Abstract

JHD's notes taken at the time. Note that there were parallel sessions, so not every talk is even mentioned. JHD has also not yet proof-read them, or checked them against the proceedings' papers: these are very much "notes in the wild".

ISSAC 2011 Notes

J.H. Davenport — J.H.Davenport@bath.ac.uk

9–11 June 2011
(updated 2 August 2011)

Contents

Chapter 1

9 June 2011

1.1 Sparse Differential Resultants — Li, Gao & Yuan

The resultant in the differential case is not fully explored, and the sparse differential resultant has not been studies before. Past work [Rit32] introduced the idea of differential resultants, and [CF97] continued, but her work was not complete. Sparse resultants are due to [Stu93].

1.1.1 Generic Intersection Theory

Consider a differential field (F, δ), variables $Y = (y_1, \ldots, y_n)$ and differential extension $F\{Y\} = F[y_i^{(k)}]$. Let $P_0, \ldots, P_n$ be $n+1$ generic differential polynomials with coefficients u_i. Is $[P_1]$ a prime ideal of dimension $n-1$? And so on?

Theorem 1 (Generic Intersection Theorem) *Let I be a prime differential ideal of dimension d and order $s.$, P_o a generic differential polynomial with coefficients (u_i) . Then ...*

The intersection theorem $\dim(V \cap W) \geq \dim(V) + \dim(W) - n$ is not true in the differential case. [Rit50].

Conjecture 1 (Ritt's dimension conjecture) $\dim(t_1, \ldots, t_r) \geq n - r$.

1.1.2 Sparse Differential Resultants

Let $P - i = u_{i,0} + \sum_{k=1}^{l_i} u_{i,k} M_{i,k}$ where the M are monomials and the u coefficients. The sparse differential resultant is

$$[P_0, \ldots, P_n] \cap$$

Lemma 1

- $\delta \operatorname{Res}(u_0, \ldots, u_k) = \ldots$

- *The sparse differential resultant has a weaker product formula*
- *The algebraic resultant is a linear combination of the input, and the sparse differential resultant a linear combination of the* derivatives *of the inputs.*
- *The differential resultant gives a sufficient condition for the existence of common solutions.*

1.1.3 A Singly-exponential algorithm

Let $\mathrm{ord}(P_i) + s_i$ and $\deg(P_i) = m_i$. Let R be the sparse differential resultant.

Theorem 2 *1.* $\mathrm{ord}(R, u_i)$

2. . . .

We use differential specialization techniques, and the algebraic generalised Chow form.

JHD found it hard to reconcile the slides with the paper.

1.2 Deflation and Certified Isolation of Singular Zeroes of Polynomial Systems — Mantzaflaris & Mourrain

Let ζ be as isolated root of I. Then there is a primary component Q whose radical is the maximal ideal at this point. The multiplicity of ζ is $\dim R/Q$. [Mac12] [Beckeretal1995] [[]Chapter 4]Mourrain1997 [DaytonZeng] [Leykinetal2006]

Plan

1. describe the multiplicity structure via differential conditions
2. deflate the root, and certify the root in a domain
3. compute geometry around singularities

Define the dual space R^* for $R = K[x_1, \ldots, x_{+n}]$. This is equivalent to formal series $K[[\delta_1, \delta_n]]$.

We compute a new system (Theorem 4.5) which has an isolated root at ζ.

1.3 Fast algorithm for Change of Ordering of Zero-dimensional Gröbner Bases with Sparse Multiplication Matrices — Faugère & Mou

Given a GB G_1 of a zero-dimensional ideal w.r.t. $<_1$, compute G_2 with respect to $<_2$. FGLM [FGLM93] solves this in $O(nD^3)$, where n is number of variables and D the number of solutions. But in many examples, the (Magma) times for FGLM are far larger than those for the initial solution.

1.3.1 FGLM revisited

1. Compute canonical basis B w.r.t $<_1$
2.
3. Handle terms in $K[x_1, \ldots, x_n]$ **one by one** with respect to $<_2$, and this Gaussian elimination is the $O(D^3)$ contribution..

Our observation is that the multiplication matrix T_i is sparse — below 7% even for a randomly-generated dense problem. For "real" problems, even sparser. We are inspired by Wiedemann. We need Berlekamp–Massey–Sakata (BMS) algorithm [Sak88],[Sakata1990]. Complexity is $O(lk^2)$ where l is the number of polynomials returned.. The main part of BMS is a routine BMSUpdate. This also handles terms in $K[x_1, \ldots, x_n]$ **one by one**

1.3.2 Shape Position Case

[BMMT94]: $f_1(x_i)$ and $x_i - f_i(x_1)$ for $i > 1$. We recover f_1 via Wiedemann, and then recover the solution over f_1 by constructing linear equations. The coefficient matrix H is a Hankel matrix and the construction is free. Hence $O(D(N_1 + n\log(D)))$ where N_1 is the number of nonzeros in T_1 the FGLM matrix. Note that this is probabilistic.

1.3.3 General case

Define E : terms $\to K$, compute its Gröbner basis $\tilde{G}$ with BMS, and check that the result is G_2. The complexity of each iteration is $O(N + \tilde{N}\overline{N}D)$ where $\tilde{N}$ is the number of polynomials in G_2 and $\overline{N}$ is the maximal number of terms in G_2. The number of iterations in bounded by $2ND$ (where the factor of 2 comes from usual BM argument).

1.3.4 Main algorithm

Input Multiplication matrices

1. Compute the linearly recurring sequence
2. Compute I with BM
3. Check the degree is D (i.e. Shape Lemma case)

Yes Reconstruct Hankel matrix and solve the rest of the system

No run BMSupdate until termination

Speedups over Magma and Singular of factors of 50 or more, often ∞. Can handle $D \approx 40,000$.

1.4 Linear Algebra to Compute Syzygies and Gröbner Bases — Cabarcas & Ding

In particular, we want to analyse the cost of redundancy in GB computation. Introduce `LASyz` (Linear Algebra to compute Syzygies), which uses known syzygies to avoid redundant computation. Note that a syzygy is simultaneously a reduction to zero and additional information about the ideal..

Let $R = k[x_1, \ldots, x_n]$, with a monomial ordering $<$, and hence lm, A syzygy is $\alpha = (\alpha_1, \ldots, \alpha_m)$ such that $\sum \alpha_i p_i = 0$. Syz(P) is a graded R-module.

Suppose P_i homogeneous . Let $P_d = \{tp | t \in M, p \in P \deg(tp) = d\}$ and $P_{(d)} = \bigcup_{j=0}^{d} P_j$. Define $\sigma_d : \text{Syz}(P_{(d)}) \to \text{Syz}(P_d)$ by ....

Example polynomials of degree 2. P_3 is not linearly independent. These syzygies will give rise to syzygies of P_4.

Theorem 3 *If P is a regular sequence of polynomials then* Syz ....

Theorem 4

Can express Faugère's criterion F_5 in terms of redundancies of syzygies: If G is an echelon form of $P_{d-d_i,j-1}$ and t is a leading monomial of ..., then tp_i is redundant in $P_{d,i}$. The problem is that this requires an incremental computation.

Claims that this is the first non-incremental method of eliminate redundant computations. Still working on complexity bounds.

1.5 FCRC Plenary — Luis van Ahn

1.5.1 Introduction

[The speaker was introduced as the inventor of "human computation". Apparently he is "Guatemalan of the Year"]

Asked how many had seen "capchas" — he invented them. They are used all over the Internet, and "random" can be unfortunate: e.g. "WAIT", which caused the help desk complaint

A Capcha is a program which can generate and grade tests that it itself cannot pass (just like some professors). Note the "Russian capcha", a limit computation, and the Indian capcha, which analyses a circuit. The American capcha is "What is 1+1".

Next project (CMU to startup to Google) is Recapcha (2007). About 200M are typed every day. Is there any way in which we can use this effort. So we want to use this effort. We are now using this effort for digitizing books. Note that OCR is flawed.

1. Scan a book
2. Take the "failed" words (confidence less than some limit)

3. Use this scan, and a word for which we **do** use the answer.

4. If the known one is right, we use the unknown one as input to the scanning process

5. Digitising 100M words/day this way (Friends Reunited, Ticketmaster, Craig's List etc.).

Note that this also generate unfortunates, e.g. "damn liberals" on `johnedqards.com`.

Note that we are using two words rather than one, but in fact this takes no more time, since they are (meant to be) words rather than random characters. Note that there are actually "capcha sweatshops", but at least this generates jobs and means that spammers have at least one good side-effect.

He spoke (on-screen) to an owner of a capcha sweatshop. He told them exactly how he dealt with IP capcha blocking ()so gave them much greater texts to process). about 750M people (10% of the world) have contributed

The average American spends 1.1 hours/day on electronic games. The next problem to tackle is that of labelling images with words. Computers can't do this, e.g. "Google images". This uses file names and textual contexts (and in error-prone). We use the "ESP game". This is a two-player on-line game. Partners don't know each other and can't communicate. The object of the game is to type the same word at the same time. The only thing in common is an image. The word they agree on is generally a good label for the image. When players have agreed on a word, that becomes a "tabu word", which lets us generate new labels. 200,000 players game 50M agreements, with people playing for up to 16 hours on the stretch (there's now a cutoff at 15 hours, or "10 from `.edu`")..

People really like the ESP game — why? It gives a wonderful feeling of anonymous intimacy.

1.5.2

Another such game is "Amazon Mechanical Turk" (AMT). — 2006. Another project (for Un. Washington) is FoldIt (2009), based on protein folding. Last years there were over 250 papers self-classifying as 'human computation".

Claim that great achievements (pyramids, Panama Canal, Man on the Moon) all used about 100,000 people. Pre-Internet limit of scale. What could we do if we could harness more? Note scale of digitization project. How do we get 100M people translating the web into every major language for free. "Machine translation is not very good yet". Note that paid translators are impossible. Wikipedia into Spanish, say, would cost greater than $50M. The main problems is the lack of bilinguals and money. Solve both with one stone — teach new languages. In the U.S. alone, over 5M people have paid $500 for language-learning software. Hence `duolingo` — learn a new language while simultaneously translating the ??. You're learning with real content. Claims the translations we get out (after

picking the best one[1]!) are about as good as professional — demonstrated a tricky German→English example.

This is the written language. He's currently doing the same for the spoken language using subtitling videos as the test case.

[1] JHD: but he nowhere explained what he meant by "best" — is there manual intervention here?

Chapter 2

9 June 2011: software presentations

2.1 Defect polytopes and counter-examples with `polymake` — Joswig & Paffenholz

2.1.1 `polytope`

Started with polytopes, now does simplicial complexes, tropical geometry etc. Recent versions include

1. graphs
2.

Polytopes are homogenized. `$P->VERTICES=<<"." data \verb.+` will input the vertices, and we can ask for properties of this, being deduced as appropriate. Objects are immutable. Link out to `Singular` and many other tools as appropriate.

2.1.2 Mathematics

Lattice polytopes are related to toric varieties. Let

$$c_t(P) := \sum_{k=0}^{d} (-1)^{d-k} \frac{(k+t)!}{k!} \sum_{F \in F_k(k)} \text{LatticeVol}(F).$$

Theorem 5 *For smooth P, $c_t(P) \geq 0$.*

Conjecture, true whether or not P is smooth. Disproved recently with a family of examples.

2.2 Software for exact integration of polynomials over polyhedra — De Loera *et al.*

Integrate with respect to the integral Lebesgue measure, so that the diagonal of the unit square has length 1, not $\sqrt{2}$.

A C++ package, developed since 2001.

2.3 Simplicial blowups and discrete normal surfaces in `simpcomp` — Effenberger & Spreer

A GAP extension (therefore uses existing language and infrastructure), in particular GRAPE (interface to `nauty`) and *homology*. Since all functions being `SC`, loading the package and typing `SC<tab>` gives a completion list.

Features include constructions of complexes from facet lists, existing ones, orbit representations, difference cycles, discrete normal surfaces. Has a library of over 70000 constructions, and allows for user libraries. New in version 1.5 we support simplicial blowups. Has import/export to XML and `polymake`, Macaulay2 export, LaTeX export.

A polyhedral Morse function separates the odd vertices from the even ones, and lets us compute the level set. Can load the K^4 Kummer surface from library.

Q Given that you're living in GAP, do you use fundamental groups etc., e.g. for bistellar moves

A Not currently, though we could. These constructs can get quite large, so we'd rather use specific algorithms.

2.4 Computing the real solutions of polynomial systems with the `RegularChains` library in Maple — Moreno Maza *et al.*

Want to solve polynomial systems, including parametric and semi-algebraic ones. The key concept is a regular chain.

1. Heavy use of types
2. top level commands such as `Triangularize` etc., and tool kits such as `ConstructibleSetTools`.

Classical tools include isolating the real solutions, real root classification, cylindrical algebraic decomposition, realtriangularize, SamplePoints, set-theoretic tools on semi-algebraic sets.

Let T be a regular chain with algebraic variables y_i and parameters u_i, P a finite set of polynomials all regular w.r.t. .sat(T), and let Q be a quantifier-free

formula of $\mathbf{Q}[u]$. Then $[Q, T, P_>]$ is a regular system, i.e. $Q(u)$ is true, T is zero and the polynomials P are positive.

Examples: Eve surface, branch cuts $\sqrt{z-1}\sqrt{z+1} \stackrel{?}{=} \sqrt{z^2-1}$ and $\sqrt{1-z}\sqrt{z+1} \stackrel{?}{=} \sqrt{1-z^2}$ (done with `SamplePoints`), chemical equilibrium (ex-LeMaire: commands like `ConservationLaws`), linear hybrid systems with $\xi' = A\xi + Bu$ where $\xi(t)$ is the state of the system, and u is the control input. Note that reachability is basically questions of real quantifier elimination.

Q Is `CylindricalAlgebraicDecomposition` part of Maple?

A Yes

2.5 — Montes

[MontesWibmer2010] Defines Gröbner cover. Code written in, and can be downloaded from,Macaulay2. Consider $K[\mathbf{a}][\mathbf{x}]$ where $\mathbf{a}$ is parameters, $\mathbf{x}$ variables. Pick $>$ on $\mathbf{x}$ and define the Gröbner base w.r.t. $>$. How does this vary with $\mathbf{a}$. First studied in [Wei92]. We are interesting in speeding this up, and also improving the output: as few segments, canonical output etc.

Theorem 6 *If I is homogeneous, then there is a unique canonical Gröbner cover [Wibner2007] with the following properties*

1. *The s_i are locally closed, disjoint segments with different behaviour over the different segments*
2. *The B_i are a set of monic regular ideals*
3. *...*

For an inhomogeneous ideal, we can homogenize, compute and dehomogenize, but now different segments may have the same (finite) behaviour.

Conjecture 2 (Casas Alberó) *If a polynomial $x^n + \sum C_k^n a_k x^k$ has common roots with all of its derivatives (not necessarily the same) then in fact it must be $a(x+b)^n$.*

He can do this for degree 5: we get a Gröbner base of $\{1\}$ except over $\{a_{n-2} = a_{n-1}^2, a_{n-3} = a_{n-1}^3, \ldots\}$.

Theorem 7 (Classical Steiner–Lehmus) *The inner bisectors of angles A and B of a triangle ABC are equal iff the triangle is isosceles with $AB = AC$.*

This is more complicated because he can't distinguish inner bisectors from outer bisectors

Q How does it work — binary decomposition?

A No — very complicated. For example uses primary decomposition.

2.6 Mathemagix — Mourrain *et al.*

Motivation is the fact that there is no reusability of code outside the system, it is hard to integrate external code with the interpreter, and there are availability barriers. Also efficiency issues. Hence the aims of the project are

1. modularity — a coherent framework item efficiency
2. combination of algebraic and verified numeric computation
3. Provide tools for connectivity of packages to each other and to interpreters, e.g. our our Mathemagix, TEXmacs.

Hence it's a suite of C++ packages.We have a new high-level language, strongly typed. Started in 2002, first SYNAPS, and now in two INRIA-based ANR's. Based on GMP and MPFR. Has tools for analysis, including transseries. Also supports LAPACK above generic coefficients. We have subdivision solvers for polynomial systems. Demonstrated some graphics facilities.

As regards the type system, we have overloading, templates (such as `Ring`) a,d the ability to define, say, squaring for any rings.

2.7 Maple 15 presentation

I assume you know about Maple in general. The package is called `Magma` for working with arbitrary small finite magmas/groupoids. See also `RegularChains`, which has added a few new commands: `RealTriangularize` and `SamplePoints`. Related to this `Solve` has a `parametric` option, where the non-generic case is a stub for re-evaluation.

On the efficiency front, we have the `Grid` package. The previous `Threads` required substantial programming care. There's also a sparse solver for linear programs (> 20 variables). Further integration of the Monagan/Pierce `SDMP` package, getting an speedup of c. 7 (14 if modular) in appropriate cases.

Tick marks can now be labelled in multiples of π. Looking inside plots is now easier: `PlotTools[getdata]`. `DEplot3D` has been extended. There's a new repository for interactive worksheets, aimed largely at the teaching of pre-calculus topics. Extensions to ODEtools and PDEtools, with the latter including jet notation. Also a new on-line help system. A new finance package — Monte Carlo pricing et., and lots of smaller items.

Chapter 3

10 June 2011

3.1 — Kauers

The standard process for solving (finding rational solutions of) linear differential/difference equations:

1. Find denominator
2. ...

This talk is about the first step.

Example: given an equation, we can show that $n+k$ is not in denominator, for then it has to be in the numerator as well, but the same argument doesn't work fro, say, $n-k$. It depends on various cancellation effects. More formally:

Given $a_i, f \in K[N-1, \ldots, n_r]$

Find a "universal

For $r=1$, this as Abramov's algorithm. For $r>1$ this might not exist.

$$y(n+1,k) - y(n,k+1) = 0 \tag{3.1}$$

has solution 1, $\frac{1}{n+k}$, $\frac{1}{(n+k)^2}$ etc. and q is unbounded.

Definition 1 *The* spread *of two polynomials u, v is defined as*

$$\{(i_1, \ldots, i_r) \in \mathbf{Z}^r : \gcd(gcd(u)n_1 + i_1, \ldots, n_r + i_r), c(n_1, \ldots, n_r) \neq 1\}$$

Last year's result: for any given equation we can compute a polynomial $Q \neq 0$ .... The spread of an irreducible polynomial is a submodule W of $\mathbf{Z}^r$.

Theorem 8 *Consider the support S of a given equation. Then factors of spread W will show up in the coefficients of extremal (isolated) points of S.* Note that if there are more than one extremal points, this is very helpful.

Theorem 9 *In this case, if $y = p/q$ is a solution, and w_1, w_2 are factors of q with spread W, we can compute from the coefficients of the equation at the extremal points ....*

This is fine if W is known, but in general it is not. When W is not given, we loop over the spreads of irreducible factors of the corner points. Then this finds all W which are not parallel to an edge in the convex hull of S. These factors are the problem but this is a rare case (which can occur: see (3.1))

Last year, we had an algorithm which, for every equation. found some factors of the denominator: this year we can find *all* factors for *some* equations.

3.2 Univariate Real Root Isolation in an extension field — Strzeboński & Tsigaridas

Given a

$$B_\alpha = \sum b_i(\alpha)x^i = \sum c_{i,j}\alpha^j x^i$$

find the roots where α ranges over the roots of Although not our point of view, this can be regarded as a triangular systems problem.

3.2.1 Indirect Approaches

Take resultants to eliminate α [Joh91]. This is $O(N^10)$.

3.2.2 Direct Approaches

- Sturm sequences. This requires knowing the signs of the evaluations, which is a recursive sign determination problem. [Joh91], but with no complexity result. We claim $O(N^8)$ using HGCD. Nonetheless this is not practical.
- Increasing precision of approximations to α. Use QIR [Abb06].

  Theorem 10 $\ln \Delta B_\alpha = O(mn\tau)$ *using resultant formulation*

  Theorem 11 $\ln \Delta B_\alpha = \Omega(mn\tau)$ *from* $A = x^m - ax^{m-1} - 1$ *and ...*

  Hence the previous one is optimal.

Practically claims that, except when extension degree is very small, the increasing precision (modified Descartes) method is generally better

3.2.3 Generalisations

We can consider more (say l) algebraic numbers. Then get roughly $O(nm^{5l})$ for the indirect approach. We are working on bounds for the direct approaches.

Average complexity is a rather different question.

3.3 Efficient Real Root Approximation — Kerber & Sagraloff

Approximate roots to an absolute precision of 2^{-L}. We are looking at the refinement phase in this talk (see next one). Use a combination of interval secant method [Abb06]. We claim to be better by a factor of d.

(Roughly speaking) Exact evaluation of $f(r$ for an l-bit rational r requires a working precision of dl, but if $|f(r)|$ is sufficiently large, the *sign* of $f(r)$ requires less precision. This tends to need precision $d+l$. This is related to [Brent1993], but he assumed exact arithmetic.

3.3.1 Exact QIR — [Abb06]

Let N be the number of sub-intervals. Guess sub–interval based on second. On success, $N := N^2$, on failure $N := \sqrt{N}$, and use bisection for $N = 2$.

1. At most a constant time slower than bisection.
2. If the interval gets smaller than

$$M_\xi = \frac{f'(\xi)|}{8d^3 2^\tau \dots}$$

then it will always succeed.

In an interval world, our secant is now a thick line. We make it thin enough to be less than 1/4 of the sub-interval size.. This gives us m as the right evaluation point (in the exact model) or one of its neighbours. We would normally evaluate at m and 9depending on sign) one of its neighbours. What we actually do is take a (given) seven points, evaluate at them until 6 of the 7 have a precise sign, and use the given interval. We also needs approximate bisection and a normalization phase, and the result is a necessary precision of $O(d\tau + l)$, and then the analysis of the exact version carries over.

Let $R := \log(|\operatorname{Res}(f, f')|)^{-1}$. Then we need $\tilde{O}(d(d\tau_R)^2 + d^2 L)$ operations. R can in fact be replaced by a separation number, and this leads to topology computation of a real algebraic curve in $\tilde{O}(d^8\tau^2)$.

3.4 A simple but efficient algorithm for complex root isolation— Sagraloff & Yap

Focus on the complex case. Determine disjoint discs each containing exactly one root of f. We have a complete and certified subdivision algorithm Ceval. $\tilde{O}(n^4\tau^2)$ bit operations. We have a C++ implementation in Core. The asymptotically fast methods by Pan % Schonhäge are [Pan1997] "have many open problems".

Our previous real Eval uses a box predicate, and we use it recursively to check that f' has no roots in the interval (and therefore any root is simple). For

the complex case, we do the same with boxes, but a more sophisticated inclusion predicate (factor of K). A key point is played by the "8-point test": evaluate real and imaginary parts at x_0+ the 8 compass points (N,S,E,W,NE,NW.SE,SW) and look at the signs of the real and imaginary points.

We have a box-discarding test. Na‘"ıvely, this leaves $O(N^3)$ boxes remaining. We can do better. Consider the multiset R of all projections of roots on the real axis. Subdivide $\mathbf{R}$ such that, if $R_j := \mathbf{R} \cap I_j$, then $width(I_j) \leq 2\delta|R_j|$. Do the same for the imaginary part and look at the product. Then f $\delta = 4s(1 + \lceil n/2 \rceil)$, the boxes are at most $O((n + \log n)^2)$. area. This gives us a bound on the tree-width for not-yet-isolated roots. Using the Davenport–Mahler bound shows the size of the induced bounds on the width of the sub-division tree:

CEval $O((n \ln n)^2)(\tau + n) = \tilde{O}(n^2\tau)$

Eval $\tilde{O}(n\tau)$

3.5 Social Robotics — Maja Mataric: FCRC Plenary

[Founding director of UC Robotics Laboratory] [No videos since human participants were being shown]

We now have better machines, with smaller brains, than ever before. Microsoft's 3D vision is revolutionizing the sensor side as well. Note there are great socio-economic drivers as well. Hence importance of HRI (Human-Robotics Interaction).

I care about the care gap — need and unavailability of 1–1 care. Technology is going to have to step in. We used to have a population pyramid, but now it's a population cylinder. What we need are robots that

- capable of acting as a personalised care giver
- can interact with human experts as required
- can detect early signs of worsening/changing conditions
- are tireless and always attentive
- do not lose their temper
- etc.

Autism diagnosis is a social interaction, which is difficult to manage, and this leads to v great variation in autism diagnosis. Robotic diagnosis can provide *better* diagnosis. We are used to putting robots into physically dangerous situations: let's think about emotionally dangerous situations.

Our robots *don't* touch people (this is basically only safe in tightly controlled situations such as robot surgery). Everything I talk about is part of interdisciplinary collaboration.

Why a robot? Rôle of embodiment: embodied communication, proxemics, diaxis, body language, presence, personality, engagement and compliance. "We are wired as social animals". We are talking about more than machines *observing* human behaviour, but their behaviour should influence human behaviour. This involves understanding social interaction, and producing appropriate responses *in real time* How do we make this last? This is a great challenge for machine learning. The robot must track a moving target.

Behaviour (in general) supercedes appearance in general (about 3/4 human height is roughly right). Believability is more important than realism. A robot that looks like Einstein is actually a mistake. Safe affordable robots are going to have to be light and simple. All robots you'll see here are autonomous — no teleoperation. We do a lot of testing with real-world patients.

Methods include some social science, interdisciplinary participatory action research, lots of CS, even more robotics, and then evaluation. Note that people do tasks better even if a robot is merely present. Note that people actually bond to their robot vacuum cleaner, and want "their" robot back from repair "because it knows them", even though in fact there is no learning. Two studies show that people prefer interacting with robots to computes (one study healthy, one Alzheimer's).

Embodied communication is hard. user expectations are critical: if the robot speaks, it should understand. Voice has to match appearance perceptions. Voices must be unfamiliar (else cognitive dissonance). How does Ekman's facial coding apply to robots: robots tend to have very simple faces compared with humans? Synthetic speech is the way to go, but emotional synthetic speech is still an unsolved problem. Connecting speech and facial expressions is very difficult. What about non-verbal communication: gestures, proxemics. We have a lot to learn about human perception of machines. Animators have done a great deal of (informal)work on non-physical gestures such as the wind-up.

We need to understand affinity and affect. Vision is great, but not good enough. Extra cameras raise privacy concerns (85% of Skype users don't use the video option when available). Wearable sensors can compensate. They are inherently more private, and must not affect behaviour or social interactions.

We can collect physiologic data, and Galvanic Skin Response can be used to predict (Support Vector Machine technology) user frustration (or at least the results of it). This did much better than humans watching facial expressions.

Personality (human and robot) plays a key rôle in human–robot interactions. Goal: model personality quantitatively. For example, extroverts tend to talk more and faster. People did better when the robot supervisor's personality matched theirs. But "learning" doesn't work since the robot can't effectively change personality drastically (confuses human) — the key is fine-tuning.

3.5.1 Robots in the wild

Autism Spectrum Disorders (ASD). 1 in 100 children are diagnosed today. We want "robots as persons". While a few children refuse to play with robots, most do play, and demonstrate social behaviours with robots that they don't with

humans. "Now I know how my teacher feels". How one moves in the presence of others is very indicative of social comfort.

Stroke rehabilitation is another area — most sufferers are left with permanent deficits due to a lack of long-term supervised rehabilitation. This is also true of some other conditions. Continual motivation has been shown to be critical for long-term recovery. If the right hand is affected, using the left is *not* the right solution, but is unfortunately "natural". Hence long-term surveillance (robots don't get tired) is the answer.

6-month study: Music Therapist for Cognitive Attention Training. The cognitive game is "name that tune", with some/few/no hints. The learning approach is to learn a user-specific performance best-worst band, then adjust the challenge level of the game incrementally to move then band up to a higher challenge level. Surprisingly, people in their 80s with Alzheimer's *do* engage productively with robots.

With non-Alzheimer's elderly patients, used "robot coaches" for chair aerobics (especially doing an imitation game). It's aso good for the human to take change and "show the robot what to do" — people being in charge[1] tend to live longer. The game is also better combined with a memory game. In all aspects, a robot beats a computer coach.

3.5.2 Major research challenges

1. Why a robot?
2. Making friends and influencing people
3. Will it last?

Robots and social integrators (autistic and other special-needs in the classroom).

3.6 Computational Aspects of Elliptic Curves and Modular Forms — Miller: ISSAC Plenary

Joined SIGSAM as an undergraduate in 1967.

There were probably only a couple of hundred people in the world who knew about elliptic curves when Koblitz and I invented ECC.

> It is possible to write endlessly about elliptic curves — this is not a threat. [Serge Lang]

Elliptic curves are a special case of Diophantine equations. This is a much harder problem than over $\mathbf{R}$ or $\mathbf{C}$. One of the attractions of number theory is that it is very easy to pose very hard problems. The general method for Diophantine equations is a procedure that is not known to terminate.

[1] Even if it's a plant, or a pet.

A major issue is the *rank* of the curve. This is the subject of Birch–Swinnerton-Dyer conjecture — a Cray Millennium $1M prize. The naïve thing to look at is the degree of the polynomial, but in fact the genus is a better invariant. Fermat also wrote in his Diophantus that $y^2 = x^3 - 2$ has infinite number of rational solutions. Bachet actually had the duplication formula, and it was only in the 20th century that Fueter, and later Mordell, really proved this. In 1901, Poincaré observed that all the solutions should be obtained from a finite number by tangent/chord. Mordell–Weil theorem. Hence we have an Abelian group. Mazur's Theorem described all possible torsion components over $\mathbf{Q}$.

Note the Hasse–Minkowski Theorem, and the Hasse Principle. Is it enough to check for real solutions, and solutions modulo p (in principle for all p, but in practice only finitely many). This fails for elliptic curves — Selmer: $3x^3 + 4y^3 + 5z^3 = 0$. The discriminant is one measure of how big a curve is, but the conductor (same primes, but exponents 1 or 2, except for 2/3) is actually a better invariant.

The normal way of proving the Mordell–Weil theorem is first to prove the Weak Mordell–Weil theorem, $E(\mathbf{Q})/2E(\mathbf{Q})$ is finite. This plus naïve height is sufficient to prove the Mordell–Weil theorem. In fact we can produce Neron–Tait height $= \lim_{n\to\infty} h(2^n P)/4^n$ really is a quadratic form. Neron could compute this via a local decomposition: the p-components are easy and Silverman solved $\mathbf{R}$. This involved 2-coverings. It was known that there were only a finite number of equivalent classes of 2-coverings, and Birch–Swinnerton-Dyer actually did this.

> Elliptic curves were in fashion, but somewhat mysterious. The rank was considered known only by God. We needed to do a 2-descent, but *not* by hand. — Swinnerton-Dyer

Conjecture 3 *Guessed that* $\prod_{p\le x} \frac{n_p}{p} \approx \log^r(x)$.

Theorem 12 (Hasse) *If F is finite of size q then* $|N_F - (q+1)| \le 2\sqrt{q}$.

Hence Hasse–Weil L-function. This converged for $\Re(s) > 3/2$ by the above, but in fact has analytic continuation. They conjecture that $r =$ order of zero at $s = 1$ (weak conjecture). The strong conjecture tells us what the appropriate coefficient of the power series:

$$\frac{L_E^{(r)}(1)}{r!} = \dots$$

Note that in the 18th century, elliptic functions were all the rage. They dropped out around 1910, but are still very relevant. The Hasse–Weil L-function is an example of a Dirichlet series. A level-N weight k functional equation is

$$\Phi(s) = \left(\frac{\sqrt{N}}{\dots}\right)^k \Phi(1-s).$$

Hecke showed that if F is a Dirichlet series with some analytic hypotheses with a functional equation N/k, then (at least for N=1,2,3,4) it comes from a modular form. Taniyama–Shimura conjectured that every elliptic curve came from a modular forms. Heegner (1958) had an explicit construction (but only recognised by Birch 1973) that would give you points on appropriate curves. Cremona checked Birch–Swinnerton-Dyer for $n \leq 120,000$.

Frey, based on Serre/Ribet, had shown that a counter-example to Fermat gave rise to a non-modular elliptic curve. Hence Wiles–Taylor's proof of Taniyama–Shimura proves Fermat's last theorem.

3.7 An Automatic Parallelization Framework for Algebraic Computation Systems — Dos Reis & Li

How can CAS benefit from computational power (e.g. concurrency) available in modern computer systems, even desktops? PhD students are one solution, but we ought to automate the knowledge/craft. This may need a paradigm shift, but this community should not be frightened of that. Recent advances in hardware have put the Programming language and Compiler communities into reactive mode. Hence we ask "Can CAS bring anything to Programming Languages here?" — yes, we can.

We should combine the following.

1. A simple idea: it is hard for a well-structured algebraic algorithm not to reflect — in one way or anther – properties related to the entities it manipulates.

2. An effective technology: semantics-based static analysis. A programming language construct: powerful semantic properties (note that we have had `assume` in Maple for 20 years).

Converting sequential `gcd(list)` into a `paraelReduce` construct: needs to know that gcd is a monoid operator[2]. This is far from being the only case. Compilers know about these properties, but only for built-in types. [LiDosReis2010] showed that there were a lot of such example in Axiom.

Q Does Axiom have pure/impure division?

A No, but there is enough static analysis to buy the equivalent.

[2] JHD: I assume it's the associativity that's being relied on. What about absorbing elements and "early abort"?

3.8 Detecting genus in vertex links for the fast enumeration of 3-manifold triangulations — Barton

2-manifolds are common, 4-manifolds undecidable, and we are looking at 3-manifolds. These a re not simplicial complexes since vertices are identified. These are recent algorithms. Algorithms are (worst-case) exponential in $n = \#$ vertices. Hence shrinking is useful.

Note that not all gluings will yield legal 3-manifolds (impossible for 2-manifolds). This occurs when the neighbourhood of a vertex is not $\mathbf{R}^d$. The probability tends to 1 [DunfieldThurston2006], but is easy to detect by Euler characteristic. This are worse for 4-manifolds, and undecidable for 5-manifolds. Note that for $d = 9$ 3-manifolds there are 10^{30} triangularizations, 10^{12} up to isomorphisms, but only 10^8 legal ones. While gluing we should never join two faces from the same object but different boundary cycles, as this creates a handle. How do we enforce this?

Use "skip lists" [Pugh1990]. In particular all operations are expected $O(\log n)$ time. In Regina, $n = 7$, we took 103 hours compute time down to 48 minutes (non-orientable case). This is producing new evidence of worst/average case complexity performance for decision algorithms.

3.9 Quadratic-Time Certificates in Linear Algebra — Kaltofen, Nehring & Saunders

A certificate [KLYZ08] is an input-dependent data structure and an algorithm that verifies, using those data, that the result of the computation is correct. Warm up: Rusin Frievald's 1979 matrix product certificate. Certify $C = A \cdot B$ by computing $C \cdot \mathbf{y}$ and $A \cdot (B \cdot \mathbf{y})$ for a random $\mathbf{y}$, and Schwartz–Zippel bounds probability of accepting a wrong C.

How do we certify rank? Illustration: suppose there are undetectable lies (bounded number k), detectable lies and truth, then we take a ball at random from a box with $5k$ (say) balls, reject detectable lies. So what we do is take $5n^{1+o(1)}$ smallish primes, and ....

Use Artin's solution to Hilbert 17th: that a semidefinite polynomial is a quotient of sums of squares, which can be used to certify that matrices are semi-definite. Encode sufficiently many similarity transforms to represent the characteristic polynomial.

For Frobenius forms we also use the fact that a given form only has finitely many bad ....

If the rank is at most $n^{2/\omega+o(1)}$ we can certify it in $n^{2+o(1)}$ bit complexity.

1. Run [Storjohann2009]'s Las Vegas
2. Record all random choices and intermediate results except in matrix multiplications

3. For the multiplications, verify as at start of talk.

3.10 Computing Hermite forms of polynomial matrices — Gupta & Storjohann

Matrices in $K[x]^{n\times n}$. Cost in terms of K-operations, with polynomial multiplication $M(d)$, gcd $G(d)$ and matrix multiplication $MM(n,d)$ ([BostanSchost2005] for a better way than $n^\omega M(d)$ over large finite fields). We have a Las Vegas algorithm which is $\tilde{O}(n^\omega d)$. Note that this is known for Smith form [Sto02], so why the difference? Probably the greater rate of degree drop in Smith.

The naïve algorithm doesn't scale correctly, and also suffers from intermediate expression swell. We therefore compute only the diagonal entries of the Hermite normal form. [BLV99] would give us $\tilde{O}(n^{\omega+1}d)$. [Storjohann2006] in fact gives us $O(n^2G(nd))$.

1. Row reduce A
2. Find a Smith form S and transform V
3. Find the degrees of the diagonal entries of the Hermite form
4. Compute the full Hermite form by fast minimal approximant basis computation [GJV03].

3.11 Symbolic-Numeric exact rational linear system solution — Saunders, Wood & Youse

Square matrices, using floating point. One option is [Dix82], which is p-adic. We also have [Wan2006], going to twice the bit-length of the Hadamard bound in the (dyadic) reals, and then finding rationals.

1. Ask a numeric solver for an approximate solution (iteratively).
2. scale this so the "useful bits" are integer bits
3. use this to update dyadic solution

Our improvement from [Wan2006] is to get more data from the numeric solution.

When reconstructing, we stop when the error

$$\left|\frac{a}{b}-\frac{n}{d}\right| < \frac{1}{2d}. \tag{3.2}$$

In some cases, we can do early termination, provided (3.2) is satisfied [Steffy2010]. Note that we know there's a common numerator, so we can use this to test for false reconstructions.

We believe that this technique could also work with sparse solvers.

3.12 Normalization of row-reduced polynomial matrices — Sarkar & Storjohann

This crops up in the minimal approximate basis computation: find the row-reduced basis M of all vectors V such that $vG \equiv 0 \pmod{x^n}$. Same cost model as section 3.10.

[GJV03] gave $\tilde{O}(n^2\sigma)$ for order σ for nearly square matrices and used this for a Las Vegas algorithm. [BLV99] used minimal approximant bases. Define the pivot element to be (the?) element of maximal degree in its row. For a Popov form, want the pivot elements all to be in different columns, and to dominate their column. [Mulders2000] gave an algorithm for this. $\begin{pmatrix} -x^{d-1}I & B \\ x^{d+1}A & I \end{pmatrix}$, whose Popov form contains AB, shows that this can't be easier than matrix multiplication.

We use an LUP decomposition, where P is a permutation matrix. We actually choose the right-most element as the pivot.

1. LUP transforms R to weak Popov form W
2. W is scaled to WX and row reduction gives us R_1 where all elements in a row have the same degree.
3. We have a special case solver which can be applied for Popov form.

This is $\tilde{O}(n^3d)$ (even $\tilde{O}(n^\omega d)$), and also deterministic.

Arbitrary rectangular matrices are an open question.

Chapter 4

11 June 2011

4.1 Border basis detection is NP-Complete — Anath & Dukkipati

To compute $F[\mathbf{x}]/I$, we can look at all terms not reducible by a Gröbner base. Conversely, given

Border bases are *nice* set of generators of an ideal — not necessarily associated to a term ordering, but it is associated to an order ideal.

Definition 2 *An order ideal is a finite collection of monomials if it is closed under division.*

Definition 3 *The border of an order ideal $\mathcal{O}$ is*

$$\partial\mathcal{O} := \{x_i m : m \in \mathcal{O}\} \setminus \mathcal{O}.$$

The study of border bases goes back to [AuzingerStetter1988]. The Gröbner basis detection problem for $S \subset F[\mathbf{x}]$ is to find a term ordering with respect to which this (finite) set is a Gröbner basis. Proposed by [GritzmannSturmdels1993]. What about border basis detection (BBD)?

Given a finite set $\mathcal{F}$ of polynomials, find a set $\mathcal{O}$ which is an order ideal and $\mathcal{F}$ is a border basis for it.

We say that the children of t, $\mathrm{ch}(t) = \{t' : \exists i x_i t' x_i = t\}$. Hence concept of parent and sibling.

Theorem 13 *BBD is in NP.*

The construction can be verified in polynomial time.

Theorem 14 *BBD is NP-hard.*

We reduce from 3,4 SAT: three variables per clause and each variable (or its complement) occurs in no more than four clauses. Consider

$$F[x_1, \ldots, x_n, y_1, \ldots, y_n, c_1, \ldots, c_m, x_{c_1}, \ldots, x_{c_m}].$$

To X_i in the 3,4SAT associate .... The fact that each clause in the 3,4SAT is either true or false is equivalent to each corresponding polynomial being in either $\mathcal{O}$ or $\partial\mathcal{O}$, and this is the key of the reduction. Intuitively, if X_i occurs in clause C_l then include in F_2 all the siblings of t_{X_i} and $t_{X_i}x)c_l/c_l$ in the set F_2. F_1 is the set of terms of degree exactly 8. Then consider $F = F_1 \cup F_2 \cup \cdots$, and

4.2 Space-efficient Gröbner-base Computation without Degree Bounds — Mayr & Richter

Gröbner bases are hard to compute [MM82]: more precisely exp-space hard. Let $<$ be an admissible ordering represented by a matrix $W \in \mathbf{Q}^{n,n}$: note that this is always true for $\mathbf{R}^{n,n}$, but is a slight restriction in our case. G is a Gröbner basis of I iff $\langle \mathrm{lm}(G)\rangle = \langle \mathrm{lm}(I)\rangle$.

Theorem 15 ([]KuhnleMayr1996) *The reduced GB can be computed in exponential space in the input size (more precisely, exponential in the number of variables).*

Proof idea. $g \in G$ is equivalent to $0 \neq g = h - \mathrm{nf}_I(h)$ and h with respect to division is a minimal monomial. Use bounds from [Dub90]:$\deg(g) \leq d^{2^n}$ and [Her26] we can write $g \in I$ explicitly as $g = \sum a_n g_i$ with $\deg a_i \leq \deg g + (sd)^{2^n}$.

So now suppose G is a Gröbner base of $(f_1, \ldots, f)s)$ with $\deg f_i \leq d$. Use [BCP83] to compute rank and adjoint matrix (and, if square, determinant and characteristic polynomial) of a matrix over $\mathbf{Q}$ in space $O(\log^2(qn))$.

Theorem 16 (This paper) $O(n^8 2^{4r} \log^2(sdq))$ *with s polynomials with q-bit coefficients and an ideal of dimension r.*

1. Compute minimally reducible monomials w.r.t some degree bound D
2. Verify completeness of this Gröbner basis with S-polynomials
3. if necessary, increase D (reusing existing space)

Theorem 17 (This paper) $O(\log^2(sD^n q))$, *where D is the degree of the representation of the S-polynomials, i.e. $D = d^{n^r}$.*

We have identified the ideal dimension r as the key reason for the exponential space complexity. We believe this adaptive algorithm has better average case behaviour (to be studied), and the new degree bounds easily translate to new complexity results.

4.3 Algorithms for Computing Triangular Decompositions of Polynomial Systems — Chen & Moreno Maza

While linear systems have a single triangular systems, this is not true for non-linear sets: shows example. Triangular Decompositions [MM99] have many

applications. Two main definitions of "solution":

1. Generic zeros only (Kalkbrener)
2. encode all zeros of the system (Lazard–Wu).

In this paper, we try to save on the algebraic complexity and to extract the common parts of computations.

4.3.1 Regular Chains

Quasi-component $W(T) = V(T) \setminus V(\text{init}(T))$. We try to compute this incrementally

4.3.2 Regular GCD

g is a regular gcd of $p, t \in A[y]$ if g is regular w.r.t. $\langle t \rangle$ and if $\deg(g, y) > 0$ then $\text{prem}(p, g) = \text{prem}(t, g) = 0$ over $A[y]$: coincides with normal gcd if A is a field. These can be computed via subresultants. Let T be a regular chain, and $A = R/\sqrt{\text{sat}(T)}$.

Theorem 18 $V(p) \cap W(T \cup t) \subseteq V(p) \cap \overline{W(T \cup t)}$

Note that the computation of $A = R/\sqrt{\text{sat}(T)}$ can be reused.

4.3.3 Intersections

Example of the trivariate case.
$r2 := \text{Res}(p_3, t_3, z)$; $r_1 := \text{Res}(r_2, t_2, y)$ and then $V(p_3) \cap W(\ldots)$.

4.3.4 Kalkbrener decomposition

Let L be a Kalkbrener decomposition of $V(F)$. Let T be a regular chain of L the height of which is greater than $\#(F)$. Then $L \setminus \{T\}$ is also a Kalkbrener decomposition — this is efficient in practice.

Showed timings from Maple 13, and of this solver against others in Maple.

By computing regular gcd in a weaker way we have a more efficient algorithm.

4.4 Computing with semi-algebraic sets represented by triangular decompositions — Xiao *et al.*

Much related work on algebraic systems, and CAD for semi-algebraic systems. Our goals:

1. investigate the geometrically-intrinsic elements

2. improve runtime

3. implement set-theoretic operations on semi-algebraic sets.

Note that we have quantifier-free logical formulae Q whose truth ensures that the corresponding regular system has solutions. We compute an RSAS from a PRSAS $B \not=, T, P_>]$ where $\forall u \in B \not=$, $[T, P_>]$ specialises well.

Definition 4 *The border polynomial set is* $\mathrm{Res}(f, T)$ *for all* f *in* $P \sup\{\frac{\partial t}{\partial \operatorname{mvar}(t)} : t \in P\}$.

Theorem 19 *If* C *is a connected component of* $Z_{\mathbf{R}}(B \not=)$*, the number of real solutions is constant over* C.

Definition 5 *Fingerprint polynomial set (FPS)* D*:* $\alpha \in Z_{\mathbf{R}}(D \not=) \Rightarrow b(a) \neq 0$ *and if the signs of all polynomials in* D *are the same at* α *and* β*, then the number of zeros of the system is the same at* α *and* β.

In theory the open augmented projection of b is a FPS.

In this work we produce
em minimal border polynomial sets in some circumstances. The *effective boundary* is an invariant of the parametric system, and this improves the generation of the FPS.

There is a relaxation technique which reduces the number of recursive calls.

4.4.1 Border Polynomials

These are the gateway to the "real" world. These are algorithmic, and not intrinsic to the system.

We say that T is canonical iff ...

Theorem 20 *For any regular chain* T *there is a canonical regular chain* T^* *with the same* sat.

In practice they are more expensive to compute, so we currently don't use them algorithmically.

The effective boundary B is a hypersurface defined by an irreducible polynomial such that there is an open ball split in two by B with the number of zeros the same in the same component *and* different between components.

4.4.2 Relaxation

There is a criterion for when we can apply relaxation. We need to test $Z_{\mathbf{R}}(\widetilde{Q_1}^h) = Z_{\mathbf{R}}(\widetilde{Q_0}^h)$

4.5 Recent Progress in Linear Algebra and Lattice basis Reduction — Villard

Consider only the simplest case of lattice reduction here — unimodular column operations. We want to get "smaller norms".

We are concerned with examples of huge bit-size $\beta = \max \log |b_{i,j}|$ and want to be linear with respect to this, as long as we are polynomial in dimension d.

Linear algebra has been one of the great successes of computer algebra — look at today's Storjohann talks. But can we compute the characteristic polynomial efficiently? Open problem.

We can compute minimal approximant bases in $O(MM(n,d))$.

What about polynomial basis reduction: $A(x)U(x) = R(x)$? First we translate into the "regular case" when U has low degree. Then there are a series of elementary steps.

Can we do the same for integer case. We need an elementary step, and a realistic estimate of its cost.

The way we see that a matrix is not-LLL is to look at the Gram–Schmidt orthogonalization. We then round the orthogonalization operations to give us an exact integer operation. We occasionally swap rows, as in Euclid. Every swap decreases the norm by a constant factor.

If we approximate the amount of reduction, when we don't necessarily observe LLL-reduction exactly as in the original paper. Hence we make a technical change to the definition of reduction: $|r_{i,j}|ale\eta||b_i^*|| + \cdots$. WE also use different QR algorithms at different phases in the reduction. The number of steps required is $O(\beta)$, but we lost polynomiality in d.

4.5.1 Toolbox

1. Elementary step: "from reduced to reduced". We make the lattice seem reduced by scaling the errant row. Then the step is to (partially) reverse this, and then do a Euclidean step.

2. The most significant bits of B suffice. The progress, if we lift σ^l, is ....

3. Low-cost unimodular transformation: the matrix we obtain from a truncated step 2 is 'close enough" to the true one, and satisfies our modified definition of reduction.

The Lehmer lift reduction takes a B which is Ξ-reduced, truncate (and therefore only Ξ'-reduced) then get a $\sigma^l BU$, which again is Ξ-reduced. We let $l \approx \sqrt{\beta}$. With 40K bits he is 5 times faster than NTL, at 200K bits he is 51 times faster. He observes that the reduction of the Gram-Schmidt lengths behaves differently between this and basic LLL.

We can also consider a recursive lifting tree. The HGCD people talk about "repairing" the approximate results, but he'd rather talk about "strengthening" the quality Ξ, apparently both before and after the truncation. He calls this

implementation $\widetilde{L}^1$. Very impressive graphs, but he notes that these examples have very small d compared with the bit size (e.g. $d = 10, 40$ but 100K bits).

Open Questions. How does this compare with [Schnorr2011], [HarrotPujolStehle2010]? What about right–left reductions? Can we get this cost down to matrix multiplication?

4.6 Computing Comprehensive Gröbner Systems and Comprehensive Gröbner Bases simultaneously — Wang

[Wei92] introduced both CGS and CGB, and gave an algorithm for both. See also [Kapur1995]. Note that these are independent questions, but some algorithms can do both. There are (expensive) transformations in both directions. Hence this paper is a new departure by proposing to compute both. Furthermore, *any* algorithm for CGS can use this idea to compute CGB.

Consider $\{ax^2+yb^2 = 0, ax^2 = y^2 = 0, ax-cy = 0\}$. Different specializations give different results, e g. $a = 1, b = 2, c = 3$ gives one solution, and others give others. A CGS is a list of pairs (constructible set; polynomial system). But note that some of these polynomial systems are not actually in the original ideal, since they only apply under certain conditions. If, however, they all are, then the union of the polynomial sets in a CGS is a CGB.

Our new technique consists of splitting a polynomial into two parts: a zero part and a non-zero part. $M(F, A) = \{(p, \tilde{p}) : p+\tilde{p} \in \langle F\rangle > \wedge\sigma_a(p) = 0 : a \in A\}$ is then a module. This then produces a CGS, and the CGB is the union of all the $p + \tilde{p}$ over all the branches.

Q Do you know any applications of CGB (as opposed to CGS)?

A I know of many CGS, but not of CGB.

Q Weispfenning was looking for a *canonical* CGB, but I doubt that such a thing exists.

A

4.7 Computing a Structured Gröbner Basis — Nagasaka

We normally want to delete redundant relations or simplify relations. But this isn't so obvious in floating point, and less so if we have *a priori* errors. Our approach to extract hidden information is to change the coefficients continuously. The real goal is to find small ideals close to the calculated one — perturbation makes the ideals later.

Idea is therefore a "structured polynomial set", i.e. a mapping from $\mathbf{C}^n$ into a set of polynomials, explaining how the coefficients change. This problem

is essentially one of Structured Low Rank Approximation, which is solved by SVD, replacing small values with exact zeroes.

Definition 6 *We say that G is a structured Gröbner basis with rank deficiency d, set of terms T and tolerance ϵ*

1. *G is a Gröbner basis for the ideal generated by F_{st}.*
2. *F and F_{st} are structured polynomial sets, with parameters p and p_{st}.*
3. *p and p_{st} are "close": to within ϵ.*
4. *T has all the terms required for computing Gröbner basis for F and F_{st}.*
5. $\text{rank}(M_T(F_{st})) - \text{rank}(M_T(F)) - d$.

We have an algorithm to compute this: a combination of F_4 and SVD. Example given, but it shows that the condition in Lemma 4 is not always sufficient.

Q Does this work for non-zero dimensional ideals?

A Yes, but it is more difficult.

4.8 A Generalized Criterion for Signature-Related Gröbner Bases — Sun

There are various signature related algorithms such as F_5, [EP09] [HA10]. Our contribution is a generalized criterion which can be specialized to existing ones, such as F_5.

Our example is $yz - x, xz - y; xy - z$. We have a Gröbner base iff the lm matching happens. We get new lm by reducing $F \in I$. Can we predict whether f is "worth reducing"? We order the polynomials in I according to their signatures, in order to "get beautiful properties". Consider $f := y^2 - z^2 = 0 \cdot f_1 - y \cdot f_2 + z \cdot f_3$, represented by $(0, -y, z)$. Since the f_i are fixed, we can regard $(0, -y, z)$ as a *representation* of f.

4.9 Signature-based algorithms to Compute Gröbner Bases — Perry & Eder

Aim to understand F_5 and others. We want a common framework, base d which we can compare algorithms. We also want to detect useless *polynomials*, and hence prove termination for some of these. This is not a new algorithm *per se*.

We have an existing Gröbner basis, and append f_{i+1} to it.

Buchberger We add new pairs, choose the "minimal" one, reduce and continue. Noetherianity gives termination. Almost all the time we are verifying the Gröbner base (getting an S-polynomial that reduces to 0), rather than actually computing it.

Signatures Now regard $p \in \langle f_1, \ldots f_i \rangle$ as $p = \sum b_k f_k$, and we are interested in $\mathrm{lm}(b_i)$. We get a well-ordering on the set of monic signatures.

We can therefore talk about signature-preserving sig-reduction. Note that there are two ways of doing this.

1. the leading term is smaller
2. terms the same but coefficients different

We say that a polynomial is sig-complete if no sig-preserving reductions are possible. Note that, for every signature, we can designate a polynomial that will generate it. He therefore has a criterion that says "we would generate a signature via other than the designated polynomial". .

We say that $(\sigma F_{i+1}, r)$ is *signature-redundant* iff .... We can map polynomials with signatures into monomials n twice as many variables (one for polynomial and one for signatures), and then Noetherianity *here* can be used to prove termination. He has an example where F_5 is "too aggressive" — throws away too much, and therefore has to do more computation later.

Note that [**?**] actually does complete reduction (though this isn't clear from their paper). This *may* be useful. We think that [**?**] is 1.5–2 times *slower* than F_5, whereas they seemed to report it was faster. We think this is because they can compute polynomials, reduce them, and then discard them, without counting them. In terms of implementations in Singular, [**?**] were relying on more compiled Singular than the F_5 implementation.

Q You said that complete reduction might be useful: is this also true for non-homogeneous ideals?

A Nothing we have said assumes homogeneity.

Bibliography

[Abb06] J.A. Abbott. Quadratic Interval Refinement for Real Roots. *Poster at ISSAC 2006*, 2006.

[BCP83] A. Borodin, S. Cook, and N. Pippenger. Parallel Computation for Well-endowed Rings and Space-bounded Probabilistic Machines. *Information and Control*, 58:113–136, 1983.

[BLV99] B. Beckermann, G. Labahn, and G. Villard. Shifted Normal Forms of Polynomial Matrices. In S. Dooley, editor, *Proceedings ISSAC '99*, pages 189–196, 1999.

[BMMT94] E. Becker, M.G. Marinari, T. Mora, and C. Traverso. The shape of the shape lemma. In *Proceedings ISSAC 1994*, pages 129–133, 1994.

[CF97] G. Carrà Ferro. A Resultant Theory for the Systems of Two Ordinary Differential Equations. *AAECC*, 8:539–560, 1997.

[Dix82] J.D. Dixon. Exact Solutions of Linear Equations Using p-adic Methods. *Numer. Math.*, 40:137–141, 1982.

[Dub90] T.W. Dubé. The structure of polynomial ideals and Gröbner Bases. *SIAM J. Comp.*, 19:750–753, 1990.

[EP09] C. Eder and J. Perry. F5C: a variant of Faugere's F5 algorithm with reduced Groebner bases. `http://arxiv.org/abs/0906.2967`, 2009.

[FGLM93] J.C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering. *J. Symbolic Comp.*, 16:329–344, 1993.

[GJV03] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the Complexity of Polynomial Matrix Computations. In J.R. Sendra, editor, *Proceedings ISSAC 2003*, pages 135–142, 2003.

[HA10] A. Hashemi and G. Ars. Extended F5 criteria. *J. Symbolic Comp.*, 45:1330–1340, 2010.

[Her26] G. Hermann. Die Frage der Endlich vielen Schritte in der Theorie der Polynomideale. *Math. Ann.*, 95:736–788, 1926.

[Joh91] J.R. Johnson. *Algorithms for Polynomial Real Root Isolation.* PhD thesis, Ohio State University, 1991.

[KLYZ08] E. Kaltofen, B. Li, Z. Yang, and L. Zhi. Exact Certification of Global Optimality of Approximate Factorizations Via Rationalizing Sums-Of-Squares with Floating Point Scalars. In D.J.Jeffrey, editor, *Proceedings ISSAC 2008*, pages 155–164, 2008.

[Mac12] F.S. Macauley. The Algebraic Theory of Modular Systems. *Cambridge University Press*, 1912.

[MM82] E. Mayr and A. Mayer. The Complexity of the Word Problem for Commutative Semi-groups and Polynomial Ideals. *Adv. in Math.*, 46:305–329, 1982.

[MM99] M. Moreno Maza. On Triangular Decompositions of Algebraic Varieties. Technical Report TR 4/99, 1999.

[Rit32] J.F. Ritt. Differential Equations from an Algebraic Standpoint. *Volume 14*, 1932.

[Rit50] J.F. Ritt. Differential Algebra. *American Mathematical Society Colloquium Proceedings vol. XXXIII*, 1950.

[Sak88] S. Sakata. Finding a Minimal Set of Linear Recurring Relations Capable of Generating a Given Finite Two-dimensional Array. *J. Symbolic Comp. pp.*, 5:321–337, 1988.

[Sto02] A. Storjohann. High-Order Lifting. In T. Mora, editor, *Proceedings ISSAC 2002*, pages 246–254, 2002.

[Stu93] B. Sturmfels. Sparse Elimination Theory. In D. Eisenbud and L. Robbiano, editors, *Proceedings Computational Alg. Geom. and Comm. Alg.*, pages 377–396, 1993.

[Wei92] V. Weispfenning. Comprehensive Gröbner Bases. *J. Symbolic Comp.*, 14:1–29, 1992.