Newton iteration

Part 2

Alin Bostan

MPRI C-2-22

October 18, 2022

Newton iteration – main theorem

1. ("Implicit function theorem") Let $\varphi \in \mathbb{K}[[x,y]]$ s.t. $\varphi(0,0)=0$ and $\varphi_y(0,0)\neq 0$. There exists a unique solution $S \in x\mathbb{K}[[x]]$ to $\varphi(x,S)=0$.
2. ("Newton iteration") Define $Y_\kappa = S \bmod x^{2^\kappa}$. Then,

$$Y_0 = 0 \quad \text{and} \quad Y_{\kappa+1} = Y_\kappa - \frac{\varphi(x,Y_\kappa)}{\varphi_y(x,Y_\kappa)} \mod x^{2^{\kappa+1}} \quad \text{for } \kappa \geq 0.$$

Proof of (1). Let $\varphi(x,y) = \sum_{j\geq 0} f_j y^j$ with $f_j = \sum_{i \geq 0} f_{j,i} x^i$.
Then $\varphi(x,S)=0$, with $S = \sum_{\ell \geq 1} s_\ell x^\ell$, is equivalent to

$$f_{0,0}=0,\ f_{1,0}s_1 + f_{0,1} = 0,\ f_{1,0}s_\kappa + \mathrm{Pol}_\kappa(s_1,\ldots,s_{\kappa-1}, f_{j,i}, i+j \leq \kappa) = 0$$

Since $f_{0,0} = \varphi(0,0) = 0$ and $f_{1,0} = \varphi_y(0,0) \neq 0$, system has a unique solution.

Proof of (2). $Y_0 = S \bmod x$, hence $Y_0 = S(0) = 0$. By Taylor's formula,

$$0 = \varphi(x,S) = \varphi(x, Y_\kappa + (S-Y_\kappa)) = \varphi(x,Y_\kappa) + \varphi_y(x,Y_\kappa)\cdot(S-Y_\kappa) + O((S-Y_\kappa)^2).$$

Now, $\varphi_y(x,Y_\kappa) \bmod x = \varphi_y(0,0) \neq 0$, hence $\varphi_y(x,Y_\kappa)$ invertible. Thus,

$$0 = \frac{\varphi(x,Y_\kappa)}{\varphi_y(x,Y_\kappa)} + S - Y_\kappa + O(x^{2^{\kappa+1}}) \implies Y_\kappa - \frac{\varphi(x,Y_\kappa)}{\varphi_y(x,Y_\kappa)} \bmod x^{2^{\kappa+1}} = S \bmod x^{2^{\kappa+1}} = Y_{\kappa+1}.$$

Examples: reciprocal and exponential, again

▷ Using $\varphi(x,y) = (F(0)^{-1} + y)^{-1} - F(x)$ to invert $F \in \mathbb{K}[[x]]$, will find

$$S = F(x)^{-1} - F(0)^{-1}$$

after using the Newton operator $\mathcal{N} : G \mapsto 2(G + \frac{1}{F(0)}) - F(G + \frac{1}{F(0)})^2 - \frac{1}{F(0)}$.

$\Longrightarrow$ this is equivalent to $\mathcal{N} : G \mapsto 2G - FG^2$ with initial value $G = F(0)^{-1}$

▷ Using $\varphi(x,y) = F(x) - \log(1+y)$, to compute exp of $F \in x\mathbb{K}[[x]]$, will find

$$S = \exp(F) - 1$$

after using the Newton operator $\mathcal{N} : G \mapsto G + (1+G)(F - \log(1+G))$.

$\Longrightarrow$ this is equivalent to $\mathcal{N} : G \mapsto G + G(F - \log G)$ with initial value $G = 1$

Newton iteration: a variant

Idea: Interlace two Newton schemes, one for solving φ, one for φ_y^{-1}

Input Some $N \in \mathbb{N}_{>0}$, the truncation $P = \mathrm{rem}(\varphi, \{x^N, y^N\})$ of a power series $\varphi \in \mathbb{K}[[x, y]]$ such that $\varphi(0,0) = 0$ and $\varphi_y(0,0) \neq 0$.

Output Polynomials $F = \mathrm{rem}(S, x^N)$ and $G = \mathrm{rem}(T, x^{\lceil N/2 \rceil})$, where S is the unique series solution in $x\mathbb{K}[[x]]$ to $\varphi(x, S) = 0$ and $T = \varphi_y(x, S)^{-1}$.

If $N = 1$, return $F = 0$, $G = \varphi_y(0,0)^{-1}$. Otherwise:

(a) Recursively call the algorithm with $\lceil N/2 \rceil$, in order to compute truncations $F_1 = \mathrm{rem}(S, x^{\lceil N/2 \rceil})$ and $G_1 = \mathrm{rem}(T, x^{\lceil \lceil N/2 \rceil / 2 \rceil})$.

(b) Compute $U_1 = \mathrm{rem}(P(x, F_1), x^N)$ and $V_1 = \mathrm{rem}(P_y(x, F_1), x^{\lceil N/2 \rceil})$.

(c) Compute $G := G_1 + \mathrm{rem}((1 - G_1 V_1) G_1, x^{\lceil N/2 \rceil})$.

(d) Compute $F := F_1 - \mathrm{rem}(G U_1, x^N)$.

(e) Return F and G.

Application: extension of recurrences

[Shoup, 1991]

Problem: Given $r, N \in \mathbb{N}$, a linear recurrence with constant coefficients of order r for $(u_n)_n$ and the first r terms $u_0, \dots, u_{r-1}$, compute $u_r, \dots, u_N$

Naive algorithm: unroll the recurrence $O(rN) \subseteq O(N^2)$

Idea: $\sum_{i \geq 0} u_i x^i$ is rational $A(x)/B(x)$, with B given by the input recurrence, and $\deg(A) < \deg(B)$

Example (Fibonacci): $F_{i+2} = F_{i+1} + F_i \quad \Longleftrightarrow \quad \sum_i F_i x^i = \frac{F_0 + (F_1 - F_0)x}{1 - x - x^2}$

Algorithm:

- Compute A from B and $u_0, \dots, u_{r-1}$ $O(\mathsf{M}(r))$
- Expand A/B modulo x^{N+1} $O(\mathsf{M}(N))$

Application: conversion coefficients $\leftrightarrow$ power sums

[Schönhage, 1982]

Any polynomial $F = x^n + a_1 x^{n-1} + \cdots + a_n$ in $\mathbb{K}[x]$ can be represented by its first n power sums $S_i = \sum_{F(\alpha)=0} \alpha^i$

Conversions coefficients $\leftrightarrow$ power sums can be performed

- either in $O(n^2)$ using Newton identities (naive way):

$$i a_i + S_1 a_{i-1} + \cdots + S_i = 0, \quad 1 \leq i \leq n$$

- or in $O(\mathsf{M}(n))$ using generating series

$$\frac{\mathsf{rev}(F)'}{\mathsf{rev}(F)} = -\sum_{i \geq 0} S_{i+1} x^i \quad \Longleftrightarrow \quad \mathsf{rev}(F) = \exp\left(-\sum_{i \geq 1} \frac{S_i}{i} x^i\right)$$

Application: special bivariate resultants

[B-Flajolet-S-Schost, 2006]

Composed products and sums: manipulation of algebraic numbers

$$F \otimes G = \prod_{F(\alpha)=0, G(\beta)=0} (x - \alpha\beta), \quad F \oplus G = \prod_{F(\alpha)=0, G(\beta)=0} (x - (\alpha + \beta))$$

Output size: $N = \deg(F)\deg(G)$

Linear algebra: χ_{xy}, χ_{x+y} in $\mathbb{K}[x,y]/(F(x), G(y))$ $O(\mathsf{MM}(N))$

Resultants: $\mathrm{Res}_y\left(F(y), y^{\deg(G)}G(x/y)\right)$, $\mathrm{Res}_y\left(F(y), G(x-y)\right)$ $O(N^{1.5})$

Better: $\otimes$ and $\oplus$ are easy in Newton representation $O(\mathsf{M}(N))$

$$\sum \alpha^s \sum \beta^s = \sum (\alpha\beta)^s \qquad \text{and}$$

$$\sum \frac{\sum(\alpha+\beta)^s}{s!} x^s = \left(\sum \frac{\sum \alpha^s}{s!} x^s\right) \left(\sum \frac{\sum \beta^s}{s!} x^s\right)$$

Corollary: Fast polynomial shift $P(x+a) = P(x) \oplus (x+a)$ $O(\mathsf{M}(\deg(P)))$

Newton iteration on power series: operators and systems

In order to solve an equation $\varphi(Y) = 0$, with $\varphi : (\mathbb{K}[[x]])^r \to (\mathbb{K}[[x]])^r$,

1. Linearize: $\varphi(Y_\kappa - U) = \varphi(Y_\kappa) - D\varphi|_{Y_\kappa} \cdot U + O(U^2)$,
   where $D\varphi|_Y$ is the differential of φ at Y.
2. Iterate: $Y_{\kappa+1} = Y_\kappa - U_{\kappa+1}$, where $U_{\kappa+1}$ is found by
3. Solve linear equation: $D\varphi|_{Y_k} \cdot U = \varphi(Y_\kappa)$ with $\operatorname{val} U > 0$.

Then, the sequence Y_κ converges quadratically to the solution Y.

Application: inversion of power series matrices

[Schulz, 1933]

To compute the inverse Z of a matrix of power series $Y \in \mathcal{M}_r(\mathbb{K}[[x]])$:

- Choose the map $\varphi : Z \mapsto I - YZ$ with differential $D\varphi|_Y : U \mapsto -YU$
- Equation for U: $-YU = I - YZ_\kappa \mod x^{2^{\kappa+1}}$
- Solution: $U = -Y^{-1}(I - YZ_\kappa) = -Z_\kappa(I - YZ_\kappa) \bmod x^{2^{\kappa+1}}$

This yields the following Newton-type iteration for Y^{-1}

$$Z_{\kappa+1} = Z_\kappa + Z_\kappa(I_r - YZ_\kappa) \mod x^{2^{\kappa+1}}$$

Complexity:

$\mathsf{C}_{\mathrm{inv}}(N) = \mathsf{C}_{\mathrm{inv}}(N/2) + O(\mathsf{MM}(r,N)) \implies \mathsf{C}_{\mathrm{inv}}(N) = O(\mathsf{MM}(r,N))$

Application: non-linear systems

In order to solve a system $Y = H(Y) = \varphi(Y) + Y$, with $H : (\mathbb{K}[[x]])^r \to (\mathbb{K}[[x]])^r$, such that $I_r - \partial H/\partial Y$ is invertible at 0.

1. Linearize: $\varphi(Y_\kappa - U) - \varphi(Y_\kappa) = U - \partial H/\partial Y(Y_\kappa) \cdot U + O(U^2)$.
2. Iterate $Y_{\kappa+1} = Y_\kappa - U_{\kappa+1}$, where $U_{\kappa+1}$ is found by
3. Solve linear equation: $(I_r - \partial H/\partial Y(Y_\kappa)) \cdot U = H(Y_\kappa) - Y_\kappa$ with $\operatorname{val} U > 0$.

This yields the following Newton-type iteration:

$$\begin{cases} Z_{\kappa+1} & = Z_\kappa + Z_\kappa(I_r - (I_r - \partial H/\partial Y(Y_\kappa))Z_\kappa) \mod x^{2^{\kappa+1}} \\ Y_{\kappa+1} & = Y_\kappa - Z_{\kappa+1}(H(Y_\kappa) - Y_\kappa) \mod x^{2^{\kappa+1}} \end{cases}$$

computing simultaneously a matrix and a vector.

Application: quasi-exponential of power series matrices

[B-Chyzak-Ollivier-Salvy-Schost-Sedoglavic 2007]

To compute the solution $Y \in \mathcal{M}_r(\mathbb{K}[[x]])$ of the system $Y' = AY$

- choose the map $\varphi : Y \mapsto Y' - AY$, with differential φ.
- the equation for U is $\quad U' - AU = Y'_\kappa - AY_\kappa \mod x^{2^{\kappa+1}}$
- the method of variation of constants yields the solution
  $U = Y_\kappa V_\kappa \mod x^{2^{\kappa+1}}$, $\; Y'_\kappa - AY_\kappa = Y_\kappa V'_\kappa \mod x^{2^{\kappa+1}}$

This yields the following Newton-type iteration for Y:

$$Y_{\kappa+1} = Y_\kappa - Y_\kappa \int Y_\kappa^{-1} \left(Y'_\kappa - AY_\kappa\right) \mod x^{2^{\kappa+1}}$$

Complexity:

$\mathsf{C}_{\text{solve}}(N) = \mathsf{C}_{\text{solve}}(N/2) + O(\mathsf{MM}(r, N)) \quad \Longrightarrow \quad \mathsf{C}_{\text{solve}}(N) = O(\mathsf{MM}(r, N))$

Bonus

Composition of series (just sketched)

▷ Naive approach (by Horner scheme) $O(N\,\mathsf{M}(N))$

▷ [Paterson and Stockmeyer, 1973] $O(\sqrt{N}(\mathsf{M}(N)+\mathsf{MM}(\sqrt{N})))$

By Shanks' 1969 baby-steps giant-steps technique: split polynomials in chunks of length $\sqrt{N}$, matrices in blocks of size $\sqrt{N}\times\sqrt{N}$.

▷ [Brent and Kung, 1978] $O(\sqrt{N\log N}\,\mathsf{M}(N))$

Similar splitting + Taylor formula.

Faster Modular Composition

Faster Modular Composition

VINCENT NEIGER, Sorbonne Université, France
BRUNO SALVY, Inria, France
ÉRIC SCHOST, University of Waterloo, Canada
GILLES VILLARD, CNRS, France

A new Las Vegas algorithm is presented for the composition of two polynomials modulo a third one, over an arbitrary field. When the degrees of these polynomials are bounded by n, the algorithm uses $O(n^{1.43})$ field operations, breaking through the 3/2 barrier in the exponent for the first time. The previous fastest algebraic algorithms, due to Brent and Kung in 1978, require $O(n^{1.63})$ field operations in general, and $n^{3/2+o(1)}$ field operations in the particular case of power series over a field of large enough characteristic. If using cubic-time matrix multiplication, the new algorithm runs in $n^{5/3+o(1)}$ operations, while previous ones run in $O(n^2)$ operations.

Our approach relies on the computation of a matrix of algebraic relations that is typically of small size. Randomization is used to reduce arbitrary input to this favorable situation.

CCS Concepts: • **Computing methodologies → Algebraic algorithms**; • **Theory of computation → Algebraic complexity theory**.

Additional Key Words and Phrases: composition of polynomials, complexity