Ex: $a(x) = \frac{x}{1-e^x}$ is not D-finite because

$\frac{1}{a(x)} = \frac{1-e^x}{x}$ is and $\frac{a'(x)}{a(x)} = + \frac{e^x x - e^x + 1}{x(1-e^x)}$

is not.

One more closure property.

Thm:
(1) If y satisfies a deq of order r, then also $f = p_0 y + \dots + p_m y^{(m)}$ with $p_0, \dots, p_m \in C(x)$ satisfies a deq of order $\leq r$.
(2) If (a_n) satisfies a rec of order r, then also $b_n := p_0(n) a_n + \dots + p_m(n) a_{n+m}$ with $p_i \in C(x)$ satisfies a rec of order $\leq r$.

Proof:
(1) Using the deq for y, each $f^{(i)}$ can be written as a $C(x)$-linear combination of $y, \dots, y^{(r-1)}$. Thus $f, \dots, f^{(r)}$ are linearly dependent.
(2) Same trick. $\boxtimes$

4 Guessing

Task: Given $a_0, a_1, \dots, a_N \in \mathbb{C}$, find a recurrence $p_0(n)a_n + \dots + p_r(n)a_{n+r} = 0$ $(n = 0 .. N-r)$ or a deq $p_0(x)a(x) + \dots + p_r(x)a^{(r)}(x) = O(x^{N-r})$ for $a(x) = \sum_{n=0}^{N} a_N x^N$.

Why? If $a_0 \dots a_N$ are the first terms of an infinite sequence and we find eqns that are relatively small (in terms of order/degree/coeffs) then we can guess that these eqns may be valid not only for the truncated seq/fps but for the whole (infinite) object. This is a way to test experimentally whether such is (with high probability) D-finite.

(In contrast, closure properties yield eqns which are provably correct.)

Depending on the context, it may be possible to prove a guessed equation by independent means.

Ex: Equations obtained by closure properties operations may not have minimal possible order (cf earlier discussion on alg fns.) You can guess a shorter one and use closure properties to prove that it is correct:

$\to$ given $\bigcirc y + \bigcirc y' + \bigcirc y'' + \bigcirc y''' = 0$ (1)

guess $\bigcirc y + \bigcirc y' + \bigcirc y'' \overset{?}{=} 0$ (2)

def $f := \bigcirc y + \bigcirc y' + \bigcirc y''$ (3)

compute $\bigcirc f + \bigcirc f' + \bigcirc f'' + \bigcirc f''' = 0$ (4)

from (1) and (3) by closure props

check suff many initvals to conclude that $f = 0$. This proves (2).

How to guess? Ansatz and linear algebra over $\mathbb{C}$

(1) choose r, d

(2) ansatz $\sum_{i=0}^{r} \sum_{j=0}^{d} p_{ij} n^j a_{n+i} \overset{!}{=} 0$ with unknown coeffs p_{ij}

(3) for each choice $n \in \{0 \dots N-r\}$ the

ansatz specializes to a linear constraint for the unknowns p_{ij}

$$\begin{array}{r l}
n=0 & \\
n=1 & \\
n=2 & \\
\end{array}
\begin{pmatrix} \text{———} \\ \text{———} \\ \text{———} \\ \vdots \end{pmatrix} \cdot \begin{pmatrix} p_{00} \\ \vdots \\ p_{rd} \end{pmatrix} = 0$$

(4) solve the system and return the eqns corresponding to its solutions.

<u>Remarks:</u>

(1) Works similarly for deqs.

(2) For recs, the linear system has $N-r+1$ eqns and $(r+1)(d+1)$ vars, so it is overdetermined when

$$N-r+1 \geq (r+1)(d+2) \iff N+1 > (r+1)(d+2).$$

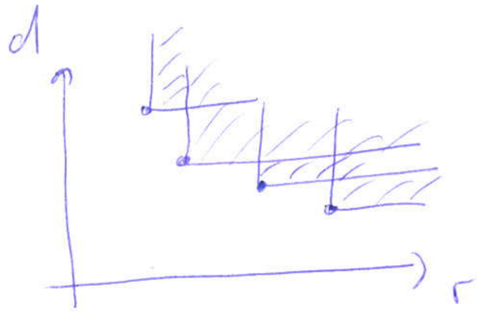
For such systems we do not expect a solution. If there is one, it's probably there for a reason.

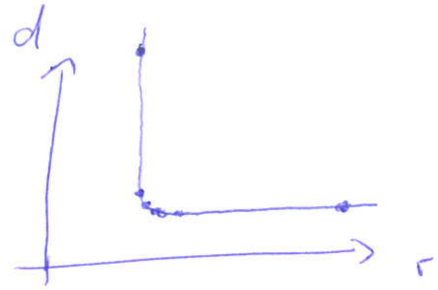
We can increase the confidence by increasing $(N+1)-(r+1)(d+2)$ (i.e. by increasing N)

(3) a guessed equation may be wrong, but if on the other hand the system has no solution for a particular choice of (r, d), then we know for sure that the seq/fps does not satisfy any eqn of this shape. (Larger equations may still exist though)

(4) If there is an equation of size (r, d), then also of size $(r+1, d)$ (replace n by $n+1$) and of size $(r, d+1)$ (multiply by n). These extra eqns are not interesting but slow down the computation, so always try to choose (r, d) such that no eqns exist for $(r-1, d)$ or $(r, d-1)$.

(5) The pair (r, d) for which an eq exists while none exist for $(r-1, d)$ or $(r, d-1)$ is usually not unique.





Most interesting in practice are those where r is minimal. Guessing works best for those where $r \cdot d$ is minimal.

(6) If char $C = 0$ (eg $C = \mathbb{Q}$) be sure to use chinese remaindering and rational reconstruction to deal with expression swell. True relations tend to have relatively short coeffs, so we can recover them with relatively few primes. Wrong solutions and intermediate expressions are usually much longer, so that there is a lot to gain by using homomorphic images.

(7) The cost of guessing is $O(N^\omega)$ ops in C. We will choose $N \approx rd$. In applications we often observe $d \approx r^2$, so the cost is $\approx r^{3\omega}$. Can we do better? Yes.

For simplicity, let's consider only the differential case.

<u>Observation:</u> Given $f_0, \dots, f_r \in C[[x]]$ and $N \in \mathbb{N}$, the set of all relations

$$p_0 f_0 + \dots + p_r f_r = O(x^N)$$

not only forms a C-VS but also a $C[x]$-module.

<u>Idea:</u> Compute a basis of this module in which the generators have smallest possible degree. A generator with

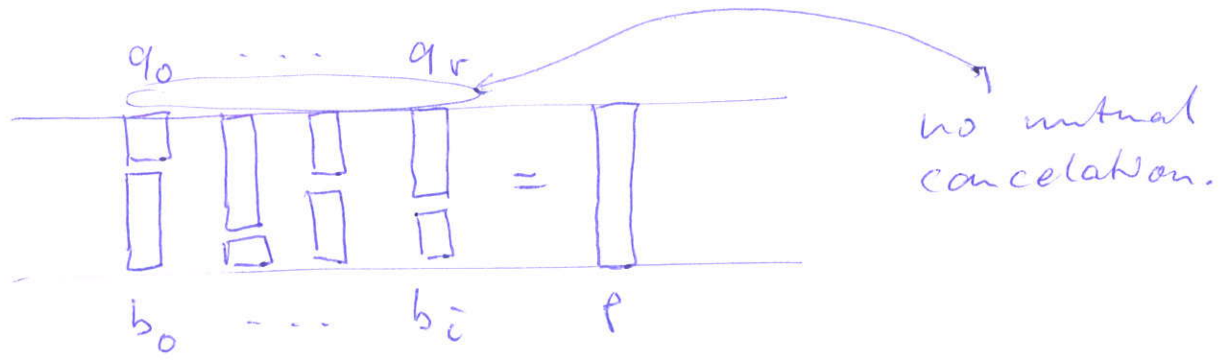
exceptionally low degree is a reasonable candidate for a relation valid for all N.

<u>Def:</u> Given $\sigma \in \mathbb{N}$, a basis $\{b_0, \dots, b_r\} \subseteq \mathbb{C}[x]^{r+1}$ is called a σ-basis for $f = (f_0, \dots, f_r) \in \mathbb{C}[[x]]^{r+1}$ if (a) $b_i \cdot f = O(x^\sigma)$ for all i and (b) for all $p \in \mathbb{C}[x]^{r+1}$ there exists a unique $q \in \mathbb{C}[x]^{r+1}$ with $q = (q_0 \dots q_r)$

$$\deg q_i + \deg b_i \leq \deg p \qquad (i = 0 \dots r)$$

and $\quad q_1 b_1 + \dots + q_r b_r = p$.

Here, $\deg(p_0 \dots p_r) := \max_{i=0}^{r} \deg p_i$.



<u>Alg:</u> (Hermite-Padé-Approximation)

Input: $\sigma \in \mathbb{N}$, $f = (f_0, \dots, f_r) \in C[[x]]^{r+1}$

st coeffs of $x^0 \dots x^{\sigma-1}$ are known

for each f_i

Output: A σ-bases $\{b_0 \dots b_r\}$ for f.

(1) Set $b_i = e_i \in C[x]^{r+1}$ $(i = 0 \dots r)$

(2) for $s = 0 \dots \sigma - 1$ do

(3) set $c_i = [x^s](b_i \circ f)$ $(i = 0 \dots r)$

(4) Set $L = \{e \mid c_e \neq 0\}$

(5) if $L \neq \emptyset$ do

(6) choose $e \in L$ with $\deg b_e$ minimal.

(7) for $i \in L \setminus \{e\}$ set $b_i = b_i - \frac{c_i}{c_e} b_e$

(8) set $b_e = x b_e$

(9) return $\{b_0 \dots b_r\}$

Correctness is not directly obvious but can be checked by a straightforward proof by induction on s.

<u>Ex:</u>

s	$\deg b_0$	$\deg b_1$	$\deg b_1$	$\operatorname{ord}(b_0 \circ f)$	$\operatorname{ord}(b_1 \circ f)$	$\operatorname{ord}(b_2 \circ f$
init	0	0	0	0	0	0
0	1	0	0	1	1	1
1	1	1	0	2	2	2
2	1	1	1	3	3	3
3	2	1	1	4	4	4
4	2	2	1	5	5	≥ 10
5	3	2	1	6	6	≥ 10
6	3	3	1	7	7	≥ 10
7	4	3	1	8	8	≥ 10
8	4	4	1	9	9	≥ 10
9	5	4	1	10	10	≥ 10

In this case, b_2 is probably a correct equation.

<u>Cost:</u> $O(r\sigma(r+\sigma)) \overset{\sigma = N - r \geq r}{=} O(rN^2)$ ops in C

$= O(r^7)$ with $d \approx r^2$

That's better than $O(r^{3\omega})$ for realistic values of ω. (though not ~~ffor~~ the theoretically best known values)

There is a divide and conquer variant of the algorithm with cost $O(r\,M(rN)\log N)$ where $M(n)$ is the time to multiply two polynomials in $\mathbb{C}[x]$ of degree $\leq n$.
(Fact: $M(n) = O(n \log n \log\log n)$)
With $N \approx rd$ and $d \approx r^2$, this is just $O(r^5 \log r)$ operations in $\mathbb{C}$. (world record)