NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2000; **00**:1–6 *Prepared using* nlaauth.cls *[Version: 2002/09/18 v1.02]*

On the solution of a quadratic vector equation arising in Markovian Binary Trees

Dario A. Bini[1], Beatrice Meini[1] and Federico Poloni[2*]

[1] *Dipartimento di Matematica, Università di Pisa. Largo Pontecorvo 5, 56127 Pisa, Italy.* {bini,meini}@dm.unipi.it
[2] *Scuola Normale Superiore. Piazza dei Cavalieri 7, 56126 Pisa, Italy.* f.poloni@sns.it

SUMMARY

We present some advances, both from a theoretical and from a computational point of view, on a quadratic vector equation (QVE) arising in Markovian Binary Trees. Concerning the theoretical advances, some irreducibility assumptions are relaxed, and the minimality of the solution of the QVE is expressed in terms of properties of the Jacobian of a suitable function. From the computational point of view, we elaborate on the Perron vector-based iteration proposed in [1]. In particular we provide a condition which ensures that the Perron iteration converges to the sought solution of the QVE. Moreover we introduce a variant of the algorithm which consists in applying the Newton method instead of a fixed-point iteration. This method has the same convergence behaviour as the Perron iteration, since it tends to converge faster for close-to-critical problems. Moreover, unlike the Perron iteration, the method has a quadratic convergence. Finally, we show that it is possible to alter the bilinear form defining the QVE in several ways without changing the solution. This modification has an impact on convergence speed of the algorithms.
Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: Markovian binary tree, branching process, Newton method, Perron vector, fixed-point iteration

1. Introduction

Markovian Binary Trees (MBTs) are a particular family of branching processes, which are used to model the growth of populations consisting of several types of individuals who evolve independently and may produce a variable number of offsprings during their lifetime. MBTs have applications in biology, epidemiology and also in telecommunication systems. We refer the reader to [2, 3] for definitions, properties and applications.

One important issue related to MBTs is the computation of the extinction probability of the population, which can be characterized as the minimal nonnegative solution

$$x^* \in \mathbb{R}^N_+, \qquad \text{with } \mathbb{R}^N_+ := \{v \in \mathbb{R}^N : v_i \geq 0, i = 1, \ldots, N\},$$

of the quadratic vector equation (QVE for short)

$$x = a + b(x, x), \tag{1}$$

*Correspondence to: Scuola Normale Superiore. Piazza dei Cavalieri 7, 56126 Pisa, Italy.

Copyright © 2000 John Wiley & Sons, Ltd.

where $a \in \mathbb{R}_+^N$, $b : \mathbb{R}_+^N \times \mathbb{R}_+^N \to \mathbb{R}_+^N$ is a vector-valued bilinear form, such that the vector $e = (1, 1, \ldots, 1)^T$ is always a solution of (1). To denote the entries of b in coordinates, we use the notation $b_{ijk} := e_i^T b(e_j, e_k)$, where e_ℓ is the ℓth vector of the canonical basis. With this choice,

$$(b(x, y))_i = \sum_{j,k} b_{ijk} x_j y_k.$$

In many papers the notation $b(s, t) = B(s \otimes t)$, with $B \in \mathbb{R}_+^{N \times N^2}$ and $\otimes$ denoting the Kronecker product, is used instead; one can see that the two representations are equivalent. We favor the former, since it highlights the symmetry features of the problem. We mention the fact that the functions obtained by fixing the first or the second argument of the bilinear form, i.e., $b(y, \cdot)$ and $b(\cdot, z)$ for suitable $y, z \in \mathbb{R}_+^N$, are linear maps from $\mathbb{R}_+^N$ to itself, and thus they can be represented by $N \times N$ matrices with nonnegative entries.

The MBT is called subcritical, supercritical or critical if the spectral radius $\rho(R)$ of the matrix

$$R := b(e, \cdot) + b(\cdot, e) \tag{2}$$

is strictly less than one, strictly greater than one, or equal to one, respectively.

Under the stated assumptions, one can prove the existence of a minimal nonnegative solution in the componentwise ordering. A proof using minimal hypotheses is presented in [4]. In the subcritical and critical cases the minimal nonnegative solution is the vector of all ones, while in the supercritical case $x^* \leq e$, $x^* \neq e$. Thus, only the supercritical case is of interest for the computation of x^*.

Moreover, in the following we shall focus on the case in which $x^* > 0$. It is shown in [4] how to detect reliably the cases when this property does not hold, and reduce them to problems of lower dimension with strictly positive minimal solution.

Several iterative methods have been proposed and analyzed for computing the vector x^*. In [2] the authors propose two fixed point iterations with linear convergence, called *depth* and *order* algorithms. Another linearly convergent algorithm, called *thicknesses* algorithm, is proposed in [3]. In [5] and in [6] two variants of Newton's method are proposed. A different algorithm, based on a Perron vector iteration, is proposed in [1]. This algorithm, unlike classical iterative methods, tends to converge faster when applied to close to critical problems.

In this paper we provide theoretical and computational advances concerning the QVE (1). We first show that the matrix R of (2) can be assumed to be irreducible, since if it were reducible, we may reduce the problem of solving (1) to the problem of solving QVEs of smaller dimension, whose associated matrix R is irreducible. Assuming that R is irreducible, we provide a new characterization of the minimal nonnegative solution x^*, in terms of the properties of the Jacobian of the function $F(x) = x - a - b(x, x)$, evaluated at $x = x^*$. This property, which complements the results in [4], allows us to give a condition which ensures that the limit of the Perron vector-based iteration provides the sought solution x^* of the quadratic vector equation.

Moreover, we introduce a variant of the Perron vector-based iteration, which consists in applying the Newton method instead of a fixed-point iteration. This method is quadratically convergent, and has the same convergence behaviour as the Perron iteration, as it tends to converge faster for close-to-critical problems. The number of iterations needed by this variant is usually lower than the number of iterations needed by the original Perron iteration. However, due to the larger complexity of the single iteration step, the Newton-based method is generally slower than the Perron iteration, in terms of total computational time.

Finally, we show that it is possible to alter the bilinear $b(x, y)$ form definining the QVE in several ways without changing the solution. This modification has an impact on convergence speed: in most

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

examples, making the wrong choice can double the number of iterations needed. We show that, at least on the experiments reported, the best results are given by a symmetrization of the original bilinear form.

The paper is organized as follows. In Section 2 we recall classical algorithms based on fixed point iterations, while in Section 3 we recall the Perron-based iteration. In Section 4 we discuss the case where the matrix R of (2) is reducible, and we reduce the QVE to smaller size QVEs whose associated matrix R is irreducible. In Section 5 the minimality of the solution x^* of the QVE is expressed in terms of properties of the Jacobian of the function $F(x)$ at $x = x^*$. This result is used in Section 6 to ensure that the limit of the Perron-based iteration provides the sought solution x^*. The Newton version of the Perron-based iteration is proposed in Section 7. In Section 8 the choice of the bilinear form $b(x, y)$ is discussed. The results of the numerical experiments are presented and discussed in Section 9. We draw conclusions in Section 10.

2. Classical iterations

Several iterative methods have been proposed and analyzed for computing the vector x^*. In [2] the authors propose two iterations with linear convergence, called *depth* and *order* algorithms. The former consists in the simple functional iteration

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

the latter in

$$(I - b(\cdot, x_k))x_{k+1} = a, \tag{3}$$

or in the alternative version

$$(I - b(x_k, \cdot))x_{k+1} = a, \tag{4}$$

obtained by swapping the "left" and "right" branches of the binary tree. The two versions have in general different convergence speed. The *thicknesses* algorithm, still linearly convergent, is proposed in [3] and consists in alternating iterations of (3) and (4).

In [5] the authors apply the Newton method to the map

$$F(x) := x - a - b(x, x), \tag{5}$$

obtaining the iteration defined by

$$(I - b(x_k, \cdot) - b(\cdot, x_k))x_{k+1} = a - b(x_k, x_k), \tag{6}$$

which converges quadratically. Its convergence speed is usually much higher than that of the previous, linearly-convergent iterations. A modification of the Newton method, which increases slightly its convergence speed, has been proposed in [6].

All these methods have probabilistic interpretations, in that their k-th iterate x_k can be interpreted as the probability of extinction of the process restricted to a special subtree $\mathcal{T}_k$. Each of them provides a sequence $\{x_k\}_k$ of nonnegative vectors, with $x_0 = (0, \ldots, 0)^T$, which converges monotonically to the minimal nonnegative solution x^*. A common feature of all these methods is that their convergence speed slows down when the problem, while being supercritical, gets *close to critical*, i.e., the vector x^* approaches the vector of all ones. This happens because the mean extinction time increases, and thus sampling larger and larger trees is needed to capture the behavior of the iteration.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

3. A Perron-vector-based iteration

In [1], the authors propose an iterative scheme based on a different interpretation. Let us suppose for now that the nonnegative matrix R, as defined in (2), is irreducible — we discuss this assumption in Section 4. Then, since irreducibility depends only on the zero pattern of the matrix, $b(u,\cdot)+b(\cdot,v)$ is irreducible for each $u,v\in\mathbb{R}^N_+$ with strictly positive entries.

If we set $y=e-x$, equation (1) becomes

$$y=b(y,e)+b(e,y)-b(y,y). \tag{7}$$

A special solution of (7) is $y^*=e-x^*$, where x^* is the minimal nonnegative solution of (1). Notice that $0\le y^*\le e$. In the probability interpretation of Markovian Binary Trees, since x^* represents the extinction probability, then $y^*=e-x^*$ can be interpreted as survival probability. In particular, y^*_i is the probability that a colony starting from a single individual in state i does not become extinct in a finite time. The three summands in the right-hand side of (7) also admit an interesting probabilistic interpretation [1].

If we set $H_y:=b(\cdot,e)+b(e-y,\cdot)$, equation (7) becomes

$$y=H_y y. \tag{8}$$

If H_y is nonnegative and irreducible (which happens for sure if $y<e$, in view of the irreducibility of R), then the Perron-Frobenius theorem implies that $\rho(H_{y^*})=1$ and y^* is the Perron vector of the matrix H_{y^*}.

This interpretation allows to design new algorithms for computing y^* and x^*. Applying a functional iteration directly to (8), or the Newton method, gives nothing new, since we just made a change of variable. However, if we define $\mathrm{PV}(M)$ as the map that associates with a nonnegative irreducible matrix M its Perron vector, we may rewrite (8) as

$$y=\mathrm{PV}(H_y). \tag{9}$$

We may apply a fixed-point iteration to solve (9), thus generating a sequence $\{y_k\}_k$ of positive vectors such that the vector y_{k+1} is the Perron vector of the matrix H_{y_k}, i.e.,

$$y_{k+1}=\mathrm{PV}(H_{y_k}). \tag{10}$$

A suitable normalization of the Perron vector, consistent with the solution, is needed to obtain a well-posed iteration. An optimal normalization choice is suggested in [1]. If we take w as the Perron vector of the nonnegative irreducible matrix R^T, then we may normalize y_{k+1} so that

$$w^T(y_{k+1}-b(y_{k+1},e)-b(e,y_{k+1})+b(y_{k+1},y_{k+1}))=0, \tag{11}$$

i.e., we impose that the residual of (8) for $y=y_{k+1}$ is orthogonal to w. With this choice, one can prove [1] that the convergence speed of the sequence $\{y_k\}_k$ defined in (10), with the normalization condition (11), is linear with a small convergence factor for close-to-critical problems, and tends to superlinear as the considered problems approach criticality. Thus, although the convergence of this method is linear, surprisingly its speed *increases* as the problem gets close to critical, unlike the classical iterations.

4. Dealing with reducible R

The following result shows that when R is reducible we can reduce a QVE to two lower-dimensional problems to be solved successively with a kind of back-substitution. Therefore, for the solution of a generic QVE, we only need to apply the Perron iteration to the case in which R is irreducible.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

Theorem 1. *Suppose that, for a QVE* (1) *with* $x^* > 0$*, we have*

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

where R_{11} *is* $M \times M$ *and* R_{22} *is* $(N-M) \times (N-M)$*. Let*

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad x^* = \begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix}, \qquad a = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

be partitioned accordingly. Let

$$P := \begin{bmatrix} I_M & 0_{M\times(N-M)} \end{bmatrix}, \qquad Q := \begin{bmatrix} 0_{(N-M)\times M} & I_{N-M} \end{bmatrix},$$

be the restrictions to the first M *and last* $N-M$ *components respectively. Let us define the bilinear form on* $\mathbb{R}_+^{N-M}$

$$b_2(u,v) := Qb(Q^T u, Q^T v).$$

Moreover, for each $y \in \mathbb{R}_+^{N-M}$*, let us define*

$$T_y := I_M - Pb(\cdot, Q^T y)P^T - Pb(Q^T y, \cdot)P^T, \qquad a_y := {T_y}^{-1}(a_1 + Pb(Q^T y, Q^T y)),$$

and the bilinear form on $\mathbb{R}_+^M$

$$b_y(u,v) := {T_y}^{-1} Pb(P^T u, P^T v).$$

Then,

1. x *solves* (1) *if and only if* T_{x_2} *is nonsingular, and its block components* x_2 *and* x_1 *solve respectively the two quadratic vector equations*
$$x_2 = a_2 + b_2(x_2, x_2) \tag{12}$$
and
$$x_1 = a_{x_2} + b_{x_2}(x_1, x_1). \tag{13}$$
2. *If* x^* *is the minimal solution to* (1)*, then* x_1^* *and* x_2^* *are the minimal solution to* (13) *and* (12) *respectively.*

Proof First notice that since P and Q are restrictions to complementary sets of entries, (1) holds if and only if it holds on both sets, i.e.,

$$x_1 = a_1 + Pb(x,x), \tag{14a}$$
$$x_2 = a_2 + Qb(x,x). \tag{14b}$$

Since $R_{ij} = \sum_{k=1}^{N}(b_{ijk} + b_{ikj})$, it follows from the block structure of R (and from $b_{ijk} \geq 0$) that $b_{ijk} = 0$ whenever $i > M$ and either $j \leq M$ or $k \leq M$. This implies that the second block row of $b(u,v)$ depends only on the second block rows of u and v. We can write this formally as $Qb(u,v) = Qb(Q^TQu, Q^TQv)$. Then, (14b) is equivalent to (12).

By exploiting bilinearity and the fact that $P^TP + Q^TQ = I_N$, we can rewrite (14a) as

$$x_1 = a_1 + Pb(P^Tx_1, P^Tx_1) + Pb(P^Tx_1, Q^Tx_2) + Pb(Q^Tx_2, P^Tx_1) + Pb(Q^Tx_2, Q^Tx_2),$$

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

or

$$T_{x_2}x_1 = a_1 + Pb(Q^T x_2, Q^T x_2) + Pb(P^T x_1, P^T x_1). \tag{15}$$

Since the right-hand side is nonnegative and x_1 is positive, the Z-matrix T_{x_2} is an M-matrix. Notice that T_{x_2} cannot be singular, otherwise $0 = a_1 = Pb(Q^T x_2, Q^T x_2) = Pb(P^T x_1, P^T x_1)$ and $\begin{bmatrix}0^T & x_2^T\end{bmatrix}^T$ would be a solution of (1), contradicting the fact that the minimal solution is strictly positive. Therefore, we may multiply (15) by its inverse to get (13). The steps in the above proof can be reversed provided T_{x_2} is nonsingular, thus the converse implication holds as well.

Let us now prove the second part of the theorem. Equation (12) admits a minimal solution due to the general existence theorem (since it admits at least a solution); suppose it is $x_2 \neq x_2^*$; then, by minimality, $x_2 \leq x_2^*$. The matrix $T_{x_2} \geq T_{x_2^*}$ is an M-matrix, thus $T_{x_2^*}^{-1} \geq T_{x_2}^{-1}$. Therefore, $a_{x_2} \leq a_{x_2^*}$ and $b_{x_2} \leq b_{x_2^*}$. We have

$$x_1^* = a_{x_2^*} + b_{x_2^*}(x_1^*, x_1^*) \geq a_{x_2} + b_{x_2}(x_1^*, x_1^*),$$

thus the equation (13) has a supersolution, and this implies that it has a solution by [4, Lemma 5]. Let x_1 be its minimal solution; then, x is a solution to (1) by the first part of this theorem, but this is in contradiction with the minimality of x^*, since $x_2 \lneqq x_2^*$. Therefore x_2^* is the minimal solution to (12). If (13) admitted a solution $x_1 \lneqq x_1^*$, then by the first part of the theorem

$$\begin{bmatrix}x_1\\ x_2^*\end{bmatrix}$$

would be a solution to (1), and this again contradicts the minimality of x^*.

□

Let $F'_x := I - b(x, \cdot) - b(\cdot, x)$ be the Jacobian of the map $F(x)$ defined in (5) (see [5, 4]). Notice that if $x > 0$, then F'_x has the same positivity pattern as R, and thus is irreducible whenever R is. Moreover, when $x = e$ is a solution to (1), then the all-ones vectors of suitable dimension solve (12) and (13), thus the Perron vector-based iteration can be applied to the reduced problems as well.

5. An alternative characterization of minimality

The following theorem provides a practical criterion to check the minimality of a solution.

Theorem 2. *Let $x > 0$ be a solution of* (1) *and assume that R is irreducible. Then, F'_x is an M-matrix if and only if x is minimal.*

Proof The implication (x^* minimal) $\Rightarrow$ (F'_{x^*} is an M-matrix) has been proved in [4]. We prove the converse here. The proof is split in two different arguments, according to whether F'_x is a singular or nonsingular M-matrix.

Let F'_x be a nonsingular M-matrix, and let $\bar{x}$ be another nonnegative solution; we need to prove that $\bar{x} - x \geq 0$. From the Taylor expansion of $F(x)$ (and the fact that $F'' \leq 0$) we have

$$0 = F(\bar{x}) = F(x) + F'_x(\bar{x} - x) + \frac{1}{2}F''_x(\bar{x} - x, \bar{x} - x) \leq F'_x(\bar{x} - x),$$

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

that is, $F'_x(\bar{x} - x) \geq 0$. It suffices to multiply by $(F'_x)^{-1} \geq 0$ to get $\bar{x} - x \geq 0$, as needed.

Let now F'_x be a singular M-matrix. Suppose that x is not minimal, and $x^* \lneqq x$ is the minimal solution to (1). Then, $F'_{x^*} \geqq F'_x$ is a (singular or nonsingular) M-matrix, by the converse implication of this theorem. Thus, by the properties of M-matrices, F'_x must be a nonsingular M-matrix, which is a contradiction.

□

Notice that this characterization of minimality allows to deduce easily the fact, claimed above, that the solution e is minimal only in the subcritical and critical cases.

6. On the limit of the Perron iteration

The following result shows that, under reasonable assumptions, the limit of the Perron vector-based iteration is the minimal solution of (1).

Theorem 3. *Suppose that R is irreducible, and that $x^* > 0$. Suppose that the Perron iteration* (10)*, with normalizing condition* (11)*, converges to a vector y^* such that $y^* \leq e$. Then, $x = e - y^*$ is the minimal solution of* (1)*.*

Proof Let us first prove that the spectral radius of H_{y^*} is 1. The iterates of the Perron iteration satisfy

$$\lambda_{k+1} y_{k+1} = H_{y_k} y_{k+1}, \tag{16a}$$

$$w^T (y_{k+1} - H_{y_{k+1}} y_{k+1}) = 0. \tag{16b}$$

Taking the limit as $k \to \infty$ in (16), we get

$$\lambda^* y^* = H_{y^*} y^*, \tag{17a}$$

$$w^T (y^* - H_{y^*} y^*) = 0. \tag{17b}$$

Notice that λ^* is well-defined, as it may be defined as the common ratio between the components of $H_{y^*} y^*$ and those of y^*. We left-multiply (17a) by w^T to get $w^T(\lambda^* y^* - H_{y^*} y^*) = 0$, which, compared to (17b), tells us that $\lambda^* = 1$. In particular, this implies that $x = e - y^*$ is a solution of (1), as we may verify directly by back-substitution.

Moreover, $\rho(H_{y^*}) = 1$, and thus $I - H_{y^*} = I - b(e - y^*, \cdot) - b(\cdot, e)$ is a singular M-matrix. Thus the Z-matrix $F'_x = I - b(e - y^*, \cdot) - b(\cdot, e - y^*) \geq I - b(e - y^*, \cdot) - b(\cdot, e)$ is an M-matrix, too. By Theorem 2, this implies that $x = e - y^*$ is minimal.

□

7. The Perron–Newton method

We may also apply Newton's method for the solution of (9).

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

We first recall the following result from [1], which provides an explicit form for the Jacobian of the iteration map $G(y)$ defining the iteration (10) with the normalization (11), i.e.,

$$G(y) := \text{the Perron vector of } H_y, \text{ normalized s.t. } w^T \left(G(y) - H_{G(y)}G(y)\right) = 0.$$

Theorem 4. *Let y be such that H_y is nonnegative and irreducible. Let $u = G(y)$, and let v be such that $v^T H_y = \lambda v^T$, where $\lambda = \rho(H_y)$. Then the Jacobian of the map G at y is*

$$JG_y = \left(I - \frac{u\sigma_1^T}{\sigma_1^T u}\right)(H_y - \lambda I)^\dagger \left(I - \frac{uv^T}{v^T u}\right) b(\cdot, u), \tag{18}$$

where

$$\sigma_1^T := w^T(I - b(e-u, \cdot) - b(\cdot, e-u))$$

and the symbol † denotes the Moore–Penrose pseudo-inverse.

With the aid of this formula, we may define the Perron–Newton method for the solution of (1) as in Algorithm 1.

input: the bilinear form b (note that a is not necessary — in fact it can be deduced from $e = a + b(e, e)$)
input: the normalization vector $w > 0$ (a good choice is taking the Perron vector of R^T, see [1])
$y \leftarrow e$;
while *a suitable stopping criterion is not satisfied* **do**
    $u \leftarrow G(y)$;
    $J \leftarrow JG_y$ (computed using (18));
    $y \leftarrow y - (I - J)^{-1}(y - u)$;
end
if $0 \le y \le e$ **then**
    $x \leftarrow e - y$;
else
    (error: no convergence);
end

Algorithm 1: The Perron–Newton algorithm

A step of Newton's method basically requires a step of the Perron vector-based fixed-point iteration associated with (9), followed by the computation of a Moore–Penrose pseudoinverse and the solution of a linear system. Thus its cost is larger than, but still comparable to, the cost of a step of the Perron vector-based functional iteration. This is compensated by the fact that the Newton method has quadratic convergence, and thus requires less iterations.

The convergence properties of the Perron–Newton method for close-to-critical problems are similar to those of the Perron vector-based functional iteration. For close-to-critical problems one has $x^* \approx e$, therefore $\rho(JG_{y^*}) \approx 0$. Hence, by the Newton–Kantorovich theorem [7] there is convergence for sufficiently close-to-critical problems. The proof of Theorem 3 can be easily adapted to show that the limit point must correspond to the minimal solution of (1) if $0 \le y^* \le e$. Moreover, since $\rho(JG_{y^*}) \approx 0$, the matrix to invert is well-conditioned and $y - G(y)$ has a simple zero.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

8. On the choice of the bilinear form b

Equation (1), and thus its solution, depend only on the quadratic form $b(t,t) := B(t \otimes t)$; however, there are different ways to extend it to a (nonnecessarily symmetric) bilinear form $b(s,t)$. Namely, for each i and each $j \neq k$, we may alter simultaneously b_{ijk} and b_{ikj}, as long as their sum remains invariant, and they both remain positive. For example, we may switch the two terms in every such pair, obtaining the bilinear form $b^T(s,t) := b(t,s)$.

Some of the solution algorithms depend essentially on the choice of the bilinear extension: for instance, the two versions of the *order* algorithm. It is easy to see that (3) applied to b^T coincides with (4) applied to b, and vice versa. Instead, in the classical Newton's method (6), the bilinear form appears only in the expressions $b(x_k,\cdot) + b(\cdot,x_k)$ and $b(x_k,x_k)$, which are unaffected by this change. Thus the classical Newton method stays the same no matter which bilinear extension we choose.

On the other hand, one can see that the Perron-vector based functional iteration and its Newton based version do depend on the bilinear extension, and their convergence speed is affected by this choice. The expression of the bilinear form ultimately reflects a modeling aspect of the problem. While in the original definition of a branching process an individual splits into two new ones in two different states, it is often convenient to identify one as the "mother" and one as the "child", even if this distinction is artificial. In fact, we can safely redefine who is the mother and who is the child, as long as we do not change the total probability that an individual in state i generates two offsprings in states j and k. This corresponds exactly to changing the bilinear form b in the described way.

Among the possibilities for the modifications of b, we list the following.

Transposition $b^T(s,t) := b(t,s)$

Symmetrization $b^S(s,t) := \frac{1}{2}\left(b(s,t) + b^T(s,t)\right)$

Desymmetrization 1

$$(b^{D1})_{ijk} := \begin{cases} b_{ijk} + b_{ikj} & \text{if } j < k \\ b_{ijk} & \text{if } j = k \\ 0 & \text{if } j > k \end{cases}$$

Desymmetrization 2

$$(b^{D2})_{ijk} := \left(b^T\right)^{D1} = \begin{cases} b_{ijk} + b_{ikj} & \text{if } j > k \\ b_{ijk} & \text{if } j = k \\ 0 & \text{if } j < k \end{cases}$$

In the following section, we report numerical experiments performed with the above bilinear extensions and compare the computational times. We do not have a definitive result on which choice gives the best convergence: as is the case with the *order* algorithm, the optimal bilinear extension may vary in different instances of the problem.

9. Numerical experiments

We performed numerical experiments to assess the speed of the proposed methods. The tests were performed on a laptop (Intel Pentium M 735 1.70Ghz) with Matlab R2010a and considered two sample parameter-dependent problems.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

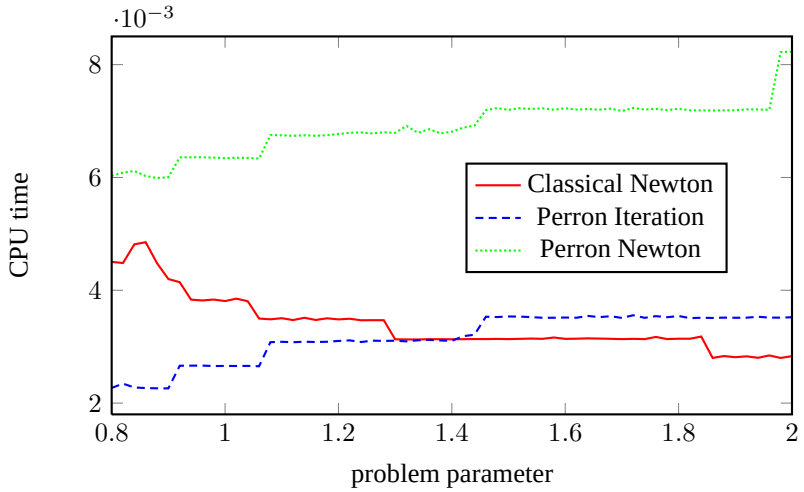

Figure 1. CPU time vs. parameter λ for P1 — lower=better

P1 a small-size Markovian binary tree with branches of varying length, described in [5, Example 1]. It is an MBT of size $N = 9$ depending on a parameter λ, which is critical for $\lambda \approx 0.85$ and supercritical for larger values of λ.

P2 a random-generated MBT of larger size ($N = 100$). It is created by generating a random bilinear form b, choosing a suitable a so that $a + b(e, e) = Ke$ for some K, and then scaling both a and b in order to eliminate K. We report the Matlab code used for its generation in Algorithm 2. Larger

```
input: the size N of the MBT and a parameter lambda> 0
e=ones(N,1);
rand('state',0);
b=rand(N,N*N);
K=max(b*kron(e,e))+lambda;
a=K*e-b*kron(e,e);
a=a/K;
b=b/K;
```

Algorithm 2: Generating a random MBT

choices of the parameter λ increase the values of a, i.e., the probability of immediate death, and thus enlarge the extinction probability making the process closer to critical. With $N = 100$, the process is critical for $\lambda \approx 4920$.

Figure 1 shows a plot of the computational times for classical Newton and the two Perron vector-based methods for different values of the parameter λ. Depth, order and thicknesses are not reported in the graph as they are much slower than these methods, as also shown by the experiments in [5]. While in close-to-critical cases the time for CN has a spike, the ones for PN and PI seems to decrease. While

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

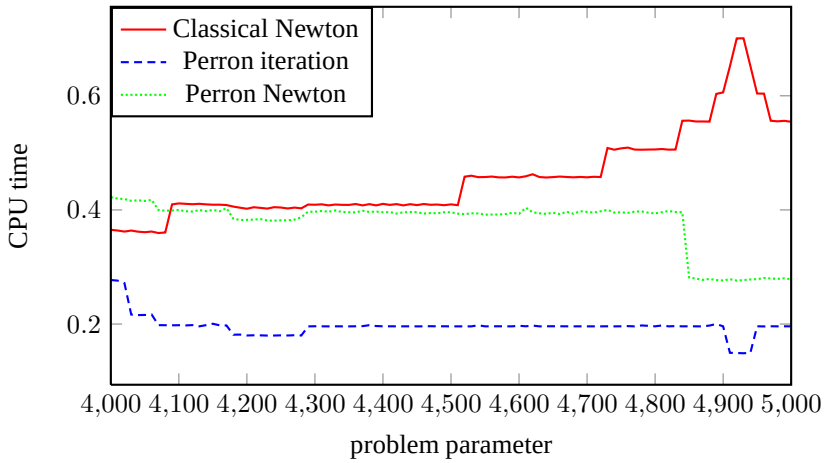

Figure 2. CPU time vs. parameter λ for P2

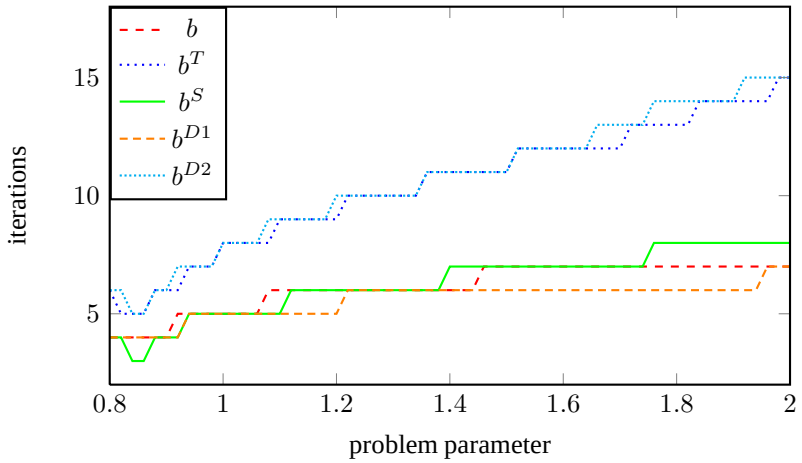

Figure 3. Number of steps needed for the Perron iteration for P1 with several variants of the bilinear form

having in theory worse convergence properties, the Perron iteration is faster than the Perron Newton method: the additional overhead of the pseudoinverse and of the computation of both left and right dominant eigenvector more than offsets the increased convergence rate.

Figure 2 shows the corresponding plot for the larger problem P2. We point out that two different methods were used to compute the Perron vectors in the two problems. For P2, we use `eigs`, which is based on an Arnoldi method [8]. On the other hand, for P1, due to the really small size of the problem, it is faster to compute a full eigenvector basis with `eig` and then select the Perron vector.

With this choice, both the Perron iteration and Perron–Newton method are faster than the classical Newton method on this larger-size problem, in the close-to-critical region.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

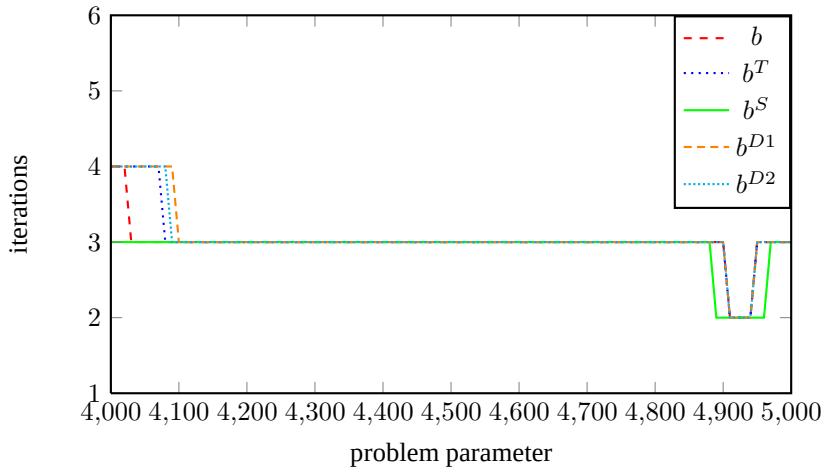

Figure 4. Number of steps needed for the Perron iteration for P2/ with several variants of the bilinear form

Figure 3 reports the number of iteration (which essentially grows as the CPU time) for the Perron iteration on P1 with several alternative bilinear forms equivalent to b. We see that among the two possible "branch switches", in this example the iteration with b converges faster than the one with b^T. Clearly this cannot be a general result: due to the involutory nature of this transposition operation, if we started with $\tilde{b} := b^T$, then the faster choice would have been $\tilde{b}^T = (b^T)^T = b$. Thus we cannot infer a rule for telling which of the two is preferable. Similarly, it is impossible to do a proper comparison among b^{D1} and B^{D2}. On the other hand, an interesting result is that the performance of the iteration with b^S seems to be on par with the better of the two.

Figure 4 reports the same comparison for the problem P2. The results are less pronounced than on the previous example: since the entries of the bilinear form b are generated randomly, the difference between the "left" and "right" branches of the binary tree should be less marked than in P1, where the two directions are intentionally unbalanced. Nevertheless, the symmetrized bilinear form consistently yields slightly lower iteration counts.

Therefore, based on these results, we suggest to apply the Perron iteration and Newton methods on the symmetrized bilinear form instead of the original one.

10. Conclusions

In this paper we presented several possible implementation variants of the Perron vector-based iteration introduced in [1]. A Newton method based on the same formulation of the problem is slightly less effective than the original iteration, although it maintains the same good convergence properties for close-to-critical problems. Moreover, we highlight the fact that there is a family of possible modifications to the bilinear form b that alter the form of solution algorithms, but not the original equation (1) and its solution. One of these modifications, the symmetrization, seems to achieve better results than the original formulation of the numerical algorithms.

Moreover, we present a couple of theoretical results on quadratic vector equations that show how

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls

to ensure that the obtained solution is the desired one, and how to deal with the problems in which an irreducibility assumption is not satisfied.

REFERENCES

1. Meini B, Poloni F. A Perron iteration for the solution of a quadratic vector equation arising in Markovian binary trees 2010. URL `http://arxiv.org/abs/1006.0577`, submitted for publication.
2. Bean NG, Kontoleon N, Taylor PG. Markovian trees: properties and algorithms. *Annals of Operations Research* 2008; **160**:31–50, doi:10.1007/s10479-007-0295-9. URL `http://dx.doi.org/10.1007/s10479-007-0295-9`.
3. Hautphenne S, Latouche G, Remiche MA. Algorithmic approach to the extinction probability of branching processes. *Methodology and Computing in Applied Probability* 2009; doi:10.1007/s11009-009-9141-7. URL `http://dx.doi.org/10.1007/s11009-009-9141-7`, to appear in print.
4. Poloni F. Quadratic vector equations 2010. URL `http://arxiv.org/abs/1004.1500`.
5. Hautphenne S, Latouche G, Remiche MA. Newton's iteration for the extinction probability of a Markovian binary tree. *Linear Algebra and its Applications* 2008; **428**(11-12):2791–2804, doi:10.1016/j.laa.2007.12.024. URL `http://dx.doi.org/10.1016/j.laa.2007.12.024`.
6. Hautphenne S, Van Houdt B. On the link between Markovian trees and tree-structured Markov chains. *European Journal of Operational Research* 2010; **201**(3):791–798, doi:10.1016/j.ejor.2009.03.052. URL `http://dx.doi.org/10.1016/j.ejor.2009.03.052`.
7. Ortega JM, Rheinboldt WC. *Iterative solution of nonlinear equations in several variables*, *Classics in Applied Mathematics*, vol. 30. Society for Industrial and Applied Mathematics (SIAM): Philadelphia, PA, 2000. Reprint of the 1970 original.
8. Lehoucq RB, Sorensen DC, Yang C. *Arpack User's Guide: Solution of Large-Scale Eigenvalue Problems With Implicityly Restorted Arnoldi Methods (Software, Environments, Tools)*. SIAM, 1997.

Copyright © 2000 John Wiley & Sons, Ltd.

Prepared using nlaauth.cls