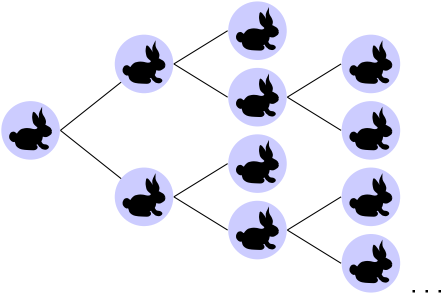
Algorithms for nonnegative quadratic vector equations

D. A. Bini[1] B. Meini[1] Federico Poloni[2,3]

[1]University of Pisa
[2]Scuola Normale Superiore, Pisa until end 2010
[3]Technische Universität Berlin (A. Von Humboldt postdoctoral fellow) now

18$^{\text{th}}$ Householder Symposium
Tahoe City, CA 2011

Markovian binary trees



MBTs model a colony of individuals that reproduce and die.
[Bean, Kontoleon, Taylor '04] [Hautphenne, Latouche, Remiche '08]

Simple example

$a = \mathbb{P}[$ dies without spawning$]$
$b = \mathbb{P}[$ spawns into two independent copies $]$

Question: starting from one individual, what is $\mathbb{P}[\text{extinction}]$?

A simple example

Simple example

$a = \mathbb{P}\,[$ dies without spawning$]$
$b = \mathbb{P}\,[$ spawns into two independent copies $]$

Question: starting from one individual, what is $\mathbb{P}\,[\text{extinction}]$?

It is a nice elementary problem: $x = \mathbb{P}\,[\text{extinction}]$

$$x = a + b\ x^2$$

- Either dies outright
- Or it spawns into two independent childs...

  ...and the progenies of both die out

The minimal solution

$x = a + b\,x^2$ has two nonnegative solutions. One is always 1,
for $a + b = 1$ (either reproduces or dies without)

Easy to prove that $\mathbb{P}\,[\text{extinction}]$ is the smaller solution. Three cases:

Subcritical $\mathbb{P}\,[\text{extinction}] = 1 > x_2$ (i.e. extinction = always)

Supercritical $\mathbb{P}\,[\text{extinction}] = x_2 < 1$

Critical limit case: 1 double solution
$\mathbb{P}\,[\text{extinction}] = 1$, but needs an infinite time on average

The vector case

Each 🐇 can be in N different states (e.g. age ranges)

$$a \in \mathbb{R}_+^N \qquad\qquad a_i = \mathbb{P}\left[\text{🐇}_i \text{ dies}\right]$$

$$b \in \mathbb{R}_+^{N\times N\times N} \qquad\qquad b_{ijk} = \mathbb{P}\left[\text{🐇}_i \text{ spawns into } \text{🐇}_j \text{ and } \text{🐇}_k\right]$$

b contains N^3 data!

Think to b as a vector-valued bilinear form

$$b : \mathbb{R}_+^N \times \mathbb{R}_+^N \to \mathbb{R}_+^N, \quad b(u,v) = \sum_{j,k} b_{ijk} u_j v_k$$

Our equation becomes

Markovian binary trees

$$x = a + b(x,x) \tag{MBT}$$

The classical algorithms

Markovian binary trees

$$x = a + b(x, x) \qquad \text{(MBT)}$$

$e = \text{ones}(N, 1)$ is always a solution
$\mathbb{P}[\text{extinction}]$ = minimal nonnegative solution

Up to 2^N nonnegative sol'ns, but there is always a minimal one:
$\widehat{x}$ s.t. $\widehat{x} \leq x$ (component-by-component) for any other solution x

Subcritical or critical: e is minimal, nothing to do

Supercritical: some other $0 \leq \widehat{x} \leq e$ is minimal: how to compute it?

The classical algorithms

Markovian binary trees

$$x = a + b(x, x) \tag{MBT}$$

Functional iterations [BKT '04]

$$x_{k+1} = a + b(x_k, x_k)$$

or something more elaborate, like

$$x_{k+1} = a + b(x_{k+1}, x_k)$$

i.e.

$$x_{k+1} = \left(I - b(\cdot, x_k)\right)^{-1} a$$

$b(\cdot, x_k)$: $\mathbb{R}_+^N \to \mathbb{R}_+^N$: just a matrix

The classical algorithms

Markovian binary trees

$$x = a + b(x, x) \tag{MBT}$$

Functional iterations [BKT '04]
Newton method [HLR '08]

$$x_{k+1} = \left(I - b(\cdot, x_k) - b(x_k, \cdot)\right)^{-1} a$$

+ variants, e.g. [Hautphenne, Van Houdt '10]

The classical algorithms

Markovian binary trees

$$x = a + b(x,x) \tag{MBT}$$

Functional iterations [BKT '04]
Newton method [HLR '08]

- When started from $x_0 = 0$, they converge monotonically:
  $0 = x_0 \le x_1 \le x_2 \le \cdots \le x^*$
- neat probabilistic interpretations:
  $x_k = \mathbb{P}\,[\text{extinction truncated to the } k\text{-th generation, or to a subtree}]$
- Become slower when close to critical:
  need more generations to capture the behaviour of the tree

Deflation

Close to a double solution, and for Newton double = trouble

But one of these solutions $x = e$ is known, we want to deflate it:
Set $y := e - x$ survival probability; (MBT) becomes

The optimistic equation

$$y = \underbrace{\left(b(e-y,\cdot) + b(\cdot, e)\right)}_{:= P_y} y = P_y y$$

Functional it'ns/Newton in this form: nothing changes, but...

Perron vector-based algorithms

The optimistic equation

$$y = \underbrace{\left(b(e-y,\cdot) + b(\cdot,e)\right)}_{:=P_y} y = P_y y$$

New way to see the same equation: y is the Perron vector of a matrix depending (linearly) on y itself

$$y = PV(P_y) \tag{PE}$$

(+suitable normalization for the eig'vec: $w^T \cdot \text{Residual} = 0$ for some w)

- Fixed point iteration based on (PE): $y_{k+1} = PV(P_{y_k})$
- Newton's method

Numerical experiments

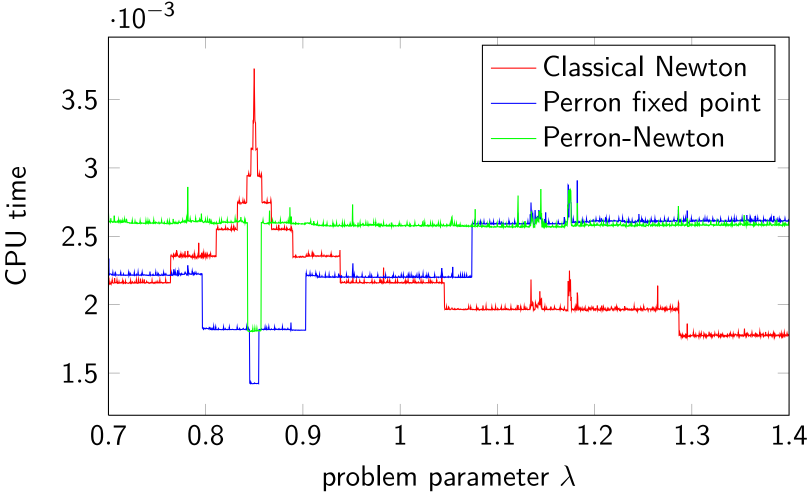


Figure: CPU time for a parameter-dependent problem [BKT '08, example 1]; lower=better

Numerical experiments

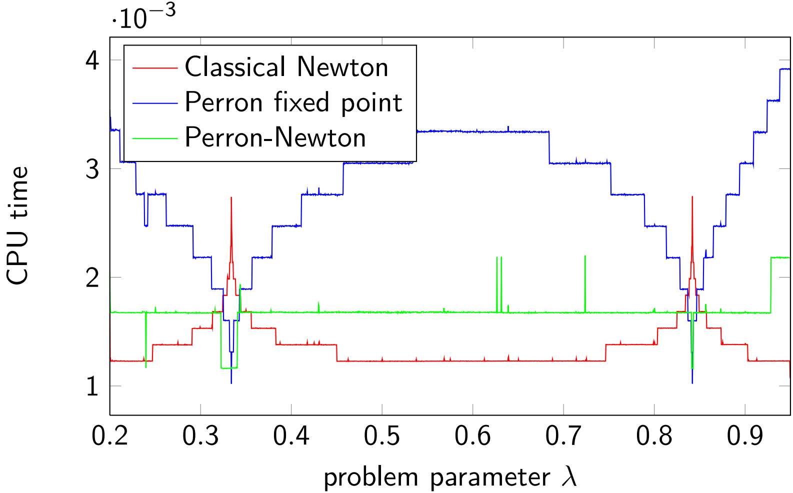


Figure: CPU time for a parameter-dependent problem [BKT '08, example 2]; lower=better

Convergence results

- Convergence is not monotonic
- Convergence is not guaranteed for very far-from-critical problems

Theorem [Meini, P., SIMAX 2011]

- Explicit formula for the Jacobian of the Perron iteration
- For a special normalization choice,
  if problem $\to$ critical then $\rho(\mathrm{Jac}) \to 0$

Thus, locally convergent for close-to-critical
with speed that tends to superlinear

Theorem [Bini, Meini, P., NLAA (to appear)]

When the algorithm converges, it converges to the right solution $\widehat{x}$

Applicability

We may ensure applicability even when strict positivity/irreducibility assumptions do not hold:

1. deflate away entries i s.t. $\widehat{x}_i = 0$: they can be determined in $O(N^3)$ from the nonzero pattern of a and b
2. all P_y have the same nonzero pattern; if they are reducible, we may split the problem into two subproblems (as with linear equations; idea: if $P_y = \begin{bmatrix} P_{11} & P_{12} \\ 0 & P_{22} \end{bmatrix}$, we can solve for the second block alone and back-substitute)

A unifying framework

Why is this problem interesting?

Mx=a+b(x,x)

$$XCX - AX - XD + B = 0 \qquad \text{(Nonsym. Riccati)}$$

$$PX^2 + QX + R = 0 \qquad \text{(QBD equation)}$$

$$\begin{cases} Ix = (Py) .{*} x + e \\ Iy = (Qx) .{*} y + e \end{cases} \qquad \text{(Transport theory)}$$

With a bit of vec(·), several matrix equations can be reduced to (MBT)
Although no known ($x = e$) solution → no PV-based algorithms

Open problem

Can we recover something similar from partial information (e.g., one known eigenpair of X)? Would carry over to many matrix equations

Common aspects

- minimal solution $x_* \geq 0$, i.e., $x_* \leq x$ for any other solution x
- functional iterations and Newton's method exhibit monotonic convergence: $0 = x_0 \leq x_1 \leq x_2 \leq \cdots \to x_*$
- close-to-critical problems: when close to a double solution, convergence is slower and more unstable

Common framework to work with several equations from different applications [P., to appear (LAA)]

Advantages:

- unified proofs: clear hypotheses, role of strict positivity of x_*
  no matrix structure or spectral properties needed
- unified algorithms: take an algorithm for one equation, apply it to the others

  Example a Newton variant [Hautphenne, Van Houdt '10] useful for the transport theory eqn

Conclusions

Open questions

- Understand doubling methods (SDA/Cyclic Reduction) in this framework:

  If we try to construct doubling for (MBT) we get Newton instead; are the two related?
- Shift technique + what happens to spectral properties?
- Perron-based algorithms without a "full" known solution $x = e$

Conclusions

Open questions

- Understand doubling methods (SDA/Cyclic Reduction) in this framework:

  If we try to construct doubling for (MBT) we get Newton instead; are the two related?
- Shift technique + what happens to spectral properties?
- Perron-based algorithms without a "full" known solution $x = e$