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

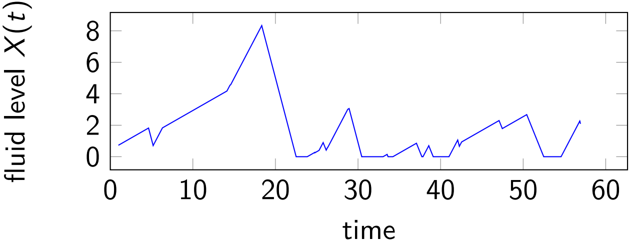
Federico Poloni
Joint work with N. Bean, G. T. Nguyen

University of Pisa

Matrix Analytic Methods in Stochastic Models (MAM) 10, 2019, Hobart, Australia

Setup

Fluid queue: "infinite-size bucket" in which fluid level changes at a rate $c_{\varphi(t)}$ which depends on the state $\varphi(t)$ of a CTMC with generator matrix Q.
Arbitrary rates c_i; no zero rates for simplicity (in this talk); $S = S_+ \cup S_-$.



τ = time of first return to starting level.

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

Algorithms for $\Psi(t)$

Laplace-Stieltjes transform [Ahn, Ramaswami '04] [Bean, O'Reilly, Taylor '08] [Abate, Whitt '95, '06, ...]

- Needs complex arithmetic.
- Known to 'lose' significant digits.

Triangular arrays [Sericola, '98], [Barbot, Sericola, Telek '01] (variant)

- Full transient analysis without complex transforms.
- Almost cancellation-free; nonnegative matrices, but no probabilistic interpretation (as far as I know).
- Expensive: need to fill certain 'triangular arrays' of $|S| \times |S_-|$ matrices, one for each negative rate.

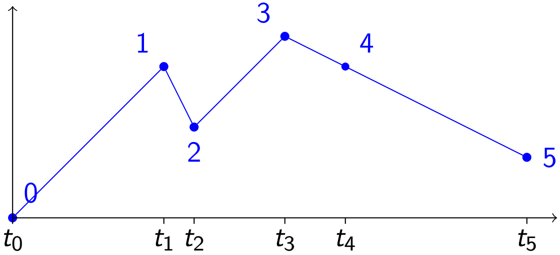
New algorithm in this talk

- No complex transforms
- Nonnegative matrices, cancellation-free: ensures forward stability.
- Comes with probabilistic interpretation (lowest-trough).

Uniformization

Uniformization: replace the CTMC with:

- State transitions according to DTMC with $P = I + \frac{1}{\lambda} Q$.
- Events with independent time increments $t_{k+1} - t_k \sim \mathrm{Exp}(\lambda)$.

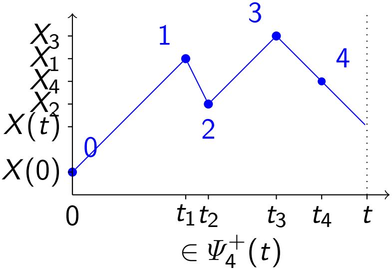


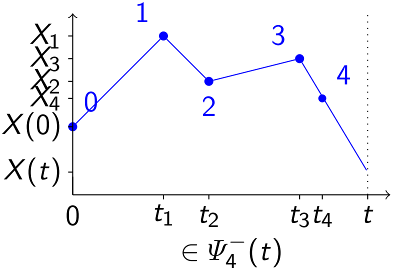
The matrices Ψ_n^+ and Ψ_n^-

Definition

$$[\Psi_n^+(t)]_{ij} = P[X(0) < X(t) < \text{all } X(t_k),\ \varphi(t) = j \in S_- \mid \varphi(0) = i \in S_+,\ n \text{ events in } (0,t)]$$

$$[\Psi_n^-(t)]_{ij} = P[X(t) < X(0) < \text{all } X(t_k),\ \varphi(t) = j \in S_- \mid \varphi(0) = i \in S_+,\ n \text{ events in } (0,t)]$$

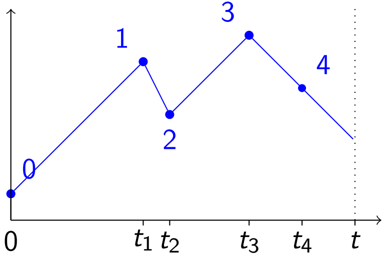


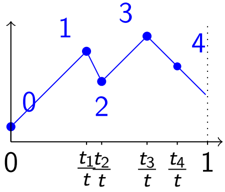


Time rescaling

Lemma

$\Psi_n^{\pm}(t)$ are the same for each t.





Proof Prob. density of observing events at $\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_n$, conditioned on n events in $(0, t)$ = prob. density of observing events at $\frac{\hat{t}_1}{t}, \ldots, \frac{\hat{t}_n}{t}$, conditioned on n events in $(0, 1)$.

Paths with these events (and the same states) are equal up to rescaling.

Meaning of Ψ_n^-

$$\Psi_n^-(t) = P[\tau \in (t_n, t)].$$

Hence

$$\begin{aligned}\Psi(t) &= P[\tau < t] \\ &= \sum_{n=0}^{\infty} P[n \text{ evts in } (0,t)] P[\tau \in (t_1,t_2),(t_2,t_3),\dots,(t_{n-1},t_n) \text{ or } (t_n,t)] \\ &= \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}(\Psi_1^- + \Psi_2^- + \cdots + \Psi_n^-).\end{aligned}$$

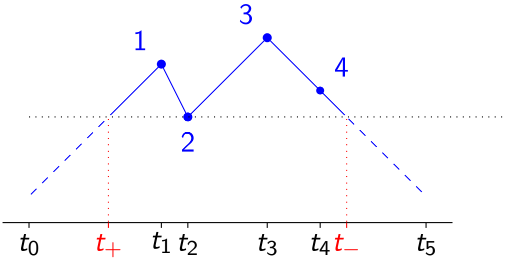
Remark The triangular arrays method also computes $\Psi_1^- + \Psi_2^- + \cdots + \Psi_n^-$ and this sum.

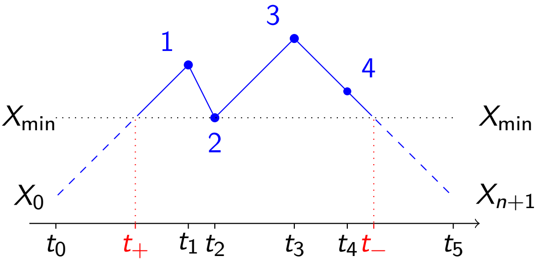
Proportion of paths in Ψ_n^+ vs. Ψ_n^-

Lemma

Let $\Psi_n = \Psi_n^+ + \Psi_n^-$. Then, $[\Psi_n^+]_{ij} = \frac{c_i}{c_i + |c_j|}[\Psi_n]_{ij}$, $[\Psi_n^-]_{ij} = \frac{|c_j|}{c_i + |c_j|}[\Psi_n]_{ij}$

Proof Assume final time $t = t_{n+1}$. Assume we have a path counted in either $[\Psi_n^+]_{ij}$ or $[\Psi_n^-]_{ij}$, but we don't know which one; fix everything apart from lengths of first and last time increments:

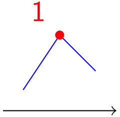




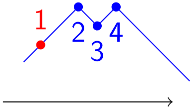
- $t_1 - t_0 \sim \mathrm{Exp}(\lambda)$
- $t_+ - t_0 \sim \mathrm{Exp}(\lambda)$ (memoryless property).
- Similarly, $t_{n+1} - t_- \sim \mathrm{Exp}(\lambda)$.
- Now, let's focus on vertical lengths: $X_{\min} - X_0 \sim \mathrm{Exp}(\frac{\lambda}{c_i})$, $X_{\min} - X_{n+1} \sim \mathrm{Exp}(\frac{\lambda}{|c_j|})$.
- $P[\text{path in } \Psi_n^+] = P[X_{\min} - X_0 \text{ longer than } X_{\min} - X_{n+1}] = \frac{\frac{\lambda}{|c_j|}}{\frac{\lambda}{c_i} + \frac{\lambda}{|c_j|}} = \frac{c_i}{c_i + |c_j|}$ (Poisson race).

Lowest-trough recursion for Ψ_n

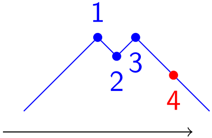
Condition on the type and position on the lowest event.

- $S_+ \to S_-$ 

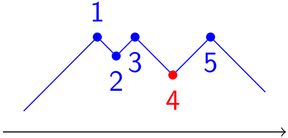
Must be only event ($n = 1$), P_{+-}.

- $S_+ \to S_+$ 

Must be first event, $P_{++}\Psi^-_{n-1}$.

- $S_- \to S_-$ 

Must be last event, $\Psi^-_{n-1}P_{--}$.

- $S_- \to S_+$ 

m-th event, $\Psi^+_{m-1}P_{-+}\Psi^-_{n-m}$.

The recursion

$$\Psi_1 = P_{+-},$$

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

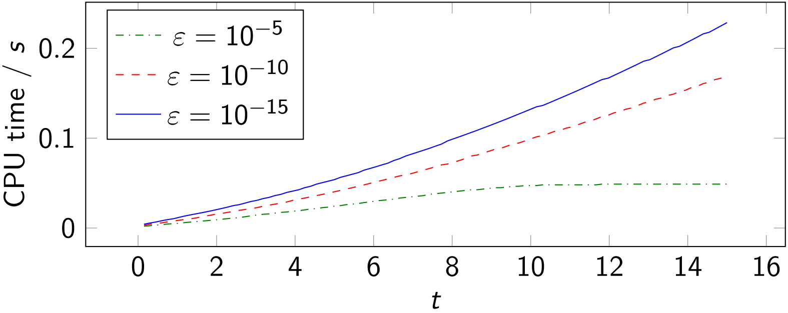
With $[\Psi_n^+]_{ij} = \frac{c_i}{c_i + |c_j|}[\Psi_n]_{ij}$, $[\Psi_n^-]_{ij} = \frac{|c_j|}{c_i + |c_j|}[\Psi_n]_{ij}$, this allows one to compute all $\Psi_n^{\pm}$.

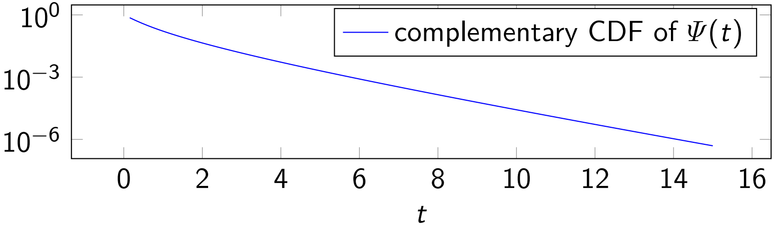
Algorithm

1. Compute enough terms of this recursion.
2. Truncate sum $\Psi(t) = \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} (\Psi_1^- + \Psi_2^- + \cdots + \Psi_n^-)$.

Computational aspects

The bad: need more terms as t gets higher: slower to compute 'tails'.





Computational aspects

The good: speed Faster than triangular arrays. With the same $\Psi_n^{\pm}$, one can compute multiple points with negligible overhead.

Algorithm	$t = 0.1$	$t = 1.1$	$t = 9.9$	$t = 15$	100 t in $[0, 15]$
Laplace-Stiltjes	**0.06** s	**0.09** s	**0.07** s	**0.12** s	5.49 s
Triangular arrays	0.17 s	0.39 s	3.54 s	6.11 s	6.11 s
This algorithm	0.08 s	**0.09** s	0.26 s	0.64 s	**0.64** s

The good: accuracy Relative errors are much smaller than with Laplace-Stiltjes transforms; reaches full machine precision.

Algorithm	$t = 0.1$	$t = 1.1$	$t = 9.9$
Laplace-Stiltjes	6.1×10^{-11}	4.2×10^{-11}	4.1×10^{-11}
Triangular arrays	6.6×10^{-16}	5.9×10^{-16}	1.2×10^{-15}
This algorithm (trunc. above)	6.5×10^{-16}	$\mathbf{1.9 \times 10^{-16}}$	$\mathbf{3.8 \times 10^{-16}}$
This algorithm (trunc. below)	$\mathbf{2.8 \times 10^{-16}}$	2.4×10^{-16}	8.4×10^{-16}

The generating function Riccati equation

The recursion

$$\Psi_1 = P_{+-},$$

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

With $[\Psi^+_n]_{ij} = \frac{c_i}{c_i + |c_j|}[\Psi_n]_{ij}$, $[\Psi^-_n]_{ij} = \frac{|c_j|}{c_i+|c_j|}[\Psi_n]_{ij}$, this allows one to compute all $\Psi^\pm_n$.

Nice fact: the generating function $\widehat{\Psi}(z) = \sum_{i=1}^{\infty} \Psi^-_n z^n$ satisfies

$$C_+^{-1}(P_{++} - z^{-1}I)\widehat{\Psi}(z) + \widehat{\Psi}(z)|C_-^{-1}|(P_{--} - z^{-1}I) + \widehat{\Psi}(z)|C_-^{-1}|P_{-+}\widehat{\Psi}(z) + C_+^{-1}P_{+-} = 0$$

which is a close relative of the Riccati equation solved in the Laplace-Stiltjes method.

Conclusions

What we did

- New lowest-trough algorithm to compute $\Psi(t)$ directly, without complex transforms.
- More accurate, and faster if one needs more than ≈ 5 points.
- Full probabilistic interpretation.

What we still don't know

- Quantitative convergence / complexity bounds on the tails?
- Other algorithms than lowest-trough? Not clear; tricky to put each sample path in the correct 'bin' Ψ_n^-.
- Fast convolution algorithms to speed up the sums? Or are these the same complex transforms that we wanted to avoid?

Conclusions

What we did

- New lowest-trough algorithm to compute $\Psi(t)$ directly, without complex transforms.
- More accurate, and faster if one needs more than ≈ 5 points.
- Full probabilistic interpretation.

What we still don't know

- Quantitative convergence / complexity bounds on the tails?
- Other algorithms than lowest-trough? Not clear; tricky to put each sample path in the correct 'bin' Ψ_n^-.
- Fast convolution algorithms to speed up the sums? Or are these the same complex transforms that we wanted to avoid?

Thanks for your attention!