N)$ とは共通部分を持たないことである．

[3)] 分離係数条件が成立していても，$\mathcal{I}$ の複素零点が u の実零点に対応する場合がある．イデアルが根基ならば，そのような事態が生じているか否かを確かめることは命題 3 により可能で，生じている場合には α_i を変更すればよい．

もし，υ の区間幅を Ψ による超直方体達の像 $\mathbf{U}_N$ の最小の区間の間隙 G_{u_N} より小さく取るならば，$L(\upsilon)$ か $R(\upsilon)$ のどちらか一方がある像 $\Psi(j_1, j_2, ..., j_N)$ に含まれることを確認するだけでもよい．

分離係数条件

ここで分離係数 α_i の存在について吟味しよう．

方程式の実解を求める目的で分離写像を使う際には，各変数 x_i についての区間 $\mathbf{X}_i$ とは各変数の満たす最小多項式 $\phi_i(x_i)$ の実根を唯一含む区間のことになる．今，ψ_i を作る，すなわち α_i を決める段階を考えよう．

まず，式（2.49）に現われる区間の幅 W_{x_i} は，例えば Sturm 列を使う方法などによって，必要に応じていくらでも小さく取ることができることは明らかである．

一方，式（2.49）の G_{x_i} は 根を含む区間の間の間隙の最小値であるから W_{x_i} を小さくとるとき増加して行き，変数 x_i の最小多項式 $\phi_i(x_i)$ の根の差の最小値に近付く．

このとき，式（2.49）の $S_{u_{i-1}}$ と $G_{u_{i-1}}$ は既に定まっているままとし，遡って変更することはない（つまり $W_{x_{i-1}}$ を小さく取り直すことをしない）とすれば，式（2.49）の左辺は正の一定値に近付き，右辺は任意に大きくできることが分かる．

このことから，x_i の最小多項式 ϕ_i の根の存在区間を必要に応じて小さくとることで，式（2.49）を満足する α_i をいつでも，特に正整数の中から，見付けるようにすることが可能である．

さらに，そのような正整数 α_i で複素分離元を与えないものは定理 2 により高々有限個であるので，必要に応じて $W_{x_{i-1}}$ を小さく取ることにより，分離写像を与えかつ複素分離元を与える 分離係数 α_i を見付けることができる．

2.2.4 アルゴリズム

これまで考察したことがらを用いて零次元根基イデアルの実零点を求めるアルゴリズムが構築できる．

アルゴリズム 2

入力: 多項式の集合 $\mathcal{F} = \{f_1, f_2, ..., f_s\}$，および区間幅の上限 ε．ただし ideal($\mathcal{F}$) は零次元根基イデアル．

出力: 実零点を丁度 1 個含む $\mathbb{R}^N$ の超直方体 $\{(\Xi_1^{(j_1)}, \Xi_2^{(j_2)}, ..., \Xi_n^{(j_N)})\}$ の集合．

Step 1 $\mathcal{F}$ のグレブナ基底 $G = \mathrm{GB}(\mathcal{F})$ を求める．

Step 2 G を用い，各 i について x_i の最小多項式 $\phi_i(x_i)$ を求める．

Step 3 各 i について $\phi_i(x_i)$ の実根を精度 ε 以下の区間幅で求める．

Step 4 分離係数条件（2.49）が成立するように分離写像 Ψ を求める．この際，必要であれば精度をあげるために Step 3 に戻る．

Step 5 $u = \Psi(x_1, x_2, ..., x_N)$ とし，u_N の最小多項式 $h_N(u_N)$ を求める．

Step 6 $h_N(x_N)$ の実解を区間で求め，解の判定により （Step 3）で求めた各変数の根区間のどの組合せが真の根になっているか決定する．

注意 2

（1） $\mathcal{I} = \text{ideal}(\mathcal{F})$ が零次元かどうかは （Step 1） の結果のグレブナ基底 G により判定できる．また，$1 \in G$ の場合は「解なし」と決定できる．

（2）（Step 2）で得られる各変数に対する最小多項式 $\phi_i(x_i)$ の既約因子を $g_i(x_i)$ としたとき，多項式集合 $G \cup \{g_1(x_1), g_2(x_2), ..., g_N(x_N)\}$ が生成するイデアルは素イデアルで，特に根基となる．これらの組合せのすべてを改めてアルゴリズムの入力としてやれば，入力イデアルが根基でない場合にも対応できる．

（3）さらに，G を用いて $\dim_{\mathbb{Q}} \mathcal{I}$ が容易に分かるので，（Step 5）で計算する h_N の次数がこの $\dim_{\mathbb{Q}} \mathcal{I}$ に一致するかどうかで，分離元であることの確認ができる．そうでなかった場合にも （Step 4） に戻ればよい．

2.2.5 実例

表 2.2.5 に実行例のタイミングデータを示す．

表 2.1: Katsura-N （N=4,5,6）のタイミングデータ

許容誤差 $< 10^{-50}$, 時間単位: CPU（秒）GC（秒）

	Katsura 4	Katsura 5	Katsura 6
実解個数/全解個数	12/16	16/32	32/64
Gröbner 基底	0.024+0.024	0.271+0.224	2.96+2.38
各変数の最小多項式	0.138+0.087	1.277+0.776	11.89+5.74
各変数の求解	0.450+0.038	2.660+1.487	16.98+8.90
分離写像の構成	0.044+0.038	0.079+0.042	0.176+0.057
像の最小多項式	0.101+0.053	1.325+0.515	23.10+4.36
解の判定	0.226+0.156	1.325+0.515	10.08+3.89
全計算時間	1.023+0.599	7.012+3.847	65.19+25.33

この実行例に用いた計算機環境はつぎのとおりである．
計算機: FreeBSD 4.2/Pentium III 800MHz/256MBytes Memory
ソフト: asir2000
1 変数多項式の求解アルゴリズム: 平野のアルゴリズム（文献 [120]: 2.3.5 項）．Hilano's Code. *cf.* Hilano, T.: Finding Real Zeros of Polynomial with Rational Coefficients.
上記タイミングデータ採取時の分離写像は表 2.2.5 のとおりである．

表 2.2: Katsura-N （N=4,5,6）の分離写像

	分離写像（Psi）
Katsura-4	$u_0 + 222930u_4 + 50268452u_3 + 16552394309u_2 + 5255214791142u_1$
Katsura-5	$u_0 + 1689u_5 + 1798378u_4 + 2438193864u_3$ $+ 686932631846u_2 + 8192041686253927u_1$
Katsura-6	$u_0 + 16122u_6 + 692801818u_5 + 1907509944054u_4$ $+ 3160345973524626u_3 + 57345894956819556952u_2$ $+ 45551156332495524109086 0u_1$

2.2.6 考察

分離写像を用いて連立代数方程式の実解を指定された絶対誤差の範囲で求めるアルゴリズムを提案してきた．今回は根の判定を改良し，1 変数多項式の根の分離および精密化に Hilano の Code を用いて高速化を行った．厳密には分離写像が複素分離元を与えない場合のチェックが必要であるが，プログラムには未反映である．根の判定や精度の上げ方についてはまだいくつかの代替案が考えられる．これらについては今後検討したい．

2.3 新しい話題へ—限定子除去法（QE)

2.3.1 限定子除去法

グレブナー基底は等式に関する制約問題について明快な解決を与える万能薬であると言えるが，工学や数学教育への応用を意識した場合，不等式による実数上の制約問題が等式制約より広範で一般的な問題として残っている．

限定子除去法[4]は 制約問題の実数性を陽に取り扱うことができる理論として開拓され，Tarski による 1930 年代の成果 [78] に始まると言われている．

それは，実数を解釈のドメインとする変数（論理変数ではなく論理式中に現われる函数の変数）のみを持ち，2 項関係記号として「等価」(“=”),「より小さいい」(“<”) のみが許される原子論理式から構成される一階術語論理の論理式クラスにおいて，束縛変数を持つ論理式から，束縛記号とそれにより束縛されている変数を除去して，自由変数のみをもつ（すなわち束縛記号（= 限定子）を持たない）等価な論理式を得る方法を与えるものである．

なじみのある簡単な例としては，

$$\exists x.(x^2 + bx + c < 0) \tag{2.54}$$

という論理式と等価で，束縛変数 x を除去し，自由変数 b, c のみを含む論理式は何か? というものである．この答えは，判別式の性質として良く知られたとおり，

$$b^2 - 4c > 0 \tag{2.55}$$

となる．QE を適用すれば，判別式の性質を知らなくても，論理式（2.54）から論理式（2.55）を得ることができる[5]．

QE は等式ばかりでなく不等式も扱えることから，注目する変量に関するユークリッド空間の領域（Semi-algebraic Set）として結果を得ることができるため，計画問題などを含む理工学上の問題に対して一般的に適用可能である．新しい変量を導入することによって，最適化問題（最大/最小問題）にも適用できるため，制御工学では早くから注目されていた．最近では，システム制御，最適制御ばかりでなく，バイオシミュレーションへの適用など，数値的手法との自然な混合手法としても注目されるようになった．

ここでは，ロバスト制御系の設計問題への著者による最近の結果 [76, 77] を一つ紹介する．

2.3.2　ロバスト制御系設計問題への Strelitz test の応用

安定性は制御系の基本的な問題である．その基本問題は，「所与の 1 変数多項式の根が全ての左半複素平面上にあるか否かを，事前に根を計算することなく決定すること」と数学的には記述される．この問題は Routh と Hurwitz とにより独立に解かれた．

[4] Quantifier Elimination （QE). Quantifier の邦語術語としては，限定子，限量子，量限定子，限定記号，限量記号，束縛記号など複数があり，それらに応じて，限定子除去，束縛記号除去法など数種の呼び方が使われている．

[5] 正確には，論理式（2.55）と等価な論理式のどれかであって，結果の論理式の表現は一意ではなく，実際に適用される QE の手続きに依存する．

安定性に関しては多くの研究があるが，木村等 [34, 35] は安定性関連の設計問題を含むロバスト制御系設計問題の十分大きなクラスが，定符号条件（SDC[6)]）問題として定式化されることを示し，ユークリッド多項式剰余列によって，元の問題を SDC に還元する初歩的な方法を提案した．

その後，穴井等 [6, 7] が SDC 問題を特殊 QE 問題として定式化し，Sturm-Habicht 列を用いることで，SDC 問題がより効率的に解けることを示した．後者の方法は，設計パラメータの特殊化によって，ユークリッド多項式剰余列が異常となるような例外的な場合にも適用可能となる点が優れている．

この節では，典型的な問題として上記の 2 つの論文で取り扱われている D 安定性に関して，SDC 帰着を用いずにより簡潔で効率的な定式化を論じる．D 安定性設計問題は，Strelitz により提案された**根和多項式** [69] によって，安定多項式問題として直接に解くことが可能であることを示す．代数的計算の立場からはここで呈示する方法が SDC 帰着の方法よりも優れている．

2.3.3 根和多項式による安定多項式判定

安定多項式

実係数の 1 変数多項式はその根がすべて左半複素平面上にあるとき安定多項式と呼ばれる．Routh-Hurwitz 判定法は多項式の安定性を判定する良く知られた決定手続きである．一方 Strelitz テストは，記号係数をもつ多項式に対してより効率のよい計算法を与えると言う意味で，代数的計算の観点から良い性質を持つ安定判定法である．

注意 3
Routh-Huruwitz 判定法は根の個数を計数する手続きと同等の手続き—修正された多項式剰余列あるいは部分終結式列—によって計算 される．一方，Strelitz テストは根の個数を数えようとはしない．安定性のみを決定しよとする場合 Routh-Huruwitz 判定法は過剰品質であると考えられる．

根和多項式

Strelitz テストの鍵となる概念 **根和多項式**を定義しよう．

$f \in \mathbb{R}[z]$ を正の次数 n をもつモニックな 1 変数多項式であるとしよう．さらに，$\{\alpha_i\}_{i=1,\ldots,n}$ を f の重複を許した全ての根から成る集合[7)]とする．

[6)] Sign Definite Condition.

[7)] 厳密に言えば，“multi-set” であるが，ここでは慣例にしたがって単に集合と呼ぶ．

多項式 f に対して，根和多項式と呼ばれる次数が $\frac{n(n-1)}{2}$ の新しいモニックな多項式 $g \in \mathbb{R}[z]$ を式（2.56）で定義する．

$$g = \prod_{1 \le i < j \le n} (z - (\alpha_i + \alpha_j)). \tag{2.56}$$

どの根 $\alpha_i + \alpha_j$ もそれが虚根である場合には，$\{\alpha_i + \alpha_j\}_{1 \le i < j \le n}$ の中にその共役を見出すことができる[8)]ので，g のどの係数も実となることに注意する．

このように定義される g を効率良く計算するために，Strelitz は Newton-Girard 公式に基づく計算アルゴリズムを提示している．これについては後に 2.3.4 項において詳述する．

次式で定義される f のより大きな根和多項式 G を考えることもできる．

$$G = \prod_{1 \le i, j \le n} (z - (\alpha_i + \alpha_j)) = 2^n f(\frac{z}{2}) \times g(z)^2.$$

すると，終結式の定義からつぎの関係式を導くことは容易である．

$$G = \mathrm{res}_t(f(t), f(z - t)).$$

この終結式を用いて g を計算することも考えられるが，空間計算量の大きさのために，ごく小さな f，すなわち次数が小さく，係数に現れる記号パラメータ[9)]の個数も少ないような多項式に対してしか用いることはできない．

根和多項式の主要性質を述べる前に，2 つの命題を示しておこう．

命題 6
多項式 f が安定ならば多項式 g も安定である．

証明
g のどの根も，左半平面上にある f の 2 根の和であるので当然左半平面にある． ∎

命題 7
$f \in \mathbb{R}[z]$ はモニックであるとする．つぎの 2 つの命題は同時に成立しえない．

1. f のすべての係数は正である．
2. f は非負の実根をもつ．

[8)] 重複がある場合にはその個数に応じて．
[9)] 設計問題では必然的に記号パラメータをもつ多項式を扱うことになる．

証明
仮に2命題がともに成立するとしよう．すると，第2の命題により，非負実数γが存在して$f(\gamma) = 0$を満たす．しかるに，第1の命題からfの係数は正であるので，$f(\gamma)$は，非負の実数γの非負の冪にfの正の係数を乗じた項の和となり，したがって0には成り得ない．これは$f(\gamma) = 0$に矛盾する． ∎

根和多項式の主性質はつぎの定理として述べられる．

定理 8 (Strelitz)
実係数のモニックな多項式fが安定である必要十分条件は，fの係数とgの係数のすべてが正となることである．

証明
（十分条件）：fが安定であるとする．fが実根γをもてばそれは負である．よって，fは$(z-\gamma)$なる線形因子をもちその係数である1および$-\gamma$は明らかに正である．また，fが一組の共役根αと$\overline{\alpha}$をもてば，fは2次因子$(z-\alpha)\times(z-\overline{\alpha}) = z^2-(\alpha+\overline{\alpha})z+\alpha\overline{\alpha}$をもち，その係数はやはり正である．なんとなれば，$\alpha\overline{\alpha}$は常に正であり，安定な$f$に対しては$\alpha$とその共役$\overline{\alpha}$の実部はともに負であるため，$-(\alpha+\overline{\alpha})$も正となるからである．$f$は重複を込めてこのような線形あるいは2次の因子の積であるに過ぎないので，その係数はすべて必然的に正となる．命題6によりgも安定多項式であるので，同様の議論によりその係数はやはり正である．
（必要条件）：fとgの係数がすべて正であるとする．fが実根γをもつとすれば，命題7によってそれは負でなければならない．fが一組の共役根αと$\overline{\alpha}$とをもつとすれば，gは根$\alpha+\overline{\alpha}$をもち，それは実である．ここでgに対して命題7を適用すれば，$\alpha+\overline{\alpha}$も負となり，それはαの実部が負であることを意味する．したがってfのすべての根は左半平面にあることになる．よって，fは安定となり，したがって命題6によりgも安定となる． ∎

本論文では，定理8に述べられた多項式が安定であるための条件をStrelitz条件と呼び，それに基づく安定性判定手続きをStrelitzテストと呼ぶ．

注意 4
モニックな多項式$f(z)$の係数が虚数を含む場合に，その根がすべて左半平面に存在することを判定するには，$f(z)\overline{f(\overline{z})} \in \mathbb{R}[z]$に対して定理を適用すれば良い．

2.3.4 根和多項式の計算

根和多項式計算の概要

2.3.3項で定義された根和多項式の係数はfの係数について対称である．

Strelitz は g（の係数) を冪和を用いて効率良く計算するアルゴリズムを与えた．そのアルゴリズムは，多項式の根の冪和に関する Newton-Girard 公式に加えて，f の冪和と g の冪和とを関係付ける漸化式をも利用して，g の係数を f の係数から計算する．

正の次数 n と $m = \frac{n(n-1)}{2}$ に対して，モニックな多項式 f と g とをつぎの形式としよう．

$$f = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0, \tag{2.57}$$

$$g = b_m z^m + b_{m-1} z^{m-1} + \cdots + b_1 z + b_0. \tag{2.58}$$

ここで，$a_n = 1$ かつ $b_m = 1$ である．

$\{\alpha_i\}_{i=1,\ldots,n}$ を f の根全体から成る集合とし，$\{\beta_i\}_{i=1,\ldots,m} = \{\alpha_i + \alpha_j\}_{1 \le i < j \le n}$ を g の根全体からなる集合とする．ここに，$m = \frac{n(n-1)}{2}$ である．

さらに，f と g それぞれの根の m 乗までの冪和を $\{\sigma_j\}_{j=0,1,\ldots,m}$ および $\{s_j\}_{j=0,1,\ldots,m}$ とし，つぎのように置く．

$$\sigma_j = \sum_{i=1}^{n} \alpha_i^j, \ (j = 0, 1, ..., m) \tag{2.59}$$

$$s_j = \sum_{i=1}^{m} \beta_i^j, \ (j = 0, 1, ..., m). \tag{2.60}$$

そうすると，f から g を計算する手順の概要はつぎのアルゴリズム 3 に示すように 3 つのステップからなる．

アルゴリズム 3 (SORP: 根和多項式)

Input: coefficients $\{a_i\}_{i=0,\ldots,n}$ of monic polynomial $f \in \mathbb{R}[z]$;

Output: coefficients $\{b_l\}_{l=0,\ldots,m}$ of the sum-of-roots polynomial of f;

Step 1: Represent σ_j by a_i, $(i = 0, 1, ..., n)$ for $j = 0, 1, ..., m$.

Step 2: Represent s_k by a_i, $(i = 0, 1, ..., n)$, through σ_j, $(j = 0, 1, ..., m)$ which are computed in Step 1, for $k = 0, 1, ..., m$.

Step 3: Represent b_l by a_i, $(i = 0, 1, ..., n)$, through s_k, $(k = 0, 1, ..., m)$ which are computed in Step 2, for $l = m, m-1, ..., 0$.

以下，アルゴリズム 3 で必要とする漸化式を与えよう．

冪乗和と多項式の係数との関係

アルゴリズム 3 の Step 1 と Step 2 とでは良く知られた Newton-Girard 公式を用いる．まず，Step 1 に対しては a_i と σ_j の関係はつぎのとおりである．

$$\sigma_0 = n,$$

$$\begin{aligned} \sigma_1 + a_{n-1} &= 0, \\ \sigma_2 + \sigma_1 a_{n-1} + 2a_{n-2} &= 0, \\ &\cdots \\ \sigma_n + \sigma_{n-1} a_{n-1} + \sigma_{n-2} a_{n-2} + \cdots + n a_0 &= 0. \end{aligned} \tag{2.61}$$

そして，$j > n$ については

$$\sigma_j + \sigma_{j-1} a_{n-1} + \cdots + \sigma_{j-n} a_0 = 0. \tag{2.62}$$

となる．これらの公式を漸化式として用いることで，a_i の多項式である σ_j が $j = 0, 1, 2, ..., m$ と j の増加順に順次計算される．

注意 5
この手順は，除算を必要とせず，$\mathbb{Z}[a_n, ..., a_0]$ における環演算のみで計算できることに注意する．

s_k と b_l とについての Newton-Girard 公式は，式（2.61）および式（2.62）において，記号 n, a_i, σ_j をそれぞれ，記号 m, b_l, s_k に置き換えることで得られる．そして，その公式により $l = m, m-1, ..., 1, 0$ と l の減少順に b_l を s_k から計算することができる．

注意 6
この手順は，$\mathbb{Z}[s_m, ..., s_0]$ における環演算の他に，整数による整除が必要となる．さらに，次項に示す公式によって s_k が $a_n, ..., a_0$ について整であることが知られるので，結果として $\mathbb{Z}[a_n, ..., a_0]$ の環演算の他に整数による整除が必要となる．

冪乗和間の関係

Strelitz は g に対する冪和（$\{s_j\}_{j=0,1,...,m}$）と f に対する冪和（$\{\sigma_i\}_{i=0,1,...,m}$）とを関連付けるつぎの漸化式を与えた．

$$2s_j = \sum_{p=0}^{j} \left(\begin{array}{c} j \\ p \end{array} \right) \sigma_p \sigma_{j-p} - 2^j \sigma_j, \ (j = 0, 1, ..., m). \tag{2.63}$$

式（2.63）はつぎの等式から得られる．

$$\left(\sum_{k=1}^{n} e^{\alpha_k \cdot t} \right)^2 = \sum_{k=1}^{n} e^{2\alpha_k \cdot t} + \sum_{p,q=1, p \neq q}^{n} e^{(\alpha_p + \alpha_q)t}. \tag{2.64}$$

上式の両辺に現れる指数函数をテイラー展開し，t に関する同次項を集めることで，つぎの式を得る．

$$\left(\sum_{j=0}^{\infty}\frac{1}{j!}\sigma_j t^j\right)^2 = \sum_{j=0}^{\infty}\frac{2^j}{j!}\sigma_j t^j + \sum_{j=0}^{\infty}\frac{2}{j!}s_j t^j. \tag{2.65}$$

t に関する同じ次数の項の係数を等値してつぎを得る．

$$\sum_{p=0}^{j}\frac{1}{p!(j-p)!}\sigma_p\sigma_{j-p} = \frac{2^j}{j!}\sigma_j + \frac{2}{j!}s_j\,(j=0,1,....). \tag{2.66}$$

この式の両辺を $j!$ 倍した後 $2s_j$ を他の 2 つの項によって表せば求める結果を得る．

注意 7
式（2.63）は s_j を σ_i で表すためには，$\mathbb{Z}[\sigma_m,...,\sigma_0]$ における環演算の他に 2 による除算が必要であることを示している．さらに検討するとこの除算は $\mathbb{Z}[a_n,a_{n-1},...,a_0]$ においては整除であることが分かる．

以上の 3 つの**注意** 5〜**注意** 7 を考慮すると，根和多項式を得る手続きは全体として，環 $\mathbb{Z}[a_n,a_{n-1},...,a_0]$ において計算することが必要十分であることが分かる．このことが，記号パラメータを有する安定判定に根和多項式を用いることができる理由である．

2.3.5　特殊 QE としての Strelitz テスト

多項式が安定であることを判定する決定手続きは，与えられた多項式の主変数を消去して，結果として「真」あるいは「偽」を与える特殊な QE であると看做すことができる．さらに，$\mathbf{p}=(p_1,...,p_k)$ を実数値を取るパラメータとするとき，多項式環 $\mathbb{R}[\mathbf{p}]$ から係数を採るならば，同じ手順がそのまま自然に質問（命題）が「真」となる条件を与えるものになる．結果の条件（命題）は，等式あるいは不等式を原子式とし，それらが論理和，論理積，論理否定などの論理演算子で構成された 1 階述語論理の論理式として表される．

一般に設計問題では，係数にパラメータをもつような多項式を扱う必要がある．定数係数の多項式を対象として作成された手続きは，パラメータの特殊化の問題に対策を講じる必要があるが， Strelitz テストでも同様に主係数問題への対策を講じる必要がある．次項ではこの主係数問題について述べる．

2.3.6　主係数問題

2.3.4 項で定義した根和多項式はモニックな実係数多項式に対してのみ計算される．数係数の多項式を扱う限りは，少なくとも理論上，とくに問題は生じない．数値であ

る主係数の逆数を多項式に乗じることで，対象となる多項式の根を変えることなく，モニックにすることが常に可能であるからである．

しかし，設計問題ではいくつかの未知パラメータを持つ多項式を扱う場合が通常である．もちろん，敢えて有理函数係数を扱うことにすれば，根和多項式を適用することは原理的には可能である．

有理函数係数を用いると，3 つの問題が生じる．ひとつは，有理函数係数を扱うことによる計算量の増大である．第 2 番目は，より本質的な問題で，パラメータに数値を与えて特殊化することで不可避的に生じる homomorphism anomaly である．簡単に言えは，f の主係数がパラメータの特殊化によって 0 になる場合について，特別な注意と処理をしなければならないということである．最後の問題は，アルゴリズムを実現する際に起こる問題であり，処理仮定において現れる多項式の主係数の符号に関する制約条件をいつも意識していなければならないということである．

注意 8
Sturm-Habicht 列は Sturm 列や部分終結式に対する修正であり，これら 3 つの問題，有理函数係数や homomorphism anomaly，符号制約問題を回避することができる．

以下に，Strelits テストを非数値係数の多項式に適用する上での，これらの問題の解決法を提案する．

2.3.7 形式的逆数法

最初に主係数が 0 にならない場合を検討し，つぎに主係数が 0 になる場合を検討する．

主係数が消えない場合

f の主係数がいくつかのパラメータ変数の多項式である場合には g の計算をつぎのように修正する．

$\kappa(>0)$ 個のパラメータを $\mathbf{p} = (p_1, p_2, ..., p_\kappa)$ とする．多項式 $f \in \mathbb{R}[\mathbf{p}]$ を

$$f = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 \tag{2.67}$$

の形とし，少なくとも主係数にはパラメータ変数を含む，すなわち，$a_n \in \mathbb{R}[\mathbf{p}]\backslash\mathbb{R}$ であるとする．

新しい不定元 t を選び，それを f に乗ずる．そうした後，主係数 ta_n のみを 1 で置き換え，

$$f_t = tf = z^n + ta_{n-1} z^{n-1} + \cdots + ta_1 z + ta_0. \tag{2.68}$$

を得る．同時に，後に g を修復するために，主係数 a_n が 0 にならないための条件を意味する，新しい等式（2.69)

$$ta_n = 1 \tag{2.69}$$

を追加する．この等式により，a_n は 0 にはなれないという制約を受けることになり，t は a_n の形式的な逆数を意味する有理函数 $\frac{1}{a_n}$ を表すことになる．こうして得られたモニックな多項式 $f_t \in \mathbb{R}[t, \mathbf{p}][z]$ は，$\mathbf{p}$ に対する同じ特殊化（数値の割り当て），およびそれによって誘起される t に対する特殊化をも含めて，後に議論する homomorphism anomaly の場合を除き，すべての特殊化において f と同じ根をもつ．

すると，この修正されたモニックな多項式 $f_t \in \mathbb{R}[t, \mathbf{p}][z]$ に対して，その根和多項式 g_t が，2.3.4 項の**アルゴリズム 3** によって，有理函数体での計算をせずに計算できる．すると，制約式（2.69）に基づき，t を $\frac{1}{a_n}$ で置き換えることにより，g が g_t から回復できる．

ゆえに，この場合には g の係数には通常 a_n の冪である分母をもつことになる．後に述べる応用においては，g の係数が正であることにのみ興味がある．それゆえ，実際の計算においては，適切な a_n の冪を掛けて g の分母をなくすことができる．そうしたときには，係数達に関する制約式の符号変化に注意を払う必要がある．一様な取り扱いを望む場合には，a_n の偶数冪を乗じるのが良い．

f が安定多項式である条件は，a_n の取りうる 2 つの可能性，すなわち $a_n > 0$ であるか，もしくは $a_n < 0$ であるか，に依存している．

$a_n > 0$ の場合には，「f と g の係数はすべて正」，となり，$a_n < 0$ の場合には，「f の係数はすべて**負**，かつ，g の係数はすべて正」となる．

主係数が消える場合

f の主係数が 0 になる場合，すなわち，

$$a_n = 0, \tag{2.70}$$

の場合には，制約条件として式（2.70）を追加した上で，問題をつぎの $n-1$ 次多項式 f_R に再帰させれば良い．

$$f_R = a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + \cdots + a_1 z + a_0 \tag{2.71}$$

この再帰は f_R の形式次数が 0 になるところで終了する．

注意 9

一般には，n 個の再帰的に生成される多項式の各々に対する安定条件の論理和を得ること になる．ただし，主係数が単に数値となるところで再帰を止めることができる．

アルゴリズム 4 に，形式的逆数法による特殊 QE アルゴリズムを示す．このアルゴリズム例では，形式的逆数とそれに関する制約条件とは結果の論理式中に残っている．

アルゴリズム 4 では 4 つの補助的な関数を利用している．それらは次のように定義される．

1. LC(f) は f の主係数を返す関数.
2. SORP(f) は f の根和多項式を返す関数.
3. Lt(f) は f の主項を返す関数である.
4. REST(f) は $f - \mathrm{Lt}(f)$ を返す関数である.

次数 $\deg_z(f) \geq 0$ なる多項式 $f \in \mathbb{R}[\mathbf{p}][z]$ に対して，**アルゴリズム** 4 で定義される関数 StableCond1(f) は，パラメータ $\mathbf{p}$ と動的に生成される不定元 t 達（再帰の数だけ異なる t が存在.）を含む 1 階術語の論理式を返し，その論理式は f が安定であることと等価な条件を与えるものである．

アルゴリズム 4 (StableCond1)
function StableCond1(f)
  begin
    if $\deg_z(f) = 0$ **then return** (**true**);
    if $\mathrm{LC}(f) \in \mathbb{R}$ **then**
      begin
        $f' := f/\mathrm{LC}(f)$;
        let $f' =: z^n + a'_{n-1}z^{n-1} + \cdots + a'_1 z + a'_0$;
        $g := \mathrm{SORP}(f')$;
        let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        return $((a'_{n-1} > 0) \wedge \cdots \wedge (a'_1 > 0) \wedge (a'_0 > 0)$
                $\wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0))$
      end
    else
      begin
        Choose a new indeterminate t;
        Construct f_t from f defined by equation (2.68);
        $g := \mathrm{SORP}(f_t)$;
        let $f =: a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0$;
        let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        PositiveCase := $(ta_n = 1) \wedge (t > 0)$

$\wedge\ (a_n > 0) \wedge (a_{n-1} > 0) \wedge \cdots \wedge (a_1 > 0) \wedge (a_0 > 0)$
$\wedge\ (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0)$;
NegativeCase := $(ta_n = 1) \wedge (t < 0)$
$\wedge\ (a_n < 0) \wedge (a_{n-1} < 0) \wedge \cdots \wedge (a_1 < 0) \wedge (a_0 < 0)$
$\wedge\ (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0)$;
NullCase := $(a_n = 0) \wedge$ StableCond1(REST(f));
return (PositiveCase $\vee$ NegativeCase $\vee$ NullCase)
end
end

注意 10
形式的逆数 t の取り扱いの違いによってこのアルゴリズムの変種がいくつかあり得る．不定元 t 達は，関数 StableCond1(f) の再帰呼び出しが起るたびに，新しく別の元が生成される．したがって，**アルゴリズム** 4 では，最大で n 個の異なる t 達が結果の 1 階術語論理式に現われる可能性がある．別の方法として，プログラム変数，PositiveCase と NegativeCase，への割り当て（代入）が，各再帰呼び出しごとに生じる際に，t を $\frac{1}{a_n}$ で置き換えることもできる．これらの違いは，結果の論理式をどのように簡約化するかに関係しており，別種の問題として論じるべきものである．

注意 11
実際のインプリメントにおいては，1 階術語論理式の論理記号の優先順位を破壊しないように注意しなければならない．上記のアルゴリズムは形式的正しさよりは簡潔さを優先しておいることを断っておく．この注意書きは**アルゴリズム** 5 においても同様である．

2.3.8　座標のスカラー線形変換

パラメータ付の主係数問題に対する第 2 の解決法として，モニックでない多項式をモニックな多項式に変換する，数係数の 1 変数多項式の因数分解で提案された良く知られた方法[10]を，採用することができる．

主係数 a_n が 0 になる場合については前項 2.3.7 と同じ方法を採用することができる．よって，ここでは $a_n \neq 0$ となる場合のみを論じる．

さて，モニックな多項式 $f_M \in \mathbb{R}[\mathbf{p}][w]$ がつぎの変数変換によって得られる．

$$f_M = a_n^{n-1} f(\frac{w}{a_n}). \tag{2.72}$$

[10]良く知られていることとそれが実際に使用されていることとは別である．因数分解では数係数が不必要に大きくなり，非効率的であるために実際には使用されることはまずない．

もし，ある a_n の特殊化が負であったならば，このスカラー線形変換 $z \mapsto w$ は点の位置を原点に関して入れ替える．すると，当然左半平面と右半平面とが交換されるため，このことに注意してアルゴリズムを構成する必要がある．

主係数 a_n が正となる制約下 $a_n > 0$ では，f_M の係数が正となる条件に加えて，f_M の根和多項式 g_M の係数が正となる条件が，所望の f の安定条件を与える．

一方，主係数 a_n が負となる制約下 $a_n < 0$ では，我々が使うべきは式（2.72）で定義される f_M ではなく，

$$\widehat{f_M}(w) = (-1)^n f_M(-w) = (-1)^n a_n^{n-1} f(\frac{-w}{a_n}) \tag{2.73}$$

で定義される $\widehat{f_M}$ である．この $\widehat{f_M}$ と，それから作られる根和多項式 $\widehat{g_M}$ の係数により，$\widehat{g_M}$ の係数が正である条件に加えて f の係数が**負**となる条件が所望の f の安定条件を与える．

アルゴリズム 5 に，スカラー線形変換による特殊 QE アルゴリズムを示す．f についての条件と補助的関数については**アルゴリズム** 4 の場合と同じである．

アルゴリズム 5 (StableCond2)
function StableCond2(f)
  begin
    if $\deg_z(f) = 0$ **then return** (**true**);
    if $\mathrm{LC}(f) \in \mathbb{R}$ **then**
      begin
        $f' := f/\mathrm{LC}(f)$;
        let $f' =: z^n + a'_{n-1}z^{n-1} + \cdots + a'_1 z + a'_0$;
        $g := \mathrm{SORP}(f')$;
        let $g =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;
        return $((a'_{n-1} > 0) \wedge \cdots \wedge (a'_1 > 0) \wedge (a'_0 > 0)$
                $\wedge (b_{m-1} > 0) \wedge \cdots \wedge (b_1 > 0) \wedge (b_0 > 0))$
      end
    else
      begin
        Construct f_M from f defined by equation (2.72);
        Construct $\widehat{f_M}$ from f defined by equation (2.73);
        $g_M := \mathrm{SORP}(f_M)$;
        $\widehat{g_M} := \mathrm{SORP}(\widehat{f_M})$;
        let $f =: a_n z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0$;
        let $g_M =: z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0$;

```
        let ĝ_M =: z^m + b'_{m-1} z^{m-1} + ··· + b'_1 z + b'_0;
        PositiveCase := (a_n > 0) ∧ (a_{n-1} > 0) ∧ ··· ∧ (a_1 > 0) ∧ (a_0 > 0)
                              ∧ (b_{m-1} > 0) ∧ ··· ∧ (b_1 > 0) ∧ (b_0 > 0);
        NegativeCase := (a_n < 0) ∧ (a_{n-1} < 0) ∧ ··· ∧ (a_1 < 0) ∧ (a_0 < 0)
                              ∧ (b'_{m-1} > 0) ∧ ··· ∧ (b'_1 > 0) ∧ (b'_0 > 0);
        NullCase := (a_n = 0) ∧ StableCond2( REST(f) );
        return ( PositiveCase ∨ NegativeCase ∨ NullCase )
    end
end
```

2.3.9　ロバスト制御系の D 安定性設計問題

この項では，D 安定性設計問題として知られる，ロバスト制御系の典型的な設計問題のひとつが Strelitz テストに帰着されることを見る．扱う例は文献 [34] から採った．

我々の立場から見ると，D 安定性は，実数または実数パラメータを係数とする多項式がそのすべての根を指定された複素平面上の領域 D の内部にもつことと等価な条件を求めることである．

$f \in \mathbb{C}[z]$ の根全体の集合を Zero(f) と書くことにすれば，パラメータを持たない f に対する D 安定性問題は，命題

$$\text{Zero}(f) \subset D \tag{2.74}$$

の真偽を判定する決定問題である．もちろん，D が任意の領域であるような場合，とくに f がパラメータをもつ場合には一般的に通用する方法はない．では扱える領域はどういう領域か?という問題はここでは追求せず，扱いが可能で制御の問題として一般的に良く使われる領域を例として，2.3.7 項あるいは 2.3.8 項で述べた Strelitz テストに基づく我々のアルゴリズムを適用することを試みる．

複素球面上の任意の円は，線形分数変換によって互いに移り合うため，複素平面上では，任意の直線（複素球面上では無限遠点を通る円）によって分割される任意の半平面 (境界は含まない）および，任意の円領域の内部を，線形分数変換によって左半平面に写すことができる．したがって，円領域の内部（半平面を含む）に多項式の根が存在するか否かは，Strelitz テストに還元できる．

2 つの領域 D_1, D_2 を扱うことが可能ならば，その交わり $D_1 \cap D_2$ も扱うことが可能である．なんとなれば，領域 D_1, D_2 に対する答は，D_1 に対する答と D_2 に対する答の論理積であるからである．

Strelitz テストの対象は実の係数をもつ多項式 $f \in \mathbb{R}[z]$ であるので，その根は実軸に対称に配置される．領域 $D \subset \mathbb{C}$ の実軸鏡映を $\mu(D) = \{\overline{z} | z \in D\}$ と書くことにする．

すると，容易に分かるように命題

$$\mathrm{Zero}(f) \subset D \tag{2.75}$$

と命題

$$\mathrm{Zero}(f) \subset (D \cap \mu(D)) \tag{2.76}$$

とは同値になる．このため，判定を行う領域が勝手に与えられたとしても，結局は実軸に対称な領域を指定したことと同等になることを注意しておく．

ここでは制御系設計でよく採用される基本的な 2 つの例が Strelitz テストにより効率よく取り扱えることを示す．

円形領域

ここでの問題は，与えられた多項式 $f \in \mathbb{R}[\mathbf{p}][z]$, $\deg_z f = n > 0$ がすべての根を複素平面上の円領域 D の内部に持つことと同等な条件を，係数に関する論理式として求めることである．ただし，$\mathbf{p}$ は実数で特殊化されるパラメータであるとする．ここで円領域の内部 D は，複素変数 z に関するつぎの制約式で指定されるものとする．

$$(z-c)\overline{(z-c)} < r^2, \tag{2.77}$$

ただし，$r > 0$ は境界となる円の半径， $c \in \mathbb{C}$ は円の中心の位置を表す．ここでの判定アルゴリズムの有効性には影響しないが，安定設計の場合には，D が実軸対称となるように c は負の実数に取られ，また D は右半平面とは共通点を持たない（従って，虚軸とも共通点を持たない）ように取られることが普通である．

与えられた円の内部 D は式（2.78）で定義される線形分数変換 $z \mapsto w$ によって，左半平面に写される．

$$w = \frac{(z-c)+r}{(z-c)-r}. \tag{2.78}$$

この逆変換は次式（2.79）で与えられる．

$$z = r\frac{w+1}{w-1} + c. \tag{2.79}$$

従って，実際的仮定として c が実数の場合には，問題は次式（2.80）で定義される多項式 $\widetilde{f} \in \mathbb{R}[\mathbf{p}][w]$ が，すべての根を左半平面に持つことと等価な条件を求めることになる．

$$\widetilde{f}(w) = (w-1)^n f(r\frac{w+1}{w-1} + c). \tag{2.80}$$

この変換では次数は変わらず，$\deg_w(\widetilde{f}) = n$ であることに注意する．

よって，問題は多項式 $\widetilde{f}$ が安定であるための等価条件を求めることに還元され，Strelitz テストを用いる**アルゴリズム** 4 あるいは**アルゴリズム** 5 で直接処理することができる．

とくに，実数軸上にない c を扱いたい場合[11]には，

$$F(w) = \widetilde{f}(w)\overline{\widetilde{f}(\overline{w})} \in \mathbb{R}[\mathbf{p}][w], \tag{2.81}$$

を作り，それを**アルゴリズム** 4 や**アルゴリズム** 5 に与えれば良い．ただし，次数が倍になる分だけの追加計算コストが必要になる．

注意 12
虚の c，すなわち円の中心が実数軸上にないような実用上の場合とは，互いに実軸に対して対称の位置にある 2 つの円の空でない交わりとして D が表されるような場合である．そのような場合に，$F(w)$ の安定条件は f の D 安定条件を与える．これは，安定度を増した設計が望まれる場合などに，根をより実軸の近くに集める方法として使われる．

楔形領域

今ひとつの例は，2 つの半平面（境界を含まない）D_1 と D_2 の共通領域として与えられる楔形領域 $D = D_1 \cap D_2$ である．目的は前項と同じく，f の全ての根が D の内部にある条件を求めることである．

考察の対象は $\mathbf{p}$ を実のパラメータとする多項式 $f \in \mathbb{R}[\mathbf{p}][z]$ であり，いかなる $\mathbf{p}$ の実数値への特殊化に対しても，根は実軸に対称に配置されているので，本項冒頭で考察したとおり，判定を行う楔形領域 D は一般性を失うことなく実軸に対称であるとしてよい．

よって，$D_1 = \mu(D_2)$ と仮定してよく，$D = D_1 \cap \mu(D_1)$ となる．ゆえに，$\mathrm{Zero}(f) \subset D_1$ を判定するのみで良いことになる．なぜなら，命題 $\mathrm{Zero}(f) \subset D_1 \cap \mu(D_1)$ は命題 $\mathrm{Zero}(f) \subset D_1$ と同値であるからである．

そこで，D_1 を，$z_1 \in \mathbb{C}$ を通り，$c \in \mathbb{R}$ $(c \neq z_1)$ で実軸と交わる直線を境界とし，境界線上のベクトル $z_1 - c$ に対して左側にある半平面としよう．

この半平面 D_1 はつぎの不等式を満たす点 z として定められる．

$$i\,\overline{(z_1 - c)}(z - c) - i(z_1 - c)\overline{(z - c)} < 0. \tag{2.82}$$

この D_1 は

$$w = i\,\overline{(z_1 - c)}(z - c). \tag{2.83}$$

[11] これは 2 つの円の交わりを D とするような場合である．**注意** 12 を参照．

で定義される変換 $z \mapsto w$ によって左半平面に写される．

これは，線形分数変換の 1 種であり，平行移動，回転および拡大/縮小の組み合せのみからなる．その逆変換は

$$z = \frac{1}{i\,(z_1 - c)}w + c. \tag{2.84}$$

となる．したがって，f の D 安定性は次式（2.85）で定義される $\widetilde{f}(w)$ に関する通常の安定性に帰着される．

$$\widetilde{f}(w) = f(\frac{1}{i\,(z_1 - c)}w + c). \tag{2.85}$$

ただし注意すべきは，$\widetilde{f}(w)$ は通常は虚数の係数を含んでいることである．そのため Strelitz テストに基づく**アルゴリズム** 4 あるいは**アルゴリズム** 5 を適用するにはつぎの F を使わなければならない．

$$F(w) = \widetilde{f}(w)\overline{\widetilde{f}(\overline{w})} \in \mathbb{R}[\mathbf{p}][w]. \tag{2.86}$$

2.3.10 考察

先行研究との比較

文献 [34] や [35] では制御系設計問題の多くが, SDC 問題に帰着されると述べられている．また, 文献 [6] や [7] においては, パラメータを持った SDC 問題も Sturm-Habicht 列計算によって効率よく解決されると主張されている．

しかしながら，計算量およびアルゴリズムの簡潔さ（これはインプリメント上のバグを少なくする．）の観点から別の定式化の方が適している問題がいくつかある．上記の文献でとり上げられた SDC 問題のいくつかは，不自然で長い迂回絡を取った上で，一見は異なる問題に帰着されたかに見えるが，通常の安定性問題と同等の問題，つまり，ある多項式が左半平面に全ての根をもつことの判定，に最終的には辿り着いている．その典型が文献 [34] の D 安定性問題である．

ここで，文献 [34] の D 安定性問題解法のプロセスを我々の言葉で表現してみよう．以下では，$< f_1, ..., f_s >$ を $f_1, ..., f_s$ により生成される多項式イデアル，$\mathrm{Zero}_{\mathbb{C}}(I)$ はイデアル $\mathcal{I}$ の複素零点全体の集合，また，$\mathrm{Zero}_{\mathbb{R}}(I)$ はイデアル $\mathcal{I}$ の実零点全体の集合とする．

1. 問題は，実係数あるいは実パラメータをもつ多項式 $f(z)$ と領域 $D \subset \mathbb{C}$ が与えられたとき，命題 $\mathrm{Zero}_{\mathbb{C}}(f(z)) \subset D$ の真偽を決定すること，もしくはそれと同等で z を含まないパラメータに関する論理式を導くことである．

2. 複素平面 $\mathbb{C}$ をユークリッド平面 $\mathbb{R}^2$ と同一視し，$D \subset \mathbb{R}^2$ と見做す．ついで，原問題を実の 2 変数多項式に対して命題 $\mathrm{Zero}_{\mathbb{R}}(< f_r(x,y), f_i(x,y) >) \subset D \subset \mathbb{R}^2$ の決定問題あるいは x, y を消去した同等な命題を得ることに変換する．ここに，$f(x+iy) =: f_r(x,y) + i f_i(x,y)$ である．

3. 領域 D はユークリッド平面の左半平面 $\mathbb{R}_{<0} \times \mathbb{R} \subset \mathbb{R}^2$ に写される．その上で，命題 $\mathrm{Zero}_{\mathbb{R}}(< f_r(x,y), f_i(x,y) >) \subset D$ は $\mathrm{Zero}_{\mathbb{R}}(< g_r(t,\omega), g_i(t,\omega) >) \subset \mathbb{R}_{<0} \times \mathbb{R}$ に変換される．ただし，これが可能なためには，$g(t+i\omega) = g_r(t,\omega) + i g_i(t,\omega)$ が成立するような，適当な $g(z) \in \mathbb{R}[z]$ を見つける必要がある．

4. イデアル $< g_r(t,\omega), g_i(t,\omega) >$ についての t の最小多項式を $m(t)$ とする．すると，$\mathbb{R}^2$ での問題は $m(t)$ に関する SDC 問題，すなわち，$m(t)$ が半開直線 $[0, +\infty)$ に根を持たないという条件を判定する，あるいは，それと同等な t を含まない条件を求める問題となる．さらに，もし $m(t)$ が $\tau \in [0, +\infty)$ に根をもつようなことが生じた場合には，追加の作業として，実の ω で $g_r(\tau,\omega) = g_i(\tau,\omega) = 0$ を満すものが存在しないということを確かめる必要がある[12)]．

5. SDC 問題を Sturm-Habicht 列計算に基づく方法で解く．

第 3 番目の問題は D の $\mathbb{R}_{<0} \times \mathbb{R}$ への変換が線形分数変換により得られたものであるなら，通常の安定性問題となんら異ならない．文献 [34] の 2 例は，ユークリッド平面上の変換として記述されていたが，まさしくそのようなものであった．結論として，第 4 番目のステップは不必要な作業である．第 2 と第 3 のステップはまとめて，線形分数変換によって置き換えることがより勝っている．そして必然的に最後のステップは Routh-Hurwitz 判定に委ねられるか，より効率を求めるならば，Strelitz テストに置き換えられるべきである．

同じ文献で検討されているゲイン余裕の問題も Strelitz テストで解かれることは同様である．さらに多くの問題を検討すれば Strelitz テストによって，スマートに解かれる問題がさらに発見されるであろう．

Strelitz テストをパラメータ付の多項式に対して適用することと，典型的な安定性の問題を Strelitz 条件で定式化する方法を提示した．その適用が安定性問題に還元できる問題に限られているとはいえ，簡単な構成のアルゴリズムは記号計算にとっては好ましいものである．

[12)] この追加作業は決して容易な作業ではない．とりわけ，$f(z)$ が記号パラメータを持っている場合には．

根和多項式の拡張

ここで用いた根和多項式のつぎのような拡張が別の安定性問題に関連して現われる.

$$G_L(z) = \prod_{i_1}(z - \alpha_{i_1}) \cdot \prod_{i_1<i_2}(z - (\alpha_{i_1} + \alpha_{i_2})) \cdots \prod_{i_1<i_2<\cdots<i_n}(z - (\alpha_{i_1} + \alpha_{i_2} + \cdots \alpha_{i_n})), \quad (2.87)$$

本節で紹介した根和多項式の計算法を修正して，G_L の各因子を個別に計算すれば，効率よく計算が可能であることを指摘しておく.

第3章　数式処理システム Risa/Asir の開発

3.1　Risa/Asir 開発の動機

Risa/Asir は，「計算機」で確実に実行可能な「代数計算」が実行できるようにして，数学研究や理工学の教育/研究で利用することを目的として開発された [146]．富士通という企業の中ではあったが，開発に携わった筆者達は，アカデミックな雰囲気の研究部署，国際情報社会科学研究所に所属していた．そこでは，商品開発のベースとしてではなく，広い意味では計算機科学の，狭い意味では数理計算科学の，先行的かつ実験的研究の道具として，またそれ自身を研究の対象として開発することが許された．そのため，システムの開発に当っては「数式処理」の応用面よりも，大規模問題にも耐え得る効率的な代数的アルゴリズムの提供を目指し，ユーザインタフェースについては優先度を低くしている[1)]．

上述のような当初の目的から，主として多項式（整式）に対する演算については，初等的な代数演算から高等代数に応用される演算までが整備されている．有理式や初等超越函数を含む代数式に関する演算も，限定的ではあるが可能であり，導函数の計算や有理函数の不定積分も可能である．ただし，初等函数や根号，分数冪を含む式の簡約化については，ごく単純な場合を除いて提供されていないため，ユーザが個別に Asir でプログラムを書いて処理をする必要がある．代数演算以外にも，浮動小数点数による通常の数値演算と任意多倍長数値演算が可能であり，初等函数やその微分係数の数値計算を併用することができることは他の多くの数式処理システムと同様である．この多倍長数値演算については Pari のライブラリ[2)]を利用して計算している． グラフィックスは平面曲線の描画のみが提供されているが，その 2 変数代数曲線の描画性能は高く，研究者には重用されている．

上述のように解析的な演算に基づく式変形の操作には制約はあるが，教育的な応用としては，中学高校で教えられるヒューリスティックスとしての多項式の因数分解をはじめとする代数計算（整式の四則演算ができる範囲）を，システマティックな方法

[1)] Risa/Asir のインタフェースは GUI（Graphical User Interface）でなく，CUI（Command line User Interface）である．陰函数描画機能では GUI 機能を一部取り入れている．

[2)] Pari: Henri Cohen 他が開発の整数論の計算のための C Library．http://pari.math.u-bordeaux.fr/

で確実に実行する方法として有効である．平面曲線の描画と初等函数の数値計算を利用することにより，微分とその応用（極大，極小，変曲点，制約最適問題）などへの教育にも適用が可能である．このことについては第 6 章に詳述する．

工学的な応用に関しては，グレブナー基底を用いた多変数連立代数方程式系の解法を適用することで，ロボットの運動学方程式の解法[3]他，多くの等式制約問題の解法への応用がある．

筆者が，データベース以後の計算機の新しい応用開拓の有力候補として，数値計算では実行困難な数理問題への解法としての数式処理の検討を考えていたとき，ICOT[4] 中期の実証プログラムの一つとして応募してはどうかと，当時の上司 辻ヶ堂[5]から示唆を受けて応募したところ，ICOT に採用された．これが 1985 年のことである．Risa/Asir の開発はまだ 4 年後であるが，ここに Risa/Asir の誕生前史が始まった．

3.2　Risa/Asir 名前の由来

Risa は，富士通株式会社 国際情報社会科学研究所（1990 年 3 月当時） 第一研究部第三研究室で，計算機代数（数式処理）の研究の成果として，室長 竹島 卓[6]，横山和弘[7]，野呂正行[8] の 3 名の手によって誕生した． 実際に C コードを書いたのは野呂である．

Risa の名前[9]が最初に世間に出たのは，1990 年 3 月 13 日 理化学研究所で当時毎年開催されていたシンポジウム「記号数式処理と先端的科学技術計算」での研究発表においてである．そのタイトルは，「数式処理システム risa（仮称）の現況—その 2」とあり，この時点ではまだ「仮称」が付いていた．1990 年 6 月 18–20 日に那須で開催されたソフトウェア科学会の数式処理研究会では「仮称」のない risa として発表された．

命名したのは筆者である．それまでは数式処理の算法（アルゴリズム）を断片的に試作した順に，「エンジン」，「エンジン 2 号」あるいは 「Nor-Yokosyma」などと呼んでいた．Macsyma の開発者の一人 R. Fateman 教授が富士通を訪問された時，日本語の Yokosyma の意味は「evil mind」であるから，「Nor-Yokosyma」は「(neither

[3] 文献 [120] の 4.8 節に筆者による 4 節リンク機構の設計計算の具体例が掲載されている．
[4] （財） 新世代コンピュータ技術開発機構．
[5] 辻ヶ堂 信．国際研 研究管理統括部長．のち工学院大学教授．
[6] 現 富士通研究所 IT コア研究所専任研究員．
[7] 現 立教大学理学部教授．
[8] 現 神戸大学理学部教授．
[9] 日本人は “L” と “R” の区別が付かないということから，英語圏ではポピュラーな女性の名前の “Lisa” と綴をまちがってはいないか？と気遣って下さる方もあるが，“Risa” が本名である．NHK 教育テレビの英会話を担当されていた Risa Stegmayer さんという方もいらっしゃる． （日本名は「りさこ」さんだということだが，そのローマ字表記では “Risa” とされている．）

Macsyma） nor evil mind」だと言ったら大笑いされた．この段階では「エンジン」はまだシステムとしての体裁を成しておらず，数式処理に必要とされる要素算法がばらばらに作成されていただけであった．

1989 年の夏，当時の国際研 小口所長[10]が，数式処理グループの研究ヒアリングをした折り，『この「エンジン 2 号」という名では外部に公言するのは問題がある，今後システムとしてまとめていくのなら，適切な名前を付けて公にし，責任を持って実行せよ．』と指示を受けた．これを受けて，「Research | Symbolic Algebra」の略称として「R|sa（発音はリサ）」という名前を案出した． ここで「|（縦棒）」は論理型言語 Prolog での記号式（Prolog term）の連接（concatenation）から取ったもので，「記号代数の研究」の意味であった． しかし，略称が意図としての「リサ」と発音してもらえないという不便があったので，「|」をローマ字「i」で置き換え，「Research Instrument for Symbolic Algebra」（記号代数のための研究道具）の略として「Risa」と呼ぶことにした．「Asir」は「Risa」の言語インターフェースとして野呂が命名した．

「Risa/Asir」は筆者等を含めた日本人は，だいたい「リサ・アジール」と呼ぶのが普通であるが，1992 年の ISSAC92 で野呂の発表 [46] を司会した Geddes 教授[11]が「リサ・エイシャ」と聞こえる発音で紹介された．（「リサ・アジァ」に近い発音をして，Asia を意識したのか，との問いを受けることもある．）

1990 年という年は，記号数式処理国際会議（ISSAC90）が東京で（アジア地域では初めて）開催された年で，富士通の数式処理グループも開催の事務局の一員として，また発表者としてその準備に忙しくしていた．この頃は Risa は一般に公開されていたわけではなく，富士通グループの計算機代数（数式処理） の基礎研究の成果として，また，それ自身が計算機代数の算法（アルゴリズム） の研究開発の道具として，富士通内部で利用される他は，数式処理研究者の間でも限られた利用であった．Risa という名前を付けてからは，多くの支援者を得ることができ，それらの人々の協力を受けて今日に至っている．

3.3 富士通での数式処理研究の開始

富士通の数式処理グループが数式処理の研究を始めた契機は，1985 年に富士通が ICOT[12]から，PSI Machine[13] 上の実証ソフトウェアのひとつとして，「数式処理システム」の開発を受託したことであった．現在，商用数式処理システムとして知られて

[10]小口 文一 富士通研究所社長（当時）．国際情報社会科学研究所 所長を兼務．
[11]K. O. Geddes. Waterloo 大学教授，Maple の開発者のひとり．
[12]新世代コンピュータ技術開発機構．第五世代コンピュータの開発母体．
[13]Personal Sequenctial Inference Machine （ = パーソナル逐次型推論機械．） ICOT の最終目的としたハードウェアは PIM（Parallel Inference Machine = 並列型推論機械）であったが，この時期（ICOT 中期）の実証ソフトウェアは PSI 上に試作する計画になっていた．

いる Maple や Mathematica などの現代的システムは，発表されたばかりであり，日本ではほとんど知られていない時期であった．それまでは，理化学研究所（当時）の後藤先生[14] が FLATS という数式処理専用マシンを開発し，その上に Reduce[15]を走らせて，高精度の電子線リソグラフィー用の電子レンズの設計に成功したというように，数式処理といえば殆んどの場合 Reduce が利用されていた．この頃までの著名な数式処理システムは，記号処理の容易さから Lisp で作られるものがほとんどであり，Reduce も Lisp 上に作成されていた，理化学研究所では，佐々木博士[16]が GAL（General Algebraic Laboratory）をやはり Lisp の上で作成していた．

筆者らは，システム開発の先輩として理化学研究所に佐々木博士を何度か訪ね，今後重要になる数式処理の分野は何か，システムを開発していくに当りどのような機能を目標にするのがよいかなど教えを受けた．

筆者は大学ではロボットの研究をしていたが，ロボットの機構と制御系を記述する方程式はロボットの自由度（主として関節の個数）が多くなると，手計算にはとても耐えられないほど大きくなり，その導出（と検算）はいつも苦労していた．そして，この計算に計算機が使えないものかと常々考えていた．IBM には数式が記号のまま計算できる PL/I FORMAC というプログラムがあると聞き，試用してみた．しかし，FORMAC を使うことができるマシンは大型の計算機であり，個人が自由に使えるようなものではなかった．

筆者は当初「記号処理」の方に興味があった．しかし，この仕事を始めるに当って新たにメンバーとなった，横山，野呂の両名は数学の研究者であった．当時の北川 敏男 国際研会長から言われた言葉は，「今度，ICOT の仕事をするにあたって 2 人を配属する．但し，この 2 名は数式処理に必要な数学については十分な力を持っているが，純粋の数学の学徒であるから，君のように計算機だ，記号処理だのということはまったく知らない．彼らの力と資質とを損なわないように，少なくとも半年は計算機には触らせないように!」というものであった．

筆者はこの会長命令を忠実に守り，当座は数式処理に関連する数学の学習を課題として 2 名に毎週セミナーをさせる計画を立てた．横山は因数分解と GCD，野呂は Risch の不定積分とグレブナ基底，という分担にした[17]．因数分解については，横山が古典的教科書から当時最新の論文までを探し求めて，Berlekamp に始まる有限体の因数分解から始め，片端から読破し，セミナーも熱心に実施していった．一方，野呂はグレブナ基底について，Buchberger[18]の論文をセミナーした．

---

[14]後藤英一．東京大学名誉教授．数式処理学会名誉会員．

[15]Rand Corp. の A. C. Hearn 博士が開発．世界中を巡って普及に努められた．

[16]佐々木 建昭．現 筑波大学教授．

[17]当時，ICOT の希望として，初等函数の不定積分機能を実現する数式処理システムを論理形言語 ESP で作成することで，契約がまとまりつつあった．グレブナ基底は理研の佐々木博士も今後の重要技術と言っていたことを勘案し，学習対象としていた．

[18]Bruno Buchberger, 記号計算研究所 RISC-Linz の創立者で前所長. それまで算法は存在しないと考

当時，筆者は沼津愛鷹山の中腹にあった国際研から，東京大田区蒲田の国際研分室に毎週 2 日通って，横山，野呂とともに関連する数学を学習した[19]．因数分解に関して，有限体や代数的拡大体，Risch 算法に関して代数函数体，超越拡大などのテクニカルタームに関して両名からの説明を聞いた．

工学系出身で数式処理の発展を願う者としては，現在の数学科以外での理工学系の高等教育は「解析」の学習に偏重しすぎていると言わざるを得ない．代数の基本的な概念である，群，環，体，モジュールなどが数学を除く理工学部の授業には採用されていないのが残念ながら現状である．理工学の応用では数値シミュレーションが欠かせないが，そのために必要な数学的背景知識は解析の知識であることがその理由であろう[20]．

しかし，このことは「数式処理に何ができ，なにができないか」について肝要なところでの理解を妨げ，個人レベルで言えば，「数式処理システム」を効果的に利用する上での障碍となり，社会レベルで言えば，「数式処理」がテクノロジーとして受け入れられるための大きな障壁となっているように思われる[21]．

3.3.1 日本の数式処理システム開発

わが国は数式処理システム開発では後進国である．その中で，Risa/Asir は現在も活発に開発が進行している国産の汎用数式処理システム[22]である． 特定分野用途としては，kan[23]がある [72, 71]．kan は現在では Risa/Asir と同じく OpneXM 開発の一環として開発が継続されている．

前項冒頭で述べたとおり，1985 年当時，人工知能（AI），知識工学などのソフトウェアは殆んどが記号処理言語 Lisp で作成されていた．著名な数式処理システムも同様

えられていたグレブナー基底の計算法（Buchberger 算法と呼ばれる）を考案．Berlekamp-Zassenhaus の因数分解算法，Risch の不定積分算法と並んで数式処理の 3 大基本算法と呼ばれる．

[19] この他，ICOT に納品するソフト作成に必要な背景知識を共有するため，富士通 SSL から 2 名の方にも参加していただいていた．

[20] 電子工学で講義があった符号理論は特殊な例外である．

[21] 数式処理システム（特に商用）では，ユーザの当座の要望をみたすために，敢えて限定された状況にしか対応できない処理を提供していることがある．そのような adhoc な処理はアルゴリズムが確立されていないか，よくあることだが，アルゴリズムがそもそも存在しない場合に処方されるため，無限定に使用すれば期待外れの結果を，時には数学的な誤りをも，生むことになり利用者から非難を受ける．反対に，確実にできることしか提供しない場合には，機能不足として軽んじられる．このような状況は数式処理にとって大変残念なことである．Risa/Asir はこの後者のような保守的な態度で開発されて来た．

[22] GAL の開発は現時点では活発とは言えない．1970 年代には NTT 横須賀通信研究所で AL（Algebraic Language）[102] が完成されていたが，一般への公開は実施されなかった．また，1986 年頃には，愛媛大学 野田研究室で Prolog をベースにした小型数式処理システム Sync[149] が試作されている．

[23] kan/sm1．1994 年，神戸大学 高山教授．代数解析のための数式処理システム．非可換/可換環の数学計算用で OpenXM プロトコルにより Risa/Asir との相互接続が可能．

であった．この Lisp の伝統に対し，第五世代コンピュータでは，人工知能の研究開発に適した Prolog という記号処理言語をベースにした ESP （Extended Self-contained Prolog）を開発言語としていた．そして，古い時代の AI のひとつの成功例である「数式処理—古い酒」を「第五世代コンピュータ—新しい皮袋」に詰め替え，Prolog（あるいは ESP）による実用的ソフトウェアの開発の実証例のひとつとしたい，というのが第五世代コンピュータ側の希望であった．当時の Prolog 処理系が Lisp の処理系ほど効率化が進んでいなかったせいもあるだろうが，ICOT 以外の数式処理研究者の目は Prolog に対してまったく冷やかであった．

このような状況と並行して当時の「数式処理」の潮流は，単に記号を処理するという意味での「数式処理（fomula manipulation）」から離れ，数式の代数的構造を利用して処理の効率を上げ，素朴な教科書的/原理的方法では計算に困難が伴うより高度な代数的処理を可能にして，実用的問題の解決に資することと，その処理能力をさらなる数学（代数）の研究に活用するという「計算機代数（computer algebra）」へと変化しつつあった．つまり，当時の世界的動向から見れば，「数式処理」は人知を真似る AI の対象ではなくなってきており，また，記号的処理の比重の相対的低下から，実現のためのプラットフォームが Lisp であるか Prolog であるかはもはや問題ではなくなっていた．はっきり言えば，「Lisp も Prolog も不要である．他のソフトウェアとの親和性と移植性を考えれば，C のようなどんなプラットフォームでも効率的な処理系が提供されるシステム記述言語で開発するべきである.」というのが，筆者等の認識であった[24].

「数式処理」=「記号処理」⊂「人工知能」という図式は一般の人々のあいだに相当長い期間[25]生きていた．「実用規模の数式処理のほとんどの計算時間は，記号（と）構造の処理にではなく，数値（整数）の計算に費やされている．数値（大規模な整数）の高速処理こそ数式処理の根源的課題のひとつである[26].」という事実への理解は少ないようである．1997 年頃，私共の上司のひとりが「数式処理は（ Prolog や Lisp で作るエキスパートシステムのような）記号処理ではなくて，（数値計算にむしろ近い普通の）数理科学計算ソフトウェアなんだね.」と言ったときにはむしろ筆者等のほうが驚いた．

[24] 当時リリースが始まって間もない Maple や Mathematica など第 3 世代の商用数式処理システムは C で書かれている．

[25] 少なくとも 1990 年代中は．

[26] もちろん，多項式などのデータ構造の処理も大規模になればなるほど重要になる．

3.3.2 ICOT での数式処理システム開発

前述のような時代背景を筆者らは認識していたが，ICOT での仕事としては論理型言語[27] とりわけ PSI 上で動作する世界初の本格的「数式処理システム」の開発を目指した．

目標とする機能は初等函数の不定積分が可能なことと決っていたので，そのために不可欠な下位機能として，多変数多項式の整数上での因数分解と GCD 計算，1 変数多項式の代数的拡大体上での因数分解と GCD 計算，最下位機能として任意多倍長整数の計算などの開発目標を立てた．最初の一年間は算法の調査に掛かりっきりで，システムの試作はほとんどできなかった．わずかに，今は懐かしい FM11 の OS9-BASIC で，任意多倍長整数計算ルーチンの習作（野呂）ができたくらいで，計算機パワーの豊富な今日からは想像もできないような貧困な計算機環境で，因数分解，不定積分，グレブナ基底などについての欧米の理論成果のキャッチアップとプログラミング技術の習得に努めていた．沼津国際研との交流の一助にと始めた「シスラボ通信」なる部内紙に，筆者はリーダとしての学習の理解度を示すために「楽しい因数分解」なる連載記事を書いた（横山監修）．これには，Berlekamp/Zassenhaus のアルゴリズムが脚色されて記述されており，そのアルゴリズムを知っている人なら補って読める．もっとも，次第に忙しく余裕がなくなったためもあり，「シスラボ通信」は 2 号が発行されたのち廃刊となってしまった．

この PSI-II 上の数式処理システム SAM は 1989 年 3 月に完成した [132]．このシステムは, Risch 算法による初等函数の不定積分が可能なシステムで, それ自体は Reduce や Maple にも実現されており[28]，珍しいものではなかったが，当時実現しているシステムがあまりない，代数拡大体上の因数分解機能を持っていた点がユニークといえる．

この期間に因数分解 [87, 129, 130, 164, 165, 168, 88] や GCD 計算 [131, 166]，代数拡大体計算 [167, 88] に関連する理論成果を得ている．

3.4 Risa/Asir の開発

3.4.1 数式処理システム Risa とその言語 Asir

ICOT への納品が完了すると，富士通の数式処理グループは SUN や DOS/V マシンのような一般的なプラットフォーム上で独自に数式処理システムの開発を始めること

[27] Prolog に代表される定理証明系をプログラムの操作意味論とする計算機言語．ICOT の ESP は Prolog を拡張し，オブジェクト指向を取り入れた論理形言語である．

[28] Mathematica の不定積分は，Risch 算法によるものではなく，パターンマッチングに基づくヒューリスティックな方法を採用していた．

になった．新しく始めるに当っては，あまり実用性のない，初等函数の不定積分などは避け，代数計算の基本に戻って，重点を多項式の効率的処理に置くことにした．新しい機能や新しい効率的算法を実現するために，データ構造を含め独自のものを新たに開発することになった．

システムが次第に大きくなり，また経験を積むにつれて，開発の初期には予見できなかったシステム上のいろいろな部分の整合性が問題になってきた．新しい発見や理論成果を取り込んだり，性能や作成しやすさの向上のためにも，システム構成を整理したほうが良いと思われるようになってくるからである．ICOT の受託終了はひとつの契機であったが，公開までには 2 度ほど大幅な見直しを経験している．

また，ICOT に納入した数式処理システム SAM の言語インターフェースは Prolog 風のものを作成していたが，新しく作成される数式処理エンジンのために C 言語風の言語インターフェースの作成も始めた．この新しい数式処理のエンジンとその言語インターフェースが Risa と Asir と呼ばれるようになるのは，1 年後の 1990 年になってからである．

Risa の言語インターフェース Asir の第 0 版は 1990 年 9 月に完成した [153, 45, 154]．その時点ですでに Asir に dbx 風のデバッガが組み込まれていたことは，試行錯誤しながら算法を完成させていくことの多い数式処理の研究開発者自身の切実な要請からであった． その後，開発者の所属した国際情報社会科学研究所の富士通研究所への組織移管，「情報社会科学研究所」へと名称の変更，さらに「情報社会科学研究所」廃止（1996 年 11 月）と，組織上の変遷があったが，開発者の竹島，横山，野呂の 3 名は，富士通研究所にて高性能数理計算グループとして「数式処理（計算機代数）」とその応用の研究開発を続けた．

1989 年は，Risa/Asir としての数式処理システム開発の実質的な初年である．それまで Prolog 向けに作成していたデータ構造などは捨て，改めて有理係数多項式の処理を主目的とするデータ構造を再定義した．その上で，四則演算，GCD，無平方分解をインプリメントした．この時点では GC（Garbage Collection）機能は未実装で，メモリ管理は C の alloc(), free() で直接コーディングされていた．そのため，コード量が増加するに従って開発には困難を来たすようになった．1989 年末頃 Boehm-Weiser による conservative garbage collector[12] を採用することでこの問題は解決した．現在にいたるまで Risa/Asir では Boehm-Weiser GC が採用されている．

また，この年の 9 月には筑波大学の井田 哲雄教授の紹介を得て横山が RISC-Linz を 1 ヶ月訪問した．この訪問により，ヨーロッパでの数式処理の先端研究について多くの情報や研究者との繋がりが得られた．これが契機となり，RISC-Linz（とりわけ Buchberger 所長[29)]）との密接な研究協力関係が築かれることとなり，そのことは富士通の数式処理研究の大きな推進力となった．

[29)] Buchberger 所長は 初期の Asir Manual 英文版の校正をしたことで，Risa/Asir の開発に直接貢献

1990 年には，多変数の無平方分解に Wang-Trager アルゴリズム [84] を採用して高速化し，さらに，多変数因数分解，GCD についても EEZ 法のアルゴリズム [83] の一部を EZ 法 [41, 82] に採用して，EEZ 法でなくても EZ 法で十分な高速化が達成できた．また，コマンド言語がプログラミング可能な形に整えられ，Asir 言語となった．理論研究では，因数分解 [89, 90, 91]，連立方程式解法 [134]，行列の有理標準形の計算法 [135, 136] などの成果があった．

1991 年には浮動小数点数がサポートされ，同時に陰函数描画アルゴリズム [75, 116] がインプリメントされた．これは，TCP/IP プロトコルで Risa/Asir の外部にある描画プロセスを呼び出すもので，分散計算の Risa/Asir への最初の実装である．なお，2000 年からは分散計算は OpenXM プロトコルで実装されるようになった．この年 1991 年には，その時点までの Risa/Asir の開発をテーマとして，富士通での数式処理研究の成果をまとめ，富士通の技術誌 [73] に掲載した．

1992 年には，それまで Risa/Asir になかった BigFloat（多倍長浮動小数点数）の演算を PARI ライブラリをリンクすることによって実現した．これによって，多くの超越函数の BigFloat 計算（BigFloat 複素数計算を含む）を Asir プログラムから呼び出して使用できるようになった．このころから，グレブナー基底計算パッケージの開発に着手し，その演算に適するよう分散表現多項式が Risa/Asir のデータオブジェクトに加えられた．また，この頃に因数分解関係の理論成果がいくつかまとまり [93, 94]，グレブナー基底を利用した連立方程式解法の理論も完成 [92] した．さらに，グレブナー基底についてのチュートリアル記事 [137] を発表した．この時点以前では，640k DOS で動作する Petit-Asir[105] 以外は，もっぱら Unix ワークステーションのみがサポートプラットフォームであったが，Macintosh を含むパソコンへの移植にも着手した．

1993 年にはグレブナー基底計算の最初の版が実装完了し，当時のデータでは世界最高速を記録していた．また DOS 版/ワークステーション版を問わず様々なマシンへの移植が進んだ．Risa/Asir の開発の協力者への便宜のために，頻繁に現況報告を行い [156, 157]，内部構造についての文書 [155] も提供した．理論方面では Hensel 構成の一般化 [96]，や多項式時間因数分解 [97] を発表した．

1994 年 6 月 1 日に Risa/Asir はそのバイナリ（V940420）の公開をネットワークニュースグループ sci.math.symbolic, sci.math に宣言し，ノンプロフィットの利用に関しては無償で自由に使って貰えるようにした．このことで，Risa/Asir は国産初の公開された数式処理システムとなった．開発者自身，富士通の英断に感謝している．この時点では動作するプラットフォームは Unix 系が主であったが，MSDOS 環境では DOSExtender （exe386, go32）を使用することによって動作する dos 版が，また Macintosh で動作する Mac 版も提供された．理論研究では，2 変数多項式の多項式時間因数分解の理論を発表 [98] した．

している．

その後も，当方が使用可能な限りのハード/OS に移植作業が継続されたが，Windows への移植は要望がありながら容易には実現できなかった．

1995 年 11 月 14 日にはグレブナ基底計算関係の機能性能が大幅に強化され，分散計算機能が拡充された新版（V950831）を公開した．同年秋の第 1 回の Risa Consortium ワークショップに参加したユーザに Windows 版（3.1, NT, 95, V950831 相当）Risa/Asir の CD が配布された．これは少し遅れて一般公開された．

1998 年には，Windows 版のインストーラ CD 付きの単行本 [119] が SEG 出版より出版された．これは表紙タイトルの内「日本で生まれた数式処理ソフト」の部分が大きな文字で書かれており，注目を引いたらしく，出版以後 Risa/Asir への問い合わせや，バイナリへの ftp アクセスも多くなった．また，この書籍の序文では，桂先生[30] が Risa/Asir を「日本が誇るべきソフト」と紹介して下さったことも Risa/Asir 普及の大きな要因となった．

2000 年には，Risa/Asir に高性能の因数分解機能（代数拡大体上を含む）やグレブナ基底計算機能があったため，暗号の研究開発[31]が短期間で進み，会社に対する貢献として認められた．

暗号応用が一区切りとなった後，野呂，横山が 2000 年 9 月，2000 年 12 月にそれぞれ大学に転出し，Risa/Asir の開発は野呂の転出先である神戸大学に本拠を移した．現在は神戸大学高山教授が主催する OpenXM Committers[32] という組織の中で野呂，齋藤を中心に竹島も参加して開発が続けられている．

2000 年 9 月に野呂が神戸大学に転出する際に，Risa/Asir のソースも公開され[33]，一般の開発者が協力できるオープンソース型の開発体制に移行した．この体制により Windows 版，Unix 版ともほとんど対等な機能性能が維持されるようになった．

3.4.2 Risa Consortium/Risa Conference

1995 年 6 月 6 日には Risa の初期ユーザによって Risa Consortium が設立された．代表は 愛媛大学 野田 教授[34]，幹事は 神戸大学 高橋 助教授[35] と上智大学 齋藤 助手[36]

[30] 桂 重俊．東北大学 名誉教授．数式処理学会 名誉会員．グレブナー基底の標準ベンチマーク問題である Katsura-n 方程式の提出者．先生は方程式を パラメータ $n = \infty$ において解いたことを業績としたい旨表明されているが，数式処理，グレブナー基底研究者には Katsura-n の作者として知られる．

[31] 楕円曲線暗号研究の副産物として，整数論の研究で使用されるモジュラー多項式 $\Phi_p(x, y)$ が 113 以下のすべて（30 個）の素数 p について得られた．数学研究上有用なデータであると判断し，富士通研究所のホームページに公開した．http://www.labs.fujitsu.com/jp/freesoft/modularpoly/

[32] http://www.openxm.org/

[33] 公開時点までの著作権は富士通が保持している．公開後の寄与に関しては OpenXM Committers の各々に著作権がある．

[34] 野田 松太郎．愛媛大学 名誉教授．

[35] 高橋 正．現 神戸大学 発達科学部 教授．

が務めた．同年 9 月 12 日〜14 日には，神戸大学滝川記念館で 88 名の参加者を得て，第 1 回 Risa Consortium ワークショップ（RisaCon）が開催された．この参加者に対して Windows 版が初めて一般配布された．1996 年には，愛媛大学工学部（第 2 回，1996 年 3 月），富士通沼津工場 富士フォーラム（第 3 回 1996 年 8 月），東北科学技術短期大学（第 4 回 1996 年 10 月）と 3 度のワークショップが開催された．その後第 5 回（1997 年）からは，第 7 回のワークショップが城西大学（1998 年 11 月）で開催された以外は，毎年 3 月に愛媛大学工学部で定期的に開催されるようになり，第 10 回愛媛大学工学部（2002 年）まで回を重ねた．その後幹事役を神戸大学高山教授が引き継ぎ，名称を Risa/Asir Conference と改めて神戸大学で毎年開催されることとなった．なお，第 5 回および第 6 回ワークショップの研究発表は城西大学紀要として出版されている．

このワークショップは当初より，Risa に限らず数式処理に関わる話題全般について，自由に知見や意見を発表/交換することができる，数式処理ユーザの交流の場として提供されており，この精神は Risa/Asir Conference にも引き継がれている．

3.4.3 Risa/Asir の特徴

Risa の特徴は，因数分解（整数上，代数拡大体上），最大公約多項式計算，グレブナ基底計算などの多項式演算の高速性，高性能の陰函数の描画機能，Big Float の計算に Pari システムを結合していることなどである．

ユーザにとって機能不足を感じる点としては，多項式以外の数式（$\log x$, $\sin x$ などの超越函数や $f(x,y)$ のような不定の函数）に対する演算機能が少なく，数式の簡約化機能やパターンマッチングによる変形をサポートしていないこと，陰函数のグラフ描画ができる以外はグラフィックス機能がないことなどがある．また，不定積分は有理函数の不定積分しかサポートされていない．これらの点は，現在の Risa が多項式を中心としたデータ構造を採用しており，一般の数式に対応するだけのフレキシブルさを持たないことが大きな原因である．この最後の点に関しては，Asir への入力テキストの parse tree を Asir 自身のデータとすることができるような言語メカニズム（Lisp にあるような quote 関数を用いて関数の evaluation を停止する機能）を導入し，それをベースに項書換えシステムなども取り入れようとする計画が，高山教授のイニシアティブの下に進行中である．

[36] 齋藤 友克．現（株）アルファオメガ代表取締役社長．

3.4.4　グレブナー基底

Risa のグレブナ基底パッケージについて，少々付け加えておきたい．1991 年に数式処理グループに加わった下山武司[37]が 1992 年に多項式のグレブナ基底計算を Asir で書き，野呂が下位部品を C でコーディングしたところ，当時でおそらく世界最高速といえるグレブナ基底計算の実装ができた．それ以来，Risa にインプリメントされたグレブナ基底計算パッケージは，自己記録を更新しつつ常に世界最高速を誇っていた．

ところが 1997 年に，パリ第 6 大学の J-C.Faugere 博士のグレブナ基底専用プログラムが驚異的な高速度を達成したとの報告が届いた．ベンチマークは Faugere 博士のホームページ（http://posso.lip6.fr/ jcf/）に公開されていたが，算法が公表されていなかったためどういう方法を用いたのか確認できないでいた．その後，これは Buchberger 算法とは異なる方法によるグレブナー基底の新しい計算法[38] であり，確かに高速に計算できると確認され，Risa/Asir にも実装されている（関数 `dp_f4_main` 他）．

1996 年には，Risa のグレブナ基底計算の実力を示す良い機会があった．前年にネットニュース投稿した「Risa を応用してあなたの問題を解決してみませんか？」との呼びかけに応じて McKay 教授[39] が持って来た問題は，当時としては，極端に大きな問題であった． それは，replicable functions of odd level をすべて決定するという数学の問題で，見掛けは 12 個の変数についての無限個の多項式制約条件を解くことであったが，McKay 教授と野呂の議論を経て，最終的には 4 変数，20 本の方程式を解くことに帰着された．方程式を構成する多項式の次数は 7〜17 次，項数 64〜1183，係数の大きさ 10 進 10 桁程度であり，Risa/Asir でも解けるかどうか分からなかった．計算結果のグレブナ基底（全次数逆辞書式順序）は，51 本の多項式，次数 5〜10，項数 59〜94，係数の大きさ 10 進 50 桁〜100 桁となり，このグレブナ基底に対して素イデアル分解 [67] を適用することによって，最終的な連立方程式の解として全部で 72 個の解を得た．これにより replicable functions of odd level は 72 通りあると決定することができた．この 72 通り（すべて有理数） は既に別の方法によって発見されてはいたが，代数拡大体上に（有理数以外の） 他の解があるかどうかは未確認であった．この 72 個の有理解以外に（複素数の範囲で捜しても）解がないことは Risa の計算によって初めて確認された．最初の計算は Ultra Sparc 170MHz を用い，所要メモリは 460MB，所要時間は 24.5 日であったが，その後のグレブナ基底計算の改良[40]により P6-200MHz で約 9.5 日，所要メモリ 50MB で済むようになった [44].

グレブナ基底は実にさまざまな応用が可能[41]で，研究者/技術者への普及も進んだた

---

[37] 1997 年より暗号研究に専念.

[38] F4 アルゴリズムと呼ばれている.

[39] John McKay, Concordia Univ. Canada.

[40] 項の指数ベクトルに対して適切に選んだ重み付けで項順序を決めることにより，さらに高速に計算される．計算機環境は不詳であるが，木村の報告 [108] によれば，2003 年 12 月時点で 2931 秒で計算できたとある.

め，最近（2004 年）では Mathematica や Maple などの商用システムでの実装も相当強力になっている．そのため代数方程式に対して気軽に使えるようになっている[42]．

姫路工業大学の関口博士[43]の著書，「多面体の数理とグラフィックス」[128] にはグレブナ基底の面白い応用が紹介されている．そこでは Risa と Reduce，Mathematica を使って，Zalgaller 多面体の類似物を作りだしグラフィックスとして見せている．この作業に，複数の数式処理システムを利用したことについて，関口博士はつぎのように述べている．「数式処理システムにはそれぞれの特性がある．Asir には強力なグレブナー基底の計算プログラムが内蔵されており，Mathematica のグラフィックス機能は大変便利である，といったように．便利な機能を使おうとしたらこのようになってしまった，というのが実情である．」関口博士の姿勢は明確である．自分の仕事にとって最善のものがあれば躊躇なく利用して成果を得るという姿勢である．このように苦労を厭わず異種システムを併用する必要があるユーザのためにも，異種数学システム間のインターフェースの標準化を考え，実験を進めていたが，今では，Risa/Asir を高山教授の提唱する OpenXM プロジェクトの一環として開発することにより，結実しつつある．

このほか Risa/Asir では，グレブナ基底計算を利用して，準素イデアル分解や素イデアル分解 [67]，代数拡大体上の因数分解を利用した Galois 群の具体的計算 [3]，Kan-sm1 との接続による b-function の計算 [52, 51]，といった数学への応用で 1990 年代の数式処理の世界をリードしていた．

3.4.5 実代数処理—実世界問題へのアプローチ

筆者は数学ばかりでなく，より広く数式処理の新しい応用が拓けることを願っている．工学では扱う変量の取る値は実数であることがほとんどであり，扱う系を記述するには等式ばかりでなく不等式も現われる．数学の教育では応用が見える形での数学概念の導入が効果的と言われるが，応用先である工学，経済学，生物学などで多く現われる数量は実数であり，実数が適切に扱えるかどうかは，数式処理システムにとっても重要な問題である．

グレブナ基底で扱える範囲は等式の制約[44]であり，その解は複素数上で考えるため，不等式を含む場合や，変量の実数性を直接に扱うことができない．実数上の問題を扱

[41] 余談であるが，現在実施されているほとんどの応用は Buchberger の初期の論文で指摘されているといわれている．

[42] 「Risa には線形方程式を解く関数はないのか？」とよく質問を受けるが，「線形方程式も代数方程式の一種なので，グレブナ基底を使って下さい．」ということにしている．ただし，グレブナ基底の中の注目する 1 変数（1 次）多項式 $ax+b$ から，その零点 $-b/a$ をとり出すことは，Risa/Asir ではユーザの作業となる．

[43] 関口次郎．現東京電気通信大学教授．

う方法として，QE（quantifier elimination）という理論がここ数年の間に実用化されて来た．

数式処理グループのひとり穴井は，1997 年に特殊ケースの QE アルゴリズムのひとつを Risa にインプリメントした．この仕事は穴井と屋並による Maple 上の実代数処理系 SyNRAC[4] に引き継がれ，種々の QE 手法が使えるプラットフォームとして整備されつつある．また，ロバスト最適設計への応用ツールや生体反応系の知見発見ツール [53, 54] などへの応用も開拓されつつある．また，前述したように教育への数式処理の導入を考えた場合にも，QE が効果的に活用できる可能性は大きい．

QE については，数値と数式の融合計算などの新しい研究成果を取り入れ，効率を改善していくことによって，最適化問題をはじめとする種々の新しい工学応用とともに，教育への応用も拓けていくものと確信している．

[44] 多項式 P で表されるある量が 0 でないという条件（$P \neq 0$ で表される）は，いわゆるサチュレーション手法により，新しい不定元 τ と多項式 $\tau P - 1$ を追加することによって（$\tau P - 1 = 0$ としたことに相当），グレブナ基底計算の枠組みに取り込める．

第4章　Risa/Asirの応用

4.1　多項式問題に関する High-Quality Computing

この節では，Risa/Asir の高性能グレブナ基底計算パッケージを利用して数値計算では困難な多変数連立代数方程式に帰着される問題[1)]を扱う．本来の問題は，8 変数 30 本の等式制約条件の下に，すべての実数解を求める問題である．数値計算ではこのような過剰制約系は直接には扱うことができない．原論文で解いたとされる方程式は 8 本に減らされているが，選択を誤っており，無限個の解（零次元解は存在しない）をもつ系となっている．以後，この原論文を引用した多くの論文はその誤りをそのまま踏襲し，それらの 8 本の方程式系からは本来決定できるはずのない真の解が得られたとしている．

このトピックには，実際上は重要な問題であるにも拘らず，数値計算では正面からはとり上げられずに暗黙のうちに処理される次のような付帯問題が浮彫りにされている．すなわち，独立な方程式系の選択，解の存在判定，零次元か否か，収束や発散が数値誤差によるものか否か，などの臨界的な状況の看過が引き起こす問題である．本節の例は，数値的には対処が困難なこのような問題が，特別の扱いをせずとも誰にでも安全に処理でき，正しい解決が得られるという，数式処理の優位さを示す好例である．

性能指標は 1996 年当時のものである．

4.1.1　科学技術計算における計算の品質

科学技術計算は，シミュレーションや解析，設計，制御，製造などの様々な領域における問題解決法を提供することによって，今日のハイテクノロジー世界の発展に本質的な役割を果たしてきた．電子計算機時代の到来以来，科学技術世界においては，数式処理（記号的/代数的計算）技術ではなく，数値計算技術が圧倒的に優位な方法論として確立されている．数値計算は，浮動小数点数形式で実数を処理することによって，より高い計算効率とより大規模な問題を解くこととを目標として来た．しかし，伊理が数値計算世界の一般的なこの趨勢について再考を促す論文 [103] を発表した．

[1)] 文献 [48] で紹介した内容の一部である．この他に，多項式問題に関する筆者の発表がいくつかある [141, 142].

伊理はつぎのように言う.「(計算機の能力は,)『より速く，より大きな問題を』解くためだけに向けるのでなく（High Performance Computing =**HPC**),『より高品質の』(High-Quality Computing = **HQC**,，そして『より信頼性のある』(High Reliability Computing =**HRC**）の方向にも向けるように努力したい.」

この節では，高品質計算（HQC）に向けての筆者らのアプローチを紹介する．そのアプローチでは，数値計算の代替として，あるいは数値計算と併用して，代数計算（数式処理）を使用する．ここではとくに多変数多項式による連立代数方程式系の解法に焦点を当て，数式処理システム Risa/Asir での処理性能を例として，代数計算の実効性を示す.

4.1.2　高品質計算（HQC = High-Quality Computing）

筆者等は計算の品質を重要視している．効率という評価基準以外に，忠実性，正確さ，精度，信頼性，安定性，頑強性，運用性，保守性，適応性，費用対効果など，さまざまな性質で表現される**品質**を考慮する必要がある．これらの評価基準は数値計算でも考慮されていることは承知しているが，改善の余地もあると考えている．代数計算（数式処理）は高品質，とりわけ忠実性，正確さ，精度および信頼性を追求する異なるアプローチである.

特に精度という観点から，数値計算と代数計算それぞれのアプローチの違いを考察してみよう.

上記の論文で伊理は数値計算の枠組み内で精度を保証することを目的として**区間演算**を使用するよう主張している．工学的立場からは，所与のあるいは計算途中で現われる実数値はすべて誤差を含むものとして取り扱われる．したがって，実数値を十分に大きな有効桁，たとえば 64 ビット，を持った浮動小数点数値として表現することは自然なことと見做される．浮動小数点数値の精度，とくに浮動小数点数値の四則演算による丸め誤差の伝搬，を評価する目的で，区間演算では真の値を含むと仮定される区間によって実数値を表現する．このアプローチは数多くの数値計算問題に適用されてきた．また，解の存在証明や唯一性証明も研究されてきた [170, 1].

他方，代数的アプローチはまったく正反対の思想に基づいて実数を取り扱う．代数的アプローチでは，数値は**正確 = exact** なもの，すなわち演算において誤差の生じる余地のないものとして扱われる．基本的な技術としては **Bignum** と呼ぶ数システムがあり，この数システム上で，計算機のメモリが許す限り[2)]，任意に大きな整数に対する四則演算が正確に実行される．Bignum を使用することでより複雑な数システムをも扱うことが可能になる．たとえば，有理数は整数の分子と分母の組として表現され，

[2)] このために，代数的計算の計算量の大きさは代数方程式に現われる変数の個数と方程式の次数や個数ばかりでなく，係数の表現サイズによっても計測される必要がある.

$\sqrt{2}$ のような代数的数は，有理数を係数とする既約多項式の根として表現される．さらには，円周率 π や自然対数の底 e などの超越数はそれ自体を記号として取り扱う．方程式や公式中の数式に現われる個々の要素は，そのような正確な（exact）係数を使って記号的に表現され操作される．そして，代数的計算が成功裡に適用できたならば，得られる結果は正確で完全な解である．既約性や多重性のような代数的な構造/性質をはじめ，問題の持つ多くの数学的性質は代数式の中に忠実に反映されている

4.1.3 多項式問題

ここでの多項式問題は数学的には与えられた変数達の間に成り立つ多項式方程式として表現されるものをいう．そのような問題は数学や理工学，それらの産業/社会への応用の問題として幅広く現われる．

たとえば，電気回路における電流と電圧の関係のような物理現象は 1 連の多項式方程式によってモデル化できる．この場合には多項式に現われる変数達は電気回路中の多数の場所における電流や電圧を表しており，多項式の係数は回路要素の抵抗やキャパシタ，インダクタンスなどの値を表している．幾何の証明問題においては，直線や円，放物線などの幾何学的対象物はそれらを構成する各点の座標を変数とした一連の多項式方程式によって記述される．

最初のモデル化では多項式問題の形をとらないけれども，適切な変換によって多項式問題に変換されるような重要な問題も数多くある．たとえば，ある種の偏微分方程式のソリトン解を求める問題が幾種類かの基底函数を導入することで，多項式問題に帰着することが知られている．この場合には，本来のパラメータと基底函数による展開の未知係数とが多項式方程式の変数となる．

線形方程式系は多項式方程式系の最も簡単な場合である．多くの問題が線形方程式系でモデル化される．これは，線形方程式系によって原問題が忠実にモデル化されるという理由による場合もあるが，よりありがちな理由は，最初の近似としては線形方程式系が有用であるというものである．こうして，線形方程式系を解くことが計算の主要な分野となっている．この目的のために多くの数値計算アルゴリズムが開発されており，それらは非常に多数の変数や方程式をもつ線形方程式系を解くことができる．

しかしながら，高次の多項式方程式系の場合には数値計算に基づく方法は多くの困難に直面する．例としてニュートン法を考えよう．ニュートン法は多変数多項式方程式系を解くための代表的な方法である．ニュートン法の長所は解（の近似値）への収束の高速性にある．ただしそれには，初期値が**真の解に十分近く選ばれていれば**という条件が付いている．1 変数の場合でも必ずしも簡単ではないが，多変数の場合にはそのような初期値を用意することは多くの場合に困難であり，それゆえに，解の全てを完全に求めることは一般には難しい．

4.1.4　多項式問題の代数的解法

多項式問題を解くために代数的解法が開発された．多項式方程式は多項式環のイデアルを定義するものとして取り扱われ，代数的な消去法により簡単な（一次式の場合も含む）多項式からなる多項式系へと変換される．

しかしながら代数的な方法は実用問題への効果的な適用がなかなか進まなかった．その理由は，1990 年代の半ばまでにおいては，代数的解法のためのソフトウェアが一般的に利用される環境が整っていなかったことと，ソフトウェア自体が実際問題に耐えられるだけの能力に欠けていたからである．この状況は 2005 年の現時点では相当変わっている．新しいアルゴリズムの工夫により大幅な効率化が達成できたことと相まって，計算機パワー自体が格段に向上し，メモリを多消費する代数的方法がパーソナルコンピュータでも軽々と実行可能になったからである．

筆者等も代数的アルゴリズムをより効率化するべく研究を重ねてきたし，それは現在も継続している．

筆者等が開発した数式処理システム Risa/Asir は，時代を追って改良されて来た効率的なグレブナー基底計算アルゴリズムをインプリメントしている．このグレブナー基底計算アルゴリズムは純粋数学の問題，ガロア群の計算 [3, 5] や準素イデアル分解 [67]，**b**-functions の計算 [52, 51]，replicable functions の決定 [44] など，従来の方法では解決することが困難であった問題を解くために適用され成功を収めている．ごく最近では量子コンピュータの物理的実現のひとつの鍵となる量子回路（CNOT）の存在を計算により厳密に証明 [33] することにも成功した．

以下，数値積分公式を導くための 8 変数の多項式問題を例として，代数的方法の有効性を示す．

4.1.5　多項式問題例—Symplectic 数値積分公式

Risa/Asir の紹介のための例題を検討していた折り，吉田の論文 [169] が目に止まった．その論文は symplectic 積分と呼ばれる安定な数値積分公式を決定する問題に関するもので，8 変数からなる 8 本の多項式方程式が示されていた．その多項式方程式を Risa/Asir に入力し，解を求めてみたところ筆者等はその結果に当惑せざるを得なかった．Risa/Asir が 1 次元の解—数値計算では決して求めることはできない—を返したからである[3]．その 1 次元解を適当に選んだ 2 変数の空間に射影したグラフ[4] を図 4.1 に示す．

[3] 吉田論文 [169] が 0 次元の解のいくつかを数値的に求めたことを報告していたし，筆者等も当然 0 次元の解を期待していた．

[4] このグラフ自体通常のグラフィックスソフトでは描画が困難であり，第 5 章で紹介する Risa/Asir の陰函数描画機能でなければ到底表示することはできなかったものである．

図 4.1: 文献に基づく問題 2 の 1 次元解の yz 平面への射影
（水平方向が y，垂直方向が z.）

文献に見る多項式問題

天体力学ではハミルトニアン系 ($\frac{dp}{dt} = -\frac{\partial H}{\partial q}$, $\frac{dq}{dt} = \frac{\partial H}{\partial p}$ with $H(p,q) = T(p) + V(q)$) の数値解における長期間安定性（すなわち，真の解がもつのと同じ性質である symplectic property）をもつ数値積分公式が必要とされる．ルンゲ-クッタ積分公式やオイラー積分公式のような良く知られた数値積分公式はこの性質を持たず，時間の経過とともに積分誤差が集積してしまう．

文献 [169, 39] に従えば，symplectic 積分公式はつぎのような形式を持つ．初期点を (p, q) とすると，短時間 τ 経過後の点 $(p(\tau), q(\tau))$ はつぎの漸化式に従う symplectic 積分スキームにより計算される．

$$q^{(0)} := q, p^{(0)} := p, q(\tau) := q^{(k)}, p(\tau) := p^{(k)}, \tag{4.1}$$

$$\begin{aligned} p^{(i+1)} &= p^{(i)} - d_i \frac{\partial V}{\partial q}|_{q^{(i)}}, \\ q^{(i+1)} &= q^{(i)} + c_i \frac{\partial T}{\partial p}|_{p^{(i+1)}}. \end{aligned} \tag{4.2}$$

ここで，未知の係数の個数を決める整数 k と未知の係数 c_i, d_i $(i = 1, 2, ..., k)$ はつぎの問題を解いて定める．

問題 1 (symplectic 積分係数の決定)
A と B とは非可換な演算子であるとする．与えられた整数 n —これが数値積分公式の次数に相当する—に対して，整数 k と実数 c_i, d_i $(i = 1, 2, ..., k)$ をつぎの等式を満すように定めよ．

$$\begin{aligned} &\exp[\tau(A + B)] = \\ &\quad [\prod_{i=1}^{k} \exp(c_i \tau A)\exp(d_i \tau B)] + O(\tau^{n+1}). \end{aligned} \tag{4.3}$$

この指数函数の等式の両辺を展開し，非可換演算子からなる対応する項の係数を等値することによって，$c_1, ..., c_k$ および $d_1, ..., d_k$. に関する一組の多項式方程式が得られる．文献 [169] では $n = 4$ かつ $k = 4$ に対する symplectic 積分に関して，つぎの多項式方程式を示している．（読みやすくするために 8 個の変数 $c_1, ..., c_4$ と $d_1, ..., d_4$ はそれぞれ a, b, c, d と w, x, y, z, に名前替えした．）

問題 2 (文献掲載の不適切な方程式)
つぎの連立方程式を変数 a, b, c, d, w, x, y, z について解け．

$$\begin{aligned} f_1 &= a + b + c + d - 1 = 0 \\ f_2 &= w + x + y + z - 1 = 0 \\ f_3 &= bw + c(w + x) + d(w + x + y) - \frac{1}{2} = 0 \\ f_4 &= bw^2 + c(w + x)^2 + d(w + x + y)^2 - \frac{1}{3} = 0 \\ f_5 &= bw^3 + c(w + x)^3 + d(w + x + y)^3 - \frac{1}{4} = 0 \end{aligned}$$

$$
\begin{aligned}
f_6 &= a^2w + (a+b)^2x + (a+b+c)^2y \\
&\quad +(a+b+c+d)^2z - \frac{1}{3} = 0 \\
f_7 &= a^3w + (a+b)^3x + (a+b+c)^3y \\
&\quad +(a+b+c+d)^3z - \frac{1}{4} = 0 \\
f_8 &= abw^2 + ac(w+x)^2 + ad(w+x+y)^2 \\
&\quad +bcx^2 + bd(x+y)^2 + cdy^2 - \frac{1}{12} = 0.
\end{aligned} \tag{4.4}
$$

文献は上記システムを Neri [42] から取ったとし，Forest *et al.* [27] の得た解

$$
\begin{aligned}
&a = d = \frac{1}{2(2-2^{\frac{1}{3}})},\ b = c = \frac{1-2^{\frac{1}{3}}}{2(2-2^{\frac{1}{3}})}, \\
&w = y = \frac{1}{2-2^{\frac{1}{3}}},\ x = \frac{-2^{\frac{1}{3}}}{2-2^{\frac{1}{3}}},\ z = 0.
\end{aligned} \tag{4.5}
$$

を示している．

不適切な方程式の代数解

筆者らは，上記文献の 8 変数連立代数方程式系は Risa/Asir の能力証明の例題のひとつとしてふさわしいと考え，この問題 2 を Risa/Asir で解いてみた．驚いたことに，Risa/Asir はわずか 3 分間ほどで非常に複雑で大量の数式[5)]を返したのである．この時使用した計算機は，CPU が 200MHZ Pentium Pro の FMV6200T2 で，OS は FreeBSD であった．

Risa/Asir の結果を信じる限り，問題 2 の連立方程式は期待される 0 次元の解[6)]は持てず，1 次元の解[7)]をもつ．

この 1 次元解を yz 平面に射影したグラフが先に示した図 4.1 である．その射影の方程式を $g(y,z) = 0$ とすれは，多項式 $g(y,z)$ は巨大な式，式（4.6）となる．

$$
\begin{aligned}
g(y,z) = (&50761728z^7 - 102021120z^6 + 85287168z^5 \\
&-38631168z^4 + 10285056z^3 - 1617408z^2 \\
&+139968z - 5184)y^{14}
\end{aligned}
$$

[5)] Risa/Asir が返した数式とは，この場合にはグレブナ基底のことである．グレブナー基底が得られれば解を求めることは容易である．

[6)] 0 次元の解とは，8 次元空間の孤立したいくつかの点であり，その個数は有限個となる．

[7)] 1 次元の解とは 8 次元空間中の曲線であり，無限個の解があることになる. 線形方程式では不定の場合の一部に相当.

$$
\begin{aligned}
&+(426622464z^8 - 1249385472z^7 + 1530814464z^6\\
&-1021579776z^5 + 407576448z^4 - 100113408z^3\\
&+14883264z^2 - 1233792z + 44064)y^{13}\\
&+\cdots\\
&\text{(164 intermediate terms are omitted)}\\
&+\cdots\\
&+(-1026432z^{16} + 8957952z^{15} - 40746240z^{14}\\
&+117348480z^{13} - 220595616z^{12} + 279410688z^{11}\\
&-247641408z^{10} + 158813568z^9 - 75551688z^8\\
&+27102240z^7 - 7396056z^6 + 1537584z^5\\
&-241654z^4 + 28096z^3 - 2304z^2 + 120z - 3)y\\
&+46656z^{15} - 419904z^{14} + 1578528z^{13}\\
&-3256416z^{12} + 4111776z^{11} - 3392928z^{10}\\
&+1917864z^9 - 765720z^8 + 219240z^7 - 45000z^6\\
&+6498z^5 - 630z^4 + 37z^3 - z^2. \qquad (4.6)
\end{aligned}
$$

この多項式は 212 の項から構成され，全次数は 21，y に関しては 14 次 z に関しては 18 である．付録 A に省略なしの多項式を掲載した．

4.1.6　真の問題とその代数的解

このような期待とは異なる結果を得たため，筆者等は原問題の方程式（4.3）を吟味することにした．すると，τ について 4 次の項まで展開すると 8 個の変数についての 30 個の多項式方程式が出現することが分かった．最初にこの問題を解いた研究者達は，彼らの物理に関する洞察に基づくなどして，この 30 本の式から，必要ならば手計算[8)]も援用して，8 本の独立な方程式を導かなければならなかったと考えられる．そして最終的には数値計算で結果を得た．我々は Risa/Asir などの数式処理システムを利用して，容易にこの手続きを追跡することができる．筆者等は原方程式（4.3）から 30 本の多項式方程式を得るために，数式処理システム REDUCE を利用した[9)]．しかる後，Risa/Asir のグレブナー基底計算を使用して 8 本の極大独立集合（30 本の方程式と同じ解を与える独立な方程式）を得た．この極大独立集合を得る簡約化は，先

[8)]むろん数式処理システムも利用したと考えられる．
[9)]REDUCE を利用した理由は，Risa/Asir が非可換代数をサポートしていないためである．

と同じ環境 FMV6200T2 を用いて 2 分で完了した．こうして得られた新しい方程式系はつぎの通りである．

問題 3 (正しい方程式)
つぎの連立方程式を変数 a, b, c, d, w, x, y, z について解け．

$$\begin{aligned}
f_1' &= a+b+c+d-1=0,\\
f_2' &= w+x+y+z-1=0,\\
f_3' &= a(b+c+d)w+(a+b)(c+d)x+\\
&\quad (a+b+c)dy-\frac{1}{6}=0,\\
f_4' &= a(w+x+y+z)^2+b(x+y+z)^2+\\
&\quad c(y+z)^2+dz^2-\frac{1}{3}=0,\\
f_5' &= a^2w+(a+b)^2x+(a+b+c)^2y+\\
&\quad (a+b+c+d)^2z-\frac{1}{3}=0,\\
f_6' &= ((c+d)bx+(b+c)dy)w+cdxy-\frac{1}{24}=0,\\
f_7' &= bw(x+y+z)^2+c(w+x)(y+z)^2+\\
&\quad d(w+x+y)z^2-\frac{1}{12}=0,\\
f_8' &= (b+c+d)^3w+(c+d)^3x+d^3y-\frac{1}{4}=0.
\end{aligned} \tag{4.7}$$

この方程式系にグレブナー基底計算に基づく素イデアル分解 [67] を適用すると，方程式系は新しく 4 つの方程式系 V_1, V_2, V_3, V_4 に分解[10]される．この分解は FMV6200T2 を用いて 22 秒で得られた．

$$\begin{aligned}
V_1 = \{&6y^3-12y^2+6y-1=0,\\
&z=0, x+2y-1=0, w-y=0, y-2d=0,\\
&y+2c-1=0, y+2b-1=0, y-2a=0\},
\end{aligned} \tag{4.8}$$

$$\begin{aligned}
V_2 = \{&48y^3-24y^2+1=0,\\
&2y+2z-1=0, x-y=0, 2w+2y-1=0,\\
&2y+d-1=0, 4y-c-1=0, 2y+b-1=0,\\
&a=0\},
\end{aligned} \tag{4.9}$$

[10]方程式系の分解とは，分解されて新しくできた方程式系の解の和集合としてもとの方程式の解が表されることをいう．

$$
\begin{aligned}
V_3 = \{&12y^2 - 9y + 2 = 0,\\
&4y - 4z - 1 = 0, 4x + 4y - 3 = 0, 2w + 2y - 1 = 0,\\
&4y - 2d - 1 = 0, 2c - 1 = 0, 2y + b - 1 = 0, a = 0\} \qquad (4.10)\\
V_4 = \{&6y^2 - 3y + 1 = 0,\\
&z = 0, 2x - 1 = 0, 2w + 2y - 1 = 0, y - 2d = 0,\\
&2y - 4c + 1 = 0, y + 2b - 1 = 0, 2y + 4a - 1 = 0\}. \qquad (4.11)
\end{aligned}
$$

素イデアル分解の理論により，すべての解はこれら 4 つの方程式系の解の和集合となる．

これらの新しい方程式系を観察するとグレブナー基底を用いる方程式解法に特徴的な長所が見て取れる．

1. どの方程式系にも変数 y のみからなる方程式が 1 つだけ存在する．
2. どの方程式系においても，y 以外の変数は，y の多項式として直ちにかつユニークに表現できる．

したがって，残された作業は y についての 1 変数多項式の解を求めることのみとなる．目下の問題については，解くべき 1 変数多項式が 2 次あるいは 3 次であるので，その根を平方根や 3 乗根を用いて正確に表現することができる[11)]．

ここでの代数的アプローチによって最終的な結果，すなわち y の異なる 10 個の根のそれぞれに対応して 10 組の解（実解 2 組，虚解 8 組）が得られた[12)]．まず，方程式系 V_1, V_2, V_3, V_4 のそれぞれに対して，y の値はつぎのとおりとなった．

$$V_1 \ : \ \frac{2^{\frac{2}{3}} + 2\cdot 2^{\frac{1}{3}} + 4}{6}, \frac{2^{\frac{2}{3}} + 2\cdot 2^{\frac{1}{3}} - 8 \pm (2^{\frac{2}{3}} - 2\cdot 2^{\frac{1}{3}})\sqrt{-3}}{12}. \qquad (4.12)$$

$$V_2 \ : \ \frac{-2^{\frac{2}{3}} - 2\cdot 2^{\frac{1}{3}} + 2}{12}, \frac{2^{\frac{2}{3}} + 2\cdot 2^{\frac{1}{3}} + 4 \pm (2^{\frac{2}{3}} - 2\cdot 2^{\frac{1}{3}})\sqrt{-3}}{24}. \qquad (4.13)$$

---

11) 根号を用いて表現できない多項式の場合（非可解の場合）にもその近似根を求めることはできる．どんなアルゴリズムでも使用できるが，有限桁数の数値を用いる場合は多項式の次数が高い場合や，係数の表現長が大きな場合には殆んどの場合適用できないことに注意を要する．たとえ任意精度計算が可能な場合でも単純なニュートン法では失敗する．有理的演算を用いた Sturm 列による根の分離に基づく代数的な方法は，このような大規模な 1 変数多項式の根を求めることに適している．（虚）複素根を求める問題は，代数的な変換により実根を求める問題に変換可能であるが，そのためには別のコスト（計算量）が必要である．

12) 根号による表現は一般にはユニークではない．ここでは 2 本ある y の 3 次多項式の根は，Forest による解の表現に合わせて，2 の 3 乗根と -3 の平方根とを用いて表現した．ガロア群計算に基づいた根の根号表現アルゴリズム [3, 5] を用いた Risa/Asir のプログラムによる根の表現は，Forest のものとは異なる 2 つの添加元，1 の虚立方根のひとつ ω および，$\alpha = \{(60\omega - 51)/2\}^{\frac{1}{3}}$，を用いたものとなる（方程式 V_1 の変数 y に対する解の場合）．

$$V_3 \quad : \quad \frac{9 \pm \sqrt{-15}}{24}. \tag{4.14}$$

$$V_4 \quad : \quad \frac{3 \pm \sqrt{-15}}{12}. \tag{4.15}$$

この y の値それぞれに対する他の変数の値は単に代入によって得られる．

つぎに 2 組の実解のみを示す．最初の実解は Forest 等 [39] の得た結果と一致する．Forest 等の結果において分母を有理化することで一致が確かめられる．

$$\begin{aligned}
a = d &= \frac{1}{2}y = \frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 4}{12} \approx 0.675604, \\
b = c &= \frac{1}{2}(1 - y) = \frac{-2^{\frac{2}{3}} - 2 \cdot 2^{\frac{1}{3}} + 2}{12} \approx -0.175604, \\
w = y &= \frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 4}{6} \approx 1.35121, \\
x &= 1 - 2y = -\frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 1}{3} \approx -1.70241, \\
z &= 0.
\end{aligned} \tag{4.16}$$

もう一つの実解はつぎのとおりである．

$$\begin{aligned}
a &= 0, \\
b = d &= 1 - 2y = \frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 4}{6} \approx 1.35121, \\
c &= 4y - 1 = -\frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 1}{3} \approx -1.70241, \\
w = z &= \frac{1}{2}(1 - 2y) = \frac{2^{\frac{2}{3}} + 2 \cdot 2^{\frac{1}{3}} + 4}{12} \approx 0.675604, \\
x = y &= \frac{-2^{\frac{2}{3}} - 2 \cdot 2^{\frac{1}{3}} + 2}{12} \approx -0.175604.
\end{aligned} \tag{4.17}$$

2 組の解は変数 a, b, c, d と z, y, x, w との間にこの順序で対称的な関係を示している．この対称性はその他の解（虚解）すべてについても存在している．

2 組の実解は $\mathbb{R}^8$ の異なる 2 点であり，その yz 平面への射影を図 4.2 において “●” で示した．先に示した不適切な問題の 1 次元解（図 4.1）との対応をとるために，同じグラフの上にプロットしてある．

図 4.2: yz 平面に射影した正しい方程式の 2 つの実解（● で示す.）

4.1.7　例から学ぶこと

筆者等は，例とした文献に見る誤りが単に不注意から生じたものではなく，むしろ，手計算と数値計算のみにたよる，従来のアプローチがもつ越えがたい限界に起因していると考える.

多項式方程式系を設定するに際して，従来の方法ではつぎのような基本的な疑問に答えることができない.

1. 方程式系は解を持つか? 数値的方法は方程式系が解をもつか否かを決定できな

い．この質問を無視して解を持たない方程式系に数値的方法を適用したとすれば，解があるか否かの情報を含めてなにも得られない．

2. 仮に方程式系が解をもつとしよう．その解は 0 次元か? 数値的方法は（原理的に）0 次元，つまり，方程式が有限個の解を持つ場合にしか適用できない．高次元解を含む場合には解（のひとつ）を求める手続きは一般に収束せず，しかもそれが 0 次元であること自体も検出できない [13)]．

3. 仮に方程式系が 0 次元であるとしよう．（多数ある方程式のなかで）最大独立集合はどれか? 数値的方法は，方程式（等式）の個数が未知数の個数に丁度一致する場合にのみ適用できる．唯一の方法は，前節で考察した例の場合でいえば，30 本の等式の中から 8 本を選ぶことをしらみつぶしに繰り返して，求解を繰り返すことである．しかし，求解を試みる場合の多くでは，解に収束しないのか，解がないのか，無限にあるのかさえ決定できす，適当なところでその選択を捨てるしか方法がない．

たとえ多項式方程式系が適切に設定できたとしても，数値的方法にはつぎのような制約や不確実性がある．

1. 数値的方法には初期値が不可欠であるが，与えた初期値がいずこかへ収束するか否かを保証する方法は存在しない．

2. 上記には，数値的悪条件の場合を高次元解の場合から区別できないこと，ならびに，プログラムの不適切さ（BUG あるいは想定外の状況）に起因した非収束と数値的悪条件に起因する非収束とを区別できないこととを含む．

3. ひとつの解が幸運にも求まったとしても，他の解についてはなんの情報も得られない．そして，理論的にも実用的にも解を全て過不足なく求めることはできない．

4. 正確（exact）な有理計算に基づいていない数値計算による方法では解の精度を保証する実際的な方法が存在しない．

筆者等は，多変数多項式方程式系を解くためには数値的方法は不適切であり，代数的方法を採用すべきであると結論する．数値的方法は，実際には保証し得ない多くの前提条件の下で適用可能なものであり，本質的に**ヒューリスティック**で，職人芸的な性質をもっている．それゆえ一般ユーザは，試行錯誤過程に多くの計算資源（人的な知的労働資源を含む）を必要とする．このゆえに筆者等は，数学理論に基づく，かつ効率化が極めて進んできた代数的方法（= 数式処理）の採用を主張する．

[13)]前節で考察した方程式系では，$\mathbb{C}^8$ の全てにおいて Jacobian が消えるため，ニュートン法（微分やその代替を利用する方法）は使用できない．

4.2　三斜内容三圓術（Malfattiの問題）－和算の数式処理

「三斜内容三圓術」は和算家安島直圓が「不朽算法（1799 年）」に論じ，解答を与えた幾何の問題である．西洋では Malfatti の問題（1803 年）として知られる．現代の数学の問題としては，代数的整数論の問題と捉えることができ，有理数上の有理函数体の代数的拡大体の表現を求めることに帰着する．しかし数学的には自明な問題とは言え，有理数体上の代数的拡大体を具体的に扱うことでさえ大変困難な問題であるのに，有理函数体上の代数的拡大体を扱うことは遥かに難しいことは想像に難くない．これを現代の計算機代数でどう扱うことができるか，富士通研究所で開発した数式処理システム Risa/Asir の超高速のグレブナ基底パッケージを利用した解法により示す．

本研究は 1995 年 11 月に数理解析研究所の研究集会で発表し，同所講究録 961 に採録されている [140].

本論文の目的は，文字で表されるパラメータを含むことが多い初等幾何学の問題を例題にして，数式処理システムの能力が現時点でどの程度かを読者に伝えることにある．本文ではグレブナ基底を多用したが，それはグレブナ基底が多変数の多項式を扱う上で大変融通の効く便利な道具であることと，Risa/Asir に実装されているものが高速で十分実用的であることに依っており，個々の計算の特殊事情によっては他の数学的方法が計算上有利な場合もあることをお断りしておく．なお，Risa/Asir は anonymous ftp によりつぎのところから取得できるので，試して見て頂きたい．

```
http://www.asir.org/
http://www.math.kobe-u.ac.jp/Asir/index.html
```

4.2.1　問題の説明

問題

「3 角形の中に 3 個の円を，どの円も互いに他の 2 つの円に外接し，しかも 3 角形の 2 辺に内接するように配置せよ.」

この問題は 3 角形 ABC の内接円の半径を 1 と規格化すると，3 変数＋ 3 パラメータ， 4 制約式（ 1 つはパラメタ間の制約式）の代数制約問題を解くことに帰着できる．x^2, y^2, z^2 をそれぞれ 角 A, 角 B, 角 C の側にある円の半径，$l = 1/\tan(\mathrm{A}/2), m = 1/\tan(\mathrm{B}/2), n = 1/\tan(\mathrm{C}/2)$ と置くと，次の方程式となる．

$$f_1(x, y; l, m) = lx^2 + my^2 + 2xy - (l + m) = 0, \tag{4.18}$$

$$f_2(y, z; m, n) = my^2 + nz^2 + 2yz - (m + n) = 0, \tag{4.19}$$

$$f_3(x, z; l, n) = nz^2 + lx^2 + 2zx - (n + l) = 0. \tag{4.20}$$

ここで，l, m, n はパラメータであり次式を満たす．

$$f_4(l, m, n) \quad = \quad lmn - (l + m + n) = 0. \tag{4.21}$$

この方程式を解き，x^2, y^2, z^2 を l, m, n で表すことが問題である．

既知の解

この問題には多くの解の表現があるが，とくにつぎのような対称性の高い美しい解が知られている．

$$\{x^2 = \frac{bc}{2a},\ y^2 = \frac{ca}{2b},\ z^2 = \frac{ab}{2c}.\} \tag{4.22}$$

ただし，

$$a \quad = \quad 1 + \tan(\mathrm{A}/4) = 1 - l + \sqrt{1 + l^2}, \tag{4.23}$$

$$b \quad = \quad 1 + \tan(\mathrm{B}/4) = 1 - m + \sqrt{1 + m^2}, \tag{4.24}$$

$$c \quad = \quad 1 + \tan(\mathrm{C}/4) = 1 - n + \sqrt{1 + n^2}. \tag{4.25}$$

3 円の半径を表す変数を，パラメタの式として一般的に表すのは結構難しく，特にこのような対称性を持ったきれいな式が知られるまでには 130 年ぐらい掛かったそうである．数式処理でこのようなきれいな解を得ることがここでのひとつの目標である．

方針

1. $\mathbf{Q}$ 上超越的なパラメータ l, m, n は代数的に独立ではなく，$\mathbf{Q}(l, m, n) = \mathbf{Q}(m, n)$ であることに注意する．ここで，m, n は互いに代数的に独立な超越元である．

2. 変数を消去して x が単独で満たすべき多項式 $g(x; m, n)$ を導く．他の変数は x の有理式として表される．

3. 式 $g(x; m, n)$ の分解体を求める．$K = \mathbf{Q}(m, n)$ の拡大体上で 1 次因子に因数分解する．

4. Galois 群を調べ，可解性・作図可能性を判定

5. 根を（平方）根号で表す．

6. どの根がもとの問題の解か検討する．既知の結果とも比較する．

技術要素と問題点

パラメータを含んでいるため**有理函数体とその拡大体上**で演算する必要がある．現時点での高度の数式処理技術を持ってしても，大きな計算量を覚悟する必要がある．

1. 変数消去 ⇒ もっぱらグレブナ基底計算による
2. 1 変数多項式の分解体の計算 ⇒Trager のノルム法による
   - 代数拡大のノルム計算 ⇒ 終結式による（この計算が一番大きい）
   - 基礎体上の因数分解—多変数多項式の因数分解
   - 代数的拡大体上の GCD 計算 ⇒ グレブナ基底による
3. Galois 群の計算— 置換群として表現
4. 代数的拡大体の表現法— 逐次拡大か多根並列添加か
   — Galois 理論，代数的整数論的考察が必要
   — 根号表現— 二重根号外しなど
5. 平面幾何の代数的形式化の限界— 正負判定など

4.2.2 グレブナー基底による変数消去

使用数式処理システムおよび計算機環境はつぎのとおりである．

• Risa/Asir: version 950831 • OS: Linux, kernel version 1.2.8

• HW: FM/V-BIBLO(FMV-475NU/S), i486 DX4 75MHz, 24MB Memory

さて，変数の消去には，終結式や擬除算による Wu-Ritt の方法などがあるが，ここではもっぱら，Risa/Asir の G-base 計算パッケージを利用する．多項式集合 $\{f_1(x,y,l,m), f_2(y,z,m,n), f_3(x,z,l,n), f_4(l,m,n)\}$ の変数順序 $z \succ y \succ x \succ l \succ m \succ n$，純辞書式項順序による G-base（8.78 秒, 827392 Bytes Used）は 22 本の多項式からなり，その中に唯一 x と m,n のみを含む x の 8 次多項式，すなわち，基礎体 $\mathbf{Q}(l,m,n) = \mathbf{Q}(m,n)$ 上の x の最小多項式（x の満たすべき方程式）がある．また多少不正確な言い方ではあるが，y は純辞書式順序の G-base の性質により x の 7 次多項式として表され，z は x と y とによって表されている．

$$\begin{aligned}\mathrm{g}(\mathrm{x}^2;\mathrm{m},\mathrm{n}) = {} & 4(m+n)^2x^8 - 8(m+n)\{nm^2 + (n^2-n)m + 1\}x^6 + \\ & 4\{n^2m^4 + (3n^3 - 3n^2 + n)m^3 + (n^4 - 3n^3 + 3n - 1)m^2 + \\ & \qquad (n^3 + 3n^2 - n)m - n^2\}x^4 + \\ & 4mn(nm-1)\{(n^2 - n + 1)m^2 - (n^2-n)m + n^2\}x^2 + \\ & \qquad n^2m^2(nm-1)^2 \end{aligned} \tag{4.26}$$

これは，x^2 に関する 4 次多項式なので，x^2 を改めて x と置き換え，以後その 4 次多項式 $g(x;m,n)$ を因数分解することが本質的になる．

1. $g(x;m,n)$ は 係数体 $\mathbf{Q}(m,n)$ （$\mathbf{Q}$-係数の m,n の有理函数体） 上既約である．
2. 可解性は明らか（4 次多項式）
3. 作図可能か？
4. 可能ならば平方根号を用いた根の表示を求む．**注：** 4 次多項式の根の公式からは作図できない． $\Rightarrow$ $\sqrt{\ }$, $\sqrt[3]{\ }$, $\sqrt[4]{\ }$ がすべて現れる！

4.2.3 分解体の計算

Trager のノルム法：K を体，α を K 上代数的な元とする．$f \in K(\alpha)[x]$ の K 上のノルム $N_{K(\alpha)/K}(f)$ が無平方のとき，その K 上の既約分解を $N_{K(\alpha)/K}(f) = n_1 \cdot n_2 \cdots n_r$ とおけば，$\{g_i = gcd(f, n_i)\}_{i=1\ldots r}$ は f の $K(\alpha)$ における既約分解 $f = g_1 \cdot g_2 \cdots g_r$ を与える．

以下，$K = \mathbf{Q}(m,n)$ とし，K の拡大体に対して Trager の方法を適用する．
三角形式の準備：x_1, x_2, x_3, x_4 をそれぞれつぎに定義する g_1, g_2, g_3, g_4 の根とする．

$$g_1(x;m,n) = g(x;m,n), \tag{4.27}$$
$$g_2(x;x_1,m,n) = \mathrm{sdiv}(g_1(x;m,n),(x-x_1)), \tag{4.28}$$
$$g_3(x;x_1,x_2,m,n) = \mathrm{sdiv}(g_2(x;x1,m,n),(x-x_2)), \tag{4.29}$$
$$g_4(x;x_1,x_2,x_3,m,n) = \mathrm{sdiv}(g_3(x;x_1,x_2,m,n),(x-x_3)). \tag{4.30}$$

ここに，$\mathrm{sdiv}(f,g)$ は f を g で除した商を与える関数である．
$K(x_1)$ 上の x_2 の最小多項式を以下のようにして求める．

1. $g_2(x;x_1,m,n)$ の変数 x を変数変換 $x \to x + x_1$ して，$g_2(x+x_1;x_1,m,n)$ とし，
2. その拡大 $K(x_1)/K$ におけるノルムを求める．ノルムは体 K 上の多項式となり，その各既約因子は $g_2(x+x_1;x_1,m,n)$ の $K(x_1)$ 上の既約因子に対応する．
3. ノルムを体 K で因数分解する．
4. ノルムの各因子において，逆の変数変換 $x \to x - x_1$ を施した上，$g_2(x;x_1,m,n)$ との gcd を取ることにより $K(x_1)$ 上の既約因子を得る．

実際のノルムの計算には終結式を用いる．すなわち，

$$\begin{aligned}&\mathrm{N}_{K(x_1)/K}(g_2(x+x_1;x_1,m,n))\\&\quad=\mathrm{res}_{x_1}(g_2(x+x_1;x_1,m,n),g_1(x_1;m,n))\text{（105 秒）}\\&\quad=2^{14}(m+n)^4\cdot N_2\cdot N_3\cdot N_4\ \text{（15.25 秒）}\end{aligned}\tag{4.31}$$

ここに，

$$\begin{aligned}N_2=(m+n)^4x^4-2(m^2+1)(m+n)^2\{(nm-1)^2+n^2+1\}x^2+\\ n^2(m^2+1)^2(nm^2-2m-n)^2,\end{aligned}\tag{4.32}$$

$$\begin{aligned}N_3=(m+n)^4x^4-2(n^2+1)(m+n)^2\{(nm-1)^2+m^2+1\}x^2+\\ m^2(n^2+1)^2(n^2m-m-2n)^2,\end{aligned}\tag{4.33}$$

$$\begin{aligned}N_4=(m+n)^2x^4-2(nm-1)^2(m^2+n^2+2)x^2+\\ (nm-1)^4(m-n)^2.\end{aligned}\tag{4.34}$$

$g_2(x;x1,m,n)$ のノルムが無平方，かつ体 K 上で 3 つの既約因子に分解する．よって，

$\Rightarrow g_2(x;x1,m,n)$ は $K(x_1)$ 上で 3 つの既約因子を持ち，

$\Rightarrow g(=g_1)$ は 1 根 x_1 の添加で 1 次因子に分解される．

そこで，分解体を $K_s=K(x_1)$ と書くことができる．

次に，g_2 の 3 つの因子を拡大体 $K(x_1)$ 上の gcd により実際に求める．たとえば，

$$h_2(x;x_1,m,n)=\gcd([N_2]_{x\to x-x_1},g_2(x;x_1,m,n))\tag{4.35}$$

は N_2 に対応する $g_2(x;x_1,m,n)$ の $K(x_1)$ 上の既約因子を与える．N_3,N_4 に対しても同様である．

有理数体の拡大体上の gcd は Risa/Asir にはチューンされた形で標準装備されているが，有理函数体上の gcd は備わっていない．（他のシステムにあるかどうか不知.）しかし，Risa/Asir の G-base を使えばこの計算が結構高速に実現できる．

式（4.35）の gcd は，$\{[N_2]_{x\to x-x_1},g_2(x;x_1,m,n),g_1(x_1;m,n)\}$ の $x\succ x_1$ による lex（純辞書式）G-base により求まる．（lex G-base を求めるには，いろいろな方法がある.）

$$\begin{aligned}h_2(x;x_1,m,n)&=m(n-m)(m+n)((n^2-1)m-2n)x\\&+4(m+n)^2x_1^3-2(m+n)((3n+1)m^2+(3n^2-3n)m+4)x_1^2+\\&\{(n+1)^2m^4+2n(3n^2-2*n+1)*m^3+\\&n(n^3-6n^2-3n+8)m^2+4n^2(n+2)m-4n^2\}x_1+\\&mn\{(n^2-1)m^4-n(n^2+n+2)m^3+\\&n(n+1)^2m^2-n^2(n+3)m+2n^2+2n\}\qquad\text{（58.2 秒）}\end{aligned}\tag{4.36}$$

$$h_3(x;x_1,m,n)=\text{...省略...（59.54 秒）}\tag{4.37}$$

$$h_4(x;x_1,m,n)=\text{...省略...（49.38 秒）}\tag{4.38}$$

これらはすべて x の 1 次式で，結局，

$$g(x;m,n) = \\ C(m,n)\times(x-x_1)h_2(x;x_1,m,n)h_3(x;x_1,m,n)h_4(x;x_1,m,n) \tag{4.39}$$

と g を分解する．ここに，$C(m,n)$ は係数体 $\mathbf{Q}(m,n)$ に属する．

4.2.4 体のタワーの構成

Galois 群の決定

x_1 は分解体 K_s の原始元である．よって，原始元である x_1 を他の根 x_2, x_3, x_4 に写したときにそれら他の根が別のどの根に移るかを調べれば，g の Galois 群は根の置換群として容易に定まる．たとえば，「$h_3(x;x_2,m,n)$ の $h_2(x_2;x_1,m,n)$ による正規形が因子 $h_4(x;x_1,m,n)$ をもつ ⇔ 根の置換 $x_1 \to x_2$ によって $x_3 \to x_4$」

Asir の正規形計算関数は p_true_nf() であるが零判定のみには p_nf() を用いるのが速い．上記は，

$$\text{p\_nf}(h_3(x;x_2,m,n),[h_2(x_2;x_1,m,n), \\ g_1(x1;m,n),h_3(x;x1,m,n)],[x_2,x_1,x],2) \tag{4.40}$$

が 0（零）になることで確かめられる（1.68 秒）．こうして次の表を得，Galois 群が決定できる．

表 4.1: 根の行き先表と Galois 群

根の行き先表

x_1	x_2	x_3	x_4	置換
x_1	x_2	x_3	x_4	1
x_2	x_1	x_4	x_3	σ
x_3	x_4	x_1	x_2	τ
x_4	x_3	x_2	x_1	$\tau\sigma$

Galois 群

V_4 （Klein の Vierergruppe）．
$\mathrm{Gal}(g) = \{1, \sigma, \tau, \tau\sigma\}$
ここに，
$\sigma := (12)(34)$, $\tau := (13)(24)$.

判別式

d 次多項式 $f(x)$ の根を $x_1, ..., x_d$ とするとき，判別式は

$$\mathrm{Discr}(f(x)) \stackrel{\mathrm{def}}{=} \mathrm{lc}_x(f)^{2d-2} \times \Pi_{i\neq j}(x_i - x_j)$$

で定義され，実際には終結式により次のように計算される．

$$\mathrm{Discr}(f(x)) = \mathrm{lc}_x(f)^{-1} \times \mathrm{res}_x(f, f').$$

$g(x; m, n)$ の判別式を計算して見よう．

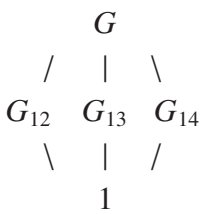
$$\begin{aligned}&\mathrm{Discr}(g(x; m, n))\\ &\quad = 2^{12}m^2n^2(m-n)^2(m+n)^2(n^2+1)^2(m^2+1)^2 \times\\ &\qquad (nm^2 - 2m - n)^2(n^2m - m - 2n)^2(nm-1)^4 \qquad (4.41)\\ &\quad (7.82\text{ 秒})（+\text{ 因数分解 }0.78\text{ 秒}）\end{aligned}$$

判別式が $\mathbf{Q}(m, n)$ 上平方ゆえ Galois 群の一般論から，$\mathrm{Gal}(g)$ は A_4 の部分群．また，g は 1 根添加で 1 次因子に分解するゆえ，群の位数は 4.

これらの観察からも，$\mathrm{Gal}(g)$ は位数 2 の 2 つの巡回群の直積，すなわち，$V_4 = Z_2 \times Z_2$ であることが分かる．

正規部分群

拡大 K_s/K の Galois 群は位数 4 の群 V_4 であることが分かった．よって K_s は K 上の 2 次体 K' を部分体としてもつ．これにより作図可能と分かる．複数あるこの体を決定しよう．まず，$G = \mathrm{Gal}(g)$ には明らかに 3 つの独立な位数 2 の正規部分群 G_{12}, G_{13}, G_{14}（図 4.3）がある．



$$G = \{1, \sigma, \tau, \tau\sigma\},$$

$$G_{12} := \{1, \sigma\},\ G_{13} := \{1, \tau\},\ G_{14} := \{1, \tau\sigma\}$$

図 4.3: 正規部分群

中間体

部分群 G_{12} に対応する中間体 K_{12} を求めよう．まず，G_{12} による K_s の原始元 x_1 の軌道 B_{12} は直ちに

$$B_{12} = \{x_1, x_2\} \tag{4.42}$$

と分かる．そこで，

$$f_{12}(x) := (x - x_1)(x - x_2) \tag{4.43}$$

と置けば，

補題： K に f_{12} の係数をすべて添加した体は K_{12} に一致する．

が成り立つ．特に，f_{12} の係数中 K に含まれないものが K_{12} の原始元となる．明らかに，原始元として

$$p_{12} := -(x_1 + x_2) \tag{4.44}$$

がとれる．p_{12} の K 上の最小多項式を $m_{12}(x)$ とすると，$m_{12}(x)$ は $\{x+(x_1+x_2), g_1(x_1), h_2(x_2; x_1)\}$ の変数順序 $x_2 \succ x_1 \succ x$ の lex G-base により得られる．

$$\begin{aligned} m_{12}(x) = {} & (m+n)^2x^2 + 2(m+n)(nm^2 + n^2m - nm + 1)x + \\ & n\{nm^4 + (2n^2 - 2n)m^3 + (-2n^2 + 2)m^2 + (2n^2 + 2n)m - n\}. \end{aligned} \tag{4.45}$$

(0.53 秒)

これは 2 次式であるから判別式を計算して拡大に必要な添加を得る．

$$\mathrm{Discr}(m_{12}(x)) = 2^2(n^2 + 1)(m + n)^2(nm - 1)^2. \tag{4.46}$$

これから直ちに，

$$K_{12} = K(-(x_1 + x_2)) = K(x_3 + x_4) = K(\sqrt{n^2 + 1}). \tag{4.47}$$

同様にして K_{13}, K_{14} が得られる．

$$K_{13} = K(-(x_1 + x_3)) = K(x_2 + x_4) = K(\sqrt{m^2 + 1}), \tag{4.48}$$

$$K_{14} = K(-(x_1 + x_4)) = K(x_2 + x_3) = K(\sqrt{(n^2 + 1)(m^2 + 1)}). \tag{4.49}$$

このとき，

$$p_{12} = \frac{-(mn^2 + m^2n - mn + 1) \pm (mn - 1)\sqrt{n^2 + 1}}{m + n}. \tag{4.50}$$

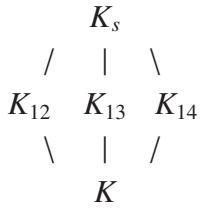


図 4.4: 中間体 K_{12}, K_{13}, K_{14}

以上より，Galois 群とその正規部分群が作る束に対応した体の束（図 4.4）が得られる．

どの 2 つも K 上互いに線形独立な 2 次拡大である．よって，

$$K_s = K(\sqrt{n^2+1}, \sqrt{m^2+1}). \tag{4.51}$$

これによって, 原問題の「美しい解答」に現れた 2 つの独立な 2 次拡大, 式 (4.24,4.25), が機械的な計算によって得られたことになる．

4.2.5　根の表現

これまでの議論から，原方程式が可解であり，しかも 2 次拡大のみで根が構成できる（すなわち作図可能である）ことが分かっている．最終段階として根の根号（平方根号）による表現を求めて見よう．

逐次拡大表現

まず，どのような場合でも通用する逐次拡大表現を求める．すでに K の 2 次拡大のひとつである中間体 K_{12} の原始元 p_{12} の K 上の平方根号による表現は式（4.50）に得られている．拡大 K_s/K_{12} に対応する根の表現を求めよう．$p_s := x_1$ が K_s の原始元である．そこで，p_s の $K(p_{12}) = K_{12}$ 上の最小多項式 $m_s(x; p_{12}, m, n)$ を求める．それには,

$\{x - x_1, p_{12} + x_1 + x_2, g_1(x_1), g_2(x_2; x_1), m_{12}(p_{12})\}$

から x_1, x_2 を消去し, x, p_{12} を残せばよい．ブロックオーダ $\{x_1, x_2\} \succ \{x, p_{12}\}$ で G-base を計算して次を得る．(1.12 秒)

$$m_s(x) = 2(m+n)x^2 + 2(m+n)p_{12}x - m(n-1)(m+n)p_{12} + mn((-n+1)m^2 + nm - n) \tag{4.52}$$

これから，p_s が p_{12}（と m,n）と平方根号を用いて次のように表される．

$$p_s = \frac{-(m+n)p_{12} \pm \sqrt{-(m^2+1)\{(m+n)p_{12} + n(nm^2 - 2m^2 - 2nm + n)\}}}{2(m+n)} \tag{4.53}$$

多根同時添加表現

ここに述べた逐次拡大表現は平方根の分枝の取り方によらず根の正しい表現を与えるが，2 重根号表現となるため人間に取って見やすいとは言えない．分解体 K_s が 2 つの独立な拡大によることが分かっているので，2 重根号無しの根の表現を与えることが可能である．そのような根の表現を求めて見よう．

多少天下り的であるが，式（4.24,4.25）に合わせて，$b := 1 - m + \sqrt{m^2+1}$，$c := 1 - n + \sqrt{n^2+1}$ と置き，$g(x;m,n) = 0$ から m,n を消去して（7.65 秒），$g_{bc}(x;b,c)$ を得る．これは，$\mu = \sqrt{m^2+1}, \nu = \sqrt{n^2+1}$ と置いたとき，$K(b,c) = K(\mu,\nu)$ かつ $m,n \in \mathbf{Q}(b,c)$ とできるように b,c を決めたわけである．この $g_{bc}(x;b,c)$ を $\mathbf{Q}(b,c)$ 上で因数分解することにより $\mathbf{Q}(b,c)$ 上の根の表現が得られる．

$$\begin{aligned} g_{bc}(x;b,c) = &\{4(c-1)(b+c-2)x + b(c-2)(-cb+2)\} \\ &\times\{4(b-1)(b+c-2)x + c(b-2)(-cb+2)\} \\ &\times\{4(c-1)(b-1)(cb-b-c)x + (c-2)(b-2)(-cb+2b+2c-2)\} \\ &\times\{4(cb-b-c)x + bc(-cb+2b+2c-2)\} \quad (4.89\text{ 秒}) \end{aligned} \tag{4.54}$$

正攻法としては $\mathbf{Q}(b,c)/\mathbf{Q}$ の拡大に関するノルムを使い，$g(x;m,n)$ の因数分解を行なう．

最後の因子から得られる根が，最初にあげた根の表示に対応することをつぎに示そう．

4.2.6 人手による根の表示との比較

式（4.23）に合わせて，$a = 1 - l + \sqrt{l^2+1}$ と置くとき，つぎの 2 つの式は同じか否か？

$$x = \frac{bc}{2a} \tag{4.55}$$

$$x = \frac{bc(bc-2b-2c+2)}{4(bc-b-c)} \tag{4.56}$$

このことを調べるために，

$$k = \frac{(bc-2b-2c+2)}{2(bc-b-c)} \tag{4.57}$$

と置き，

$$k = 1/a \tag{4.58}$$

が成り立つかどうかを調べよう．$\{2(bc-b-c)k-(bc-2b-2c+2), lmn-l-m-n, a^2-2a-2l(1-a), b^2-2b-2m(1-b), c^2-2c-2n(1-c)\}$ の変数順序 $b \succ c \succ l \succ m \succ n \succ a \succ k$ の lex G-base を計算すると，それは $(n^2+1)(ka-1)((k-1)a-2k+1)$ を含む．これから，

$k = 1/a$ あるいは $k = (a-1)/(a-2)$

となるが，実の平面幾何の問題として $1 < a < 2$ かつ $0 < x < 1$ なので，後者は不適当．よって，代数的には決定できないが実の図形の問題とした場合には $k = 1/a$ が成立し，数式処理である程度機械的に解いた結果は人手によるものに一致する．

4.2.7　考察

必要な主だった計算をまとめる．

（1）有理函数体の 4 次の拡大体上での因数分解—終結式と有理数体上の因数分解

（2）有理函数体の拡大体上での GCD 計算—G-base 計算

（3）有理函数体の拡大体上で正規形の計算—G-base 計算

これらの計算に Risa/Asir は十分効果的であった．Katsura6 や Symplectic 数値積分公式の導出などに比べると，Risa/Asir にとっては比較的小さな問題であったといえる．

本稿で題材とした「三斜内容三圓術」は，まったく自明というのでもなく，また数式処理システムにとってまったく手が出せないというのでもなく，代数計算への数式処理機能の使い方が示せる例題として手頃であった．規模の大きな問題に対するグレブナ基底の計算には種々のノウハウがあり，本稿では説明し切れなかった．実際，上手くすれば本稿で参考に示した計算時間はさらに短縮できる．こういったノウハウを蓄積し共有して行くことが数式処理の今後に取って大切であろう．

第5章　平面図形の精密描画とその応用

5.1　平面代数曲線の精密描画

数式処理システム Risa/Asir の際立った特長のひとつに，平面代数曲線の忠実な[1)]描画機能がある．

筆者等は，1996 年に陰函数の描画法に関する米国特許 [75] を取得した．これは，画素に埋没する微小成分を例外としてそれ以外は確実に正しく（忠実に）描画する方法で，筆者等は境界法と名付けた．本節の内容は，その境界法では捉えきれなかった，画素の内部に埋没する微少成分をも確実に補足し描画する方法として，1998 年から 2000 年に学会誌などに発表された [116, 61, 118] ものである．これらの方法の拡張として，3 次元の忠実な描画アルゴリズムへの発展も文献 [144] に検討されている．

次節（5.2 節）では，指定した通過点を通る代数曲線を零点にもつ 2 変数多項式（以下，ある平面曲線をその零点のグラフとして持つ 2 変数多項式を陰函数と呼ぶことにする．）を，数式処理を用いた未定係数法および implicitization によって構成する方法を論じる．得られる陰函数のグラフ（平面代数曲線）を描画して示すことにより作品の妥当性を確認するが，陰函数のグラフを「忠実に」描画する手段としては本章に述べる方法に優る方法は従来は存在しなかった．

本アルゴリズムの研究は齋藤[2)]のアイデアに基づくが，筆者の貢献はつぎの 2 点である．

1. グラフが画素境界上にあるか否かを，Sturm 列を利用して判定することを提案し，境界法を確立したこと．

2. 画素内部に埋没する成分の存在判定に用いる 2.2 節の分離写像を用いる方法に対する理論の整備と厳密化を行ったこと．

[1)]「精密な」と言わず敢えて「忠実な」という理由は，数学的に明確な「正しい描画」という概念を定義し，それを基礎にアルゴリズムを構成しているからである．

[2)]齋藤友克のアイデア（隣接する格子点での符号の変化を利用）に基づき，1991 年に Risa/Asir に陰函数の描画機能がインプリメントされた．

なお，本節で例示するハートの形をグラフに持つ陰函数は次章に詳述する方法で作成された作品である．

5.1.1　目的とアプローチ

任意の数学的に定義された 2 変数方程式 $f(x,y) = 0$ の解を画面もしくは紙面の上に描画することを目的とする．特に，$f(x,y)$ が多項式の場合は，y は x の代数函数（一般に多価函数）であり，このような代数函数の零点を描画する問題は，容易な問題と考えられていた．ところで，真の解は幅のない連続な曲線である．しかし，実際には描画する装置に依存する解像度の制限内での小さくとも有限の領域をもつ “点” の集合として，幅のある “曲線” として描かれる．このことを考慮した上で，描かれた図形がどの程度正確かという問に対して数学的に厳密な解答を与えることは長く放置されていた．この問題に関して，Fateman[24] による正しい図形の描画に関する論文がある．また，位相的に正しい図を描画するアルゴリズムについて，谷口らによる文献 [163] が存在する．また，基礎的な概念は筆者等による文献 [116, 75] により与えられている．

従来，2 変数函数の零点を描画するためにはつぎのようにしていた．(i) 零点のいくつかを数値的に求める．(ii) その点を含む分枝をなんらかの手段で追跡する，あるいは補間して表示する．このような従来の数値計算に基づく方法は，特異点以外では数値的近似が可能であるとは言えるものの，明示的にどの程度の近似であるかを示すためには，個々の函数ごとに吟味が必要であった．これに対し，本節で提案する Faithful plot は，描画されたグラフの忠実さという概念が明確であり，その概念を満足するアルゴリズムは実行可能である．

本論文で提案する手続きの特徴として，つぎの 2 点を上げることができる．第一は，計算を有理的計算に限定することにより正しい結果を与えるアルゴリズムを構成している．従来の手法で提案されているすべての計算が無限大の精度をもって正確に計算できるという仮定の上のアルゴリズムに比べ，結果の正しさの保証に明らかに有利である．第二には，表示された図形の正しさの検証が被表示函数によらず不必要であることである．

計算を有理的計算に限定したことが意味を持つためには，多項式の係数は有理数を含む最小の代数的閉体の元に限るという制限を持つことになる．さらに，計算量を考慮するならば，代数的数を避け有理数係数の多項式のみを扱うことが現実的には妥当である．従来から数値計算で取り扱われてきた事例は，あたかも代数的や超越数が計算できるかのように見えるが，実際の描画対象はこの制約の範囲内の多項式である[3)]．したがって，本アルゴリズムの描画対象は有理係数 2 変数多項式であるが従来の描画

[3)] 超越数や代数的数を含んだり，超越函数や代数函数を扱っているかのように見えるが，実態はそ

対象との違いは単に見掛けのものに過ぎない.

5.1.2 新しい描画概念の定義

アルゴリズムを構成するにあたって必要な概念を導入する．また，利用する数学的な用語ならびに定義，定理を示す．ただし，グレブナー基底や分離写像に基づく実根の計算については，それぞれ，2.1 節や 2.2 節を参照されたい.

Cell の概念

本アルゴリズムにおける描画の最小単位として Cell と呼ばれる集合を定義する.

定義 9 (Cell の定義)
描画空間 D を Cell と呼ばれる閉集合 C_k に完全分割する.

1. $D = \bigcup_{k=1}^{m} C_k, \quad C_k^i \bigcap C_j^i = \phi, \quad k \neq j,$
2. C_k は連結,
3. 各 C_k の Jordan 測度はゼロではない.

とする．ここで C_k^i は，C_k の内点の全体とする.

この Cell の例としては，描画空間を埋めつくす矩形タイルがある．一般に Cell の境界は区分的に代数函数により定義できるものであれば，本論文で提案するアルゴリズムの骨子が適用できる．しかし，実用的には境界は直線であるほうが計算上有利である．よって

Cell としては境界を含む矩形とする．我々は，Cell の大きさを「描画の精度」あるいは「解像度」と呼ぶ.

アルゴリズムの計算量をさらに軽減するため，分割の境界は単に代数的数とはせず有理数とする．その理由は，計算量的に不利な代数的数の計算を避けるためである．以上の条件により，一般性を失うことなく描画空間の領域 D をつぎのような矩形領域にとることができる.

$$D = \{(x, y) | a_1 \leq x \leq a_{m+1},\ b_1 \leq y \leq b_{n+1}\}.$$

れらの近似値や（区分的）有理式近似を扱っているにすぎない．数値計算ライブラリで使用している近似式（有理式）が与えられるなら，超越函数（と仮想的に見做している函数）にさえも，本稿の陰函数描画アルゴリズムを適用することができる.

そして，この矩形領域を分割する各 Cell $C_{i,j}$ は

$$C_{i,j} = \{(x,y) \in \mathbb{R}^2 \mid a_i \le x \le a_{i+1},\ b_j \le y \le b_{j+1}\},$$
$$i = 1, ..., m,\ j = 1, ..., n,$$

とする．

忠実な描画（Faithful plot）の定義

本論文の基本的立場は，『“函数を描画する” とは “ $C_{i,j}$ と函数の零点集合が共通部分を持つ” ことを判別することである』とするものである．

定義 10 (Faithful plot)
描画が忠実であるとは，描画される $C_{i,j}$ は解曲線と共通部分を必ず持ち，描画されない $C_{i,j}$ は解曲線と共通部分を持たないことである．

ここで “描画する”（plot）とは，表示装置の該当する画素を “塗る” ことである．（紙に印刷する場合も印刷インクを解像度に応じて “塗る” こととする．）この描画基準による描画を実行するには，代数方程式の解が与えられた領域内に存在するかどうかの構成的判定が必要である．一般の（超越的な元，自然対数の底 e や三角函数などを含む）2 変数方程式に対するこの要請を満足する理論およびアルゴリズムは知られていない．1 変数代数方程式の場合は Sturm の定理により判定できる．また，多変数代数方程式の場合の数学的な理論は Roy らの研究 [57] がある．一方，2 変数の代数方程式に特化した場合は文献 [60, 115] の結果がある．本論文では多数の領域での解の存在判定に適した文献 [60, 115] の方法を発展させ，不備を補った文献 [38] の方法を用いる．

5.1.3　数学的準備

本論文で扱う体 K は，特に断らない限り有理数体 $\mathbb{Q}$ とする．また K 上の多項式環（K を係数体とするすべての多項式の作る環）を $K[x_1, \cdots, x_n]$ とする．

無平方多項式

定義 11 (無平方多項式)
多項式が重根を持たないときその多項式を無平方多項式という．
多項式を割り切る最大次数の無平方多項式をその多項式の無平方部分と呼ぶ．

Sturm の定理

Sturm の定理 [36] によれば，1 変数多項式の実根の存在が厳密に判定できる．

定義 12 (多項式の Sturm 列)
$f(x)$ を実係数無平方多項式とする．従って，$f(x)$ とその導函数 $f'(x)$ は自明な共通因子（係数体の元，特に 1） 以外を持たない．

つぎの漸化式で定まる多項式の列 $\{f_i(x)\}_{i=0,\cdots,r}$ を多項式の Sturm 列と呼ぶ．

$$\begin{aligned} f_0(x) &= f(x), \\ f_1(x) &= f'(x), \\ f_{i-2}(x) &= q_{i-1}(x)f_{i-1}(x) - f_i(x) \\ &\quad \deg(f_i(x)) < \deg(f_{i-1}(x)), \\ &\quad q_i \in \mathbb{R}[x], \\ &\quad \text{for } i = 2, 3, \cdots r-1, \\ f_r(x) &= 0. \end{aligned}$$

ここで，$f'(x)$ は $f(x)$ の導函数である．

定理 13 (Sturm の定理)
$a < b$ で a が $f(x)$ の根でないならば，半開区間 $(a, b]$ にある $f(x)$ の実根の個数は $V(a) - V(b)$ に等しい．

ただし，$V(a)$ は多項式の Sturm 列の変数に数値 a を代入した Sturm 列 $f_0(a)$, $f_1(a)$, $f_2(a), \cdots, f_r(a)$ の符号変化の回数とする．

多項式イデアルとその零点集合

定義 14 (多項式イデアル)
$X = x_1, ..., x_n$ を変数の有限集合とする．K を係数体とする多項式環 $R = K[X]$ の部分集合 $\mathcal{I} \in R$ がつぎの性質を持つとき, $\mathcal{I}$ を $K[X]$ のイデアル（多項式イデアル）と呼ぶ．

- 任意の $\mathcal{I}$の元 f, g に対し $f + g \in \mathcal{I}$.
- $\mathcal{I}$の任意の元 f と多項式環 R の任意の元 g に対し $fg \in \mathcal{I}$.

多項式イデアル $\mathcal{I}$に対し，ある有限個の $\mathcal{I}$の元 $f_1, ..., f_m$ が存在し，すべてのイデアルの元は，$a_1 f_1 + \cdots + a_m f_m$ と表すことができる．ここで a_i は多項式環 R の元である．この $f_1, ..., f_m$ をイデアル $\mathcal{I}$の生成元，あるいは基底と呼び，$\mathcal{I} = \text{ideal}(f_1, ..., f_m)$ と表す．

定義 15 (イデアルの零点集合)
$\mathcal{I}$を多項式イデアルとする．イデアル $\mathcal{I}$を構成するすべての多項式の共通零点をイデアルの零点と呼ぶ．イデアルの全ての零点から成る集合を Zero($\mathcal{I}$) と表す．

定義 16 (零次元イデアル)
イデアルの零点の個数が有限個であるときこのイデアルを零次元イデアルと呼ぶ．

命題 17
ideal($f_1, \cdots, f_m$) の零点集合は $f_1, \cdots, f_m$ の共通零点である．また，その逆も成り立つ．

このことから，連立方程式の根は方程式を構成する多項式の生成するイデアルの零点であり，同じイデアルを生成する生成元の共通零点に一致することが分かる．したがって，方程式を解くとは，同じイデアルを生成し，ある意味で簡単な生成元の集合を求めることと解釈できる．グレブナー基底はそのような性質を持った生成元集合であり，連立方程式を解くために効果的に利用できる．

最小多項式

イデアルの次元が零次元であると仮定する．つぎのような最小多項式と呼ばれる多項式が構成できる．

定義 18 (最小多項式)
$\mathcal{I} \subset R = K[x_1, \cdots, x_n]$ は零次元イデアルであるとする．$f \in R$ の$\mathcal{I}$に関する最小多項式 $\psi \in K[t]$ とは多項式であって，つぎの性質を持つ．

1. ψ はモニック[4)]．
2. $\psi(f) \in \mathcal{I}$．
3. モニックで $\psi'(f) \in I$ を満たす任意の $\psi' \in K[t]$ に対して，$\deg(\psi) \leq \deg(\psi')$．

ある項順序での Gröbner 基底が求まっているとする．このとき線形計算によって，容易に最小多項式が求められる [92]．特に，$f = x_i$ の場合には，$\psi(x_i) \in \mathcal{I}$ となり，$\psi(x_i)$ 自身がイデアル $\mathcal{I}$の元である．このことから，$f_1, \cdots, f_m$ の共通零点の x_i-座標が満たすべき一変数方程式は x_i の最小多項式として得られる．

[4)]多項式がモニックであるとは，最高次数の係数が 1 であることである．

特殊点

$f(x,y)$ の零点を描画するために手掛かりとする曲線上の点 (x,y) はつぎのものである．

（1） 特異点 $f(x,y)=0$ ， $\partial f(x,y)/\partial x=0$ かつ $\partial f(x,y)/\partial y=0$

（2） x-特異点 $f(x,y)=0$ かつ $\partial f(x,y)/\partial x=0$

（3） y-特異点 $f(x,y)=0$ かつ $\partial f(x,y)/\partial y=0$

本論文においては，特異点，x-特異点，y-特異点を特殊点と呼ぶ．

平面を直線に写す写像による正方形領域の直線上の像

平面上に正方形 $\mathcal{S}_{i,j}$ $i=1,\cdots m$, $j=1,\cdots n$ が以下のように配置されているとする．

$$\begin{aligned}\mathcal{S}_{i,j} &= \{(x,y)|a_i \le x \le a_i+w, \\ &\quad b_j \le y \le b_j+w\} \\ &\quad 0<a_1,a_2,\cdots,a_m \quad (a_i+w<a_{i+1}) \\ &\quad 0<b_1,b_2,\cdots,b_n \quad (b_j+w<b_{j+1})\end{aligned} \tag{5.1}$$

a_i, a_{i+1} の最小距離を E とする．同様に b_j, b_{j+1} の最小距離を M とする．

平面上の点 $(x,y)\in\mathbb{R}^2$ を直線上の点 $w\in\mathbb{R}$ に写す線形写像 $\phi(x,y)$ を考える．この写像によってすべての $\mathcal{S}_{i,j}$ の像が交差しないための条件を求める．

この線形写像は特殊点の存在する Cell を判定するために利用される．

定理 19
上記の仮定の下で，$\mathcal{S}_{i,j}$，$i=1,\cdots,m$, $j=1,\cdots,n$ の線形写像 $\phi=y+\alpha x$ による像は，

$$\frac{w}{E-w}<\alpha<\frac{M-w}{a_m-a_1+w} \tag{5.2}$$

であれば交差しない．

これは 2.2 節の分離係数条件に他ならない．ただしここでは，アルゴリズムを簡潔にするために，矩形領域を正方形領域と単純にしてある点に注意する．

5.1.4　アルゴリズム

アルゴリズムの構成

アルゴリズムの全体はつぎの 2 部分からなる．

Part A　特殊点を含む Cell を決定する．

Part B　境界線上に零点を持つ Cell を決定する．

Part A により描画される Cell

特異点，x-特異点もしくは y-特異点を持つ Cell が検出される．本アルゴリズムでは，x-特異点か y-特異点のどちらかを判定すれば目的は達せられる．なぜならば，解曲線が境界を通過している Cell は **Part B** により検出される．解曲線が Cell 境界を通過せず，しかも特殊点を内部にもつ Cell のみを検出できればよい．この場合は，Cell の内部に孤立特異点が存在するか，閉じた代数曲線が存在する．前者は明らかに，x-特異点であると同時に y-特異点である．後者の場合は，代数曲線は Jourdan 閉路であるから，少なくとも 2 点 x-特異点と y-特異点が存在する．

境界上に零点を持たず，内部に閉路あるいは孤立特異点を含む Cell は，このステップを経た段階ですべて検出され描画される．つぎの段階において検出される Cell も検出されるが，このステップを欠くことはできない．

Part B により描画される Cell

解曲線が Cell の境界に接触もしくは境界を横断している Cell がすべて検出され描画される．以上の処理により，すべての描画されなければならない Cell は，過不足なく捕捉される．よってこのアルゴリズムを確定すればよい．

特殊点を含む Cell の決定—Part A のアルゴリズム

特殊点は，特異点，x-特異点，y-特異点いずれかである．特異点は y-特異点でもある．前項の考察より y-特異点を検出すれば十分である．よって連立方程式，

$$f(x,y) = 0, \tag{5.3}$$

$$\frac{\partial f}{\partial y} = 0 \tag{5.4}$$

を解けばよい．

$f(x,y)$ が既約で，かつ $\partial f/\partial y$ が恒等的に 0 でなければ，この連立代数方程式のなすイデアルは 0 次元であることに注意する．

多項式 $f(x,y)$ の $\mathbb{Q}$ 上の既約分解は容易に実行できる．もとの零点は各既約成分の零点の和集合である．よって描画上は多項式 $f(x,y)$ は $\mathbb{Q}$ 上既約であるとして一般性

を失わない．さらに，$\partial f/\partial y$ が消えることは $f(x,y)$ が実は一変数多項式 $f(x)$ である特別な場合であり，これも容易に検出可能である．よって，描画アルゴリズムの対象 $f(x,y)$ は既約でかつ，y の次数は 1 以上と仮定する．

この条件の下で，連立方程式 {（5.3）,（5.4）} の零次元の解が存在する Cell の決定には 2.2 節の零次元方程式の実解を求める方法が好都合に使用できる．

[手続き]

Step 1 計算が容易な項順序で ideal$(f(x,y), \partial f/\partial y)$ の Gröbner 基底 Γ を求める．（通常は全次数逆辞書式順序を用いる．）

Step 2 Γ を利用して，x の最小多項式，y の最小多項式を求める．

Step 3 最小多項式の無平方部分を求め，それらをそれぞれ $M_x(x), M_y(y)$ とする．

Step 4 $M_x(x)$ と $M_y(y)$ とに適当な限界公式による根の限界範囲内で Sturm 法を適用し，Cell のサイズよりも小さく，かつ式（5.2）を満たす α が存在するような w の幅で根の存在範囲を分離する．

- x に関する根の存在区間が
  $[a_i, a_i + w], i = 0, \cdots, m-1,$
- y に関する根の存在区間が
  $[b_j, b_j + w], j = 0, \cdots, n-1$

とすると，方程式の根の存在範囲の候補は式（5.1）で表される正方形領域となる．

Step 5 式（5.2）を満たす有理数 α を用いて，$u = \alpha x + y$ の ideal$(f, \partial f/\partial y, M_x(x), M_y(y))$ に関する最小多項式 $N(u)$ を計算する．

Step 6 Sturm 法を利用して $N(u)$ の根を 2 分探査により分離する．ここで，分離した根の存在する u の区間と，xy-平面上に Step 4 で求めておいた正方形領域のうち方程式の零点の存在する正方形領域とは，1 対 1 に対応する．

Step 7 $N(u)$ の根を含む区間に対応する xy-平面上の正方形領域を決定する．これにより，描画されるべき Cell が決定される．

境界アルゴリズム—Part B のアルゴリズム

各々の $C_{i,j}$ に対し境界上の解の存在を判定することは，$n + m + 2$ 本の直線

$$(a_0, b_0)\text{–}(a_0, b_n), \cdots, (a_m, b_0)\text{–}(a_m, b_n)$$

と

$$(a_0, b_0)\text{–}(a_m, b_0), \cdots, (a_0, b_n)\text{–}(a_m, b_n)$$

上で $f(x, y)$ の Sturm 列を求めればよい．正確にいうと，$f(a_i, y), i = 0, \cdots, m$，および，$f(x, b_j), j = 0, \cdots, n$ の 各々に対して Sturm 列を計算する．

この Sturm 列に対し定理 13 を適用し符号の判定を各々の Cell の境界をなしている区間

$$\begin{aligned} (a_i, b_{j-1}) &- (a_i, b_j), \\ (a_{i-1}, b_j) &- (a_i, b_j), \\ & 1 \le i \le n,\ 1 \le j \le m \end{aligned}$$

で実行する．このアルゴリズムにより各 Cell $C_{i,j}$ の境界上に解が存在するか否かが厳密に判定できる．

5.1.5　実行例

アルゴリズムの実装は C 言語により富士通研究所の開発した数式処理システム Risa/Asir [46] の外部関数として構成されている．使用した計算機は，CPU が intel Pentum III 500Mhz OS が FreeBSD 3.2 である．

孤立特異点と埋没点の実例

実際の事例として孤立特異点を持つ場合と孤立特異点ではない Cell の中に埋没する成分を持つ場合の描画例を示す．2 変数多項式 Heart(x, y) をつぎのものとする．

$$\begin{aligned} \mathrm{Heart}(x, y) = & \frac{93392896}{15625} x^6 \\ & + (\frac{94359552}{625} y^2 + \frac{91521024}{625} y - \frac{249088}{125}) x^4 \\ & + (\frac{1032192}{25} y^4 - 36864 y^3 - \frac{7732224}{25} y^2 \\ & \qquad - 207360 y + \frac{770048}{25}) x^2 \\ & + 65536 y^6 + 49152 y^5 - 135168 y^4 - 72704 y^3 \\ & + 101376 y^2 + 27648 y - 27648 \end{aligned}$$

この多項式は 4 個の孤立特異点を次の位置に持つ.

$$(\pm\sqrt{17150/123201}, 386/351),$$
$$(\pm\sqrt{8575/2888}, 41/76).$$

描画は, Heart(x, y) および Heart$(x, y) - 1/50$ に対して行う. 表示領域は $-2 \le x \le 2$, $-2 \le y \le 2$ の正方形領域であり, 2400×2400 の Cell に分割した場合の描画に要した時間は, Heart(x, y) は 3.0930 秒, Heart(x, y) – 1/50 は 3.5614 秒である. 図 5.1 に Heart(x, y) の零点を示す. 1 個の Cell に埋没する成分[5)]が 4 個の小円で囲んで示されている. これらは孤立特異点であり, 本アルゴリズムにより正しく捕捉されていることが確認できる.

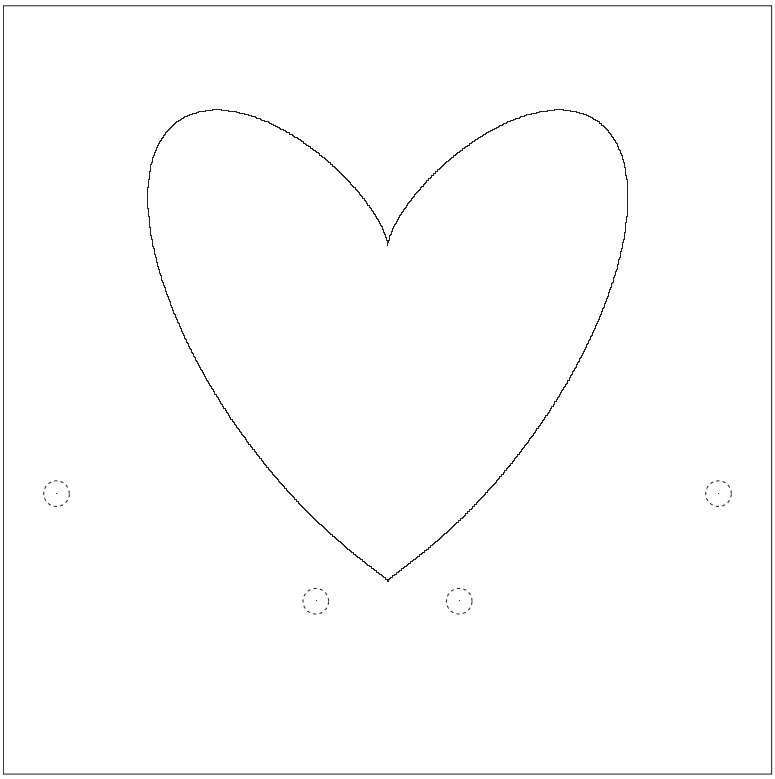


図 5.1: ハート函数

[5)] これらの成分は, 本来は 1 個の Cell に対応する 1 点 (1 画素) として正しく描画されているが, 微少さのゆえに印刷の都合上表示されない場合もある. アルゴリズムにより正しく検出されていることを示すために, 小円で囲んで示した. 小円はグラフの一部ではないことを注意しておく.

図 5.2 は，Heart$(x, y) - 1/50$ の零点のグラフである．Heart(x, y) に存在した孤立特異点（零次元成分）は一次元成分に変わっている．中央下部の一次元成分は Cell の境界を跨いだため楕円状の形状が判別でき表示されている．しかし両脇の一次元成分は Cell の中に完全に埋没してしまうため，1 点（1 画素）だけが表示される．孤立点の場合と同様に，その場所が小円で囲まれて示されている．

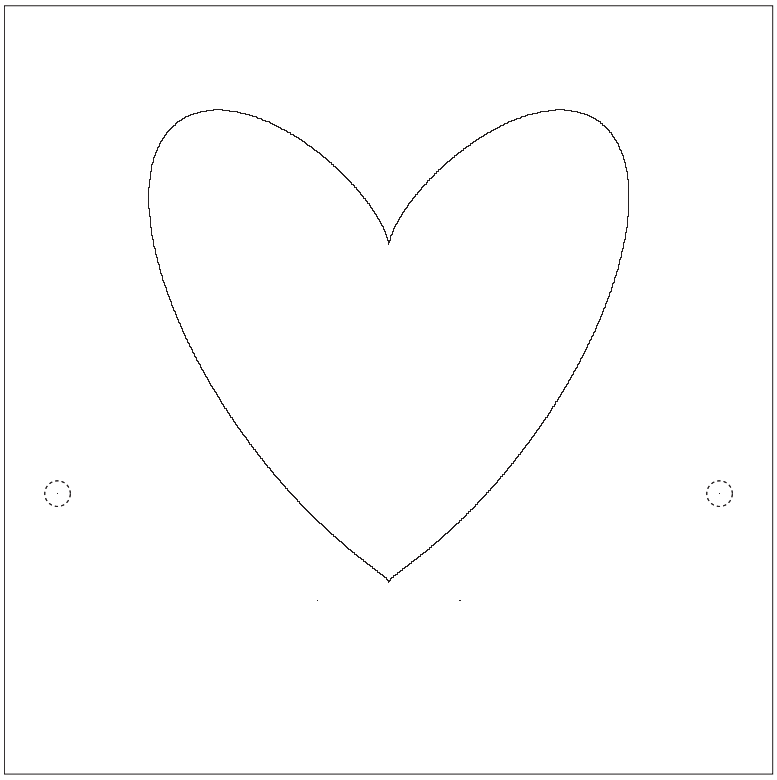


図 5.2: ハート函数 $-1/50$

この 1 点のように見える成分のある $1.722 < x < 1.724,\ -0.541 < y < -0.539$ の領域を拡大表示することで図 5.3 のように楕円状の成分の存在が分かる.

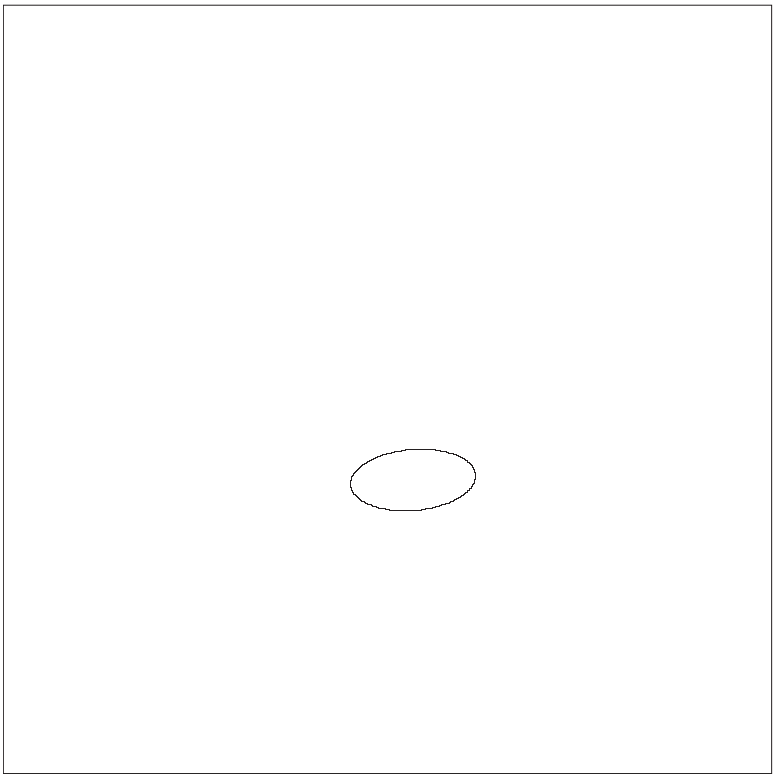


図 5.3: 図 5.2 の小円で囲まれた領域の拡大

従来の事例

文献 [163] と同様の例を本アルゴリズムによって描画する実験を行った．その結果, 文献 [147] に含まれている例題 90 個はすべて 2 秒以下で描画できた．特に, 文献 [163] で描画できなかったとされる文献 [147] の ex. 90 も 1.2270 秒で問題なく描画した（図 5.4）.

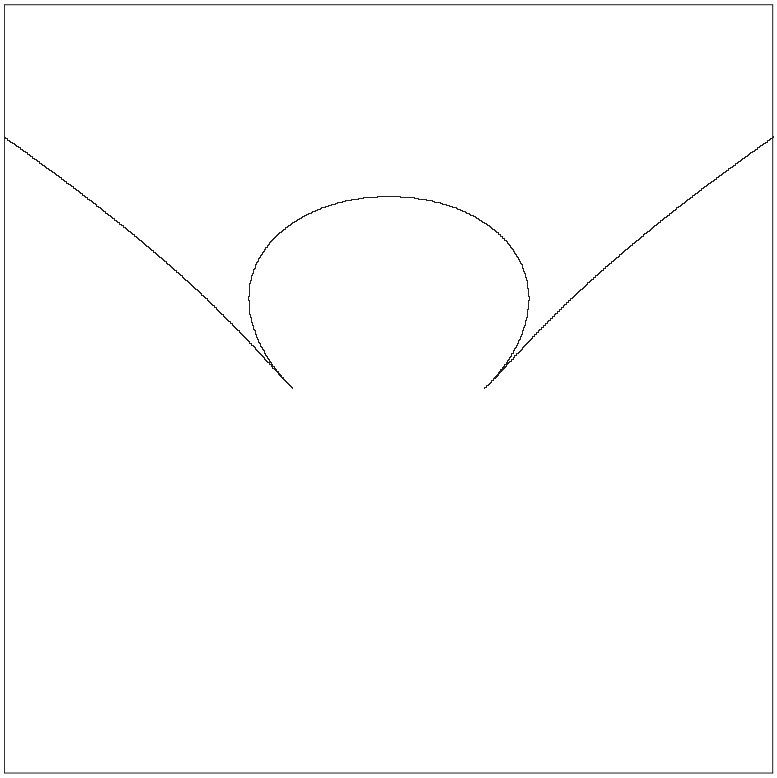


図 5.4: 文献 [147] ex.90 のグラフ

また文献 [163] の摂動を加えた実例も自動的に描画した.

これらの人工的な例に比べて次数や係数がはるかに大きい多項式に対しても本アルゴリズムが適用できることが多くの実例[6)]で確かめられている.

[6)] 4.1 節の式（4.6）や次節（5.2 節）のトランプのスペードやクラブ図柄などは，次数が高く，係数の表現長も数百桁となるような事例であり，しかも多くの特異点を含むため従来の方法では描画できなかった.

5.1.6 結論

従来のアルゴリズムとの比較

従来の描画アルゴリズムは，そのほとんどが数値計算を基礎として構成されてきた．これに対し，代数的算法に基づいた描画アルゴリズムの研究は，特異点の正確な位置や，連結成分や分枝などの正確な幾何学的構造を調べるためになされてきており，実際的な意味でグラフを描画するために利用されることは少なかった．先駆的研究としては，文献 [163] が存在する．この研究では，代数曲線の分枝の追跡がある程度可能となった．しかし代数函数の構造によっては失敗する場合もある（文献 [147] ex.90）．文献 [163] は，基本的な方式として終結式を利用しているため別の問題点を持ち込んでいる．一般に終結式を多変数多項式に適用した場合，連立方程式の真の解以外の解を与える場合がある．消去された変数に関する元の多項式の主係数を終結式の解がゼロとする場合である．

例えば，$f(x,y) = x^2y - 1, \partial f/\partial x = 2xy$ の y を消去した終結式は $2x$ となるが，この解 $x = 0$ は原方程式の解を構成しない．一方，$g(x,y) = (x^2 - x - 1)^2y^2 + (2x^3 - 3x^2 - 6x + 1)y + 5$ と $\partial g/\partial x$ との終結式は $-20(3x^4 - 6x^3 - 38x^2 + 26x - 17)(x^2 - x - 1)^2$ である．この因子 $x^2 - x - 1$ の根 ξ は原方程式の主係数を消すにもかかわらず $(\xi, \xi - 1)$ が原方程式の解となる．

これらの例外的な場合や例外の例外などの処理アルゴリズムは Gröbner 基底を使わない場合には極めて複雑にならざるを得ない．しかし，Gröbner 基底を用いれば簡潔なアルゴリズムが構成できる．

また文献 [163] では，追跡型のアルゴリズムの欠点を除去することが重要になり，この点からもアルゴリズムの構成が複雑にならざるを得ない．

一方，純粋に代数計算のみにより構成された図形描画のアルゴリズムとしては，得られた図形の忠実性を保証する必要がない場合であれば文献 [116, 75] による研究が存在する．

数値計算に基づく描画では追跡型のアルゴリズムがよく研究されている．しかし，一般の追跡型のアルゴリズムでは，初期値の設定によっては辿れない分枝が存在する場合がある．

また，刻み幅を十分小さくしなければ微小な構造を見逃すことになるが，そうすればごく珍しいケースのために全体の計算量の増大を招くことになる．計算量の増大を避けるために，刻み幅を小さく取れないとすれば，計算途中で混入する誤差に対する詳細な検討が問題ごとに個別に必要になる．

また，実際問題での数学的に厳密な議論をする際に現われる多項式は次数が 10 次をはるかに越え，係数の表現長も分母分子を合わせて百桁を越えるような問題がいくらでもでてくる．そのような場合に，通常の固定サイズの浮動小数点数ではそもそも

数が表現しきれないために，計算を誤ってしまうことになる．

区間演算を利用し数値計算の不安定性を救済する方法も存在するが，高次の多項式や係数が大きい場合には，区間が非実用的に広がりすぎてしまう欠点がある．（例えば Heart(x, y) など．）是正するための計算量の増大に見合う描画品質の向上は著者らの経験では見られなかった [116].

提案したアルゴリズム

純粋に代数的算法のみに基づく本論文の提案するアルゴリズムはつぎの点で有用である．

- 有理係数 2 変数の多項式の零点を任意の精度で描画する．
- 特異点，特に孤立特異点を持つ場合でもその絶対的位置が判定できる．
- 微小成分を持つ場合その絶対的位置が判定できる．
- 過去の計算を破棄することなく描画精度を上げること，あるいは拡大表示できる[7)]．
- 検証不要でこれらの結果が全て信頼できる．

本節では，2 変数多項式で定義される平面曲線を，ある正しさの基準を満足するように描画する手法を提案した．このアルゴリズムの基本は，表示しようとする領域を必要な大きさの Cell と呼ばれる部分集合に分割し，個々の Cell に関して解の存在を判別することである．この判別に誤差の概念を混入させないのでアルゴリズムの構成が単純になっている．

また，得られた図形が提案した基準に照らして正しいことを保証している．この保証は，必要精度として表示装置の解像度限界を選択した場合，理論的に正しい図形であることをも保証している．従来，函数の零点を描画した図形は，単に零点の配置の概略を表示するのみであると考えられてきたことを斟酌すると，どのような未知の図形に対しても描かれた図形が正しいという保証があることは優れた特長といえる．本アルゴリズムは，従来の数値計算を基礎とするアルゴリズムと比較した場合，確かに多大な計算を必要とする．しかし，従来のアルゴリズムでは描画できない場合や，特別な配慮がいる場合であっても描画できる．特に孤立特異点がある場合や微細構造を持つ場合に安定に描画することは従来困難であった．

[7)] 荒い分割に対する忠実な描画を事前に求め必要な部分に関して更に細かい分割を行って精度を向上させることも本アルゴリズムによって可能である．

高次元の図形描画

3 次元以上の場合に，本アルゴリズムを適用することは現時点では困難である．n 変数多項式の場合には，その解の次元は 0 から $n-1$ 次元まで存在する可能性がある．このとき解が与えられた Cell に含まれているか否かの判定をすることは純粋な理論問題として未解決である．本論文で提示した手法を拡張することによって，$n-1$ 次元と 0 次元の解を描画するアルゴリズムは構成できる．それ以外の次元の解をも描画することは今後の課題[8)]としたい．

[8)] 3 次元の忠実な描画アルゴリズムについては文献 [144] に検討されている．

5.2　数式処理を用いた陰函数の作成とその描画

我々はいろいろな数式で表される問題を解くに当たって，幾何学的直観を利用することが多い．とくに問題が 2 変数の連立方程式に還元される場合には，連立方程式を構成する個々の方程式の解を平面上の曲線で表すことにより，変数の変域や解の存在範囲，交点の個数（重複度），配置などが容易に見てとれる．それらを同一画面に重ねて表示すれば最終的な答がかなりの正確さで推察できる．こうして答の予想がつけば，予想に基づいた証明や計算の方針を立てることができ，正しい答を見いだすことは予想がつかない場合に比べてはるかに容易となる．ここでの方程式とは陰函数を表す 2 変数の数式であり，方程式を平面上の曲線で表すということは陰函数をグラフに描くということである．

本節では，数式処理に陰函数の描画機能が組み合わされると，変数間の関係や方程式の解の存在およびその性質（パラメータの摂動に対する安定性など）が，直観的にかつ的確に理解できるという数理科学上のメリットばかりでなく，思いもよらない美しく奇妙で，示唆に富んだ図形を手に入れることができることを報告したい．後者の効果は数理科学の初等的教育において，造形の楽しみを与えつつ函数や方程式と図形の関係を理解する上でも役立つものと期待できる．

5.2.1　問題設定

つぎのような問題を考える．

問題 4
ある閉曲線があったとき，この閉曲線を近似する陰函数[9)]表示を求めよ．

この問題を設定した動機は, 当時国際研で開発されていた数式処理システム Risa/Asir に陰函数の描画ライブラリがあり，その機能を用いて図形 ` 矢印 ` の表示ができないかというものであった．

方法としては，全く違った二通りのものが考えられた．一つは，矢印を bitmap 化し，更に描画ライブラリに bitmap イメージも画面出力できるような機能を追加することにより表示する方法．もう一つは, 矢印を陰函数で書くことにより表示する方法. この方法ではライブラリに手を加えることなく実行できる．現実的には初めの方式の方が簡便と思われるが，敢えて後者の方法を取ることにした．

ここで扱うのは代数曲線なので，曲線内に部分的に線分となる所があれば，曲線は直線を含むことになる．つまり，そのような曲線は閉曲線ではない．よって求める曲

[9)]本論文では，所与の平面代数曲線を零点のグラフに持つ 2 変数多項式のことを，そのグラフの「陰函数」と呼ぶことにする．

線は有限個の点を除き曲線上の殆んどすべての点で曲率が 0 にならない．このことから，陰函数がイメージした図形より多少丸みを帯びたものになることは原理的に避けられない．

次の図 5.5 に示す 鏃形の凹四辺形 を例として議論を進める．

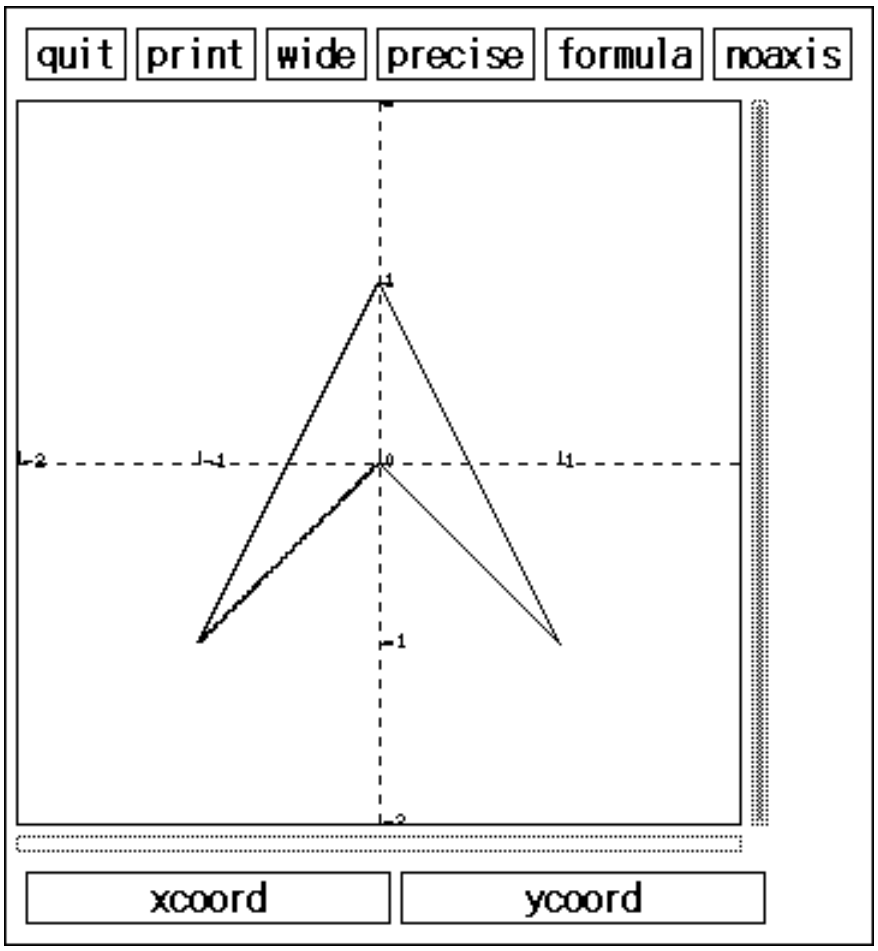


図 5.5: 鏃の形の凹四辺形

この四辺形の 4 つの頂点 A(0,1), B(−1,−1), C(0,0), D(1,−1) を特徴点と考え，これらの点を通る 図 5.5 に近いグラフをもつ陰函数を求めたい．

描画の条件はあまり制限は付けずつぎのものとする．

条件１. 点 A,B,C,D を通る

条件２. Jordan 閉曲線であること．

5.2.2　失敗その１

求める陰函数を，$F(x,y) = a_{n,0}x^n + a_{n-1,1}x^{n-1}y + \cdots + a_{0,0}$ とおいて，四点 A,B,C,D を通るように未定係数法で求める．この場合はとりあえず $n = 2$，すなわち $F(x,y) = a{+}bx{+}cxy{+}dx^2{+}exy{+}fy^2$ とおき, 方程式 $F(0,1) = F(-1,-1) = F(0,0) = F(1,-1) = 0$ を解いて未定係数 a, b, c, d, e, f を定める．解は以下のようになる．

$$F(x,y) = \begin{cases} \alpha x - y - 2x^2 + \alpha xy + y^2 & (f \neq 0) \\ x + xy & (f = 0) \end{cases}$$

ただし α は任意の実数．これらの零点のグラフは，図 5.6 及び 図 5.7 となる．

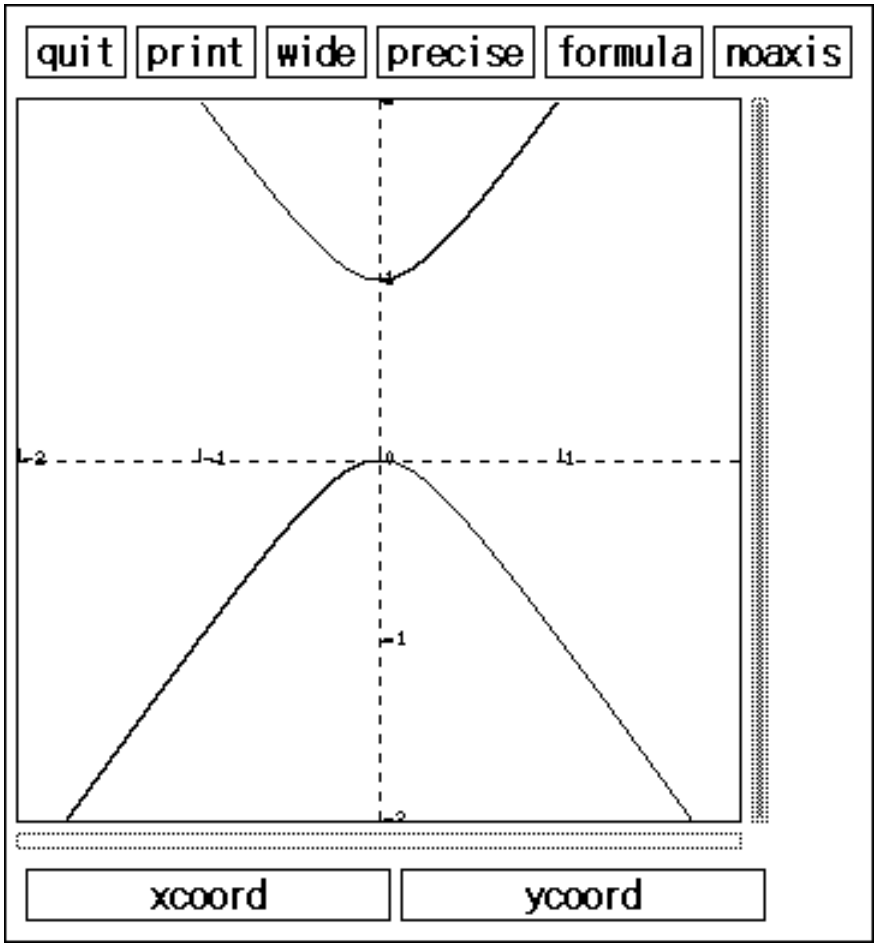


図 5.6: 双曲線 : $-y - 2x^2 + y^2 = 0$ ($\alpha = 0$)

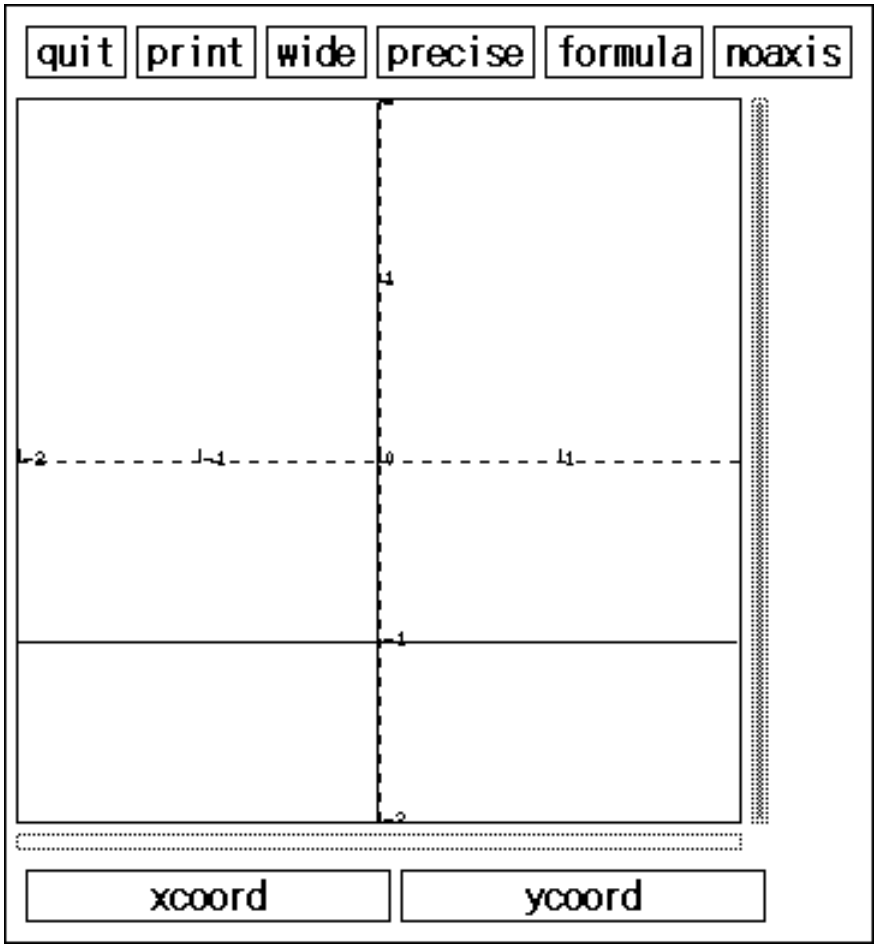


図 5.7: 2 直線 : $x + xy = 0$

この様に，得られた図形は双曲線または 2 直線となり，閉曲線にはならない．この方法で，n を増やしていってもパラメータが増えていくだけで思うように求めることができない．

陰函数そのものの未定係数法では，良い結果がえられないため別の方法を考える必要がある．

5.2.3 失敗その 2

曲線上の点 (x, y) は，A,B,C,D,A を この順序で通ってなくてはいけない事から曲線を x, y の各座標の媒介変数 t に関するパラメータ表示により表現することを試みる．

そこで，x, y を次の多項式とおいてみる．なお，$x(t), y(t)$ の次数を 5 としたのは，パラメータによる変化の自由度を増すためである．

$$\begin{cases} x(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5, \\ y(t) &= b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5. \end{cases}$$

この係数 $a_0, ..., a_5, b_0, ..., b_5$ を次の表に従って決定する．

表 5.1: 矢印のデータ表 1

t	−2	−1	0	1	2
x	0	−1	0	1	0

t	−2	−1	0	1	2
y	1	−1	0	−1	1

$x(t), y(t)$ を求め，$a_5 = -1/10, b_5 = 0$ を代入すると，次のようになる．

$$\begin{cases} x(t) &= -\frac{1}{10} t^5 + \frac{1}{6} t^3 + \frac{14}{15} t, \\ y(t) &= \frac{5}{12} t^4 - \frac{17}{12} t^2. \end{cases}$$

こうして求められた $(x(t), y(t))$ を $-2 \leq t \leq 2$ で動かすと，図 5.8 が得られ，求めたい図形に近くなった．

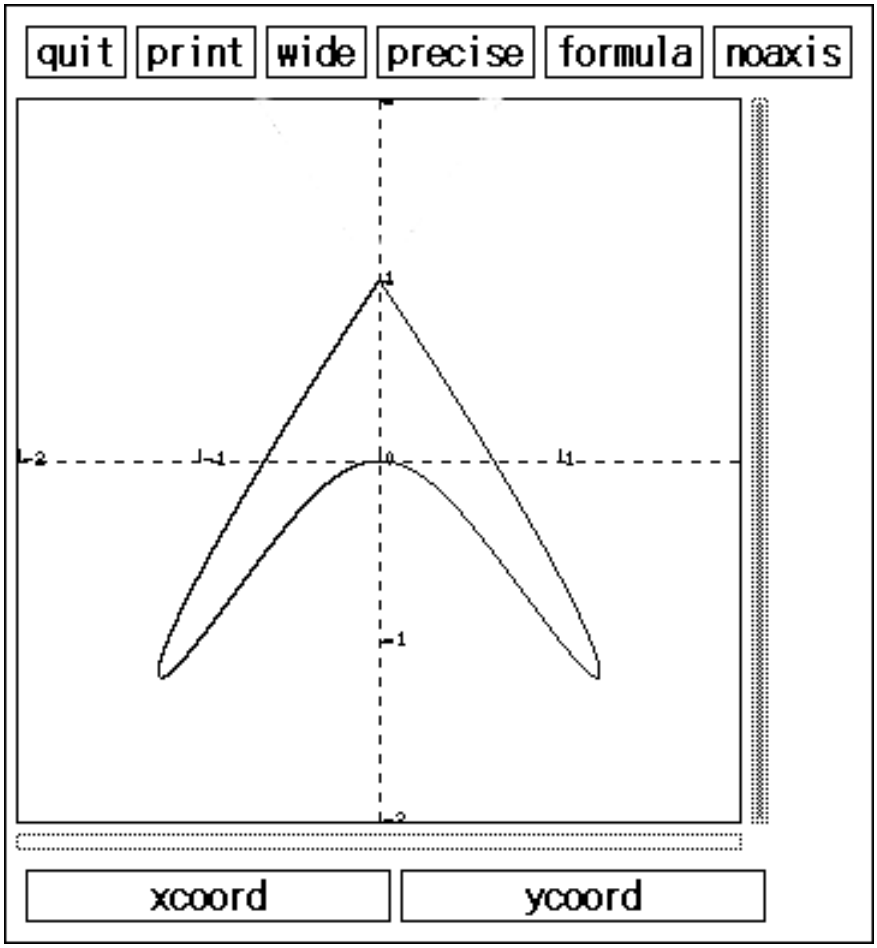


図 5.8: 媒介変数表示で作成された鏃

一方，我々の求めたいものは陰函数であるから，$x - x(t) = 0$, $y - y(t) = 0$ から，終結式あるいはグレブナー基底 (消去法) を用いて t を消去し，次の式

$$\begin{aligned} F(x,y) = \quad & -5859375x^4 + (3487500y^2 - 8080000y + 2357475)x^2 \\ & +46656y^5 - 613440y^4 + 2536528y^3 - 3419360y^2 + 1449616y \end{aligned}$$

を得る．これを表示させると図 5.9 となり，この図形は閉曲線ではない事がわかる．

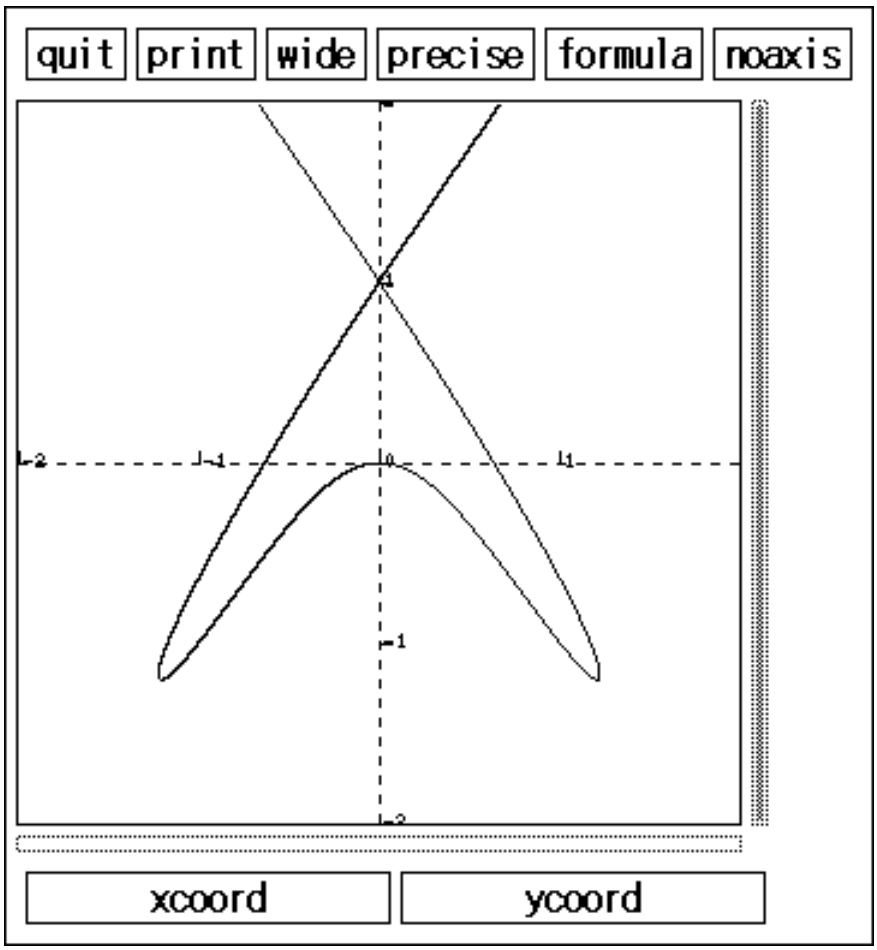


図 5.9: 陰函数で作成された鏃（閉じていない）

これは，媒介変数 t の $t < -2$ 及び $t > 2$ が影響している[10]わけであるが，これを回避する方法を見つける必要がある．

5.2.4　閉じた図形の構成

パラメータを実数全体を走るように構成した媒介変数表示により同様の事を試みる．求めたいものは閉曲線であるから，これを行なうには 実数全体 $\mathbb{R}$（無限遠を含む）で有界な函数を用いる必要がある．

[10]媒介変数の虚数領域が関与しているために，媒介変数表示のグラフと陰函数のグラフとが異なる場合も有り得ることに注意．たとえば，$x(t) = (t^8 - 4t^6 - 10t^4 - 4t^2 + 1)/(t^8 + 4t^6 + 6t^4 + 4t^2 + 1)$, $y(t) = (2t)/(t^2 + 1)$, から t を消去すれば放物線 $f(x, y) = x + 2y^2 - 1 = 0$ を得るが，t を実数全体に取っても，放物線の $x < -1$ の部分は存在しえない．

多項式は有界ではないので失敗した．では，有理式ではどうか．例えば，図 5.10 に示す $y = \frac{1}{x^2+1}$ は実数全体で有界である．

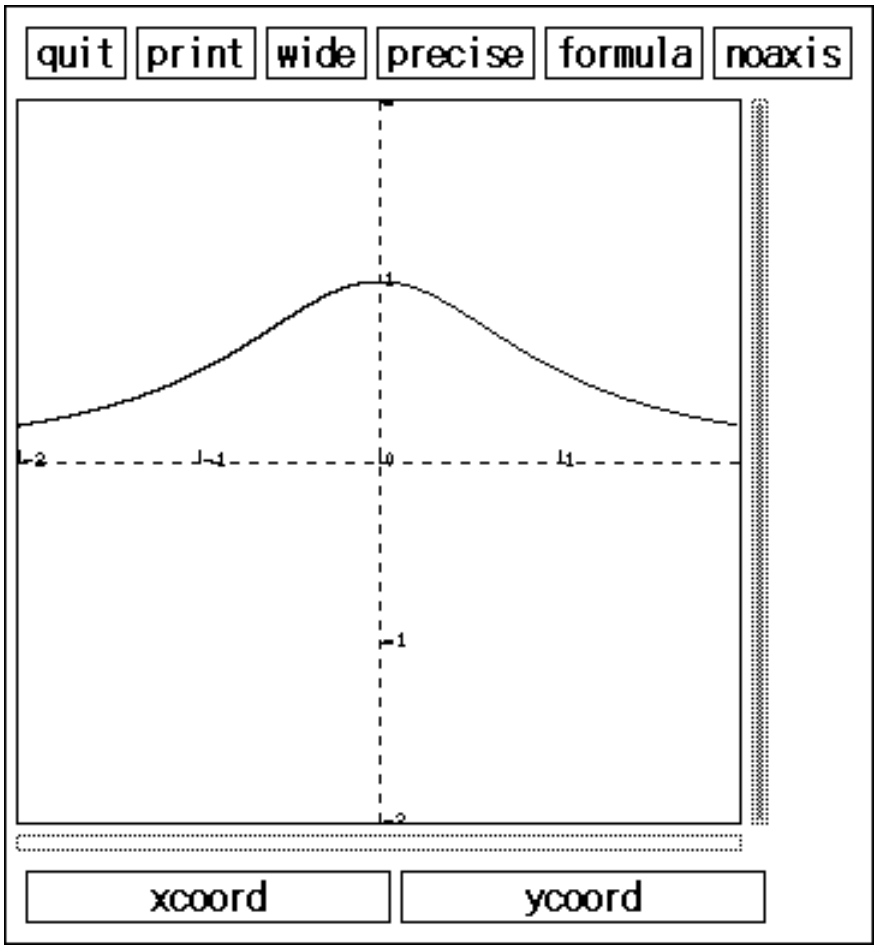


図 5.10: $\mathbb{R}$ で有界な有理式 $y = \frac{1}{x^2+1}$

この様に，$\mathbb{R}$ で有界なものとして有理函数なら使えそうである．

$\mathbb{R}$ で有界となるためには deg 分子 $\leq$ deg 分母 であり，分母 $= 0$ が実数で解を持たない事が条件となる．そこで，x, y を次の有理式でパラメータ表示させてみる．

$$\begin{cases} x &= \dfrac{a_{2n}t^{2n} + \cdots + a_0}{t^{2n} + 1}, \\ y &= \dfrac{b_{2n}t^{2n} + \cdots + b_0}{t^{2n} + 1}. \end{cases}$$

あとは，通るべき点の個数から n を決定し各点の座標を代入し，係数を決定する．

ただし，点を $\mathbb{R}$ の無限遠に対応づけたい時は，

$$\lim_{n\to\infty} \frac{a_{2n}t^{2n} + \cdots + a_0}{t^{2n}+1} = a_{2n}$$

であるから，(a_{2n}, b_{2n}) に対応させれば良い．

与えられた点は A,B,C,D の四つであるから $n = 2$ で解けば良い．

$$\begin{cases} x &= \dfrac{a_4t^4 + a_3t^3 + a_2t^2 + at + a_0}{t^4+1}, \\ y &= \dfrac{b_4t^4 + b_3t^3 + b_2t^2 + bt + b_0}{t^4+1}. \end{cases}$$

表 5.2: 矢印のデータ表 2

t	$-\infty$	-1	0	1
x	0	-1	0	1

t	$-\infty$	-1	0	1
y	1	-1	0	-1

これより x, y を求めれば，x, y はそれぞれパラメータ a, b を持つ以下の様な式になる．

$$\begin{cases} x(t) &= \dfrac{at^3 + (-a+2)t}{t^4+1}, \\ y(t) &= \dfrac{t^4 + bt^3 - 3t^2 - bt}{t^4+1}. \end{cases}$$

これらを，多項式の時と同様に終結式あるいはグレブナー基底で t を消去し，x, y のみの式にして，更に因数分解により本質的な部分を取り出せば次の陰函数が得られる．

$$\begin{aligned} F(x,y) = {} & (b^4+8b^2+25)x^4 + (((-4b^3-16b)a + 4b^3 + 4b)y \\ & +(-b^3-7b)a + 2b^3 + 8b)x^3 + (((6b^2+8)a^2 + (-12b^2-4)a + 8b^2 \\ & -12)y^2 + ((3b^2+7)a^2 + (-6b^2-26)a + 10b^2 + 42)y + (-b^2-3)a^2 \\ & +(2b^2-6)a - b^4 - 6b^2 - 3)x^2 + ((-4ba^3 + 12ba^2 - 16ba + 8b)y^3 \\ & +(-3ba^3 + 6ba^2 - 6ba)y^2 + (2ba^3 - 6ba^2 + (2b^3+16b)a - 2b^3 \\ & -6b)y + ba^3 + (-b^3-3b)a - 2b)x + (a^4 - 4a^3 + 8a^2 - 8a + 4)y^4 \\ & +(a^4 - 2a^3 - 4a^2 + 12a - 12)y^3 + (-a^4 + 4a^3 + (-b^2-7)a^2 \\ & +2b^2a + 12)y^2 + (-a^4 + 2a^3 + (b^2+3)a^2 + (-2b^2-4)a - 4)y \end{aligned}$$

この式の a, b を変化させる事で，点 A,B,C,D を通るいろいろな曲線が得られる．これらを Risa/Asir の陰函数描画システムで描かせてみると，次の様になる．

（1） $a = 0, b = 0$ の場合陰函数の式は

$$F(x, y) = 25x^4 + (-12y^2 + 42y - 3)x^2 + 4y^4 - 12y^3 + 12y^2 - 4y$$

である．これに対する図形は図 5.11 となる．

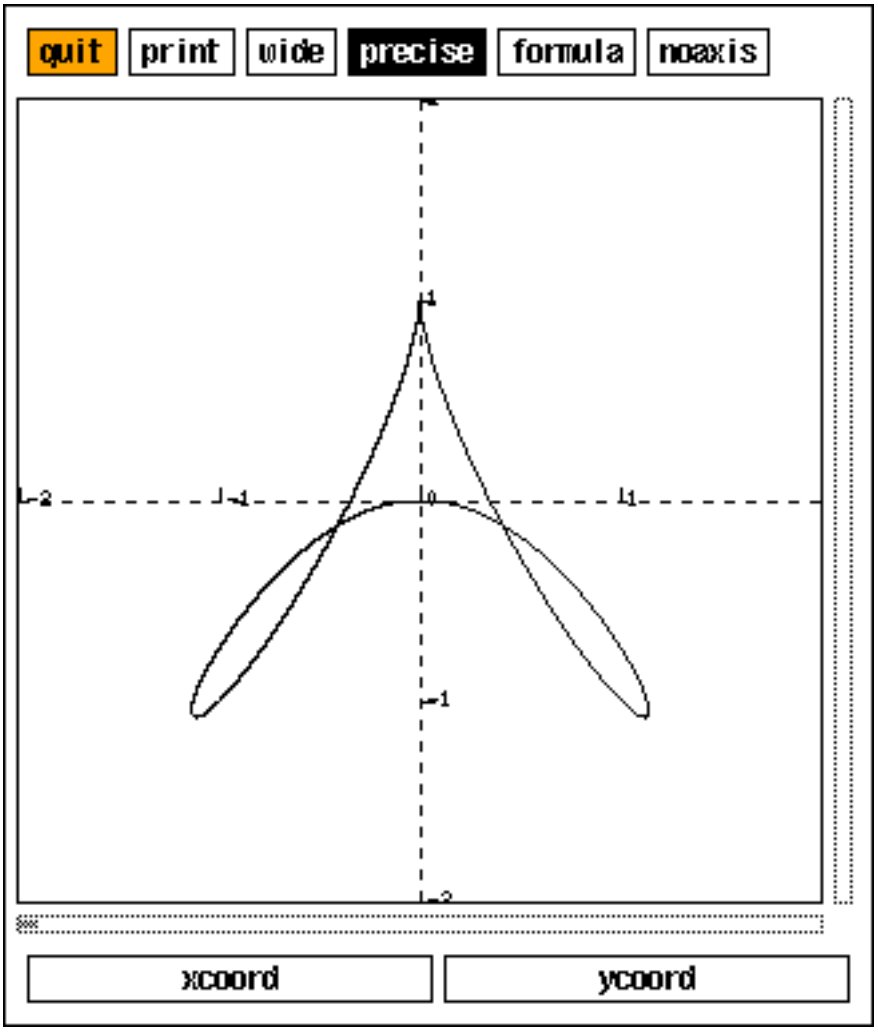


図 5.11: 鏃:$(a, b) = (0, 0)$ の場合

（2） $a = 1, b = 0$ の場合陰函数の式は

$$F(x, y) = 25x^4 + (-8y^2 + 23y - 12)x^2 + y^4 - 5y^3 + 8y^2 - 4y$$

である．これに対する図形は図 5.12 となる．

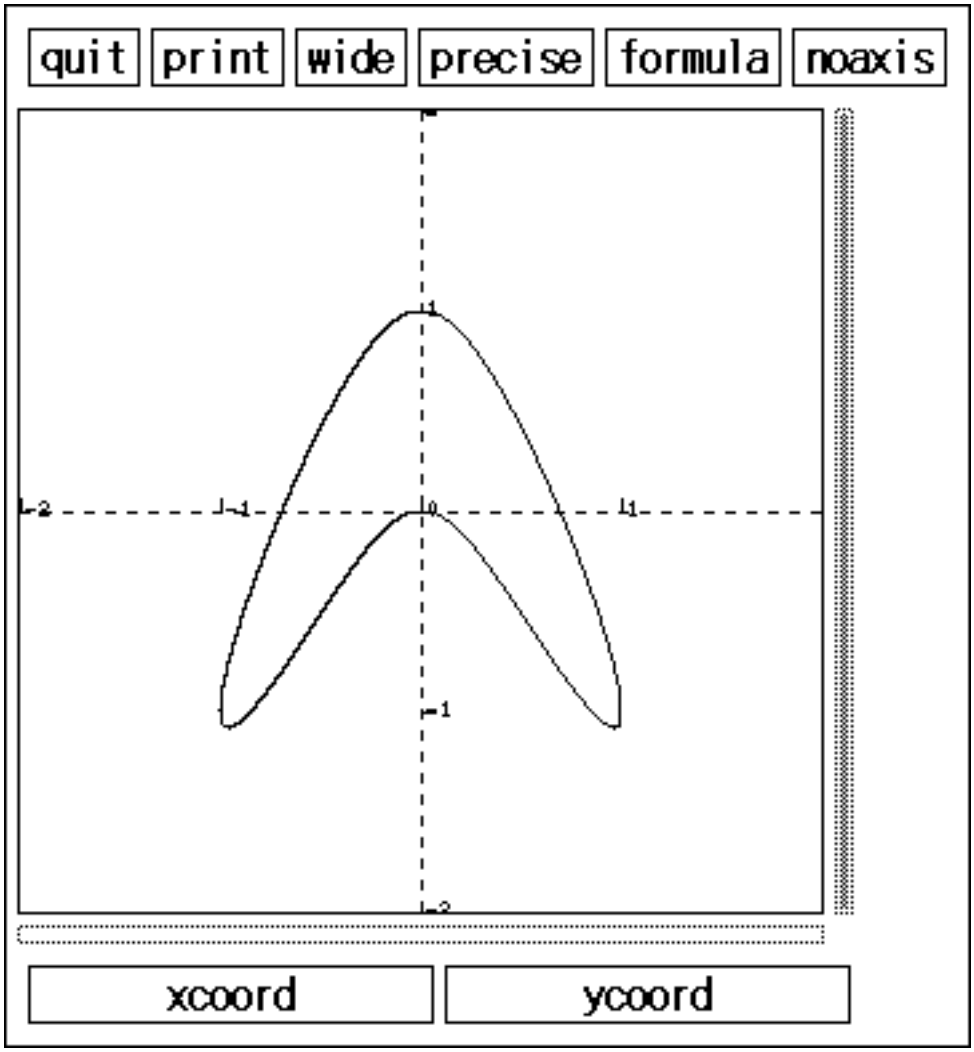


図 5.12: 鏃:$(a, b) = (1, 0)$ の場合

（ちなみに （2） の曲線は，$(0, 2)$ に孤立特異点を持っている事がわかった.）

この図 5.12 が，求めていた鏃の陰函数である．よってこれで目的は達成された．目的を達成するために用いたアルゴリズムのポイントは，有理函数による各座標のパラメータ表示をした事である．後は，これに従って具体例を作っていけば良い.

以下に，いろいろな例をあげる．陰函数内のパラメータをいろいろ変えると，思いもよらぬ図形が現れ，興味深い.

5.2.5 特異点（尖点）を持たせる場合

今までの例は，有界な閉曲線で特異点を持たせないものであった（結果として特異点が得られることはあるが，意図としてそうしたものではない）．しかし実際には，積極的に特異点（尖点，交点）を持たせた方が，より面白い図形が得られる場合が多い．

例えば，トランプの札の種類（ダイア，ハート，スペード，クラブ）を表す形を，陰函数で表すそうとした時，どうしても尖った所（尖点）を作るアルゴリズムが必要になってくる．

大筋としての方針は，前節の有理函数によるパラメータ表示で良いと思われる．では，$x = x(t), y = y(t)$ のパラメータにどのような条件をつければ尖点が，現れてくるだろうか．その問題は，次の補題によって解決される．

補題 20 (補題)
曲線が $(x(t_0), y(t_0))$ で尖点を持ったならば $x = x(t), y = y(t)$ は，$t = t_0$ において $x'(t_0) = y'(t_0) = 0$ を満たす．

以下にこの補題に基づいて尖点を持たせることで，トランプの 4 種類の札の図形を表す陰函数を作成して見せる．

5.2.6 4 種のトランプ札の陰函数

• ハート

点 $(0,-1),(1,1),(0,3/4),(-1,1)$ を通り，かつ $(0,-1),(0,3/4)$ で，尖点を持つ陰函数を求める．媒介変数表示式は次の 6 次の有理式とし，未定係数を表 5.3 に従って決定する．

$$\begin{cases} x(t) &= (a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 + a_6 t^6)/(t^6 + 1) \\ y(t) &= (b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 + b_6 t^6)/(t^6 + 1). \end{cases}$$

表 5.3: ハートのデータ表

t	$-\infty$	-1	0	1
x	0	-1	0	1
x'	0		0	

t	$-\infty$	-1	0	1
y	-1	1	3/4	1
y'	0		0	

$x(t), y(t)$ のパラメータ（未定係数）はそれぞれ 7 個あるが，有効な制約条はそれぞれ 5 個[11]しかない．そのため任意にとれるパラメータがそれぞれ 2 個ずつあり得る． 実際，パラメータを決定する方程式はいずれも 2 次元であり，これを解くと，a_5, a_4, b_5, b_4 を任意定数として媒介変数表示が次のように求められる．

$$\begin{cases} x(t) &= (a_5t^5 + a_4t^4 + (-a_5+2)t^3 - a_4t^2)/(t^6+1), \\ y(t) &= (-t^6 + b_5t^5 + b_4t^4 - b_5t^3 + (-b_4+9/4)t^2 + 3/4)/(t^6+1). \end{cases}$$

未定係数が 4 つもあると試行錯誤が複雑になりすぎるため，直観であるが，t^5 の係数を 0 とひとまず置くことにする．すなわち，$a_5 = b_5 = 0$ と置いてつぎを得る．

$$\begin{cases} x(t) &= (a_4t^4 - 2t^3 - a_4t^2)/(t^6+1), \\ y(t) &= (-t^6 + b_4t^4 + (-b_4+9/4)t^2 + 3/4)/(t^6+1). \end{cases}$$

続いて，$x - x(t) = 0$ と $y - y(t) = 0$ とから終結式あるいはグレブナー基底を用いて t を消去し，因数分解によって本質的な部分を取り出す．こうして求められた多項式内の任意定数 a_4, b_4 に試行錯誤的に値を代入しては，Risa/Asir の陰函数描画を多数試みた．最終的に，$a_4 = 0, b_4 = 1/2$ を代入し，さらに x 軸方向を 5/4 倍に拡大することで，求めるハート型の陰函数 Heart(x, y) 式（5.5）を得た．

$$\begin{aligned} \mathrm{Heart}(x,y) &= 364816x^6 + (9214800y^2 + 8937600y - 121625)x^4 \\ &+(2520000y^4 - 2250000y^3 - 18877500y^2 - 12656250y + 1880000)x^2 \\ &+4000000y^6 + 3000000y^5 - 8250000y^4 \\ &-4437500y^3 + 6187500y^2 + 1687500y - 1687500. \end{aligned} \tag{5.5}$$

これを図示すると図 5.13 のようになる．なお，多項式内の変数 a_4, b_4 に関して $a_4 = 0$ については，求める図形が左右対称である，つまり $x(t)$ は，奇函数でなくてはいけない事から当然なのであるが，もう片方の $b_4 = 1/2$ は，かなり発見的手法で見つけた値である．（ちなみに，$b_4 = -1$ とすると，リンゴの切り口のような図形が出来上がる．）

[11] $t \to -\infty$ では，もともと $x'(t) \to 0, y'(t) \to 0$ となっているため，$t \to -\infty$ で微分が消える条件は制約とならない．

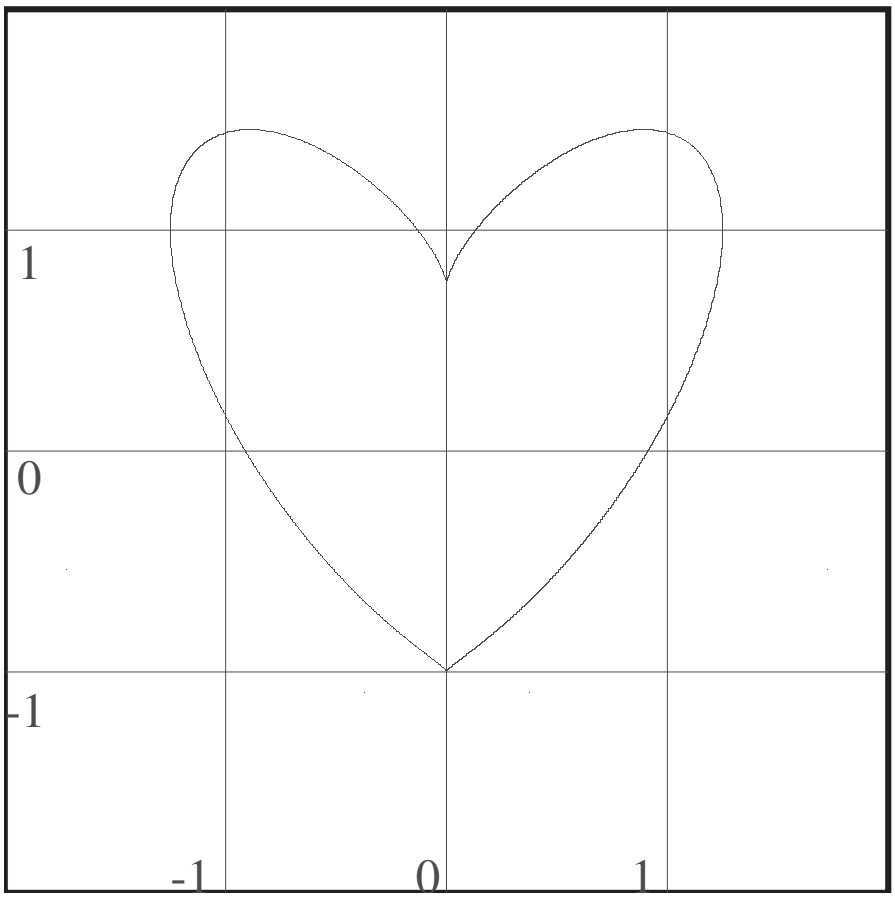


図 5.13: ハート

• ダイア

$x(t), y(t)$ の分子を 6 次式とし，ハートの例 と同じく，表 5.4 から求める．$x(t), y(t)$

表 5.4: ダイアのデータ表

t	$-\infty$	-1	0	1
x	0	-1	0	1
x'	0	0	0	0

t	$-\infty$	-1	0	1
y	-1	0	1	0
y'	0	0	0	0

のパラメータ（未定係数）はそれぞれ 7 個あり，有効な制約条件もそれぞれ 7 個[12)]であるため，任意にとれるパラメータはこの場合にはないことが推定される． 実際，パラメータを決定する方程式は零次元であり，任意に取れるパラメータはこの場合にはない．決定された $x(t), y(t)$ を用いて，$x - x(t) = 0, y - y(t) = 0$ から媒介変数 t を消去し，$y \to 2y/3$ なる変数変換で y 方向に 1.5 倍に引伸ばすことで，ダイアの陰函数 Diamond(x, y) 式（(5.6) が得られた．

$$\begin{aligned}\text{Diamond}(x, y) = {} & 2916x^6 + (10449y^2 - 8748)x^4 + (9504y^4 \\ & +44712y^2 + 8748)x^2 + 256y^6 - 1728y^4 + 3888y^2 - 2916. \end{aligned} \tag{5.6}$$

このグラフを図 5.14 に示す．

[12)] ハートの場合と同様，$t \to -\infty$ では，もともと $x'(t) \to 0, y'(t) \to 0$ となっている．

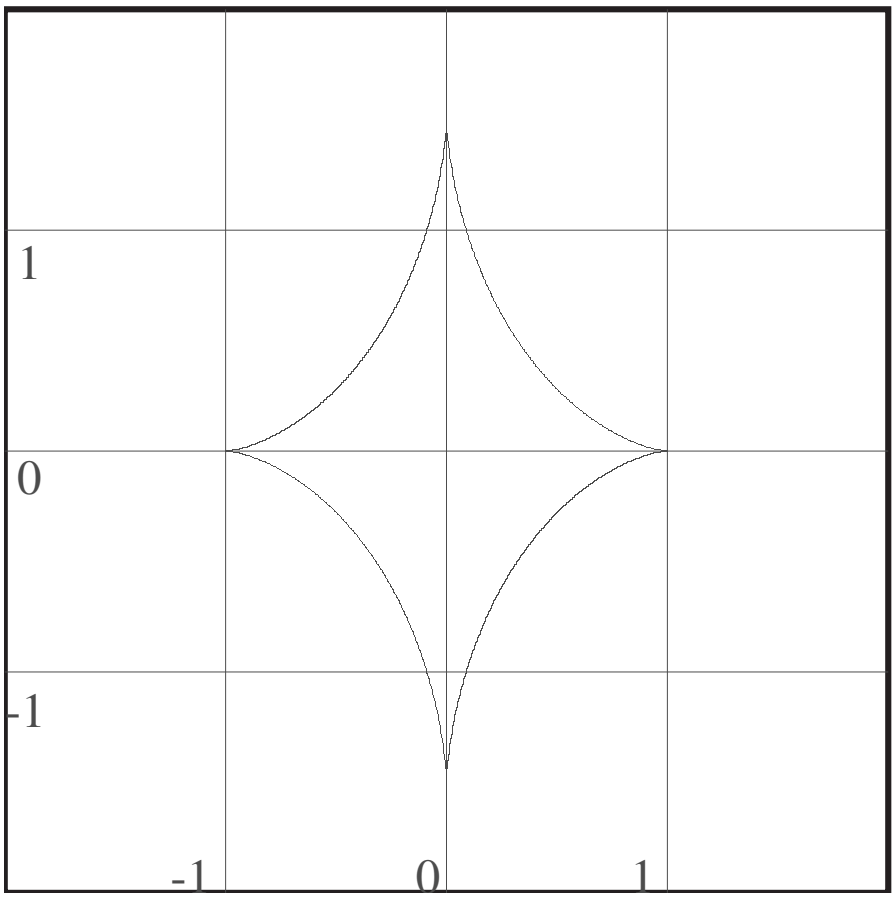


図 5.14: ダイア

残ったスペードとクラブは，ハートやダイヤの例のようには簡単でない．形が複雑で，どの点を通らせるように設定すべきか，さらに媒介変数 t がどの値の時に通らせるべきか，ひいては与える点の個数はどうするか，$x(t), y(t)$ の次数はどうか，など．これらの自由度は非常に高く，試行錯誤を繰り返すことになった．これらの事から，結果的には見つけることの出来たスペードとクラブの陰函数は，ある意味では偶然見つけられたものと言えなくもない．

• スペード

$x(t), y(t)$ の分子分母は，8 次多項式．通過点データは表 5.5 のとおり．これまで同様

表 5.5: スペードのデータ表

t	$-9/5$	$-11/10$	$-4/5$	0	4/5	11/10	9/5
x	$-2/7$	$-3/10$	-1	0	1	3/10	2/7
y	$-1/4$	$-1/8$	0	5/3	0	$-1/8$	$-1/4$
x', y'				0			

に係数について解き，パラメータ t を消去したのち，任意に選べる係数の値を，$a_8 = 0$, $b_8 = -5/14$（a_8, b_8 は，$x(t), y(t)$ の分子の 8 次（最高次）の係数）と決める．さらに，x を $2/3x, y$ を $2/3(y + 3/4)$ とアフィン変換して形を整えると，図 5.15 に示すスペードの陰関数 Spade(x, y) が得られた．

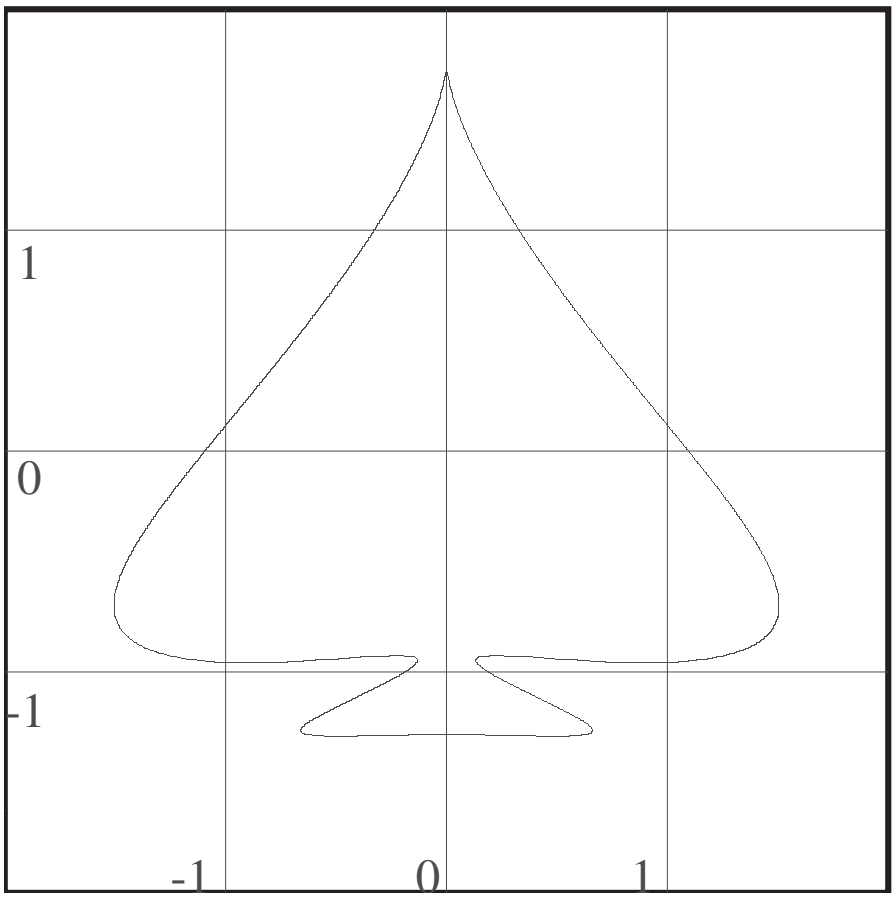


図 5.15: スペード

多項式 Spade(x, y) は付録 B に掲載した.

• **クラブ** $x(t), y(t)$ の分子分母は，10 次多項式．通過点データは表 5.6 のとおり．こ

表 5.6: クラブのデータ表

t	-3	$-5/2$	$-5/4$	$-1/2$	$-2/5$	$2/5$	$1/2$	$5/4$	$5/2$	3
x		-2	0	0	-2	2	0	0	2	
y		-1	0	0	$2/3$	$2/3$	0	0	-1	
x', y'	0									0

れまで同様，係数について制約を解き，t を消去して得られた陰函数において，任意にとれるパラメータを次のとおり決定する．

$a_{10} = 0, b_{10} = -31/30$（$a_{10}, b_{10}$ は，$x(t), y(t)$ の分子の 10 次（最高次）の係数），x を $14/3x, y$ を $4/3(y + 1/2)$ とアフィン変換すれば，クラブ（図 5.16）の陰関数 $\mathrm{Club}(x, y)$ が出来上がる．

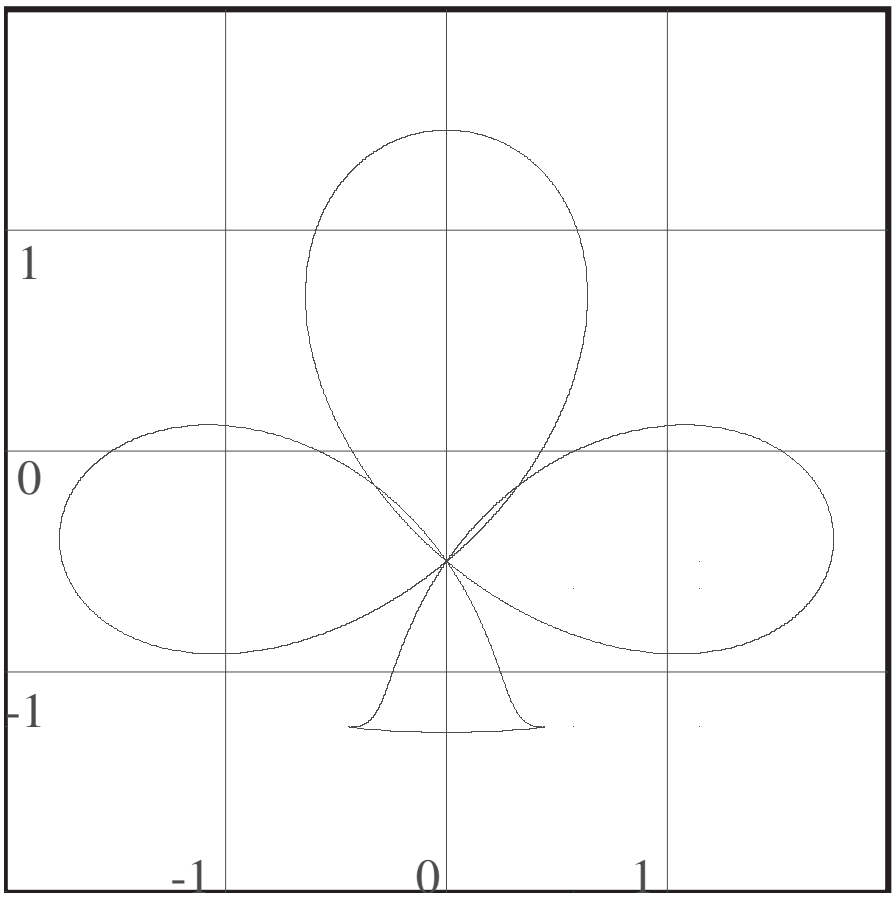


図 5.16: クラブ

多項式 Club(x, y) は付録 C に掲載した.

5.2.7 付記

この研究は，筆者が出題し，オーストリア RISC-Linz の Bernhard Wall 氏（1992 年 1 月〜3 月の間，国際研に客員研究員として滞在.）のアイデアを参考として，下山が中心になって行った [139]．また，尖点を導入した方が図形的に面白いというヒントは齋藤（上智大学）による．なお，Risa/Asir の陰函数描画機能は齋藤のアイデアとプロトタイプ アルゴリズムを 強化・改良し，Risa/Asir の TCP/IP 機能で数式処理システムに結びつけたものである．これらのトランプの図柄を表す陰函数データ（2 変数多項式）は次数が大きくかつ係数の表現も相当大きなことや，特異点（尖点，自交点など）をもち，通常の描画アルゴリズムにとって不利な状況であるため，市販の図

形描画ツールや Risa/Asir 以外の数式処理システムでは描画に失敗するものが多い[13]. このことは 4.1.5 項の 2 変数多項式（4.6）$g(y,z)$ についても同様である.

Risa/Asir では，文献 [116] および特許 [75] に記載された，信頼できる方法で任意の 2 変数多項式の零点を描画できる.

[13] アルゴリズム上の問題，または代数計算能力の問題.

第6章　Risa/Asirの教育への応用

数学教育への数式処理システム[1)]の利用は数式処理システムの歴史とともに開始されているが，計算機および数式処理システムが高価だった時代にはその適用は限られていた．1990 年代の後半パソコンが急激に普及し始めた結果，教育現場での数式処理システム利用も盛んになってきた．

数式処理システム利用に限らず，教育へのテクノロジーの利用は，教育機会の平等性の観点からも，特殊で高価なテクノロジーではないことが望ましい．その点で，Risa/Asir はオープンソースのフリーソフトであり，パソコンさえ用意できれば[2)]，誰でもが利用可能である．

また，我が国は数式処理システム開発後進国であると言ったが，外国製の商用ソフトで分厚い外国語のマニュアルと格闘することなしに，母国語（日本語）のマニュアルが利用できることも Risa/Asir にとっては有利な点である．

唯一の問題は，Risa/Asir の機能がもっぱら多項式（整式）の処理に限られており，一般的な微分積分を含む数式では柔軟な式の取り扱いが困難で，教材に関してそれらを回避する工夫や，特別なプログラミングが必要な点である．

しかしながら，適切な教材を選び，プログラミングを含む適切な利用上のシナリオを工夫すれば，数学概念を教え/学ぶことにおいて，Risa/Asir というテクノロジーが十分活用できるということも分かってきた．

本章では，本来の開発目的とは異なる分野である教育/学習への適用を試み、その可能性を検討する。

6.1　教育応用へのアプローチ

Risa/Asir はもともと理工学の研究開発のための技術として開発されている．これまでは少数の人が Risa/Asir を数学教育に利用していた．なかでも，藤本等のグループは独自に開発した手書き数式認識システムおよび科学技術文書エディタ（Infty Editor）

[1)]教育関係においては Computer Algebra System の略称を用いて CAS と呼ぶことが多いが，本論文では数式処理システムを用いる．

[2)]Windows98 以上が動作するパソコンなら Risa/Asir は軽快に動作する．Unix 系フリー OS 上で動作させるなら，OS もほぼ無料である．

と Risa/Asir とを OpenXM プロトコルによって結合し，数学の教材作成と授業での実践を試みている [107]．さらに，Risa/Asir を手書き数式入力システムおよび科学技術文書エディタとともに PDA（Portable Digital Assistance）に移植し，AsirPad と名付けたシステムを試作した．数式処理を使う上で，パーソナルコンピュータは，機能は高いが，コストも高く，起動終了に時間が掛り機動性に欠ける．一方，グラフ電卓は，機動性に富み比較的低価格ではあるが，表示精度に欠けカラー表示性能も低いことが難点である．AsirPad は，これらの丁度中間で，機動性と低価格，数式入力の容易さなどを両立させたシステムとなっており，高等学校での試行では授業を受けた生徒，見学した現場教師や父兄の評価は高いと報告されている [160, 161, 162, 28].

E. Varbanova は数式処理ソフト DERIVE[3] を用いて微分積分学の教材を作成し, 数式処理システムが微分概念の学習に如何に貢献できるかを示した [81]．筆者は Risa/Asir の教育への応用の道をさぐるべく, Varbanova の研究て扱われている題材が Risa/Asir で実施可能かどうかを再検討することにより, 工夫次第でそれらの題材の多くが Risa/Asir によっても実行可能であることを確認した.

教育/学習に数式処理システムを利用するというアプローチには多くの議論がある．ある概念に関する知識は単にその定義を知るだけでは獲得できるものではない．多くの適切な訓練を実施したり，さらにはより高位の概念を創造するなどの，概念のさまざまな様相において概念を応用することができたとき，初めて獲得できるものである．「応用できなければ分かったことにはならない.」ということである．数式処理システムの能力は，学生/生徒が 3 つの異なる観点，すなわち，数値的な観点と代数的あるいは解析的観点，そして視覚的観点とから，数学概念に接近し，理解することを支援するために利用できる.

Varbanova は微分概念の応用についての教育/学習に関してひとつのシナリオを考察した．題材は日本の微分積分学に関する教科書から採用し，DERIVE システムを利用することによって数式処理システムが数学概念の教育と学習に効果的であることを示した.

微分積分学における重要な概念は基礎概念である「微分」の上に構築される．それゆえ，もし数式処理システムが，微分に関する学生/生徒の理解を深めることができるなら，その上の諸重要概念も効果的に理解させられるはずである.

以下の節では，DERIVE で実行した題材:（1）グラフの傾き，接線，法線，（2）テイラー展開と函数の多項式近似，（3）函数，導函数，2 階導函数とグラフ，およびいくつかの発展的トピックについて，Risa/Asir で同様の教材が作成できることを確認する.

[3] http://education.ti.com/us/product/software/derive/features/features.html　教育用に開発された商用数式処理ソフト.

6.2 応用で利用する Risa/Asir の基本機能

ここでは本章で共通に使用される Risa/Asir のビルトイン関数について説明する．

1. **主たる道具**: 初等函数の微分機能
   `diff(F,X)`: 式 `F` を 変数 `X` で微分し，その結果を返す．

2. **補助道具 1**: 式の評価．

   - `subst(E,X,A)`: 式 `E` の中に出現する変数 `X` を `A` で置き換える．ここに，`A` は置き換え後の式 `E` が数式として意味があるような任意の式である．
   - `deval(E)`, `eval(E)`: 式 `E` を数値的に評価（計算）して結果の数値を返す．`deval(E)` は `E` を倍精度浮動小数点数値で計算し， `eval(E)` は `E` を任意多倍長浮動小数点数値で計算する．

3. **補助道具 2**: グラフ描画．

   - `ifplot(E,options)`: 式 `E` の表す 2 変数函数の零点を描画する．
   - `plotover(E,0,0,Color)`: 式 `E` の表す 2 変数函数の零点を，先に `ifplot` で描画したのと同じウィンドウに色 `Color` で重ねて描画する．

その他，`f(x) := x^2` のような構文で，新しい函数 $f(x)\,(= x^2)$ を定義することができる．このように定義した函数は数式が許される任意の場所で使用することができる．たとえば，$f(3)$ は $9\,(= 3^2)$ として評価される．

Risa/Asir の名前に関する特徴としてつぎのことは注意する必要がある．すなわち，英小文字で始まる英数字の識別子（identifier）は，英大文字で始まる英数字の識別子とは全く異なる使い方がされるということである．前者（英小文字で始まる識別子），例えば `x` や `sin` など，は数式の中に表れる不定元（x，しばしば変数とも呼ばれる．）や数学函数（sin）を表すのに用いられる．一方，英大文字で始まる識別子，例えば `X` や `Fx` など，はプログラミング言語 *Asir* のプログラム変数（*Asir* が扱うデータの格納先を識別する．）を表すために用いられる．

6.3 グラフの傾き，接線，法線

Risa/Asir は与えられた函数 $f(x)$ の導函数を計算することができる．対象となる函数は，多項式函数，有理函数，冪乗函数，指数函数，や通常の初等函数（$\exp(x)$, $\sin(x)$, $\cos(x)$, $\tan(x)$ やそれらの逆関数など）である．根号を用いた代数函数，例えば $\sqrt{x}$, は分数冪を用いた表現，`x^(1/2)`，により代用される．

例 2
函数 $f(x) = \sqrt{x}$ が与えられたとき，$y = f(x)$ によって定義される曲線の $x = 1$ に於ける傾きを求めよ．

これは Asir ではつぎのように計算される．

```
[12] f(x):=(x)^(1/2);       /* function sqrt(x) is defined as f(x).*/
[14] df(x):=diff(f(x),x);   /* derivative of f(x). */
[15] df(x);
1/2*((x)^(-1/2))        /* negative exponent is not reduced, here. */
[16] /* slope of the graph y=f(x), i.e. y=sqrt(x) at (1,f(1))=(1,1)*/
M=df(1);        /* Or, in general, we should use subst(df(x),x,1); */
1/2*((1)^(-1/2))     /* the result is not sufficiently simplified. */
[17] /* Use deval( ) to get a number value, an approximate value.  */
Mf=deval(df(1));   /* same result will be obtained by Mf=deval(M); */
0.5
```

この結果が正しいことは手計算と比較することによって容易に分かる．

注意 13
DERIVE では接線の傾きは正確な数値 $\frac{1}{2}$（有理数）として与えられる．しかし, Risa/Asir では近似値 （浮動小数点数値 0.5）として与えられる[4)]．

ついで傾きの角度をラジアンで求める．これには函数 $\tan^{-1}(x)$ を用いればよい．$\tan^{-1}(x)$ は多価函数であるが, 数学ソフトでは通常その主値を与える関数として `atan(x)` を提供している．Risa/Asir も同様であり，傾き角のラジアン値はつぎのようにして得られる．

```
[19] ThetaInRadian = deval(atan(Mf));
0.463648
```

角度の単位を「ラジアン」から「度」に変換するには，変換係数として $\left(\frac{180}{\pi}\right)_{deg/rad}$ を `ThetaInRadian` に乗じれば良い．

```
[20] ThetaInDegree = ThetaInRadian * deval(180/@pi);
26.5651
```

注意 14
DERIVE では通常の記法と同様の 1° という記法で数値 $\left(\frac{\pi}{180}\right)_{rad/deg}$ を表現することができる．

[4)] 数値計算に於ける浮動小数点数値（近似値）と数式処理（代数的計算）での有理数値（正確な値）を区別することの重要性は，学生に別途説明することが必要である．

つぎに 点 $(x,y)=(1,f(1))$ における接線の方程式を求めるにはつぎのようにする．手計算では，点 $(x,y)=(x_0,y_0)$ を通り傾きが m の直線の公式 $y-y_0=m(x-x_0)$ に，$x_0=1, y_0=f(x_0)=1, m=\mathtt{Mf}=0.5$ を代入することにより目的の接線の方程式が得られる．数式処理システムを用いるなら，この各ステップを追って式を構成すればよい．次の Risa/Asir のコマンドラインでは，Lhs, Rhs は使おうとしている公式の左辺と右辺の式を保持させるためのプログラム変数である．

```
[21] /* The left hand side of the equation y-y_0: */
Lhs = y-1;
y-1
[22] /* The right hend side m (x-x_0): */
Rhs = Mf*(x-1);
0.5*x-0.5
```

左辺から右辺を引くことで，つぎの x,y の 1 次式が得られる．

```
[23] /* So, an implicit form of the equation of the */
/* tangent line is */
IMP_EqTangLine = Lhs - Rhs;
-0.5*x+y-0.5
```

これは方程式の右辺を 0 と規格化したときの左辺の 1 次式である．

注意 15
DERIVE は，等式をデータとして扱うことができる．しかし，Risa/Asir は等式はデータではない．そのため，Risa/Asir のユーザが等式を扱う一つの方法は，等式の右辺をいつも 0 となるように規格化した上で，その左辺の多項式のみを扱うことである．Risa/Asir で等式が扱えないという不便は現在見直し中であり，近い将来には改善される予定である．

従属変数 y を独立変数 x の函数として表す標準形 $y=ax+b$ を得るためには，IMP_EqTangLine を y から引けばよい．ここでも，Risa/Asir では等式が扱えないために，右辺の 1 次式を得ることで代替する．

```
[25] Rhs_y = Lhs_y - IMP_EqTangLine;
0.5*x+0.5
```

注意 16
DERIVE では接線の切片標準形 $(y=ax+b)$ の右辺を求める関数 TANGENT(E,X,X0) が提供されている．これは，任意の式 E により $y=$ E で定義されるグラフの接線について，その方程式の右辺の 1 次式 $(ax+b)$ を直接計算することができる．

Risa/Asir はそのような関数を標準関数として提供してはいないが，Risa/Asir の基本機能を用いて定義することにより，同様の関数を容易にシステムに追加することができる．

つぎにその関数定義の一例を示す．

```
def derive_tangent_line_RHS(F,X,X0){
  return deval(subst(diff(F,X),X,X0))*(X-X0)
            +deval(subst(F,X,X0));
}
```

この Asir 関数 derive_tangent_line_RHS(F,X,X0) は y = F のグラフの X = X0 における接線の方程式（標準形）の右辺を与える．つぎが使用例である．

```
[27] derive_tangent_line_RHS(f(x),x,1);
0.5*x+0.5
```

問題を少し発展させて陰的に定義される函数について，その接線を求めることも可能である．陰函数の微分の概念は学んでいるものとして，その知識を応用することが目的である．

平方根函数を陰的に表現して $g(x,y) = y^2 - x = 0$ と置く． こうして陰的関係 $g(x,y) = 0$ の零点のグラフを考える．Risa/Asir では前と同様にして $g(x,y)$ を定義する．

```
[28] g(x,y):=y^2-x;
[29] g(x,y);
-x+y^2
```

注意 17

DERIVE は IMP_TANGENT(E,X,Y,X0,Y0) という関数により，E = 0 の表すグラフ（曲線）の，点 (X, Y) = (X0, Y0) における接線の方程式の右辺の 1 次式 aX + b を計算させることができる．ここに，E は 2 変数 X,Y に関する式である．

この DEIRVE の関数と同じ機能を果たす Risa/Asir の関数
derive_IMP_tangent_line_RHS(F,X,Y,X0,Y0) も容易にプログラムすることができる．これは，F が X と Y の表す 2 変数の式であるとき，F = 0 の表すグラフ（曲線）の点 (X0, Y0) における接線の方程式の右辺の 1 次式を与える．ただし，点 (X0, Y0) は F = 0 を満さねばならない．すなわち，その点は F = 0 の定義する曲線上になければならない．本例では (X0, Y0) = (1, 1) である．Asir 関数の定義はつぎのとおりである．

```
def derive_IMP_tangent_line_RHS(F,X,Y,X0,Y0){
  DFX=diff(F,X); DFY=diff(F,Y);
  M= -DFX/DFY;
  return deval(subst(M,X,X0,Y,Y0))*(X-X0)+Y0;
}
```

F に f，X に x，Y に y を与えたとき，$\frac{\partial f}{\partial x}$ は diff(F,X) で，$\frac{\partial f}{\partial y}$ は diff(F,Y) でそれぞれ得られ，したがって，上記プログラムで M= -DFX/DFY; により陰函数の接線の傾き $m = -\frac{\partial f}{\partial x}/\frac{\partial f}{\partial y}$ が得られることを解説すれば，つぎの使用例と併せて，陰函数で定義されるグラフの接線と偏微分/微分の関係を学生が再確認することができる．つぎが使用例である．

```
[30]derive_IMP_tangent_line_RHS(g(x,y),x,y,1,1);
0.5*x+0.5
```

曲線の法線の式を得るためにも同様のアプローチができる．法線の傾きは接線の傾きの逆数の符号を逆にして得られるので，$f(x) = \sqrt{x}$ のグラフへの $(x, f(x)) = (1, 1)$ における接線の傾きはつぎのとおり．

```
[31] Mf_normal = -1/Mf;
-2
```

接線の方程式を求めたときと同様に，左辺と右辺を LhsNormal と RhsNorml に与える．

```
[35] LhsNormal = y-1;
y-1
[36] RhsNormal = Mf_normal*(x-1);
-2*x+2
```

左辺から右辺を引くと陰的表現（の方程式の左辺）が得られる．

```
[37] EqNormalLine = LhsNormal - RhsNormal;
2*x+y-3
```

学生達はディスプレイ上に結果を表示することで，数式処理システムを利用して得られたこれらの結果の正しさを確認できる．Asir で $\sqrt{x}$ の曲線，接線，法線のグラフを描くことは ifplot 関数[5] により行う．ただし，グラフ描画のオプション機能は限定されている．

[5] ifplot は implicit function plot から来ており，2 変数多項式の零点を描画する機能を持つ．引数を y-f(x) と 2 変数函数にしているのはその故である．

```
[38] ifplot(y-f(x),[x,-1,3],[y,-0.5,3.5]);
0
[39] plotover(IMP_EqTangLine,0,0,0xff0000);
0
[40] plotover(EqNormalLine,0,0,0x0000ff);
0
```

上記コマンドラインで Risa/Asir により描画されたグラフは，図 6.1，図 6.2，および，図 6.3 に示されている．

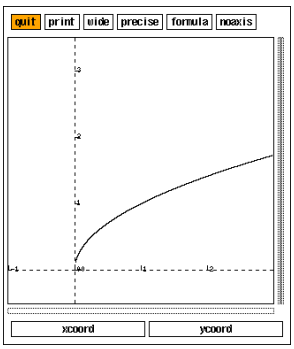


図 6.1: $y = \sqrt{x}$ [38] ifplot による．

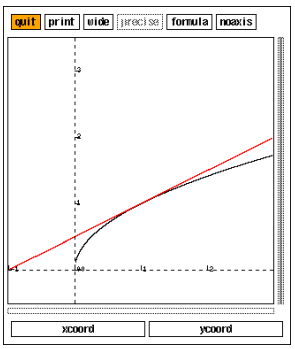


図 6.2: 接線 [39] plotover による重ね描き．

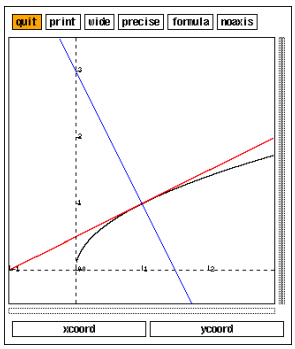


図 6.3: 法線 [40] plotover による重ね描き．

微分積分学の学習をさらに進めると，曲線に対する接平面や法線を求める問題も現われる．その際には，ここで学んだと同じ論証の方法や同様の手順を適用する機会が学生達には生じよう．

注意 18
現状の Risa/Asir は数式の簡約化が得意ではない．教育の場では，根号を含む数式が頻繁に現われるため，根号を含む数式の簡約化への強い要望がある．それは，完全性が要求されるならば非常に困難な問題として知られているが，よく現われる簡単な場合に関しては，今が研究テーマとする良い時期である．

6.4　テイラー展開と函数の多項式近似

この節の話題は，学生達が数学に対する正しい感覚を発達させ，厳格な知識と批判的な考証の必要性に気付くよう支援する，重要なものである．演習問題を通して学生

達は，広く近似概念について多くを学び取り，かつ「生兵法は怪我の素[6)]」という格言をも経験することになろう． 後に，学生達は同じ概念を様々な文脈—近似的解析解，ODE の数値解，方程式の近似根，数値積分など—においても考察することになる．

学生達はテイラー展開公式を打ち切り冪級数，すなわち各項が昇冪の順に並べられた多項式として教えられる．基本操作としての微分操作に基づくテイラー展開アルゴリズムの設計はテイラー展開概念自体の理解を強固なものにするための良い訓練になる．この節では，DERIVE が提供するものと同じ既成の Taylor 展開関数が使用できることを仮定し，それを如何に利用するかを考える．

6.4.1 $\sin(x)$ のテイラー展開

対象函数を $\sin(x)$ とし，簡単のために展開の中心は原点 $x = 0$ に固定しておく．また学生達は，テイラー展開の剰余項 $R_n = (x^{n+1}/(n+1)!)f^{(n+1)}(\theta x)$，および打ち切り誤差とその限界の見積もり $E_n(x)$ についての知識が既にあるものと仮定する．目下の問題に対する打ち切り誤差限界の見積もりは，

$$E_n(x) = \frac{|x|^{n+1}}{(n+1)!} \tag{6.1}$$

である．つぎの 3 つの問題を考察しよう．

1. 区間を与えてその中での誤差を計算すること．
2. 近似の次数と要求精度とを与えて，x の最大区間幅を求めること．
3. x の区間と許容誤差とを与えて 最小の近似次数を求めること．

グラフィクスによる可視化はこれらすべての問題において近似を理解する上で大変役立つ．

テイラー展開を行う Asir 関数 `taylor(F,X,X0,Order)` はつぎのように定義される．この関数は `X` = `X0` の周りでの函数 `F` の `Order` 次のテイラー展開を返す．

[6)]“A little knowledge is a dangerous thing.”

```
def taylor(F,X,X0,Order){
  S=subst(F,X,X0);        /* テイラー展開の初期値 */
  DF=diff(F,X);  /* 導函数の初期値．定数項に対応 */
  Dn=                /* 分母の初期値．定数項に対応 */
  for(I=1; I<=Order; I++){
    Dn*=I;
    S+=subst(DF,X,X0)/Dn*(X-X0)^I;
    DF=diff(DF,X);
  }
  return S;
```

さらにつぎの補助関数 simplify_constants(S) も使用する．これは式 S 中に現われ，最終的には定数値となる部分式を定数値に簡約化する．たとえば，つぎのような簡約を行う．$\sin(0) \to 0, 1^{\frac{1}{2}} \to 1$.

```
def simplify_constants(S){
  S1=S;                      S1=subst(S1,sin(0),0);
  S1=subst(S1,cos(0),1);     S1=subst(S1,exp(0),1);
  S1=subst(S1,exp(1),@e);    S1=subst(S1,log(@e),1);
  S1=subst(S1,log(1),0);     S1=subst(S1,tan(0),0);
  S1=subst(S1,(1)^(1/2),1);
  return S1;
}
```

6.4.2　低次の近似 $n = 3$ の場合

先ず 3 次のテイラー展開を考察する．

プログラム変数 F に $\sin(x)$ を与え，Ft=taylor(F,x,0,3) によって $\sin(x)$ の 3 次のテイラー展開式を計算し，Ft に格納する．simplify_constants(Ft) によって $\cos(0)$ や $\sin(0)$ などの定数式を定数に置き換え式全体を簡約する．

```
[21] F=sin(x);
sin(x)
[22] Ft=taylor(F,x,0,3);
```

```
-1/6*cos(0)*x^3-1/2*sin(0)*x^2+cos(0)*x+sin(0)
[23] Ft=simplify_constants(Ft);
-1/6*x^3+x
```

注意 19
[22] の結果と [23] の結果とを比較すれば，テイラー展開にとって良い簡約化機能が重要であることが了解される．ここで用いた簡約化関数は単に当座の問題に対処するためのものであり，万能のものではない．

つぎに区間 $-1/5 \le x \le +1/5$ におけるテイラー展開の誤差を計算する．誤差限界は式（6.1）において $n = 3$ として与えられる．その誤差限界は区間の両端 $x = \pm 1/5$ で最大になることが容易に分かる．よって，区間内の誤差限界を得るには $\frac{1}{(3+1)!}(1/5)^{3+1}$ を計算すればよい．

```
[24] Emax=1/fac(3+1) * (1/5)^(3+1);
1/15000
```

この誤差限界はタイトでないことに注意すべきである．数学的知識に基づき，我々は $\sin(x)$ の原点周りのテイラー展開に偶数次項が存在しないことを知っている．なぜなら，偶数次項は因子 $\sin(0)$ を含むため展開には現われないからである．したがって，誤差の限界式としては式（6.1）において $n = 4$ とする必要があった．

改善された誤差限界の見積もりはつぎのようになる．

```
[25] Emax=1/fac(4+1) * (1/5)^(4+1);
1/375000
[26] 1.0 *Emax;
2.66667e-06
```

こうして，3 次のテイラー展開はここで指定された x の区間において，小数点以下に少なくとも 10 進 5 桁の有効桁をもっていることが結論できる．

注意 20
DERIVE を使用した場合には, 実数値に対する絶対値函数が利用可能なことに加え，不等式に対する DERIVE の演算機能の支援をうまく利用することができる DERIVE が提供するそれらの機能により，利用者は与えられた関係式 $|x| \le 1/5$ から目的の不等式

$$\frac{|x|^5}{5!} \le \frac{(1/5)^5}{5!}$$

を手計算で得る導出過程を，ステップバイステップでシミュレートすることができる．このように等式/不等式が取り扱えることで，DERIVE は，明示的に与えられた関係式からの目的の関係式の導出を容易にするという特長を持っている．

つぎに展開区間をより広く $-1 \leq x \leq +1$ 取った場合にどういうことが観察できるか調べてみよう．誤差限界は前段での議論と同様，$n = 4$ かつ誤差の最大は区間両端すなわち $x = \pm 1$ において生じるとしてつぎのように計算できる．

```
[27] Emax=1/fac(4+1)*(1)^(4+1);
1/120
[28] 1.0*Emax;
0.00833333
```

この場合には近似精度は少数点以下僅かに 10 進 2 桁でしかない．前段の結果と比較することにより，区間幅が増加すれば近似精度は減少し，区間幅が減少すれば近似精度は増加する，ということが分かる．この観察事実はつぎの演習問題を解くことにより確かめられる．

逆問題を考える．すなわち，展開の次数と要求精度とが与えられたとき，許容される x の区間幅の最大値を求める．

今，$n = 3$ とし，最大許容誤差を Emax $= 0.0005$ とする．すると，問題はつぎの不等式（6.2）の解を求めることとなる．

$$\frac{|x|^{(4+1)}}{(4+1)!} \leq \texttt{Emax} = 0.0005. \tag{6.2}$$

注意 21

Risa/Asir はいわゆる solve 関数を提供していない．したがって，我々は x の範囲を陽に定める不等式を手計算によって導いておく必要がある．手計算の結果は $|x| \leq (5! \times 0.0005)^{(1/5)}$ となる．

式の右辺 $(5! \times 0.0005)^{(1/5)}$ が許される x の絶対値の最大を与える．そこでこの値をつぎのように計算する．まず，分数冪の底は

```
[29] 0.0005*fac(5);
0.06
```

となる．この 5 乗根を取ることでつぎのとおり x の最大許容値を得る．

```
[30] deval(0.06^(1/5));
0.569679
```

この結果により，テイラー展開の近似は少なくとも区間 $-0.569679 < x < 0.569679$ において成立している[7)]．

[7)]数学的には本質的ではないが，丸めによる誤差のためにこの区間を定める不等式は保守的に（安全側に）取り扱うべきである．このことは深刻な問題を解析している場合には特に注意するべきである．本例では第 6 番目の桁にある 9 は数値誤差により汚されている可能性がある．

ついで展開次数 $n = 3$ は変えず，最大許容誤差を Emax = 0.00001 とかなり小さくとった場合を調べてみよう．前回と同様に問題は次の不等式（6.3）を満す x を求めることとなる．

$$\frac{|x|^{(4+1)}}{(4+1)!} \leq \texttt{Emax} = 0.00001. \tag{6.3}$$

手計算により x について解くと，$|x| \leq (5! \times 0.00001)^{(1/5)}$ を得る．右辺を計算することにより x のとれる最大値は

```
[31] deval( (0.00001*fac(5))^(1/5));
0.260517
```

と計算される．よって，この場合には x の区間は $-0.260517 < x < +0.260517$ となり，前回の区間 $-0.569679 < x < 0.569679$ の幅に対して 1/2 以下となった．むろんこの現象は期待していた範囲内，すなわち許容誤差が減少すると区間幅が減少するという予測の範囲内である．

最初の問に対する直接の逆問題を吟味するために，$n = 3$ に対して最大許容誤差を Emax $= 2.666 \times 10^{-6}$ と取って，このようにした場合に近似が保証される区間が元の問題の区間 $-1/5 \leq x \leq +1/5$ に整合することを確かめよう．これはつぎの計算でただちに確認される．

```
[32] deval( (2.6666*10^(-6)*fac(5))^(1/5));
0.199999
```

この計算結果は，保証される区間が $-0.199999 \leq x \leq +0.199999$ となることを示しており，数値計算誤差を考慮した上で元の問題の区間を再現しているということができる．

6.4.3 高次の近似 $n = 10$ の場合

ここでは展開の次数を $n = 10$ と取った場合を吟味する．テイラー展開は Asir によってただちに計算される．

```
[33] F10=taylor(sin(x),x,0,10);
-1/3628800*sin(0)*x^10+1/362880*cos(0)*x^9+1/40320*sin(0)*x^8
 -1/5040*cos(0)*x^7-1/720*sin(0)*x^6+1/120*cos(0)*x^5
 +1/24*sin(0)*x^4-1/6*cos(0)*x^3-1/2*sin(0)*x^2+cos(0)*x+sin(0)
[34] F10=simplify_constants(F10);
1/362880*x^9-1/5040*x^7+1/120*x^5-1/6*x^3+x
```

近似される元の函数 sin(x) とテイラー展開で得られた近似多項式とを視覚化して比べてみると，展開次数を上げることによる近似精度への効果が良く分かり教育には効果的である[8]．これは陰函数描画関数 `ifplot` と `plotover` とにより実行できる．

```
[35] ifplot(y-sin(x),[x,-8,8],[y,-2,2]);
0
[36] plotover(y-F10,0,0,0xff0000);
0
[37] plotover(y-Ft,0,0,0x0000ff);
0
```

Risa/Asir により描画されたグラフを図 6.4，図 6.5，図 6.6 にそれぞれ示した．

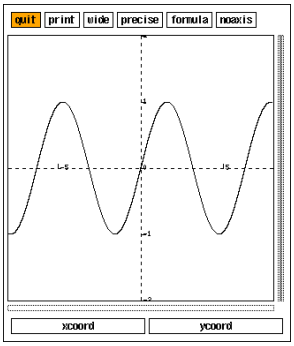


図 6.4: $y = \sin(x)$
（[36] の `ifplot`）

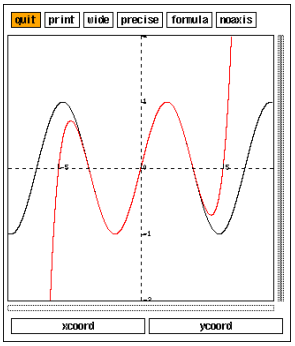


図 6.5: テイラー展開
次数 $n = 10$
（[37] の `plotover`）

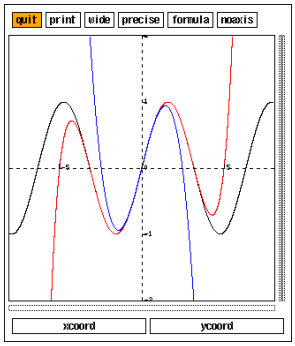


図 6.6: テイラー展開
次数 $n = 3$
（[38] の `plotover`）
$\sin(x)$ との一致範囲が 10 次のテイラー展開に比べて狭い．

これらのグラフにより，学生には，展開の次数が近似精度や近似区間に及ぼす影響が直観的に理解できる．さらに学生への質問として，得られた多項式を近似式として使用することが適当である範囲はどうか?，とグラフ上で答えさせることも可能である．

最後の質問はつぎのとおりである．

[8] $\sin(x)$ は超越的な函数であるため数式処理の意味での「正確な（exact）」計算はできず，厳密に言えば $\sin(x)$ がどのような近似式で計算されるのかが問題になる．しかし，一般に権威ある数値計算ライブラリを用いた計算であるなら，ここで例とするテイラー展開近似によるものよりも「近似の精度が十分高い（sufficiently accurate =精密）」近似式による計算であり，本例のような比較の目的には問題ない．

問題 5
x の区間とその区間内での最大許容誤差が指定されたとき，その近似条件を満す最小の展開次数を求めよ．

ここで，問題を複雑にしないために，かつ実用的な思考を習慣付けるために，若干の事前の考察が必要である．sin 函数は周期函数であり，かつ $\sin(x) = \sin(\pi - x)$ が成立する．したがって，区間 $-\pi/2 \le x \le +\pi/2$ を対象とすれば十分である．よって，この区間に対して最大許容誤差を保証する最小の展開の次数 n を求めることを課題とすれば教育の目的は十分達成できる．

ここでも誤差限界は式（6.1）により与えられる．その値は区間の両端で最大値をとることも明らかである．

事前の手計算によって，区間 $-\pi/2 \le x \le +\pi/2$ において所要の精度を満すための制約条件は式（6.4）となる．

$$\max_{-\pi/2 \le x \le +\pi/2} \left(\frac{|x|^{(n+1)}}{(n+1)!} \right) \le \texttt{Emax}. \tag{6.4}$$

ここでは最大精度を $\texttt{Emax} = 0.001$ と指定した場合を考察する．すると，問題は上記制約式を満す n を $\texttt{Emax} = 0.001$ かつ $x = \pi/2$ で求めることとなる．すなわち，

$$\frac{|\pi/2|^{(n+1)}}{(n+1)!} \le 0.001 \tag{6.5}$$

を解けばよい．実際にはこれと同値でより計算機での計算向きに変形した

$$(n+1)! - 1000 \times |\pi/2|^{(n+1)} \ge 0. \tag{6.6}$$

を用いる．この問題は指数函数を含む離散問題である．たとえ n が実数を走る変数だとしてもその解を陽に表示することはできそうもない．そうではあるが，我々は数式処理システムを利用しているのであるから，その腕力に頼ることが可能である．つまり，正整数 n を 1 から始めて適当な範囲まで増加しながら，非負の値が得られるまで，不等式（6.5）の左辺を数値的に評価することを繰り返せば良い．つぎの Risa/Asir の実行例では M は $n+1$ のことである．

```
[6] for(M=2; M<15; M++)
      print([M,fac(M) -1000*(deval(@pi/2))^M]);
[2,-2465.4]
[3,-3869.78]
[4,-6064.07]
[5,-9443.12]
[6,-14301.7]
```

```
[7,-18556]
[8,3255.43]                    <<<<ここで初めて正の値が得られている.
[9,304659]
[10,3.53735e+06]
[11,3.97731e+07]
[12,4.78776e+08]
[13,6.22667e+09]
[14,8.71777e+10]
```

上記の実行結果の各行のリスト（`[2,-2465.4]` など）は `M` の値（`2` など）とそれに対するの不等式（6.6）の左辺の値（`-2465.4` など）を示している．よって，右側の値が非負になる最初の `M`$(= n+1)$ の値が求める最小の n を与えるものである．上記実行結果からは `M` $= 8$ となり，したがって，近似精度条件を満すな展開次数 n として $n \geq 7$ が得られる．

では展開次数を $n = 7$ として，テイラー展開が区間 $-\pi/2 \leq x \leq +\pi/2$ に対して所望の精度条件を満しているか検証してみよう．次数 $n = 7$ に対するテイラー展開は次の通り得られる．

```
[23] F7=taylor(sin(x),x,0,7);
-1/5040*cos(0)*x^7-1/720*sin(0)*x^6+1/120*cos(0)*x^5
 +1/24*sin(0)*x^4-1/6*cos(0)*x^3-1/2*sin(0)*x^2+cos(0)*x+sin(0)
[24] F7=simplify_constants(F7);
-1/5040*x^7+1/120*x^5-1/6*x^3+x
```

検証は, 精度を保証した区間内の 1 点 $x = \pi/6 < \pi/2$ とその区間外の 1 点 $x = 5\pi/6 > \pi/2$ とにおいて，正確[9]な値と多項式近似による値とを比較することにより行う．

まず，肯定的な結果が期待される精度保証される区間 $-\pi/2 \leq x \leq +\pi/2$ の内部に取った 1 点ではつぎのとおりとなる．

```
[25] deval(subst(F7,x,@pi/6));
0.5
[26] deval(sin(@pi/6));
0.5
```

期待どおり 2 つの値が一致[10]していることが確認できる．

一方，否定的な結果が期待される精度保証される区間外の 1 点ではつぎのとおりとなる．

[9]前言したが，ここでの「正確」とは数式処理的（数学的といったほうがより適切）な exact の意味ではなく，数値計算的な accurate の意味である．数値計算である限り，いずれの値も「exact」ではあり得ず近似値でしかない．

[10]有効桁（この場合は表現桁で 10 進 6 桁）の範囲で一致することが確認できることを意味する．

```
[27] deval(subst(F7,x,5*@pi/6));
0.485029
[28] deval(sin(5*@pi/6));
0.5
```

こちらも期待どおりである．すなわち，2 つの値は異なっている．

実用的な観点からは，7 次のテイラー展開を利用して $\sin(5\pi/6)$ の精密な値が求まることを，学生に指摘しておくとよい．この問題を設定するに当って考察したことより，$\sin(5\pi/6) = \sin(\pi/6)$ が成立しており，$\sin(\pi/6)$ は精度保証区間内の値として計算できるからである．

では，F7 に得られた 7 次の多項式近似と $\sin(x)$ の 2 つの函数のグラフを重ねて視覚化しよう，近似誤差となるそれらの函数の差 F7 − $\sin(x)$ も見せると良い．

```
[29] ifplot(y-sin(x), [x,-0.5,3.5],[y,-0.5,1.5]);
0
[30] plotover(y-F7,0,0,0xff0000);/* 0xff0000 は赤の色コード */
0
[31] ifplot(y-(F7-sin(x)), [x,-0.5,3.5],[y,-0.1,0.1]);
0
```

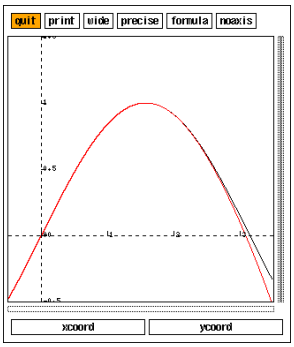


図 6.7: $y = \sin(x)$（上側）と 7 次の近似多項式（下側）

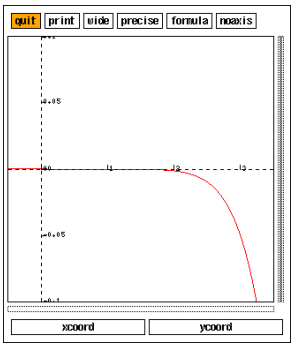


図 6.8: 7 次の場合の近似誤差

$\sin(x)$（上側）と F7（下側）のグラフを 図 6.7 に示す．誤差 F7 − $\sin(x)$ は図 6.8 に示した．

これらのグラフからは $\sin(x)$ が正の区間 $0 \le x \le +\pi/2$ において 7 次の近似多項式で良く近似されていることが観察できる．函数 $\sin(x)$ も 7 次の近似多項式も共に奇函数であるので，負の区間 $-\pi/2 \le x \le 0$ に対しても良い近似であると言える．

6.5　函数，導函数，2 階導函数とグラフの形状

この節では，一定の区間における函数のグラフの振る舞いと，同じ区間における 1 階および 2 階の導函数の符号との間の興味深い関係について議論する．

函数 $f(x) = x^2 \exp(-x)$ を例として $y = f(x)$ のグラフを吟味する．

問題 6 (第一問)
グラフはどの象限にあるか? その理由を言え．

答は Risa/Asir のグラフ描画により視覚的に確かめることができる．

```
[1] f(x):=x^2*exp(-x);
[2] f(x);
exp(-x)*x^2
[3] ifplot(y-f(x),[x,-2,6],[y,-0.3,1.2]);
0
```

実行例の入力 [3] で ifplot によって得たグラフを図 6.9 に示す．

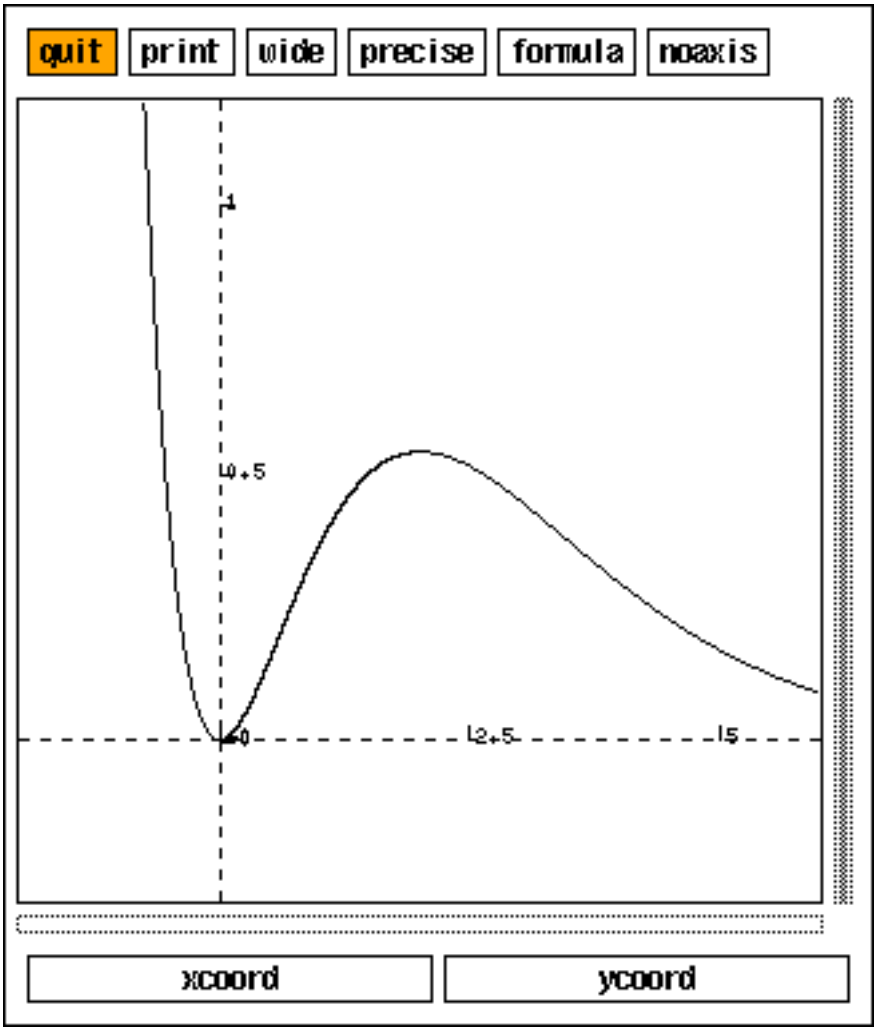


図 6.9: $f(x) = x^2 \exp(-x)$

グラフは，第 1 象限と第 2 象限とに存在することが分かる．すべての実数 x に対して $\exp(-x)$ が正であること，ならびに， $x = 0$ で 0 になることを除き，すべての実数で x^2 が正であること，の 2 つを，理由の説明のために注意しておこう．

Risa/Asir に限らず数式処理システムでは $f(x) = x^2 \exp(-x)$ の導函数を得ることは容易である．また，陽函数なのでそのグラフを描くことも容易である．

つぎの実行例では `df(x)` が導函数 $f'(x)$ を表すように定義している．

```
[4] df(x):=diff(f(x),x);
[5] df(x);
-exp(-x)*x^2+2*exp(-x)*x
```

```
[6] fctr(df(x));
[[-1,1],[exp(-x),1],[x,1],[x-2,1]]
```

[6] では導函数を因数分解している．これは後で，導函数の零点を調べるために役立つ．函数 $f(x)$ とその導函数 $f'(x)$ のグラフを重ね描きした図を図 6.10 に示す．

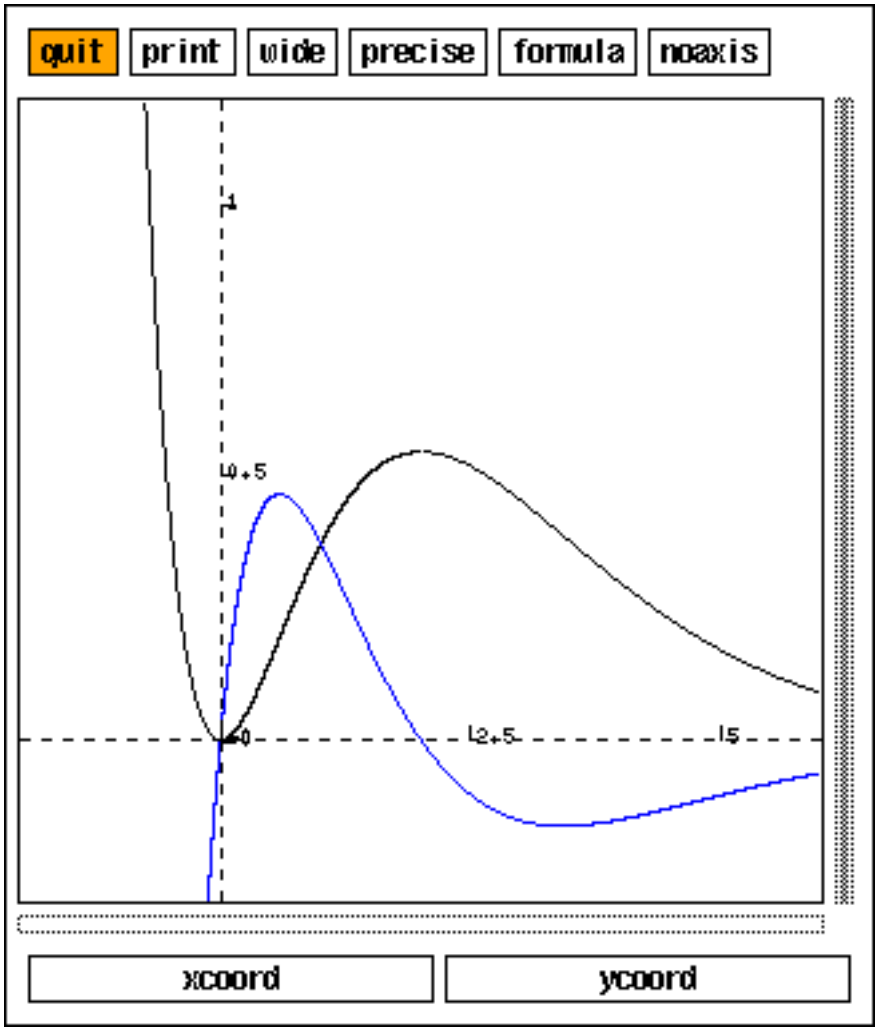


図 6.10: $f(x)$ とその 1 階導函数 $f'(x)$

ここで，第 2 問を提出する．

問題 7 (第 2 問)
区間を指定して，函数 $f(x)$ のグラフの振る舞いとその導函数 $f'(x)$ の符号に関して観察するところを述べよ．

2 階導函数も数式処理システムでは容易に計算できる．つぎの実行例では ddf(x) は 2 階導函数 $f''(x)$ を表すように定義している．

```
[13] ddf(x):=diff(df(x),x);
[14] ddf(x);
exp(-x)*x^2-4*exp(-x)*x+2*exp(-x)
```

函数 $f(x)$ のグラフとその 2 階導函数 $f''(x)$ のグラフとを同じ座標に重ね描きして，図 6.11 に示す．

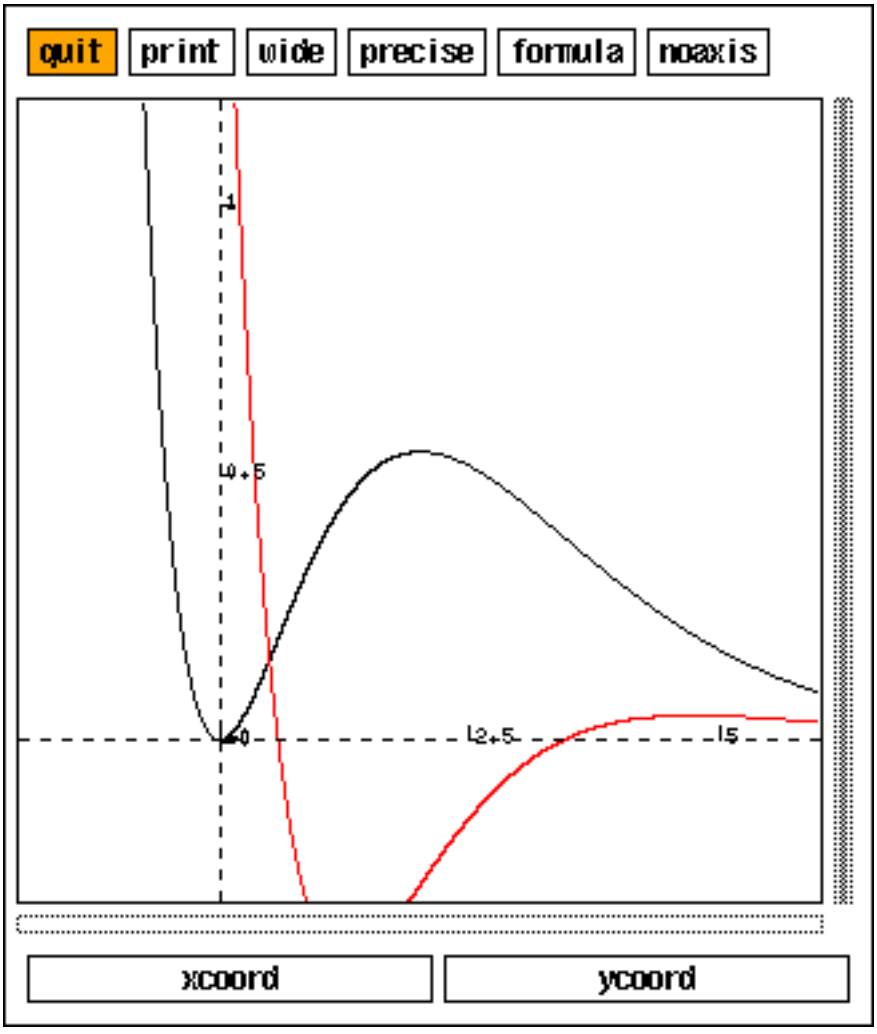


図 6.11: 函数 $f(x)$ と 2 階導函数 $f''(x)$

視察によれば $x > 0$ に 2 個の零点が存在する．つぎの実行例のように計算すればそ

の近似値が得られる．実行例中の Asir の関数 `fctr(E)` は `E` を因数分解する．また，`pari(roots,P)` は 1 変数多項式 `P` のすべての（実虚含めた）根を（重複を込めて）計算する．

```
[15] fctr(ddf(x));
[[1,1],[exp(-x),1],[x^2-4*x+2,1]]
[16] pari(roots, x^2-4*x+2);
[ 0.5857864376269049511 3.414213562373095048 ]
```

図 6.11 に関してつぎの第 3 問を設けることができる．

問題 8 (第 3 問)
区間を示した上で函数 $f(x)$ のグラフとその 2 階導函数 $f''(x)$ の符号に関して観察するところを述べよ．

最後に，$f(x), f'(x)$ および $f''(x)$ の 3 つのグラフを同じ座標上に重ね描きして図 6.12 に示す．

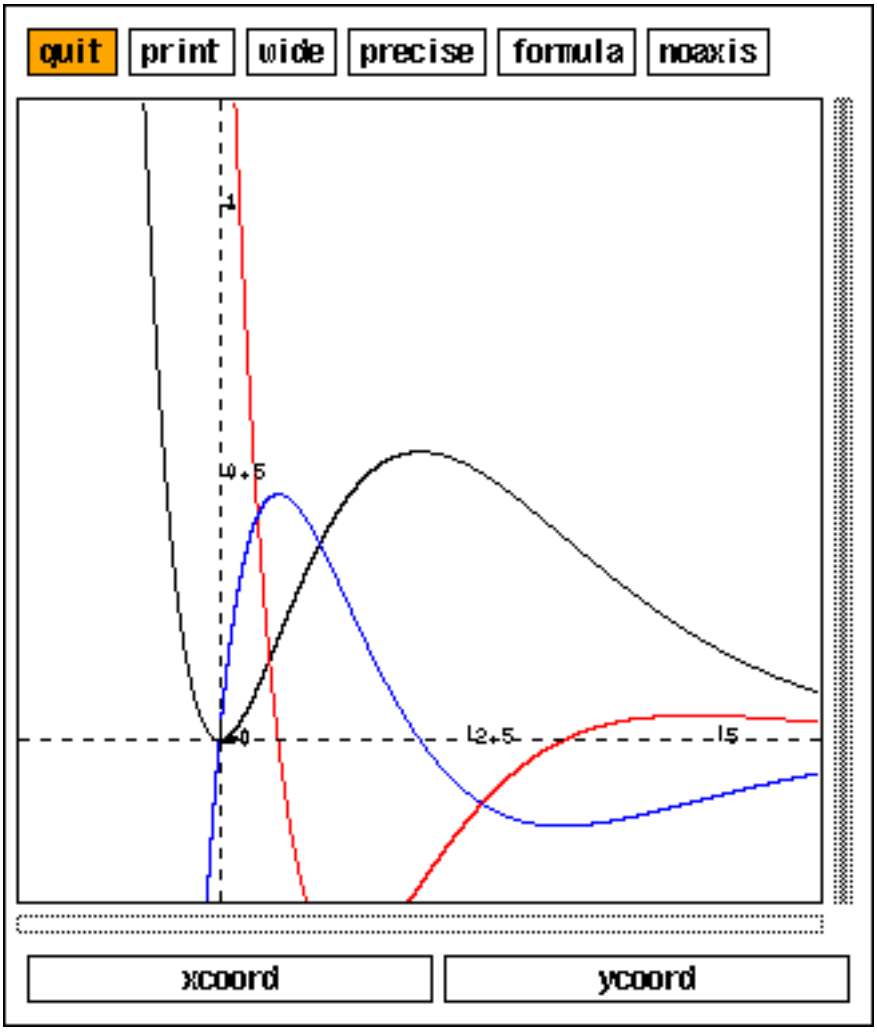


図 6.12: 函数 $f(x)$，導函数 $f'(x)$，2 階導函数 $f''(x)$

ここでつぎの 2 つの質問が提出できる．

問題 9 (第 4 問)
4 つの図，図 6.9 と図 6.10，図 6.11，図 6.12 に基づいて，$f(x)$ のグラフの振る舞いと $f'(x)$ および $f''(x)$ の符号との間の関係について一般的に成り立つ仮説を述べよ．

問題 10 (第 5 問)
幾何学的な観点からつぎのものを求めよ．
（1） $f(x)$ の増加区間/減少区間．
（2） $f(x)$ が凹/凸となっている区間．

（3） 極大/極小，変曲点，x 方向の極限.
数式処理システム（Risa/Asir）を利用して答を検証せよ.

Varbanova が用意した模範解答 [55] を表 6.1 に示す.

表 6.1: 函数のグラフの形状と 1 階導函数，2 階導函数の符号

x	I_0	0	I_1	0.59	I_2	2	I_3	3.4	I_4
$f'(x)$	−	0	+	+	+	0	−	−	−
$f''(x)$	+	+	+	0	−	−	−	0	+
↘/↗	↘	min	↗	↗	↗	max	↘	↘	↘
∪/∩	∪	∪	∪	−	∩	∩	∩	−	∪
shape	◟	∪	◞	／	◜	∩	◝	＼	◟

表中では x の区間の記号としてつぎを用いている: $I_0 = (-\infty, 0), I_1 = (0, 0.59), I_2 = (0.59, 2), I_3 = (2, 3.4), I_4 = (3.4, +\infty)$. ∪ と ∩ はグラフの凹と凸を，◟ , ◞ , ◜ , ◝ はそれぞれ円周を各象限で切り取った形状を象ったものである．また, 2 つの記号 “／” と “＼” とはいずれも変曲点を表している.

これらの演習によって，学生は図 6.10 と図 6.11，図 6.12 に示されたグラフと，表 6.1 中の $\{2, 4\}$ 行や $\{3, 5\}$ 行，$\{4, 5, 6\}$ 行のそれぞれとの間に 1 対 1 の関係が存在することを容易に理解できる.

以上に紹介したように，数式処理システム Risa/Asir は学生達が数学的概念，数学的対象，数学的命題を，様々な視点—記号的，数値的，視覚的—から学ぶことを支援できる.

6.6　日常問題への応用

これまで見てきたように，Risa/Asir は基本的な数学函数とその微分や導函数を使った計算が可能である．このことにより Risa/Asir は，日常生活における数学の役割について，学生達がより多くを学ぶための強力な道具となる．マイクロ経済学（Marginal Analysis）やライフサイエンス，社会科学などにおける微分とその応用を効果的に学ぶことを可能にする．以下に 3 例を示す.

例 3 (Marginal Analysis)
ある日用品 q 単位を生産するための総費用が $C(q) = q^3 + 5q + 162$ であると仮定する．このとき,

1. 単位当りの生産に要する平均費用が最小になるのは生産量がいくらの時か?
2. 単位当りの生産に要する平均費用が限界費用（marginal cost）に等しくなるのは生産量がいくらの時か?
3. 平均費用と限界費用とを $q > 0$ の函数と見て，同じ座標上にグラフをを描け.

解答例 1

1. 生産量が $\sqrt[3]{81} = 3\sqrt[3]{3} = 4.32675$ のとき.
2. 上記（1）の答えに同じ.
3. 図 6.13 に Risa/Asir で描画したグラフを示す．2 つの曲線は $x = 4.3$ 付近で交差し，その点で平均費用が最小になっているとの見当がつく.

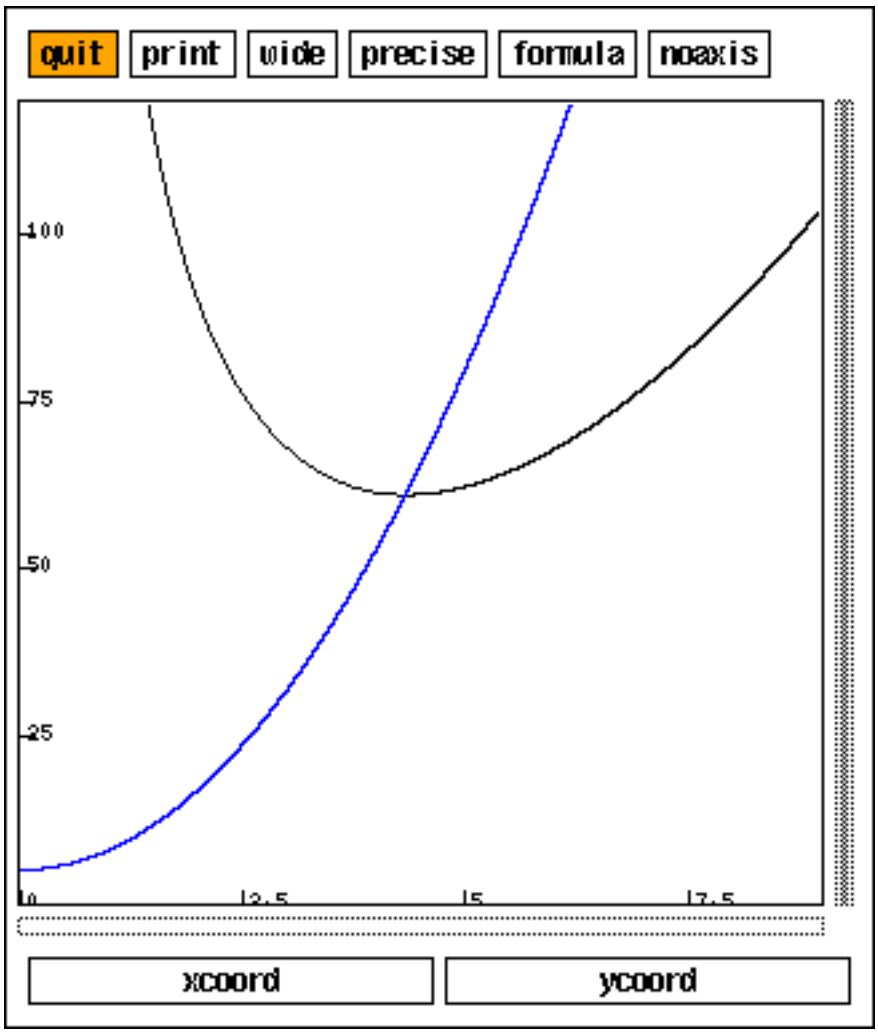


図 6.13: 平均費用（下に凸），および 限界費用（単調増加）

経済学の法則によれば，通常の環境では平均費用 ($\frac{C(q)}{q}$) はそれが限界費用 ($C'(q)$) に等しくなるとき最小になる，とされる．証明は定義どおり数式に表現すれば容易である．すなわちつぎがなりたつ．

$$\left(\frac{C(q)}{q}\right)' = 0 \Leftrightarrow \frac{C(q)}{q} = C'(q).$$

我々はこの例を通してマイクロ経済学の知識と最適化問題とを教えることができる．

例 4 (ライフサイエンス)

ある地域の人口統計の 5 ヶ年予測によれば，今後 t 年後にはその地域の人口は $P(t) = -t^3 + 9t^2 + 48t + 50$ 千人であるという．

1. 人口が最も急速に増加する時は 5 年間の間のいつか?

2. 人口が最も緩慢に増加する時は 5 年間の間のいつか?

解答例 2

1. 人口増加率最大は $t = 3$ で 75 千人/年.

2. 人口増加率最小は $t = 0$ で 48 千人/年.

$P'(t)$ の区間 $0 \leq t \leq 5$ における最大最小問題となる．P' に対して前節において展開した方法を用いればよい.

例 5 (最適設計)

$250m^3$ の体積を持ち正方形の底面をもつ閉じた箱を作りたい．上面と底面とに使用する材料は $1m^2$ 当り 2 ドル，側面に使用する材料は $1m^2$ 当り 1 ドルである．この箱を 300 ドルを越えない費用で製作可能か？

解答例 3

答: コスト条件を満す箱の製作は際どいところで不可能．この箱の製作費用の最小値は底面の一辺が $5m$ 側面の高さが $10m$ の時で，丁度 300 ドルである．このことを考慮すれば，材料の変更や容積の変更で条件を緩和することにより，解が存在するようにできる余地はあることが分かる.

底面の 1 辺を $x\,m$，側面の高さを $y\,m$ と置く．$f_0(x,y) = x^2y - 250$，$f_1(x,y) = 2 \times 2x^2 + 1 \times 4xy$ とおけば，体積の制約 $f_0 = 0$ の下で f_1 を最小にする問題となる．ラグランジュの未定乗数法にしたがって，3 変数多項式 $g(x,y,\lambda) = f_1(x,y) + \lambda \times f_0(x,y)$ の最小値を求めればよい．これには，Risa/Asir の微分機能とグレブナー基底を用いた連立方程式解法が有効に利用できる.

Risa/Asir は 3 次元描画機能を現在もたないが、陰関数描画を利用して等高線を表示することができる．高さ 0 からレベル差 30 で費用 $f_1(x,y)$ が一定レベルとなる等高線を順次描き（図 6.14），これに束縛条件 $f_0(x,y) = 0$ のグラフ（図 6.15）を重ねて描けば，束縛条件の下での最小は，レベルが丁度 300 の等高線と束縛条件の曲線が接する点で生じることが視認できる（図 6.16）.

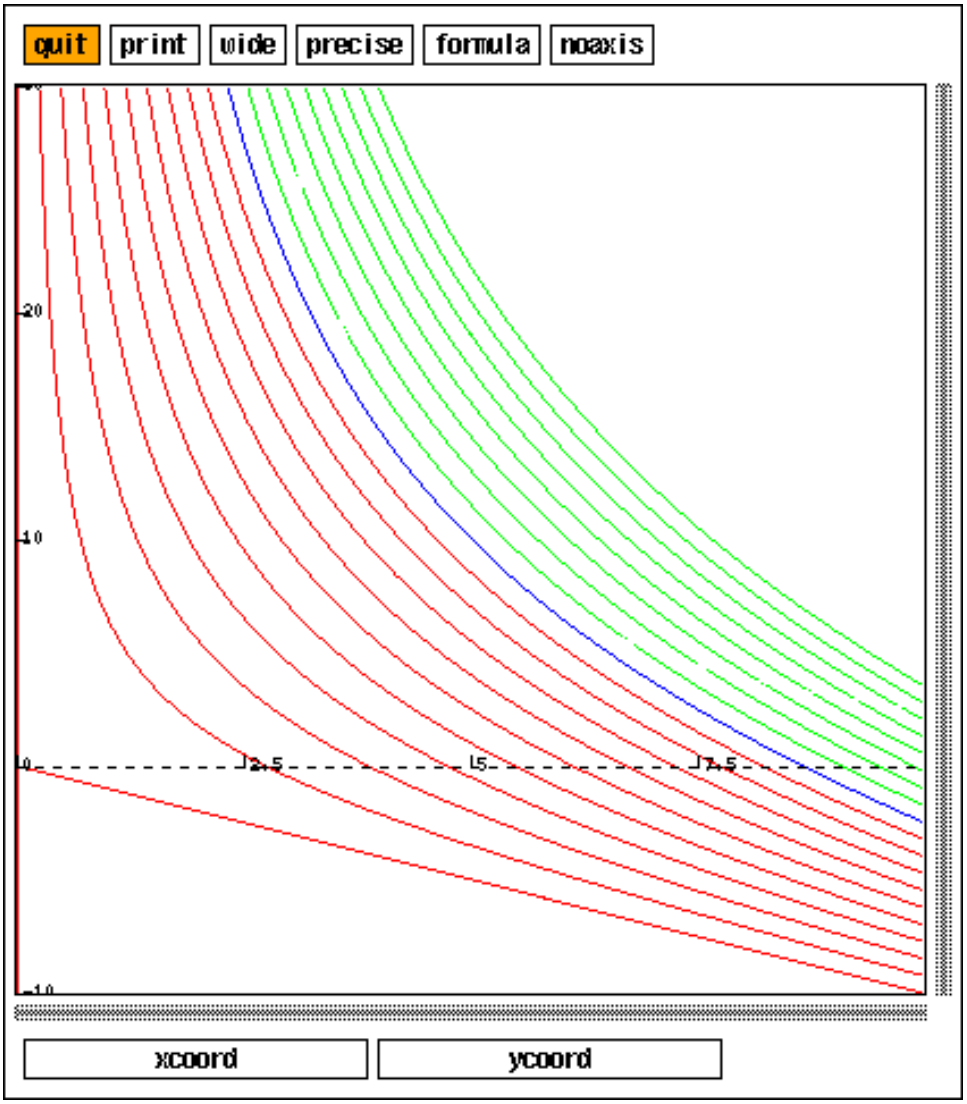


図 6.14: 費用等高線（レベル差 30）

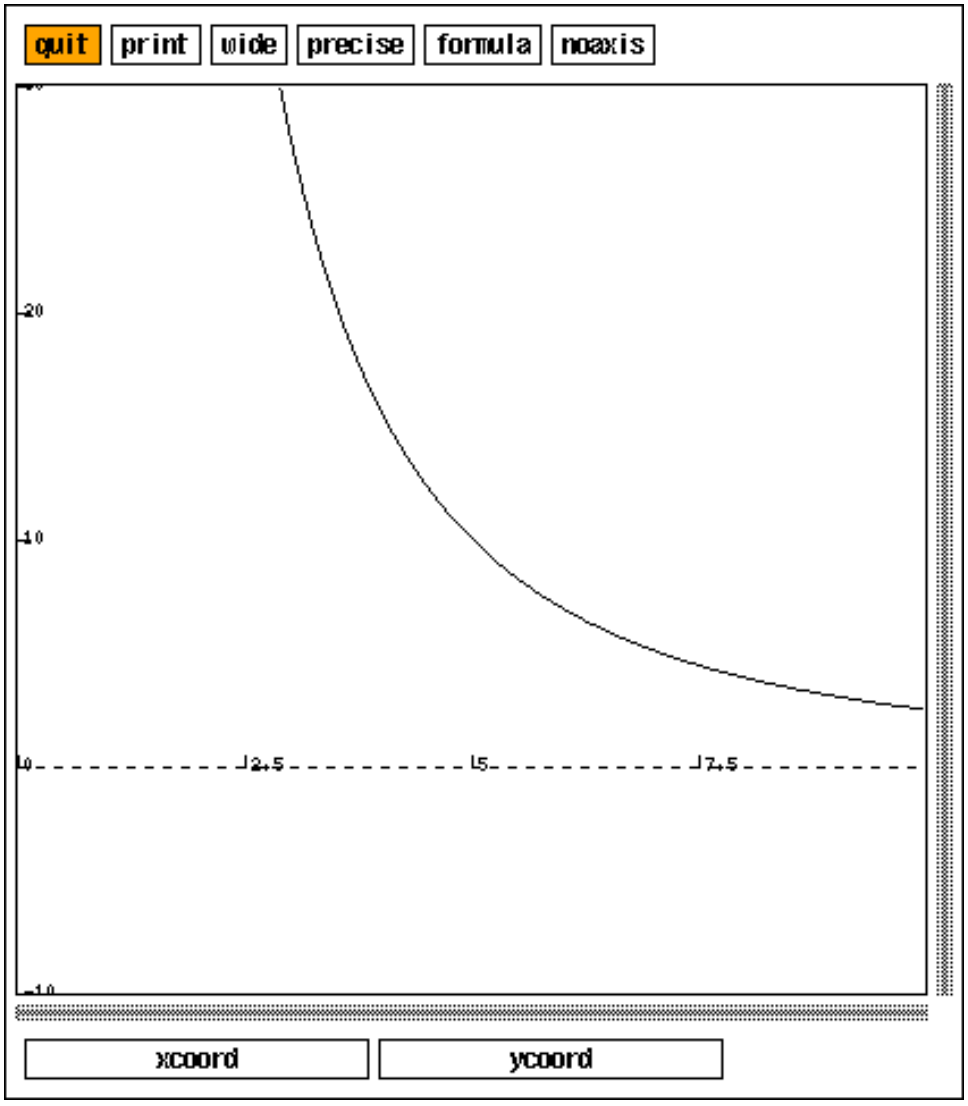


図 6.15: 束縛条件（容積一定）のグラフ

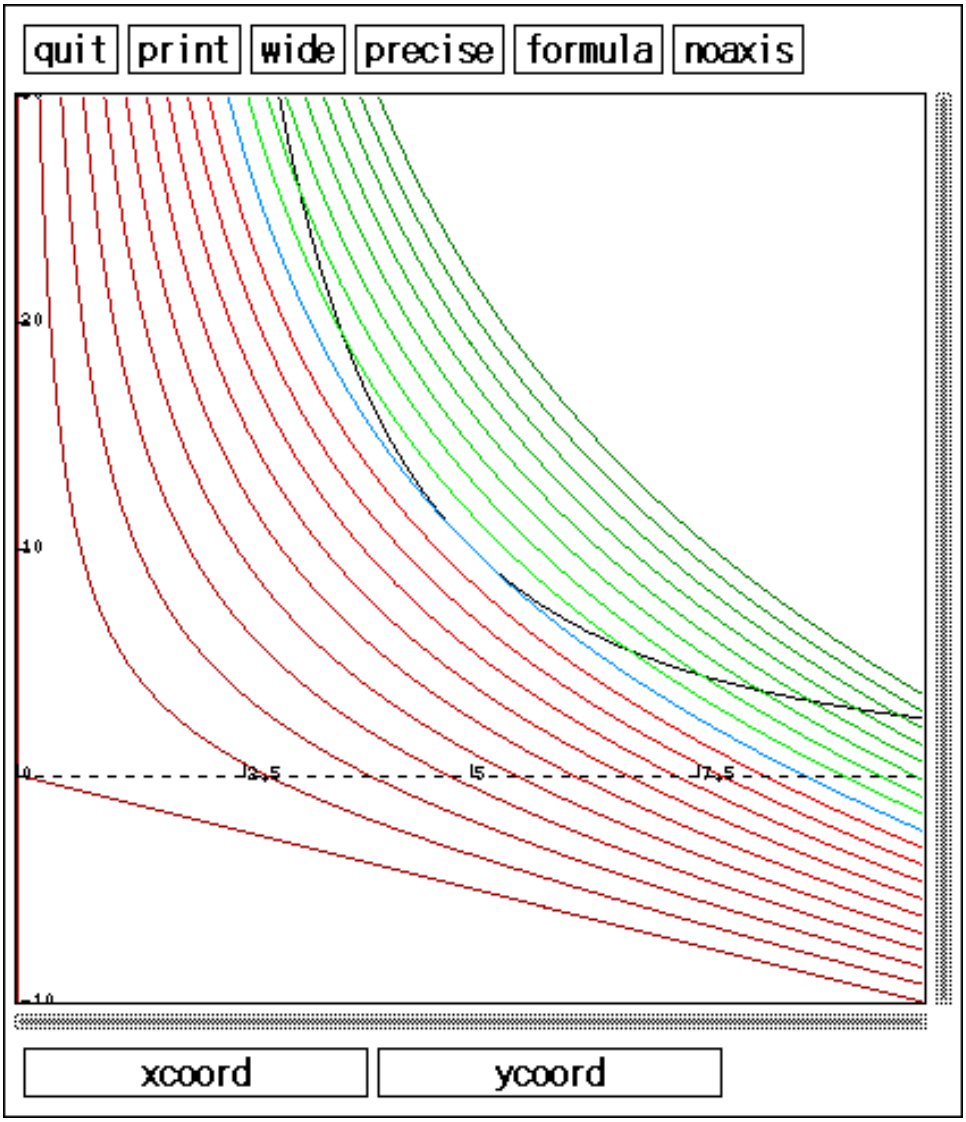


図 6.16: 費用等高線（レベル差 30）および容積一定の曲線の重ね合わせ．横軸：底辺の 1 辺の長さ．縦軸：箱の高さ．

なお，この問題は QE の問題としてよりスマートに定式化することができる．Risa/Asir にとっても QE は今後の開発課題となる．

6.7 考察

本章で検討したトピックは学校で教える数学の中の限られた部分に過ぎない．しかし，検討した範囲で Risa/Asir を適用すれば数学の教育を強化できることを示している．この試みを通じて Risa/Asir に関していくつかの重要な観察がなされた．それは Risa/Asir が数学教育の強力な指導法/支援環境となるために今後の展開が計られなければならないことがらである．一例をあげるならば，数式の簡約機能に見るようないくつかの弱点を機能強化により克服することである．本章で検討したトピックでは、函数ごとにグラフの色を変える機能が必要であったが，それはただちに Risa/Asir にインプリメントされた．このように Risa/Asir では必要な機能が容易にインプリメントできる。

数式の簡約化機能への要求は以前よりもはるかに強くなっている．そのような要求に刺激を受け，項書換えシステムの機能を Risa/Asir に組み込む計画が開始した．既存の数式処理システムによる教育の経験に照らして見ると，Risa/Asir には数学教育における多くの活動を支援する能力があると言える．今後の展開において、このシステムの可用性が数学教師にとっては数学の教え方を近代化する良い挑戦となり，その結果，学生にとっては数学の授業がより面白く魅力的なものとなることが期待できる。

第7章　考察

7.1　数式処理の普及発展と教育の役割

7.1.1　工学系での数学教育

前章での検討例でも見られるように，高等学校から理工学系の大学学部での数学教育においては，数式処理システムの利用は，数値的評価とその視覚化（グラフ化）を中心として実施されている．これは，数学モデルに現われる諸変量を，理工学において取り扱われる物理量（長さ，面積，時間，力学/電磁気学/物性の諸量など）や現実世界において取り扱われる諸量（人口/経済/金融/労働/ロジスティックスやその他社会科学の諸量など）として認識し，現実問題をモデル化する能力を修得する上で中心的で重要な役割を果たしている．

このような現状での主要な使用法においては，数式処理の数式の式のままでの取り扱いは付随する計算の労力を軽減するために使われているにすぎない．実世界での数量に対する感覚を身に付けることが重視されることに異論はないが，数式処理の本来の能力を最大限に活用できる応用/教材の開拓が必要である．

ひとつの例として，関係の導出（消去イデアルの計算）後の誤りのない視覚化（陰函数描画）が挙げられよう．消去イデアル計算による関係の導出についてはグレブナー基底の利用法のひとつとして良く知られた方法であり、ここでは説明は省略する。

数式処理の最大の特徴は，大きな計算量という代償を支払う代わりに，数値誤差の入り込む余地のない「正確」な数値を扱い，「正確で破綻のない[1)]」数値の計算とそれに基づく数式の計算ができる点にある．

たとえば，5.1 節で述べたようにグラフの描画においてさえ，従来の方法では「すでに既知のグラフ」の精度を上げるか，「真」らしいと「信じる」まで試行錯誤を繰り返すしかない．そのため，4.1 節の 2 変数多項式（4.6）や 5.2 節のスペードやクラブの陰函数が与えられても，グラフが未知である場合，それを正確に描画することはほとんど不可能か，あるいは，正しい描画であることを保証することは困難であった．（描画のために特別の解析と特別の手だてを講じれば描画は可能であろうが，与えられる式ごとに個別の作業が必要である．）数値的な計算のみに基づく方法では，「無限

[1)]“exact and infallible.”

の精度で計算できるならば」という現実にはあり得ない仮定の下にアルゴリズムを構成せざるを得ず，多くの分枝や特異点を持ち，次元の異なる解が混在するような関係（陰函数）のグラフの描画には本質的な困難を伴っている．

しかし，数式処理（の考え方）を利用した 5.1 節のアルゴリズムならばアルゴリズムは頑健（robust）で描かれたグラフは無条件に「正しい」ことが保証（infallible）されている．

4.1 節で「高性能計算から高品質計算へ」との伊理の言葉を引用したが，このような「計算の品質」への帰依を広く科学技術の世界に浸透させるには，科学技術の研究/開発/実践の場のみでなく，次代の人材を育成する教育の場においても「計算の品質」を重視し，学習者が「計算の品質」を実感できる教材が扱われてしかるべきである．数式処理における計算は上例のグラフ描画と同様、基本的に exact, robust, infallible である．それゆえ，数式処理は信頼できる「保証付き計算」や「検証用原器」の役割を果たすことができ，そうさせるべきであろう．

しかし,「電卓を使わせると数学的な計算能力や思考力が低下する.」というような，ただ「ブラックボックス」として利用されることの弊害も考慮されねばならない．そのためには，背景となる数学の概念の修得とそれを知識として活用[2)]できるような教育が強く望まれる．

次に，数式処理を活用できる人材を育成するために必要な工学系での代数教育の特徴を論じる．

7.1.2　工学系における代数教育の必要性

計算機代数を初めて学ぶ工学系出身者は，用いられる数学の用語や考え方の違いに一種のカルチャーショックを受ける．20 年間数式処理の研究に携わる間に，その最大の理由が「工学部では代数を教えない」ことにあると確信するに至った．

線形「代数」の授業は，複素数の要素を持つ数値行列と数値ベクトルの扱いに主眼が置かれていて，数式処理では計算上大きな違いとなる加減乗除の四則だけで計算できる概念—連立 1 次方程式解法アルゴリズム—と代数的数の導入が必要な概念—Schmidt の直交化や複素根を扱う固有値，固有ベクトル，Jordan 標準形など—はとくに注意することなく教えられる．整数も有理数も代数的数も虚数も超越数もすべて複素数として計算すればなんの区別も必要ないからである．

しかし数式処理の立場では，殆んどすべての複素数は表現できないし計算もできない[3)]わけであるから，どのような数をどう扱うか（表現/計算の仕方）は基本的問題な

[2)]現実世界の問題を数学の言葉でモデル化し，数学として解き，その答えを再び現実世界の言葉で表現する．

[3)]constructive real という概念があり，超越数を構成的に計算できるようにするという理論があるが，効率を重視する立場の数式処理で実用化されることは当面ないであろう．

のである．

数学的概念としての複素数は，任意の代数方程式の根を含むと言う意味で代数的に（代数関係で）閉じた体系であり，そのことから数学理論の上の見通しを良くするのであるが，数式処理のような構成的計算の立場からは極めて扱いづらいものであり，その違いを明確に認識する必要がある．

極言すれば，数学系出身者と工学系出身者との認識の違いのそもそもの根元は，自然数，整数，有理数，代数的数，超越数などの数が区別される特徴を，群，環，体，商体，拡大体，閉体の理論やそれらに付随する緒性質の概念とともに学んだか，それとも学ばなかったかの違いにあると確信する．

不幸なことに，浮動小数点数という数（？）体系が工学系の数理計算では一般に広く受け入れられており，数値計算の世界では，浮動小数点数値を「real」と呼び、高々 6 桁ないし 16 桁程度の有限桁の整数を「integer」と呼ぶ慣例が定着してしまっている．そして，浮動小数点数値は有理数（有限の桁数しかない）でしかないにも関わらず，それらがあたかも「実数すべて」であるかのような—「浮動小数点数 = 実数」という—錯覚を与えている．このため，数式処理における苦労（理論やアルゴリズムの整備）は多くの人には理解されず，数式処理研究者における特異な「こだわり」に過ぎないとみられている．

しかし，これは頑迷な「こだわり」ではなく，数式処理を利用する人には基本的な数学背景として，ぜひとも理解して欲しい事柄なのである．

代数的複素数に加えて超越的な複素数まで含めることで，極限，微分，積分からテイラー展開，微分方程式論，複素関数論，実関数論と，工学系の数学では解析系の学習内容がほとんどを占めている．そして，先に述べたように「数式処理への無理解と偏見」を持った人材が，工学系の技術者/研究者の圧倒的多数を占めるようになっている．このような現状は数式処理にとってまことに不幸と言わざるを得ない．

この状況を脱して，数式処理の応用を広め，数式処理自身もさらに発展して行くためには，大学の初等教育での解析系の数学（複素数と微積分）への偏重を改める[4)]ことが少なくとも必要であると考える． その教育の方法も抽象的な「代数」を教えるというのではなく，「（構成的）計算」の視点から教える—計算機に機械的に計算させるには「構成的な数学」に基づくことが必要—ことが重要である．

つぎに，どのような概念を学ぶべきか，数学以外に必要な教科も含めて列挙する。

1. 情報処理: プログラミング, 記号処理の初歩
2. 代数学（群，環，体の概念），とくに解析学と数値解析への偏重に対抗した代数学の基礎的概念/考え方の手ほどきが必要．

[4)]「偏重を改める」のであって，「代数」に偏重するべきだというのではない．解析系の数学が現代の理工学で必要とされる数学の主要部分を成すことは否定できない．

3. 代数計算の基礎理論．代数方程式の根が数式処理（代数的処理）ではどう取り扱われているか．

4. 多項式イデアルとその零点集合．イデアル的視点からの代数方程式．

5. 函数と式の違い．ガロア体での多項式の計算（根，微分）が好例．$\mathbb{Z}/2\mathbb{Z}[x]$ における $f_0(x) = 1$ と $f_2(x) = x^2 + x + 1$ の違い．

6. 実数として代数的数を取り扱う原理．通常は実数としての性質を扱うことが困難なこと．

7. 有理数，代数的数，超越数，実数，複素数，有理数の複素拡大 ($\mathbb{Q}(i)$) などの特性と数式処理との関連．計算可能なものは有理数，代数的数まで．効率的に計算でき，実用できるものは有理数．複素数一般は計算できないが，有理数の複素拡大は計算できること．複素数は実数の 2 次の代数拡大であること．

8. 多項式の各項の指数は非負の整数であること．n が記号のとき，$x^n - 1 = (x - 1)(x^{n-1} + \cdots + 1)$ の意味は? 記号の指数を持つ式の取り扱いは簡単ではないこと．

9. 分数冪と根号の計算．

10. 計算が可能な対象：有理数と不定元から四則演算と自然数冪演算とのみを用いて構成される式に限ること．数式処理で扱えるのは基本的には多項式であること．

7.1.3　数値計算との融合

解析系の数学は複素数の近似的代替（浮動小数点数）を用いることによって，理論をシミュレートすることができる．いわゆる数値計算である．

数値計算は，材料/ナノテクノロジー，原子力，宇宙科学/宇宙開発，機械電気製造，化学産業，航空機/造船/自動車/高速鉄道，土木/建築，地震/海洋/資源探査，バイオ産業，気象/天文などの先端テクノロジーのほとんど全ての分野で，設計，製造，解析，シミュレーション，計測/制御に用いられており，その需要は拡大の一途を辿ってきた．一方，数式処理は核物理でノーベル賞受賞に貢献したことや，最近では公開鍵暗号の設計などでの注目されることはあるものの，単発的であり，日常的な企業活動や研究機関の主活動に使用されているとは言い難い．

この理由のひとつには，前節でのべた工学系技術者教育の問題があるが，解析系の問題記述に対しては，現在までの数式処理技術が対応できず，かといって，数値での計算（任意多倍長計算も可能）は数値計算技術に比べて効率上対抗できないことが主たる要因である．

一方，数値計算でもその性質上計算が困難な課題もある．ひとつの例が，非線形非凸最適化問題である．現実の設計問題で非線形非凸であるものは多く，線形近似や凸近似，あるいは凸緩和といった近似の手法を用いて解いているのが現状である．

数式処理の技術のみへのこだわりを捨て，数値計算の長所，quickness や scalability と，数式計算の長所，exactness, robustness, infallibility とをともに活かすことによって，数値と数式の双方の計算で相補って問題を解決する方向をより進める必要がある．久しく言われ続けていた「数値数式融合」である．

この方向での研究活動の中には，近似代数計算と呼ばれる研究分野があるが，QE 手法のひとつ CAD（Cylindrical Algebraic Decomposition）に数値計算を部分的に導入してその効率を改善し，数値的には解くことが困難な最適化問題に適用しようとの研究もある．筆者のグループでは生体反応系のシミュレーションと実験値からの反応パラメータの推定問題に対して，QE と数値シミュレーションとを併用して実施する方式を開発し，その効果を確かめた．この方法はバイオだけでなく，少ない実験データから知見（反応パラメータの推定値など）を得るための方法論として，今後の展開が期待される．これは，見かたによっては数値計算に数式処理的要素を加味したやりかたと言える．

このような方面への応用をはかるとき，前節の列挙の中にも記述したが，実数の数式処理的取り扱いの原理（QE，実代数）は知っておくことが望ましい．

7.2 まとめ

筆者がこの分野に興味を持ち始めたころから 20 年が過ぎた．数式処理は，数学研究のための有用なツールとなったばかりでなく，ビジュアルな GUI とともに，表計算，プレゼンテーションツールなどのデスクトップユーティリティの中で，数値計算やグラフィックス，ビジュアライザー，データベースと互いに連携することにより，教育やエンジニアリングにおいて手軽に使えるツールとなって来ている．このことは IT の目覚ましい発展の典型例として記憶されることになろう．

数式処理のさらに新しい可能性を拓くには，古くから「数値数式融合」というキーワードで指摘されているとおり，「計算の質の確保」と「計算量の壁を越えること」との両立を目指す必要がある．この目的は，正確ではあるが計算量の大きい数式処理に，数値計算の高速性を利用しようという所にある．「数値数式融合」には別の側面もあることが QE の応用を試みている中で見いだされた．それは，情報量不足で数値計算が適用できない最適化問題などに，QE を利用してコンサーバティブではあるが有意の解を与えることができるという例である．

また，教育への応用を拡大するためには，教材を作成する側にとってアプリケーションのプログラムが容易に書けるような，洗練されたユーザインタフェースを持った開

発環境も必要になる．また，与えられた教材を学生/生徒が効果的に利用できるようにするには，柔軟な式変形をサポートすることも重要な要素である．前述の QE が等式ばかりでなく，不等式を含む制約問題や最適化問題の一般的な解法を与えていることを考慮すれば，QE の利用方法を開拓することは教育にとって大きな価値があると考える．

Risa/Asir は，前者（数値数式融合）についてはグレブナ基底や QE を手掛りに発展させることができる．また後者（教育への応用）については，QE パッケージの整備や，Asir 言語とその処理系への機能追加とともに OpenXM のような異種システム連携を進めて行く必要がある．

謝辞

多くの協力者の方々のこれまでのご支援に感謝の言葉を捧げたい．横山，野呂の両氏をはじめとする（元）同僚各氏，Risa Consortium の野田代表，齋藤，高橋の両幹事をはじめとする多くの研究仲間の方々には，Risa の普及ばかりでなく研究上の多くの協力をいただき，励ましと好意的な助言，有用な示唆をいただいた．それらは研究を継続する上で大きな力となった．筆者の不才のため，多くの助言を実現できずにいることをお詫びしたい．

富士通の上司/役員の方々にも感謝を申し述べたい．このような研究の機会を与えられたことは大変幸運なことであった．さらに，企業の中で 20 年にもわたって基礎的な研究を継続することができ，オープンソース化まで許していただいたことは Risa が世に広まり発展した最大の要因であった．富士通は日本の数式処理の発展に大きな役割を果たした．その事実はこの分野の人々に正しく評価されていると自負している．研究の初期に励まして下さった初代国際情報社会科学研究所 北川所長と第 3 代 榎本所長[5]，ICOT から離れ独自に研究することを認めて下さった小口文一所長，Risa の公開に尽力して下さった佐藤所長[6] と戸田所長[7]の理解と支援の賜である．

最後にもう一度，Risa に関わったすべての関係者に感謝する．

[5]榎本 肇，東京工業大学名誉教授.
[6]佐藤 繁，元富士通研究所 社長
[7]戸田 光彦，現 新潟大学教授

付 録A　4.1.5項の多項式 $g(y,z)$

$$
\begin{aligned}
g(y,z) = (&170550761728z^7 - 102021120z^6 + 85287168z^5 - 38631168z^4 \\
&+10285056z^3 - 1617408z^2 + 139968z - 5184)y^{14} \\
&+(426622464z^8 - 1249385472z^7 + 1530814464z^6 - 1021579776z^5 \\
&+407576448z^4 - 100113408z^3 + 14883264z^2 - 1233792z + 44064)y^{13} \\
&+(1345932288z^9 - 5972216832z^8 + 10507428864z^7 - 9976296960z^6 \\
&+5708454912z^5 - 2058400512z^4 + 471090816z^3 - 66462336z^2 \\
&+5286816z - 182304)y^{12} \\
&+(1743441408z^{10} - 15108083712z^9 + 39147369984z^8 - 51536715264z^7 \\
&+40568822784z^6 - 20396448000z^5 + 6705011520z^4 - 1433652480z^3 \\
&+192095712z^2 - 14670720z + 489024)y^{11} \\
&+(-107495424z^{11} - 23149089792z^{10} + 92601211392z^9 - 164463644160z^8 \\
&+170576948736z^7 - 113777872128z^6 + 50731655616z^5 - 15258106944z^4 \\
&+3051375840z^3 - 388455264z^2 + 28498176z - 919296)y^{10} \\
&+(-2877742080z^{12} - 23990639616z^{11} + 149784857856z^{10} - 345334399488z^9 \\
&+453036094848z^8 - 383863214592z^7 + 221262057216z^6 - 88437432960z^5 \\
&+24471984384z^4 - 4590285120z^3 + 556037136z^2 - 39225600z + 1225800)y^9 \\
&+(-2906855424z^{13} - 18562618368z^{12} + 165484353024z^{11} - 478005898752z^{10} \\
&+771480070272z^9 - 806220687744z^8 + 580445269056z^7 - 295759632960z^6 \\
&+107400375360z^5 - 27557406144z^4 + 4869472608z^3 - 562607424z^2 \\
&+38217960z - 1158120)y^8 \\
&+(-423263232z^{14} - 9383205888z^{13} + 111547901952z^{12} - 411136280064z^{11} \\
&+820886762880z^{10} - 1051230825216z^9 + 930249183744z^8 - 589384332288z^7 \\
&+271260129312z^6 - 90709519680z^5 + 21761912016z^4 - 3640417056z^3 \\
&+402289416z^2 - 26355168z + 775152)y^7 \\
&+(976416768z^{15} + 1037380608z^{14} + 25252964352z^{13} - 179269770240z^{12} \\
&+498297495168z^{11} - 815899692672z^{10} + 897784119360z^9 - 703570829760z^8 \\
&+403312495680z^7 - 170705200896z^6 + 53191614384z^5 - 12023215344z^4 \\
&+1913168160z^3 - 202777200z^2 + 12831840z - 366624)y^6
\end{aligned}
$$

$$
\begin{aligned}
&+(540089856z^{16}+6169540608z^{15}-22891113216z^{14}+2703310848z^{13}\\
&+123573423744z^{12}-333786292992z^{11}+492508014336z^{10}-488970912384z^{9}\\
&+349543885248z^{8}-184852354176z^{7}+72919712160z^{6}-21369417408z^{5}\\
&+4579931520z^{4}-696132288z^{3}+70953228z^{2}-4343544z+120654)y^{5}\\
&+(20155392z^{17}+4138573824z^{16}-19292442624z^{15}+35388264960z^{14}\\
&-19414774656z^{13}-47681343360z^{12}+134709822144z^{11}-183489064128z^{10}\\
&+167335699392z^{9}-110705591808z^{8}+54645444720z^{7}-20277692496z^{6}\\
&+5629127400z^{5}-1150102872z^{4}+167626620z^{3}-16472340z^{2}+977022z\\
&-26406)y^{4}\\
&+(-30233088z^{18}+969698304z^{17}-4484947968z^{16}+9706438656z^{15}\\
&-10946109312z^{14}+2116171008z^{13}+15682517568z^{12}-32761580544z^{11}\\
&+38863922688z^{10}-32302205568z^{9}+19899646992z^{8}-9258708384z^{7}\\
&+3263915304z^{6}-865686240z^{5}+169773768z^{4}-23849304z^{3}\\
&+2267428z^{2}-130568z+3436)y^{3}\\
&+(17915904z^{18}-3919104z^{17}-253248768z^{16}+867065472z^{15}\\
&-1612151424z^{14}+2239057728z^{13}-2729467584z^{12}+3012072480z^{11}\\
&-2843134560z^{10}+2164382928z^{9}-1286122320z^{8}+587255400z^{7}\\
&-203959080z^{6}+53244072z^{5}-10253376z^{4}+1410984z^{3}-131148z^{2}\\
&+7371z-189)y^{2}\\
&+(-1026432z^{16}+8957952z^{15}-40746240z^{14}+117348480z^{13}\\
&-220595616z^{12}+279410688z^{11}-247641408z^{10}+158813568z^{9}\\
&-75551688z^{8}+27102240z^{7}-7396056z^{6}+1537584z^{5}-241654z^{4}\\
&+28096z^{3}-2304z^{2}+120z-3)y\\
&+46656z^{15}-419904z^{14}+1578528z^{13}-3256416z^{12}+4111776z^{11}\\
&-3392928z^{10}+1917864z^{9}-765720z^{8}+219240z^{7}-45000z^{6}+6498z^{5}\\
&-630z^{4}+37z^{3}-z^{2}.
\end{aligned}
$$

付 録B　スペードの多項式

Spade(x, y) =

$$
\begin{array}{rrrrrrrrrrl}
 26975 & 06385 & 66396 & 58800 & 96087 & 54799 & 73009 & 21889 & 59880 & 07536 & \\
 72663 & 78947 & 71077 & 41418 & 59273 & 19935 & 43227 & 61506 & 26408 & 12543 & \\
 27599 & 07791 & 98177 & 28000 & & & & & & & x^8 \\
-42345 & 45193 & 08939 & 92624 & 70505 & 53991 & 80686 & 32176 & 58604 & 94542 & \\
 17791 & 17383 & 38704 & 17288 & 32976 & 52115 & 97693 & 80673 & 26198 & 18945 & \\
 31728 & 01536 & 00000 & 00000 & 0 & & & & & & x^6y^2 \\
-33566 & 67568 & 16833 & 15763 & 89085 & 02560 & 34053 & 75438 & 10130 & 03838 & \\
 81912 & 93775 & 95845 & 00537 & 26224 & 79445 & 04172 & 95498 & 69509 & 50632 & \\
 01550 & 26141 & 84014 & 64320 & 0 & & & & & & x^6y \\
+18866 & 26062 & 51815 & 03291 & 95854 & 70192 & 28578 & 53634 & 68451 & 02367 & \\
 27669 & 74853 & 67516 & 10955 & 93763 & 48639 & 25892 & 34712 & 84561 & 35390 & \\
 59407 & 56713 & 76581 & 75488 & 0 & & & & & & x^6 \\
+27429 & 46917 & 57877 & 85740 & 60573 & 67364 & 83794 & 92573 & 13071 & 01090 & \\
 34922 & 05291 & 16889 & 01241 & 18956 & 67926 & 80570 & 55013 & 40722 & 14889 & \\
 80076 & 92083 & 20000 & 00000 & 00 & & & & & & x^4y^4 \\
-13384 & 20163 & 66273 & 12445 & 81499 & 23923 & 58532 & 53860 & 42668 & 97577 & \\
 35939 & 20944 & 50591 & 68871 & 56435 & 75714 & 61594 & 98501 & 20299 & 98782 & \\
 66275 & 26811 & 36405 & 21728 & 00 & & & & & & x^4y^3 \\
-19246 & 00176 & 26073 & 23504 & 42299 & 24170 & 97056 & 49294 & 51180 & 42898 & \\
 38484 & 86064 & 68075 & 28493 & 96334 & 87619 & 43000 & 66909 & 43244 & 97865 & \\
 98973 & 07200 & 62987 & 75552 & 000 & & & & & & x^4y^2 \\
-25088 & 55699 & 25613 & 04872 & 36871 & 58590 & 11725 & 18351 & 21605 & 16566 & \\
 15044 & 85099 & 08705 & 80122 & 18400 & 21077 & 97829 & 07187 & 00401 & 59333 & \\
 96253 & 64457 & 19178 & 79402 & 496 & & & & & & x^4y \\
-10110 & 02089 & 46326 & 93419 & 31967 & 57599 & 45329 & 03089 & 07784 & 62366 & \\
 20811 & 82613 & 67121 & 91068 & 77926 & 62835 & 43154 & 35995 & 83390 & 90273 & \\
 51865 & 59330 & 91671 & 46629 & 632 & & & & & & x^4 \\
-39011 & 26541 & 93891 & 72818 & 65044 & 20798 & 98550 & 83991 & 86159 & 82296 & \\
 96169 & 42535 & 69361 & 64650 & 30206 & 77690 & 67378 & 73547 & 47374 & 86094 & \\
 34275 & 31571 & 20000 & 00000 & 00 & & & & & & x^2y^6 \\
+48902 & 16392 & 06469 & 90802 & 82092 & 03606 & 84842 & 66441 & 48870 & 91066 & \\
 23168 & 65311 & 31335 & 68367 & 12779 & 18100 & 98033 & 78623 & 18789 & 26143 & \\
 41352 & 49602 & 54884 & 23321 & 600 & & & & & & x^2y^5
\end{array}
$$

（次ページに続く.）

+24384 34008 86721 36865 33940 89817 98597 88795 92324 82869
40459 69821 03399 08611 00417 89117 65518 52114 28354 27504
23539 28349 19751 65091 8400 x^2y^4
+46202 34306 42089 73460 46072 73678 03439 90386 02396 68969
57527 60742 90110 22721 48764 55742 18165 57551 06203 73496
80977 28298 55025 50153 7280 x^2y^3
+47389 13158 72825 64272 57835 65168 99700 75227 65480 01357
82557 79309 67888 66382 68621 14970 63717 45099 77702 80524
40262 42089 18086 36791 4240 x^2y^2
+26562 24965 99595 01780 22518 09545 38436 56799 61430 21218
33808 84366 51170 88672 01849 19427 01216 50616 39284 49245
45434 56886 64113 38335 2416 x^2y
+62967 33382 74437 15064 41613 79479 21700 02491 23329 76466
06382 51543 65387 46101 57741 25037 83010 52331 63548 08728
12630 72221 49574 81475 472 x^2
+12888 63184 58470 04778 23923 06455 54859 90255 93578 02913
60184 89204 79639 52483 09591 42971 63171 33552 81334 88684
91567 57447 68000 00000 000 y^8
−41784 44304 48173 55458 05421 21912 03634 26553 28622 15571
18358 73243 01100 38598 49278 46131 02076 43998 31945 89235
38846 20960 00000 00000 00 y^7
−90463 29260 60988 98907 63582 30496 03271 86407 34593 97684
06648 33957 99938 32461 90955 07126 79941 30166 91981 86526
56788 36802 47600 00000 000 y^6
−17640 01017 22928 84184 82147 57205 87130 26251 70733 93064
78523 97097 86651 90574 77537 44607 63417 07125 39106 48574
24881 51833 77400 00000 000 y^5
+22459 65063 02896 53837 48889 05573 30015 18658 65184 68143
11240 25128 13779 27597 77376 68001 49196 01430 23836 68110
61078 83742 20358 62500 0000 y^4
+15015 82735 96200 34598 62865 79083 32817 63933 63991 37896
56095 52916 06990 21642 81588 44465 03088 29757 30146 63017
65520 78887 47861 34375 0000 y^3
−16856 07467 27439 65408 91130 47709 79488 15187 72534 20148
56144 19921 24575 81794 19597 12723 93147 45980 01222 39658
66902 47154 24664 13281 2500 y^2
−21115 85110 11820 24109 68282 77441 43267 12892 79197 63497
76858 09912 27404 54916 60169 75359 26605 93541 50240 89519
70360 48858 10941 68554 6875 y
−61288 52507 15364 33484 07125 03524 60073 63053 67086 70542
80996 71679 11605 20142 11177 18774 43422 24416 16678 95480
83577 07242 60456 81640 625.

付 録C　クラブの多項式

$\mathrm{Club}(x, y) =$

$$
\begin{array}{rrrrrrrrrrl}
 68049 & 29736 & 01098 & 24914 & 90352 & 87585 & 35653 & 87905 & 95847 & 28738 & \\
 77779 & 60750 & 98966 & 80797 & 00322 & 85499 & 21429 & 28157 & 34749 & 13021 & \\
 20516 & 81322 & 98557 & 43409 & 50498 & 38342 & 23012 & 94819 & 31516 & 28338 & \\
 07545 & 49267 & 52940 & 63212 & 62047 & 49153 & 09285 & 28669 & 38576 & 27431 & \\
 03926 & 30234 & 52223 & 94588 & 89017 & 71094 & 39034 & 89720 & 32000 & 0000 & x^{10} \\
+38573 & 59629 & 10616 & 10388 & 30855 & 66017 & 25831 & 67200 & 98029 & 71798 & \\
 05405 & 95571 & 41712 & 45390 & 59543 & 02500 & 36456 & 73697 & 35286 & 46910 & \\
 70567 & 76721 & 29569 & 22003 & 31395 & 40190 & 97584 & 01717 & 65992 & 40927 & \\
 24085 & 97533 & 55573 & 51260 & 95145 & 32436 & 57922 & 13783 & 88407 & 99618 & \\
 31358 & 39678 & 07829 & 07682 & 03904 & 00423 & 08264 & 89282 & 56000 & 00000 & x^8y^2 \\
-15832 & 32267 & 43895 & 19396 & 78583 & 02293 & 13115 & 62271 & 78673 & 06176 & \\
 09906 & 22466 & 40101 & 58923 & 77074 & 69639 & 60911 & 55449 & 86882 & 54962 & \\
 07904 & 47566 & 12279 & 15005 & 80902 & 16901 & 25343 & 44997 & 08540 & 78070 & \\
 54508 & 49815 & 29824 & 52674 & 84003 & 92583 & 71460 & 50853 & 54958 & 31458 & \\
 67863 & 68021 & 72333 & 01200 & 30407 & 06611 & 82124 & 19793 & 71520 & 0000 & x^8y \\
-19549 & 83611 & 72763 & 31323 & 54724 & 28367 & 52333 & 28966 & 25471 & 06830 & \\
 78445 & 03735 & 40362 & 89471 & 39886 & 74829 & 52199 & 67890 & 04894 & 08422 & \\
 85858 & 73028 & 54269 & 19168 & 66129 & 60837 & 32643 & 82397 & 43804 & 45993 & \\
 47900 & 31106 & 15109 & 01877 & 05745 & 18049 & 85194 & 56079 & 77218 & 10304 & \\
 26069 & 33227 & 83558 & 95774 & 83115 & 79616 & 30232 & 50043 & 08683 & 16160 & \\
 00 & /27 & & & & & & & & & x^8 \\
+10102 & 18872 & 90276 & 75183 & 04811 & 34213 & 69309 & 42449 & 32759 & 61751 & \\
 19234 & 69426 & 36856 & 19365 & 27025 & 47227 & 81967 & 31557 & 94349 & 32449 & \\
 48725 & 00401 & 20245 & 63425 & 59455 & 02093 & 59033 & 83933 & 63150 & 02910 & \\
 13224 & 44349 & 87391 & 37145 & 79154 & 47246 & 19183 & 03614 & 36960 & 37926 & \\
 50247 & 74647 & 99467 & 32587 & 76349 & 80151 & 70785 & 71646 & 97600 & 00000 & \\
 0 & & & & & & & & & & x^6y^4 \\
+37598 & 67310 & 19330 & 87851 & 70436 & 38811 & 22888 & 64833 & 62865 & 43706 & \\
 14971 & 72021 & 01883 & 94484 & 20130 & 60373 & 63432 & 69762 & 98576 & 65486 & \\
 44132 & 75449 & 98738 & 65587 & 37587 & 82005 & 54582 & 97678 & 90822 & 69179 & \\
 74849 & 61970 & 10733 & 80591 & 16691 & 85976 & 95646 & 59137 & 03926 & 90035 & \\
 86314 & 44575 & 97589 & 73628 & 89090 & 06092 & 12019 & 34537 & 53630 & 72000 & \\
 0 & & & & & & & & & & x^6y^3
\end{array}
$$

（次ページに続く.）

+29450 00666 59929 11621 92857 84710 18730 41641 74687 43879
35217 00763 37353 17715 70470 16014 31793 13218 43323 48308
56748 73502 42614 33072 38182 30629 20378 93622 81523 41725
73936 46418 78079 50776 41234 69903 05906 46925 44346 16767
67226 43552 43369 90041 34410 17573 59456 49920 78079 59040
00 /3 x^6y^2

+10703 84430 76734 62497 98674 03066 85107 78414 53392 60296
64817 39926 23126 46618 82284 56225 67675 37417 43820 54903
24392 64639 26892 72151 66141 54418 53315 57367 18792 28174
84869 38605 46247 82555 47012 06025 28899 42995 16115 19340
30778 72905 06194 26334 58099 03933 45862 31820 58866 82809
95840 /729 x^6y

+20110 71727 82162 15770 19352 62818 65812 09113 96831 35124
97623 76699 86957 81616 70021 31129 02078 72304 04404 97289
00828 66175 06452 09094 43854 14617 95051 54458 19943 35776
79080 73032 98966 24158 14769 55169 91214 77451 51172 65449
49408 39360 01257 59790 11695 00470 17032 24228 75852 72032
4544 /243 x^6

+17313 60182 79807 84666 65300 69802 87828 05257 23330 87966
16590 85350 22838 32285 44568 83883 60068 21756 98924 72925
73952 86562 84624 79092 99531 66423 63547 45101 95042 35941
97226 29987 15122 20680 22041 47073 14829 82356 01336 44218
33344 54468 89894 43077 31830 24840 41087 01597 69600 00000
0 x^4y^6

+11943 89234 36504 75645 37934 56204 34604 37570 40261 91615
13664 05172 64929 05115 52589 41899 13596 41882 16369 97338
06907 24443 39486 16469 45646 62926 49591 48664 86497 06909
46967 58631 10791 22917 70061 70922 24421 53065 99440 52860
56128 78481 62347 99680 56237 69862 32762 02654 54862 33600
00 x^4y^5

+38601 33248 85727 70192 21373 02154 39699 86205 05350 93324
55492 32930 73498 60586 24669 82002 72818 27592 66778 53159
74896 23154 71376 59449 02483 60434 85702 41816 33842 30259
04192 41025 40101 94554 67915 91229 08723 27309 24412 51879
16991 07195 99986 33417 17731 46649 91132 43641 13675 55072
00 x^4y^4

（次ページに続く.）

$$
\begin{aligned}
&+13708\ 55810\ 84562\ 23226\ 10486\ 15430\ 74684\ 85601\ 68780\ 48062 \\
&\ \ 40864\ 05225\ 52557\ 17927\ 43603\ 31429\ 78303\ 78571\ 07228\ 09137 \\
&\ \ 16485\ 90632\ 48543\ 24594\ 40872\ 47237\ 08802\ 60245\ 79283\ 98691 \\
&\ \ 38825\ 40153\ 86837\ 62763\ 22325\ 40847\ 18195\ 78995\ 04233\ 61666 \\
&\ \ 54642\ 14753\ 07347\ 36451\ 13261\ 78857\ 38314\ 54290\ 93714\ 47666 \\
&\ \ 2784\ /27 && && && && && && x^4y^3 \\
&+66340\ 42464\ 92026\ 66014\ 34377\ 20173\ 67619\ 19720\ 25881\ 50064 \\
&\ \ 23625\ 90291\ 14412\ 69067\ 10512\ 61215\ 50976\ 55332\ 22351\ 91102 \\
&\ \ 60514\ 00755\ 36902\ 87614\ 81283\ 54907\ 59472\ 19554\ 51610\ 09219 \\
&\ \ 69526\ 32910\ 48554\ 52813\ 47892\ 38459\ 30936\ 86207\ 63272\ 83229 \\
&\ \ 40985\ 52578\ 60373\ 34237\ 41457\ 72486\ 47334\ 73725\ 63740\ 14897 \\
&\ \ 152\ /81 && && && && && && x^4y^2 \\
&-89190\ 54951\ 20375\ 05521\ 36523\ 29562\ 35085\ 35198\ 53049\ 49014 \\
&\ \ 85399\ 48655\ 91753\ 43018\ 54066\ 89126\ 09886\ 13868\ 85100\ 22574 \\
&\ \ 76509\ 56615\ 06779\ 63943\ 99934\ 06494\ 75853\ 69386\ 36631\ 88512 \\
&\ \ 68988\ 11427\ 26283\ 79967\ 08246\ 50170\ 18307\ 42514\ 24727\ 86612 \\
&\ \ 59103\ 53671\ 70449\ 83947\ 79394\ 24082\ 80649\ 52104\ 05518\ 72149 \\
&\ \ 504\ /27 && && && && && && x^4y \\
&-19521\ 46279\ 29343\ 82085\ 75137\ 69373\ 01880\ 51301\ 82839\ 03947 \\
&\ \ 46275\ 20845\ 23466\ 05102\ 95903\ 03731\ 22262\ 24959\ 79305\ 35796 \\
&\ \ 81063\ 47356\ 57511\ 20325\ 67930\ 18615\ 65500\ 24923\ 67956\ 68092 \\
&\ \ 10104\ 31599\ 66754\ 20806\ 84441\ 75776\ 65813\ 82483\ 69750\ 25625 \\
&\ \ 71578\ 64983\ 39692\ 38203\ 61063\ 69149\ 68767\ 94023\ 20678\ 29105 \\
&\ \ 28 && && && && && && x^4 \\
&+16831\ 78726\ 65965\ 43228\ 13188\ 92697\ 47257\ 48650\ 43430\ 77971 \\
&\ \ 77092\ 84665\ 22045\ 08142\ 86491\ 42921\ 03022\ 73058\ 09448\ 65460 \\
&\ \ 49990\ 06854\ 54628\ 17987\ 26540\ 82036\ 49368\ 96750\ 25557\ 07969 \\
&\ \ 04457\ 80680\ 50515\ 79952\ 79985\ 66981\ 17894\ 56615\ 68002\ 81052 \\
&\ \ 86790\ 67655\ 04672\ 27597\ 81247\ 16507\ 00464\ 44650\ 49600\ 00000 \\
&\ \ 0 && && && && && && x^2y^8 \\
&+13878\ 87361\ 81347\ 50738\ 83379\ 67340\ 78892\ 79020\ 22649\ 77594 \\
&\ \ 10160\ 27989\ 38720\ 37248\ 16972\ 94824\ 12227\ 92659\ 73340\ 51956 \\
&\ \ 99629\ 42892\ 40168\ 96015\ 81320\ 70963\ 61237\ 04292\ 48646\ 99771 \\
&\ \ 32412\ 78112\ 82990\ 31631\ 48216\ 05906\ 46621\ 62524\ 82674\ 76102 \\
&\ \ 02016\ 13360\ 87108\ 79902\ 20245\ 12169\ 54929\ 81527\ 59099\ 39200 \\
&\ \ 00 && && && && && && x^2y^7
\end{aligned}
$$

（次ページに続く.）

+47119 97171 48860 32298 07879 27313 72175 75244 14895 99336
74146 75596 55045 89910 13857 91951 38497 11502 68766 95622
94511 90092 04684 36140 09758 58449 58677 51910 57366 99790
11776 50226 08123 93315 00090 66219 83353 91557 39441 39766
46846 89643 42864 80857 45644 98611 54370 89370 41404 23168
00 x^2y^6

+28217 37893 24532 56623 17209 46021 66757 41675 49206 09497
87358 64488 72499 70173 66852 59130 37359 28201 23849 02019
94528 05886 76572 13391 38514 25765 84456 57323 10052 80333
56609 00132 44018 85365 81914 14664 53475 13111 10607 00828
60089 09410 94355 41327 60063 50499 08856 96815 74707 97594
624 /3 x^2y^5

+12544 58899 27522 71319 35635 47044 59370 10178 15474 84286
72858 12944 39520 56612 02692 41796 97062 13827 53359 72377
32156 34374 25785 25789 90499 91414 63792 24762 13930 48853
08830 17512 11952 15159 01963 53051 89147 88132 26770 62088
71522 57586 64804 19180 29043 65494 78005 50122 30982 70888
7552 /9 x^2y^4

+14646 12923 85237 33260 71242 26020 42934 46630 59926 70741
20769 89929 83847 25486 90256 47177 45739 90505 01354 98253
87654 63290 16232 60568 64980 11765 19762 07901 85065 73280
32254 98111 93006 12026 18636 16680 35820 38328 78029 49229
39029 29061 96821 43937 33536 79227 16709 84773 37195 86978
2016 /9 x^2y^3

+11414 85354 13472 75889 84535 15543 01620 17292 34448 42614
13765 68375 58497 87798 00093 71278 06080 78845 04892 37579
31986 79962 15385 10573 81079 66977 33752 30304 36545 28406
80507 00089 34582 05563 11877 25629 38990 10986 75888 49093
57602 51844 16590 86438 83130 88653 92294 84081 48934 62264
3200 /9 x^2y^2

+52501 87860 84529 68817 52596 34944 55778 48354 45062 07312
72466 96693 12448 82321 75922 93179 20324 62846 09148 24052
33797 46498 20711 18462 88789 51712 71762 10986 55803 92757
24822 84909 07226 04758 16624 61372 35202 23240 72122 45133
13019 45172 28397 83694 09425 98856 48986 60638 53799 04194
56 x^2y

（次ページに続く.）

+84782 53361 90405 37141 11719 01238 99118 17309 46133 29586
64464 12129 40444 17206 42229 61869 82955 35845 04026 62361
15002 09085 33442 20736 43290 58100 83338 92942 83064 12895
88413 19484 75763 09820 65118 19872 31646 09217 07738 01522
32695 19125 60723 06680 73794 54760 49196 78380 70346 66419
2 x^2
+65419 41471 69356 67998 53573 62358 14241 44725 81091 61139
87413 57460 55189 47690 57171 41911 83070 17980 49935 16546
09874 25848 74881 19844 98136 80285 00099 97818 68772 09957
95622 80424 05822 74124 42225 61408 52047 71660 63974 43731
57051 92681 00117 99873 63726 01132 95499 88528 12800 00000 y^{10}
+61059 73771 36808 29664 29838 78531 37554 51691 53904 71752
05131 61002 25362 23671 85355 32300 33517 41937 97720 49987
84588 24269 60685 21108 26410 33133 71259 14711 12664 99855
43065 86124 65254 34028 43889 05689 60296 13143 86528 49102
38191 67261 13166 45578 41056 68316 10171 25701 68115 20000
0 y^9
+22348 48204 57602 12009 45370 00578 14289 15929 42432 09093
93728 08228 74946 10902 65228 27356 22546 93152 98584 75412
76624 26306 22462 16940 65306 44509 40219 86736 30242 39028
03773 21162 30586 40470 58871 58769 42018 88741 11113 85388
65160 48556 99730 70672 59230 58474 75144 85475 68333 61920
00 y^8
+36887 50615 84440 60594 83316 88933 39057 49519 80083 87916
66744 00817 46178 07099 80440 98966 76092 47539 34331 21344
46228 07566 41217 29786 31612 04924 89734 00424 51346 10191
18775 45247 99241 73061 10162 25012 58164 64248 64923 10984
76321 09506 50102 85583 48961 53764 34032 12496 35860 48000
00 y^7
+11156 00001 90072 97536 75347 16642 34974 76959 21975 28834
91280 27251 47724 39499 23901 99907 90902 74397 32320 02320
91302 03192 01315 75910 36098 45406 80899 45878 79809 00862
93093 88977 90228 89764 48069 01325 46434 62589 32024 26832
23181 20704 83819 10923 97175 68373 55920 01326 89324 67097
60 y^6

（次ページに続く.）

$$
\begin{aligned}
&-59935\ 53346\ 24310\ 88778\ 33311\ 90376\ 39936\ 85044\ 42378\ 84635\\
&\ \ 33261\ 92070\ 15869\ 79899\ 06527\ 34745\ 13636\ 93302\ 10350\ 88281\\
&\ \ 13714\ 32336\ 14031\ 04181\ 68279\ 33913\ 46326\ 95349\ 47903\ 72694\\
&\ \ 58489\ 05389\ 82045\ 06983\ 02814\ 19721\ 03781\ 51820\ 22732\ 87432\\
&\ \ 20993\ 52184\ 88241\ 57358\ 59346\ 84139\ 81716\ 05698\ 49494\ 16263\\
&\ \ 68 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad y^5\\
&-10978\ 88118\ 89491\ 87821\ 86450\ 95984\ 09802\ 97010\ 10601\ 43786\\
&\ \ 86034\ 65488\ 70860\ 91111\ 85422\ 98921\ 35209\ 89832\ 05905\ 95402\\
&\ \ 33747\ 86186\ 85436\ 81224\ 92763\ 04809\ 69507\ 23487\ 06285\ 68695\\
&\ \ 31597\ 03460\ 06106\ 20930\ 10580\ 28303\ 39558\ 47497\ 41753\ 14485\\
&\ \ 22419\ 76175\ 85976\ 16163\ 00920\ 88614\ 23505\ 55750\ 84124\ 36185\\
&\ \ 088 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad y^4\\
&-90831\ 14977\ 88531\ 39190\ 18018\ 79450\ 26508\ 11068\ 79323\ 00746\\
&\ \ 68357\ 51013\ 32152\ 96914\ 83618\ 50819\ 81901\ 81202\ 99500\ 69853\\
&\ \ 57957\ 44000\ 39111\ 20801\ 77977\ 88121\ 10776\ 35886\ 94344\ 85769\\
&\ \ 35644\ 94527\ 74764\ 38877\ 28966\ 85178\ 74882\ 16162\ 27122\ 20302\\
&\ \ 11295\ 25385\ 25857\ 27698\ 96915\ 00860\ 80652\ 41812\ 58746\ 23938\\
&\ \ 56 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad y^3\\
&-40909\ 39860\ 01761\ 12004\ 29742\ 70531\ 85446\ 05053\ 72802\ 70794\\
&\ \ 09098\ 55278\ 15278\ 54443\ 93918\ 39343\ 51396\ 64068\ 50200\ 09643\\
&\ \ 26195\ 10680\ 89121\ 40765\ 58296\ 57790\ 83797\ 36989\ 27990\ 25075\\
&\ \ 51876\ 39320\ 58034\ 74578\ 55240\ 04156\ 93001\ 35538\ 98932\ 21269\\
&\ \ 62533\ 72166\ 71552\ 49007\ 94760\ 42819\ 06123\ 81947\ 06957\ 79205\\
&\ \ 12 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad y^2\\
&-96985\ 14769\ 39117\ 67357\ 16065\ 05306\ 62373\ 70111\ 52542\ 56050\\
&\ \ 25037\ 88155\ 10682\ 16545\ 18865\ 39610\ 11235\ 39760\ 01040\ 99925\\
&\ \ 28897\ 35422\ 22451\ 97566\ 19253\ 75412\ 51607\ 75770\ 19359\ 87301\\
&\ \ 17086\ 71302\ 03566\ 84608\ 15238\ 88436\ 99703\ 45076\ 75473\ 29646\\
&\ \ 17031\ 33490\ 55613\ 92970\ 67560\ 17607\ 14182\ 14372\ 32632\ 88934\\
&\ \ 4 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad y\\
&-94912\ 68824\ 90969\ 31782\ 93480\ 27582\ 10096\ 24778\ 39152\ 88912\\
&\ \ 40719\ 34500\ 40769\ 49540\ 05610\ 19960\ 00411\ 15796\ 43295\ 55610\\
&\ \ 95611\ 55439\ 44131\ 16494\ 38010\ 65284\ 91752\ 40021\ 08910\ 38155\\
&\ \ 01007\ 63093\ 12121\ 00963\ 96889\ 39867\ 25307\ 09581\ 20650\ 43596\\
&\ \ 34493\ 54945\ 73856\ 22301\ 77805\ 78072\ 48755\ 65346\ 19482\ 39872.
\end{aligned}
$$

参 考 文 献

[1] Alefeld, G. and Herzberger, J.: Introduction to Interval Computations. Academic Press, 1983.

[2] Alonzo, M. E., Becker, E., Roy, M.F., Wörmann, T.: Zeros, multiplicities and idempotents for zero dimensional systems, in Gonz/'alez-Vega, L. *et al.* (ed.), *Algorithms in Algebraic Geometry and Applications*, Birkhäuser, Basel, 1–16, 1996.

[3] Anai, H., Noro, M., and Yokoyama, K.:Computation of the splitting fields and the Galois groups of polynomials, *Progress in Mathematics*, **143**, 29–50, Birkhäuser, 1996.

[4] Anai, H., Yanami, H., SyNRAC: a Maple package for solving real algebraic constraints. *Presented at International Workshop on Computer Algebra Systems and their Applications (CASA), Proc. ICCS 2003, LNCS*, **2657**, 828–837, Springer-Verlag, 2003.

[5] Anai, H. and Yokoyama, K.: Radical representation of polynomial roots, Technical Report ISIS–RR–94–13E, Fujitsu Laboratories, ISIS, 1994.

[6] Anai, H. and Hara, S.: A Robust Control System Design by a Special Quantifier Elimination Methods using a Sturm-Habicht sequence, *Proc. IMACS-ACA'99*, El Escorial(Spain), 1999.

[7] Anai, H. and Hara, S.: Fixed-structure robust controller synthesis based on sign definite condition by a special quantifier elimination, *Proc. ACC2000*, Chicago (USA), 2000, 1312–1316.

[8] Batut, C., Bernardi, D., Cohen, H., and Olivier, M.: User's Guide to PARI-GP, 1995.

[9] Becker, E., Marinari, M. G., Mora, T., Traverso, C.: The shape of the Shape Lemma, *Proc. ISSAC'94*, ACM Press, 129–133, 1994.

[10] Becker, T. and Weispfenning, V.: Gröbner Bases. Graduate Texts in Math. 141, Springer-Verlag, 1993.

[11] Berlekamp, E. R.: Factoring Polynomials over Finite Fields, Bell System Tech. J. **46**, 1853–1849, 1967.

[12] Boehm, H. and Weiser, M.: Garbage collection in an uncooperative environment. *Software Practice & Experience*, 807—820, 1988.

[13] Böge, W., Gebauer, R., and Kredel, H.: Some examples for solving systems of algebraic equations by calculating Gröbner bases, *J. Symb. Comp.*, **2**, 83–98, 1986.

[14] Brown, R. S. : On Euclid's Algorithm and the Computation for Polynomial Greatest Common Divisors, *Journal of Association for Computing Machinery*, **18**, 476–504, 1971.

[15] Brown, R. S. and Traub, J. F. : On Euclid's Algorithm and the Theory of Subresultants, *Journal of Association for Computing Machinery*, **18**, 505–514, 1971.

[16] Buchberger, B.: Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal (An Algorithm for Finding the Basis Elements in the Residue Class Ring Modulo a Zero Dimensional Polynomial Ideal), PhD Thesis, Mathematical Institute, University of Innsbruck, Austria, 1965.

[17] Buchberger, B.: Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Geichungssystems, *Aequ. Math.*, **4**, 3, 374–383, 1970.

[18] Buchberger, B.: A criterion for detecting unnecessary reductions in the construction of Gröbner bases. *Proc. EUROSAM'79 (LNCS 72)*, Springer-Verlag, 3–21, 1979.

[19] Buchberger, B., A Note on the Complexity of Constructing Gröbner-bases, *Proc. EUROCAL'83*, 137–145, 1983.

[20] Buchberger, B., Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory, in *Multidimensional Systems Theory* (editor N. K. Bose), D. Reidel Publ. Comp., 184–232, 1985.

[21] Collins, G. E. : Subresultans and Reduced Polynomial Remainder Sequences, *Journal of Association for Computing Maxhinery*, **14**, 128–142, 1967.

[22] Collins, G. E. : Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition, *LNCS*, **33**, 134–183, Springer-Verlag, 1975.

[23] Davenport, J. H.: On the integration of algebraic functions, LNCS, **102**, Springer, 1981.

[24] Fateman, R.: Honest Plotting, Global Extrema, and Interval Arithmetic, *Proceedings of ISSAC '92*, New York, 216–223, 1992.

[25] Faugère, J. C., Gianni, P., Lazard, D., and Mora, T.: Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering, *J. Symb. Comp.*, **16**, 4, 329–334, 1993.

[26] Ford, D. J., Norton, S., and MacKay, J.: More on replicable functions. *Comm. Alg.*, **22**, 5175–5193, 1994.

[27] Forest, E. and Ruth, R. D.: Fourth-order symplectic integration, *Physica D*, **43**, 105–117, 1990.

[28] Fujimoto, M., Suzuki, M.: AsirPad—a computer algebra system with handwriting interface on PDA, 11th International Conference on Applications of Computer Algebra（ACA'2005), 31 July-3 Aug. 2005, Nara Women's University.

[29] Gebauer, R. and Moeller, H. M.: On an installation of Buchberger's algorithm. *J. Symb. Comp.*, **6**, 2/3, 275–286, 1988.

[30] Giovini, A., Mora, T., Nielsi, G., Robbiano, L., and Traverso, C.: "One sugar cube, please" OR Selection strategies in the Buchberger algorithm. Proc. ISSAC'91, ACM Press, 49–54, 1991.

[31] Kahrimanian, H. G.: Analytical Differentiation by a Digital Computer, M.A. Thesis, Temple University, Philadelphia, Pa., 48pages, May 1953.

[32] Katsura, S., Fukuda, W., Inawashiro, S., Fujiki, N. M. and Gebauer, R. : Distribution of Effective Field in the Ising Spin Glass of the $\pm J$ Model at $T = 0$, *Cell Biophysics*, **11**, 309–319, 1987.

[33] Kawano, Y., Kimura, K., Sekigawa, H., Shirayanagi, K., Noro, M., Kitagawa, M., and Ozawa, M. : Existence of the exact CNOT on a quantum computer with the exchange interaction, *Quantum Information Processing*, **4**, 2, 65-85, Springer, 2005

[34] Kimura, T. and Hara, S.: A robust control system design by a parameter space approach based on sign definite condition, *Proc. KACC'91*, Seoul(Korea), 1533–1538, 1991.

[35] Kimura, T. and Hara, S.: Robust Control Analysis Considering Real Parametric Perturbations Based on Sign Definite Conditions, *Proc. IFAC'93*, **1**, 37–40, Sydney(Australia), 1993.

[36] Knuth, D. E.: *Fundamental Algorithms*, The Art of Computer Programming, **2**, Addison-Wesley, Reading, Massachusetts, 2nd edition, 1981.

[37] Kobayashi, H., Moritsugu, S., Hogan, R. W., On Solving Systems of Algebraic Equations, paper presented at ISSAC'88, Rome (July 1988), *JSC*, **8**, 6, 1989.

[38] Kondoh, Y., Saito, T. and Takeshima, T. : A New Algorithm for Real Roots of a Zero-Dimensional System by a Linear Separating Map, in *Computer Mathematics*, *eds. Shirayanagi, K. and Yokoyama, K.*, *Proceedings of ASCM2001 held at Matsuyama, Japan (Sept. 26–28, 2001)*, World Scientific, 56–65, 2001.

[39] Koseleff, P. V.: Relations among Lie Formal Series and Construction of Symplectic Integrators. Proc. AAECC-10 (LNCS 673), Springer-Verlag, 215–230, 1993.

[40] Loos, R. and Weispfenning, V. : Applying linear quantifier elimination, *The Computer Journal*, **36**, 5, 450-462, 1993.

[41] Moses, J. and Yun, D. Y. Y. : The EZGCD Algorithm, *Proc. ACM Annual Conf.*, ACM, 159–166, 1973.

[42] Neri, F.: Lie algebras and canonical integration, Dept. of Physics, University of Maryland, preprint, 1987.

[43] Nolan, J.: Analytical differentiation on a digital computer, M.A. Thesis, Math. Dept., MIT, Cambridge, Mass., 71pages, May 1953.

[44] Noro, M., McKay J. : Computation of replicable functions on Risa/Asir, *Proc. PASCO'97*, ACM Press, 1997.

[45] Noro M., Shimoyama T. and Takeshima T. : Asir User's Manual, ftp://ftp.fujitsu.co.jp/pub/isis/asir/

[46] Noro, M., Takeshima, T.: Risa/Asir—A Computer Algebra System, *Proc. ISSAC92*, 387–396, ACM Press, 1992.

[47] Noro, M. and Yokoyama, K.: New methods for the change-of-ordering in Gröbner basis computation, Technical Report ISIS–RR–95–8E, Fujitsu Laboratories, ISIS, 1995.

[48] Noro, M. and Takeshima, T.: High-Quality Computing of Polynomial Problems by Risa/Asir, *FUJITSU Scientific and Technical Journal*, **32**, 256–270, 1996.

[49] Noro,M., Yokoyama, K.: A Modular Method to Compute the Rational Univariate Representation of Zero-Dimensional Ideals, *J. Symbolic Computation*, **11**, 1999.

[50] Norman, A. C., Moore, P. M. A.: Implementing the New Risch Algorithm, Proc. 4th International Symposium on Advanced Comp. Methods in Theor. Phys., CNRS, Marseilles, 1977.

[51] Oaku, T.:Algorithms for *b*-functions, restrictions, and algebraic local cohomology groups of *D*-modules, *Advances in Applied Mathematics*, **19**, 61–105, 1997.

[52] Oaku, T.: Algorithms for the b-function and D-modules associated with a polynomial, Proc. MEGA'96; J. Pure Appl. Algebra, **117** & **118**, 495–518, 1997.

[53] Orii, S., Anai, H. and Horimoto, K. : Symbolic-Numeric Optimization for Biological Kinetics by Quantifier Elimination, *Genome Informatics 2004, poster presentation at 15th Int. Conf. on Genome Informatics*, Yokohama, Japan, 2004.

[54] Orii, S., Anai, H. and Horimoto, K. : Symbolic-numeric Estimation of Parameters in Biochemical Models by Quantifier Elimination, *BIOINFO2005, Proc. International Joint Conference of InCoB, AASBi and KSBI, held in Busan, Korea*, 272-277, 2005.

[55] Pankov, I., Varbanova, E.A. and Watkins, A.J. : Teaching and learning mathematics with technology—a Balkan experience, *Proc. 11th Int. Conf. on Technology in Collegiate Mathematics, New Orleans, USA*, 1998.

[56] Ponder, C. G., Parallelism and Algorithms for Algebraic Manipulation: Current Work, SIGSAM Bulletin, **22**, 7–14, 1988.

[57] Pedersen, P., Roy, M.-F. and Szpirglas, A.: Counting real zeros in the multivariate case Computational Algebraic Geometry, *Progress in Mathematics*, Bierkhäuser, 203–223, 1993.

[58] Rich, Albert D., Stoutemeyer, David R. : Capabilities of the MUMATH-78 computer algebra system for the INTEL-8080 microprocessor (invited), EUROSAM 1979: 241–248.

[59] Risch, R.: The Problem of Integral in Finite Terms, Trans. Amer. Math. Soc. **139**, 167–189, 1969.

[60] Saito, T.: An extension of Sturm's theorem to two dimensions, *Proceedings of the Japan Academy*, **73** A, 18–19, 1997.

[61] Saito, T., Kondoh, Y., Miyoshi, Y. and Takeshima, T. : Fathful plotting on a two dimensional pixel space, in *Jhosai Mathmatical Monographs*, **2**, *ed. Kiyoko Nishizawa*, *Proc. The Fourth Symposium on Nonlinear Analysis (NLA99) held at Josai Univ. Sept. 16-18, 1999*, 77–86, Josai Univ., 2000.

[62] Sasaki, T. and Kanada, Y., Parallelism in Algebraic Computation and Parallel Algorithms for Symbolic Linear Systems, *Proc*. SYMSAC'81 (editor P. S. Wang), ACM, 160–167, 1981.

[63] Sasaki, T., Murao, H. : Efficient Gaussian elimination method and linear systems, ACM Trans. Math. Software, **8**, 277–289, 1982.

[64] Sasaki, T. : Some Algebraic Algorithms based on Head Term Elimination over Polynomial Rings, *Proc. EUROCAL'87* (*Lecture Notes in Comp. Sci.*,**378**), 348–354, 1987.

[65] Sasaki, T. and Takeshima, T.: A Modular Gröbner Basis Method for Algebraic Equations, 京大・数解研・研究集会「数式処理と数学研究への応用」, 1987 年 11 月, 数理解析研究所 講究録 **646**, 1988 年 2 月.

[66] Sasaki, T. and Takeshima, T : A Modular Method for Gröbner-basis Construction over $\mathbb{Q}$ and Solving System of Algebraic Equations, in *J. of Inf. Process.*, **12**, 4, 371–379, 1990; see also, A Modular Gröbner-basis Method for Algebraic Equations, in Reports of RIKEN Symposium, Feb. 1988.

[67] Shimoyama, T. and Yokoyama, K.:Localization and primary decomposition of polynomial ideals, *J. Symbolic Computation*, **22**, 247–277, Accademic Press, 1996.

[68] Shimoyama, T. and Kaneko, T. : Quadratic Relation of S-box and its Application to the Linear Attack of Full Round RES, *Proc. the 18th Annual Int. Cryptology Conf., Advances in Cryptology–CRYPTO'98, LNCS* **1462**, 200-211, 1998.

[69] Strelitz, SH.: On the Routh-Hurwitz Problem, *American Mathematical Monthly*, **84**(7), 1977, 542–544.

[70] Suzuki, M., Sasaki, T., Sato, M., Fukui, Y : A hybrid algeraic-numeric system ANS and its perliminary implementation, LNCS **378**, 163–171, 1989.

[71] Takayama, N.: sm1 Reference Manual, 1994.

[72] http://www.math.s.kobe-u.ac.jp/KAN/index.html

[73] Takeshima, T., Noro, M. and Yokoyama, K.: Report on Computer Algebra Research, *FUJITSU Scientific and Technical Journal*, **27**, 338–359, 1991.

[74] Takeshima T., Yokoyama K. and Noro M.: Experience with Risa/Asir for Polynomial System Problems, invited presentation at Posso Workshop on Software, Université Paris VI, Feb. 1995.

[75] Takeshima T., Noro M. and Saito T. : Graphic Drawing Apparatus for Generating Graphics of Implicit Functions, U.S.Patent No.5590255, 1996.

[76] Takeshima, T.: Minimal polynomial test for boundary polynomials—another SDC formulation of *D*-stability, preprint presented at Risa Conference 05 held at Kobe Univ., March 2005.

[77] Takeshima, T.: Strelitz test for stable polynomials and its application to design problems of control systems, 数式処理, **11**, 3& 4（合併号）, 日本数式処理学会, 153–164, 2005.

[78] Tarski, A. : *Decision Method for Elementary Algebra and Geometry*, University of Carifornia Press, Berkeley, California, 1951.

[79] Traverso, C. : Gröbner Trace Algorithm, *Proc. ISSAC'88* (LNCS 358), 125–138, 1988.

[80] Trinks, W., On Improving Approximate Results of Buchberger's Algorithm by Newton's Method, SIGSAM Bulletin, **18**, 7–11, 1984.

[81] Varbanova, E.: A CAS Supported Environment for Learning and Teaching Calculus, *Proc. 1st KAIST Int. Sympos. on Enhancing University Mathematics Teaching, KAIST, Korea,* May, 2005.

[82] Wang, P. S. and Rothschild, L. P. : Factoring Multivariate Polynomials over the Integers, *Mathematics of Computing*, **29**, 935–950, 1975.

[83] Wang, P. S. : An Improved Multivariate Polynomial Factoring Algorithms, *Mathematics of Computing*, **32**, 1215–1231, 1978.

[84] Wang, P. S. and Trager, B. M. : New Algorithms for Polynomial Square-free Decomposition over the Integers, *SIAM J. Computing*, **8**, 300–305,, 1979.

[85] Weispfenning, V. : Quantifier elimination for real algebra - the cubic case, *Proc. ISSAC 94, Oxford, 1994*, ACM Press, 258-263, 1994.

[86] Weispfenning, V. : Quantifier elimination for real algebra - the quadratic case and beyond, *AAECC* **8**, 85-101, 1997.

[87] Yokoyama, K. and Takeshima, T. : Factorization of Uni-variate Polynominals over Finite-fields, *Fujitsu IIAS Research Report*, No. 69, 1986.

[88] Yokoyama K., Noro M. and Takeshima T. : Computing Primitive elements of extension fields, *J. Symbolic Computation*, **8**, 553–580, 1989.

[89] Yokoyama, K. and Takeshima, T. : On factoring multivariate polynomials over algebraically closed fields, *RISC-LINZ Research Report*, 90–26, 1990.

[90] Yokoyama, K., Takeshima, T. and Noro M., On factoring multivariate polynomials over algebraically closed fields, ISSAC'90, Poster Presentation, 1990.

[91] Yokoyama, K. Takeshima, T. and Noro M., On determining the solvability of polynomials, *Proc. ISSAC'90*, ACM PRESS, 127–134, 1990.

[92] Yokoyama, K., Noro, M., and Takeshima, T.: Solutions of systems of algebraic equations and linear maps on residue class rings. *J. Symbolic Computation*, **14**, 399–417, 1992.

[93] Yokoyama, K., Noro, M. and Takeshima, T. : A New Approach to Polynomial-Time Factorization of Bivariate Integral Polynomials, 理化学研究所シンポジウム「代数計算とその先端的科学技術計算への応用」予稿, 1992 年 3 月.

[94] Yokoyama, K., Noro, M. and Takeshima T.: A Polynomial-Time Algorithm for Factoring Bivariate Polynomials by Using their Zeros, ソフトウェア科学会 数式処理研究会 予稿, 1992.

[95] Yokoyama, K., Takeshima, T. and Noro, M. : A Polynomial-Time Algorithm for Factoring Bivariate Polynomials by Using their Zeros, ISSAC'92, Poster Presetation, 1992.

[96] Yokoyama, K. and Takeshima, T. : On Hensel Construction of Eigenvalues and Eigenvectors of Matrices with Polynomial Entries, *Proc. ISSAC'93*, ACM PRESS, 218–224, 1993.

[97] Yokoyama, K., Noro, M. and Takeshima, T. : A New Approach to Polynomial-Time Factorization of Bivariate Integral Polynomials, *Fujitsu IIAS-SIS Research Report*, RR–92–10E, 1993.

[98] Yokoyama, K., Takeshima, T. and Noro, M. : Multi-modular Approach to polynomial-time factorization of bivariate integral pollynomials, *J. Symbolic Computation*, **17**, 545–563, 1994.

[99] Winkler, F. : p-adic Methods for the Computation of Gröbner Bases, paper presented at EUROCAL'87, June 1987, also in JSC, **6**, 2 & 3, 287–304, 1988.

[100] Zassenhaus, H.: On Hensel factorization I, J. Number Theory 1, 291–311, 1969.

[101] 池原悟，岡田博，池田義則，神原慎一，小田泰充，今福幸春，角田俊晴：数式処理言語 AL の設計と評価，研究実用化報告，**26**，8，2463–2482，NTT 電気通信研究所，1977.

[102] 池原悟，岡田博： 数式処理言語 AL とその処理方式，情報処理，**20**，2，情報処理学会，158–165，1979.

[103] 伊理 正夫，計算の品質，*bit*，**28**，9，52–55，1996.

[104] 桂重俊，鈴木祥介，竹島卓 : Reduce，Mathematica，Maple，Asir—初等計算および Katsura equation，第 4 回 Risa Consortium 研究集会 予稿，1996; 東北科学技術短期大学紀要 **3**，62–72，1997.

[105] 加藤昭彦, 野呂 正行, 竹島卓 : 数式処理システム risa のパーソナルコンピュータへの移植について，日本数式処理学会 第 1 回大会予稿，数式処理，**1**，2，57–60，1992.

[106] 加藤昭彦，野呂正行，竹島卓 : 数式処理システム Risa/Asir におけるリスト処理，高階関数処理ライブラリの作成，日本数式処理学会第 2 回大会 予稿 （1993 年 5 月），数式処理，**2**，2，48–51，1993.

[107] 金堀 利洋，西村 博人，藤本 光史，鈴木 昌和: 数学の授業におけるインタラクティブなコンテンツを含んだ授業教材作成システム，電子情報通信学会技術研究報告 **103**，536，117–122，2003.

[108] 木村 欣司，野呂 正行: グレブナー基底計算のための weight 生成アルゴリズム，数理解析研究所 講究録，**1395**，1–7，2004.

[109] 近藤祐史，三好善彦，齋藤友克，竹島卓 : 数式処理と画像処理，埼玉女子短期大学研究紀要，**6**，41–49，1995.

[110] 近藤祐史，齋藤友克，竹島卓 : 線形写像による判定を用いた代数方程式の実解の代数的解法について，京大・数解研・研究集会，1999 年 11 月，数理解析研究所 講究録 **1138**，2000.

[111] 近藤祐史，齋藤友克，竹島卓 : 線形写像による判定を用いた代数方程式の実解の代数的解法について，日本数式処理学会 第 9 回大会報告（2000 年度），数式処理 **8**，1，26–27，2000.

[112] 近藤祐史，齋藤友克，竹島卓 : An Efficient Real Root Locating of Polynomial Equations by Linear Separating Maps—線形分離写像による判定を用いた代数方程式の実解の定位，京大・数解研・研究集会，2000 年 12 月，数理解析研究所 講究録 **1199**，192–202，2001.

[113] 近藤祐史，齋藤友克，竹島卓 : A New Algorithm for Real Roots of a Zero-Dimensional System by a Linear Separating Map, 京都大学数理解析研究所 研究集会 （2001 年 11 月）.

[114] 齋藤 友克，野田 松太郎: 代数方程式系のゼロ次元の解の存在位置の判定，数式処理，**6**，1，4–5，1997.

[115] 齋藤 友克，野田 松太郎: 2 変数代数方程式の実特異点を含む区間の決定，*Proc. 2nd Risa Consortium*，131–139，1998.

[116] 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**，2，2–21，1998.

[117] 齋藤友克，近藤祐史，竹島卓 : 任意精度によるゼロ次元代数方程式の解の位置判定，日本数式処理学会 第 8 回大会報告（1999 年度），数式処理 **7**，3，39–40，1999.

[118] 齋藤友克，近藤祐史，三好善彦，竹島卓 : 2 変数代数曲線の忠実な描画，情報処理学会論文誌，**41**，4，1009–1017，2000.

[119] 齋藤 友克，竹島 卓，平野 照比古 :RISA/ASIR，日本で生まれた数式処理ソフト，リサ アジール ガイドブック，SEG 出版，129 頁，1998 年 10 月.

[120] 齋藤友克，竹島卓，平野照比古 : グレブナー基底の計算 実践篇 Risa/Asir で解く，東京大学出版会，302 頁，2003 年 6 月.

[121] 三枝義典，阿部昭博，佐々木建昭，増永良文，元吉文男，佐々木睦子 : 数式処理システム GAL における数学公式データベースのインデキシング手法，電子情報通信学会論文誌，J74–D–I，577–585，1991.

[122] 佐々木建昭，元吉文男 : 国産数式処理システム GAL における内部表現，記号処理 **029-005**，1984.

[123] 佐々木建昭，元吉文男 : 国産数式処理システム GAL における制御機構と簡単化，記号処理 **031-008**，情報処理学会 記号処理研究会，1985.

[124] 佐々木建昭，元吉文男 : 国産数式処理システム GAL におけるパターンマッチング，記号処理 **035-001**，情報処理学会 記号処理研究会，1985.

[125] 佐々木建昭，竹島卓 : グレブナ基底の並列計算と連立代数方程式，情報処理学会論文誌，**30**，1555–1561，1989.

[126] 沢田 浩之: 新たな条件式の導入による多変数連立代数方程式の解法，情報処理学会論文誌，**36**，12，2761–2770，1995.

[127] 白柳潔 : グレブナ基底入門，システム/制御/情報，**39**，5，225–232，1995.

[128] 関口 次郎: 多面体の数理とグラフィックス，牧野書店，1996.

[129] 竹島卓，横山和弘，野呂正行 : 有限体上の 1 変数多項式の次数別因子への因数分解，日本ソフトウェア科学会第 3 回大会論文集，293–296，1986.

[130] 竹島卓 : 有限体上のある種の線形変換の固有多項式の性質について，京大・数解研・研究集会「数式処理とその数学研究への応用」予稿，1986 年 11 月.

[131] 竹島 卓，横山和弘 野呂正行 : modular 算法による多項式 GCD 計算について，京大・数解研・研究集会「数式処理と数学研究への応用」，1987 年 11 月，数理解析研究所 講究録 **646**，1988 年 2 月．

[132] 竹島 卓，野呂 正行，須永 知之，井深 克憲，塚元 有子 : PSI 上の数式処理システム SAM，情報処理学会第 39 回全国大会予稿集，118–118，1989.

[133] 竹島 卓，佐々木建昭 : A Parallel Groebner Basis Method for Solving a System of Algebraic Equations, presented at Joint Symposium PP'89, Feb. 1989.

[134] 竹島 卓，横山和弘 : 連立代数方程式の一解法—剰余環上の線形写像の固有ベクトルの利用，数式処理通信，**6**，4，1990.

[135] 竹島卓，横山和弘: 行列の有理標準形の一計算法 （行列の固有値・固有ベクトルの代数的解法），京大・数解研・研究集会「数式処理と数学研究への応用」予稿，1990 年 11 月，数理解析研究所 講究録 **753**，1991 年 5 月．

[136] 竹島卓，横山和弘，野呂 正行 : 行列の有理標準形の一計算法 （行列の固有値・固有ベクトルの代数的解法），ワーク ショップ「新しい数式処理システムの開発」，愛媛大学工学部情報工学科，1991 年 3 月．

[137] 竹島卓, 横山和弘 : 代数制約の処理, コンピュータソフトウェア, **8**, 6, 474–488, 1992.

[138] 竹島卓，横山和弘 : 零因子をもつ環上の逆行列計算について，京大・数解研・研究集会「数式処理における理論と応用の研究」 予稿，1992 年 11 月，数理解析研究所 講究録 **848**，1993 年 9 月．

[139] 竹島卓，下山武司 : 陰関数の構成と追跡，日本数式処理学会 第 1 回大会予稿，数式処理，**1**，2，61–64，1992.

[140] 竹島卓，穴井宏和 : 三斜内容三圓術（Malfatti の問題）の数式処理—有理関数体上のタワーの構成，京大・数解研・研究集会，1995 年 11 月，数理解析研究所講究録 **941**，1996 年 3 月．

[141] 竹島卓 : Risa システムによる Polynomial Solving への応用，数式処理学会 第 6 回大会 口頭発表，1997 年 5 月．

[142] 竹島卓 : 代数的計算技法から見た数理ソフトの現状と展望，日本応用数理学会年会予稿，名古屋大学，1997 年 10 月．

[143] 竹島卓 : 数式処理システムの現代暗号への応用，情報処理学会 第 40 回プログラミングシンポジウム 予稿集，箱根ホテル小湧園，1999 年 1 月，121–130，1999.

[144] 竹島卓，近藤祐史，齋藤友克 : Cplot の 3 次元描画への拡張について，日本数式処理学会 第 9 回大会報告（2000 年度），数式処理 **8**，1，42–43，日本数式処理学会，2000.

[145] 竹島卓，近藤祐史，齋藤友克 : 整数行列の固有値固有ベクトルの計算法について，日本数式処理学会 第 12 回大会報告（2003 年度），数式処理 **12**，2，20–21，2003.

[146] 竹島 卓: Risa/Asir の開発，数式処理，**12**，1，9–22，日本数式処理学会，2005.

[147] 坂井忠次: グラフと追跡，培風館，東京 （1963）.

[148] 徳山五郎，池原悟，岡田博： 数式処理言語，研究実用化報告，**24**，1，199–218，NTT 電気通信研究所，1977.

[149] 野田 松太郎，岩下 英俊 : パーソナルなハイブリッド処理システム SYNC の設計，情報処理学会論文誌，**30**，4，419–426，1989.

[150] 野呂正行，竹島 卓 : 整式処理システム NOR-YOKOSYMA の現況，理研シンポジウム「代数的計算法」予稿，1988 年 2 月.

[151] 野呂正行，竹島卓 : 並行処理による数式処理の試み，京大・数解研・研究集会「数式処理と数学研究への応用」予稿，1988 年 11 月，数理解析研究所 講究録 **685**，1989 年 3 月.

[152] 野呂正行，竹島卓，横山和弘，須永知之，塚本有子 : 数式処理システム risa（仮称）の現況—その 2，理研シンポジウム予稿，26–31，1990 年 3 月.

[153] 野呂正行, 竹島卓 : 数式処理システム risa の C-like ユーザ言語 ASIR と dbx-like debugger，日本ソフトウェア科学会数式処理研究会 予稿，ラフォーレ修善寺，1990 年 10 月.

[154] 野呂正行，竹島卓 : 数式処理システム risa のインタフェース，ワークショップ「新しい数式処理システムの開発」，愛媛大学，1991 年 3 月.

[155] 野呂正行，竹島卓 : 数式処理システム risa/asir の内部構造，京大・数解研・研究集会「数式処理と数学研究への応用」，1991 年 11 月，数理解析研究所 講究録 **811**，1992.

[156] 野呂正行，竹島卓 : Risa/Asir の現況，愛媛大学ワークショップ「数式処理システムとその周辺」 予稿，1993 年 3 月.

[157] 野呂正行，竹島卓，数式処理システム Risa/Asir のインプリメンテーション，情報処理学会 記号処理研究会 予稿 1993 年 3 月.

[158] 野呂正行: Asir による replicable function の計算—McKay’s problem，第 4 回 Risa Consortium 研究集会 予稿，東北科学技術短期大学，1996 年 10 月.

[159] 林 平馬他，微分積分学序論: 学術図書出版，2002．Hayashi, H. textit et al.: (Japanese textbook) Bibun-Sekibun-Gaku Joron, Gakujutu Tosho Shuppan, 2002.

[160] 藤本 光史: PDA 用手書き数式入力インターフェース AsirPad の開発，京都大学数理解析研究所 講究録 **1395**，「Computer Algebra—Design of Algorithms, Implementations and Applications」132–137, 2004.

[161] 藤本 光史，鈴木 昌和，金堀 利洋: 数学授業における PDA と手書き数式インターフェースの有効性に関する研究，preprint，日本数式処理学会第 14 回大会，広島大学学士会館，15–17，June 2005.

[162] 藤本 光史: 手書き数式インターフェースを活用した実践授業について, preprint, Risa/Asir Conference 2005，2005 年 3 月，神戸大学.

[163] 谷口行信，杉原厚吉: 検出もれのない代数曲線の追跡法，情報処理学会論文誌，**33**，10，1245–1253，1992.

[164] 横山和弘, 野呂正行, 竹島卓 : 有限体上の一変数多項式の因数分解について, ICOT Technical Report TR-181，Institute for New Generation Computer Technology，1986.

[165] 横山和弘, 野呂正行, 竹島卓 : ある種の環上の多項式の因数分解について（Lenstra の方法とその一般化），ICOT Technical Report TR-155，Institute for New Generation Computer Technology，1986.

[166] 横山和弘，竹島卓 : modular 算法による多項式 GCD 計算について，第 4 回国際研夏期シンポジウム 報告集，富士通，148–152，1987.

[167] 横山和弘，野呂正行，竹島卓 : Q の代数的拡大体の原始元の算出法について，第 4 回国際研夏期シンポジウム 報告集，富士通，34–50，1987.

[168] 横山和弘，竹島卓 : Euclid 環上の因数分解および GCD について—格子算法の応用，コンピュータソフトウェア，**8**，1，42–61，1988.

[169] 吉田春夫 : シンプレクティック数値解法，数理科学，**33**，6，37–46，1995.

[170] 特集「精度保証付き数値計算とその応用」，情報処理，**31**，9，1990.

数式処理システム Risa/Asir の開発と応用
2005 年 12 月発行
発行人　竹島卓
著者　竹島卓
印刷・製本　株式会社 平河工業社