On the convergence of the power series algorithm

Gerard Hooghiemstra [*]   Ger Koole [†]

Abstract

In this paper we study the convergence properties of the power series algorithm, which is a general method to determine (functions of) stationary distributions of Markov chains. We show that normalization plays a crucial role and that the convergence can be improved by introducing some minor changes in the algorithm. We illustrate this with several numerical examples.

Key words and phrases. Coates graphs, Markov chains, power series, stationary probabilities

AMS 1991 *Subject classifications.* Primary 60K25; Secondary 60J25

1 Introduction

The power series algorithm (PSA) is a general method to determine stationary distributions of Markov chains. The main idea behind the PSA is the following. By introducing an artificial parameter λ in the transition rates of the Markov chain the stationary probabilities become functions of λ. The coefficients of the power series expansions around $\lambda = 0$ of these functions can be calculated recursively if certain conditions on the transition rates are satisfied. The partial sums are taken as approximations of the steady state probabilities. For many models λ has some logical interpretation, such as the load of the system.

The PSA was introduced in Hooghiemstra et al. [8], and applied to a coupled processor model (to which we will come back later). They observed that in general there is no guarantee that the power series converges for certain values of the load λ, although the PSA applied to their model converges as long as λ is such that the system is stable (cf. Bavinck et al. [2]). After that Blanc and some of his students (cf. Blanc [3], Van den Hout [9] and the papers cited there) applied the PSA to many queueing models. For some of these models and certain choices of λ the PSA did not converge. To deal with this problem Blanc introduced two different methods, namely conformal mappings and the ε-algorithm. Using a conformal map is an analytical method by which one hopes to isolate singularities

---

[*]Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands

[†]Vrije Universiteit, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands

in the power series. The ε-algorithm is a general method for finding limits for diverging or slowly converging series (cf. Wynn [11]). Recently Koole [10] showed that the PSA can be applied to any Markov chain; concerning convergence he showed that the ε-algorithm gives the exact solution to the balance equations for finite state chains.

In this paper we focus on the convergence of the PSA. In our opinion this is strongly related to normalization. We illustrate this by first analyzing an example with 4 states.

Example (truncated coupled processor model) Consider a simple 4-state continuous time Markov chain, with state space $S = \{0, 1, 1', 2\}$, and transition intensities q_{xy} defined by $q_{01} = q_{01'} = q_{12} = q_{1'2} = \lambda$, $q_{10} = q_{1'0} = \mu$, and $q_{21} = \mu_1 = \mu - \mu_2 = \mu - q_{21'}$, $\mu_1 \in [0, \mu]$. All other transitions rates are 0. This model can be seen as a truncation of the coupled processor model that we will study later.

Some easy calculations show that the steady state probabilities p_x can be written as $p_x = r_x / \sum_{y \in S} r_y$, with $r_0 = \mu^2(\lambda + \mu)$, $r_1 = \lambda(\mu^2 + 2\lambda\mu_1)$, $r_{1'} = \lambda(\mu^2 + 2\lambda\mu_2)$ and $r_2 = 2\lambda^2(\lambda + \mu)$. Now consider p_x as a function of λ. Note first that all p_x are rational functions, which are analytic in 0, because $R = \sum_{y \in S} r_y = (\lambda + \mu)(2\lambda^2 + 2\mu\lambda + \mu^2)$, has no zero in $\lambda = 0$. Thus power series expansions around $\lambda = 0$ exist. The PSA is able to find the coefficients of the power series expansion for p_x.

Assume (without restricting generality) that $\mu = 1$. Because $p_0 = 1/(2\lambda^2 + 2\lambda + 1)$ and the zeroes of $2\lambda^2 + 2\lambda + 1$ are located at $(-1 \pm i)/2$, the power series expansion of p_0 converges for λ in the interval $[0, 1/\sqrt{2}) \approx [0, 0.707)$. Also p_2 has denominator $2\lambda^2 + 2\lambda + 1$, but p_1 and $p_{1'}$ both have denominator $(\lambda + 1)(2\lambda^2 + 2\lambda + 1)$ (unless $\mu_1 = \mu_2$), giving an additional zero at $\lambda = -1$. This leads to the same convergence interval. Thus the partial sums found by the PSA converge up to $\lambda = 1/\sqrt{2}$ to the exact solution, beyond this value the series diverge.

In practice this does not mean that the use of the PSA is limited to $\lambda < 1/\sqrt{2}$. Using a conformal mapping would allow for any λ (as all zeroes lie in the complex halfplane with negative real part). The ε-algorithm is also capable of finding the correct values. However, it is of theoretical interest to understand why the PSA fails to perform well for $\lambda > 1/\sqrt{2}$, i.e., to try to derive general results on the location of the poles of the power series, and to construct algorithms that converge better than the PSA.

In the PSA, p_x for $x \neq 0$ is determined by the balance equations, and p_0 by the normalizing equation (i.e., $\sum_{y \in S} p_y = 1$). Now if we would solve

$$\{\sum_y p'_x q_{xy} = \sum_y p'_y q_{yx}, \ x \neq 0, \ p'_0 = 1\}$$

instead of

$$\{\sum_y p_x q_{xy} = \sum_y p_y q_{yx}, \ x \neq 0, \ \sum_y p_y = 1\},$$

then we would get as solution $p'_0 = 1$, $p'_1 = \lambda(1 + 2\lambda\mu_1)/(\lambda + 1)$, $p'_{1'} = \lambda(1 + 2\lambda\mu_2)/(\lambda + 1)$, and $p'_2 = 2\lambda^2$. The expansions of p'_x converge for all λ if $\mu_1 = \mu_2$ (the factor $1 + \lambda$ cancels out), and only p_1 and $p_{1'}$ have poles at $\lambda = -1$ if $\mu_1 \neq \mu_2$. Thus by *normalizing afterwards*

we increase the convergence region to $\lambda \in [0, \infty)$ if $\mu_1 = \mu_2$, and $\lambda \in [0, 1)$ if $\mu_1 \neq \mu_2$. We denote this variant of the PSA as PSA/N.

We saw in the example that the PSA/N improves drastically the convergence compared to the PSA. A further improvement can be obtained by removing the last pole, i.e., by starting with $p'_0 = 1 + \lambda$ instead of $p'_0 = 1$. This however would call for insight in the structure of the stationary probabilities of the model at hand.

In the next section we introduce the PSA. Then we discuss the structure of the steady state distributions. Based on that we compare PSA with PSA/N. We conclude with numerical experiments for the coupled processor model, for the $M/M/s$ queue with critical customers, and for the shortest queue model.

2   The power series algorithm

Consider a continuous time Markov chain with countable state space S. As in Koole [10], we start from:

Assumption 2.1 *The states can be classified in levels $0, 1, \ldots$ (we shall denote these levels by L) such that:*
(i) The level 0 consists of only one state (denoted by 0);
(ii) The states within each level can be ordered such that there are no transitions to higher ordered states within that level;
(iii) Transitions to lower level states are possible in each state $x \neq 0$.

We assume that our Markov chain has transitions of the following form:

$$q_{xy} = \lambda^{(L(y)-L(x))^+} q'_{xy} \tag{1}$$

where $x^+ = \max(0, x)$, $\lambda > 0$, and all q'_{xy} are independent of λ. The idea is to write the stationary probability p_x or $p(x)$ as

$$p(x) = \sum_{k=0}^{\infty} b_k(x) \lambda^{L(x)+k}, \tag{2}$$

where it is assumed that the series converges for all relevant λ.

For many queueing models the transitions are of the form (1) for a level function L satisfying Assumption 2.1, with λ the arrival rate of the system. For other chains, where the choice of λ is less obvious, we can always find a function L satisfying Assumption 2.1 (see [10]), introduce the parameter λ in the transitions as in (1), apply the PSA, and take $\lambda = 1$ afterwards.

The central idea of the PSA is that, for an appriopriate level function L, the $b_k(x)$ can be computed recursively by substituting the power-series (2) in the equilibrium equations

of the Markov chain. This procedure renders:

$$\left(\sum_{y:L(y)\leq L(x)} q'_{xy} \right) b_k(x) = \sum_{y:L(y)\leq L(x)} q'_{yx} b_k(y)$$
$$+ \sum_{y:L(y)>L(x)} q'_{yx} b_{k-L(y)+L(x)}(y) - \sum_{y:L(y)>L(x)} q'_{xy} b_{k-L(y)+L(x)}(x), \qquad (3)$$

where we assume that $b_k(x) = 0$ for all x if $k < 0$. Note that Assumption 2.1 guarantees that for $x \neq 0$ the coefficients $b_k(x)$ can be calculated recursively, by computing the coefficients $b_k(x)$ in increasing order of k and in increasing level of x.

The $b_k(0)$ can be obtained from the norming condition, $\sum_{y\in S} p_y = 1$, which yields $b_0(0) = 1$ and, for $k > 0$,

$$b_k(0) = - \sum_{x:1\leq L(x)\leq k} b_{k-L(x)}(x). \qquad (4)$$

Equations (3) and (4) together constitute the PSA. In practice all coefficients $b_k(x)$ with $L(x) + k \leq K$ are computed for some K, after which the partial sums are computed. Note that, due to (4), the sum of all partial sums is always equal to 1.

Example (truncated coupled processor model, continued) We illustrate the PSA by applying it to the example of the introduction. With $L(0) = 0$, $L(1) = L(1') = 1$ and $L(2) = 2$, Equation (3) gives (for $\mu = 1$):

$$b_k(1) = b_k(0) + \mu_1 b_{k-1}(2) - b_{k-1}(1),$$
$$b_k(1') = b_k(0) + \mu_2 b_{k-1}(2) - b_{k-1}(1'),$$
$$b_k(2) = b_k(1) + b_k(1').$$

The normalization gives

$$b_k(0) = -b_{k-1}(1) - b_{k-1}(1') - b_{k-2}(2).$$

Starting with $b_0(0) = 1$, we find $b_0(1) = b_0(1') = 1$ and $b_0(2) = 2$. Then $b_1(0) = -2$, $b_1(1) = -3 + 2\mu_1$, $b_1(1') = -3 + 2\mu_2$, $b_1(2) = -4$, $b_2(0) = 2$, etc. These are indeed the first indices of the power series expansions of the functions that we found earlier. Using their functional expressions we know that the power series converge only for $\lambda < 1/\sqrt{2}$.

Thus the PSA computes the coefficients of the power series of the steady state probabilities around zero. However, there is no guarantee whatsoever that these power series converge, and indeed, for some models and choices of λ, they do not. To study this phenomenon, we start with the study of the structure of the steady state probabilities of finite Markov chains.

4

3   The structure of finite chains

Finding the equilibrium distribution of finite irreducible Markov chains means solving a matrix equation. From numerical point of view Gauss-elimination is the most effective algorithm. Here we are interested in theoretical results concerning the PSA. To solve the matrix equation we use Cramer's rule, as in [10], or apply a Lemma of Freidlin and Wentzell [7], which uses the transition structure of the chain. We introduce both techniques and prove their equivalence. Then we discuss the consequences for the PSA.

Consider a finite state Markov chain ($|S| = N < \infty$) with a single recurrent class. In this section we number the states with $1, \ldots, N$, to avoid confusion with the numbering of the rows and columns of the corresponding transition matrix. In the consecutive sections however we return to the old numbering of states starting from 0. Let H be its infinitesimal generator, i.e., $h_{xy} = q_{xy}$ if $x \neq y$, and $h_{xx} = - \sum_y q_{xy}$. Construct H' from H by replacing the first column by $e = (1, \ldots, 1)^t$. Then the system

$$\{\sum_y p_x q_{xy} = \sum_y p_y q_{yx}, x \neq 1, \sum_y p_y = 1\},$$

which unique solution is the steady state vector, can be written in matrix notation as $p^t H' = e_1^t$, where $e_1 = (1, 0, \ldots, 0)^t$.

To compute the stationary probabilities p_x we can apply Cramer's rule, that is,

$$p_x = \frac{|H'_x|}{|H'|},$$

where H'_x is obtained from H' by replacing the xth row by e_1, and where we denote by $|\cdot|$ the determinant of a matrix.

Example (truncated coupled processor model, continued)   We illustrate Cramer's rule with our 4-state example. We find

$$H' = \begin{pmatrix} 1 & \lambda & \lambda & 0 \\ 1 & -(\lambda+\mu) & 0 & \lambda \\ 1 & 0 & -(\lambda+\mu) & \lambda \\ 1 & \mu_1 & \mu_2 & -\mu \end{pmatrix}, \ H'_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & -(\lambda+\mu) & 0 & \lambda \\ 1 & 0 & -(\lambda+\mu) & \lambda \\ 1 & \mu_1 & \mu_2 & -\mu \end{pmatrix}.$$

This gives $|H'_0| = -\mu^2(\lambda+\mu) = -r_0$ (indeed, $|H'_x| = -r_x$, holds for all x), and $|H'| = -(\lambda+\mu)(2\lambda^2 + 2\mu\lambda + \mu^2) = -R$. Note that in this example we stick to the old numbering of states $\{0, 1, 1', 2\}$ instead of $\{1, 2, 3, 4\}$.

Although very simple, Cramer's rule does not give us much insight in the form of the stationary probabilities. The following result from Freidlin and Wentzell [7] does.

Consider the directed graph G with nodes the elements of S and (directed) edges $x \to y$, for $x \neq y$, with weight equal to the intensity q_{xy} (this graph is equal to the so called modified

Coates graph of the infinitesimal generator H, see [5], Chapter 3; we will come back to this in the proof of Theorem 3.2).

For fixed $x \in S$, we define the set $\mathcal{T}(x)$ of trees (subgraphs) with root x as follows. A tree $T \in \mathcal{T}(x)$ is a spanning subgraph that satisfies:

- Every node $y \in S$, with $y \neq x$, is the initial node of precisely one directed edge.

- The tree does not contain any circuits.

For $T \in \mathcal{T}(x)$ we define

$$w(T) = \prod_{y \to z \in T} q_{yz}. \tag{5}$$

A variant of the following lemma is contained as Lemma 3.1 in Freidlin and Wentzell [7]. The authors give a proof of this lemma for finite state discrete time Markov chains, but it is equally valid for finite state, continuous time Markov chains. We omit the proof, since only obvious changes are involved.

Lemma 3.1 *Consider an irreducible continuous time Markov chain with finite state space S. The stationary probabilities p_x of this chain are proportional to the numbers:*

$$r_x = \sum_{T \in \mathcal{T}(x)} w(T). \tag{6}$$

Example (truncated coupled processor model, continued) We illustrate this lemma again with our 4-state example. Every set $\mathcal{T}(x)$ contains 4 elements. For example,

$$\mathcal{T}(0) = \{1 \to 2, 2 \to 1', 1' \to 0\} \cup \{1 \to 0, 2 \to 1', 1' \to 0\} \cup$$

$$\{1 \to 0, 2 \to 1, 1' \to 0\} \cup \{1 \to 0, 2 \to 1, 1' \to 2\}.$$

This gives $r_0 = \lambda\mu\mu_2 + \mu^2\mu_2 + \mu^2\mu_1 + \lambda\mu\mu_1 = \mu^2(\lambda + \mu)$, the same expression as r_0 in the introduction. The other terms are similar and also given in the introduction.

Remark (countable state spaces) It is not easy to generalize the lemma of Freidlin and Wentzell to Markov chains with countable (infinite) state space, because the numbers r_x given in (6) then consist of infinite sums of infinite products. However for simple cases, where for each state the number of trees is finite and where the products (5) are uniformly convergent this can be proven. An example of this is a birth-death process with q_{xx-1} such that $0 < \prod_{x=1}^{\infty} q_{xx-1} < \infty$.

Remark (order of stationary probabilities) Lemma 3.1 constitutes an alternative proof of Lemma 1.3 of [10], which states that $p_x = O(\lambda^{L(x)})$ for transitions of the form (1) and L satisfying Assumption 2.1. Indeed, every tree into x contains a path from 0 to x, and therefore its weight has at least coefficient $\lambda^{L(x)}$. On the other hand $r_0 = O(1)$, because there is a tree consisting only of transitions to lower level states and thus $\sum_y r_y = O(1)$. Therefore $p_x = O(\lambda^{L(x)})$.

In the theorem below we give the relation between the two methods. Although the theorem can be proven directly, we prefer a proof using Coates graphs.

Theorem 3.2 $|H'_x| = (-1)^{N+1} r_x$, *for all $x \in S$.*

Proof Let $A = H^t$, where H is the infinitesimal generator of the Markov chain. Then

$$|H'_x| = \Delta_{x,1},$$

where Δ_{ij} are the cofactors of A. The Coates graph of A is an N-node, weighted, labeled, directed graph, such that for $a_{ji} \neq 0$ there is an edge directed from node i to node j, with associated weight a_{ji}, $i, j = 1, 2, \ldots, N$ (cf. [5], p. 142). The *modified* Coates graph of A is obtained from the Coates graph of A by changing the weight associated with each of the self-loops (i, i) to the sum of the weights of the edges having i as terminal node, while keeping all other edge weights unaltered (cf. [5], p. 190). Because H is a generator the column sums of A are all equal to zero so the modified Coates graph is obtained from the Coates graph by deleting all self-loops. Note that the graph G introduced in the beginning of this section is obtained from the modified Coates graph of A by changing the direction of all edges. According to Theorem 3.13 of [5],

$$\Delta_{x,1} = \sum_{R(x)} (-1)^{a_x - 1} w(R(x)),$$

where $R(x)$ is a 1-semifactor (cf. [5], p. 194) of the modified Coates graph of A, with the property that 1 and x belong to the same component of $R(x)$, and where a_x is the number of even components of $R(x)$ (a component is said to be even if it contains an even number of edges). It follows from the definition of a 1-semifactor and the fact that the modified Coates graph of A has no self-loops that the trees $T(x)$ in $\mathcal{T}(x)$ correspond one-to-one with the 1-semifactors $R(x)$ by simply changing the direction of each edge in $T(x)$. Since all trees are connected and the number of edges of a tree is $N - 1$, we have $a_x = 0$, if N is even, and $a_x = 1$, if N is odd. Hence

$$|H'_x| = \Delta_{x,1} = (-1)^{N-1} \sum_{T(x)} w(T(x)) = (-1)^{N+1} r_x.$$

4 Improving the PSA

The relevance of the results of Section 3 to the PSA is as follows. Let all intensities q_{xy} be functions of some variable λ. If all q_{xy} are analytic on a region A, then Lemma 3.1 immediately implies that the numbers r_x, $x \in S$, are analytic on A. More specifically, if all q_{xy} are polynomials (of finite degree), then so are the r_x, and thus the power series expansions of the r_x (which are equal to the polynomials) exist everywhere. Thus ideally an algorithm should find the polynomials r_x. The PSA as introduced in Section 2 does not do this: it finds the coefficients of the rational functions $r_x / \sum_y r_y$, thus normalization

is done as part of the algorithm. This introduces all zeroes of $\sum_y r_y$ as poles of the power series, and thereby limits its convergence. All that can be said about $\sum_y r_y$ is that for finite state chains there are no zeroes on $[0, \infty)$, which shows that the series converge for λ small enough.

Van den Hout [9] gave as example the M/M/1/N queue, the exponential queue with finite waiting room. For this model the steady state probabilities p_x are proportional to $r_x = \lambda^x \mu^{N-x}$ and so norming introduces singularities in the points where: $(\lambda/\mu)^{N+1} - 1 = 0$.

There are two different methods to try to eliminate the zeroes of $\sum_y r_y$ from the solution of the PSA. One is by fixing p_0 beforehand, and one is by normalizing to a term unequal to 1. We consider them one by one.

Fixing p_0 in advance. To eliminate the zeroes of $\sum_y r_y$ from the solution of the PSA we change the algorithm such that it finds the solution to

$$\left\{ \sum_y p'_x q_{xy} = \sum_y p'_y q_{yx}, \ x \neq 0, \ p'_0 = r_0 \right\}. \tag{7}$$

Because $r_x / \sum_y r_y$, $x \in S$ is the unique solution to

$$\left\{ \sum_y p_x q_{xy} = \sum_y p_y q_{yx}, \ x \neq 0, \ \sum_y p_y = 1 \right\}, \tag{8}$$

we see that (7) has as unique solution $p'_x = r_x$ for all $x \in S$. This converges for all λ. Normalizing afterwards gives the stationary probabilities. However, to apply this algorithm, we need to know r_0 beforehand, and finding it is usually as difficult as finding the equilibrium distribution. For some models however, some of which we study later on, we have an explicit expression for p_0, which gives an idea of the form of r_0. (Note that the numerator of p_0 need not be equal to r_0, see the 4-state example.) For more complicated models we suggest taking $p'_0 = 1$. This leads to finding the solution of:

$$\left\{ \sum_y p'_x q_{xy} = \sum_y p'_y q_{yx}, \ x \neq 0, \ p'_0 = 1 \right\}.$$

This can easily be implemented: instead of determining $b_k(0)$ for $k > 0$ by equation (4), we simply take $b_k(0) = 0$ if $k > 0$ (and of course $b_0(0) = 1$). The solution of this system is equal to $p'_x = r_x/r_0$; the corresponding algorithm is denoted as the PSA/N. Thus the PSA/N, compared to the PSA, replaces the singularities which are equal to the zeroes of $\sum_y r_y$ by those coming from r_0. The set of zeroes of r_0 can be contained in the set of zeroes of $\sum_y r_y$. This was the case in the example of the introduction, where r_0 has a zero at $-\mu$, and $\sum_y r_y$ has zeroes $-\mu$ and $\mu(-1 \pm i)/2$. If this is true than the PSA/N performs at least as good as the PSA. In theory it can also be the case that r_0 has zeroes that do not occur in $\sum_y r_y$. This could make the PSA/N perform worse than the PSA, depending on the location of these zeroes. In all our examples the PSA/N performed better than the PSA.

As said, in some cases we know p_0. An example is the $M/M/s$ queue with critical customers (see below), where p_0 is equal to the probability of an empty system in a regular $M/M/s$ queue. Given that it is of the form $p_0 = r'_0/R'$, we can use PSA/N with $p'_0 = r'_0$ instead of $p'_0 = 1$. In general $r'_0 \neq r_0$, thus even in this case there is no guarantee that the PSA/N converges for all λ.

Normalizing to another expression   A second possibility for eliminating the zeroes of $\sum_y r_y$, also leading to the solution $p'_x = r_x$, is solving

$$\Big\{ \sum_y p'_x q_{xy} = \sum_y p'_y q_{yx}, \ x \neq 0, \ \sum_y p'_y = \sum_y r_y \Big\}.$$

Compared to (8) we replaced $\sum_y p_y = 1$ by $\sum_y p'_y = \sum_y r_y$. Again, this would call for knowledge about the solution. In case p_0 is known (as for the $M/M/s$ queue with critical customers), we can try taking $\sum_y p'_y$ equal to its denominator. Another possibility is trying to estimate $\sum_y r_y$ directly from the coefficients of the power series generated by the PSA, possibly using the ε-algorithm (see below). This is a promising research direction, although the first numerical results are not that promising (see $M/M/s$ with critical customers). The correpoding algorithm is called the PSA/D.

Remark (ε-algorithm)   The ε-algorithm (cf. Wynn [11]) is a numerical procedure that converts partial sums $\sum_{k=0}^{m+2r} c_k \lambda^k$ of power series into the quotients of polynomials in λ, say $P_{m+r}(\lambda)/Q_r(\lambda)$, and a remainder term. The subscript is the degree of the polynomials. In cases where the power series has non-essential singularities (poles), the zeroes of $Q_r(\lambda)$ will converge to the poles of the power series and thereby cancel the negative impact of these poles. Koole [10] showed that for finite continuous time Markov chains the PSA together with the ε-algorithm, if applicable, gives exact results. This follows because, according to Cramer's rule or Freidlin and Wentzell's lemma, the stationary probabilities are rational functions of λ, i.e. quotients of polynomials.

For queues that can be modeled as a continuous time Markov chain with infinite state space the following reasoning can be applied. If we have an ergodic, infinite state space Markov chain, we often can (at least in theory) cut off the state space. When the equilibrium equations in the interior of the state space of the cut off chain are equal to the equations of the original model, we only have to check tightness to obtain weak convergence of the stationary probabilities of the cut off model to those of the original model. Assume that this can be done. Consecutively we can apply the PSA together with the ε-algorithm to the finite cut off model. In this way the power-series approach together with the ϵ-algorithm can be justified for simple models like the coupled processor, the shortest queue, and the $M/M/s$ with critical customers and to many other examples.

Some more reflection shows that in practice we do not have to consider these cut off models, since in the interior these models constitute the same equilibrium equations, and hence the same coefficients for the power-series. Hence we might as well apply the PSA and the ϵ-algorithm to the original model immediately, but we insist that it should be checked whether the cut off model indead converges to the original model.

Remark (analyticity and countable state spaces) The example of Aaronson and
Gilat (cf. [8]), modified by van den Hout [9] to a continuous time setting, shows that
there exist models (of course with infinite state space) where the transitions are analytic,
whereas the stationary probabilities are not even continuous. In this model $S = \mathbb{N} \cup \{0\}$,
the transition intensities are given by $q_{n,n-1} = 1$, $n \geq 1$, and $q_{0n} = 2^{-n} + \lambda^2 n^{-(2+\lambda^2)}$. Note
that

$$\sum_{n=1}^{\infty} q_{0n}(\lambda) = 1 + \lambda^2 \sum_{n=1}^{\infty} \frac{1}{n^{\lambda^2+2}} = 1 + \lambda^2 \zeta(2 + \lambda^2),$$

where ζ denotes the Riemann zeta function. The above series has a fat tail, which intro-
duces the problem.

The state 0 has only one tree and all branches of this tree have weight 1, so that $r_0 = 1$.
For $n \geq 1$, the set $\mathcal{T}(n)$ consists of a countable number of trees: $\mathcal{T}(n) = \{T_k(n) : k \geq n\}$,
where $T_k(n)$ has weight $q_{0k}(\lambda)$. Hence

$$r_n = \sum_{k=n}^{\infty} q_{0k}(\lambda), \tag{9}$$

which are analytic functions of λ for $\lambda \in \mathbb{R}$. However normalisation introduces singulari-
ties:

$$\sum_{n=0}^{\infty} r_n(\lambda) = 1 + \sum_{n=1}^{\infty} \sum_{k=n}^{\infty} q_{0k}(\lambda) = 1 + \sum_{k=1}^{\infty} k q_{0k}(\lambda) = 3 + \lambda^2 \zeta(1 + \lambda^2).$$

The function $\sum r_n(\lambda)$ is *not* continuous in the point $\lambda = 0$, because

$$\lim_{\lambda \to 0} \lambda^2 \zeta(1 + \lambda^2) = 1.$$

This example needs further study. Presumably the analyticity of the numbers r_n is not
inheritated by the analyticity of the transition probabilities, because of the large number
of trees that influence the uniformity. Also note that it is not possible in this example to
cut off the state space to a finite set $\{0, 1 \ldots, N\}$, with N big, and keeping the equilibrium
equations on the interior $\{0, 1, \ldots, N-1\}$ the same as in the original model.

Remark (poles in different states) The 4-state example showed that different states
can have stationary probabilities with different poles. However, Example 2.5 on page 36
of [9] is *not* an example of this situation. In this example the author considers the steady
state probabilities p_n, of the queue lengths of a special $H_2/M/1$ queue. He shows that p_n
can have singularities arbitrary close to the origin for each $n \geq 1$, whereas p_0 is analytic.
The reason why this example is not a valid example of a situation where different states
have steady state probabilities with different poles is that the steady state probabilities p_n
are not equal to the equilibrium probabilities π_n of the imbedded Markov chain, at arrival
epochs. So instead of considering the steady state probabilities p_n we should study the
analiticity of the equilibrium probabilities π_n. Consulting [6] we note that

$$\pi_n = (1 - r)r^n, \quad n \geq 0,$$

so that for *all* n the probabilities π_n have the same singularities as the function r (which
in this specific example has two branching points close to the origin).

5 Numerical examples

In this section we report on our numerical experiences, in decreasing order of convergence properties, for the following three models: the coupled processor model, the $M/M/s$ queue with critical customers, and the shortest queue model.

Coupled processor model The coupled processor model consists of a system with two customer classes, with arrival rates ρ_1 and ρ_2, and a single server. If customers of both types are available then the server shares its resources, resulting in departure rates μ_1 and μ_2, but if only one class is available then the full capacity $\mu_1 + \mu_2 = \mu$ is used for this single class. The coupled processor was the first model studied with the PSA (cf. [8]). To apply the PSA we change the arrival rates to $\lambda\rho_1$ and $\lambda\rho_2$. Writing out (3) for this model gives, for $(n, m) \neq (0, 0)$,

$$
\begin{aligned}
\mu b_k(n, m) \;=\; & \rho_1(n > 0)b_k(n - 1, m) + \rho_2(m > 0)b_k(n, m - 1) + \\
& (\mu_1 + \mu_2(m = 0))b_{k-1}(n + 1, m) + \\
& (\mu_2 + \mu_1(n = 0))b_{k-1}(n, m + 1) - (\rho_1 + \rho_2)b_{k-1}(n, m),
\end{aligned}
$$

where the indicator function is simply notated as (A) for some logical expression A.

For this model we know that the total number of customers in the system behaves as an $M/M/1$ queue with arrival rate $\lambda(\rho_1 + \rho_2)$ and departure rate $\mu_1 + \mu_2$. Therefore $p_0 = 1 - \frac{\rho_1 + \rho_2}{\mu_1 + \mu_2}\lambda$, thus it is useless to perform the PSA/N (with $p_0' = 1$) next to the PSA, it even creates numerical instabilities for a load close to or equal to 1. We found that the PSA performs very well, leading to the conclusion that there are no poles in the stability region. In Table 1 there are some numerical results illustrating this. As in Section 2 the number K denotes the highest power of the approximation, and $\mathrm{E}X_i$ denotes the average number of customers in queue i. Note that $\mathrm{E}X_1 + \mathrm{E}X_2$ should be $\frac{\rho_1 + \rho_2}{\mu_1 + \mu_2 - \rho_1 - \rho_2}$, as for the corresponding $M/M/1$ queue. It is remarkable that for a load equal to 1 the system gives $\mathrm{E}X_1 + \mathrm{E}X_2 = K$, as high as is possible with terms of order $\leq K$. For this case all probability mass is indeed concentrated on the states with K customers, and thus for a load equal to 1 the PSA indicates that

$$
\lim_{t \to \infty} \mathrm{P}\{(X_1(t), X_2(t)) = (n, m)\} = 0, \quad (n + m < K)
$$

as it should be.

ρ_1	ρ_2	μ_1	μ_2	K	EX_1	EX_2
1.00	1.00	2.00	2.00	20	0.500	0.500
1.50	1.50	2.00	2.00	20	1.495	1.495
1.50	1.50	2.00	2.00	50	1.500	1.500
1.75	1.75	2.00	2.00	50	3.496	3.496
1.75	1.75	2.00	2.00	80	3.500	3.500
2.00	2.00	2.00	2.00	K	$K/2$	$K/2$
2.00	1.00	1.00	3.00	20	2.528	0.462
2.00	1.00	1.00	3.00	50	2.538	0.462
2.50	1.00	1.00	3.00	50	6.508	0.483
2.50	1.00	1.00	3.00	80	6.517	0.483
3.00	1.00	1.00	3.00	K	$K - 0.500$	0.500

Table 1: Queue sizes in the coupled processor model
for different parameter values.

As said, taking $p_0' = 1$ in the PSA/N might create numerical problems because $p_0 \to 0$ as $\rho_1 + \rho_2 \to \mu_1 + \mu_2$. To solve this problem we can start with $p_0' = \mu_1 + \mu_2 - \lambda(\rho_1 + \rho_2)$. In fact, this choice of p_0' corresponds to $p_0' = (\mu_1 + \mu_2)p_0$, and indeed the performance of the PSA/N corresponds for this choice exactly with the PSA. A starting value which makes $p_0' = 0$ for a critical load, might also work well for other models for which we are interested in the performance of the system under an almost critical load, even if we do not know p_0.

$M/M/s$ **queue with critical customers**   Now we turn to the $M/M/s$ queue with critical customers. This is a system with a single class of customers, but after some random time in the queue a customer becomes *critical*, after which it should be served with priority. The model was introduced in Adan and Hooghiemstra [1]. Suppose that the state is (n, m), i.e., there are n regular (non-critical) and m critical customers. Then the following transitions are possible:
$(n, m) \to (n + 1, m)$ with rate λ, an arrival;
$(n, m) \to (n - 1, m + 1)$ with rate $n\theta$, a regular customer becomes critical;
$(n, m) \to (n, m - 1)$ with rate $\mu_2 \min\{m, s\}$, a departure of a critical customer;
$(n, m) \to (n - 1, m)$ with rate $\mu_1 \min\{n, (s - m)^+\}$, a departure of a regular customer.
Note that, in constrast with [1], we allow the service rate to depend on the state of the customer. From a practical point of view this might be of interest, the customers could for example represent parts needing repair: corrective maintenance (serving the critical customers) often takes longer than preventive maintenance (regular customers).
For this model Equation (3) gives, for $(n, m) \neq (0, 0)$,

$$\left(\theta n + \mu_2 \min\{m, s\} + \mu_1 \min\{n, (s - m)^+\}\right)b_k(n, m) =$$

$$-b_{k-1}(n, m) + (n > 0)b_k(n - 1, m) + \theta(n + 1)b_{k-1}(n + 1, m - 1)$$

$$+\mu_1 \min\{n + 1, (s - m)^+\}b_{k-1}(n + 1, m) + \mu_2 \min\{m + 1, s\}b_{k-1}(n, m + 1).$$

12

Let us first study the case that $\mu_1 = \mu_2 = \mu$. Then the total number of customers forms an $M/M/s$ queue with rates λ and μ. It is well known that the steady state distribution for this simple multi-server queue can be written as a quotient, with for each state the same denominator. Therefore it make sense to try PSA/N. For $s = 3$ we have

$$p_0 = \frac{2\mu(3\mu - \lambda)}{6\mu^2 + 4\lambda\mu + \lambda^2},$$

and this function of λ has (for $\mu = 1$) simple poles at $\lambda = -2 \pm i\sqrt{2}$. Observe that $|-2 \pm i\sqrt{2}| = \sqrt{6} \approx 2.449$. Output can be found in Table 2. In the table we indicated, besides the parameters, the version of the PSA used, with or without normalization, or with normalization to a term $\neq 1$; furthermore we supplied the total expected number of customers computed by the algorithm together with the theoretical value if $\mu_1 = \mu_2$ or an approximation if $\mu_1 \neq \mu_2$. This is indicated by an "f" (formula) or an "a" (approximation) in the last column. If in two consecutive rows the same parameters were used, we left the corresponding entries blank.

s	λ	θ	μ_1	μ_2	K	algorithm	EX_1	EX_2	Total, alg.	Total, f/a
1	0.90	1.00	1.00	1.00	80	PSA	0.832	8.166	8.998	9.000 (f)
		10.00					0.089	8.909	8.998	
3	0.50	1.00	1.00	1.00	20	PSA	0.251	0.252	0.503	0.503 (f)
	1.00						0.519	0.527	1.045	1.045 (f)
	2.00				50		1.271	1.618	2.889	2.889 (f)
	2.50						0.577	2.658	3.235	6.007 (f)
						PSA/N	1.926	4.080	6.006	
						PSA/D	1.926	4.085	6.011	
	2.75				80	PSA	-4135.987	-4462.306	-8598.293	11.988 (f)
						PSA/N	2.392	9.595	11.987	
						PSA/D	2.393	9.515	11.908	
	3.00				80	PSA/N	2.896	37.656	40.552	∞ (f)
						PSA/D	3.000	77.000	80.000	
3	2.00	1.00	2.00	1.00	80	PSA	0.805	0.932	1.737	1.736 (a)
	2.50						1.294	2.359	3.653	3.649 (a)
	2.75						1.805	6.464	8.269	8.197 (a)
		10.00					-47.305	-934.180	-981.485	11.414 (a)
						PSA/N	0.266	11.433	11.699	
						PSA/D	0.703	19.079	19.783	

Table 2: Queue sizes in the $M/M/s$ queue with critical customers for different parameter values.

The first entries of Table 2 indicate that the value of θ has no influence on the convergence, i.e., the poles introduced by the marginal distribution given the total number of customers lie outside the stability region of the Markov chain. Thus the only zeroes are introduced by the normalization factor of the $M/M/s$ queue, and thus the PSA/N (with $p_0' = 1$) should converge in all cases. This is in complience with the other results. Even for $\mu_1 \neq \mu_2$ the PSA/N converged in all cases that we considered.

We also did computations for the variant of the PSA where one normalizes to the denominator of the stationary probabilities of the $M/M/s$ queue. The results are also in Table 2, the algorithm is denoted by PSA/D. The PSA/D performed better than the PSA, and in some cases, especially those where λ is high, also better than the PSA/N. Note that for $\mu_1 = \mu_2$ the PSA/D is equivalent to the PSA/N with $p_0' = 2\mu(3\mu - \lambda)$. For $\mu_1 \neq \mu_2$ the PSA/N with $p_0' = 3\mu_2 - \lambda$ and the PSA/D are not equivalent anymore, PSA/N outperforms both the PSA, the PSA/N with $p_0' = 1$, and the PSA/D, especially for λ large. The choice of p_0' is based on the fact that the system is stable for $\lambda \in [0, s\mu_2)$. Its performance is empressive, for a load equal to 1 it gives $EX_1 + EX_2 = K$, and even for loads > 1 the approximation of p_0 for $\mu_1 = \mu_2$ is equal to its formula value, e.g., -0.053 for $s = 3$, $\lambda = 4$, $\theta = \mu_1 = \mu_2 = 1$ and $K = 80$. In Table 3 we compare the approximations of p_0 for different algorithms. We conclude from these and other experiments (also for other s) that the best algorithm is PSA/N with $p_0' = s\mu - \lambda$.

s	λ	θ	μ_1	μ_2	K	PSA	PSA/N, $p_0' = 1$	PSA/N, $p_0' = s\mu - \lambda$	PSA/D	ε-alg.
3	2.75	1.00	2.00	1.00	20	0.093	0.098	0.087	0.087	0.087
3	2.90	1.00	2.00	1.00	20	0.063	0.071	0.042	0.045	0.042
3	3.00	1.00	2.00	1.00	20	0.044	0.055	0.000	0.008	0.000
3	4.00	1.00	2.00	1.00	20	7.576	0.001	0.156	-4.258	0.156

Table 3: Empty system probabilities in the $M/M/s$ queue with critical customers for different loads and algorithms.

Shortest queue model Finally we study the shortest queue model. This model, which has already been studied extensively by Blanc, is as follows. Customers arrive according to a Poisson process with rate λ. At arrival they are sent to the shortest of two parallel queues, in case of a tie each queue receives the customer with equal probability. The queues operate as single server queue, with rates μ_1 and μ_2.

For this model Equation (3) gives, for $(n, m) \neq (0, 0)$,

$$
\begin{aligned}
&\big((n > 0)\mu_1 + (m > 0)\mu_2\big)b_k(n, m) = \\
&\big((n - 1 < m, n > 0) + \tfrac{1}{2}(n - 1 = m, n > 0)\big)b_k(n - 1, m) \\
&+\big((m - 1 < n, m > 0) + \tfrac{1}{2}(m - 1 = n, m > 0)\big)b_k(n, m - 1) \\
&+\mu_1 b_{k-1}(n + 1, m) + \mu_2 b_{k-1}(n, m + 1) - b_{k-1}(n, m).
\end{aligned}
$$

For this model none of the steady state probabilities is known. We confine us to the total average number of customers. In Table 4 the results of our numerical experiments can be found.

λ	μ_1	μ_2	K	algorithm	Total, alg.	Total, ε
0.50	1.00	1.00	20	PSA	0.551	0.551
1.00	1.00	1.00	20	PSA	1.176	1.426
1.00	1.00	1.00	20	PSA/N	1.426	1.426
1.00	1.00	1.00	20	PSA/D	1.426	1.426
1.50	1.00	1.00	20	PSA	-993.515	3.672
1.50	1.00	1.00	20	PSA/N	3.617	3.672
1.50	1.00	1.00	20	PSA/D	3.442	3.672
1.50	1.00	1.00	50	PSA/N	3.668	3.672
1.50	1.00	1.00	50	PSA/D	5.010	3.672
1.50	1.00	1.00	80	PSA/N	3.672	3.672
1.50	1.00	1.00	80	PSA/D	12.170	3.672
1.75	1.00	1.00	80	PSA/N	8.961	7.822
1.75	1.00	1.00	80	PSA/D	1828913.261	7.822
1.00	1.00	2.00	20	PSA	0.853	0.853
1.50	1.00	2.00	80	PSA	484.366	1.542
1.50	1.00	2.00	80	PSA/N	1.542	1.542
2.00	1.00	2.00	80	PSA/N	0.910	2.775
2.00	1.00	3.00	80	PSA/N	1.717	1.717
3.00	1.00	3.00	80	PSA/N	-0.740	4.296
3.00	1.00	5.00	80	PSA/N	2.049	2.049

Table 4: Queue sizes in the shortest queue model
for different parameter values.

Concerning the PSA/D, we wrote a program to look for poles on the negative reals. Having found one (say $-a$), we eliminated it by multiplying the normalization factor by $(\lambda + a)$. If $\mu_1 \neq \mu_2$, no poles were found, thus we have to assume that they are all complex. As we did not determine their location, the PSA/D did as good as the PSA. In case $\mu_1 = \mu_2 = 1$, a pole was located at -1. After taking $\sum_y p'_y = 1 + \lambda$, convergence improved and we found a further pole at ≈ -1.477766. The effect of removing this one was mixed: we assume that there are other complex poles play a role. The results in the table done with the PSA/D are for $\sum_y p'_y = 1 + \lambda$. The total given directly by the algorithms was compared with the total given by the PSA with the ε-algorithm afterwards.

The PSA/N with $p'_0 = 1$ and the PSA/N with $p'_0 = \mu_1 + \mu_2 - \lambda$ did perform similarly. This was to be expected, as the performance degrades well before the limit of the stability region, indicating that there are other reasons for the bad performance of PSA/N then its behaviour for $\lambda \approx \mu_1 + \mu_2$.

6   Conclusions

In this paper we studied the convergence properties of the PSA. We showed the crucial role of normalization and studied two alternative algorithms, the PSA/D and the PSA/N. The PSA/N with $p'_0 = 1$ performed best in most cases, although $p'_0 = \lambda' - \lambda$ is better if there is some value λ' for which it is known that $p'_0 \to 0$ if $\lambda \to \lambda'$, and if we are interested in values of λ close to λ'. This is illustrated with extensive numerical experiments.

References

[1] I.J.B.F. ADAN, G. HOOGHIEMSTRA, *The M/M/c queue with critical jobs*, To appear in Math. Operat. Res. 47(3), 1998.

[2] H. BAVINCK, G. HOOGHIEMSTRA, E. DE WAARD, *An application of Gegenbauer polynomials in queueing theory*, J. Comp. Appl. Math. 49, pp. 1–10, 1993.

[3] J.P.C. BLANC, *Performance analysis and optimization with the power-series algorithm*, in Performance Evaluation of Computers and Communication Systems, editors: L. Donatiello and R. Nelson, Springer-Verlag, pp. 53-80, 1993.

[4] J.P.C. BLANC, *A numerical study of a coupled processor model*, In Computer Performance and Reliability, editors: G. Iazeolla, P.J. Courtois and O.J. Boxma, North-Holland, pp. 289-303, 1988.

[5] W-K. CHEN, *Applied Graph Theory*, North-Holland, Amsterdam, 1971.

[6] J.W. COHEN, *The Single Server Queue*, North-Holland, Amsterdam, 1982.

[7] M.I. FREIDLIN, A.D. WENTZELL, *Random Peturbations of Dynamical Systems*, Springer-Verlag, New York, 1984.

[8] G. HOOGHIEMSTRA, M.S. KEANE, S. VAN DER REE, *Power series for stationary distributions of coupled processor models*, SIAM J. Appl. Math. 48, pp. 1159–1166, 1988.

[9] W.B. VAN DEN HOUT, *The Power Series Algorithm, A Numerical Approach to Markov Processes*, PH.D. thesis, Tilburg University, 1996.

[10] G.M. KOOLE, *On the use of the power series algorithm for general Markov processes, with an application to a Petri net*, INFORMS J. on Computing 9, pp. 51–56, 1997.

[11] P. WYNN, *On the convergence and stability of the epsilon algorithm*, Siam J. Num. Anal. 3, pp. 91–122, 1996.