James' ACA Lecture Notes

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

June 2009
(While on Sabbatical at the University of Waterloo)

Contents

Chapter 1

25 June

1.1 Gröbner bases via linear algebra and applications to comprehensive Gröbner bases — Suzuki

We can *compute* rather than *require* the term order. We will show, in particular, how this can be extended to parametric systems. We use a combination of Reduced Row Echelon Form and `mult`. They change the weightings dynamically to keep $|T|$ small, where T is the term support of the basis being considered, i.e. where the linear algebra is being done. To get comprehensive Gröbner bases, we have to ensure that variables are weighted greater than parameters, though.

Has various optimizations to avoid redundant rows. Has an implementatio in PARI/GP, which is only 1075 lines. However, it is still slower (apparently factors of thousands) than standard GB working.

1.2 Bitsize bounds for triangular sets in positive dimension — Schost

There are bounds, which are probably wrong by an extra log, which probably comes from the fact that we are evaluating at integers — he does not know how to evaluate with roots of unity.

1.3 Trigonometric and hyperbolic curves: implicitization and parameterization — Giovanna Coral and Lalo Gonzalez-Vega

Hyperbolic curves are where x and y are given as series in $\cosh m\theta$ and $\sinh m\theta$. We can check if there are simplifications (a technical phrase in this context) of

hyperbolic parameterizations, which might be either polynomial simplifications or hyperbolic ones.

1.4 Gröbner bases for parameterization — Moroz

Given $P_i(x_1, \ldots, x_n, \mathbf{T})$, can we represent the solutions to this as $x_j = Q_j(z_1, \ldots, z_m, \mathbf{T})$, where $\mathbf{T}$ are *additional* parameters. Considering only the case of a *finite* number of solutions. As the parameters change, solutions can go to infinity, or cross, or ....

Lazard used Gröbner bases for parameterization, and we can compute Gröbner bases in such a way that denominators depend only on $\mathbf{T}$.

The size of the coefficients in a lexical GB are generically bigger that the Rational Univariate Representation [DS04], and this in indeed borne out in practice. `tdrl` bases are much smaller, but cannot be used directly since we want elimination. To get this effect, we use a block degree ordering [CCT97] We order with x_1 and $\mathbf{T}$ in one block, and the remaining x_i in the second (more important) block.

There are some quite surprising size results: with his method the first parameterization is smaller with his method, but not the second. There was lively debate speaker/Schost/Moreno Maza.

1.5 Solving Parametric Systems with the RegularChains library in Maple — Changbo Chen *et al.*

What do we mean by solving a parametric system: many things, including the following.

- When does it have solutions, or finitely many solutions;
- Compute the solutions as continuous functions of the parameters.

Defines regular chain, and regular semi-algebraic set. Then RealRootClassification produced a classification on condition that the border polynomials do not vanish. The variety of the border polynomial is the minimal discriminant variety of the zero set.

Theorem 1 *Let CS be a parametric constructible set of* $\mathbf{Q}(u, x)$. *??*

Gave examples from Maple.

1.6 Sweeping for singular solutions of polynomial systems with parameters — Verschelde

1.6.1 Problem Statement

[SommeseWampler2005] A homotopy is a family of systems, depending on a parameter. h, which may be a natural

$$h(\lambda, x) = \text{unit circle}$$

or artificial

$$h(\lambda, x) = (1 - \lambda)f(x) + \lambda g(x)$$

(where g is the"good", known, system and f is the one we wish to understand), homotopy. Two motivating questions — can all complex solutions turn real; what are the **real** irreducible solution components. Global problems involve a description of the distriminant variety, which may involve solving more dfficult systems. Instead we look at a local problem: what happens as our parameter moves from a well-understood starting point.

1.6.2 Description

The most common type of singularity is a quadratic turning point. They correspond to a double solution, where the corank of the Jacobian is 1. We can detect it by monitoring the orientation of the tangent vectors. We locate the singularity by shooting.

1.6.3 Application

Neural Network Model [Noonberg1989]. Computing the winding number can be a powerful clue. In fact there are four special values.

1.7 Towards an efficient implementation for the resolution of structured linear systems — Lacelle

Solving $A.X = Y$, and A may be structured, even if not sparse, for example Sylvester's. [KKM79a, KKM79b][1] — we define a displacement operator ϕ and let the displacement rank be $\text{rank}(\phi(A)) =: \alpha$. Toeplitz is an example, where our displacement operator is

$$\phi^{+}(A) := A - \searrow A$$

[1] JHD rewrote his notes after looking at the first of these: hope they now make more sense. This work deserves to be better known.

Table 1.1: Cooperman's table

Group	State Space	Memory
Fischer Fi_{23}	1.17×10^{10}	1TB
Baby Monster	1.35×10^{10}	7TB
Janko J_4	1.31×10^{11}	8TB
Rubik	1.4×10^{12}	8TB

with $\text{rank}(\phi^+(A)) = 2$. If we take a polynomial model of $T = L.U$ with L and U lower/upper-triangular Toeplitz matrices. Gives us matrix-vector product in $2M(n)$, where M is *polynomial* (not matrix) multiplication cost.

Note the Strassen divide-and-conquer for matrix inverses. [Mor80] [BA80] give inversion in $O(\alpha^2 M(n) \log n)$. [Kal94] used randomization to remove the *generic rank profile* condition. We will divide the complexity by a constant 4–8.

$A.B$ in time $(2\alpha\beta + 2)M(n)$ and returns an $A.B$ deifned by $\alpha + \beta + 1$ generators.

Let $\gamma = A_{1,1}^{-1} A_{1,2}$ then $\Delta = A_{2,2} - A_{2,1}\gamma$, but γ also crops up in A^{-1} and this can reduce intermediate step from 10 to 6 multiplications[2]. [GohbergKailathKoltracht1986] gives resolution in $O(\alpha n^2)$ plus $O(n^2)$ per system (same A different Y). This leads to inversion in $O(\alpha n^2)$.

MBA is based on FFT which has staircase behaviour, but in fact we can avoid this behaviour. Claims that an unbalanced policy for splitting is better than a balanced policy.

Shows an example in NTL, where unbalanced lives slightly below the previous (touching at powers of 2!). The combination of all of these produces a pretty smooth curve. Another major improvement is the use of "less generic" randomisations

Q What about Pan's book?

A I've used some of it, but there is more to be used.

Q What about accuracy?

A Finite fields!

1.8 Parallel disk-based computation and computational group theory — Coopermann

See table 1.1 for the sizes of objects involved. [KGC07] proves that you can do Rubik's cube in 26 moves, and [Rokicki2008] has it down to 22 moves now.

The local discs on a cluster are larger than needed, typically 1TB *each*. The bandwidth of 50 discs is the same as the bandwidth of RAM. But the other killer is latency. Solutions:

[2]This doesn't help as much as one would think by itself, but when combined with otehr reductions, does indeed deliver.

1. delayed duplicate detection;
2. wait until there are millions of hash queries, then sort[3] on the hash index and scan the disc sequentially;
3. for pointer chasing, wait until millions of pointers are available, then sort and scan the disc sequentially.

Note that in fact it is RAM speed that is the real bottleneck, not CPU time or disc time. But we took a year debugging Rubik's cube. Hence we have now developed SPDE, a software package to help automate this. Longer term goals include a package `Roomy` with types such as `RoomyList` and operators `RoomyList_map`, `RoomyList_remdup` etc.

Q Do you have a complexity model?

A SPDE — yes; Roomy not as such.

Q Is the E_8 related work related?

A Not that I know of — I'll have to check.

1.9 Multitasking Polynomial Homotopy Computation — Verschelde

$$h(x,t) = (1-t)g(x) + tf(x) = 0$$

where $t = 1$ gives f, the problem we want to solve, while $t = 0$ is g, the good system we can solve. Various homotopy packages, mostly parallelised with MPI. Our software is PHC — Polynomial Homotpoy Continuation. v1.0 is ACM TOMS 785. MPI is on top of sockets, so we use the Python binding. PHCpack is written in Ada, compiled with gnu-ada compiler (now 64-bit). Uses it on his eight-core Macintosh. Also works on Windows. Claims that Ada is the language of choice *because* of its tasking model: he describes Ada generic as being the analogy of C++ template.

[3]Note that external sorting has no latency issues.

Chapter 2

26 June

2.1 Memory Efficiency in Polynomial Multiplicaton — Dan Roche, Waterloo

Univariate polynomial multiplication (the same essentially applies for integers apart from the carry complications). Assume $R[x]$ where R is a "unit cost" ring. In the standard model:

Read-only input

Write-only output

Read/Write work space

there is little that can be done, but claims that read/write output is more realistic.

Q Why is the standard model that way, then?

A I'm not quite sure.

A—JHD "Blame Alan [Turing], and the technology of his time".

2.1.1 Karatsuba

[Maeder1993] claims $2n$ work space for Karatsuba. [Thome2002] claims $O(n)$. We claim

- $O(\log n)$ extra space Karatsuba
- $O(1)$ extra space FFT *under certain conditions*
- Implementations for $\mathbf{Z}/p\mathbf{Z}$.

To improve Karatsuba, we need variants

V0 Compute $f \times g$ (the classical one)

V1 Compute $f \times g + h$

V2 Compute $(f_0 + f_1) \times g + h$

Restrict the modulus to 29-bits to allow for some delayed reductions. Switch over at $n \leq 32$ (determined experimentally).

There's a problem of three versions × odd/even size inputs × equal/off-by-one input sizes, so

2.1.2 FFT

Assume $\deg f + \deg g < n = 2^k$, and assume size of output $=2^k$. Assume the base ring already contains the 2^k-th primitive root of unity ω. One problem is that we need the powers of ω out of order. We can't afford to

- precompute — space costs
- recompute — time costs
- access data out of order — cache costs

so use the alternative fourmulation which computes the results "out of order", but this is OK for multiplication, since the reverse FFT will undo the permutation.

There's then a complicated mixture of folding and FFT illustrated graphically to do the operation with no extra space. Use NTL's "floating point Barrett reduction". In the limit he's seeing about twice the speed of NTL, but there is some (unexplained) step behaviour.

2.2 Speeding-up Newton iteration using variants of Newton — Schost

Actually his Masters' student's work. Solve

$$(x^6 + x^4 + 1)f'(x)^2 = 1 + 75f(x)^4 + 16f(x)^6$$

with appropriate initial conditions[1]. If we use Newton iteration, naïve would be $O(n^2)$, but we can get $O(M(n))$. Aim: hack the constant in the O, use "short products" or "middle product", and aim for moderate-sized (100s of terms) answers. The appropriate building blocks are:

Plain product $n \times n \to 2n$ - time $M(n)$

Short product $n \times n \to n$ - time $m(n)$

[1]The application is point-counting on elliptic curves in cryptography, so we are concerned with "crypto sizes", and therefore generally Karatsuba-sizes.

Middle product $2n \times n \to n$ [the middle of the expected $3n$] - time $M(n) + O(n)$.

Already using prior technology, we can do the inverses in about 40% of the "naïve" method.

Doing all this by hand, tracking the precise bits required etc., is tedious, so we can view the computation as a DAG, and do automatic code generation. His code is now abut 5 times faster than before, and 5 times the cost of a single Karatsuba multiplication.

Q "Can you explain the step behaviour in the performance graph?"

A No.

2.3 Balanced Dense Polynomial Multiplication on Multicores — Yuzhen Xie & Marc Moreno Maza

Considering *multivariate* polynomials over finite fields, and using FFTs. We take for granted

1D FFT as a sequential black box;

`modpn` library as shipped in Maple 13;

Cilk++ as parallel programming paradigm.

Let d_i and d'_i be the degrees of f and g in x_i, and assume there is a primitive s_ith[2] root of unity, $s_i \geq d_i + d'_i + 1$, in the field. Thanks to Dr Frigo for cache-efficient matrix transpose we get practically linear speedup up to 16 cores. For bivariate multiplication, $n = 1024$ stops being linear at 10 cores, $n = 4096$ at 14 cores etc.

However, for irregular data, such as bivariate large $\times$ small or 8-variate (therefore all small) we do *much* worse. This shows up when she compares L2 cache miss rates.

In general, if $s = \prod_{i=1}^{n} s_i$, the cost is $\frac{9}{2} s \log n + (n+1)s$. Let L be the size of an L2 cahce line Let $Q(s_1, \ldots, s_n)$ be the total number of cache misses. then

$$Q(s_1, \ldots, s_n \leq cs\frac{n+1}{L} + cs\left(\frac{1}{s_1} + \cdots\right).$$

2.3.1 Solution 1: contract from multivariate to bivariate

Store coefficients (only — monomials implicit) in a contiguous array. Then the contraction does *not* actually change the shape of the coeffcients array. On one core, going from 4-variate to bivariate saves a factor of two, increasing on more cores, going up to 32-fold speedup on 16 cores. Even for fairly symmetric problems (1K×2K) for example, contraction helps.

[2]It appears that s_i has to be a power of 2.

2.3.2 Solution 2

"Extend" univariate to bivariate: choose a base b and let $x^e \mapsto u^{\mathrm{erem}b} v^{\mathrm{equo}b}$. The forward conversion is trivial in memory, but the map back is a memory-bound problem. However, can be done with only one traversal. This solution, as well as allowing parallelism, also gets round the need to extend Schönhage–Strassen when we don't have large enough roots of unity in the base field. We do 10 times faster than univariate on 16 cores.

2.3.3 Balanced multiplication

$$\deg(p,u) + \deg(q,v) = \deg(q,u) + \deg(p,v).$$

We can combine the two above techniques to contract small degree variables and split a very high degree to get a balanced bivariate problem. Again sees a 10-fold speedup over naïve on 16-cores, and about 3-fold over each of the other techniques taken separately.

Q About splitting — these seems similar to Schönhage–Strassen square root techniques.

A Presumably.

Q (SMW) all graphs stop at 16 cores (presumably because that's what you have). What would happen when you go further?

A (MMM) we are dying to get our hands on machines with more cores! But we haven't considered the "more cores than degree" problem.

A (YX) but of course more cores implies more memory, so the problems will get larger in all dimensions.

2.4 SpiRal — Johnson

Reasearch goal: teach computers to "write" fast libraries. Functionality: lar transforms, BLAS etc. We are at the prelimiinary stage of generation for GPUs.

We code algorithms as rewrite rules. So a 4-point DFT is

$$(DFT_2 \otimes I_2) \cdot T_2^4 \cdot (I_2 \otimes DFT_2) \cdots L_4^{2,2} \tag{2.1}$$

where L is the appropriate permutation. Have over 50 rules, and therefore we could generate many possibilities for each case. After the rewrite rule manipulation, SPL does the vector/parallel processing and generation, e.g. p-way load balancing and cache line size optimisation. Then Σ-SPL does loop optimisation, and then we go to C generation. Note that $I_p \otimes A$ is naturally p-way parallel. However, there are problems, notably "false sharing", i.e. different processors writing to different elements of the same cache line. We can spot this by trying to generate $\ldots \otimes I_m$ where m is the cache line size.

Similar tricks, in a *uniform* framework, work for GPUs, or even MPI.

2.4.1 Adding modular DFT to Spiral — student

`ModDFT(n,p,w,i)` where `i` is the forward/inverse flag. Need a Cooley–Tukey-like breakdown rule for this, morally similar to the **C**-based one. In fact, the breakdown equation is similar to (2.1), apart from the manipulation of the roots of unity.

The unparser (C generator) needs a lot of work to fold the `%` operators. Can replace `%p` by `x>p?x-p:x` etc., and indeed can use this to generate conditional move instructions, thus avoiding explcit branches. Typically less than 50% slower than complex DFT, and actually faster at the point where complex DFT has hit the cache size but ModDFT hasn't yet (smallet data!).

Future work includes importing Montgomery-like methods, and explciti assembler generation of the conditional move instructions.

Q–Coopermann Much of this deals with hardware restrictions — nVidia has lots! Have you approached them?

A Not yet — we wnat to get better data first. Note that we don't need to know the "secrets" since test-based optimisation will find them anyway.

2.5 Linear Algebra Modulo Tiny Primes — Saunders *et al.*

Actually mainly interesting in rank. Extending LinBox. Motivating examples as follows.

1. Matrices of size 9^8, and we want the 3-rank of the matrix, which we expect to be much smaller. We have done the 9^7 case, taking 4 days on a single processor.

2. Brouwer, algebraic graph theory, in particular distance-regular graphs, where we have graphs of size 10^6.

Let G be the additive group of $\mathbf{F}_{3^8} \times \mathbf{F}_{3^8}$, to which we add a non-commutative multiplication. $m_{i,j}$ is formulaic, so we don't necessarily store it.

- We use three bits per field elements, and "semi-normalise", i.e. 21 walues per word., then

  ```
  z:=x+y
  z:=z+(z>>2) & mask
  ```

  adds the 21 numbers[3].

- Use two bits *in two different words!*, so two 64-bit words hold 64 values. Addition then takes 12 operations.

[3] We may need two copies of the second line — it wasn't clear.

- Packing in mantissae of floats, using 4 bits/entry. Can therefore use highly-optimised f.p. multiplications, and BLAS (followed by post-normalisation. Goes from 350M op/sec naïvely to 3000 with pakced integers to 20000 in this format.

Note that even though $r << n$, we can't afford to store the $r \times n$ independence basis. We therefore have a heuristic lower bound, which we then nede to certify.

2.6 FFT-based Dense Polynomal Arithmetic on Multicores — Marc Moreno Maza and Yuzhen Xie

We want to produce a BPAS which will support polynomial systems solvers. It follows from the previous talk (2.3) that balanced bivariate is a good kernel. We want addition, multiplcation exact division and normal form with respect to a reduced monic triangular set. The last is the most difficult. We want to look at the TFT/FFT trade-off: theoretical analysis followed by experimentation. The answer in fact isn't clear: TFT often wins, but not always.

Has figures from Cilkscreen. NormalForm makes *many* calls to `map`.

2.7 Code generation and autotuning in computer algebra — Anatole Ruslanov, with Jeremy Johnson, Werner Krandick & David Richardson

Computer architecture is

- very complex and efficient
- proprietary (and undocumented in places)
- rapdily evolving

Aim, to generate and test many implementations, either live or at installation. There are many algorithms, each of which have many variants. Pipelines are a major problem — AMD's tend to be simpler than Intel's. How many functional units do we have? How many operations can they perform? How do compilers map on to these? Many computer algebra operations rely on subroutines, e.g. for bignum arithmetic.

We are lookimg at fast implementation of Taylor shift, e.g. Pascal's triangle.

- Signed digits
- Reduced radix (to reduce need for carry propagation).

- Register Tiling

Speedup over GMP is maximum of 7-fold when GMP is just into the multi-digit range, tending to 3-fold as $n \to \infty$.

Have a PERL-based code generator, that unrolls loops[4] and chooses tile sizes. The best performances, on an Opteron, come at 10×10 or 12×12 tilings, which fits with 16 registers, less two for hoursekeeping and two for one side of the tile. Similar, but less pronounced, effects on other architectures.

Q How fast?

A 5–10 minutes.

[4]Claims that compilers still do best if this is done for them. There were too many loops (in total 6–7) for Sun, Intel or gcc to do efficient tiling. Their system unrolls to the depth of the chosen tiling.

Chapter 3

27 June

3.1 Which CASC will fill the gap? — Böhm, Austrian Centre for Didactics of Computer Algebra

["CASC" is an acronym for "Computer Algebra and Symbolic Computation" in some educational circles. JHD finds the mere title of the centre awesome.]

Austria was the first country to have a nation-wide (secondary school) Derive licence. What, now that Derive is off the market? Many schools use TI-92, some MathCAD, some Macsyma. No Maple at the secondary level. However, many Austrian[1] curricula *mandate* the use of CASC, but not which. The Austrian government has just (2 days ago) mandated national examinations at the end of grammar schools, with effect from 2013. No-one knows how this will fit with the CASC requirement in curricula.

The speaker mentioned[2] TI-Nspire, Voyage 200, wxMaxima, YACAS, MuPad, GeoGebra (not currently a CASC, but trying to use YACAS as a front end), WIRIS.

The speaker demonstrated WIRIS, which he said had a rare feature, conversion of units. It also has a feature much nicer than Derive, *viz.* ability to plot form the algebra window directly (he admitted that some other systems also allowed this). Again, aimed at secondary schools, there is a conics package and a fairly graphical elementary combinatorics package. Several complaints of the form "DERIVE has XXX, but WIRIS doesn't".

The Austrian national contract with WIRIS only allows school to use the on-line version, which might cause serious problems in examinations.

There is a German menu line, but the commands used in Austria are in English, as was the case with Derive. Conversely, the Estonians have a complete translation, which means that the Estonians can't use his programs. There is

[1] Also one German Länd, and Slovakia, according to ES/DJJ.

[2] The speaker (after questions), had not heard of SAGE.

also a Spanish version.

3.2 SciLab and Maxima Envronment: towards free software in Numerical analysis — University of Málaga

The fundamental goal was to find an alternative to MatLab. The problems are cost (it appears that they don't have a site licence), and the temptations to illegal software. The criteria for a replacement were

- Good mathematical engine, with calculus etc.,;
- A sound development team with longevity
- functionality (libraries etc.)
- as much MatLab compatililty as possible.

They were left with Octave and Scilab (INRIA), after discarding Yacas, Maxima and others JHD has not heard of. The main Scilab problem was the interface, especially as students were used to MatLab. The students were also used to symbolic differentiation[3], so have a menu-like interface to Maxima to provide this.

He demonstrated a side-by-side comparison of MatLab and SciLab, and indeed there were strong similarities. Typical differences are ^ versus **. Their interface is run-time switchable from English to Spanish, and this is driven off a simple text file, so other languages are easy.

3.3 Developing an automated learning assistant for vector calculus — Yaacub (National University of Malaysia), with Steinberg and Wester (NMSU)

We have a pre-set textbook, and the aim was to enhance this. Interactivity was an important requirement. We want the user to create knowledge representations, possibly multiple (numeric, symbolic and graphical) representations.

Q Are the examples generated randomly?

A "according to the syllabus"

[3] In answer to a question, this is *the* pressing symbolic need.

3.4 Getting from x to y without a crash: experiences teaching with Maple and MatLab — Jeffrey (UWO)

"A car may be merely the way of getting from A to B, but you still need to know how to drive". We often get students who freeze up in front of computer, or indeed *vice versa*

3.4.1 MatLab

- "Which window do we use, and which file do we use?". Most "MatLab in Linear Algebra" books start with the interactive window, whereas we actually want to students to *write* something, than merely *do* something. Furthermore, when it's time to define a function, they are already in the window. See the "canoeing joke" [JC01, p. 7].
- "Commas or spaces?" Books tend to use spaces, but students can't see them, or their absence, so I always teach commas.
- What linear algebra *textbook* teaches

$$a + \begin{pmatrix} b & c \\ d & e \end{pmatrix} = \begin{pmatrix} a+b & a+c \\ a+d & a+e \end{pmatrix}?$$

  but MatLab does this (and worse, as we shall see).
- Difference between `*` and `.*`, and worse `/` and `./` (the first is matrix division, even though we always teach that there is no such thing, which in fact becomes least-squares solutoin of linear equations, the second is element-by-element division).
- functions: `function a = FunctionName (a y)` *does* define a function, but its name is *not* `FunctionName`, but rather the name of the file this is stored in!

3.4.2 Maple

- Biggest problem is differences is semantics of copying and assignment between Maple and MatLab.
- Also in Maple `f(x,y):= x^2+y^2` does *not* define a function, rather the value of the particular expression `f(x,y)`, and not `f(u,v)`.

3.4.3 So?

I set questions of the form "what is the difference between (examples including) `/` and `./`. As a professor, I can set questions that *force* people to read the documentation, which gives me an advantage over the system writers.

3.5 A Comparison of Equality in Computer Algebra and Correctness in Mathematical Pedagogy — Bradford, Davenport, Sangwin

Q—Jeffrey This would have been very useful a few years ago, when we set a question on integrating factors (it was $\exp \log x$), and then had a debate in the marking room as to whether $\exp(n \log x) \to x^n$ was forbidden, optional or required.

A Noted. Isn't a little unfair on the students to set a question and *then* argue about the marking scheme?

3.6 Symbolic Math in Engineering Education — Jiro Doke (MathWorks)

This did seem to JHD to be a MatLab/SimuLink sales talk. He mentioned the new notepad-like MuPad interface. MuPad apparently has its (own?) built-in editor. It seemed as though passing data between MuPad and MatLab isn't totally trivial, especially as we want to convert MuPad *data* inot MatLab *functions*.

Q—Jeffrey

A There is still the command-line interface to the symbolic toolbox.

3.7 Conditioning numerical representations of algebraic sets — Guan (and Verschelde)

In exact arithmetic there is a commuting diagram between homotopy and "moving from extrinsic representation (condition number K_E) to intrinsic representation (condition number K_I)". For a bivariate example with random perturbations (in Maple) of 10^{-6}, $K_E \approx 10^{-4}$, but $K_I \approx 10^{-1}$. For a twisted cubit example, though, the differences are not so great (factors of 2–4). The conclusion is that there *is* no conclusion: the intrinsic is simpler, but the greater condition number may require smaller step sizes.

3.8 Chebyshev expansion for yet another paving algorithm — Moroz

Given a multivariate polynomial inequation $P(X_1, \ldots, X_n) \neq 0$ in a bix, what is the volume of each connected component? Very naïve implementatio by subdivision, better by quad-tree.

- Symbolic optimization (Lagrange)
- Narrowing operators
- Interval analysis
- Coefficient analysis, e.g. Bernstein bases.

Chebyshev basis extends to dimension > 1, but can we still get quadratic convergence?

Chapter 4

28 June

4.1 Computing cylindrical algebraic decomposition via Triangular Decomposition — Changbo Chen (and Marc Moreno Maza)

Aims:

- Understand the relationship between triangular decomposition and CAD;
- Make CAD more efficient (by leveraging the efficient polynomial tools being build for triangular decomposition).

Recap definition of a cylindrical algebraic decomposition, and Collins' algorithm [Col75] by projection and lifting. His algorithm (to appear in ISSAC 2009) is as follows.

Initial Partition Decompose $\mathbf{C}^n$ into disjoint constructive sets $c_1 \ldots, C_n$ such that each f_i is either identically or never zero on each C_j. Given polynomials $f_1, \ldots, f_s$, we compute an *intersection-free basis* of the $s+1$ sets $f_1 = 0$, $\ldots$, $f_s = 0$, $\prod f_1 \neq 0$. Given a regular system in $k[y_1 < \cdots y_{n-1} < y_n]$,we regard $y_1, \ldots, y_{n-1}$ as parameters, and compute a comprehensive triangular decomposition.

Make Cylindrical Transform this into another decomposition which is complex-cylindrical. This and Separatezeros are mutually recursive.

Make Semi-algebraic This works via a theorem of [Col75].

> **Theorem 2** *If* init(p) *does not vanish in* R *and the number of distinct roots of* p *is constant on* R*, then* R *is* delineable *on* R*.*
>
> This extends to a set of polynomials.

Maple demo (thanks to JHD and John May). On the parabola $ax^2 + bc + c$ with $x > a > b > c$, they get a 27-cell decomposition, whereas the Collins–Hong projection operator or McCallum's gives 115. McCallum–Brown [Bro01] also gives 27, but this projection may fail.

For most, but not all, examples, the first two phases, essentially complex, dominate. This implies that work on the RegularChains library will be very productive here also.

4.2 Numerical determination of the local dimension of a solution of a polynomial system — Dan Bates

Introduces numerical algebraic gemetry — used to be just polynomial system solving, but now does genus computation etc. All based on homotopy continuation. "Grew out of robotics". Numerical decomposition is "at least one point on every component of the algebraic set"

Given a polynomial system f, consider each f_i by itself, times all degree 1 monomials ... times all degree k monomials, and take the matrix of partial derivatives at p. Let the rank be $m(k)$ (essentially a Hilbert function value). If p is not isolated, this grows forever (i.e. until we reach a bound on the dimension of the ideal), whereas if p is isolated, it stops at the multiplicity. If it's not isolated, intersect with a hyperplane to reduce the dimension by 1. For largish examples, he is seeing factors of 10–30. He also has advantages from 16-fold parallelism, using them for doing the homotopy following.

Q–JHD How do you ensure that your slicing is 'generic'.

A We don't — all these applications are "with probability 1".[1]

4.3 Applications of Symbolic–Numeric Methods in Geodesy — R.H. Lewis, with B. Pálancz, P. Zaletnyik, J. Awange

Many geodesic problems can be represented as polynomial systems. The GPS equation:

$$(X - a_i)^2 + (Y - b_i)^2 + (Z - c_i)^2 + (T - P_i)^2 = 0 \tag{4.1}$$

for each satellite, where P_i is the measured distance. This can be very ill-conditioned, though.

[1] The speaker subsequently pointed out that there are retrospective certification methods.

4.4 Partitioning Multivariate Polynomial Equations via Vertex Cuts — Bard

Some tricks work for small fields, others for large ones. A trivial case is where the f_i are in the x_i, and the g_j in the x_j. For linear equations this would correspond to a block diagonal matrix. Now imagine permuting a block diagonal matrix — can we recognise this?

In general, we build a "variable sharing graph", disconnectedness in which will detect separability: admittedly a rare condition. Suppose our graph is "not very well connected". If F is small, then we can guess values for the variables which connect the graph, and, for each guess, solve the resultant separated systems. It is, of course, also very parallelisable.

Theorem 3 (Menger) *If there are n vertex-distinct paths in the graph from A to B, then ....*

Suppose our graph has vertices $V_1 \cup C \cup V_2$, where there are no V_1–V_2 edges, and $|V_1| \approx |V_2|$, this will be a "balanced vertex cut" The problem is NP-hard, but in practice there are good software solutions, e.g. METIS, for balanced edge cuts, and conversion is relatively simple (?).

It is also often the case (e.g. finding outcircles), where there may be variables that we don't really care about.

Applications with breaking of Trivium, where we assume some bits are known by, e.g. electron microscopy techniques (apparently there's a lab at UC London working on this), and, if these are in the cut set, then the combinatorics are seriously reduced.

4.5 Vector Space Bases Associated to Vanishing Ideals of Points — Lundqvist

The vanishing ideal of a set of points is the set of polynomials that vanish at all such points. They are important for interpolation problems. The main algorithm is [MB82], which is in fact in polynomial time. However we claim there are others, and using Gröbner bases for vanishing ideals is overkill, since it ignores the key fact that we know the vanishing set.

See `www.math.su.sw/~samuel`. Assumes that the points are distinct.

4.6 Computations Modulo Regular Chains — Wei Pan, Marc Moreno Maza & Xin Li

Example — computing the intersections of two plane curves: we compute the resultant, and then need gcds modulo the resultant. Say that G is a *regular* gcd of P and Q if $\mathrm{lc}(G, y)$ is a regular element of $A[y]$ as well as being a gcd. In practice $A = k[x_1, \ldots, x - n]/\mathrm{sat}(T)$. This may not exist, and we may need to

split sat(T). First proposed by [MorenoMaza2000], though related ideas appear in [Kalkbrener1993],[MMR95]. Our aim is a new algorithm to compute regular gcds.

Let T be a triangular set. T_v is the polynomial with main variable v, and $T_{<v}$ is all those whose main variable is less than v. Suppose $\deg(P) > \deg(Q) = q > 0$. Let $S_d(P,Q)$ be the d-th subresultant, and $s - D = \text{coeff}(S_d, x^d)$. If $s_D = 0$ then S_d is *defective*. Let T be a regular chain such that $R := \text{Res}_y(P,Q) \in \text{sat}(T)$ and the initials of P and Q are regular with respect to T.

Theorem 4 *If* $\text{lc}(S_d, y)$ *is regular modulo* T*, then* S_d *is not defective.*

Example: $P = x_2^2y^2 - x_1^4$ and $Q = x_1^2y^2 - x_2^4$, then $R = (x_1^6 - x_2^6)^2$, and $S_1 = (x_1^6 - x_2^6)$, which is nonzero modulo $\langle R \rangle$, but is not regular modulo $\langle R \rangle$.

Let $b_i = 2d_id_{i+1}$ and $B = \prod(b_1 + 1)$. We compute S_j via FFT on a grid of points which do not cancel init(P) or init(Q). In the normal case (degree 1), we get $\tilde{O}(d_{n+1}B)$.

Timings for generic cases are sub-second for fast triangularise, seconds for lex GB, and hours for `solve`, all in Maple. For highly non-equiprojectable cases, they beat Magma's new code by a factor of ≈ 6. Random dense trivariates are also much faster than Magma's `regularize`.

4.7 A Note on the Performance of Sparse Matrix–Vector Multiplications with Column Reordering — Haque & Hossein, presented by Moreno Maza

The vector is assumed dense, but we can play games with the matrix. We want to improve spatial and temporal locality of access to the matrix. We know existing schemes:

CRS Compressed Row Storage;

FSB Fixed-Size Blocks storage, where $A = \sum A_i$ with each A_i having fixed-length blocks (stored in a pretty perverse way);

??? (I think the previous scheme after column permutation).

In general, trying to find the optimal encoding is very expensive. They appear to have a heuristic for finding a good column ordering based on "intersection weight". Apparently have good results for Gray codes.

Bibliography

[BA80] R.R. Bitmead and B.D.O. Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *J. Linear Algebra Appl.*, 34:103–116, 1980.

[Bro01] C.W. Brown. The McCallum projection, lifting, and order-invariance. Technical Report MOTS2001, 2001.

[CCT97] M. Caboara, P. Conti, and C. Traverso. Yet Another Ideal Decomposition Algorithm. In T. Mora and H. Matson, editors, *Proceedings AAECC 12*, 1997.

[Col75] 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.

[DS04] X. Dahan and É. Schost. Sharp Estimates for Triangular Sets. In J. Gutierrez, editor, *Proceedings ISSAC 2004*, pages 103–110, 2004.

[JC01] D.J. Jeffrey and R.M. Corless. Teaching Linear Algebra with and to Computers. `http://epatcm.any2any.us/EP/EP2001/ATCMP127/fullpaper.pdf`, 2001.

[Kal94] E. Kaltofen. Asymptotically fast solution of Toeplitz-like singular linear systems. In *Proceedings ISSAC 1994*, pages 297–304, 1994.

[KGC07] D. Kunkle and G. Gene Cooperman. Twenty-six moves suffice for Rubik's cube. In C.W.Brown, editor, *Proceedings ISSAC 2007*, pages 235–242, 2007.

[KKM79a] T. Kailath, S.-Y. Kung, and M. Morf. Displacement ranks of a matrix. *Bull. A.M.S.*, 1:769–773, 1979.

[KKM79b] T. Kailath, S.-Y. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *J. Math. Appl.*, 68:395–407, 1979.

[MB82] H.M. Möller and B. Buchberger. The Construction of Multivariate Polynomials with Preassigned Zeros. In *Proceedings EUROCAM 82 [Springer Lecture Notes in Computer Science 144*, pages 24–31, 1982.

[MMR95] M. Moreno Maza and R. Rioboo. Polynomial Gcd Computation over Towers of Algebraic Extensions. In G. Cohen, M. Giusti, and T. Mora, editors, *Proceedings AAECC 11*, pages 365–382, 1995.

[Mor80] M. Morf. Doubling algorithms for Toeplitz and related equations. In *Proceedings IEEE Internat. Conf. on Acoustics*, pages 954–959, 1980.