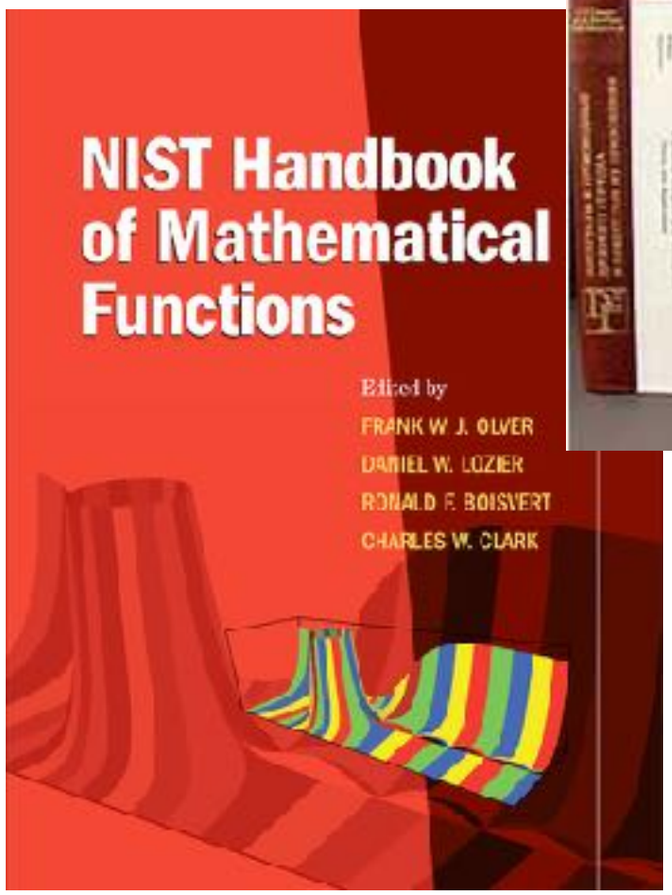
CR06: Modern Algorithms for Symbolic Summation and Integration

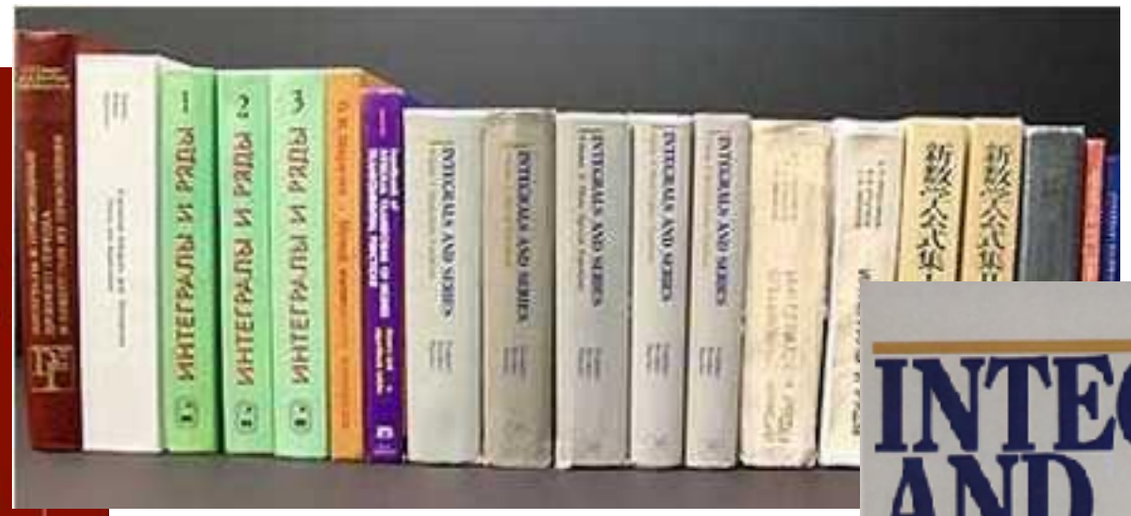
Alin Bostan, Bruno Salvy, Gilles Villard

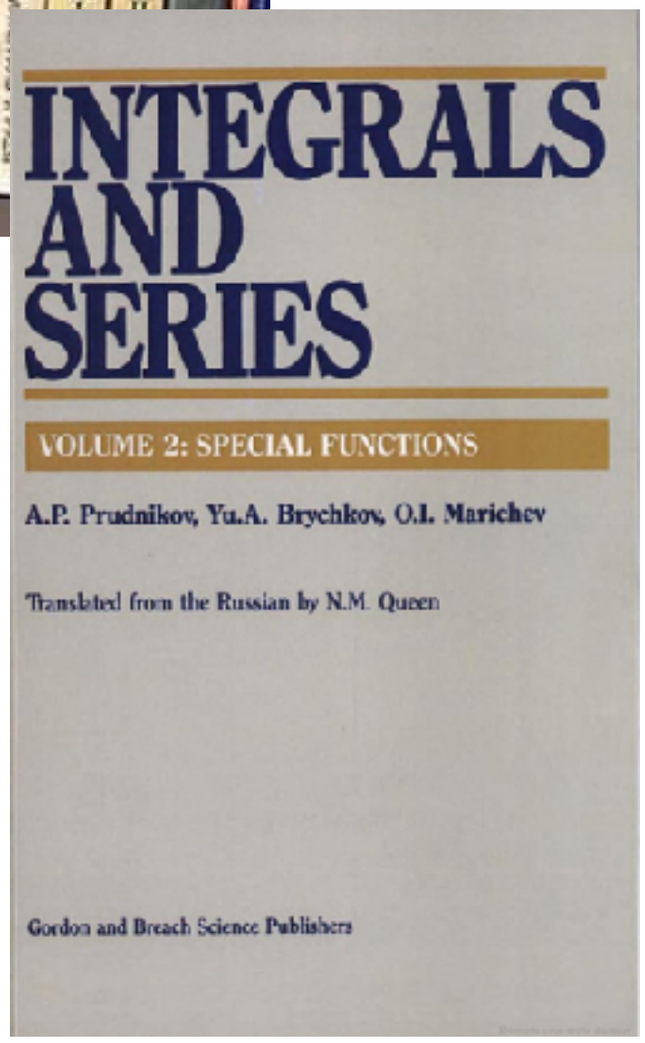
Lecture 0: Introduction

Summation and Integration from Books



https://dlmf.nist.gov





5 volumes
> 3500 pages

Aim: automate this.
As much as possible.
Efficiently.

I. Context

Logic vs Symbolic Computation

Thm. (Richardson-Matiyasevich) In the class of expressions built from one variable x and the constant 1 with the operations $+, -, \times$ and composition with the fonctions $\sin(\cdot)$ and absolute value $|\cdot|$, recognizing 0 is undecidable.

What can we do against that?

Use algebra!

Computer Algebra

Effective mathematics: what can we compute exactly?
And complexity: how fast? (also, how big is the result?)

Systems with several million users

50+ years of algorithmic progress

Equations as a Data-Structure

Example 1: $$\frac{\sin\frac{2\pi}{7}}{\sin^2\frac{3\pi}{7}} - \frac{\sin\frac{\pi}{7}}{\sin^2\frac{2\pi}{7}} + \frac{\sin\frac{3\pi}{7}}{\sin^2\frac{\pi}{7}} = 2\sqrt{7}$$

Lect. 2

has a simple proof using $e^{i\pi/7}$ root of $x^7 + 1$ and resultants.

Example 2: $\sin^2 + \cos^2 = 1$

has a simple proof using $y'' + y = 0$.

Blackboard proof

Algorithm in Lect. 3

Example 3: Mehler's identity

$$\sum_{n=0}^{\infty} H_n(x)H_n(y)\frac{u^n}{n!} = \frac{\exp\left(\frac{4u(xy - u(x^2+y^2))}{1-4u^2}\right)}{\sqrt{1-4u^2}}$$

II. Efficiency (basic results)

One Second of Computation

With a good polynomial or integer library,
1 sec. is sufficient to

multiply two integers with 30,000,000 digits;
multiply two polynomials of degree 650,000;
multiply two matrices of size 850x850;
(but factor an integer with 42 digits only).

1 sec. is in the asymptotic regime of the algorithms

Multiplication Algorithms (1/2)

Let $\mathsf{M}(n)$ be a bound on the number of coefficient operations needed to multiply two polynomials of degree at most n. Then,

$$\mathsf{M}(n) = \begin{cases} O(n^2) & \text{by the naive algorithm;} \\ O(n^{\log_2 3}) & \text{by Karatsuba's algorithm;} \\ O(n^{\log_k(2k-1)}) & \text{by Toom-Cook's algorithm;} \\ O(n \log n) & \text{by FFT (with primitive roots of 1).} \end{cases}$$

All of them are useful

They all satisfy $\mathsf{M}(n_1) + \mathsf{M}(n_2) \leq \mathsf{M}(n_1 + n_2)$, $\mathsf{M}(mn) \leq m^2 \mathsf{M}(n)$.

Multiplication is almost as cheap as addition

Multiplication Algorithms (2/2)

Let $\mathsf{M}_{\mathbb{Z}}(n)$ be a bound on the number of bit operations needed to multiply two integers of at most n bits. Then,

$$\mathsf{M}_{\mathbb{Z}}(n) = \begin{cases} O(n^2) & \text{by the naive algorithm;} \\ O(n^{\log_2 3}) & \text{by Karatsuba's algorithm;} \\ O(n^{\log_k(2k-1)}) & \text{by Toom-Cook's algorithm;} \\ O(n \log n) & \text{by FFT}. \end{cases}$$

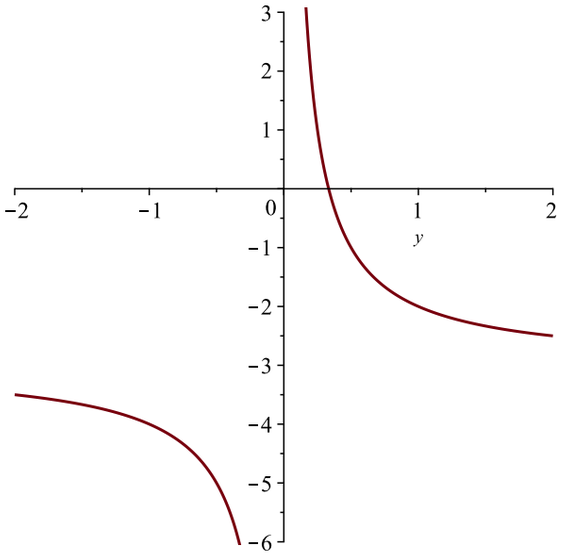
All of them are useful

Harvey & van der Hoeven (2019)

They all satisfy $\mathsf{M}_{\mathbb{Z}}(n_1) + \mathsf{M}_{\mathbb{Z}}(n_2) \leq \mathsf{M}_{\mathbb{Z}}(n_1 + n_2)$, $\mathsf{M}_{\mathbb{Z}}(mn) \leq m^2 \mathsf{M}_{\mathbb{Z}}(n)$.

Multiplication is almost as cheap as addition

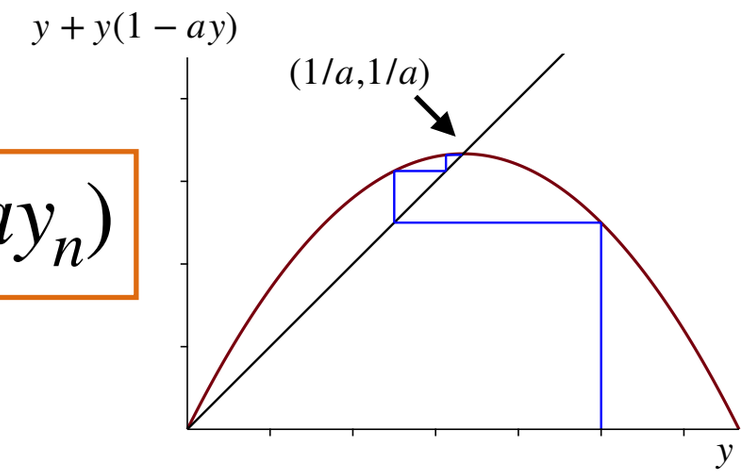
Reciprocal of Power Series by Newton's Iteration



$$\phi(y) = 1/y - a$$

$$\boxed{y_{n+1} = \mathcal{N}(y_n) := y_n + y_n(1 - ay_n)}$$

No division needed!



For any y, $\quad y = a^{-1} + O(X^{k}) \Longrightarrow \mathcal{N}(y) = a^{-1} + O(X^{2k})$.

The number of correct terms is doubled at each iteration.

$$a = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 + \frac{1}{120}x^5 + \frac{1}{720}x^6 + O\left(x^7\right)$$

$$y_0 = 1 + O(x),$$

$$y_1 = 1 - x + O(x^2),$$

$$y_2 = 1 - x + \frac{1}{2}x^2 - \frac{1}{6}x^3 + O(x^4)$$

Blackboard proof

```
def InvertSeries(ma):
    n = len(ma)
    if n==1: return [-1/ma[0]]
    k = -(-n//2)        # ceil(n/2)
    s = InvertSeries(ma[:k])+[0]*(n-k)
    t = Mul(ma,s,n)     # -a*s
    t[0] += 1           # 1-a*s
    return Add(s,Mul(s,t,n))# s+s(1-a*s)

def NewtonInvert(a):
    return InvertSeries(ScalarMul(-1,a))
```

Complexity:

$$C(N) \le C(\lceil N/2 \rceil) + 2\mathsf{M}(N) = O(\mathsf{M}(N)) .$$

Division is not harder than multiplication!

Euclidean Division of Polynomials

$$(A(X), B(X)) \mapsto (Q(X), R(X)) \text{ s.t. } \begin{cases} A(X) = B(X)Q(X) + R(X), \\ \deg R(X) < \deg B(X)\,. \end{cases}$$

$$\frac{A(X)}{B(X)} = Q(X) + \frac{R(X)}{B(X)}$$

$X \mapsto 1/T$
and
$\times\, T^{\deg A - \deg B}$

$$\frac{T^{\deg A} A(1/T)}{T^{\deg B} B(1/T)} = T^{\deg A - \deg B} Q(1/T) + \frac{\overbrace{T^{\deg A} R(1/T)}^{O(T^{\deg A - \deg B + 1})}}{T^{\deg B} B(1/T)}$$

The coefficients of Q come first

1. Compute $\tilde{A} = T^{\deg A} A(1/T), \tilde{B} = T^{\deg B} B(1/T)$
2. Compute $\tilde{Q} = \tilde{A} \times \text{Inverse}(\tilde{B} + O(T^{\deg A - \deg B + 1}))$
3. Recover $Q = T^{\deg A - \deg B} \tilde{Q}(1/T)$ (for free)
4. Deduce $R = A - BQ$.

Complexity:
If $\deg A = cn$
& $\deg B = n$,
division in
$O(\mathsf{M}(n))$.

A similar method
is used for integers.

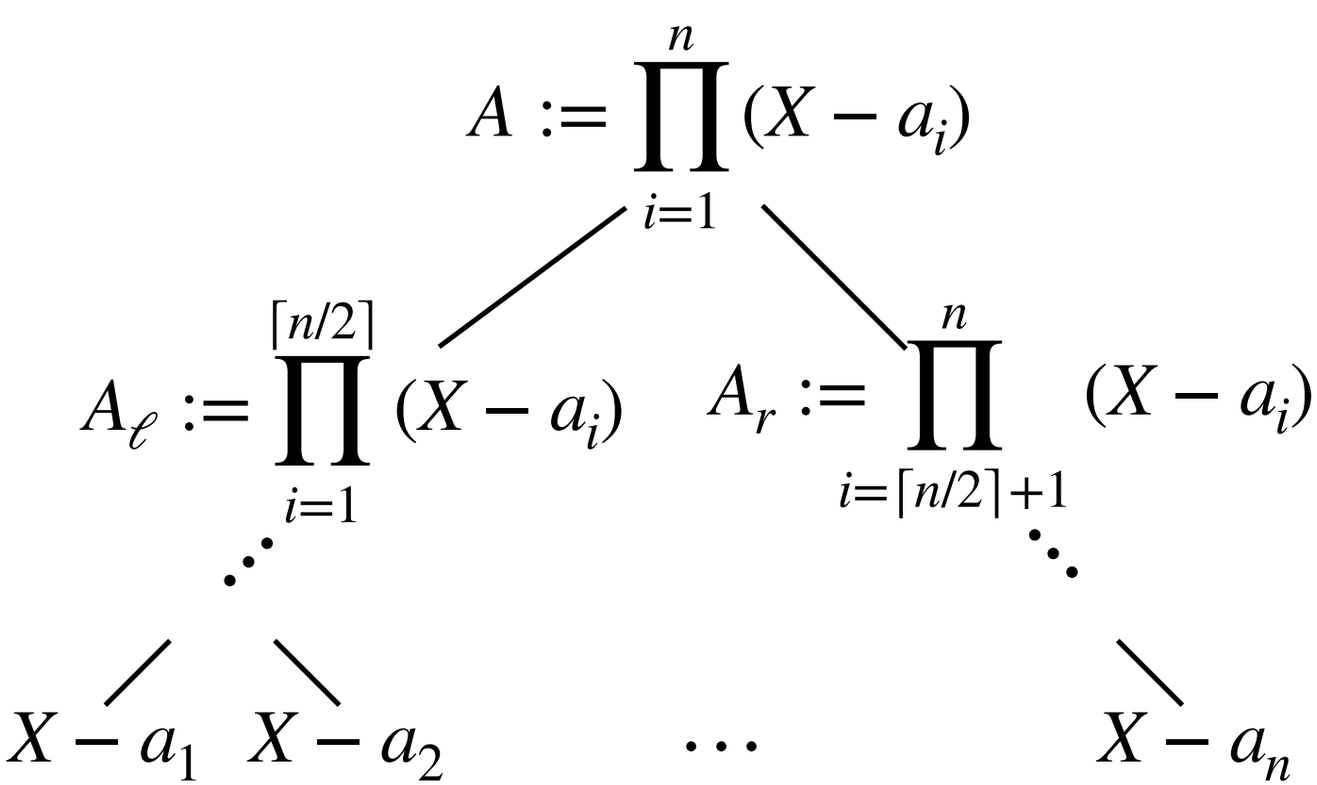
Multipoint Evaluation

Input: $P \in \mathbb{K}[X]_{<n}, (a_1, \ldots, a_n) \in \mathbb{K}^n$

Output: $(P(a_1), \ldots, P(a_n))$

Naïvely: $O(n^2)$

Step 1: construct a product-tree $\mathcal{T}_A$



Step 2: use it

input: $(P, \mathcal{T}_A)$

If $\deg A = 1$ return P

Else return

$\mathrm{Eval}(P \bmod A_\ell, \mathcal{T}_{A_\ell}),$

$\mathrm{Eval}(P \bmod A_r, \mathcal{T}_{A_r})$.

Complexity:
$C(n) \leq 2C(n/2) + O(\mathsf{M}(n))$
$= O(\mathsf{M}(n) \log n)$.

Interpolation

Input: $(a_1, \ldots, a_n, b_1, \ldots, b_n) \in \mathbb{K}^{2n}$, a_i distinct.

Output: $P \in \mathbb{K}[X]_{<n}$ s.t. $P(a_i) = b_i$, $i = 1, \ldots, n$.

Principle: partial fraction decomposition gives

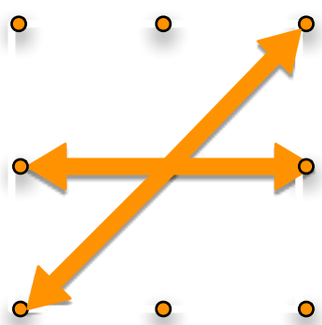
$$\frac{P(X)}{A(X)} = \sum_{i=1}^{n} \frac{b_i}{A'(a_i)} \frac{1}{X - a_i}, \text{ with } A(X) = \prod_{i=1}^{n} (X - a_i).$$

1. Compute A, A'; $O(\mathsf{M}(n)\log n)$
2. Multipoint-evaluation $\rightarrow (A'(a_1), \ldots, A'(a_n))$; $O(\mathsf{M}(n)\log n)$
3. $c_i := b_i / A'(a_i)$, $i = 1, \ldots, n$; $O(n)$
4. $\sum c_i/(X - a_i)$ by divide-and-conquer; $O(\mathsf{M}(n)\log n)$
5. Return its numerator.

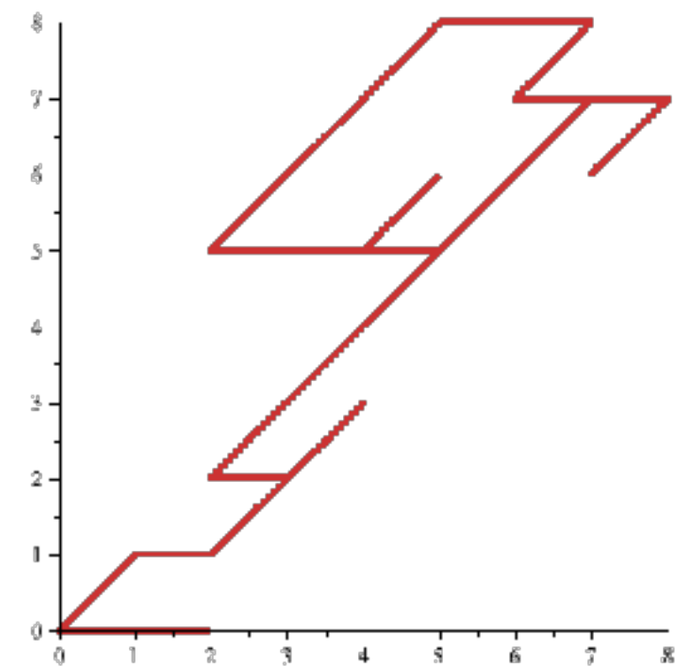
Total complexity: $O(\mathsf{M}(n)\log n)$.

Need for efficiency: the example of Gessel's walks

$$G(x,y,t) := \sum_{n\geq 0} \sum_{i,j} f_{i,j;n} x^i y^j t^n$$



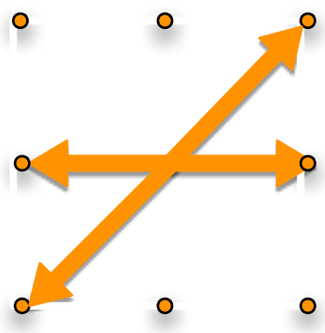
- 79 inequivalent step sets;
- long history of special cases;
- Gessel's was left;
- conjectured **not** soln LDE.



Thm. [Bostan-Kauers 2010]

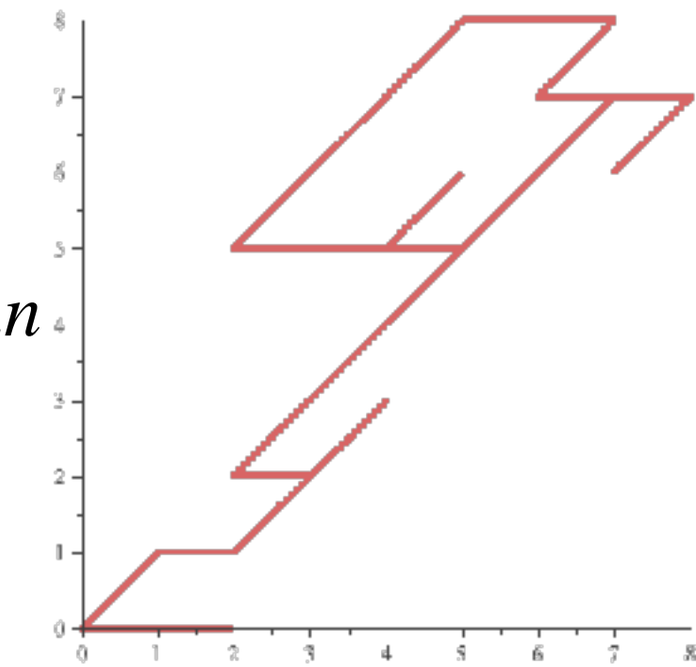
G is algebraic! (and thus soln LDE)

Computer-driven discovery and proof



Gessel's walks

$$G(x,y,t) := \sum_{n\geq 0}\sum_{i,j} f_{i,j;n} x^i y^j t^n$$



- compute G up to t^{1000};
- conjecture LDE (with 1.5 billion coeffs!); Lect. 6
- check for sanity (bit size, more coeffs, Fuchsian, p-curvature);
- Oho!
- Conjecture polynomials (deg $\leq$(45,45,25), 25 digit coeffs);
- Proof by (big) resultants. Lect. ≈3-9

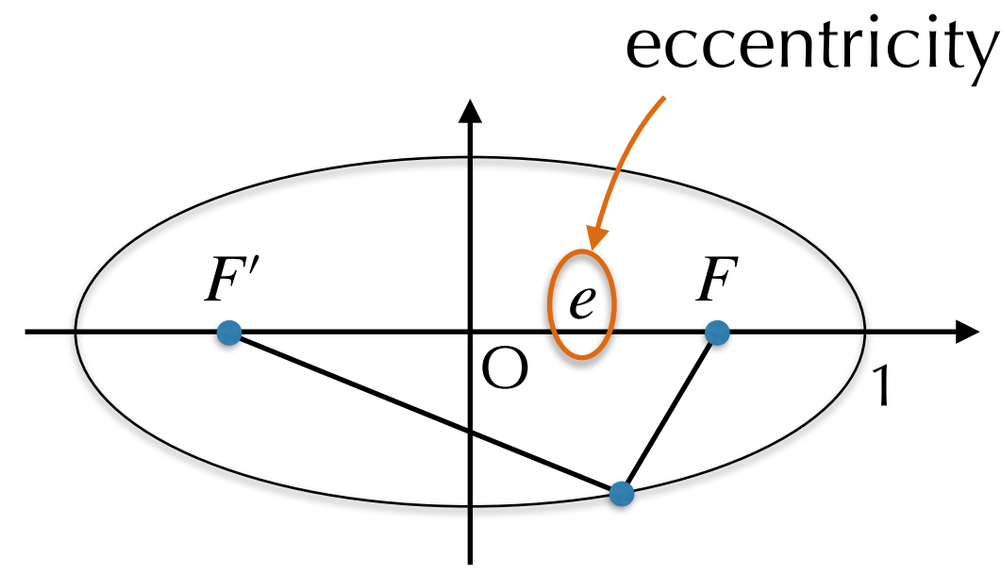
minimal pol. ≈30 Gb

G is algebraic!

Computer-driven discovery and proof

III. Examples of Symbolic Summation & Integration

Perimeter of an ellipse



$$p(e) = 2\int_{-1}^{1} \sqrt{\frac{1-e^2x^2}{1-x^2}}\, dx = ?$$

Algorithm in Lect. 11

$$e(1-e^2)p'' + (1-e^2)p' + ep = 0 \quad \text{(Euler 1733)}$$

$$\to p = 2\pi - \frac{\pi}{2}e^2 - \frac{3\pi}{32}e^4 + \cdots .$$

Irrationality of $\zeta(3)$

Thm. (Apéry 1978) $$\zeta(3) = \sum_{n=1}^{\infty} \frac{1}{n^3} \notin \mathbb{Q}\,.$$

Ingredient in the proof:

$$a_n = \sum_{k=0}^{n} \binom{n}{k}^2 \binom{n+k}{k}^2$$

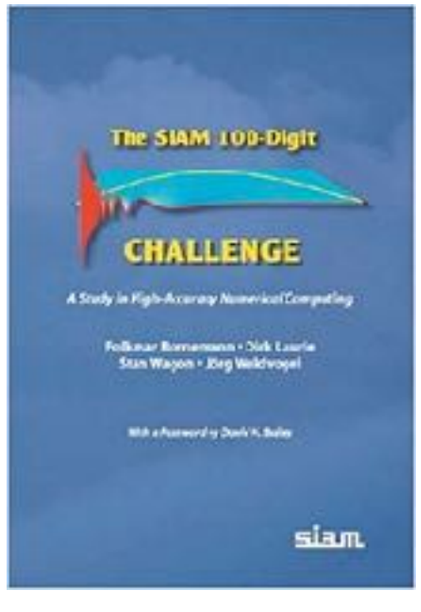
satisfies

Algorithm in Lect. 12

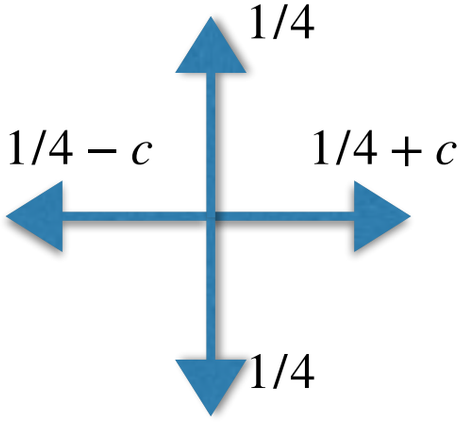
$$(*) \qquad (n+1)^3 a_{n+1} = (2n+1)(17n^2+17n+5)a_n - n^3 a_{n-1}\,.$$

Neither Cohen nor I had been able to prove (*) in the intervening two months.
A. van der Poorten

A numerical problem



Problem 6. *A flea starts at (0, 0) on the infinite two-dimensional integer lattice and executes a biased random walk: At each step it hops north or south with probability 1/4, east with probability 1/4 + c, and west with probability 1/4 − c. The probability that the flea returns to (0, 0) sometime during its wanderings is 1/2. What is c?*



Two quantities of interest: $\begin{cases} p(c) & := \text{Prob(return to 0)}, \\ q_n(c) & := \text{Prob(at 0 at step } 2n)\,. \end{cases}$

$$\mathbb{E}(\#\text{ returns}) = \sum_{k=1}^{\infty} k p(c)^k (1 - p(c)) = \frac{p(c)}{1 - p(c)} = \sum_{n=1}^{\infty} q_n(c)$$

Key: a binomial sum

Algorithm in Lect. 12

$$q_n(c) := \sum_{k=0}^{2n} \binom{2n}{2k} \binom{2k}{k} \binom{2n-2k}{n-k} \left(\frac{1}{4} + c\right)^k \left(\frac{1}{4} - c\right)^k \left(\frac{1}{4}\right)^{2n-2k}$$

Ready for computer algebra

Conclusion: Overview of the Course

0- Introduction — Today

1- Algorithms for polynomial matrices — G. Villard, We in Sept.

2- Algorithms for recurrence and differential equations — A. Bostan, Mo in Sept-Oct.

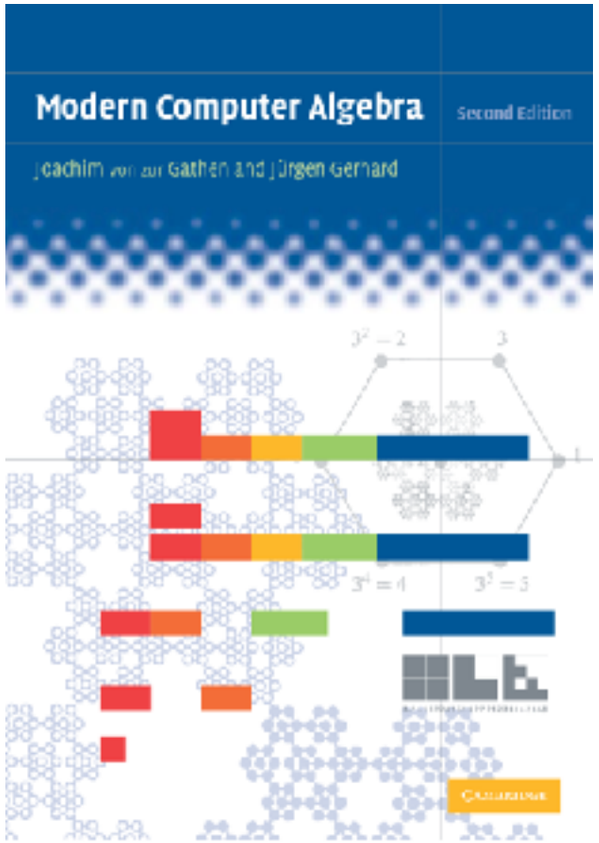
3- Algorithms for symbolic summation and integration — AB and BS, Mo-We Oct

Every week, at least one research result of the past 10 years.

References for this lecture

The slides are designed to be self-contained.

Here are books that I recommend if you want to learn more:



Feedback

Web site for the slides and references.

Questions or comments:

Alin.Bostan@inria.fr
Bruno.Salvy@inria.fr
Gilles.Villard@ens-lyon.fr