# Computer Algebra and Formal Proof

James Davenport[1]
University of Bath
J.H.Davenport@bath.ac.uk

21 July 2017

[1]Thanks to EU H2020-FETOPEN-2016-2017-CSA project $\mathcal{SC}^2$ (712689) and the Isaac Newton Institute through EPSRC K/032208/1

# Computer Algebra Systems

- Large multi-author multi-decade systems
- Often written in a kernel+library approach, superficially similar to theorem-provers, but in practice the kernel isn't formally specified, and the libraries are where the semantics live, and aren't verified
- The semantics are often variable, informal, and indeed changing

e.g. "Now integrates more definite integrals in terms of Meijer $G$-functions"

- Intended for human consumption

Therefore can't be imported into a theorem-prover as proven lemmas.

# Does this mean the two fields can't talk?

Not at all, and the fundamental reason is that it is generally easier to verify a result than to derive it.
Excellent discussion in "A Sceptics Approach" [HT98].
However, the precise nature of the co-operation will depend critically on the nature of the computation being considered: not "one size fits all".

# Greatest Common Divisors (of polynomials)

"$g$ is the greatest common divisor of $f_1$, $f_2$ [and more]" is actually two assertions:

1. $g$ divides $f_1$, $f_2$ (implicitly over $\mathbf{Z}[x_1, \ldots, x_n]$);
2. Any $h$ that also divides them divides $g$.

Note that $g$ is not unique: $-g$ would do as well. CA systems enforce uniqueness by making the leading coefficient positive, but this then depends on the definition of "leading". *If* this matters, there's going to be a tricky communication over the meaning of "leading".

# Verifying Greatest Common Divisors

1a) Verify that $g$ divides $f_1$, $f_2$. *Or*

1b) ask the system for $h_1 = f_1/g$ etc. and verify that $f_1 = gh_1$ etc.

* The second is probably easier.

2a) The system will have computed $p, v_2, \ldots, v_n$ such that $h_i(v_2, \ldots, v_n) \pmod p$ are relatively prime and have the same degree as the original $h_i$.

TP Euclid in one variable (I probably wouldn't bother with the (mod $p$) part), and a one-off theorem

CA should provide a means of telling you $p, v_2, \ldots, v_n$ (they currently don't)

2b) "Ask for the Bézout coefficients" [HT98].

# "Ask for the Bézout coefficients"

- Easy enough in one variable
  - Given $f_1$, $f_2 \in \mathbf{Z}[x]$ ask CA for $F_1$, $F_2 \in \mathbf{Q}[x]$ such that $F_1 f_1 + F_2 f_2 = g$
  - This plus 1) shows $g$ is a gcd (up to integer factors)
- In $n$ variables it's harder: $\forall i \in [1, \ldots, n]$ needs
  - $F_1^{(i)}$, $F_2^{(i)} \in \mathbf{Q}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)[x_i]$ such that $F_1^{(i)} f_1 + F_2^{(i)} f_2 = g$
  - This plus 1) shows $g$ is a gcd (up to *integer* factors)
  - If we didn't care about contents, it's easier.

# Factorisation (of polynomials)

"$g$ factors as $f_1 \cdot f_2 \cdots f_m$" is actually two assertions:

1. $g = f_1 \cdot f_2 \cdots f_m$;
2. The $f_i$ are irreducible (implicitly over $\mathbf{Z}[x_1, \ldots, x_n]$).

Note that the $f_i$ are not unique: $-f_i$ would do as well. CA systems enforce uniqueness by making the leading coefficient positive for $f_2, \ldots f_m$, and put all the content in $f_1$, but this then depends on the definition of "leading". *If* this matters, there's going to be a tricky communication over the meaning of "leading", and the order of the $f_i$.

# Verifying Factorisation

1a) Verify that $g = f_1 \cdot f_2 \cdots f_m$;

2) Depending on $f$ (and on the implementation)

2(i) The system will have computed $p, v_2, \ldots, v_n$ such that $f_i(v_2, \ldots, v_n) \pmod p$ are irreducible and have the same degree as the original $f_i$. *Or*

2(ii) The system will have computed $p_j, v_2, \ldots, v_n$ such that the factorisations of $f_i(v_2, \ldots, v_n) \pmod{p_j}$ are incompatible with $f_i$ being reducible (and they have the same degree as the original $f_i$). *Or*

2(iii) it's worse than that.

CA should provide a means of telling you which, and $p, v_2, \ldots, v_n$ (they currently don't)

TP Cantor–Zassenhaus [CZ81] in $\mathbf{Z}_p[x]$, and a one-off theorem for 2(i), or some messy combinatorics for 2(ii).

# 2(iii): it's worse than that

The classic example is $x^4 + 1$, which is irreducible, but factors modulo every prime into two quadratics (which may be reducible).

- The usual approach in computer algebra is to factor modulo $p$, lift the factors to a factorisation modulo $p^n$ by Hensel's Lemma, and then deduce that this is incompatible with the Landau–Mignotte bounds [Mig74] on factors of $g$.
- An alternative approach would be to ask the CA system for a largish $p$ such that the factors modulo $p$ were already incompatible with the Landau–Mignotte bounds.
- You might need large prime Berlekamp for the second, rather than Cantor–Zassenhaus if the prime really is large.

Whichever way one goes, one needs enough (complex) analysis to prove the Landau–Mignotte bounds.

# Indefinite Integration

When one types $\int f \mathrm{d}x$ into an algebra system, one gets three kinds of result:

1. Some formula $F$;
2. The same integral echoed back;
3. A hybrid $F + \int g \mathrm{d}x$.

It is expected that $F' = f$ in the first case (or $F' + g = f$ in the third).
Verifying a type 1 result is in principle easy: one differentiates $F$ and checks that it is equal to $f$. The problems are:

- $F$ may contain constructs the prover doesn't know, and the prover may be unable to prove equality.
- Even if not, the mathematical equality may be difficult (see [HT98])

# Indefinite Integration

The meaning of type 2/3 results is less clear. For certain classes $\mathcal{C}$ of functions, there are theorems (e.g. [Ris69] for the elementary transcendental functions) that allow one to assert $\nexists F \in \mathcal{C} : F' = f$, i.e. "$f$ in unintegrable (in $\mathcal{C}$). However:

- Such theorems are relatively complicated (though purely algebraic) and I know of no attempts to formalise them;
- The implementations of these in algebra systems tend to be incomplete;
- The classes $\mathcal{C}$ for which such theorems exist are much smaller than the classes in which algebra systems actually return type 1 results anyway.

Also I know of no use for such a negative result.

# Definite Integration

Assuming one has a type 1 result from indefinite integration, definite integration should be simple: $\int_a^b f \mathrm{d}x = \left[F\right]_{x=a}^{x=b}$.

**Theorem (Fundamental Theorem of Calculus [Apo67, §5.3])**

*Let $f$ and $F$ be functions defined on a closed interval $[a, b]$ such that $F' = f$* throughout $[a, b]$. *If $f$ is Riemann-integrable on $[a, b]$, then*

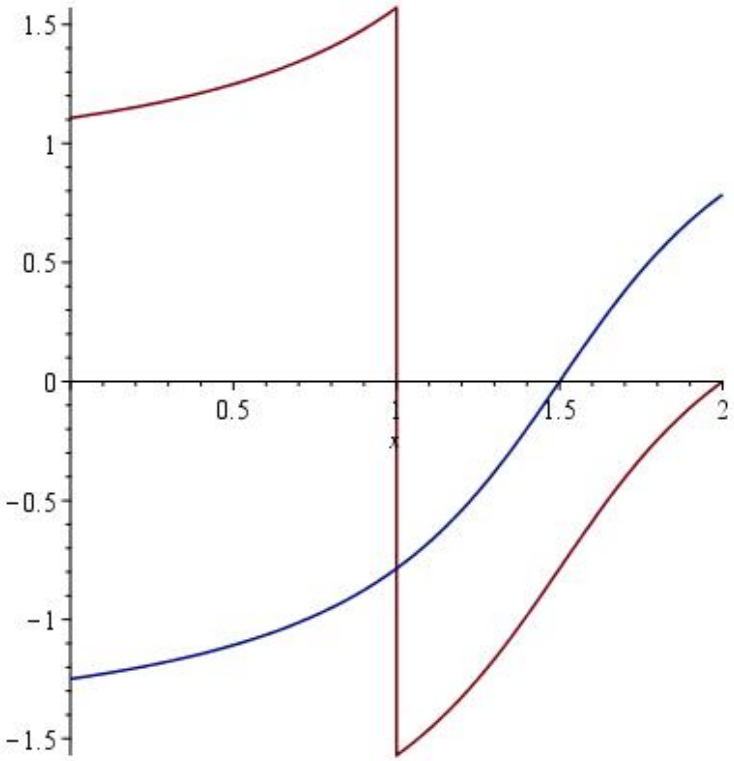
$$\int_a^b f(x)\mathrm{d}x = F(b) - F(a).$$

# Definite Integration

The integral of a positive function over a positive range cannot be negative. But $\int \frac{1}{x^2}\mathrm{d}x = \frac{-1}{x}$, therefore $\int_{-1}^{1} \frac{1}{x^2}\mathrm{d}x = \left[\frac{-1}{x}\right]_{-1}^{1} = -2$

$\int \frac{1}{2x^2-6x+5}\mathrm{d}x = \arctan\left(\frac{x-2}{x-1}\right)$, therefore

$\int_0^2 \frac{1}{2x^2-6x+5}\mathrm{d}x = \left[\arctan\left(\frac{x-2}{x-1}\right)\right]_0^2 = 0 - \arctan 2 \approx= -1.1$



# (Polynomial) Equation Solving (C)

To solve $f_i = 0$: $f_i \in k[x_1, \ldots, x_n]$.
The general technique is to compute a Gröbner basis, which *can* be computed in TP [The98, The01, CP99], but we'd probably rather not.

CA $G := \{g_i\}$ is a Gröbner base for the $\{f_i\}$.

TP1 Verify the $\{g_i\}$ are a Gröbner base:
$\forall i : g_i \rightarrow^{G \setminus \{g_i\}*} 0$,

TP2 Verify $\forall i : f_i \rightarrow^{G*} 0$, i.e. $(\{f_i\}) \subseteq (\{g_i\})$.

TP3 Verify $(\{g_i\}) \subseteq (\{f_i\})$.

i.e. "Ask for the Bézout coefficients": each $g_i = \sum h_{i,j} f_j$, so ask for, and verify this.

But1 I know of no CAS that routinely produces then,

But2 The obvious algorithm: `tdeg` GB followed by FGLM to `plex`, doesn't produce them

But3 They may be very large.

# (Polynomial) Equation Solving (**R**)

Once one allows **R**, one has to allow $\neq$, $\leq$ etc.
The algorithmic method of choice has been the *cylindrical algebraic decomposition* (CAD) of $\mathbf{R}^n$ into connected regions $C_i$ in each of which every polynomial is sign invariant, and arranged cylindrically: $\forall i, j, k$: $\pi_k(C_i)$ and $\pi_k(C_j)$ are equal or disjoint, where $\pi_k$ is the projection onto the first $k$ coordinates. Then the problem is reduced to inspecting one sample point per region.
This also allows quantifier elimination (because of cylindricity).
The initial algorithm [Col75] has had many improvements, but not exactly simplifications: more topology gets imported.
Probably needs animplementation within [TP] [Mah07]

# (Polynomial) Equation Solving (**R**: II)

Two alternative methods for computing CAD.

- Regular Chains [CM16]
  1. [CA]Decompose $\mathbf{C}^n$ cylindrically by regular chains
  2. [TP]Verify this (how?)
  3. [TP?]`MakeSemiAlgebraic`
- Comprehensive Gröbner Bases [Wei92]
  1. [CA]Build a CGB
  2. [TP]Verify this [KY15]
  3. [TP?]Use this to build CAD [FIS15]

Or Just produce a single cell of the CAD [Bro15]

- Inspired by [JdM13]

# Questions?

# Bibliography



T.M. Apostol.
*Calculus, Volume I, 2nd edition*.
Blaisdell, 1967.


C.W. Brown.
Open Non-uniform Cylindrical Algebraic Decompositions.
In *Proceedings ISSAC 2015*, pages 85–92, 2015.


C. Chen and M. Moreno Maza.
Quantifier elimination by cylindrical algebraic decomposition based on regular chains.
*J. Symbolic Comp.*, 75:74–93, 2016.

# Bibliography

II



G.E. Collins.
Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition.
In *Proceedings 2nd. GI Conference Automata Theory & Formal Languages*, pages 134–183, 1975.


T. Coquand and H. Persson.
Gröbner bases in type theory.
*International Workshop on Types for Proofs and Programs*, pages 33–46, 1999.


D.G. Cantor and H. Zassenhaus.
A New Algorithm for Factoring Polynomials over Finite Fields.
*Math. Comp.*, 36:587–592, 1981.

# Bibliography

III



R. Fukasaku, H. Iwane, and Y. Sato.
Real Quantifier Elimination by Computation of Comprehensive Gröbner Systems.
In D. Robertz, editor, *Proceedings ISSAC 2015*, pages 173–180, 2015.


J. Harrison and L. Théry.
A sceptic's approach to combining HOL and Maple.
*J. Automat. Reason.*, 21:279–294, 1998.


D. Jovanović and L. de Moura.
Solving non-linear arithmetic.
*ACM Communications in Computer Algebra*, 46(3/4):104–105, 2013.

# Bibliography


D. Kapur and Y. Yang.
An Algorithm to Check Whether a Basis of a Parametric Polynomial System is a Comprehensive Gröbner Basis and the Associated Completion Algorithm.
In D. Robertz, editor, *Proceedings ISSAC 2015*, pages 243–250, 2015.


A. Mahboubi.
Implementing the cylindrical algebraic decomposition within the Coq system.
*Math. Struct. in Comp. Science*, 17:99–127, 2007.


M. Mignotte.
An Inequality about Factors of Polynomials.
*Math. Comp.*, 28:1153–1157, 1974.

# Bibliography

V



R.H. Risch.
The Problem of Integration in Finite Terms.
*Trans. A.M.S.*, 139:167–189, 1969.


L. Théry.
A Certified Version of Buchberger's algorithm.
In *Automated Deduction — CADE-15*, pages 349–364, 1998.


L. Théry.
A machine-checked implementation of Buchberger's algorithm.

*J. Automat. Reason.*, 26:107–137, 2001.


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