Faster Algorithms for Sparse Decomposition and Sparse Series Solution to Differential Equations

4/29/22

Saiyue Lyu
Master of Mathematics in Computer Science Thesis Presentation

Supervisors : Mark Giesbrecht, Arne Storjohann

UNIVERSITY OF WATERLOO

What are Sparse Polynomials?

- Polynomials with relatively few nonzero terms compared to their degrees,

  e.g., $f = 3x^{100} + 5x^{40} + x^2$.

- Given a sparse polynomial $f = 3x^{100} + 5x^{40} + x^2$:

  $f = 3x^{100} + 5x^{40} + x^2 \quad \rightarrow \quad \tau(f) = 3,$ the sparsity of f;

  $f = 3x^{100} + 5x^{40} + x^2 \quad \rightarrow \quad 100, 40, 2,$ the exponents of f;

  $f = 3x^{100} + 5x^{40} + x^2 \quad \rightarrow \quad x^{100}, x^{40}, x^2,$ the supports of f.

- The sparse representation of f : a set of coefficients and exponents :

$$\{(3,100), (5,40), (1,2)\},$$

  has size $O(\tau(f) log n)$, where n is the degree of f.

Why Sparse Polynomials?

- Very common in computing systems in practice.
- Degree can be exponentially larger than the bit length of the representation.
- Some efficient algorithm for dense polynomials can take exponential operations when change to sparse case.
- **Computing with sparse polynomials can be hard!**
- Some problems are unknown to be NP or NP-complete, e.g., sparse polynomial divisibility.

Some problems are NP-hard, e.g., gcd of sparse polynomials.

Polynomial Perfect Power Problem can be tackled:)

This Presentation

- Solve the Polynomial Perfect Power Problem in dense setting
- Solve the Polynomial Perfect Power Problem in sparse setting
- Solve Sparse Polynomial Decomposition inspired by the Polynomial Perfect Power Problem
- Generalize the Polynomial Perfect Power Problem to Differential Equations

Solve the Polynomial Perfect Power Problem in dense setting

Solve the Polynomial Perfect Power Problem in sparse setting

Solve Sparse Polynomial Decomposition inspired by the Polynomial Perfect Power Problem

Generalize the Polynomial Perfect Power Problem to Differential Equations

Polynomial Perfect Power Problem

Let F be any finite field. Given $f = \sum_{i=0}^{n} f_i x^i \in F[x]$ with $\deg(f) = n, f_n \neq 0$, suppose that $n = rs$ for some $r, s \in \mathbb{N}$, determine whether there exists $h \in F[x]$ with $\deg(h) = s$ such that $f = h^r$.

- $n = rs$, the degree of the input f;
- s, the degree of the output h;
- r, the desired perfect power;
- $[x^k]f = f_k$, the kth coefficient of f;
- $[x^k]h = h_k$, the kth coefficient of h;

In the dense setting

$$f = h^r$$

A private observation by Koiran(2011)

$$f'h = rfh'$$

$$[x^k](f'h) = \sum_{i=0}^{k} (k+1-i) f_{k+1-i} h_i \quad = \quad [x^k](rfh') = \sum_{i=0}^{k} r(i+1) f_{k-i} h_{i+1}$$

$$\boxed{\boxed{r} f_0 k h_k = \sum_{i=0}^{k-1} (k-i-ri) f_{k-i} h_i}$$

$char(F)$ does not divide r → the "tame" case

$char(F)$ divides r → the "wild" case

the "wild" case $\xrightarrow{\text{reduced to}}$ the "tame" case

$f_0 \neq 0$:

$$h_k = \frac{1}{r \boxed{f_0} k} \sum_{i=0}^{k-1} (k-i-ri) f_{k-i} h_i$$

$f_0 = 0$:

$$h_{s-k} = \frac{1}{r \boxed{f_n} k} \sum_{i=0}^{k-1} (k-i-ri) f_{n-k+i} h_{s-i}$$

In the dense setting

$$f = h^r$$

A Combinatorial Proof
Section 3.1.3

$$\boxed{\boxed{r} f_0 k h_k = \sum_{i=0}^{k-1} (k - i - ri) f_{k-i} h_i}$$

$char(F)$ does not divide r

$char(F)$ divides r

the "tame" case, $\xleftarrow{\text{reduced to}}$ the "wild" case

$f_0 \neq 0$

$$h_k = \frac{1}{r \boxed{f_0} k} \sum_{i=0}^{k-1} (k - i - ri) f_{k-i} h_i$$

$f_0 = 0$

$$h_{s-k} = \frac{1}{r \boxed{f_n} k} \sum_{i=0}^{k-1} (k - i - ri) f_{n-k+i} h_{s-i}$$

UNIVERSITY OF WATERLOO

Solve the “Wild” Case

Following *von zur Gathen (1990),*

- the “tame” case : $\mathrm{char}(F)$ does not divide r, i.e., $r \neq 0$
- the “wild” case : $\mathrm{char}(F)$ divides r

- Say $\mathrm{char}(F) = p$, $r = p^{\alpha} \cdot q$, for some integer α and $\gcd(p, q) = 1$.
- Recall properties for $a, b, c \in F, \alpha \geq 1$, we have :

$$(a+b)^{p^{\alpha}} = a^{p^{\alpha}} + b^{p^{\alpha}}, \qquad (a+b+c)^{p} = a^{p} + b^{p} + c^{p}$$

- Then $f = h^{r} = (h_0 + h_1 x + \cdots h_s x^{s})^{p^{\alpha} \cdot q}$

$$= \left(h_0^{p^{\alpha}} + h_1^{p^{\alpha}} x^{1 \cdot p^{\alpha}} + \cdots + h_s^{p^{\alpha}} x^{s \cdot p^{\alpha}}\right)^{q}$$

$$= \left(\widetilde{h_0} + \widetilde{h_1} x^{1 \cdot p^{\alpha}} + \cdots + \widetilde{h_s} x^{s \cdot p^{\alpha}}\right)^{q} = \tilde{h}^{q}, \qquad p \text{ does not divide } q$$

- **The “wild” case can be reduced to the “tame” case.**

UNIVERSITY OF WATERLOO

Can we directly move to the sparse setting?

$$h_k = \frac{1}{r f_0 k} \sum_{i=0}^{k-1} (k - i - ri) f_{k-i} h_i$$

$$h_{s-k} = \frac{1}{r f_n k} \sum_{i=0}^{k-1} (k - i - ri) f_{n-k+i} h_{s-i}$$

$\Rightarrow$ $O(s^2)$

- Need to compute for each k;
- Cost in terms of degree instead of sparsity;
- How to find the next nonzero exponent?

- Solve the Polynomial Perfect Power Problem in dense setting
- Solve the Polynomial Perfect Power Problem in sparse setting
- Solve Sparse Polynomial Decomposition inspired by the Polynomial Perfect Power Problem
- Generalize the Polynomial Perfect Power Problem to Differential Equations

Sparse Polynomial Perfect Power Problem

Let F be any finite field. Given sparse $f = \sum_{i=0}^{n} f_i x^i \in F[x]$ with $\deg(f) = n, f_n \neq 0$, suppose that $n = rs$ for some $r, s \in \mathbb{N}$, determine whether there exists $h \in F[x]$ with $\deg(h) = s$ such that $f = h^r$, if so, find h in a manner whose costs is polynomial $logn, \tau(f), \tau(h)$.

- $\tau(f)$, the sparsity of f;
- $\tau(h)$, the sparsity of h;
- Polynomial time in input sparsity and output sparsity.

Some Background

- With previous work by *Erdös(1949), Schinzel(1987) and Zannier(2007,2008,2009),* we can assume that if f is sparse, then h is also sparse.
- *Giesbrecht and Roche(2008,2011)* presented an algorithm using kind of Newton iteration in Poly$(logn, \tau(f), \tau(h))$.
- This thesis will give a faster algorithm.

Using the modular trick

- Suppose we have computed $h_0, \cdots, h_{k-1}$, define :

$$\check{h} = \sum_{i=0}^{k-1} h_i x^i, \quad \hat{h} = \sum_{i=k-1}^{s} h_i x^{i-(k+1)};$$

- Then $h = \check{h} + h_k x^k + \hat{h}\, x^{k+1}$.
- Have

$$\begin{aligned} 0 &= f'h - rfh' \\ &\equiv f'\check{h} - rf\check{h}' - rkfh_k x^{k-1} \quad mod\ x^k \\ &\equiv f'\check{h} - rf\check{h}' - rkf_0 h_k x^{k-1} \quad mod\ x^k \\ &\equiv Rx^{k-1} \qquad mod\ x^k \end{aligned}$$

- Rx^{k-1} is the residual or error of the current step, this is a “Newton-like” method;
- When $f_0 \neq 0$, we can have $h_k = R/(rf_0 k)$.

UNIVERSITY OF WATERLOO

A Fast Algorithm

Algorithm 3: SparsePerfectPowerNonzeroConstAttempt

Input: sparse polynomial $f \in \mathsf{F}[x]$ of degree n s.t. $f_0 \neq 0$ and integer $s, r \in \mathbb{Z}$ s.t. $n = rs$

Output: sparse polynomial $h \in \mathsf{F}[x]$ such that $f = h^r$

$h_0 = f_0^{1/r}$;
$h = h_0$;
$k = 1$;
while degree$(h) < s$ **do**
    $\boxed{res = f'h - rfh';}$
    $k = \text{ldegree}(res) + 1$;
    $h_k = \frac{1}{rf_0k} \cdot \text{coeff}(res, k-1)$;
    $h += h_k \cdot x^k$;
end
return h;

$$O(\tau(f)\tau(h)^2)$$

A little bit faster

- Modify the computation of **res**, compare how **res** changes in each iteration;
- Use **hlo** to note h in the last iteration, have :

$$res = f' \cdot \left(hlo + h_k x^k\right) - rf\left(hlo + h_k x^k\right)'$$

$$= res' + \boxed{f' h_k x^k - rfkh_k x^{k-1}}$$

- Avoid the multiplication of two polynomials.

The Faster Algorithm

Algorithm 4: SparsePerfectPowerNonzeroConst

Input: sparse polynomial $f \in \mathsf{F}[x]$ of degree n s.t. $f_0 \neq 0$ and integer $s, r \in \mathbb{Z}$ s.t. $n = rs$

Output: sparse polynomial $h \in \mathsf{F}[x]$ such that $f = h^r$

```
h_0 = f_0^{1/r};
h = h_0;
res = f' h_0;
k = 1;
while ldegree(res) < s + 1 do
    k = ldegree(res) + 1;
    h_k = coeff(res, k − 1)/(r f_0 k);
    h += h_k · x^k;
    res += f' · h_k x^k − r k f · h_k x^{k−1};
end
return h;
```

$$O(\tau(f)\tau(h))$$

What if $f_0 = 0$?

- Find the lowest exponents of f, say d, then $f = \bar{f}x^d$ for some $\bar{f} \in F[x], \bar{f_0} \neq 0$.
- **$f_0 = 0$ can be reduced to the $f_0 \neq 0$ case**, the cost would be the same :

Algorithm 5: SparsePerfectPowerZeroConst

Input: sparse polynomial $f \in \mathsf{F}[x]$ of degree n s.t. $f_0 = 0$ and integer $s, r \in \mathbb{Z}$ s.t. $n = rs$

Output: sparse polynomial $h \in \mathsf{F}[x]$ such that $f = h^r$

$d = \mathrm{ldegree}(f)$;

$\boxed{\bar{f} = f / x^d;}$

$\bar{h} =$ Algorithm 4_SparsePerfectPowerNonzeroConst$(\bar{f}, r, s)$;

$h = \bar{h} \cdot x^{d/r}$;

return h;

So far

- We solved the Sparse Polynomial Perfect Power Problem, no matter “tame” or “wild” case, $f_0 = 0$ or $f_0 \neq 0$ case.

Remark

- Algorithm 4&5 actually compute the root assuming that $f = h^r$;
- But if not, the algorithms still output something:)
- That is actually a solution to a more general problem : $f = h^r \ mod \ x^s$
- To sufficiently check whether $f = h^r$, we can check whether $f'h = rh'f$ and $lc(f) = lc(h)^r$, the multiplication is much faster than computing the power. The cost would be $O(\tau(f)\tau(h))$. (*Giesbrecht and Roche 2008*)
- In the thesis, we also discuss the perfect power for rational functions.

This Presentation

- Solve the Polynomial Perfect Power Problem in dense setting
- Solve the Polynomial Perfect Power Problem in sparse setting
- Solve Sparse Polynomial Decomposition inspired by the Polynomial Perfect Power Problem
- Generalize the Polynomial Perfect Power Problem to Differential Equations

UNIVERSITY OF WATERLOO

Polynomial Decomposition Problem

Let F be any finite field. Given $f = \sum_{i=0}^{n} f_i x^i \in F[x]$ with $\deg(f) = n, f_n \neq 0$, suppose that $n = rs$ for some $r, s \in \mathbb{N}$, determine whether there exists $g, h \in F[x]$ with $\deg(g) = r$ and $\deg(h) = s$ such that $f = g \circ h$.

Reduction of *von zur Gathen(1990)*

- In the following, we will assume the "tame" case;
- If $f = g \circ h$ and α, β are the leading coefficients of f, h, then an affine linear transformation yields :

$$\bar{f} = \frac{f}{\alpha} = \frac{g(h)}{\alpha} = \frac{g\left(\beta \cdot \frac{h-h(0)}{\beta} + h(0)\right)}{\alpha} = \left(\frac{g\left(\beta x + h(0)\right)}{\alpha}\right) \circ \frac{h-h(0)}{\beta} = \bar{g} \circ \bar{h},$$

$$\bar{f} = \tfrac{f}{\alpha},\ \bar{g}(x) = \tfrac{1}{\alpha} \cdot g\left(\beta x + h(0)\right), \bar{h} = \tfrac{h-h(0)}{\beta}$$

- Thus, we can assume f, g, h are monic and $h(0) = 0$.

Observations of *von zur Gathen(1990)*

- Define the reversal of polynomials to be :

$$\tilde{f}(x) = x^n \cdot f\left(\frac{1}{x}\right), \quad \tilde{h}(x) = x^n \cdot h(\frac{1}{x})$$

von zur Gathen(1990) observed that :

- If $f = g(h(x))$, then $\tilde{f}(x) \equiv \tilde{h}(x)^r \; mod \; x^s$;
- Since deg $\left(\tilde{h}\right)$ =deg (h)=s and $h(0) = 0$, if such h exists it is unique.
- Polynomial Decomposition Problem can be solved via the computation of root h and a Taylor expansion for g, in the dense setting.

Sparse Polynomial Decomposition Problem

Let F be any finite field. Given sparse $f = \sum_{i=0}^{n} f_i x^i \in F[x]$ with $\deg(f) = n, f_n \neq 0$, suppose that $n = rs$ for some $r, s \in \mathbb{N}$, determine whether there exists $g, h \in F[x]$ with $\deg(g) = r$ and $\deg(h) = s$ such that $f = g \circ h$, if so, find h in a manner whose costs is polynomial $log n, \log s, \tau(f), \tau(h), r$.

- By *Zannier(2007,2008,2009),* we can assume that g is of low degree and h is sparse if f is sparse;
- Follow the same "tame" assumption;
- Use the same reduction and reversal tricks;
- Our faster sparse perfect power algorithm would lead to a polynomial-time algorithm for sparse polynomial decomposition;
- g can be found using interpolation.

Sparse Polynomial Decomposition Algorithm

Algorithm 9: SparsePolyDecomp

Input: sparse polynomial $f \in \mathsf{F}[x]$ of degree n,integer $s, r \in \mathbb{Z}$ s.t $n = rs$

Output: sparse polynomials $g, h \in \mathsf{F}[x]$ such that $f = g \circ h$

$\alpha = \mathrm{lc}(f),\ \beta = \alpha^{1/r},\ h_0 = f_0^{1/r};$

$\bar{f} = f/\alpha;$

$\tilde{f} = \mathrm{reversal}(\bar{f});$

$\tilde{h} =$ Algorithm 6_SparsePerfectPowerMod$(\tilde{f}, r, s);$

$\bar{h} = x^s \cdot \tilde{h}(1/x);$

$h = \beta \cdot \bar{h} + h_0;$

$O(\tau(f)\tau(h))$

// Next we recover g;

Fix a set $\mathcal{S} \subseteq F$ with $2n$ elements

Let $a_1 = h(0)$;

For i from 2 to r do

  Choose a random $a_i \in S$ repeatedly until $h(a_i) \notin \{h(a_1), \ldots, h(a_{i-1})\}$;

Interpolate $g \in \mathsf{F}[x]$ from points $\{(f(a_1), h(a_1)), \ldots, (f(a_r), h(a_r))\}$;

return g, h;

$O(r\tau(h)logs)$

$O(rlog^2r)$

$O(r\tau(f)logn)$

Check the correctness of the result

- In the perfect power case, we check $f = h^r$ by using the trick $f'h = rh'f$;
  this is fast even in the sparse setting;
- We don’t know if there exists such a certificate for general polynomial decomposition;
- Can do a randomized test, a deterministic one is unknown.

- Solve the Polynomial Perfect Power Problem in dense setting
- Solve the Polynomial Perfect Power Problem in sparse setting
- Solve Sparse Polynomial Decomposition inspired by the Polynomial Perfect Power Problem
- Generalize the Polynomial Perfect Power Problem to Differential Equations

UNIVERSITY OF WATERLOO

Perfect Power Problem to Differential Equation

- Solving $f'h = rh'f$ is in fact solving a differential equation over power series :

$$0 = (f' - rf\mathcal{D})h,$$

where $\mathcal{D}$ is the differential operator, $(f' - rf\mathcal{D})$ is a linear differential operator.

- Consider the general linear differential operator with polynomial coefficients :

$$\mathcal{L} = f_0 + f_1\mathcal{D} + f_2\mathcal{D}^2 + \cdots + f_\ell\mathcal{D}^\ell,$$

for $f_0, \cdots, f_\ell \in F[x]$ (change of notation), $f_\ell \neq 0$, ℓ is the order of $\mathcal{L}$.

- We will use $f_{i,j}$ to denote the jth coefficient of f_i.

Sparse Linear Differential Equation Problem

Given $\mathcal{L} = f_0 + f_1\mathcal{D} + f_2\mathcal{D}^2 + \cdots + f_\ell\mathcal{D}^\ell$, where $\tau(f_i) \leq t$ and $m \in \mathbb{N}$. The problem is to find an h such that

$$\mathcal{L}h \equiv 0 \ \ mod\ x^m.$$

- h is a "modulo approximation" to a power series solution to the differential equation;
- Sometimes we have a priori knowledge that h is sparse;
- Given $h_0, \cdots, h_{\ell-1}$, if $f_\ell(0) \neq 0$, then h exists and is unique;

  (Undetermined Coefficient Method);
- The goal is to design an algorithm with $(\ell + t + \tau(h))^{O(1)}$ operations in F.

UNIVERSITY OF WATERLOO

Similar "Newton-like" Method

$$
\begin{aligned}
0 &= f_0 h + f_1 h' + \cdots + f_\ell h^{(l)} \\
&= f_0(\check{h} + h_k x^k + \hat{h} x^{k+1}) + f_1(\check{h} + h_k x^k + \hat{h} x^{k+1})' + \cdots + f_\ell(\check{h} + h_k x^k + \hat{h} x^{k+1})^{(l)} \\
&= \left(f_0 \check{h} + f_1 \check{h}' + \cdots + f_\ell \check{h}^{(l)}\right) + \frac{k!}{(k-\ell)!} h_k f_\ell(0) x^{k-\ell} \mod x^{k-\ell+1} \\
&= R x^{k-\ell} \mod x^{k-l+1},
\end{aligned}
$$

$$
h_k = -R/\left(\frac{k!}{(k-\ell)!}\boxed{f_\ell(0)}\right)
$$

$$
\begin{aligned}
res &= f_0 \cdot (hlo + h_k \cdot x^k) + f_1 \cdot (hlo + h_k \cdot x^k)' + \cdots + f_\ell \cdot (hlo + h_k \cdot x^k)^{(\ell)} \\
&= res' + h_k f_0 x^k + k h_k f_1 x^{k-1} + \cdots + \frac{k!}{(k-\ell)!} h_k f_\ell x^{k-\ell}.
\end{aligned}
$$

General Differential Equation Algorithm

Algorithm 12: GeneralCase

Input: polynomial $f_0, f_1, \cdots, f_\ell \in \mathsf{F}[x]$ of degree $n_0, n_1, \cdots, n_\ell$ respectively s.t. $f_\ell(0) \neq 0$ and integer $m \in \mathbb{Z}$

Output: polynomial $h \in \mathsf{F}[x]$ such that $f_0 h + f_1 h' + \cdots + f_\ell h^{(\ell)} \equiv 0 \bmod x^m$

Pick any $h_0, h_1, \cdots, h_{\ell-1}$;

$h = h_0 + h_1 x + \cdots + h_{\ell-1} x^{\ell-1}$;

$res = f_0 h + f_1 h' + \cdots + f_\ell h^{(\ell)}$;

$k = 1$;

while $\mathrm{ldegree}(res) < m + 1$ **do**

- $k = \mathrm{ldegree}(res) + \ell$;
- $h_k = -\mathrm{coeff}(res, k - \ell) / (-\frac{k!}{(k-\ell)!} f_\ell(0))$;
- $h += h_k \cdot x^k$;
- $\boxed{res += \sum_{i=0}^{\ell} \frac{k!}{(k-i)!} h_k \cdot f_i x^{k-i};}$

end

return h;

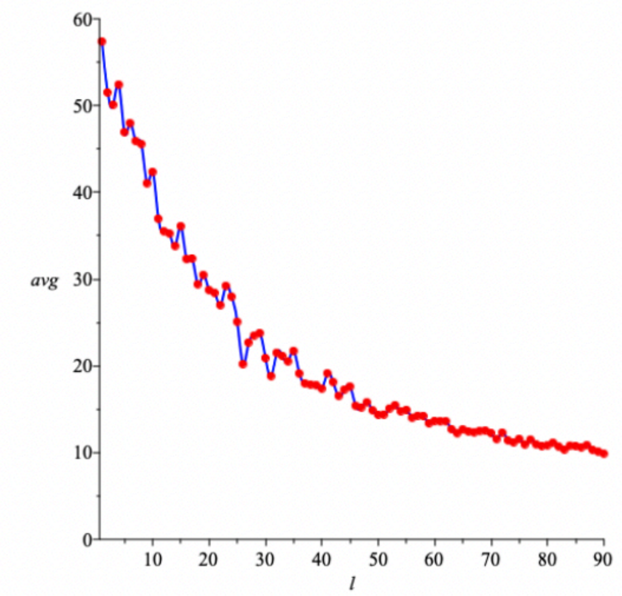
$O(t \ell \tau(h))$

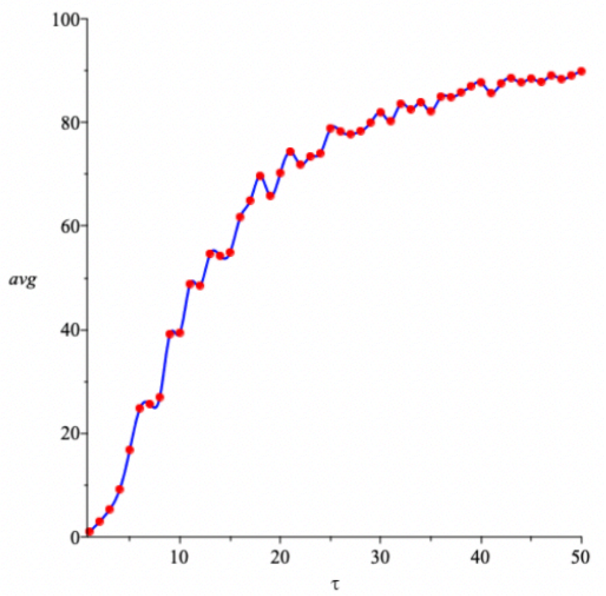
Implementation

- We implement our algorithms in Maple as proof of concept;
- The algorithm works for any finite field, the implementation is over $\mathbb{Z}$ computationally;

For the general differential equation :

- The solution sparsity decreases as the order ℓ increases;
- The solution sparsity increases as the coefficient sparsity bound t increases.

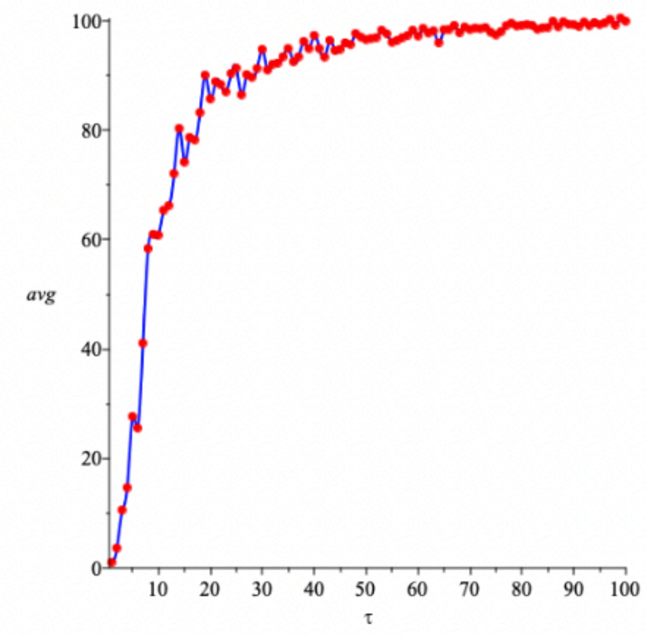


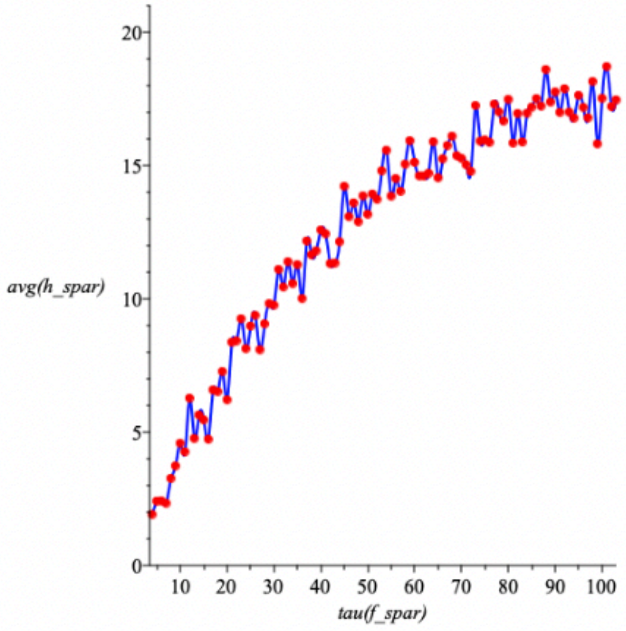


UNIVERSITY OF WATERLOO

Implementation

- For the perfect power problem as a special case, i.e., $\ell = 1$.
- The solution sparsity increases as the input sparsity $\tau(f)$ increases.
- Perfect power structure seems lead to a more linear relationship.





UNIVERSITY OF WATERLOO

Summary

- Solve the Polynomial Perfect Power Problem in dense setting
- Solve the Polynomial Perfect Power Problem in sparse setting
- Solve the Sparse Polynomial Decomposition
- Generalize the Polynomial Perfect Power Problem to Differential Equations

Open Problems

- An optimal algorithm for sparse perfect power in $O(\tau(f) + \tau(h))$?
- Sparse Polynomial decomposition in the "wild" case?
- A deterministic algorithm to check the sparse polynomial decomposition?
- A fast algorithm to solve $\mathcal{L}h \equiv 0 \ mod\ x^m$ when $f_\ell(0) = 0$?
- An optimal algorithm for $\mathcal{L}h \equiv 0 \ mod\ x^m$ in $O(t + \ell + \tau(h))$?
- Any other families of differential equations we can expect sparse Taylor series?

Thank you:)

- Supervisors

  Mark Giesbrecht, Arne Storjohann

- Readers

  George Labahn, Robert Corless

- My supportive family and friends

UNIVERSITY OF WATERLOO