A new algorithm for time-dependent first-return probabilities of a fluid queue

N. G. Bean
School of Mathematical Sciences
The University of Adelaide
Adelaide, Australia
nigel.bean@adelaide.edu.au

G. T. Nguyen
School of Mathematical Sciences
The University of Adelaide
Adelaide, Australia
giang.nguyen@adelaide.edu.au

F. Poloni
Dipartimento di Informatica
Università di Pisa
Pisa, Italy
federico.poloni@unipi.it

1. INTRODUCTION

Let $(\mathcal{X}, \varphi) = \{X(t), \varphi(t)\}_{t\geq 0}$ be a fluid queue, where φ is the environment, modelled as a continuous-time Markov chain with state space $\mathcal{S}$ and generator $Q \in \mathbb{R}^{N\times N}$, $N = |\mathcal{S}|$, and $\mathcal{X}$ is the fluid level with $dX(t)/dt = c_{\varphi_t}$ for $t \geq 0$. One key quantity in the steady-state analysis of a fluid queue is its first return matrix Ψ with entries

$$\Psi_{ij} = \mathbb{P}[\tau < \infty,\, \varphi(\tau) = j \in \mathcal{S}_- \mid \varphi(0) = i \in \mathcal{S}_+], \qquad (1)$$

where $\tau = \min\{t > 0 \colon X(\tau) = X(0)\}$ is the first return time, $\mathcal{S}_+ = \{i \in \mathcal{S} : c_i > 0\}$ and $\mathcal{S}_- = \{i \in \mathcal{S} : c_i < 0\}$.

Similarly, for its transient analysis one is interested in computing the time-dependent matrix $\Psi(t)$ with elements

$$[\Psi(t)]_{ij} = \mathbb{P}[\tau < t,\, \varphi(\tau) = j \in \mathcal{S}_- \mid \varphi(0) = i \in \mathcal{S}_+].$$

To compute $\Psi(t)$, one of the most popular methods in the literature is not to evaluate it directly, but to determine its Laplace-Stieltjes transform $\hat{\Psi}(s)$ in the complex plane [2, 7], then relying on algorithms for the inverse transform [1]. This approach works well in practice, but has two drawbacks. First, it requires working with complex arithmetic to compute results that are real positive quantities; second, its results are not accurate to full machine precision, due to intrinsic inaccuracies in these inverse transforms.

A different, direct algorithm not based on Laplace transforms can be obtained with small modifications to [5], where the authors focus on the busy period, $\Psi(t)\mathbf{1}$. Following their technique, one can also obtain an algorithm for $\Psi(t)$. To the best of our knowledge, this algorithm has no probabilistic interpretation; its proof is based on algebraic verification that the resulting function satisfies the Kolmogorov equations.

In this work, we propose another algorithm, which works directly on probability matrices and has a direct physical interpretation. Moreover, it is essentially *subtraction-free*, i.e., it requires only sums and products of positive quantities. Subtraction-free algorithms have already been proposed for various similar tasks, see, e.g., [9, 3, 8]. Their main advantage is that, using the subtraction-free property often one can prove excellent stability properties, obtaining a forward relative error of the order of the machine precision. (Note that the algorithm obtained from [5] is *almost* subtraction-free, but it still performs subtractions between the rates c_i.)

We explain in Section 2 the recurrence on which our algorithm is based, then explore in Section 3 how to truncate the infinite sum appearing in it. In Section 4 we compute the complexity of the algorithm and compare it to that of competing methods. Finally, in Section 5 we present some experiments for comparison.

MAM10 2019, Hobart, Australia

2. THE NEW ALGORITHM

For simplicity, in this short exposition we restrict ourselves to the case in which no rate c_i equals 0, so $\mathcal{S} = \mathcal{S}_+ \cup \mathcal{S}_-$. Suppose that the fluid is uniformized, i.e., φ is replaced by its uniformized discrete-time Markov chain $\varphi^d = \{\varphi^d(t)\}_{t\geq 0}$ with transition matrix $P = I + \lambda^{-1}Q$, for a suitable $\lambda > 0$. Note that the matrix $\Psi(t)$ remains unchanged after the uniformization.

Let $\{t_i\}_{i\in\mathbb{N}}$ be the sequence of Poisson epochs of φ^d, $t_0 = 0$, and set for brevity $X_i = X(t_i)$ and $\varphi^d_i = \varphi^d(t_i)$. Define for $n \in \mathbb{N}^+$ and $t \in \mathbb{R}^{>0}$ the quantity $m_n = \arg\min_{\ell=1,2,\dots,n} X_\ell$, and two matrices $\Psi^+_n(t)$ and $\Psi^-_n(t)$ with entries

$$[\Psi^+_n(t)]_{ij} = \mathbb{P}[X_{m_n} > X(t) > X(0),\, \varphi^d(t) = j \in \mathcal{S}_- \mid \varphi^d(0) = i \in \mathcal{S}_+,\, t_n < t \leq t_{n+1}],$$

$$[\Psi^-_n(t)]_{ij} = \mathbb{P}[X_{m_n} > X(0) > X(t),\, \varphi^d(t) = j \in \mathcal{S}_- \mid \varphi^d(0) = i \in \mathcal{S}_+,\, t_n < t \leq t_{n+1}],$$

respectively, from which it follows that $\Psi^+_0(t) = \Psi^-_0(t) = 0$.

The first non-trivial result is the following.

LEMMA 1. *The matrices $\Psi^\pm_n(t)$ are independent of t.*

PROOF. This follows from a rescaling argument, we shall prove $\Psi^+_n(t) = \Psi^+_n(1)$ for any $t \in \mathbb{R}^{>0}$. The same argument then holds for $\Psi^-_n(t)$.

Consider an arbitrary but fixed time $t > 0$. Conditioned on there being n Poisson events in $[0, t]$, the epochs t_k, $k = 1, \dots, n$, are uniformly distributed in $[0, t]$, which implies that the epochs t_k/t, $k = 1, \dots, n$, are uniformly distributed in $[0, 1]$. Corresponding with each sequence of transition times $\{\widehat{t}_0, \dots, \widehat{t}_n\}$ and an associated sequence of states $\{\widehat{\varphi}^d_0, \dots, \widehat{\varphi}^d_n\}$, is a unique sample path $\widehat{x}_{[0,t]}$ of the fluid level

$\mathcal{X}$ over $[0,t]$. Similarly, corresponding to $\{\widehat{t}_0/t,\dots,\widehat{t}_n/t\}$ and the same state sequence $\{\widehat{\varphi}_0^d,\dots,\widehat{\varphi}_n^d\}$, is a unique sample path $\widehat{y}_{[0,1]}$ of the rescaled process $\{1/tX(s/t)\}_{s\in\mathbb{R}\geq 0}$ on $[0,1]$.

As the distribution of the states is determined by P alone (and not by t), the probability density associated with $\widehat{x}_{[0,t]}$ is the same as the probability density associated with the rescaled sample path $\widehat{y}_{[0,1]}$. Thus,

$$\begin{aligned}[\Psi_n^+(t)]_{ij} &= \mathbb{P}[X_{m_n} > X(t) > X(0),\, \varphi^d(t) = j \in \mathcal{S}_- \mid \\ &\qquad \varphi^d(0) = i \in \mathcal{S}_+,\, t_n < t < t_{n+1}] \\ &= \mathbb{P}[X(t_{m_n}/t) > X(1) > X(0),\, \varphi^d(1) = j \in \mathcal{S}_- \mid \\ &\qquad \varphi^d(0) = i \in \mathcal{S}_+,\, t_n/t < 1 < t_{n+1}/t] \\ &= \mathbb{P}[X(t^*_{m^*_n}) > X(1) > X(0),\, \varphi^d(1) = j \in \mathcal{S}_- \mid \\ &\qquad \varphi^d(0) = i \in \mathcal{S}_+,\, t^*_n < 1 < t^*_{n+1}],\end{aligned}$$

where $\{t_i^*\}_{i\in\mathbb{N}}$ is a sequence of epochs of a Poisson process of the same rate λ and $m_n^* = \arg\min_{\ell=1,2,\dots,n} X(t_\ell^*)$. The last equality follows from the fact that conditioned on there being n events in $[0,1]$, $t_i^*, i = 1,\dots,n$ are uniformly distributed on $[0,1]$. □

Next, we show that $\Psi(t)$ can be computed using $\Psi_n^-(t)$ alone. To that end, observe that conditioned on there being n Poisson events in $[0,t]$ and $\varphi(0) \in \mathcal{S}_+$, there are four sets of exhaustive and mutually exclusive sample paths, those with

1. $X_{m_n} > X(t) > X(0)$, which contribute to $\Psi_n^+(t)$ — these have not returned to level $X(0)$ by time t and therefore do not contribute to $\Psi(t)$;
2. $X_{m_n} > X(0) > X(t)$, which contribute to $\Psi_n^-(t)$ — these have returned to level $X(0)$ at some time τ such that $t_n < \tau < t$, and contribute to $\Psi(t)$;
3. $X(t) > X_{m_n} > X(0)$ — these have not returned to $X(0)$ by time t and thus do not contribute to $\Psi(t)$;
4. $X(0) > X_{m_n}$, these have returned to $X(0)$ at some time τ such that $\tau < t_{m_n}$ — they contribute to $\Psi(t)$.

(For the first two sets, it is not possible that $\varphi^d(t) \in \mathcal{S}_+$.) Note also that in the second set there are n Poisson events in $(0,\tau)$, and in the fourth set there are k, for $k = 1,\dots,n-1$, events in $(0,\tau)$. By Lemma 1, we abbreviate $\Psi_n^\pm(t)$ to $\Psi_n^\pm$, and have the following.

Lemma 2. *We have*

$$\Psi(t) = \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=1}^{n}\Psi_k^-. \tag{2}$$

Proof. Conditioning on the number of events n in $[0,t]$ and on the index k of the last event before τ, we have

$$\begin{aligned}[\Psi(t)]_{ij} &= \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=1}^{n}\mathbb{P}[t_k < \tau < t_{k+1},\, \varphi^d(\tau) = j \in \mathcal{S}_- \mid \\ &\qquad \varphi^d(0) = i \in \mathcal{S}_+, t_n < t < t_{n+1}] \\ &= \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\left(\Psi_n^-(t) + \sum_{k=1}^{n-1}\Psi_k^-(t_{k+1})\right) \\ &= \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=1}^{n}\Psi_k^-(t),\end{aligned}$$

where the last equality follows from Lemma 1. □

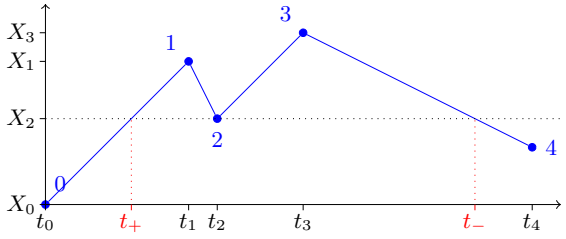

Figure 1: A sample path in $\Psi_3^+(t_4)$ with $m_3 = 2$.

Lemma 3. *Setting $\Psi_n = \Psi_n^+ + \Psi_n^-$, we have*

$$[\Psi_n^+]_{ij} = \frac{c_i}{c_i + |c_j|}[\Psi_n]_{ij}, \quad [\Psi_n^-]_{ij} = \frac{|c_j|}{c_i + |c_j|}[\Psi_n]_{ij}. \tag{3}$$

Proof. The sample paths counted in $[\Psi_n(t_{n+1})]_{ij} = [\Psi_n]_{ij}$ are those in which $X_{m_n} > \max(X_0, X_{n+1})$, and, in addition, the fluid level increases with rate $c_i > 0$ in (t_0, t_1) and decreases with rate $c_j < 0$ in (t_n, t_{n+1}). We shall show, that under these conditions the probability that $X_0 > X_{n+1}$ is $|c_j|/(c_i + |c_j|)$; this implies the thesis.

A sample path counted in $[\Psi_n(t_{n+1})]_{ij}$ is determined uniquely by (i) the sequence of states $\{\varphi_0^d,\dots,\varphi_n^d\}$, and (ii) the i.i.d. increments $t_1 - t_0, t_2 - t_1, \dots, t_{n+1} - t_n$, independent from the states. If we fix the values of a sequence of states, and of the increments $t_2 - t_1, \dots, t_n - t_{n-1}$ (*excluding* the first and last), this determines uniquely the shape of the sample path, apart from the lengths of the first and last legs. In particular, the times $t_+ \in (0, t_1]$ and $t_- \in [t_n, t_{n+1})$ at which $X(t_+) = X(t_-) = X_{m_n}$ are uniquely determined by this choice, and they are independent from $t_1 - t_0$ and $t_{n+1} - t_n$. (See Figure 1 for an example that shows how the various quantities are defined.)

Recall that we have conditioned on $X_{m_n} > X_0$, i.e., the length of $t_1 - t_0 \sim \mathrm{Exp}(\lambda)$ exceeds $t_1 - t_+$; by the memoryless property, $t_+ - t_0$ is also $\mathrm{Exp}(\lambda)$. The corresponding difference in level is $X_{m_n} - X_0 = X(t_+) - X_0 = c_i(t_+ - t_0) \sim \mathrm{Exp}(\lambda/c_i)$. Analogously, $X_{m_n} - X_{n+1} = X(t_-) - X_{n+1} = |c_j|(t_{n+1} - t_-) \sim \mathrm{Exp}(\lambda/|c_j|)$, which is independent from $X_{m_n} - X_0$. The probability that $X_0 > X_{n+1}$ equals the probability that $X_{m_n} - X_0 < X_{m_n} - X_{n+1}$, which is thus $(\lambda/c_i)/(\lambda/c_i + \lambda/|c_j|) = |c_j|/(c_i + |c_j|)$. □

Our main result is the following recursion, which, together with (3), allows one to determine $\Psi_n^\pm$.

Theorem 4. *The following recurrence holds.*

$$\Psi_1 = P_{+-}, \quad \textit{and, for all } n > 1,$$

$$\Psi_n = P_{++}\Psi_{n-1}^- + \sum_{m=2}^{n-1}\Psi_{m-1}^+ P_{-+}\Psi_{n-m}^- + \Psi_{n-1}^+ P_{--}.$$

Proof. We consider the signs of the rate of the fluid before and after the event at time t_{m_n}.

At time t_{m_n}, if the rate transitions from $\mathcal{S}_+$ to $\mathcal{S}_+$, there cannot be other transitions before t_{m_n}, because otherwise $X_{m_n-1} < X_{m_n}$, which would contradict the choice of m_n; hence $m_n = 1$. Furthermore, the interval (t_1, t) contains $n-1$ Poisson epochs, and $X(t) < X_1 < \min(X_2,\dots,X_{m_n})$; but, up to a relabelling of the epochs, this is exactly the definition of Ψ_{n-1}^-. So the probability of this case is $P_{++}\Psi_{n-1}^-$.

Symmetrically, if the rate goes from $\mathcal{S}_-$ to $\mathcal{S}_-$, then it must be the last, $m_n = n$, otherwise $X_{m_n+1} < X_{m_n}$, and it comes after a path in Ψ^+_{n-1}; this produces the term $\Psi^+_{n-1}P_{--}$.

In the case the rate goes from from $\mathcal{S}_+$ to $\mathcal{S}_-$, then there cannot be events before nor after t_{m_n}. Hence it must be the case that $m_n = n = 1$. So in Ψ_1 only we have a term P_{+-}, corresponding to the probability of this single transition.

Finally, if the rate changes from $\mathcal{S}_-$ to $\mathcal{S}_+$, then there must be at least one transition before and after it, so $2 \le m_n \le n-1$. In $(0, t_{m_n})$ we have $m_n - 1$ events and observe a path in $\Psi^+_{m_n-1}$; in (t_{m_n}, t) we have $n - m_n$ events and observe a path in $\Psi^-_{n-m_n}$. This produces the summands $\Psi^+_{m_n-1}P_{-+}\Psi^-_{n-m_n}$, for each possible value of m_n. □

3. FINITE APPROXIMATION

In a numerical algorithm, we need to truncate or approximate the sum (2), which has an infinite number of terms. We propose two different procedures to do this.

Algorithm A1. Note that $B_n = \sum_{k=1}^n \Psi_k^-$ is a non-decreasing sequence, and $\lim_{n\to\infty} B_n = \Psi(\infty) = \Psi$. Generalizing the approach in [5], whenever n is sufficiently large we can replace B_n with Ψ, which may be computed directly with various algorithms (see, e.g., [6]). More specifically, if n' is the smallest integer such that $(\Psi - B_{n'}) \sum_{n>n'} e^{-\lambda t} \frac{(\lambda t)^n}{n!} < \varepsilon$, then we can approximate $\Psi(t)$ with

$$\Psi'(t) = \sum_{n=0}^{n'} e^{-\lambda t} \frac{(\lambda t)^n}{n!} B_n + \Psi \sum_{n=n'+1}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!}, \qquad (4)$$

with error bounded by

$$\begin{aligned}\Psi'(t) - \Psi(t) &= \sum_{n=n'+1}^{\infty} (\Psi - B_n) e^{-\lambda t} \frac{(\lambda t)^n}{n!} \\ &\le \sum_{n=n'+1}^{\infty} (\Psi - B_{n'}) e^{-\lambda t} \frac{(\lambda t)^n}{n!} < \varepsilon.\end{aligned}$$

Algorithm A2. Another truncation strategy consists in swapping the order of summation in (2), obtaining

$$\Psi(t) = \sum_{k=1}^{\infty} \Psi_k^- \sum_{n=0}^{k-1} e^{-\lambda t} \frac{(\lambda t)^n}{n!}.$$

This sum can be truncated when Ψ_k is sufficiently small. If we stop the computation of the coefficients at the same index n' as above, we get

$$\Psi''(t) = \sum_{k=1}^{n'} \Psi_k^- \sum_{n=0}^{k-1} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \le \Psi(t). \qquad (5)$$

Note that $\Psi''(t) \le \Psi(t) \le \Psi'(t)$, so this method produces an explicit inclusion interval for each entry of $\Psi(t)$.

The quantities $\sum_{n=0}^{k-1} e^{-\lambda t} \frac{(\lambda t)^n}{n!}$ and $\sum_{n=n'+1}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!}$ are related to the upper and lower incomplete Gamma functions by classical identities [4, Eqn. (8.69)], and they can be computed exactly with the routines included in the mathematical libraries of many languages.

4. COMPLEXITY ANALYSIS

We compare the complexities of various algorithms to compute $\Psi(t)$. The primitives required by all of them are linear algebra operations between matrices whose dimensions are either $|\mathcal{S}_+|$ or $|\mathcal{S}_-|$. The most general way to bound the cost of each of these operations is $O(N^3)$ floating point operations (flops). We shall use this expression in all the costs in the following.

Algorithms A1, A2: computing the recursion in Thm. 4, and then using (4) or (5), respectively, which give approximations from above and from below. They require $O(N^3(n')^2)$ flops: we compute n' steps of the recurrence in Thm. 4, and each step requires $O(n')$ linear algebra operations. The value of n' required to obtain a prescribed accuracy varies with t. We show in Section 5 (and, especially, in Fig. 2) how the two values are related in an example. We often consider A1 and A2 together because, once one computes the recursion in Thm. 4, both estimates can be obtained with a minimal time overhead.

Algorithm BST: the algorithm obtained by modifying the approach suggested in [5] (named after the initials of its authors). It is based on filling certain triangular arrays, each element of which is a matrix of size $|\mathcal{S}| \times |\mathcal{S}_-|$ in our modified version. There are r such arrays, where $1 \le r \le |\mathcal{S}_-|$ is the number of *distinct* negative rates, and each of them has size $O(n')^2$. Filling in each element requires $O(1)$ linear algebra operations, hence the total complexity of this algorithm is $O(N^3 r (n')^2)$ flops, which is a factor of r more than A1 or A2.

Algorithm LST: the algorithm based on inverse Laplace-Stieltjes transforms (LSTs). Our tests used the Euler algorithm in [1] for inversion, which requires evaluating the transform $\hat{\Psi}(s)$ in several nodes. The number of nodes recommended in [1] is $2d+1$, where d is the number of (decimal) digits of precision required. With ε equal to the machine precision u, $d = 16$. Each evaluation was carried out with a number of steps h of Newton's method [7, Alg. 4]; each step requires $O(1)$ linear algebra operations. This gives a total cost of $O(N^3 dh)$ flops. Note, h is related to the drift δ of the fluid by $\varepsilon = O(\delta^{2^h})$, due to quadratic convergence results for Newton's methods.

Several points. There is another important factor that works against LST. Often, one needs to evaluate $\Psi(t)$ for m different values of t. In A1/A2, the most expensive part is computing the recursion; but it is sufficient to do it once, with a truncation criterion n' determined by the largest of the values t. Thus, the total cost to compute $\Psi(t_1), \ldots, \Psi(t_m)$ is $O(N^3(n')^2 + N^2 n' m)$ flops for A1 and A2, where the second terms comes from applying (4) and/or (5) m times. For BST, a similar analysis holds; the complexity is $O(N^3 r (n')^2 + N^2 n' m)$ flops.

In contrast, we do not know of an established numerical scheme in the literature that allows one to compute inverse Laplace transforms at multiple points, so our best estimate for LST is m times the cost for one point, i.e., $O(N^3 dhm)$ flops. This fact makes A1 and A2 a very convenient choice when multiple evaluations are required.

A further remark is that a more careful analysis of the N^3 term for the linear algebra cost would show that A1, A2, BST scale better than LST when the matrix Q is sparse, or when one among $|\mathcal{S}_+|$ and $|\mathcal{S}_-|$ is much smaller than the other.

5. NUMERICAL EXPERIMENTS

We ran some tests to compare the various numerical algorithms. Where not specified otherwise, the truncation

Algorithm	$t = 0.1$	$t = 1.1$	$t = 9.9$
BST	6.6×10^{-16}	5.9×10^{-16}	1.2×10^{-15}
LST	6.1×10^{-11}	4.2×10^{-11}	4.1×10^{-11}
A1	6.5×10^{-16}	1.9×10^{-16}	3.8×10^{-16}
A2	2.8×10^{-16}	2.4×10^{-16}	8.4×10^{-16}

Table 1: Relative errors obtained with the various algorithms in a simple example.

Algorithm	$t = 0.1$	$t = 1.1$	$t = 9.9$	$t = 15$	$t = 0{:}15$
BST	0.17	0.39	3.54	6.11	6.11
LST	0.06	0.09	0.07	0.12	5.49
A1 & A2	0.08	0.09	0.26	0.64	0.64
n'	17	47	182	243	243

Table 2: CPU times (seconds) obtained with the various algorithms.

threshold ε in A1, A2, and BST is equal to the machine precision $\mathtt{u} \approx 2 \cdot 10^{-16}$.

Relative errors. We generated a toy model with $|\mathcal{S}_+| = 2, |\mathcal{S}_-| = 3$ with the MATLAB instructions

```
rng('default');
T = rand(N); T = T - diag(T*ones(N,1));
C = blkdiag(diag(0.2 * rand(Nplus,1)), ...
                -diag(rand(Nminus,1)));
```

which produced a model with a negative drift of -0.096.

We evaluated for each computed value $\tilde{X}$ the relative forward error $\|\tilde{X} - X\|_\infty / \|X\|_\infty$. Here X is an 'exact' reference value obtained by running both LST and A1 with higher working precision, as well as a higher number of nodes for LST, and checking that their results coincide up to the machine precision $\mathtt{u}$.

Table 1 confirms that LST (with normal working precision and the default number of nodes $d = 16$) can obtain only an accuracy of the order of 10^{-11}. These results are consistent with the theory in [1, Sect. 7], which predicts an accuracy (for well-behaved functions) of $0.6d$ decimal digits when working with $2d + 1$ nodes and d decimal digits of precision. Further tests show that this is the typical behaviour, at least for small values of $|\mathcal{S}|$, although computing these reference values is extremely slow.

Running times. We generated a more challenging test, with the same code but $|\mathcal{S}_+| = 10, |\mathcal{S}_-| = 11$. Table 2 shows the CPU times obtained for several values of t and, in the last column, for when 100 values equispaced in $[0, 15]$ are computed simultaneously. The results show that LST requires a constant time to compute each time sample, irrespective of t, while the other algorithms get slower as t increases. On the other hand, these algorithms can compute the 100 time samples of $\Psi(t)$ basically with no overhead w.r.t. the cost of computing the one with the largest t. This confirms the theory in Section 4.

To see how the time to compute the recursion in Thm. 4 varies with t and with the accuracy required, we plot in Fig. 2 these times, for various values of ε. The reason for the plateau observed for $\varepsilon = 10^{-5}$ is, at $t \approx 11$ the first return has already been observed w.p. $1 - 10^{-5}$, so $\Psi(t)$ coincides with $\Psi = \Psi(\infty)$ at that level of accuracy. Fig. 3 shows the complementary CDF of the first return time observed when starting from the state with the largest positive rate. Observing the two plots, one sees that the values of t that require a high CPU time correspond to the 'tail' of the distribution of $\Psi(t)$, and are needed only for a fine analysis at high levels of accuracy.

Note that these algorithms do not require a negative drift to work; the main effect of the drift is that $\Psi(t)$ and Ψ_n^- decay more slowly with t and n respectively, thus increasing the computational time.

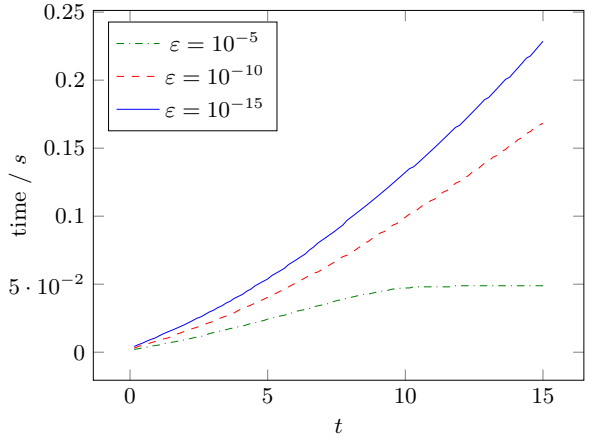

Figure 2: CPU times required by the new algorithm to compute the quantities Ψ_n to various required accuracies for $\Psi(t)$.

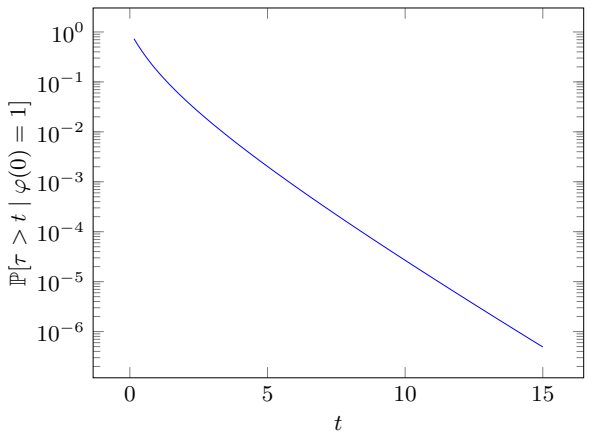

Figure 3: Complementary CDF (survival function) of the first return time starting from the state with the largest positive rate

6. CONCLUSIONS

We introduced a subtraction-free algorithm to compute the time-dependent first return probabilities of a fluid queue, $\Psi(t)$. Experimentally, this algorithm is faster than that obtained by generalizing the results in [5], and also faster and more accurate than the one based on the inverse LST, when one computes several values of $\Psi(t)$ simultaneously. Its main drawback is that the CPU time required increases when one is interested in computing very accurately the values of $\Psi(t)$ within the very tail of the distribution.

7. REFERENCES

[1] J. Abate and W. Whitt. A unified framework for numerically inverting Laplace transforms. *INFORMS J. Comput.*, 18(4):408–421, 2006.

[2] S. Ahn and V. Ramaswami. Efficient algorithms for transient analysis of stochastic fluid flow models. *J. Appl. Probab.*, 42(2):531–549, 2005.

[3] A. S. Alfa, J. Xue, and Q. Ye. Accurate computation of the smallest eigenvalue of a diagonally dominant M-matrix. *Math. Comp.*, 71(237):217–236, 2002.

[4] G. B. Arfken and H. J. Weber. *Mathematical methods for physicists*. Elsevier Academic press, sixth international edition, 2005.

[5] N. Barbot, B. Sericola, and M. Telek. Distribution of busy period in stochastic fluid models. *Stoch. Models*, 17(4):407–427, 2001.

[6] N. G. Bean, M. M. O'Reilly, and P. G. Taylor. Algorithms for return probabilities for stochastic fluid flows. *Stoch. Models*, 21(1):149–184, 2005.

[7] N. G. Bean, M. M. O'Reilly, and P. G. Taylor. Algorithms for the Laplace-Stieltjes transforms of first return times for stochastic fluid flows. *Methodol. Comput. Appl. Probab.*, 10(3):381–408, 2008.

[8] G. T. Nguyen and F. Poloni. Componentwise accurate fluid queue computations using doubling algorithms. *Numer. Math.*, 130(4):763–792, 2015.

[9] C. A. O'Cinneide. Entrywise perturbation theory and error analysis for Markov chains. *Numer. Math.*, 65(1):109–120, 1993.