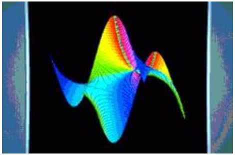
Newton Iteration: From Numerics to Combinatorics, and Back

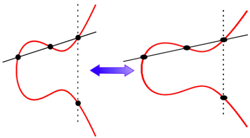
Bruno Salvy
Bruno.Salvy@inria.fr

Algorithms Project, Inria

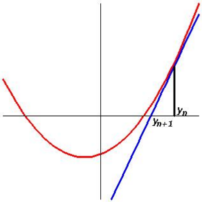


Analco, January 16, 2010

I Introduction

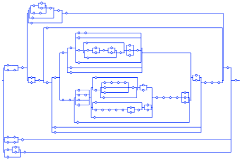


ALgorithms



ANalysis

COmbinatorics



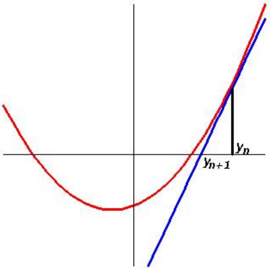
Newton Iteration

To solve $\phi(y) = 0$, iterate

$$y_{n+1} = y_n - u_{n+1}, \quad \phi'(y_n)u_{n+1} = \phi(y_n).$$

Good case: quadratic convergence if

- the initial point is close enough;
- the root is simple.



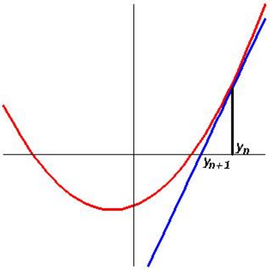
Newton Iteration

To solve $\phi(y) = 0$, iterate

$$y_{n+1} = y_n - u_{n+1}, \quad \phi'(y_n)u_{n+1} = \phi(y_n).$$

Good case: quadratic convergence if

- the initial point is close enough;
- the root is simple.



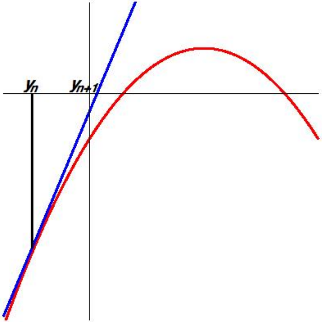
Proof: simple root at ζ

$$\left.\begin{aligned}\phi(z) &= \phi'(\zeta)(z-\zeta) + O((z-\zeta)^2)\\ \phi'(z) &= \phi'(\zeta) + O(z-\zeta)\end{aligned}\right\} \Rightarrow z - \zeta = \frac{\phi(z)}{\phi'(z)} + O((z-\zeta)^2).$$

Examples of Quadratic Convergence

$$\phi(y) = y - 1 - y^2/8$$

$$y_{n+1} = y_n - \frac{y_n - 1 - y_n^2/8}{1 - y_n/4}$$

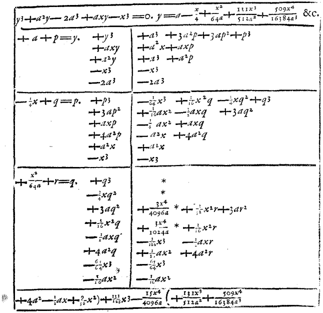


$$\begin{aligned}
y_0 &= 0.\\
y_1 &= 1.000000000000000000000000000\\
y_2 &= 1.166666666666666666666666666\\
y_3 &= 1.171568627450980392156862745\\
y_4 &= 1.171572875250620178747408840\\
y_5 &= 1.171572875253809902396620740
\end{aligned}$$

Examples of Quadratic Convergence

$$\phi(y) = y - 1 - zy^2$$

$$y_{n+1} = y_n - \frac{y_n - 1 - zy_n^2}{1 - zy_n^2}$$


$y^3 + a^2y - 2a^3 + axy - x^3 = 0$. $y = a - \frac{x}{4} + \frac{x^2}{64a} + \frac{131x^3}{512a^2} + \frac{509x^4}{16384a^3}$ &c.

$$y_0 = 0$$

$$y_1 = 1$$

$$y_2 = 1 + z + 2z^2 + 4z^3 + 8z^4 + 16z^5 + 32z^6 + 64z^7 + \dots$$

$$y_3 = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 428z^7 + \dots$$

[Newton 1671]

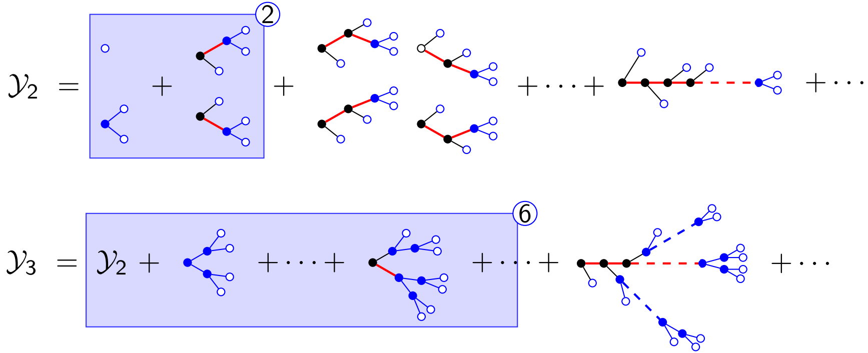
Examples of Quadratic Convergence

$$\phi(\mathcal{Y}) = \mathcal{Y} \setminus (\mathcal{E} \cup \mathcal{Z} \times \mathcal{Y}^2)$$

Examples of Quadratic Convergence

$$\phi(\mathcal{Y}) = \mathcal{Y} \setminus (\mathcal{E} \cup \mathcal{Z} \times \mathcal{Y}^2)$$

$\mathcal{Y}_0 = \varnothing \qquad \mathcal{Y}_1 = \circ$



[Décoste, Labelle, Leroux 1982]

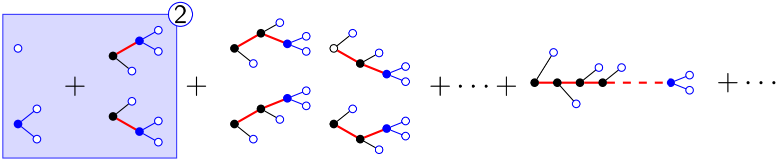
Examples of Quadratic Convergence

$$\phi(\mathcal{Y}) = \mathcal{Y} \setminus (\mathcal{E} \cup \mathcal{Z} \times \mathcal{Y}^2)$$

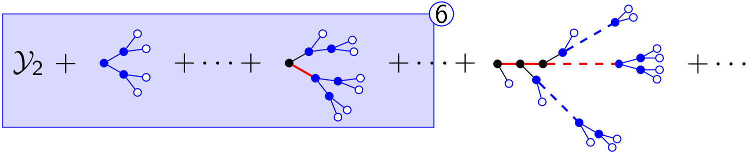
$$\mathcal{Y}_{n+1} = \mathcal{Y}_n \cup \textsc{Seq}(\mathcal{Z} \times \mathcal{Y}_n \times \square \cup \mathcal{Z} \times \square \times \mathcal{Y}_n) \times (\mathcal{E} \cup \mathcal{Z} \times \mathcal{Y}_n^2 \setminus \mathcal{Y}_n).$$

$\mathcal{Y}_0 = \varnothing \qquad \mathcal{Y}_1 = \circ$

$\mathcal{Y}_2 =$



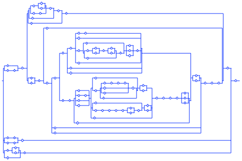
$\mathcal{Y}_3 =$



[Décoste, Labelle, Leroux 1982]

Motivation: Random Generation

$$\mathcal{Y} = \mathcal{S} \cup \mathcal{P},$$
$$\mathcal{S} = \mathrm{Seq}(\mathcal{Z} \cup \mathcal{P}, \mathrm{card} > 1),$$
$$\mathcal{P} = \mathrm{Set}(\mathcal{Z} \cup \mathcal{S}, \mathrm{card} > 0).$$



Definition (Generating function)

$Y(z) = y_0 + y_1 z + \cdots + y_n z^n + \cdots$ (y_n: nb of objects of size n).

Algorithms for uniform random generation in size N use either

- $(y_0, \ldots, y_N)$ (recursive method);
- $Y(x)$ for some $x > 0$ (Boltzmann sampler).

[Nijenhuis & Wilf 1978; Flajolet, Zimmermann, Van Cutsem 1994; Duchon, Flajolet, Louchard, Schaeffer 2004]

Combinatorial Newton Iteration

Arbitrary Combinatorial Specification

$\Downarrow$

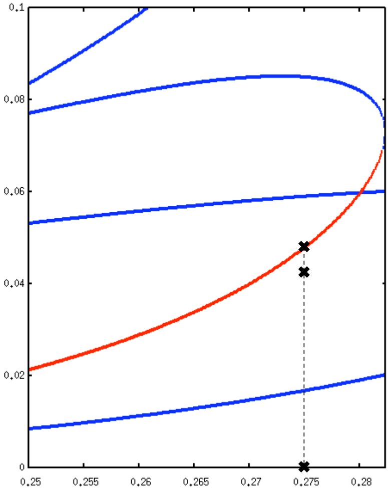
Combinatorial Newton iteration for $\mathcal{Y}$

$\Downarrow$

Newton iteration for the gf $Y(z)$
$((y_0, \ldots, y_N)$ fast)

$\Downarrow$

Numerical Newton iteration
starting from 0 converges to the value of $Y(x)$.



Combinatorial Newton Iteration

Arbitrary Combinatorial Specification

$\Downarrow$

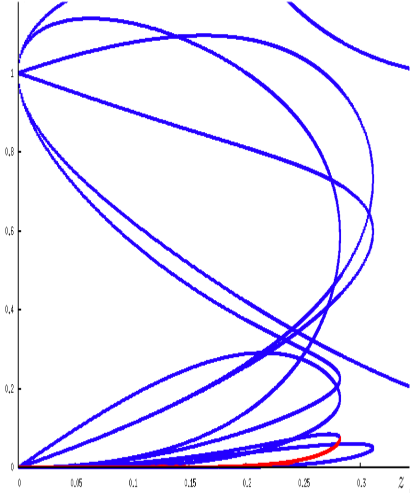
Combinatorial Newton iteration for $\mathcal{Y}$

$\Downarrow$

Newton iteration for the gf $Y(z)$
($(y_0, \dots, y_N)$ fast)

$\Downarrow$

Numerical Newton iteration starting from 0 converges to the value of $Y(x)$.



[Pivoteau, Salvy, Soria 2008]

II Newton Iteration: Algorithms

Basic Idea: Newton Iteration has Good Complexity

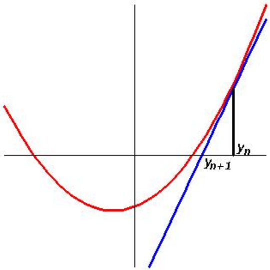
To solve $\phi(y) = 0$, iterate

$$y_{n+1} = y_n - u_{n+1}, \quad \phi'(y_n) u_{n+1} = \phi(y_n).$$

Quadratic convergence

$\Updownarrow$

Divide-and-Conquer



Basic Idea: Newton Iteration has Good Complexity

To solve $\phi(y) = 0$, iterate

$$y_{n+1} = y_n - u_{n+1}, \quad \phi'(y_n)u_{n+1} = \phi(y_n).$$

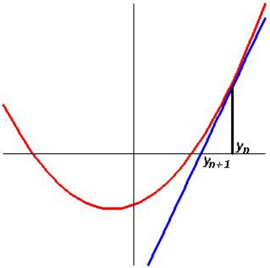
Quadratic convergence

$\Updownarrow$

Divide-and-Conquer

To solve at precision N

1. Solve at precision $N/2$;
2. Compute ϕ and ϕ' there;
3. Solve for u_{n+1}.



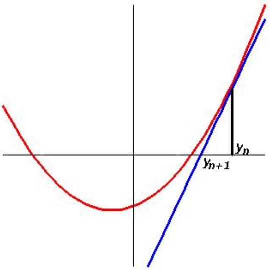
$\mathrm{Cost}(y_n) = \mathrm{constant} \times \mathrm{Cost}(\text{last step}).$

Basic Idea: Newton Iteration has Good Complexity

To solve $\phi(y) = 0$, iterate

$$y_{n+1} = y_n - u_{n+1}, \quad \phi'(y_n)u_{n+1} = \phi(y_n).$$

Quadratic convergence
$\Updownarrow$
Divide-and-Conquer



To solve at precision N

1. Solve at precision $N/2$;
2. Compute ϕ and ϕ' there;
3. Solve for u_{n+1}.

$$\text{Cost}(y_n) = \text{constant} \times \text{Cost(last step)}.$$

Useful in conjunction with fast multiplication (quasi-linear):

- power series at order N: $O(N \log N)$ ops on the coefficients;
- N-bit integers: $O(N \log N \log\log N)$ bit ops.

Application to Inverses

$$\phi(y) = a - 1/y \Rightarrow 1/\phi'(y) = -y^2 \Rightarrow \boxed{y_{n+1} = y_n - y_n(ay_n - 1).}$$

Cost: a small number of multiplications

Works for:

1. Numerical inversion;
2. Reciprocal of power series;
3. Inversion of matrices.

Application to Inverses

$$\phi(y) = a - 1/y \Rightarrow 1/\phi'(y) = -y^2 \Rightarrow \boxed{y_{n+1} = y_n - y_n(ay_n - 1).}$$

Cost: a small number of multiplications

Works for:

1. Numerical inversion;
2. Reciprocal of power series;
3. Inversion of matrices. $\Phi : Y \mapsto A - Y^{-1}$

$$\Phi(Y+U) = \Phi(Y) + \underbrace{Y^{-1}UY^{-1}}_{D\Phi|_Y U} + O(U^2),$$

$$D\Phi|_Y\, U = \Phi(Y) \Rightarrow U = Y(A - Y^{-1})Y.$$

Application to Inverses

$$\phi(y) = a - 1/y \Rightarrow 1/\phi'(y) = -y^2 \Rightarrow \boxed{y_{n+1} = y_n - y_n(ay_n - 1).}$$

Cost: a small number of multiplications

Works for:

1. Numerical inversion;
2. Reciprocal of power series;
3. Inversion of matrices.

Consequences: fast computation of

1. Logarithm of power series: $\log f = \int (f'/f)$;
2. exponential of power series: $\phi(y) = a - \log y$.

[Schulz 1933; Cook 1966; Sieveking 1972; Kung 1974; Brent 1975]

Application: Power Sums

$$F = t^N + a_{N-1}t^{N-1} + \cdots + a_0 \leftrightarrow S_i = \sum_{F(\alpha)=0} \alpha^i, \quad i = 0, \ldots, N.$$

Application: Power Sums

$$F = t^N + a_{N-1}t^{N-1} + \cdots + a_0 \leftrightarrow S_i = \sum_{F(\alpha)=0} \alpha^i, \quad i = 0, \ldots, N.$$

Fast conversion using the generating series:

$$\frac{\operatorname{rev}(F)'}{\operatorname{rev}(F)} = -\sum_{i \geq 0} S_{i+1} t^i \leftrightarrow \operatorname{rev}(F) = \exp\left(-\sum \frac{S_i}{i} t^i\right).$$

Application: Power Sums

$$F = t^N + a_{N-1}t^{N-1} + \cdots + a_0 \leftrightarrow S_i = \sum_{F(\alpha)=0} \alpha^i, \quad i = 0, \ldots, N.$$

Fast conversion using the generating series:

$$\frac{\operatorname{rev}(F)'}{\operatorname{rev}(F)} = -\sum_{i\geq 0} S_{i+1} t^i \leftrightarrow \operatorname{rev}(F) = \exp\left(-\sum \frac{S_i}{i} t^i\right).$$

Application: composed product and sums

$$(F, G) \mapsto \prod_{F(\alpha)=0, G(\beta)=0} (t - \alpha\beta) \quad \text{or} \quad \prod_{F(\alpha)=0, G(\beta)=0} (t - (\alpha + \beta)).$$

Application: Power Sums

$$F = t^N + a_{N-1}t^{N-1} + \cdots + a_0 \leftrightarrow S_i = \sum_{F(\alpha)=0} \alpha^i, \quad i = 0, \ldots, N.$$

Fast conversion using the generating series:

$$\frac{\operatorname{rev}(F)'}{\operatorname{rev}(F)} = -\sum_{i \geq 0} S_{i+1} t^i \leftrightarrow \operatorname{rev}(F) = \exp\left(-\sum \frac{S_i}{i} t^i\right).$$

Application: composed product and sums

$$(F, G) \mapsto \prod_{F(\alpha)=0, G(\beta)=0} (t - \alpha\beta) \quad \text{or} \quad \prod_{F(\alpha)=0, G(\beta)=0} (t - (\alpha + \beta)).$$

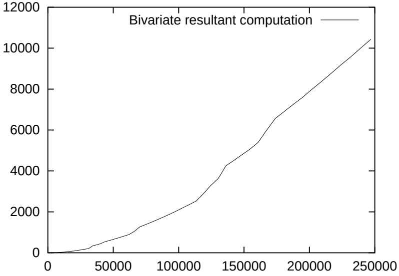
Easy in Newton representation: $\sum \alpha^s \sum \beta^s = \sum (\alpha\beta)^s$ and

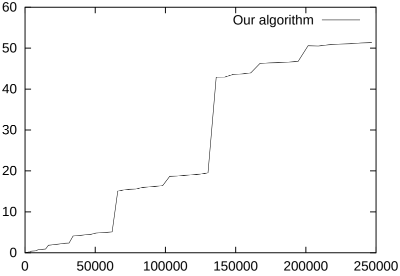
$$\sum \frac{\sum (\alpha + \beta)^s}{s!} t^s = \left(\sum \frac{\sum \alpha^s}{s!} t^s\right) \left(\sum \frac{\sum \beta^s}{s!} t^s\right).$$

[Schönhage 1982; Bostan, Flajolet, Salvy, Schost 2006]

Timings

Applications (crypto): over finite fields, degree > 200000 expected.





Timings in seconds vs. output degree N, over $\mathbb{F}_p$, 26 bits prime p

Linear Differential Equations of Arbitrary Order

Given a linear differential equation with power series coefficients,

$$a_r(t)y^{(r)}(t) + \cdots + a_0(t)y(t) = 0,$$

compute the first N terms of a basis of power series solutions.

Linear Differential Equations of Arbitrary Order

Given a linear differential equation with power series coefficients,

$$a_r(t)y^{(r)}(t) + \cdots + a_0(t)y(t) = 0,$$

compute the first N terms of a basis of power series solutions.

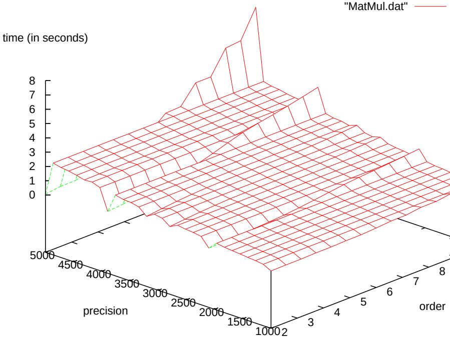
Algorithm

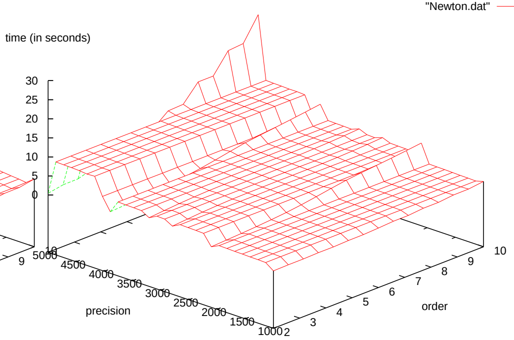
1. Convert into a system $\Phi : Y \mapsto Y' - A(t)Y$ ($D\Phi = \Phi$);
2. $D\Phi|_Y(U) = \Phi(Y)$ rewrites $U' - AU = Y' - AY$;
3. Variation of constants: $U = Y \int Y^{-1}(Y' - AY)$;
4. Y^{-1} by Newton iteration too.

Special case: good exponential.

[Bostan, Chyzak, Ollivier, Salvy, Schost, Sedoglavic 2007]

Timings



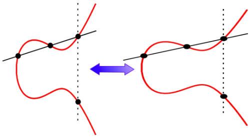


Polynomial matrix multiplication vs. solving $Y' = AY$.

Non-Linear Differential Equations

Example from cryptography:

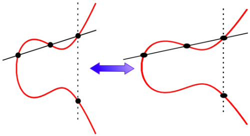
$$\phi : y \mapsto (x^3 + Ax + B)y'^2 - (y^3 + \tilde{A}y + \tilde{B}).$$



Non-Linear Differential Equations

Example from cryptography:

$$\phi : y \mapsto (x^3 + Ax + B)y'^2 - (y^3 + \tilde{A}y + \tilde{B}).$$



Differential:

$$D\phi|_y : u \mapsto 2(x^3 + Ax + B)y'u' - (3y^2 + \tilde{A})u.$$

Solve the linear differential equation

$$D\phi|_y \, u = \phi(y)$$

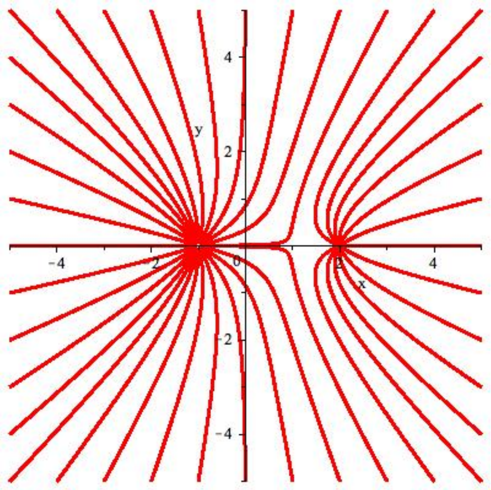
at each iteration.

Again, quasi-linear complexity.

[Bostan, Morain, Salvy, Schost 2008]

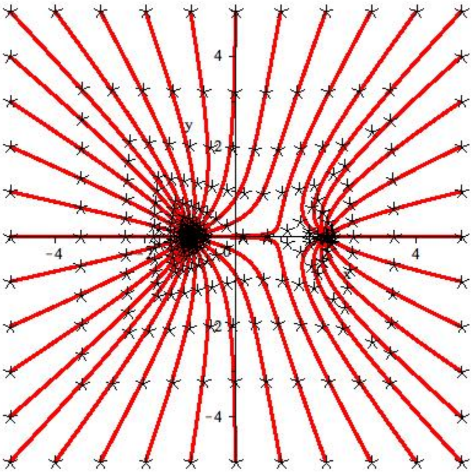
III Newton Iteration: Numerical Convergence

Multiplicities and Scales



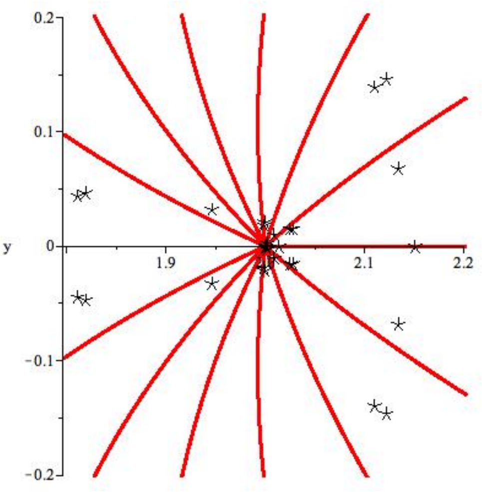
Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



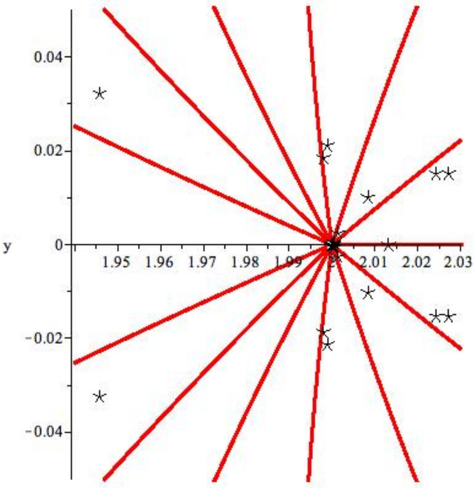
Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



Newton field for $(z+1)^2(z-2)$

Approximate Zeroes

Definition (Approximate Zero)

A point such that the sequence $z_0 = z$, $z_{k+1} := z_k - f'(z_k)^{-1} f(z_k)$ is well defined and satisfies

$$\|z_k - \zeta\| \leq \frac{\|z - \zeta\|}{2^{2^k - 1}}, \qquad k \geq 0.$$

Theorem (Smale's α-theorem)

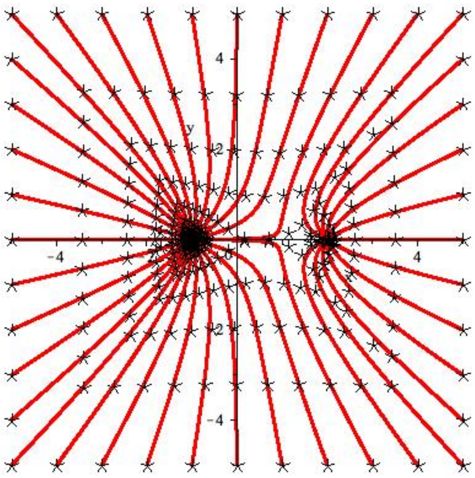
If

$$|f'(z_0)^{-1} f(z_0)| \sup_k \left| f'(z_0)^{-1} \frac{f^{(k)}(z_0)}{k!} \right|^{1/(k-1)} < 0.130\ldots,$$

then z_0 is an approximate zero.

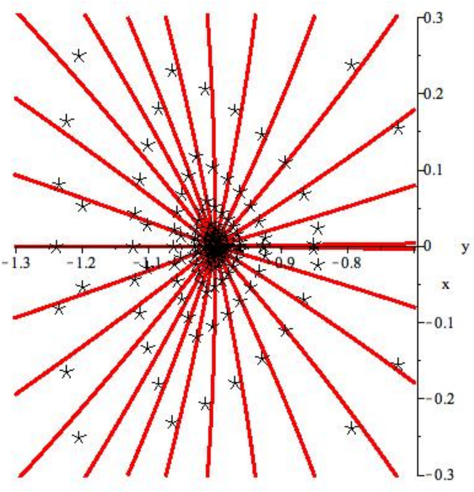
Also, a version for analytic maps in higher dimension.

Multiplicities and Scales



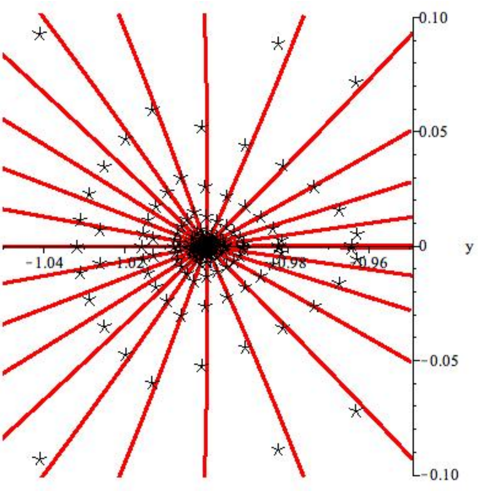
Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



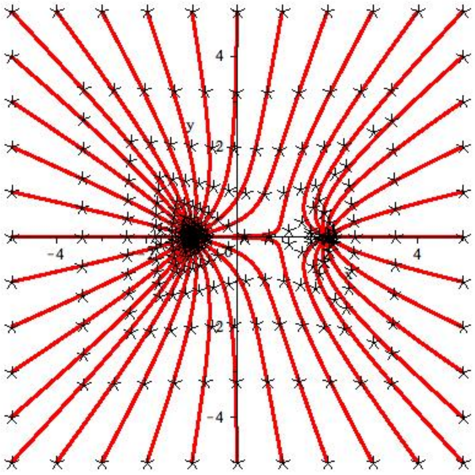
Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



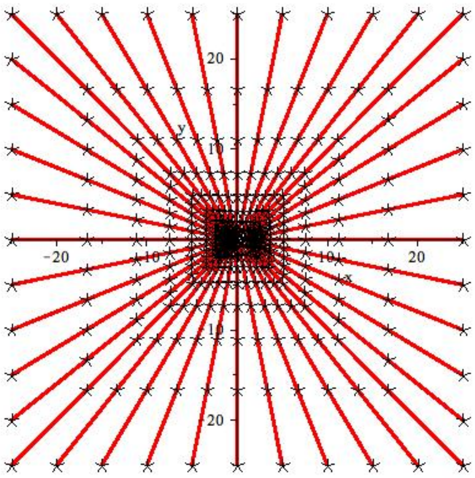
Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



Newton field for $(z+1)^2(z-2)$

Multiplicities and Scales



Newton field for $(z+1)^2(z-2)$

Cures for Multiplicity m

1. Deflation: compute the root of the $(m-1)$th derivative;
2. Schröder iteration:

$$\left.\begin{aligned}\phi(z) &= \phi'(\zeta)(z-\zeta) + O((z-\zeta)^2)\\ \phi'(z) &= \phi'(\zeta) + O((z-\zeta))\end{aligned}\right\}$$

$$\Rightarrow z-\zeta = \frac{\phi(z)}{\phi'(z)} + O((z-\zeta)^2).$$

3. Mixture: Schröder iteration on a derivative.

Cures for Multiplicity m

1. Deflation: compute the root of the $(m-1)$th derivative;
2. Schröder iteration:

$$\left.\begin{aligned} \phi(z) &= \phi^{(m)}(\zeta)(z-\zeta)^m + O((z-\zeta)^{m+1}) \\ \phi'(z) &= m\phi^{(m)}(\zeta)(z-\zeta)^{m-1} + O((z-\zeta)^m) \end{aligned}\right\}$$

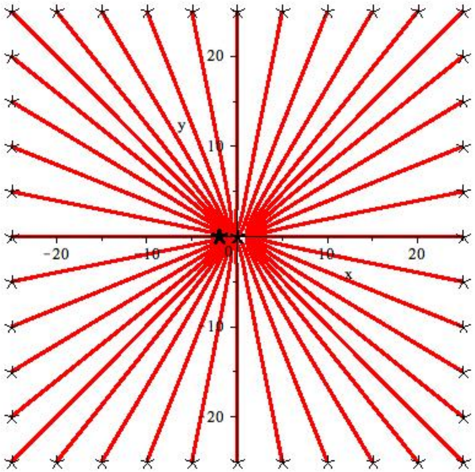
$$\Rightarrow z-\zeta = m\frac{\phi(z)}{\phi'(z)} + O((z-\zeta)^2).$$

3. Mixture: Schröder iteration on a derivative.

Can be used to speed-up the location of multiple roots or clusters of roots $\rightarrow$ α-theorems for clusters.
Also extends to systems (with a reasonable complexity).

[Giusti, Lecerf, Salvy, Yakoubsohn 2005 & 2007]

Multiplicities and Scales



IV Combinatorial Newton Iteration

Symbolic Method

Language and Gen. Fcns (labelled)

$$\begin{array}{cc}
\mathcal{A} \cup \mathcal{B} & A(z) + B(z) \\
\mathcal{A} \times \mathcal{B} & A(z) \times B(z) \\
\mathcal{A}' & A'(z) \\
\textsc{Seq}(\mathcal{C}) & \frac{1}{1-C(z)} \\
\textsc{Cyc}(\mathcal{C}) & \log \frac{1}{1-C(z)} \\
\textsc{Set}(\mathcal{C}) & \exp(C(z))
\end{array}$$

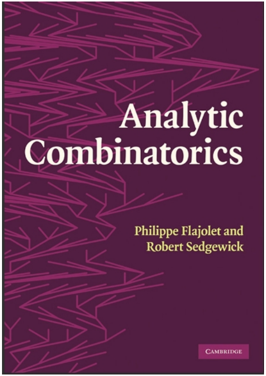
Examples

- Binary trees: $\mathcal{B} = \mathcal{Z} \cup \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$;

$$B(z) = z + zB(z)^2$$

- General trees: $\mathcal{T} = \mathcal{Z} \times \textsc{Set}(\mathcal{T})$;

$$T(z) = z \exp(T(z))$$



All these explicit series can be computed fast by Newton iteration.

Symbolic Method

Language and Gen. Fcns (labelled)

$\mathcal{A} \cup \mathcal{B}$	$A(z) + B(z)$
$\mathcal{A} \times \mathcal{B}$	$A(z) \times B(z)$
$\mathcal{A}'$	$A'(z)$
$\textsc{Seq}(\mathcal{C})$	$\frac{1}{1-C(z)}$
$\textsc{Cyc}(\mathcal{C})$	$\log \frac{1}{1-C(z)}$
$\textsc{Set}(\mathcal{C})$	$\exp(C(z))$

Examples

- Binary trees: $\mathcal{B} = \mathcal{Z} \cup \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$;
$$B(z) = z + zB(z)^2$$
- General trees: $\mathcal{T} = \mathcal{Z} \times \textsc{Set}(\mathcal{T})$;
$$T(z) = z\exp(T(z))$$

System (Σ): $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{H}}(\mathcal{Z}, \boldsymbol{\mathcal{Y}})$

Definition (Well-founded)

$\boldsymbol{\mathcal{H}}(\varnothing, \boldsymbol{\varnothing}) = \boldsymbol{\varnothing}$ and Jacobian $\partial\boldsymbol{\mathcal{H}}/\partial\boldsymbol{\mathcal{Y}}$ nilpotent at $(\varnothing, \boldsymbol{\varnothing})$.

Proposition

If (Σ) well-founded,

- $\boldsymbol{\mathcal{Y}}_{n+1} = \boldsymbol{\mathcal{H}}(\mathcal{Z}, \boldsymbol{\mathcal{Y}}_n)$ *converges to a limit species $\boldsymbol{\mathcal{Y}}$ [Joyal]*
- $\mathbf{Y}(z)$ *is analytic in a neighborhood of 0.*

Symbolic Method

Language and Gen. Fcns (unlabelled)

$\mathcal{A} \cup \mathcal{B}$	$A(z) + B(z)$
$\mathcal{A} \times \mathcal{B}$	$A(z) \times B(z)$
$\mathcal{A}'$	$A'(z)$
$\textsc{Seq}(\mathcal{C})$	$\frac{1}{1-C(z)}$
$\textsc{PSet}(\mathrm{C})$	$\exp(\sum(-1)^i C(z^i)/i)$
$\textsc{MSet}(\mathrm{C})$	$\exp(\sum C(z^i)/i)$
$\textsc{Cyc}(\mathrm{C})$	$\sum_{k\geq 1} \frac{\phi(k)}{k} \log \frac{1}{1-C(z^k)}$

Examples

- Binary trees: $\mathcal{B} = \mathcal{Z} \cup \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$;

$$B(z) = z + zB(z)^2$$

- General trees: $\mathcal{T} = \mathcal{Z} \times \textsc{Set}(\mathcal{T})$;

$$T(z) = z\exp(T(z) + T(z^2)/2 + \cdots)$$

System (Σ): $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{H}}(\mathcal{Z}, \boldsymbol{\mathcal{Y}})$

Definition (Well-founded)

$\boldsymbol{\mathcal{H}}(\varnothing, \boldsymbol{\varnothing}) = \boldsymbol{\varnothing}$ and Jacobian $\partial\boldsymbol{\mathcal{H}}/\partial\boldsymbol{\mathcal{Y}}$ nilpotent at $(\varnothing, \boldsymbol{\varnothing})$.

Proposition

If (Σ) well-founded,

- $\boldsymbol{\mathcal{Y}}_{n+1} = \boldsymbol{\mathcal{H}}(\mathcal{Z}, \boldsymbol{\mathcal{Y}}_n)$ *converges to a limit species* $\boldsymbol{\mathcal{Y}}$ *[Joyal]*
- $\mathbf{Y}(z)$ *is analytic in a neighborhood of 0.*

Derivative

$$\mathcal{Y} = \mathcal{S} \cup \mathcal{P},$$
$$\mathcal{S} = \mathrm{Seq}(\mathcal{Z} \cup \mathcal{P}),$$
$$\mathcal{P} = \mathrm{Set}(\mathcal{Z} \cup \mathcal{S}).$$

$$\frac{\partial \boldsymbol{\mathcal{H}}}{\partial \boldsymbol{\mathcal{Y}}} =$$

Nilpotent at $(\varnothing, \boldsymbol{\varnothing})$: successive powers

Derivative

$$\mathcal{Y} = \mathcal{S} \cup \mathcal{P},$$
$$\mathcal{S} = \mathsf{Seq}(\mathcal{Z} \cup \mathcal{P}),$$
$$\mathcal{P} = \mathsf{Set}(\mathcal{Z} \cup \mathcal{S}).$$

construction	derivative
$\mathcal{A} \cup \mathcal{B}$	$\mathcal{A}' \cup \mathcal{B}'$
$\mathcal{A} \times \mathcal{B}$	$\mathcal{A}' \times \mathcal{B} \cup \mathcal{A} \times \mathcal{B}'$
$\textsc{Seq}(\mathcal{B})$	$\textsc{Seq}(\mathcal{B}) \times \mathcal{B}' \times \textsc{Seq}(\mathcal{B})$
$\textsc{Cyc}(\mathcal{B})$	$\textsc{Seq}(\mathcal{B}) \times \mathcal{B}'$
$\textsc{Set}(\mathcal{B})$	$\textsc{Set}(\mathcal{B}) \times \mathcal{B}'$

$$\frac{\partial \boldsymbol{\mathcal{H}}}{\partial \boldsymbol{\mathcal{Y}}} = \begin{pmatrix} \varnothing & \mathcal{E} & \mathcal{E} \\ \varnothing & \varnothing & \mathsf{Seq}(\mathcal{Z} \cup \mathcal{P}) \times \mathcal{E} \times \mathsf{Seq}(\mathcal{Z} \cup \mathcal{P}) \\ \varnothing & \mathsf{Set}(\mathcal{Z} \cup \mathcal{S}) \times \mathcal{E} & \varnothing \end{pmatrix}$$

Nilpotent at $(\varnothing, \boldsymbol{\varnothing})$: successive powers

$$\begin{pmatrix} \varnothing & \mathcal{E} & \mathcal{E} \\ \varnothing & \varnothing & \varnothing \\ \varnothing & \mathcal{E} & \varnothing \end{pmatrix}, \quad \begin{pmatrix} \varnothing & \mathcal{E} & \varnothing \\ \varnothing & \varnothing & \varnothing \\ \varnothing & \varnothing & \varnothing \end{pmatrix}, \quad \begin{pmatrix} \varnothing & \varnothing & \varnothing \\ \varnothing & \varnothing & \varnothing \\ \varnothing & \varnothing & \varnothing \end{pmatrix}.$$

Combinatorial Newton Iteration

Theorem

For any well-founded system $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{H}}(\mathcal{Z}, \boldsymbol{\mathcal{Y}})$*, starting with* $\mathcal{Y}_0 = \varnothing$*, each iteration*

$$\mathcal{Y}_{n+1} = \mathcal{N}_{\mathcal{H}}(\mathcal{Y}_n) = \mathcal{Y}_n \cup \bigcup_{k \geq 0} \left(\frac{\partial \boldsymbol{\mathcal{H}}}{\partial \boldsymbol{\mathcal{Y}}}(\mathcal{Z}, \mathcal{Y}_n) \right)^k \times (\mathcal{H}(\mathcal{Z}, \mathcal{Y}_n) - \mathcal{Y}_n)$$

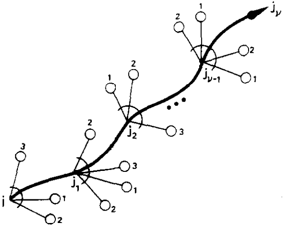
doubles the contact with $\mathcal{Y}$.

Generation by increasing Strahler numbers.

The big $\bigcup_{k \geq 0}$ can itself be computed by a Newton iteration.

$\rightarrow$ Efficient enumeration, numerical evaluation.

[Pivoteau, Salvy, Soria 2008]



Examples of Polynomial Systems

Random generation following given XML grammars

Grammar	nb eqs	max deg	nb sols	oracle (s.)	FGb (s.)
rss	10	5	2	0.02	0.03
PNML	22	4	4	0.05	0.1
xslt	40	3	10	0.4	1.5
relaxng	34	4	32	0.4	3.3
xhtml-basic	53	3	13	1.2	18
mathml2	182	2	18	3.7	882
xhtml	93	6	56	3.4	1124
xhtml-strict	80	6	32	3.0	1590
xmlschema	59	10	24	0.5	6592
SVG	117	10		5.8	$>$1.5Go
docbook	407	11		67.7	$>$1.5Go
OpenDoc	500			3.9	

[Darrasse 2008]

V Conclusion

Conclusion

1. Newton iteration is good for your complexity!

THE END

Conclusion

1. Newton iteration is good for your complexity!

THE END

2. Next steps:
   - faster location of singularity;
   - faster iteration close to the singularity (Schröder?).