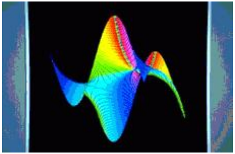
Differential Equations for Algebraic Functions

Bruno Salvy
Bruno.Salvy@inria.fr

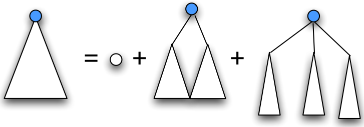
Algorithms Project, Inria



Joint work with A. Bostan, F. Chyzak, G. Lecerf & É. Schost

I Introduction

Example: Binary-Ternary Trees



c_N = number of trees with N nodes

Generating series:

$$\alpha = 1 + 2z + 10z^2 + 66z^3 + \cdots + c_N z^N + \cdots$$

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

More generally, context-free languages:

- their enumeration;
- their random generation.

Aim

$c_0, \ldots, c_N$ for large N.

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence

$$c_N = \sum_{i+j=N-1} c_i c_j + \sum_{i+j+k=N-1} c_i c_j c_k, \qquad N \geq 1.$$

Complexity: $O(N^3)$ ops

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $\alpha_{k+1} = 1 + z\alpha_k + z\alpha_k^3$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $\alpha_{k+1} = 1 + z\alpha_k + z\alpha_k^3$

$$\begin{aligned}
\alpha_0 &= 1 \\
\alpha_1 &= 1 + 2z \\
\alpha_2 &= 1 + 2z + 10z^2 + 16z^3 + 8z^4 \\
\alpha_3 &= 1 + 2z + 10z^2 + 66z^3 + 248z^4 + \dots \\
\alpha_4 &= 1 + 2z + 10z^2 + 66z^3 + 488z^4 + \dots
\end{aligned}$$

Complexity: $O(NM(N))$ ($M(N)$ for series product)

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78]

$$\alpha_{k+1} = \alpha_k - \frac{\alpha_k - (1 + z\alpha_k^2 + z\alpha_k^3)}{1 - 2z\alpha_k - 3z\alpha_k^2}$$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78]

$$\alpha_{k+1} = \alpha_k - \frac{\alpha_k - (1 + z\alpha_k^2 + z\alpha_k^3)}{1 - 2z\alpha_k - 3z\alpha_k^2}$$

$\alpha_0 = 1$

$\alpha_1 = 1 + 2z + 10z^2 + 50z^3 + \cdots$

$\alpha_2 = 1 + 2z + 10z^2 + 66z^3 + 498z^4 + 4066z^5 + 34970z^6 + 311042z^7 + \cdots$

Complexity: $O(M(N))$ ($M(N)$ for series product)

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78] $O(M(N))$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78] $O(M(N))$
4. Linear recurrence [Comtet 64, Chudnovsky[2] 86]
   1. linear differential equation [Abel 1827, Cockle 1861]

      $$2z(z-2)(z^2+11z-1)\alpha''+(3z^3+12z^2-89z+6)\alpha'-3(z+3)\alpha = z+3,$$

   2. translate

      $$(2n+6)(2n+7)c_{n+3} = (46n^2+227n+279)c_{n+2} - 3(6n^2+10n+3)c_{n+1} - n(2n+1)c_n.$$

Complexity: $O(N)$.

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78] $O(M(N))$
4. Linear recurrence [Comtet 64, Chudnovsky^2 86] $O(N)$

Computations for Binary-Ternary Trees

$$\alpha = 1 + z\alpha^2 + z\alpha^3.$$

1. Non-linear recurrence $O(N^3)$
2. Iterate $O(NM(N))$
3. Newton iteration [Kung & Traub 78] $O(M(N))$
4. Linear recurrence [Comtet 64, Chudnovsky^2 86] $O(N)$

Our Result

Even faster! (wrt degree)

Algorithms and Complexities

$$P(z,\alpha) = 0, \qquad \deg P = D$$

1. Non-linear recurrence $O(N^D)$
2. Iterate if $\alpha = P(z,\alpha)$: $O(\sqrt{D}NM(N))$ (baby steps/giant steps)
3. Newton iteration $O(\sqrt{D}M(N))$
4. Linear recurrence $O(D^{?}N)$.

Algorithms and Complexities

$$P(z, \alpha) = 0, \qquad \deg P = D$$

1. Non-linear recurrence $O(N^D)$
2. Iterate if $\alpha = P(z, \alpha)$: $O(\sqrt{D} NM(N))$
3. Newton iteration $O(\sqrt{D} M(N))$
4. Linear recurrence $O(D^? N)$.

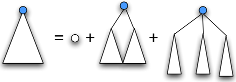
Theorem (BoChLeSaSc07)

1. *the recurrence computed through Cockle's differential equation leads to $O(D^2 M(D) N)$ ops;*
2. *there exist other recurrences leading to $O(DM(D)N)$ ops.*

Also, results in terms of D_z and D_y separately.

Nicer Recurrence on our Example

$$\alpha = 1 + z\alpha^2 + z\alpha^3$$



- Classical way:
  1. Linear differential equation [Abel 1827, Cockle 1861]

     $$2z(z-2)(z^2+11z-1)\alpha'' + (3z^3+12z^2-89z+6)\alpha' - 3(z+3)\alpha = z+3,$$

  2. translate

     $$(2n+6)(2n+7)c_{n+3} = (46n^2+227n+279)c_{n+2} - 3(6n^2+10n+3)c_{n+1} - n(2n+1)c_n.$$

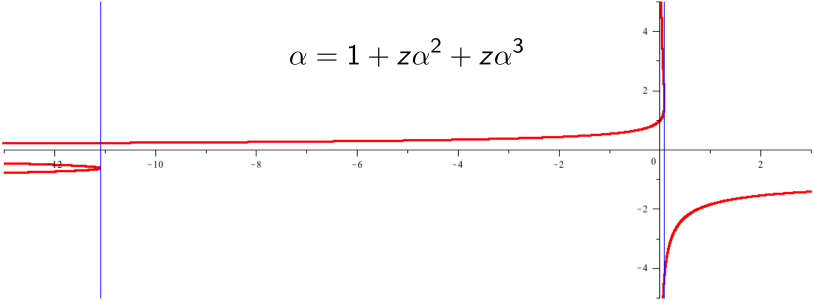
- Shorter recurrence:

$$(n+2)(2n+5)(5n+3)c_{n+2} = (110n^3+396n^2+445n+150)c_{n+1} + n(2n+1)(5n+8)c_n.$$

Questions

Minimal order for differential equation? for recurrence?
Minimal "size"? Efficiency?

Apparent Singularities Pollution



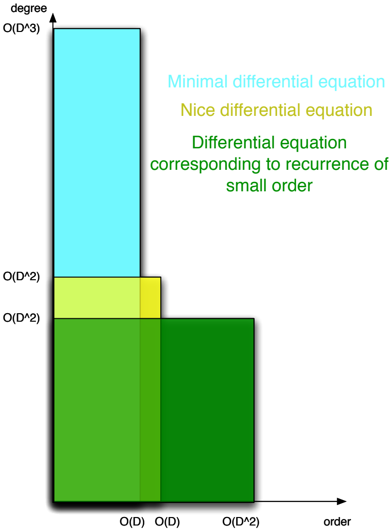
- Cockle's differential equation:

$$2z(z-2)(z^2+11z-1)\alpha''+(3z^3+12z^2-89z+6)\alpha'-3(z+3)\alpha = z+3,$$

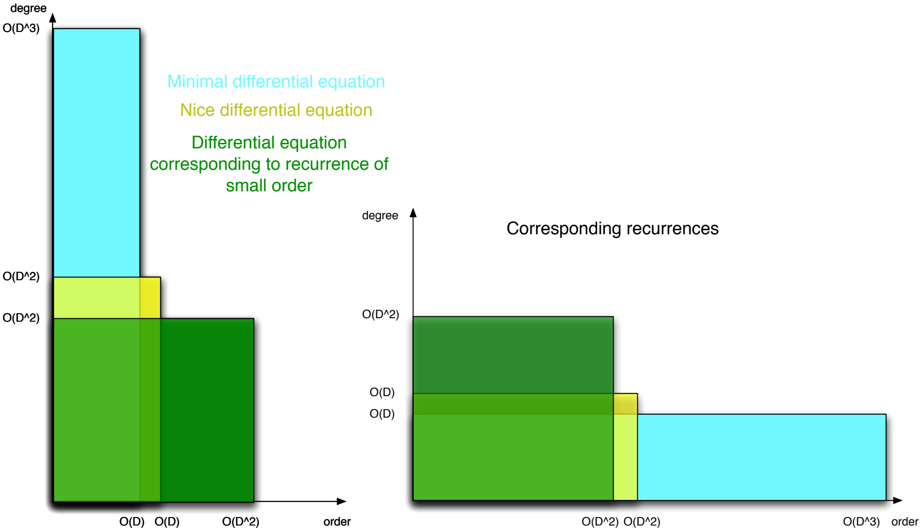
- differential equation associated to shorter recurrence:

$$10z(z^2+11z-1)\alpha''' - (2z^3-33z^2-442z+25)\alpha'' + \cdots = 0.$$

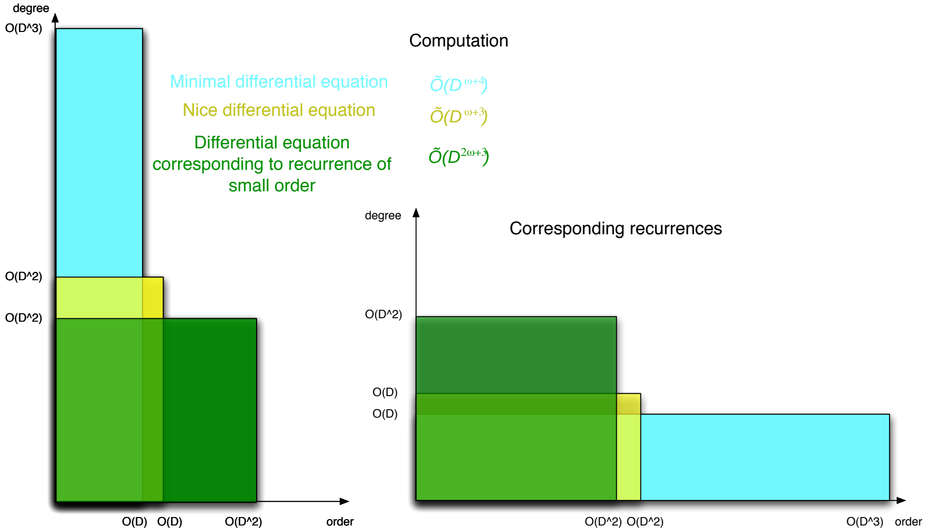
Our Results


degree
O(D^3)
O(D^2)
O(D^2)
O(D) O(D) O(D^2)
order
Minimal differential equation
Nice differential equation
Differential equation corresponding to recurrence of small order

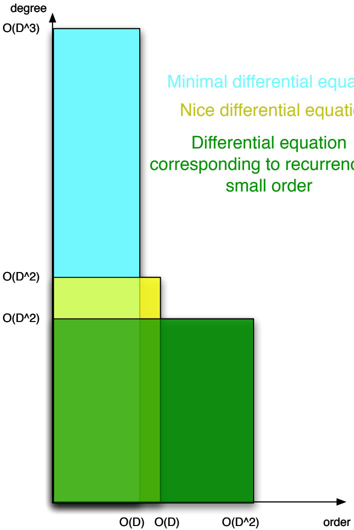
Our Results


degree
O(D^3)
O(D^2)
O(D^2)
O(D) O(D) O(D^2) order
Minimal differential equation
Nice differential equation
Differential equation corresponding to recurrence of small order
degree
Corresponding recurrences
O(D^2)
O(D)
O(D)
O(D^2) O(D^2) O(D^3) order

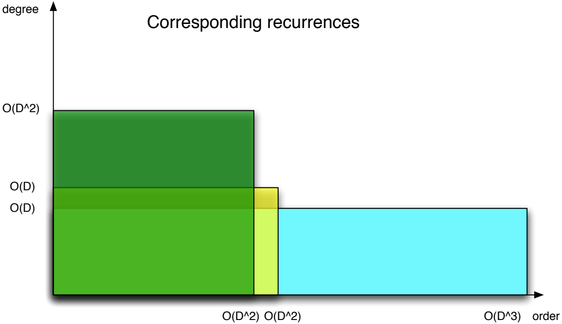
Our Results


degree
O(D^3)
O(D^2)
O(D^2)
O(D) O(D) O(D^2) order
Computation
Minimal differential equation $\tilde{O}(D^{\omega+4})$
Nice differential equation $\tilde{O}(D^{\omega+3})$
Differential equation corresponding to recurrence of small order $\tilde{O}(D^{2\omega+3})$
degree
Corresponding recurrences
O(D^2)
O(D)
O(D)
O(D^2) O(D^2) O(D^3) order

Our Results



	Computation	Unrolling the recurrence
Minimal differential equation	$\tilde{O}(D^{\omega+4})$	$\tilde{O}(D^2 M(D) N)$
Nice differential equation	$\tilde{O}(D^{\omega+3})$	$\tilde{O}(D M(D) N)$
Differential equation corresponding to recurrence of small order	$\tilde{O}(D^{2\omega+3})$	$\tilde{O}(M(D^3) N)$



II Algorithms

Cockle's Algorithm — Example

$$\alpha(z) - (1 + z\alpha^2(z) + z\alpha^3(z)) =: P(z,\alpha) = 0$$

$$\rightarrow \begin{cases} P_y(z,\alpha)\alpha'(z) + P_z(z,\alpha) = 0 \\ A(z,y)P(z,y) + B(z,y)P_y(z,y) = 1. \qquad \text{(Bézout)} \end{cases}$$

$$\alpha' = -BP_z \bmod P =: u_1\alpha^2 + v_1\alpha + w_1 1,$$

Vector space of dimension 3

Cockle's Algorithm — Example

$$\alpha(z) - (1 + z\alpha^2(z) + z\alpha^3(z)) =: P(z,\alpha) = 0$$

$$\rightarrow \begin{cases} P_y(z,\alpha)\alpha'(z) + P_z(z,\alpha) = 0 \\ A(z,y)P(z,y) + B(z,y)P_y(z,y) = 1. \qquad \text{(Bézout)} \end{cases}$$

$$\alpha' = -BP_z \bmod P =: u_1\alpha^2 + v_1\alpha + w_1 1,$$

$$\alpha'' = (u_1'\alpha^2 + v_1'\alpha + w_1') + (2u_1\alpha + v_1)\alpha' =: u_2\alpha^2 + v_2\alpha + w_2 1,$$

$$\alpha''' = (u_2'\alpha^2 + v_2'\alpha + w_2') + (2u_2\alpha + v_2)\alpha' =: u_3\alpha^2 + v_3\alpha + w_3 1.$$

$$\begin{pmatrix}\alpha & \alpha' & \alpha'' & \alpha'''\end{pmatrix} = \begin{pmatrix}\alpha^2 & \alpha & 1\end{pmatrix} \underbrace{\begin{pmatrix} 0 & u_1 & u_2 & u_3 \\ 1 & v_1 & v_2 & v_3 \\ 0 & w_1 & w_2 & w_3 \end{pmatrix}}_{A}$$

$V \in \ker A$ has for coordinates the coefficients of a differential equation.

Padé and Padé-Hermite approximants

Definition (Padé-Hermite Approximant)

The vector of polynomials $(P_1, \ldots, P_k)$ with $\deg P_i \leq d_i$ is a Padé-Hermite approximant of type $(d_1, \ldots, d_k)$ for a vector of power series $(f_1, \ldots, f_k)$ when

$$P_1 f_1 + \cdots + P_k f_k = O(x^{d_1 + \cdots + d_k + k - 1}).$$

Special cases: (given one series y)

- $k = 2$, $f_1 = -1$, $f_2 = y$: Padé approximant;
- $f_i = y^{i-1}$, $i = 1, \ldots, k$: algebraic approximants;
- $f_i = y^{(i-1)}$, $i = 1, \ldots, k$: differential approximants.

Padé and Padé-Hermite approximants

Definition (Padé-Hermite Approximant)

The vector of polynomials $(P_1, \dots, P_k)$ with $\deg P_i \leq d_i$ is a Padé-Hermite approximant of type $(d_1, \dots, d_k)$ for a vector of power series $(f_1, \dots, f_k)$ when

$$P_1 f_1 + \dots + P_k f_k = O(x^{d_1 + \dots + d_k + k - 1}).$$

Algorithms and complexity $(D = d_1 + \dots + d_k)$:

- Linear algebra: $O(D^\omega)$ ops;
- minimal basis in $O(k^\omega M(D))$ ops [Beckermann-Labahn94];
- genset in $O(k^\omega M(D/k))$ ops [Storjohann06].

Cockle's Algorithm via Truncated Series

Algorithm, non-degenerate case

1. Compute $\alpha^{(k)} = u_{k,1}1 + u_{k,2}\alpha + \cdots + u_{k,D}\alpha^{D-1}$ with power series coefficients $u_{k,i}$, for $k = 1, \ldots, D$;
2. find linear relation (diff. eqn) with power series coefficients (Newton in the linear algebra stage);
3. compute Padé approximants to recover rational coefficients.

[Chudnovsky^2 86, Cormier-Singer-Trager-Ulmer 02]

Complexity: $O(r^\omega M(\eta))$, η bound on degree coeffs.

Good bound → good algorithm

Differential Equation by Padé-Hermite Approximants

Algorithm

Input: P irreducible, order r and degree bound d;

1. $\sigma :=?$;
2. Compute a series expansion for α at precision σ (Newton);
3. Compute a Padé-Hermite approximant $(P_0, \dots, P_r)$ of type $(d, \dots, d)$ for $(\alpha, \dots, \alpha^{(r)})$;
4. return $L \cdot y = P_0 y + \cdots + P_r y^{(r)}$.

Good bounds $\rightarrow$ good algorithm

Differential Equation by Padé-Hermite Approximants

Algorithm

Input: P irreducible, order r and degree bound d;

1. $\sigma := ?$;
2. Compute a series expansion for α at precision σ (Newton);
3. Compute a Padé-Hermite approximant $(P_0, \dots, P_r)$ of type $(d, \dots, d)$ for $(\alpha, \dots, \alpha^{(r)})$;
4. return $L \cdot y = P_0 y + \cdots + P_r y^{(r)}$.

Good bounds $\rightarrow$ good algorithm

Lemma (Truncated Series $\rightarrow$ Full Series)

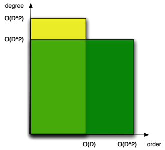
$P(x, \alpha) = 0, L \cdot \alpha = O(x^\sigma), \sigma \geq D(4Dr + d - r) \Rightarrow L \cdot \alpha = 0.$

1. $\alpha^{(k)} = W_k / P_y^{2k-1} \rightarrow$ a polynomial $Q(z, y)$ such that $L \cdot \alpha = 0$ iff $Q(z, \alpha) = 0$, together with degree bounds on Q.
2. The resultant $R(z)$ of P and Q w.r.t. y has degree $< \sigma$.
3. $R = O(x^\sigma) \Rightarrow R = 0 \Rightarrow P | Q$ (P irreducible).

III Bounds

Bounds by Creative Telescoping

$$\alpha(z) = \frac{1}{2\pi i} \oint \underbrace{\frac{yP_y(z,y)}{P(z,y)}}_{F(z,y)} dy.$$

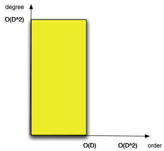


Creative telescoping: an algorithm for differentiation under $\int$ and integration by parts.

1. Find $\Lambda = A(z, \partial_z) + \partial_y B(z, \partial_z, y, \partial_y)$ s.t. $\Lambda \cdot F = 0$;
2. return A.

Bounds by Creative Telescoping

$$\alpha(z) = \frac{1}{2\pi i} \oint \underbrace{\frac{yP_y(z,y)}{P(z,y)}}_{F(z,y)} dy.$$



Creative telescoping: an algorithm for differentiation under $\int$ and integration by parts.

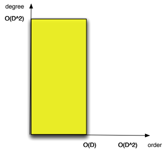
1. Find $\Lambda = A(z,\partial_z) + \partial_y B(z,\partial_z,y,\partial_y)$ s.t. $\Lambda \cdot F = 0$;
2. return A.

Bounds by counting dimensions

$$z^i \partial_z^j \partial_y^k \cdot F = \frac{Q}{P^{j+k+1}}, \qquad \deg Q \leq i + (j+k+1)D.$$

Bounds by Creative Telescoping

$$\alpha(z) = \frac{1}{2\pi i} \oint \underbrace{\frac{yP_y(z,y)}{P(z,y)}}_{F(z,y)} dy.$$



Creative telescoping: an algorithm for differentiation under $\int$ and integration by parts.

1. Find $\Lambda = A(z,\partial_z) + \partial_y B(z,\partial_z,y,\partial_y)$ s.t. $\Lambda \cdot F = 0$;
2. return A.

Bounds by counting dimensions

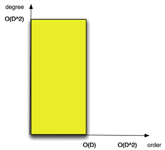
$$z^i \partial_z^j \partial_y^k \cdot F = \frac{Q}{P^{j+k+1}}, \qquad \deg Q \le i + (j+k+1)D.$$

Taking $i \le N_z$, $j+k \le N_\partial$,

$$\dim(\text{lhs}) = (N_z+1)\binom{N_\partial+2}{2}, \quad \dim(\text{rhs}) = \binom{(N_\partial+1)D+N_z+2}{2}.$$

Bounds by Creative Telescoping

$$\alpha(z) = \frac{1}{2\pi i} \oint \underbrace{\frac{y P_y(z,y)}{P(z,y)}}_{F(z,y)} \, dy.$$



Creative telescoping: an algorithm for differentiation under $\int$ and integration by parts.

1. Find $\Lambda = A(z, \partial_z) + \partial_y B(z, \partial_z, y, \partial_y)$ s.t. $\Lambda \cdot F = 0$;
2. return A.

Bounds by counting dimensions

$$z^i \partial_z^j \partial_y^k \cdot F = \frac{Q}{P^{j+k+1}}, \qquad \deg Q \leq i + (j+k+1)D.$$

Taking $i \leq N_z$, $j + k \leq N_\partial$, $N_z = 4D^2$, $N_\partial = 4D$,

$$\dim(\text{lhs}) = (N_z+1)\binom{N_\partial + 2}{2} > \dim(\text{rhs}) = \binom{(N_\partial+1)D + N_z + 2}{2}.$$

$\rightarrow$ Recurrence of order $\leq 4(D^2 + D)$, coeffs of deg $\leq 4D$.

IV Conclusion

Conclusion

- Summary: Good bounds + Newton + Padé or Padé-Hermite approximants = good algorithms;
- Also in the paper:
  1. Bounds in terms of D, D_x, D_y simultaneously;
  2. Fast heuristic algorithms using these bounds;
  3. Experiments and conjectures on bounds in generic cases;
  4. Lower bounds;
  5. Degenerate cases;
  6. Handling of algebraic extensions.
- Future:
  1. Other cases of creative telescoping (smaller certificates? better efficiency?);
  2. Bit complexity;
  3. Positive characteristic.

Recurrence Unrolling

Problem

Given initial conditions and $p_k(n)u_{n+k} + \cdots + p_0(n)u_n = 0$, with p_i's of degree d, compute $u_0, \ldots, u_N$ efficiently for N large.

Direct: $O(Nkd)$ ops; Better: $O(NkM(d)/d)$.

$\rightarrow$ The degree of the coefficients does not matter (much).

Recurrence Unrolling

Problem

Given initial conditions and $p_k(n)u_{n+k} + \cdots + p_0(n)u_n = 0$, with p_i's of degree d, compute $u_0, \dots, u_N$ efficiently for N large.

Direct: $O(Nkd)$ ops; Better: $O(NkM(d)/d)$.

$\rightarrow$ The degree of the coefficients does not matter (much).

Algorithm: Fast Evaluation of $P(x)$ on $0, \dots, N$ [Bostan *et alii* 07]

Idea: expand generating series $\mathcal{P}(z) = \sum_{k \geq 0} P(k)z^k = \dfrac{Q(z)}{(1-z)^{d+1}}$.

1. Compute $S(z) := (1-z)^{-d-1} \bmod z^d$;
2. Compute N/d times
$$\frac{A(z)}{(1-z)^{d+1}} = \underbrace{b_0 + \cdots + b_{d-1}z^{d-1}}_{B(z)} + \frac{z^d C(z)}{(1-z)^{d+1}}$$
by $B := AS \bmod z^d$; $z^d C := A - B(1-z)^{d+1}$.

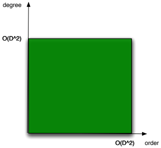
Timings

Cockle's algorithm over $F = \mathrm{GF}_{9973}$ for random dense polynomials with $D_z = D_y = N$:

N	ser.	rat.	η	$\deg_z$
1	.002	.002	2	2
2	.003	.004	17	10
3	.02	.03	69	36
4	.10	.16	182	92
5	.47	.98	380	190
6	1.86	4.56	687	342
7	5.80	16.5	1127	560
8	15.5	49.9	1724	856
9	38.0	138	2502	1242
10	72.7	340	3485	1730

- Always faster than Magma's built-in routine (rat.).
- Bound η off by a factor 2?

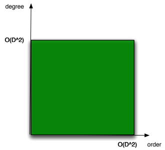
Better Bound on Order Using y



Aim

$F = yP_y/P$, we want: $A(z, \partial_z) \cdot F = \partial_y \cdot G$, $A \neq 0$.

Better Bound on Order Using y



Aim

$F = yP_y/P$, we want: $A(z, \partial_z) \cdot F = \partial_y \cdot G$, $A \neq 0$.

1. Decompose $P =: \tilde{P} + R$, with $\deg \tilde{P} = D$, $\deg R < D$;
2. Populate $V_d := \{Q/P^{d+1} \mid \deg Q \leq Dd + D + d\}$ with
   - $F_d := \mathrm{Vect}(\{z^i (z\partial_z)^j \cdot F \mid i, j \leq d\})$;
   - $H_d := \partial_y \cdot \mathrm{Vect}(\{\frac{cz^{d+1}\tilde{P}^d + H}{P^d} \mid \deg H \leq dD + d\})$.
3. Count dimensions:

$$\dim F_d = (d+1)^2; \quad \dim H_d = \binom{dD+d+2}{2} + 1 - \underbrace{(d+1)}_{\text{kernel}};$$

$$\dim V_d = \binom{Dd+D+d+2}{2}.$$

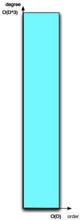
4. Conclude: $d \geq D^2 + D \Rightarrow \dim F_d + \dim H_d > \dim V_d$.

$\rightarrow$ Recurrence of order $\leq D^2 + D$.

History

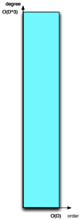
- Abel (1827): Existence of linear differential equation;
- Cockle (1861–1862): Algorithm for linear differential equation of minimal order;
- Harley (1862): Name "Differential resolvent";
- Tannery (1875): Rediscovery of Cockle's method;
- Comtet (1964): Application to series expansion (by hand);
- Chudnovsky^2 (1986): Complexity point of view;
- Cormier, Singer, Trager, Ulmer (2002): $\rightsquigarrow$ Degree bound in $O(D^5)$ for the differential resolvent;
- Nahay (2003–2004): Deg. bound in $O(D^3)$ for α^λ with $\lambda \notin \bar{\mathbb{Q}}$;
- Tsai (2000): Weyl closure $\rightsquigarrow$ removal of apparent singularities.

Degree bounds for the differential resolvent



$$\mathrm{Wr}(\alpha_1,\dots,\alpha_r,\alpha) = \begin{vmatrix} \alpha_1 & \dots & \alpha_r & \alpha \\ \alpha_1' & \dots & \alpha_r' & \alpha' \\ \vdots & & \vdots & \vdots \\ \alpha_1^{(r)} & \dots & \alpha_r^{(r)} & \alpha^{(r)} \end{vmatrix} = 0.$$

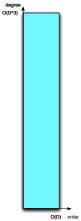
Degree bounds for the differential resolvent



$$\mathrm{Wr}(\alpha_1,\dots,\alpha_r,\alpha) =$$

$$\begin{vmatrix} \alpha_1 & \dots & \alpha_r & \alpha \\ \frac{W_1(z,\alpha_1)}{P_y(z,\alpha_1)} & \dots & \frac{W_1(z,\alpha_r)}{P_y(z,\alpha_r)} & \alpha' \\ \vdots & & \vdots & \vdots \\ \frac{W_r(z,\alpha_1)}{P_y(z,\alpha_1)^{2r-1}} & \dots & \frac{W_r(z,\alpha_r)}{P_y(z,\alpha_r)^{2r-1}} & \alpha^{(r)} \end{vmatrix} = 0.$$

Degree bounds for the differential resolvent



$$\mathrm{Wr}(\alpha_1,\dots,\alpha_r,\alpha)\prod_i P_y(z,\alpha_i)^{2r-1} =$$

$$\begin{vmatrix} P_y(z,\alpha_1)^{2r-1}\alpha_1 & \dots & P_y(z,\alpha_r)^{2r-1}\alpha_r & \alpha \\ W_1(z,\alpha_1)P_y(z,\alpha_1)^{2r-2} & \dots & W_1(z,\alpha_r)P_y(z,\alpha_r)^{2r-2} & \alpha' \\ \vdots & & \vdots & \vdots \\ W_r(z,\alpha_1) & \dots & W_r(z,\alpha_r) & \alpha^{(r)} \end{vmatrix} = 0.$$

Degree bounds for the differential resolvent

$$L := \frac{1}{\prod_{1\le i<j\le r}(\alpha_i-\alpha_j)}\operatorname{Wr}(\alpha_1,\dots,\alpha_r,\alpha)\prod_i P_y(z,\alpha_i)^{2r-1} = \frac{1}{\prod_{1\le i<j\le r}(\alpha_i-\alpha_j)}\times$$

$$\begin{vmatrix} P_y(z,\alpha_1)^{2r-1}\alpha_1 & \dots & P_y(z,\alpha_r)^{2r-1}\alpha_r & \alpha \\ W_1(z,\alpha_1)P_y(z,\alpha_1)^{2r-2} & \dots & W_1(z,\alpha_r)P_y(z,\alpha_r)^{2r-2} & \alpha' \\ \vdots & & \vdots & \vdots \\ W_r(z,\alpha_1) & \dots & W_r(z,\alpha_r) & \alpha^{(r)} \end{vmatrix} = 0.$$

Degree bounds for the differential resolvent

$$L := \frac{1}{\prod_{1\le i<j\le r}(\alpha_i-\alpha_j)}\operatorname{Wr}(\alpha_1,\dots,\alpha_r,\alpha)\prod_i P_y(z,\alpha_i)^{2r-1} = \frac{1}{\prod_{1\le i<j\le r}(\alpha_i-\alpha_j)}\times$$

$$\begin{vmatrix} P_y(z,\alpha_1)^{2r-1}\alpha_1 & \dots & P_y(z,\alpha_r)^{2r-1}\alpha_r & \alpha \\ W_1(z,\alpha_1)P_y(z,\alpha_1)^{2r-2} & \dots & W_1(z,\alpha_r)P_y(z,\alpha_r)^{2r-2} & \alpha' \\ \vdots & & \vdots & \vdots \\ W_r(z,\alpha_1) & \dots & W_r(z,\alpha_r) & \alpha^{(r)} \end{vmatrix} = 0.$$

- L is polynomial and symmetric in $\alpha_1,\dots,\alpha_r$;
- $\deg_z(k\text{th row}) = O(D^2)$, $\deg_{\alpha_i}(i\text{th col}) = O(D^2)$;
- $\Rightarrow$ if $r = D_y$, $\deg_z L = O(D^3)$;
- if $r < D_y$, symmetrize first wrt $\alpha_1,\dots,\alpha_{D_y}$.

Precise bounds (rather than $O()$) available, and necessary in algorithm