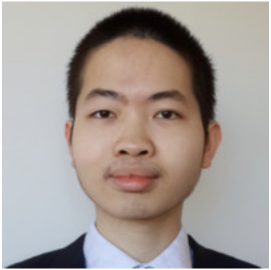
On backward smoothing algorithms

Nicolas Chopin

ENSAE, Institut Polytechnique de Paris

June 5, 2023

Joint work with



Hai-Dang Dau, Oxford University

Outline

Rejection sampling

To sample from density p, with proposal q such that $p(x) \leq Cq(x)$:

Rejection sampling

Repeat:

- Sample $X \sim q$
- Sample $U \sim \mathcal{U}[0, 1]$

until $U \leq p(X)/Cq(X)$.

Rejection sampling

To sample from density p, with proposal q such that $p(x) \leq Cq(x)$:

Rejection sampling

Repeat:

- Sample $X \sim q$
- Sample $U \sim \mathcal{U}[0,1]$

until $U \leq p(X)/Cq(X)$.

The running time of this algorithm is **random**. It follows a Geometric distribution with parameter $1/C$.

Random execution time

Is this a good thing, or a bad thing?

Random execution time

Is this a good thing, or a bad thing?
Suppose you need to run N times an algorithm with random execution time. Then:

Random execution time

Is this a good thing, or a bad thing?
Suppose you need to run N times an algorithm with random execution time. Then:

- non-parallel implementation: total running time $=$ sum

Random execution time

Is this a good thing, or a bad thing?
Suppose you need to run N times an algorithm with random execution time. Then:

- non-parallel implementation: total running time = sum
- parallel implementation: total running time = max

Random execution time

Is this a good thing, or a bad thing?
Suppose you need to run N times an algorithm with random execution time. Then:

- non-parallel implementation: total running time = sum
- parallel implementation: total running time = max

Behaviour of sum/max will depend on the **tails** of the distribution.

Rejection sampling with random rate

Running time of rejection sampling follows a Geometric($1/M$), so exponential tails. But imagine each time you perform rejection sampling, the target and/or the proposal change. Then running time is a mixture of Geometric, which might have heavy tails, or even infinite expectation.

Sampling from a distribution with support of size N

Target distribution is $p(n) \propto w(n)$ for $n = 1, \ldots, N$.

Direct method

- Compute normalise weights, $W_n = w(n) / \sum_{m=1}^{N} w(m)$.
- Sample $U \sim \mathcal{U}[0, 1]$.
- find index k such that $\sum_{m=1}^{k-1} W_m \leq U < \sum_{m=1}^{k} W_m$.

Sampling from a distribution with support of size N

Target distribution is $p(n) \propto w(n)$ for $n = 1, \ldots, N$.

Direct method

- Compute normalise weights, $W_n = w(n) / \sum_{m=1}^{N} w(m)$.
- Sample $U \sim \mathcal{U}[0, 1]$.
- find index k such that $\sum_{m=1}^{k-1} W_m \leq U < \sum_{m=1}^{k} W_m$.

Deterministic running time, complexity $\mathcal{O}(N)$.

Sampling from a distribution with support of size N

Rejection with uniform proposal. Requires to know C such that $w(n) \leq C$.

Rejection sampling with uniform proposal

Repeat:

- Sample $X \sim \mathcal{U}\{1, \ldots, N\}$,
- Sample $U \sim \mathcal{U}[0, 1]$,

Until $U \leq w(X)/C$.

Complexity is $\mathcal{O}_P(1)$.

Sampling from a distribution with support of size N

Rejection with uniform proposal. Requires to know C such that $w(n) \leq C$.

Rejection sampling with uniform proposal

Repeat:

- Sample $X \sim \mathcal{U}\{1, \ldots, N\}$,
- Sample $U \sim \mathcal{U}[0, 1]$,

Until $U \leq w(X)/C$.

Complexity is $\mathcal{O}_P(1)$.
However, probability that running time is larger than previous algorithm is non-zero (and might be large).

Sampling from a distribution with support of size N

Rejection with uniform proposal. Requires to know C such that $w(n) \leq C$.

Rejection sampling with uniform proposal

Repeat:

- Sample $X \sim \mathcal{U}\{1, \ldots, N\}$,
- Sample $U \sim \mathcal{U}[0, 1]$,

Until $U \leq w(X)/C$.

Complexity is $\mathcal{O}_P(1)$.
However, probability that running time is larger than previous algorithm is non-zero (and might be large).

Hybrid strategy

- start with rejection sampling;
- if no sample has been accepted after N trials, switch to direct method.

Running time is then:

1. The min of the two approaches (up to constants)
2. random but **bounded**.

Relevance to particle smoothing

The particle smoothing algorithms discussed today sample recursively from empirical distributions of size N, in order to generate a **single** trajectory.

MCMC

Another example of an algorithm whose execution time is **deterministic**: MCMC. However, biased?

Outline

State-space models

$$\begin{array}{ccccccc} X_0 & \rightarrow & X_1 & \rightarrow & \cdots & \rightarrow & X_T \\ \downarrow & & \downarrow & & & & \downarrow \\ Y_0 & & Y_1 & & \cdots & & Y_T \end{array}$$

- $X_0, \ldots, X_T$: unobserved, possibly non-homogeneous Markov process
- $Y_0, \ldots, Y_T$: observations that are conditionally independent given $X_0, \ldots, X_T$. Typical case: Y_t is a noisy observation of X_t
- **Notation** $X_{0:T} := (X_0, \ldots, X_T)$, let $M_t(x_{t-1}, \mathrm{d}x_t)$ be the Markov transition from X_{t-1} to X_t, with probability density $m_t(x_{t-1}, x_t)$

Online smoothing

- We wish to approximate

$$\mathbb{E}[\psi_0(X_0) + \psi_1(X_0, X_1) + \ldots + \psi_t(X_{t-1}, X_t) | Y_{0:t}]$$

  preferably in an online fashion.

- Motivation 1: some SSM depends on a parameter θ, i.e. $p_\theta(x_{0:T}, y_{0:T})$

$$\nabla_\theta \log p_\theta(y_{0:t}) = \int \nabla_\theta \log p_\theta(x_{0:t}, y_{0:t}) p_\theta(x_{0:t}|y_{0:t}) \mathrm{d}x_{0:t}$$

  and

$$\nabla_\theta \log p(x_{0:t}, y_{0:t}) = \nabla_\theta [\log p(x_0) + \log p(y_0|x_0) + \\ + \sum_s \log p(x_s|x_{s-1}) + \log p(y_s|x_s)]$$

Particle filter

- A particle filter creates, at each time t, a set of N particles $X_t^{1:N}$ with N (normalised) weights $W_t^{1:N}$ which approximate $p(x_t|y_{0:t})$:

$$\sum_n W_t^n \varphi(X_t^n) \approx \int p(x_t|y_{0:t})\varphi(x_t)\mathrm{d}x_t$$

- A particle filter is a genetic algorithm, i.e. each particle X_t^n has an *ancestor*. Tracing the genealogy of a particle until time 0 gives an approximation[1] of the smoothing distribution $p(x_{0:t}|y_{0:t})$

[1] Del Moral & Miclo 2001

Population degeneracy

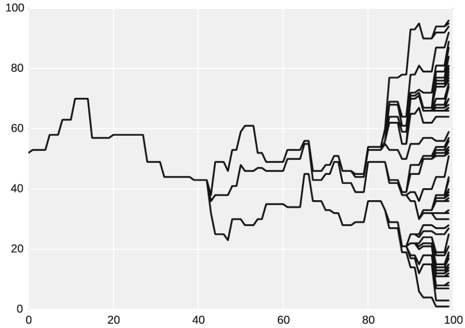
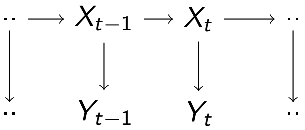


Figure extracted from Chap. 12, Chopin & Papaspiliopoulos 2020

- A fixed population of size N evolves from one generation to another.
- At generation t, each individual chooses *one* ancestor from generation $t-1$.
- After some generations, all individuals at time t have the same ancestor at time 0.
- Well-known phenomenon even outside of particle filter literature: Wright-Fisher model, Genetic drift, etc..

Backward sampling algorithms[2]



$$
\begin{aligned}
p(x_{t-1}|x_t, y_{0:T}) &\propto p(x_{t-1}|y_{0:t-1})p(x_t|x_{t-1}) \\
&\approx \sum_n \frac{W_{t-1}^n m_t(X_{t-1}^n, x_t)}{\sum_j W_{t-1}^j m_t(X_{t-1}^j, x_t)} \delta_{X_{t-1}^n}
\end{aligned}
$$

- Re-use the particles involved in the approximation of the filtering distribution $p(\mathrm{d}x_{t-1}|y_{0:t-1})$ to approximate the smoothing distribution $p(\mathrm{d}x_{t-1}|y_{0:T})$.
- The mixture distribution resamples a new ancestor for x_t, instead of reusing the old ancestor.

[2] Godsill, Doucet, West 2004

Online smoothing recursion[3]

$$\mathbb{E}[\psi(X_0)|Y_{0:T}] \approx \left[W_T^1 \dots W_T^N\right] \hat{B}_T^N \hat{B}_{T-1}^N \dots \hat{B}_1^N \begin{bmatrix} \psi(X_0^1) \\ \vdots \\ \psi(X_0^N) \end{bmatrix}.$$

where

$$\hat{B}_t^N[n, n'] := \frac{W_{t-1}^{n'} m_t(X_{t-1}^{n'}, X_t^n)}{\sum_j W_{t-1}^j m_t(X_{t-1}^j, X_t^n)}$$

- Translation to the matrix language of the previous slide.
- We can write RHS as $\left[W_T^1 \quad \dots \quad W_T^N\right] S_T^N$ where

$$S_{T+1}^N = \hat{B}_{T+1}^N S_T^N.$$

[3] Del Moral, Doucet, Singh 2010

Unbiased estimation of the transition matrix[4]

- $\hat{B}_t^N$ is a transition matrix. An unbiased sparse estimation can be constructed as follows
- For each n, sample $J_n^1, J_n^2 \overset{\text{iid}}{\sim} \hat{B}_t^N[n, \cdot]$.
- Define a new matrix $\hat{B}_t^{N,\text{PaRIS}}$ as

$$\hat{B}_t^{N,\text{PaRIS}}[n,k] := \begin{cases} 1/2 & \text{if } k = J_n^1 \text{ or } k = J_n^2 \\ 0 & \text{otherwise} \end{cases}$$

- $\hat{B}_t^{N,\text{PaRIS}}$ is sparse, accelerating the update $S_{T+1}^N = \hat{B}_{T+1}^N S_T^N$.
- Alternative view: given a particle X_t^n, resample two new ancestors $X_{t-1}^{J_n^1}$ and $X_{t-1}^{J_n^2}$.

[4] Olsson & Westerborn 2017

Outline

Rejection backward sampler[5]

- Recall that $\hat{B}_t^N[n, n'] := \frac{W_{t-1}^{n'} m_t(X_{t-1}^{n'}, X_t^n)}{\sum_j W_{t-1}^j m_t(X_{t-1}^j, X_t^n)}$.
- Sampling from $\hat{B}_t^N[n, \cdot]$ takes $\mathcal{O}(N)$. Running this for all n takes $\mathcal{O}(N^2)$.
- But $\hat{B}_t^N[n, \bullet] \propto W_{t-1}^{\bullet} m_t(X_{t-1}^{\bullet}, X_t^n)$ and usually $|m_t|$ is upper bounded.
- Thus one can sample from $\hat{B}_t^N$ using rejection sampling from the proposal distribution $W_{t-1}^{1:N}$ (recall that $\sum_n W_{t-1}^n = 1$)
- Sample from $\hat{B}_t^N[n, \cdot]$: using the *same* proposal for *different* n's.
- If m_t is also lower bounded away from 0, then the complexity is reduced to $\mathcal{O}(N)$.

[5] Douc, Garivier, Moulines, Olsson 2011

Low rejection rate problem[6][7]

- Even for linear Gaussian models, m_t isn't lower bounded away from 0.
- Many papers still repeat the claim that the complexity is linear, but note that rejection sampler might work badly.
- Proposed solution in the literature: stop the rejection sampler at some threshold, then use the "naive" sampler.
- **Unanswered questions.** What are the mathematical properties of the execution times when m_t is not bounded away from 0? How much improvement does early stopping bring? How should the threshold be chosen?

[6] Taghavi, Lindsten, Svensson, Schon 2013

[7] Olsson & Westerborn 2017

Our contributions

- We only consider PaRIS algorithm in this slide.
- Proposition 1: the expectation of the execution time for PaRIS-reject is infinite.
- Theorem 3.2: stopping the rejection sampler for each $\hat{B}_t^N[n, \cdot]$ after N trials gives an algorithm of overall complexity $\mathcal{O}(N \log^{d/2} N)$, in linear Gaussian models.

- Thm 3.2 is more complicated than Prop. 1
- Analysis for FFBS is more complicated than PaRIS (see Appendix).
- Choosing the threshold N is good enough.

A super-simplified example for intuition

Let $X \sim \mathcal{N}(0,1)$, then

$$\mathbb{E}\left[e^{X^2/2}\right] = \int_{\mathbb{R}} e^{x^2/2} \frac{e^{-x^2/2}}{\sqrt{2\pi}} = +\infty$$

but

$$\begin{aligned}\mathbb{E}\left[\min(e^{X^2/2}, N)\right] &= \int_{\mathbb{R}} \min(e^{x^2/2}, N) \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \mathrm{d}x \\ &= \int_{|x| \le \sqrt{2\log N}} \frac{1}{\sqrt{2\pi}} \mathrm{d}x + N \int_{|x| > \sqrt{2\log N}} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \mathrm{d}x \\ &\le \sqrt{\frac{4\log N}{\pi}} + \frac{1}{\sqrt{\pi \log N}}\end{aligned}$$

Outline

Sparse matrix estimation

- Recall that we approximate $\hat{B}_t^N$ by

$$\hat{B}_t^{N,\mathrm{PaRIS}}[n,k] := \begin{cases} 1/2 & \text{if } k = J_n^1 \text{ or } k = J_n^2 \\ 0 & \text{otherwise} \end{cases}$$

- $J^{n,1}$ and $J^{n,2}$ are sampled from $\hat{B}_t^N[n,\cdot]$, *without* taking into account the ancestor of X_t^n during the filtering step.
- Alternative idea: set $J^{n,1}$ to that ancestor; move $J^{n,1}$ through *one* MCMC step keeping invariant $\hat{B}_t^N[n,\cdot]$, and assign the result to $J^{n,2}$.
- The algorithm is **much faster** than rejection sampler, even if using early stopping:
  - In our experience, early stopping might take in expectation as much as 10 or 20 trials.
  - Here we have 1 trial only! Even if $J^{n,1} = J^{n,2}$ for a certain n (the MCMC proposal gets rejected), that won't be the case for other n.

Stability of the sparse estimation

- Bunch & Godsill (2013) uses this kind of MCMC step in the offline smoother, but does not prove stability.
- Proving stability, in particular non-asymptotic estimates, is difficult due to the sparse estimation.
- Olsson and Westerborn (2017) proved a CLT for PaRIS as $N \to \infty$ and T fixed, then showed that the asymp. variance is stable as $T \to \infty$.
- It's difficult to derive a CLT in our context since we don't want to impose a specified form on the MCMC kernel.
- Under strong assumptions, the original matrix $\hat{B}_t^N$ is Doeblin, i.e. it satisfies a contraction property. $\left\| \hat{B}_t^N f \right\|_{\text{osc}} \leq \rho \|f\|_{\text{osc}}$.
- Thus the product $\prod_t \hat{B}_t^N$ contracts exponentially fast.
- Each of the matrix $\hat{B}_t^{N,\mathrm{PaRIS}}$ is sparse and no longer contracts well. But their product is!

Our contribution: general stability theorem

Theorem

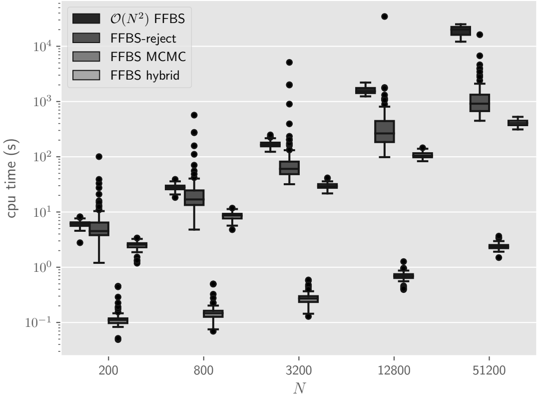
Under appropriate hypotheses, the quadratic error of the online smoothers are bounded by

$$\frac{C \sum_{t=0}^{T} \|\psi_t\|_\infty^2}{N} \left(1 + \sqrt{\frac{T}{N}}\right)^2 .$$

- Applies to PaRIS, MCMC in both offline and online situations.
- Consider $\mathbb{L}^2$ error which helps treating the aforementioned difficulty easier.
- First stability result for a truly $\mathcal{O}(N)$ smoother.

Outline

Numerical xp



Linear Gaussian state-space models, details in paper.

Outline

Conclusion

- the running time of rejection-based particle smoothers proposed in the literature may have infinite expectation (or variance).

Conclusion

- the running time of rejection-based particle smoothers proposed in the literature may have infinite expectation (or variance).
- Use hybrid rejection sampler (i.e. rejection with stopping) instead, or even better: MCMC.

Conclusion

- the running time of rejection-based particle smoothers proposed in the literature may have infinite expectation (or variance).
- Use hybrid rejection sampler (i.e. rejection with stopping) instead, or even better: MCMC.
- See our paper *arXiv 2207.00976*; also contains a coupling-based smoother for models with an intractable transition density.

Conclusion

- the running time of rejection-based particle smoothers proposed in the literature may have infinite expectation (or variance).
- Use hybrid rejection sampler (i.e. rejection with stopping) instead, or even better: MCMC.
- See our paper *arXiv 2207.00976*; also contains a coupling-based smoother for models with an intractable transition density.
- Some algorithms are implemented in particles (Python): `https://github.com/nchopin/particles`