Automatic Differentiation Accelerates Inference for Partially Observed Stochastic Processes

https://arxiv.org/abs/2407.03085

Kevin Tan[1], Giles Hooker[1] and Ed Ionides[2]
[1]University of Pennsylvania, [2]University of Michigan

STATS 810
December 9, 2024

Background

Partially Observed Markov Processes

- Unobserved Markov process $\{X_t, t \geq t_0\}$, observations $y_1^*, ..., y_N^*$ at timesteps $t_1, ..., t_N$.

Partially Observed Markov Processes

- Unobserved Markov process $\{X_t, t \geq t_0\}$, observations $y_1^*, ..., y_N^*$ at timesteps $t_1, ..., t_N$.
- Unknown parameter $\theta \in \Theta$, to be estimated.
- **Notation:**
  - $f_{X_n|X_{n-1}}(x_n \mid x_{n-1}; \theta)$ is the **process model**.

Partially Observed Markov Processes

- Unobserved Markov process $\{X_t, t \geq t_0\}$, observations $y_1^*, ..., y_N^*$ at timesteps $t_1, ..., t_N$.
- Unknown parameter $\theta \in \Theta$, to be estimated.
- **Notation:**
  - $f_{X_n|X_{n-1}}(x_n \mid x_{n-1}; \theta)$ is the **process model**.
  - $\texttt{process}(x_n, \theta)$ is the **simulator** corresponding to the process model.

Partially Observed Markov Processes

- Unobserved Markov process $\{X_t, t \geq t_0\}$, observations $y_1^*, ..., y_N^*$ at timesteps $t_1, ..., t_N$.
- Unknown parameter $\theta \in \Theta$, to be estimated.
- **Notation:**
  - $f_{X_n|X_{n-1}}(x_n \mid x_{n-1}; \theta)$ is the **process model**.
  - $\texttt{process}(x_n, \theta)$ is the **simulator** corresponding to the process model.
  - $f_{Y_n|X_n}(y_n \mid x_n, \theta)$ is the **measurement model**.

Particle Filters

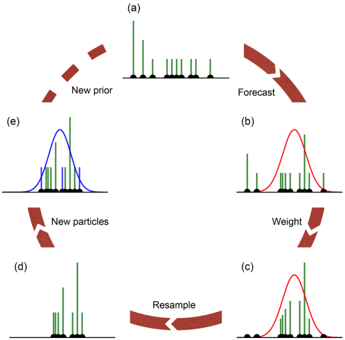

Figure 1:
– Dirac measure approximation of the Bayes filter.
– Yields a filtering distribution and likelihood estimate.
– Graphic from Berg et al. (2019).

- Monte Carlo approximation of the Bayes filter.

Particle Filters

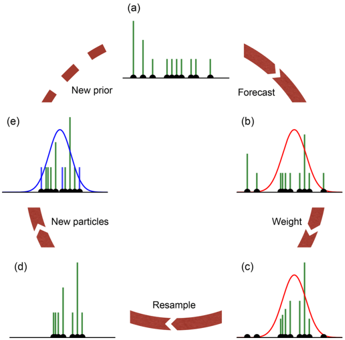

Figure 1:
– Dirac measure approximation of the Bayes filter.
– Yields a filtering distribution and likelihood estimate.
– Graphic from Berg et al. (2019).

- Monte Carlo approximation of the Bayes filter.
    - Maintain belief on current state x_n^F.

Particle Filters

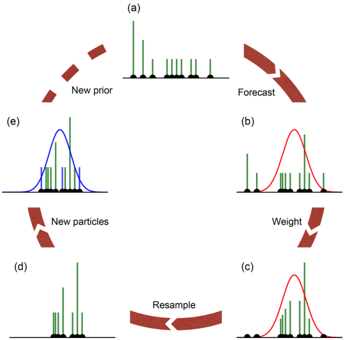

Figure 1:
– Dirac measure approximation of the Bayes filter.
– Yields a filtering distribution and likelihood estimate.
– Graphic from Berg et al. (2019).

- Monte Carlo approximation of the Bayes filter.
  - Maintain belief on current state x_n^F.
  - Simulate forward to get x_{n+1}^P, observe observation y_{n+1}^*.

Particle Filters

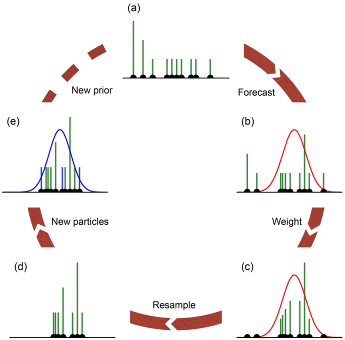

Figure 1:
– Dirac measure approximation of the Bayes filter.
– Yields a filtering distribution and likelihood estimate.
– Graphic from Berg et al. (2019).

- Monte Carlo approximation of the Bayes filter.
  - Maintain belief on current state x_n^F.
  - Simulate forward to get x_{n+1}^P, observe observation y_{n+1}^*.
  - Update x_{n+1}^F accordingly.

Automatic Differentiation

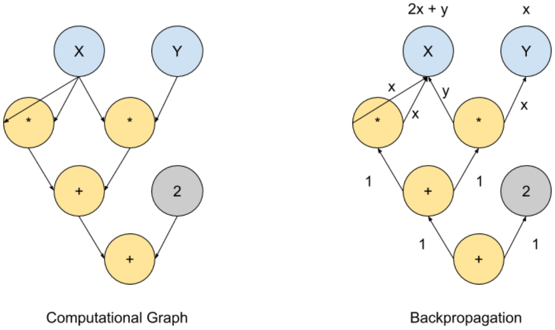

Figure 2: $f(x,y) = x^2 + xy + 2$, $f'(x,y) = (2x+y, x)$. – Figure from https://avinashselvam.medium.com

- Evaluates the gradient of a (scalar or vector-valued) computer program w.r.t. its arguments.

Automatic Differentiation

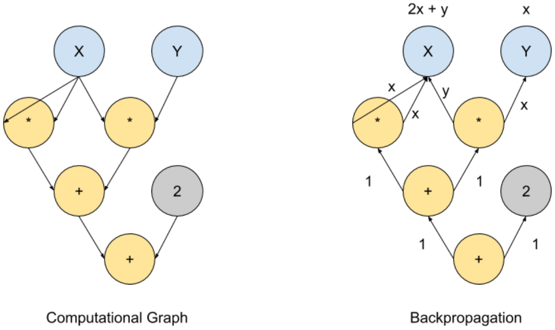

Figure 2:
$f(x, y) = x^2 + xy + 2$,
$f'(x, y) = (2x + y, x)$.
– Figure from `https://avinashselvam.medium.com`

- Evaluates the gradient of a (scalar or vector-valued) computer program w.r.t. its arguments.
- Traverses computational graph (of primitive functions) with chain rule.

Motivation

Maximum Likelihood Inference is Hard in POMPs

1. **Intractable likelihood functions**! Bypassed with e.g. simulated likelihood, likelihood-free inference, particle filters, etc.

Maximum Likelihood Inference is Hard in POMPs

1. **Intractable likelihood functions**! Bypassed with e.g. simulated likelihood, likelihood-free inference, particle filters, etc.
2. May not have access to transition densities, only a **simulator**.

Maximum Likelihood Inference is Hard in POMPs

1. **Intractable likelihood functions**! Bypassed with e.g. simulated likelihood, likelihood-free inference, particle filters, etc.
2. May not have access to transition densities, only a **simulator**.
   - IF2 (Ionides et al., 2015), particle MCMC (Andrieu et al., 2010), only full-information methods that can deal with this.

Maximum Likelihood Inference is Hard in POMPs

1. **Intractable likelihood functions**! Bypassed with e.g. simulated likelihood, likelihood-free inference, particle filters, etc.
2. May not have access to transition densities, only a **simulator**.
   - IF2 (Ionides et al., 2015), particle MCMC (Andrieu et al., 2010), only full-information methods that can deal with this.
3. **Significant Monte-Carlo noise in likelihood estimate** makes accurate parameter estimation difficult.

IF2 Plateaus Quickly But Struggles to Find the MLE

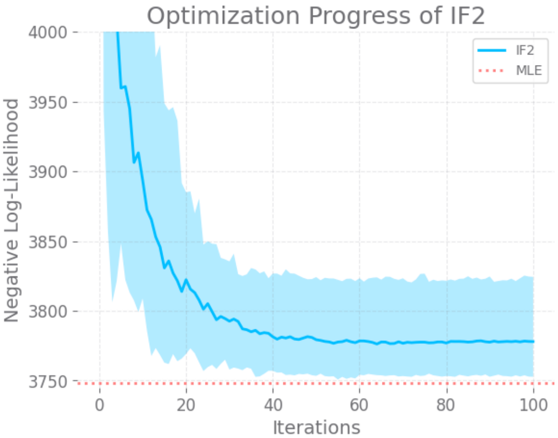

Figure 3: Performance of IF2 on King et al. (2008) Dhaka cholera model. Shaded area represents 0th to 80th percentile, solid line is median of 100 runs. While IF2 makes quick initial progress, it fails to find the MLE.

Hang on, wait a minute!

- Original iterated filtering algorithm: (noisy) approximation of score (Ionides et al., 2006).

Hang on, wait a minute!

- Original iterated filtering algorithm: (noisy) approximation of score (Ionides et al., 2006).
- **Would performing automatic differentiation (AD) on the likelihood estimate from the particle filter lead to a less noisy score approximation?**

Hang on, wait a minute!

- Original iterated filtering algorithm: (noisy) approximation of score (Ionides et al., 2006).
- **Would performing automatic differentiation (AD) on the likelihood estimate from the particle filter lead to a less noisy score approximation?**

Problem!

But the particle filter has discrete stochastic resampling! How can we differentiate this?

Previous Work: AD for Particle Filters

Previous Work

Authors	Method
Poyiadjis et al. (2011)	Particle approximation of score $O(N^2)$ variance, unbiased
Naesseth et al. (2018)	Backprop through vanilla PF Asymptotic bias
Corenflos et al. (2021)	Optimal transport resampling Consistent, $O(J^2)$ runtime
Ścibior and Wood (2021)	Stop-gradient trick Recovers Poyiadjis et al. (2011) with AD
Singh et al. (2022)	Fixed-lag smoothing Need transition densities

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.
    - Gradient estimates of Naesseth et al. (2018) (MOP-0), Poyiadjis et al. (2011) and Ścibior and Wood (2021) (MOP-1) are special cases.

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.
    - Gradient estimates of Naesseth et al. (2018) (MOP-0), Poyiadjis et al. (2011) and Ścibior and Wood (2021) (MOP-1) are special cases.
    - Does not need transition densities, only a differentiable simulator.

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.
  - Gradient estimates of Naesseth et al. (2018) (MOP-0), Poyiadjis et al. (2011) and Ścibior and Wood (2021) (MOP-1) are special cases.
  - Does not need transition densities, only a differentiable simulator.
  - Can optimize a bias-variance tradeoff.

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.
  - Gradient estimates of Naesseth et al. (2018) (MOP-0), Poyiadjis et al. (2011) and Ścibior and Wood (2021) (MOP-1) are special cases.
  - Does not need transition densities, only a differentiable simulator.
  - Can optimize a bias-variance tradeoff.
- Promising hybrid algorithm, warm-starts gradient descent (using this estimator) with IF2.

Our Contributions

- New theoretical framework/algorithm/gradient estimator we call MOP-α.
  - Gradient estimates of Naesseth et al. (2018) (MOP-0), Poyiadjis et al. (2011) and Ścibior and Wood (2021) (MOP-1) are special cases.
  - Does not need transition densities, only a differentiable simulator.
  - Can optimize a bias-variance tradeoff.
- Promising hybrid algorithm, warm-starts gradient descent (using this estimator) with IF2.
- Outperforms IF2 on Cholera model of King et al. (2008).

Smooth Extensions to the Particle Filter

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Idea

Don't differentiate $\hat{\ell}(\theta)$ directly, differentiate through a (suitably) reweighted bootstrap filter.

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Idea

Don't differentiate $\hat{\ell}(\theta)$ directly, differentiate through a (suitably) reweighted bootstrap filter.

- Particle filter run with state transitions & resampling under $\phi \in \Theta$.

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Idea

Don't differentiate $\hat{\ell}(\theta)$ directly, differentiate through a (suitably) reweighted bootstrap filter.

- Particle filter run with state transitions & resampling under $\phi \in \Theta$.
- Reweight to evaluate likelihood at nearby $\theta \in \Theta$ (same resampling).

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Idea

Don't differentiate $\hat{\ell}(\theta)$ directly, differentiate through a (suitably) reweighted bootstrap filter.

- Particle filter run with state transitions & resampling under $\phi \in \Theta$.
- Reweight to evaluate likelihood at nearby $\theta \in \Theta$ (same resampling).
- Fix seed to treat particles, weights under ϕ as constants.

Intuition

Problem

Want to differentiate through the particle filter. But the particle filter has discrete stochastic resampling!

Idea

Don't differentiate $\hat{\ell}(\theta)$ directly, differentiate through a (suitably) reweighted bootstrap filter.

- Particle filter run with state transitions & resampling under $\phi \in \Theta$.
- Reweight to evaluate likelihood at nearby $\theta \in \Theta$ (same resampling).
- Fix seed to treat particles, weights under ϕ as constants.
- **Now only need to differentiate through likelihood ratios!**

What is this reweighting?

The properly weighted estimate at θ with proposal from ϕ is

$$\hat{\mathcal{L}}(\theta) = \prod_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} \left(\underbrace{f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}, \theta)}_{\text{Measurement Model}} \cdot \frac{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \theta)}{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \phi)} \right).$$

What is this reweighting?

The properly weighted estimate at θ with proposal from ϕ is

$$\hat{\mathcal{L}}(\theta) = \prod_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} \left(\underbrace{f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}, \theta)}_{\text{Measurement Model}} \cdot \frac{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \theta)}{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \phi)} \right).$$

Which we write as

$$\hat{\mathcal{L}}(\theta) = \prod_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}, \phi) \cdot \underbrace{s_{n,j} \cdot r_{n,j}}_{\text{Correction Term}},$$

What is this reweighting?

The properly weighted estimate at θ with proposal from ϕ is

$$\hat{\mathcal{L}}(\theta) = \prod_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} \left(\underbrace{f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}, \theta)}_{\text{Measurement Model}} \cdot \frac{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \theta)}{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}, \phi)} \right).$$

Which we write as

$$\hat{\mathcal{L}}(\theta) = \prod_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}, \phi) \cdot \underbrace{s_{n,j} \cdot r_{n,j}}_{\text{Correction Term}},$$

where the multiplicative correction terms are

$$\underbrace{s_{n,j} = \frac{g_{n,j}^{\theta}}{g_{n,j}^{\phi}} = \frac{f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}; \theta)}{f_{Y_n|X_n}(y_n^*|x_{n,j}^{P,\phi}; \phi)}}_{\text{Measurement Model Likelihood Ratios}}, \quad \underbrace{r_{n,j} = \frac{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}; \theta)}{f_{X_n|X_{n-1}}(x_{n,j}^{P,\phi}|x_{n-1,j}^{F,\phi}; \phi)}}_{\text{Process Model Likelihood Ratios}}.$$

New Problem

Problem

Need likelihood ratios w.r.t. process and measurement models!

New Problem

Problem

Need likelihood ratios w.r.t. process and measurement models!

Solution

Offload process model derivatives to differentiable simulator.

New Problem

Problem

Need likelihood ratios w.r.t. process and measurement models!

Solution

Offload process model derivatives to differentiable simulator.

- Run particle filter twice under same seed: one with process model at ϕ, another with process model at θ. Resample according to ϕ.

New Problem

Problem

Need likelihood ratios w.r.t. process and measurement models!

Solution

Offload process model derivatives to differentiable simulator.

- Run particle filter twice under same seed: one with process model at ϕ, another with process model at θ. Resample according to ϕ.
- Correct only with resampling likelihood ratios.

New Problem

Problem

Need likelihood ratios w.r.t. process and measurement models!

Solution

Offload process model derivatives to differentiable simulator.

- Run particle filter twice under same seed: one with process model at ϕ, another with process model at θ. Resample according to ϕ.
- Correct only with resampling likelihood ratios.
- Introduce additional discounting parameter α for bias-variance tradeoff.

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at** ϕ to obtain $X_{n,j}^{P,\phi}, X_{n,j}^{F,\phi}$. Fix seed throughout. Set initial weights $w_{0,j}^{F,\theta} = 1/J$.

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at** ϕ to obtain $X_{n,j}^{P,\phi}, X_{n,j}^{F,\phi}$. Fix seed throughout. Set initial weights $w_{0,j}^{F,\theta} = 1/J$.
2. **At each timestep,** propagate weights $w_{n,j}^{P,\theta} := (w_{n-1,j}^{F,\theta})^{\alpha}$.

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at** ϕ to obtain $X^{P,\phi}_{n,j}, X^{F,\phi}_{n,j}$. Fix seed throughout. Set initial weights $w^{F,\theta}_{0,j} = 1/J$.
2. **At each timestep,** propagate weights $w^{P,\theta}_{n,j} := (w^{F,\theta}_{n-1,j})^{\alpha}$.
3. Simulate process model $X^{P,\theta}_{n,j} \sim f_{X_n|X_{n-1}}\big(\,\cdot\,|X^{F}_{n-1,j};\theta\big)$, evaluate measurement model $g^{\theta}_{n,j} = f_{Y_n|X_n}(y^*_n|X^{P,\theta}_{n,j}\,;\theta)$, evaluate conditional likelihood under ϕ, $L^{\phi}_n = \frac{1}{J}\sum_{m=1}^{J} g^{\phi}_{n,m}$, **as usual.**

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at** ϕ to obtain $X^{P,\phi}_{n,j}, X^{F,\phi}_{n,j}$. Fix seed throughout. Set initial weights $w^{F,\theta}_{0,j} = 1/J$.
2. **At each timestep,** propagate weights $w^{P,\theta}_{n,j} := (w^{F,\theta}_{n-1,j})^\alpha$.
3. Simulate process model $X^{P,\theta}_{n,j} \sim f_{X_n|X_{n-1}}\big(\cdot\,|X^F_{n-1,j};\theta\big)$, evaluate measurement model $g^\theta_{n,j} = f_{Y_n|X_n}(y^*_n|X^{P,\theta}_{n,j}\,;\theta)$, evaluate conditional likelihood under ϕ, $L^\phi_n = \frac{1}{J}\sum_{m=1}^J g^\phi_{n,m}$, **as usual.**
4. **Resample according to** ϕ**:** $k_{1:J} \sim \mathbb{P}\big(k_j = m\big) \propto g^\phi_{n,m}$.

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at** ϕ to obtain $X^{P,\phi}_{n,j}, X^{F,\phi}_{n,j}$. Fix seed throughout. Set initial weights $w^{F,\theta}_{0,j} = 1/J$.
2. **At each timestep,** propagate weights $w^{P,\theta}_{n,j} := (w^{F,\theta}_{n-1,j})^{\alpha}$.
3. Simulate process model $X^{P,\theta}_{n,j} \sim f_{X_n|X_{n-1}}\big(\cdot\,|X^{F}_{n-1,j};\theta\big)$, evaluate measurement model $g^{\theta}_{n,j} = f_{Y_n|X_n}(y^*_n|X^{P,\theta}_{n,j}\,;\theta)$, evaluate conditional likelihood under ϕ, $L^{\phi}_n = \frac{1}{J}\sum_{m=1}^{J} g^{\phi}_{n,m}$, **as usual.**
4. **Resample according to** ϕ**:** $k_{1:J} \sim \mathbb{P}\big(k_j = m\big) \propto g^{\phi}_{n,m}$.
5. **Correct filtering weights:** $w^{F,\theta}_{n,j} = w^{P,\theta}_{n,k_j} \times g^{\theta}_{n,k_j}/g^{\phi}_{n,k_j}$.

Algorithm: Measurement Off-Policy, MOP-α

1. **Run a particle filter once at ϕ** to obtain $X^{P,\phi}_{n,j}, X^{F,\phi}_{n,j}$. Fix seed throughout. Set initial weights $w^{F,\theta}_{0,j} = 1/J$.
2. **At each timestep,** propagate weights $w^{P,\theta}_{n,j} := (w^{F,\theta}_{n-1,j})^{\alpha}$.
3. Simulate process model $X^{P,\theta}_{n,j} \sim f_{X_n|X_{n-1}}\big(\cdot\,|X^{F}_{n-1,j};\theta\big)$, evaluate measurement model $g^{\theta}_{n,j} = f_{Y_n|X_n}(y^*_n|X^{P,\theta}_{n,j};\theta)$, evaluate conditional likelihood under ϕ, $L^{\phi}_n = \frac{1}{J}\sum_{m=1}^{J} g^{\phi}_{n,m}$, **as usual.**
4. **Resample according to ϕ:** $k_{1:J} \sim \mathbb{P}\big(k_j = m\big) \propto g^{\phi}_{n,m}$.
5. **Correct filtering weights:** $w^{F,\theta}_{n,j} = w^{P,\theta}_{n,k_j} \times g^{\theta}_{n,k_j}/g^{\phi}_{n,k_j}$.

Special Case: If $\theta = \phi$, only need to run one particle filter! Set $X^{P,\theta}_{n,j}, X^{F,\theta}_{n,j}$ to be copies of the ϕ counterparts, where gradients don't propagate. Explains the **stop-gradient** trick of Ścibior and Wood (2021).

Bias-Variance Tradeoff with α

- α balances a tradeoff between:

Bias-Variance Tradeoff with α

- α balances a tradeoff between:
    - Maintaining memory of each particle's ancestral trajectory (most extreme when $\alpha = 1$).

Bias-Variance Tradeoff with α

- α balances a tradeoff between:
  - Maintaining memory of each particle's ancestral trajectory (most extreme when $\alpha = 1$).
  - Considering only the single-step transition dynamics (when $\alpha = 0$).

Bias-Variance Tradeoff with α

- α balances a tradeoff between:
  - Maintaining memory of each particle's ancestral trajectory (most extreme when $\alpha = 1$).
  - Considering only the single-step transition dynamics (when $\alpha = 0$).
- **Exponentially-weighted moving average** where α controls the amount of discounting.

Bias-Variance Tradeoff with α

- α balances a tradeoff between:
  - Maintaining memory of each particle's ancestral trajectory (most extreme when $\alpha = 1$).
  - Considering only the single-step transition dynamics (when $\alpha = 0$).
- **Exponentially-weighted moving average** where α controls the amount of discounting.
- Bias small if $y^*_{n+1:N}$ not informative for x_n given $y^*_{0:n}$ (Corenflos et al., 2021).

Bias-Variance Tradeoff with α

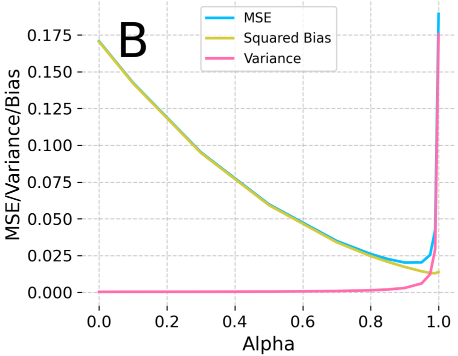

Figure 4: Bias-variance tradeoff for gradient estimates at MLE for trend in King et al. (2008) cholera model. When $\alpha = 1$, estimate close to unbiased with high variance. When $\alpha = 0$, lower variance but biased. MSE seems to be minimized at $\alpha = 0.97$.

Guarantee

Proposition (Correctness of MOP-α)

When either $\alpha = 1$ or $\theta = \phi$, MOP-α targets the posterior and is unbiased for the likelihood under θ. When θ is evaluated at ϕ, the likelihood estimate agrees with the bootstrap filter. Its gradient when $\alpha = 1$ is the estimate of Poyiadjis et al. (2011),

$$\frac{1}{J}\sum_{j=1}^{J} \nabla_\theta \log f_{Y_{1:N}|X_{0:N}}(y^*_{1:N}|x^{A,F,\theta}_{1:N,j}),$$

and when $\alpha = 0$, is the gradient estimator of Naesseth et al. (2018),

$$\frac{1}{J}\sum_{n=1}^{N}\sum_{j=1}^{J} \nabla_\theta \log f_{Y_n|X_n}(y^*_n|x^{F,\theta}_{n,j};\theta),$$

i.e. differentiating through a vanilla particle filter (Ścibior and Wood, 2021).

Practical Maximum Likelihood Inference

One possible procedure:

- Warm-start gradient descent, or some other first or second order procedure (using the gradient estimate given by MOP-α), with the output of an initial search of IF2.

One possible procedure:

- Warm-start gradient descent, or some other first or second order procedure (using the gradient estimate given by MOP-α), with the output of an initial search of IF2.
- Run IF2, aggressively cooling to a fixed learning rate, till search stalls.

One possible procedure:

- Warm-start gradient descent, or some other first or second order procedure (using the gradient estimate given by MOP-α), with the output of an initial search of IF2.
- Run IF2, aggressively cooling to a fixed learning rate, till search stalls.
- Refine this coarse solution with gradient descent.

One possible procedure:

- Warm-start gradient descent, or some other first or second order procedure (using the gradient estimate given by MOP-α), with the output of an initial search of IF2.
- Run IF2, aggressively cooling to a fixed learning rate, till search stalls.
- Refine this coarse solution with gradient descent.

Iterated Filtering with Automatic Differentiation (IFAD)

Convergence Analysis

Proposition (Convergence of IFAD-1)

Consider a variant of IFAD-1 where one stops if the gradient estimate $||g(\theta_n)|| \leq \sigma\epsilon$. Assume $-\ell$ is γ-strongly convex, $\gamma I \preceq \nabla^2_\theta(-\ell) \preceq \Gamma I$. $\sigma = \frac{4\Gamma}{(1-\beta)} > 4$. Define G as in Assumption 4.
Let $H(\theta_n)$ be a matrix possibly dependent on θ_n with minimum eigenvalue always greater than some $c > 0$. Choose learning rate η and error tolerance ϵ such that, where β is the Armijo condition hyperparameter,

$$\eta \leq \frac{c(1-\beta)}{\Gamma}, \;\; \epsilon \leq \frac{c(1-\beta)}{2\Gamma}||g(\theta_n)||.$$

Then, with J large enough, the following holds with probability at least $1-\delta$:

$$\ell(\theta^*) - \ell(\theta_{n+1}) \leq \left(1 - \eta\beta\frac{8\gamma}{9c}\right)(\ell(\theta^*) - \ell(\theta_n))$$

and the algorithm terminates when $||\nabla_\theta \ell(\theta_n)|| \leq (1+\sigma)\epsilon$.

Convergence Analysis

- Similar to Roosta-Khorasani and Mahoney (2016), uses concentration inequalities from Del Moral and Rio (2011).
- Gradient stage of IFAD-1 converges linearly to the MLE if:

Convergence Analysis

- Similar to Roosta-Khorasani and Mahoney (2016), uses concentration inequalities from Del Moral and Rio (2011).
- Gradient stage of IFAD-1 converges linearly to the MLE if:
    1. The log-likelihood surface is γ-strongly convex in a neighborhood of the MLE.

Convergence Analysis

- Similar to Roosta-Khorasani and Mahoney (2016), uses concentration inequalities from Del Moral and Rio (2011).
- Gradient stage of IFAD-1 converges linearly to the MLE if:
  1. The log-likelihood surface is γ-strongly convex in a neighborhood of the MLE.
  2. The IF2 stage of IFAD successfully reaches a (high-probability) basin of attraction of the MLE.

Convergence Analysis

- Similar to Roosta-Khorasani and Mahoney (2016), uses concentration inequalities from Del Moral and Rio (2011).
- Gradient stage of IFAD-1 converges linearly to the MLE if:
  1. The log-likelihood surface is γ-strongly convex in a neighborhood of the MLE.
  2. The IF2 stage of IFAD successfully reaches a (high-probability) basin of attraction of the MLE.
- **Conjecture:** This applies to the entirety of IFAD, as IF2 converges very quickly to a neighborhood of the MLE and behaves a lot like SGD.

Runtime

- Warm-start:

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.
    - Dhaka cholera model of King et al. (2008): 40 iterations, usually 100-200 for global search.

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.
    - Dhaka cholera model of King et al. (2008): 40 iterations, usually 100-200 for global search.
- MOP-α:

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.
    - Dhaka cholera model of King et al. (2008): 40 iterations, usually 100-200 for global search.
- MOP-α:
  - Gradient takes 3.75x time of `pfilter()`, in line with cheap gradient principle (Kakade and Lee, 2019).

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.
    - Dhaka cholera model of King et al. (2008): 40 iterations, usually 100-200 for global search.
- MOP-α:
  - Gradient takes 3.75x time of `pfilter()`, in line with cheap gradient principle (Kakade and Lee, 2019).
- Runtime **linear, and not quadratic**, in number of particles.

Runtime

- Warm-start:
  - Initial warm-start "convergence" happens fairly quickly in practice.
    - Dhaka cholera model of King et al. (2008): 40 iterations, usually 100-200 for global search.
- MOP-α:
  - Gradient takes 3.75x time of `pfilter()`, in line with cheap gradient principle (Kakade and Lee, 2019).
- Runtime **linear, and not quadratic**, in number of particles.
- Re-implementation in `JAX` (Bradbury et al., 2018) led to 16x speedup v.s. `pomp` package of King et al. (2008).

Numerical Experiments

Cholera in Bangladesh

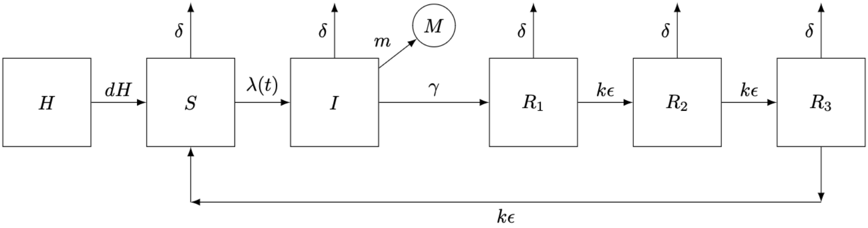

Figure 5: Illustration of SIR model from King et al. (2008).

- Dhaka model from King et al. (2008), also used in Ionides et al. (2015) to benchmark IF2.

Cholera in Bangladesh

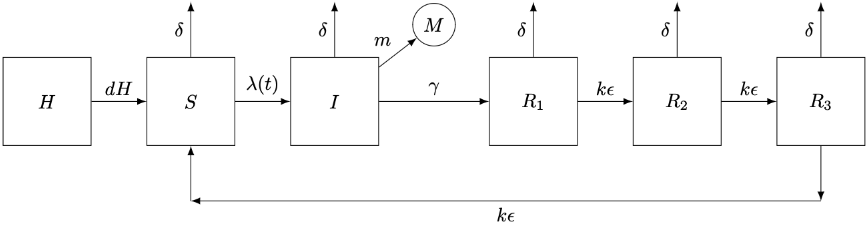

Figure 5: Illustration of SIR model from King et al. (2008).

- Dhaka model from King et al. (2008), also used in Ionides et al. (2015) to benchmark IF2.
- Stochastic SIR compartmental model with transition uncertainty driven by Brownian motion.

Cholera in Bangladesh

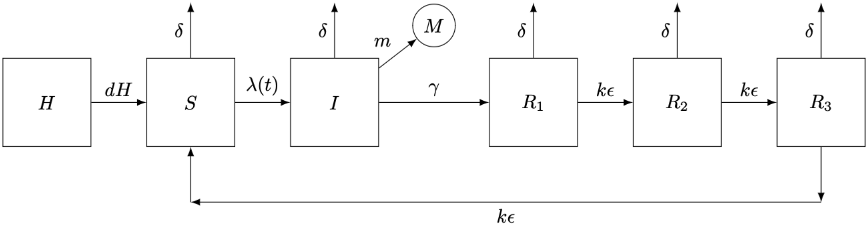

Figure 5: Illustration of SIR model from King et al. (2008).

- Dhaka model from King et al. (2008), also used in Ionides et al. (2015) to benchmark IF2.
- Stochastic SIR compartmental model with transition uncertainty driven by Brownian motion.
- Force of infection $\lambda(t)$ modeled with splines for seasonality, etc.

Results

Method	Best Log-Likelihood	Rank
IFAD-0.97	**-3750.21**	1
IFAD-0	-3752.17	2
IFAD-1	-3754.63	3
IF2 (Ours)	-3764.10	4
IF2 (Ionides et al. (2015))	-3768.63	5
MOP-1 Alone (100 searches)	-3797.38	6

Table 1:
– IFAD performs the best among all methods, finding the MLE.
– Our implementation of IF2 outperforms that of Ionides et al. (2015) but underperforms IFAD.

- Benchmarked IFAD against IF2 on a challenging global search problem.

Results

Method	Best Log-Likelihood	Rank
IFAD-0.97	**-3750.21**	1
IFAD-0	-3752.17	2
IFAD-1	-3754.63	3
IF2 (Ours)	-3764.10	4
IF2 (Ionides et al. (2015))	-3768.63	5
MOP-1 Alone (100 searches)	-3797.38	6

Table 1:
– IFAD performs the best among all methods, finding the MLE.
– Our implementation of IF2 outperforms that of Ionides et al. (2015) but underperforms IFAD.

- Benchmarked IFAD against IF2 on a challenging global search problem.
- Performed 44 searches each.

Results

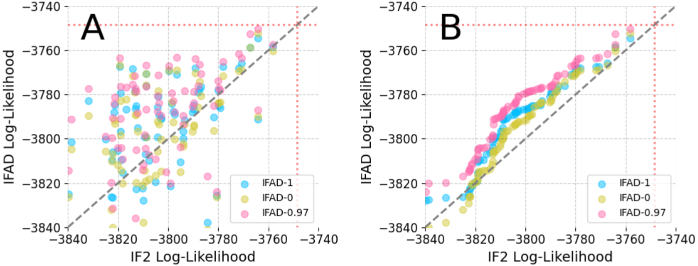

Figure 6:

–**Left:** Paired searches from the same starting point.

–**Right:** Q-Q plot of ranked IFAD searches against ranked IF2 searches.

– IFAD has the edge and manages to find the MLE.

– No IF2 search successfully gets within 7 log-likelihood units of it.

Results

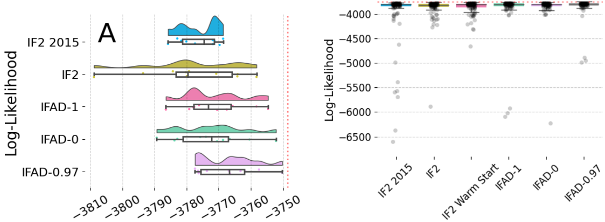

Figure 7:
– **Left:** Results of best run out of every ten runs, simulating procedure of running a few searches and choosing the best one.
– **Right:** MOP alone drastically underperforms all other methods, failing to get close to the MLE.
– IF2 warm start necessary in challenging nonconvex and noisy problems.

Results

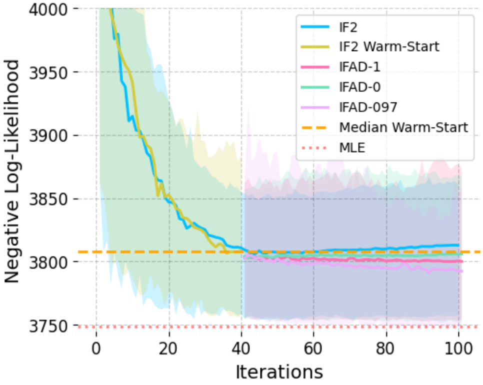

Figure 8:

– Solid lines depict the median negative log-likelihood at each iteration.

– Shaded area depicts the best search at any iteration and the 80% percentile.

Results

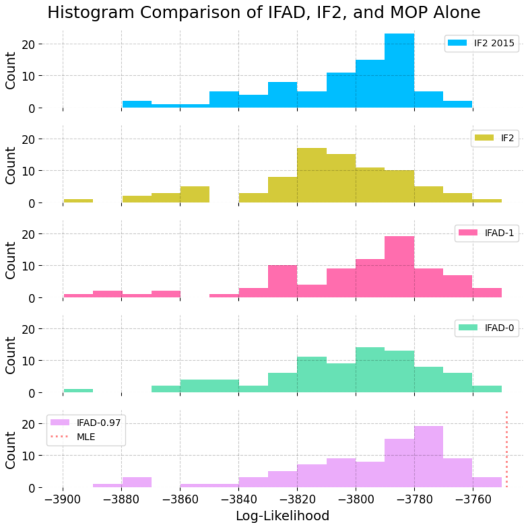

Figure 9: Histogram comparison of IFAD (with various α) and IF2. Right is better.

Takeaways

1. IF2 converges quickly to a neighborhood of the MLE but fails to find the MLE.

Takeaways

1. IF2 converges quickly to a neighborhood of the MLE but fails to find the MLE.
2. Gradient steps perform better at fine-grained refinement.

Takeaways

1. IF2 converges quickly to a neighborhood of the MLE but fails to find the MLE.
2. Gradient steps perform better at fine-grained refinement.
3. Performing gradient steps alone without a warm start leads to the search getting stuck in local minima and saddle points.

Takeaways

1. IF2 converges quickly to a neighborhood of the MLE but fails to find the MLE.
2. Gradient steps perform better at fine-grained refinement.
3. Performing gradient steps alone without a warm start leads to the search getting stuck in local minima and saddle points.
4. IFAD combines the best of IF2 and MOP, approaching the MLE quickly and successfully performing refinement – even on a very difficult global search problem.

Conclusions and Future Work

Conclusion

- New theoretical framework/algorithm/gradient estimator that encompasses a few existing gradient estimates.

Conclusion

- New theoretical framework/algorithm/gradient estimator that encompasses a few existing gradient estimates.
- Promising hybrid algorithm that warm-starts gradient descent (using this estimator) with IF2.

Conclusion

- New theoretical framework/algorithm/gradient estimator that encompasses a few existing gradient estimates.
- Promising hybrid algorithm that warm-starts gradient descent (using this estimator) with IF2.
- Outperforms IF2 on Dhaka cholera model of King et al. (2015).

Future Work

- MOP-α cannot handle discrete latent states. Maybe likelihood ratios can get around this.

Future Work

- MOP-α cannot handle discrete latent states. Maybe likelihood ratios can get around this.
- Convergence rate of IF2 not known, conjecture similar behavior to SGD.

Future Work

- MOP-α cannot handle discrete latent states. Maybe likelihood ratios can get around this.
- Convergence rate of IF2 not known, conjecture similar behavior to SGD.
- Only gradient descent explored, other methods like Newton's method, ADAM, L-BFGS, etc. may perform better.

Future Work

- MOP-α cannot handle discrete latent states. Maybe likelihood ratios can get around this.
- Convergence rate of IF2 not known, conjecture similar behavior to SGD.
- Only gradient descent explored, other methods like Newton's method, ADAM, L-BFGS, etc. may perform better.
- Extensions to panel and spatiotemporal data, e.g. with a block particle filter in the lens of Ionides et al. (2022) and Ning and Ionides (2023).

Future Work

- MOP-α cannot handle discrete latent states. Maybe likelihood ratios can get around this.
- Convergence rate of IF2 not known, conjecture similar behavior to SGD.
- Only gradient descent explored, other methods like Newton's method, ADAM, L-BFGS, etc. may perform better.
- Extensions to panel and spatiotemporal data, e.g. with a block particle filter in the lens of Ionides et al. (2022) and Ning and Ionides (2023).
- Python counterpart to the popular `pomp` R package by King et al. (2016).

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society, Series B*, 72:269–342.

Berg, D., Bauser, H., and Roth, K. (2019). Covariance resampling for particle filter – state and parameter estimation for soil hydrology. *Hydrology and Earth System Sciences*, 23:1163–1178.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Corenflos, A., Thornton, J., Deligiannidis, G., and Doucet, A. (2021). Differentiable particle filtering via entropy-regularized optimal transport. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2100–2111. PMLR.

Del Moral, P. and Rio, E. (2011). Concentration inequalities for mean field particle models. *The Annals of Applied Probability*, 21(3).

Ionides, E. L., Bretó, C., and King, A. A. (2006). Inference for nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the USA*, 103:18438–18443.

Ionides, E. L., Nguyen, D., Atchadé, Y., Stoev, S., and King, A. A. (2015). Inference for dynamic and latent variable models via iterated, perturbed Bayes maps. *Proceedings of the National Academy of Sciences of the USA*, 112(3):719—-724.

Ionides, E. L., Ning, N., and Wheeler, J. (2022). An iterated block particle filter for inference on coupled dynamic systems with shared and unit-specific parameters. *Statistica Sinica*, pages pre–published online.

Kakade, S. and Lee, J. D. (2019). Provably correct automatic subdifferentiation for qualified programs. *arXiv*, *1809.08530*.

King, A. A., Domenech de Cellès, M., Magpantay, F. M. G., and Rohani, P. (2015). Avoidable errors in the modelling of outbreaks of emerging pathogens, with special reference to Ebola. *Proceedings of the Royal Society of London, Series B*, 282(1806):20150347.

King, A. A., Ionides, E. L., Pascual, M., and Bouma, M. J. (2008). Inapparent infections and cholera dynamics. *Nature*, 454:877–880.

King, A. A., Nguyen, D., and Ionides, E. L. (2016). Statistical inference for partially observed Markov processes via the R package pomp. *Journal of Statistical Software*, 69:1–43.

Naesseth, C., Linderman, S., Ranganath, R., and Blei, D. (2018). Variational sequential Monte Carlo. In Storkey, A. and Perez-Cruz, F., editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 968–977. PMLR.

Ning, N. and Ionides, E. L. (2023). Iterated block particle filter for high-dimensional parameter learning: Beating the curse of dimensionality. *Journal of Machine Learning Research*, 24:1–76.

Poyiadjis, G., Doucet, A., and Singh, S. S. (2011). Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80.

Roosta-Khorasani, F. and Mahoney, M. W. (2016). Sub-sampled Newton methods I: Globally convergent algorithms. *arXiv*, *1601.04737*.

Ścibior, A. and Wood, F. (2021). Differentiable particle filtering without modifying the forward pass. *arXiv*, 2106.10314.

Singh, A., Makhlouf, O., Igl, M., Messias, J., Doucet, A., and Whiteson, S. (2022). Particle-based score estimation for state space model learning in autonomous driving. *arXiv*, *2212.06968*.

Appendix

Previous Work

- Poyiadjis et al. (2011), particle approximation of the score.

Previous Work

- Poyiadjis et al. (2011), particle approximation of the score. Ścibior and Wood (2021): AD + "stop-gradient trick".

Previous Work

- Poyiadjis et al. (2011), particle approximation of the score.
  Ścibior and Wood (2021): AD + "stop-gradient trick".
- Naesseth et al. (2018), backpropagate through vanilla particle filter.

Previous Work

- Poyiadjis et al. (2011), particle approximation of the score.
  Ścibior and Wood (2021): AD + "stop-gradient trick".
- Naesseth et al. (2018), backpropagate through vanilla particle filter.
- Corenflos et al. (2021), optimal transport resampling.

Previous Work

- Poyiadjis et al. (2011), particle approximation of the score. Ścibior and Wood (2021): AD + "stop-gradient trick".
- Naesseth et al. (2018), backpropagate through vanilla particle filter.
- Corenflos et al. (2021), optimal transport resampling.
- Singh et al. (2022), fixed-lag smoothing.

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
  - Need modifications to use the bootstrap filter.

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
    - Need modifications to use the bootstrap filter.
    - Sometimes special cases needed, e.g. transitions factoring into a policy and deterministic model (Singh et al. (2022)).

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
  - Need modifications to use the bootstrap filter.
  - Sometimes special cases needed, e.g. transitions factoring into a policy and deterministic model (Singh et al. (2022)).
- Computationally expensive, quadratic in number of particles.

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
  - Need modifications to use the bootstrap filter.
  - Sometimes special cases needed, e.g. transitions factoring into a policy and deterministic model (Singh et al. (2022)).
- Computationally expensive, quadratic in number of particles.
  - Optimal transport resampling (Corenflos et al. (2021)), marginal particle filters (Ścibior and Wood (2021)).

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
  - Need modifications to use the bootstrap filter.
  - Sometimes special cases needed, e.g. transitions factoring into a policy and deterministic model (Singh et al. (2022)).
- Computationally expensive, quadratic in number of particles.
  - Optimal transport resampling (Corenflos et al. (2021)), marginal particle filters (Ścibior and Wood (2021)).
- High variance, or asymptotically biased.

Previous Work: Issues

- Not (yet) compatible with desire for simulation-based inference.
  - Need modifications to use the bootstrap filter.
  - Sometimes special cases needed, e.g. transitions factoring into a policy and deterministic model (Singh et al. (2022)).
- Computationally expensive, quadratic in number of particles.
  - Optimal transport resampling (Corenflos et al. (2021)), marginal particle filters (Ścibior and Wood (2021)).
- High variance, or asymptotically biased.
  - Either one drops resampling terms and accepts asymptotic bias (Naesseth et al. (2018)), or has variance quadratic in horizon (Poyiadjis et al. (2011), Ścibior and Wood (2021))

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.
    - Each particle has its own set of parameters.

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.
  - Each particle has its own set of parameters.
  - Particle parameters perturbed at every timestep.

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.
    - Each particle has its own set of parameters.
    - Particle parameters perturbed at every timestep.
    - Parameters resampled with their particles according to likelihood.

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.
  - Each particle has its own set of parameters.
  - Particle parameters perturbed at every timestep.
  - Parameters resampled with their particles according to likelihood.
- Treats parameters as augmentations of the state space, evolving according to a random walk.

(Improved) Iterated Filtering (Ionides et al. (2015))

- Only (practical) full-information, frequentist method of parameter estimation in POMPs that does not need transition densities.
  - Each particle has its own set of parameters.
  - Particle parameters perturbed at every timestep.
  - Parameters resampled with their particles according to likelihood.
- Treats parameters as augmentations of the state space, evolving according to a random walk.
- **Consistent estimates** via viewing this as a sequential Bayes map on the parameter distribution.

When Does IF2 Struggle?

- **IF2 is great at approaching the MLE quickly!**

When Does IF2 Struggle?

- **IF2 is great at approaching the MLE quickly!**
- Performs parameter updates at every timestep, so it does not need to wait to filter through the entire trajectory before each update.

When Does IF2 Struggle?

- **IF2 is great at approaching the MLE quickly!**
- Performs parameter updates at every timestep, so it does not need to wait to filter through the entire trajectory before each update.
- **But struggles at squeezing out the last few units of log-likelihood.**

When Does IF2 Struggle?

- **IF2 is great at approaching the MLE quickly!**
- Performs parameter updates at every timestep, so it does not need to wait to filter through the entire trajectory before each update.
- **But struggles at squeezing out the last few units of log-likelihood.**
  - Especially true in highly nonlinear, nonconvex, and noisy settings, e.g. **disease models**.

Assumptions

Assumption (Smooth Neighborhood)

There exists a neighborhood $\mathcal{N}(\phi)$ *around any* $\phi \in \Theta$ *where for all* $\theta \in \mathcal{N}(\phi)$ *and almost every* $\omega \in \Omega$*, the Monte Carlo estimate of the likelihood at* $\theta \in \mathcal{N}(\phi)$ *with the system evolving according to* ϕ *conditional on* ω*,* $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$*, is twice differentiable in* θ*.*

Assumptions

Assumption (Smooth Neighborhood)

There exists a neighborhood $\mathcal{N}(\phi)$ around any $\phi \in \Theta$ where for all $\theta \in \mathcal{N}(\phi)$ and almost every $\omega \in \Omega$, the Monte Carlo estimate of the likelihood at $\theta \in \mathcal{N}(\phi)$ with the system evolving according to ϕ conditional on ω, $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$, is twice differentiable in θ.

- **Justification:** For suitably nearby θ and ϕ, the resampling indices for $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$ and $\hat{\mathcal{L}}(\phi, \phi, \omega, J)$ are the same.

Assumptions

Assumption (Smooth Neighborhood)

There exists a neighborhood $\mathcal{N}(\phi)$ around any $\phi \in \Theta$ where for all $\theta \in \mathcal{N}(\phi)$ and almost every $\omega \in \Omega$, the Monte Carlo estimate of the likelihood at $\theta \in \mathcal{N}(\phi)$ with the system evolving according to ϕ conditional on ω, $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$, is twice differentiable in θ.

- **Justification:** For suitably nearby θ and ϕ, the resampling indices for $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$ and $\hat{\mathcal{L}}(\phi, \phi, \omega, J)$ are the same.
- This eliminates any discontinuities from resampling.

Assumptions

Assumption (Smooth Neighborhood)

There exists a neighborhood $\mathcal{N}(\phi)$ around any $\phi \in \Theta$ where for all $\theta \in \mathcal{N}(\phi)$ and almost every $\omega \in \Omega$, the Monte Carlo estimate of the likelihood at $\theta \in \mathcal{N}(\phi)$ with the system evolving according to ϕ conditional on ω, $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$, is twice differentiable in θ.

- **Justification:** For suitably nearby θ and ϕ, the resampling indices for $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$ and $\hat{\mathcal{L}}(\phi, \phi, \omega, J)$ are the same.
- This eliminates any discontinuities from resampling.
- For small enough $\mathcal{N}(\phi)$, the likelihood ratios are bounded.

Assumptions

Assumption (Smooth Neighborhood)

There exists a neighborhood $\mathcal{N}(\phi)$ around any $\phi \in \Theta$ where for all $\theta \in \mathcal{N}(\phi)$ and almost every $\omega \in \Omega$, the Monte Carlo estimate of the likelihood at $\theta \in \mathcal{N}(\phi)$ with the system evolving according to ϕ conditional on ω, $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$, is twice differentiable in θ.

- **Justification:** For suitably nearby θ and ϕ, the resampling indices for $\hat{\mathcal{L}}(\theta, \phi, \omega, J)$ and $\hat{\mathcal{L}}(\phi, \phi, \omega, J)$ are the same.
- This eliminates any discontinuities from resampling.
- For small enough $\mathcal{N}(\phi)$, the likelihood ratios are bounded.
- The likelihood only changes by a factor of the likelihood ratios, which are bounded and smooth in θ if the densities used in the calculation are.

Assumptions

Assumption (Continuity of the Likelihood)

$\ell(\theta)$ proper is continuous in a neighborhood $\{\theta : \ell(\theta) > \lambda_1\}$ for some $\lambda_1 < \sup_\varphi \ell(\varphi)$.

Assumptions

Assumption (Continuity of the Likelihood)

$\ell(\theta)$ *proper is continuous in a neighborhood* $\{\theta : \ell(\theta) > \lambda_1\}$ *for some* $\lambda_1 < \sup_\varphi \ell(\varphi)$.

Assumption (Bounded Measurement Model)

There is an $\epsilon > 0$ *with* $\epsilon^{-1} > f_{Y_n|X_n}(y_n{}^* \mid x_n; \theta) > \epsilon$ *for all* $1 \leq n \leq N, x_n \in \mathcal{X}$ *and* $\theta \in \Theta$.

Assumptions

Assumption (Continuity of the Likelihood)

$\ell(\theta)$ proper is continuous in a neighborhood $\{\theta : \ell(\theta) > \lambda_1\}$ for some $\lambda_1 < \sup_\varphi \ell(\varphi)$.

Assumption (Bounded Measurement Model)

There is an $\epsilon > 0$ with $\epsilon^{-1} > f_{Y_n|X_n}({y_n}^* \mid x_n; \theta) > \epsilon$ for all $1 \leq n \leq N, x_n \in \mathcal{X}$ and $\theta \in \Theta$.

Assumption (Locally Bounded Derivative)

Let $\mathcal{M}$ be an open subset of Ω. There exists some function $G(\theta)$ and a constant $G = \sup_{\theta \in \mathcal{N}} G(\theta) < \infty$ such that

$$\|\nabla \ell(\theta, \phi, \omega, J)\|_2 < G(\theta) \leq G < \infty$$

for every $\phi \in \mathcal{M}, \theta$ in smooth neighborhood $\mathcal{N}(\phi)$, and almost every $\omega \in \Omega$.

Algorithm: Measurement Off-Policy, MOP-α

Algorithm 3 Measurement Off-Policy-α

Input: Number of particles J, timesteps N, measurement model $f_{Y_n|X_n}(y_n^*|x_n,\theta)$, simulator $\texttt{process}(x_{n+1}|x_n,\theta)$, evaluation parameter θ, behavior parameter ϕ, seed ω.
if $\theta \neq \phi$ **then**
  Run a particle filter once with parameters ϕ to obtain $X_{n,j}^{P,\phi}, X_{n,j}^{F,\phi}$, with seed ω.
else
  Set $X_{n,j}^{P,\phi}, X_{n,j}^{F,\phi}$ to be copies of $X_{n,j}^{P,\theta}, X_{n,j}^{P,\theta}$ in the rest of the algorithm.
end if
Initialize filter particles $X_{0,j}^{F,\theta} \sim f_{X_0}(\cdot\,;\theta)$, relative weights $w_{0,j}^{F,\theta} = 1$ for j in $1\!:\!J$. Fix ω.
for $n = 1, ..., N$ **do**
  Prediction weights with discounting: $w_{n,j}^{P,\theta} = \big(w_{n-1,j}^{F,\theta}\big)^{\alpha}$ for j in $1\!:\!J$
  Simulate for prediction: $X_{n,j}^{P,\theta} \sim f_{X_n|X_{n-1}}\big(\,\cdot\,|X_{n-1,j}^{F};\theta\big)$ for j in $1\!:\!J$
  Evaluate measurement density: $g_{n,j}^{\theta} = f_{Y_n|X_n}(y_n^*|X_{n,j}^{P,\theta}\,;\theta)$ for j in $1\!:\!J$
  Before-resampling conditional likelihood: $L_n^{B,\theta,\alpha} = \dfrac{\sum_{j=1}^{J} g_{n,j}^{\theta} w_{n,j}^{P,\theta}}{\sum_{j=1}^{J} w_{n,j}^{P,\theta}}$
  Conditional likelihood under ϕ: $L_n^{\phi} = \frac{1}{J}\sum_{m=1}^{J} g_{n,m}^{\phi}$
  Normalize weights: $\tilde{g}_{n,j}^{\phi} = \dfrac{g_{n,j}^{\phi}}{J L_n^{\phi}}$ for j in $1\!:\!J$
  Apply systematic resampling to select indices $k_{1:J}$ with $\mathbb{P}\big(k_j = m\big) = \tilde{g}_{n,m}^{\phi}$
  Resample particles: $X_{n,j}^{F,\phi} = X_{n,k_j}^{P,\phi}$
  Filter weights corrected for resampling: $w_{n,j}^{FC,\theta} = w_{n,j}^{P,\theta} \times \dfrac{g_{n,j}^{\theta}}{g_{n,j}^{\phi}}$ for j in $1\!:\!J$
  Resample filter weights: $w_{n,j}^{F,\theta} = w_{n,k_j}^{FC,\theta}$ for j in $1\!:\!J$
  After-resampling conditional likelihood: $L_n^{A,\theta,\alpha} = L_n^{\phi}\dfrac{\sum_{j=1}^{J} w_{n,j}^{F,\theta}}{\sum_{j=1}^{J} w_{n,j}^{P,\theta}}$
end for
return likelihood estimate $\hat{\mathcal{L}}(\theta) = \mathcal{L}(\theta,\phi,\omega,J) := \prod_{n=1}^{N} L_n^{A,\theta,\alpha}$, filtering distribution $\{(X_{N,j}^{F,\theta}, w_{N,j}^{F,\theta})\}$.

Why did we bother with this?

- Lets us work with only a differentiable simulator.

Why did we bother with this?

- Lets us work with only a differentiable simulator.
- Discounting parameter α helps with a bias-variance tradeoff (explained later).

Why did we bother with this?

- Lets us work with only a differentiable simulator.
- Discounting parameter α helps with a bias-variance tradeoff (explained later).
- Gradients returned by the likelihood estimate

$$\hat{\mathcal{L}}(\theta) := \prod_{n=1}^{N} L_n^{A,\theta,\alpha} = L_n^{\phi} \frac{\sum_{j=1}^{J} w_{n,j}^{F,\theta}}{\sum_{j=1}^{J} w_{n,j}^{P,\theta}}$$

have nice properties.

Iterated Filtering with Automatic Differentiation (IFAD)

Algorithm 1 Iterated Filtering with Automatic Differentiation

```
Input: Number of particles J, timesteps N, measurement model p(y*_n | x_n, θ), simulator process(x_{n+1} | x_n, θ),
IF2 cooling schedule η_t, MOP-α discounting parameter α.
Initialize θ_0, t = 0.
Run IF2 until initial "convergence" under cooling schedule η_t to obtain {Θ_j, j = 1, ..., J}, set θ_t := (1/J) Σ_{j=1}^J Θ_j.
while Procedure not converged: do
    Draw ω ∈ Ω
    Perform forward pass by running Algorithm 3 with discounting parameter α to obtain ℓ(θ_t, θ_t, ω, J).
    Obtain g(θ_t) = ∇_{θ_t}(−ℓ(θ_t, θ_t, ω, J)) via backpropagation, H(θ_t) such that λ_min(H) ≥ c.
    Obtain η with line search or an annealing schedule.
    Update θ_{t+1} := θ_t − η(H(θ_t))^{-1} g(θ_t), t := t + 1.
end while
Return θ̂ := θ_t.
```

Figure 11: Warm-starting first/second order iterative optimization with IF2.

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.
- Gradient descent gets stuck in saddle points and poor local minima when the likelihood is nonconvex.

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.
- Gradient descent gets stuck in saddle points and poor local minima when the likelihood is nonconvex.
- Warm-starting gradient methods with IF2:

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.
- Gradient descent gets stuck in saddle points and poor local minima when the likelihood is nonconvex.
- Warm-starting gradient methods with IF2:
    - Under regularity conditions, the likelihood is well-behaved near the MLE.

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.
- Gradient descent gets stuck in saddle points and poor local minima when the likelihood is nonconvex.
- Warm-starting gradient methods with IF2:
  - Under regularity conditions, the likelihood is well-behaved near the MLE.
  - Issues with saddle points and local minima alleviated.

Why does this help?

- IF2 converges quickly, but struggles with last few log-likelihood units.
- Gradient descent gets stuck in saddle points and poor local minima when the likelihood is nonconvex.
- Warm-starting gradient methods with IF2:
  - Under regularity conditions, the likelihood is well-behaved near the MLE.
  - Issues with saddle points and local minima alleviated.
- Combining these two lets us enjoy the best of both worlds.

Convergence Analysis

Guarantee currently only holds for IFAD-1, as it forms a particle approximation of the score as in Poyiadjis et al. (2011).

Convergence Analysis

Guarantee currently only holds for IFAD-1, as it forms a particle approximation of the score as in Poyiadjis et al. (2011).

- Can be extended if the bias for the gradient estimate given by MOP-α for $\alpha < 1$ is small enough.

Convergence Analysis

Guarantee currently only holds for IFAD-1, as it forms a particle approximation of the score as in Poyiadjis et al. (2011).

- Can be extended if the bias for the gradient estimate given by MOP-α for $\alpha < 1$ is small enough.
- e.g. α close to 1, or $y^*_{n+1:N}$ uninformative of current state x_n given past and current measurements $y^*_{0:n}$.

Convergence Analysis

Guarantee currently only holds for IFAD-1, as it forms a particle approximation of the score as in Poyiadjis et al. (2011).

- Can be extended if the bias for the gradient estimate given by MOP-α for $\alpha < 1$ is small enough.
- e.g. α close to 1, or $y^*_{n+1:N}$ uninformative of current state x_n given past and current measurements $y^*_{0:n}$.
- Otherwise, need to handle biased gradient descent.

Cholera in Bangladesh

The **transition dynamics** follow this series of stochastic differential equations driven by Brownian motion:

$$
\begin{aligned}
dS &= (k\epsilon R_k + \delta(S - H) - \lambda(t)S)\,dt + dH - (\sigma SI/H)dB, \\
dI &= (\lambda(t)S - (m + \delta + \gamma)I)\,dt + (\sigma SI/P)dB, \\
dR_1 &= (\gamma I - (k\epsilon + \delta)R_1)\,dt, \\
&\vdots \\
dR_k &= (k\epsilon R_{k-1} - (k\epsilon + \delta)R_k)\,dt,
\end{aligned}
$$

Cholera in Bangladesh

The **force of infection**, $\lambda(t)$, is modeled by splines s_j,

$$\lambda(t) = \exp\left\{\beta_{\text{trend}}\ (t - t_0) + \sum_{j=1}^{6} \beta_j s_j(t)\right\}(I/P) + \exp\left\{\sum_{j=1}^{6} \omega_j s_j(t)\right\},$$

where

Cholera in Bangladesh

The **force of infection**, $\lambda(t)$, is modeled by splines s_j,

$$\lambda(t) = \exp\left\{\beta_{\text{trend}}\ (t - t_0) + \sum_{j=1}^{6} \beta_j s_j(t)\right\}(I/P) + \exp\left\{\sum_{j=1}^{6} \omega_j s_j(t)\right\},$$

where

- β_j model seasonality in the force of infection.

Cholera in Bangladesh

The **force of infection**, $\lambda(t)$, is modeled by splines s_j,

$$\lambda(t) = \exp\left\{\beta_{\text{trend}}\ (t - t_0) + \sum_{j=1}^{6} \beta_j s_j(t)\right\}(I/P) + \exp\left\{\sum_{j=1}^{6} \omega_j s_j(t)\right\},$$

where

- β_j model seasonality in the force of infection.
- β_{trend} models the trend in the force of infection.

Cholera in Bangladesh

The **force of infection**, $\lambda(t)$, is modeled by splines s_j,

$$\lambda(t) = \exp\left\{\beta_{\text{trend}}\ (t - t_0) + \sum_{j=1}^{6} \beta_j s_j(t)\right\}(I/P) + \exp\left\{\sum_{j=1}^{6} \omega_j s_j(t)\right\},$$

where

- β_j model seasonality in the force of infection.
- β_{trend} models the trend in the force of infection.
- ω_j represent seasonality of a non-human environmental reservoir of disease.

Cholera in Bangladesh

The **measurement model** for observed monthly cholera deaths is given by

$$Y_n \sim \mathcal{N}(M_n, \tau^2 M_n^2),$$

where M_n is the true number of cholera deaths in that month.