Particle Filters and Sequential Monte Carlo

AC 209b: Advanced Section

February 16, 2022

Outline

1. State-space models
2. Kalman Filter
3. Building blocks of particle filters
4. Particle filter algorithm
5. Extensions and modifications

State-Space Models

State-space models (SSM)

A state-space models is a latent process that varies over time, with noisy observations of the underlying process.

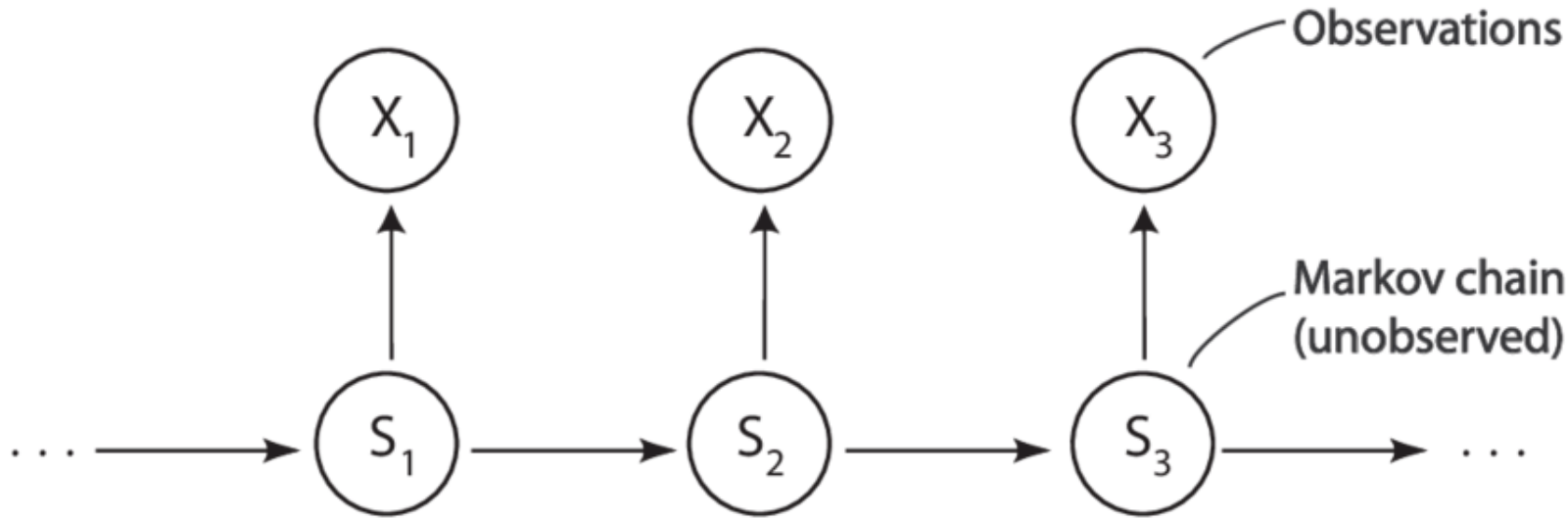


Image credit: Bulla (2006)

SSM Example: SIR Model

- Common model for epidemiology
- SIR = susceptible, infected, recovered
- If we have a noisy measurement of number of infections, can we estimate the true number of patients in each category over time?

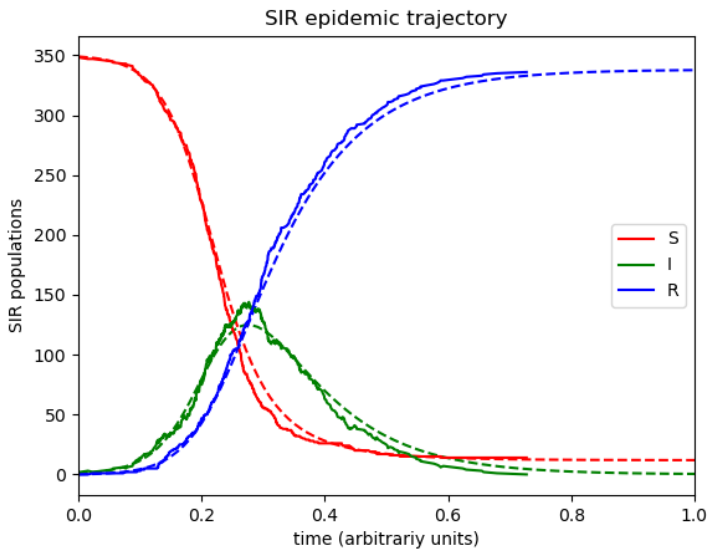


Image credit: Wikipedia

Example model

Linear Gaussian model:

$$
\begin{aligned}
y_t &= \beta\theta_t + \epsilon_t \\
\theta_t &= \alpha\theta_{t-1} + \eta_t \\
\varepsilon_t &\overset{iid}{\sim} N(0, \sigma_\varepsilon^2) \\
\eta_t &\overset{iid}{\sim} N(0, \sigma_\eta^2)
\end{aligned}
$$

Also known as Normal random walk, Dynamic linear model.

SSM Components: Transition model

- θ_t is the latent state at time t
- y_t is the observation at time t
- Note: can easily be extended to multidimensional setting
- Also called innovation model

Transition model

- $p(\theta_t \mid \theta_{t-1})$: describes how latent process varies over time
- Assumes Markov property:
  $p(\theta_t \mid \theta_{t-1}) = p(\theta_t \mid \theta_1, \theta_2, \ldots \theta_{t-2}, \theta_{t-1})$

Linear Gaussian transition model: $p(\theta_t \mid \theta_{t-1}) \sim N(\alpha\theta_{t-1}, \sigma_\eta^2)$

SSM Components: Observation model

- θ_t is the latent state at time t
- y_t is the observation at time t

Observation model

- $p(y_t \mid \theta_t)$: describes how latent states translate into observations
- Also called measurement model

Linear Gaussian observation model: $p(y_t \mid \theta_t) \sim N(\beta\theta_t, \sigma_\varepsilon^2)$

Applications of state-space models

Note that SSMs can also apply to variation over *position* rather than time, such as a DNA sequence or spatial variation.

- Epidemiology: SIR and more complicated models
- Ecology: population abundance (on your homework!)
- Geology: earthquake risk

Image credit: Wikimedia

Applications of state-space models

Image credit: Wikimedia

- Meteorology: storm strength
- Health: blood pressure or other noisy/imperfect measurements
- Bioinformatics: DNA sequencing

Hidden Markov Models

- Hidden Markov Models (HMM) and State-Space Models (SSM) have identical structure
- Sometimes the terms are used interchangeably!
- Essentially the same from a statistical point of view! Just impacts probability distributions.

SSMs and HMMs

- HMM models assume latent state is discrete, e.g. a binary indicator or category: is it sunny or raining? which gene mutation does the patient have?
- SSM models assume latent states can be continuous: what is the true value a company's stock? what is the patient's immunity level?

Questions of interest for SSMs

- Filtering: estimating the latent Markov process given the noisy observations up to and including that time point
  - $p(\theta_t \mid y_{1:t})$
  - Can also sometimes refer to the goal of estimating the *whole* latent process $p(\theta_{1:T} \mid y_{1:T})$ or a subset $p(\theta_{1:t} \mid y_{1:t})$.
- Smoothing: estimating the latent Markov process given the noisy observations at *all* time points (including future observations)
  - $p(\theta_t \mid y_{1:T})$
- Model parameter estimation: estimating parameters that inform the state-space model. For today, we will always assume these are known!
  - $p(\alpha, \beta, \sigma^2_\varepsilon, \sigma^2_\eta \mid y_{1:T})$

Questions of interest for SSMs

I will be focusing on *filtering*. Generally the easiest of these three aims.

First we consider Linear, Gaussian models:

- All of these problems are easier.
- Can often derive analytic, closed-form solutions (i.e. you can write down a solution for $p(\theta_t \mid y_{1:t})$).

Next for non-linear or non-Gaussian models:

- Much more difficult!
- Solution: particle filters.

Kalman Filter

Filtering for Linear Gaussian model

Returning to Linear Gaussian model:

$$
\begin{aligned}
y_t &= \beta\theta_t + \epsilon_t \\
\theta_t &= \alpha\theta_{t-1} + \eta_t \\
\varepsilon_t &\overset{iid}{\sim} N(0, \sigma^2_\varepsilon) \\
\eta_t &\overset{iid}{\sim} N(0, \sigma^2_\eta)
\end{aligned}
$$

Filtering: interested in posterior distribution $p(\theta_1, \ldots \theta_T \mid y_1, \ldots y_T)$.
We assume $\alpha, \beta, \sigma^2_\varepsilon, \sigma^2_\eta$ are known.

Filtering: prior distribution

Result 1: sequential distribution of prior.

First, consider the prior density $p(\theta_1, \ldots \theta_T)$.

$$
\begin{aligned}
p(\theta_1, \ldots \theta_T) &= p(\theta_T \mid \theta_{T-1}, \ldots \theta_1) p(\theta_{T-1}, \ldots \theta_1) \\
&= p(\theta_T \mid \theta_{T-1}) p(\theta_{T-1}, \ldots \theta_1) \quad \text{by Markov property} \\
&= p(\theta_T \mid \theta_{T-1}) p(\theta_{T-1} \mid \theta_{T-2}) p(\theta_{T-2}, \ldots \theta_1) \\
&= \ldots \\
&= p(\theta_T \mid \theta_{T-1}) p(\theta_{T-1} \mid \theta_{T-2}) \ldots p(\theta_2 \mid \theta_1) p(\theta_1)
\end{aligned}
$$

Filtering: likelihood

Result 2: sequential distribution of likelihood.

Next, consider the likelihood density $p(y_1, \ldots y_T \mid \theta_1, \ldots \theta_T)$.

$$\begin{aligned} p(y_1, \ldots, y_T \mid \theta_1, \ldots, \theta_T) &= p(y_T \mid y_{T-1}, \ldots, y_1, \theta_T, \ldots, \theta_1) \\ &\quad \times p(y_{T-1}, \ldots y_1, \theta_T, \ldots, \theta_1) \\ &= p(y_T \mid \theta_T) p(y_{T-1}, \ldots y_1, \theta_T, \ldots, \theta_1) \\ &= \ldots \\ &= p(y_T \mid \theta_T) \ldots p(y_2 \mid \theta_2) p(y_1 \mid \theta_1) \end{aligned}$$

Filtering: posterior

Result 3: sequential distribution of posterior.

Finally, consider the posterior density $p(\theta_1, \ldots \theta_T \mid y_1, \ldots y_T)$.
Combining results from above:

$$
\begin{aligned}
& p(\theta_1, \ldots \theta_T \mid y_1, \ldots y_T) \\
& \propto \underbrace{p(\theta_T \mid \theta_{T-1}) p(\theta_{T-1} \mid \theta_{T-2}) \ldots p(\theta_2 \mid \theta_1) p(\theta_1)}_{\text{prior}} \\
& \times \underbrace{p(y_T \mid \theta_T) \ldots p(y_2 \mid \theta_2) p(y_1 \mid \theta_1)}_{\text{likelihood}}
\end{aligned}
$$

Re-arranging terms:

$$
\begin{aligned}
& p(\theta_1, \ldots \theta_T \mid y_1, \ldots y_T) \propto \\
& p(y_T \mid \theta_T) p(\theta_T \mid \theta_{T-1}) \ldots p(y_2 \mid \theta_2) p(\theta_2 \mid \theta_1) p(\theta_1)
\end{aligned}
$$

Filtering:MVN

Result 4: the posterior distribution is Multivariate Normal.

We've already shown:

$$p(\theta_t \mid \theta_{t-1}) \sim N(\alpha\theta_{t-1}, \sigma^2_\eta)$$

$$p(y_t \mid \theta_t) \sim N(\beta\theta_t, \sigma^2_\varepsilon)$$

Then all components of the posterior are Normal, so $p(\theta_1, \ldots \theta_T \mid y_1, \ldots y_T)$ is MVN.

Filtering: Marginal distribution

We may be interested in *just* $p(\theta_t \mid y_{1:t})$. Could marginalize over $\theta_1, \ldots \theta_{t-1}$:

$$p(\theta_t \mid y_1, \ldots y_t) = \int p(\theta_1, \ldots \theta_t \mid y_1, \ldots y_t) d\theta_1, \ldots \theta_{t-1}$$

Could also be interested in forecast $p(\theta_{t+1} \mid y_{1:t})$. Then would also need to marginalize over θ_t.
This marginalization is potentially difficult!

More efficient approach is Kalman filter: recursive algorithm for optimal linear forecasts θ_{t+1} and y_{t+1} from $y_{1:t}$.

Kalman Filter result

Define:

$$a_t = E(\theta_t \mid y_{1:t-1})$$
$$v_t = Var(\theta_t \mid y_{1:t-1})$$

Then one can show:

$$a_{t+1} = \alpha a_t + \alpha v_t \beta \frac{y_t - \beta a_t}{\beta^2 v_t + \sigma_\varepsilon^2}$$
$$v_{t+1} = \sigma_\eta^2 + \alpha^2 v_t - \frac{\alpha^2 \beta^2 v_t^2}{\beta^2 v_t + \sigma_\varepsilon^2}$$

Kalman Filter result

$$a_{t+1} = \alpha a_t + \alpha v_t \beta \frac{y_t - \beta a_t}{\beta^2 v_t + \sigma_\varepsilon^2}$$

$$v_{t+1} = \sigma_\eta^2 + \alpha^2 v_t - \frac{\alpha^2 \beta^2 v_t^2}{\beta^2 v_t + \sigma_\varepsilon^2}$$

- Takeaway: recursive algorithm!
- Values of a_{t+1} and v_{t+1} depend only y_t, previous values a_t and v_t, and known constants.
- Choose appropriate starting values a_1 and v_1 and proceed; for many models, initial conditions do not make a big impact (for inference on θ_T at the end of the process).

Derivation of Kalman filter: MVN distribution

$$p(\theta_{t+1} \mid y_{1:t}) \propto \int \underbrace{p(y_t \mid \theta_t)}_{\text{likelihood}} \underbrace{p(\theta_{t+1} \mid \theta_t)}_{\text{transition}} \underbrace{p(\theta_t \mid y_{1:t-1})}_{\text{“prior”}} d\theta_t$$

Again, all distributions on the right are Normal, so left side is MVN, and joint distribution of $p(\theta_{t+1}, y_t)$ is MVN.

Derivation of Kalman filter

Then we can use MVN properties. If two (scalar) variables X and Y are joint MVN:

$$E(X \mid Y) = E(X) + \frac{Cov(X, Y)}{Var(Y)}(Y - E(Y)))$$

$$Var(X \mid Y) = Var(X) - \frac{Cov(X, Y)^2}{Var(Y)}$$

Take $X = \theta_{t+1}$ and $Y = y_t$. We will always be conditioning on $y_{1:t-1}$.

Derivation of Kalman filter

Plugging in to our context:

$$E(\theta_{t+1} \mid y_t, y_{1:t-1}) =$$
$$E(\theta_{t+1} \mid y_{1:t-1}) + \frac{Cov(\theta_{t+1}, y_t \mid y_{1:t-1})}{Var(y_t \mid y_{1:t-1})} (y_t - E(y_t \mid y_{1:t-1}))$$

We will now need to evaluate each of the expressions on the right-hand side.

Derivation of Kalman filter

We can evaluate the "unconditional" expectation of θ_{t+1} (not conditional on y_t):

$$\begin{aligned} E(\theta_{t+1} \mid y_{1:t-1}) &= E(\alpha\theta_t + \eta_t \mid y_{1:t-1}) \\ &= \alpha a_t \end{aligned}$$

And the expectation of y_t:

$$\begin{aligned} E(y_t \mid y_{1:t-1}) &= E(E(y_t \mid \theta_t, y_{1:t-1}) \mid y_{1:t-1}) \\ &= E(\beta\theta_t \mid y_{1:t-1}) = \beta a_t \end{aligned}$$

Derivation of Kalman filter

Unconditional variance of θ_{t+1}:

$$\begin{aligned} Var(\theta_{t+1} \mid y_{1:t-1}) &= Var(\alpha\theta_t + \eta_t \mid y_{1:t-1}) \\ &= \alpha^2 v_t + \sigma_\eta^2 \end{aligned}$$

Variance of y_t:

$$\begin{aligned} Var(y_t \mid y_{1:t-1}) &= Var(\beta\theta_t + \epsilon_t \mid y_{1:t-1}) \\ &= \beta^2 v_t + \sigma_\varepsilon^2 \end{aligned}$$

Derivation of Kalman filter

Finally covariance:

$$\begin{aligned} Cov(\theta_{t+1}, y_t \mid y_{t-1}) &= Cov(\alpha\theta_t + \eta_t, \beta\theta_t + \epsilon_t \mid y_{t-1}) \\ &= \alpha\beta Var(\theta_t \mid y_{t-1}) \\ &= \alpha\beta v_t \end{aligned}$$

Plugging all of these expressions gets us to our original result.

Summary: Kalman Filter

With the following model:

$$y_t = \beta\theta_t + \epsilon_t$$
$$\theta_t = \alpha\theta_{t-1} + \eta_t$$
$$\varepsilon_t \overset{iid}{\sim} N(0, \sigma_\varepsilon^2)$$
$$\eta_t \overset{iid}{\sim} N(0, \sigma_\eta^2)$$

Define:

$$a_t = E(\theta_t \mid y_{1:t-1})$$
$$v_t = Var(\theta_t \mid y_{1:t-1})$$

We can show:

$$a_{t+1} = \alpha a_t + \alpha v_t \beta \frac{y_t - \beta a_t}{\beta^2 v_t + \sigma_\varepsilon^2}$$
$$v_{t+1} = \sigma_\eta^2 + \alpha^2 v_t - \frac{\alpha^2 \beta^2 v_t^2}{\beta^2 v_t + \sigma_\varepsilon^2}$$

Building blocks of particle filters

What is a particle?

- The term particle refers to approximating a distribution given a finite number of random values (or vectors, if multivariate).
- For example, we may approximate $p(\theta_t)$ by the empirical distribution of N particles drawn from $p(\theta_t)$.
- Just another name for a *sample!*
- Note that a particle can be multidimensional: each particle could contain a vector of parameters all sampled at once, such as if θ_t is multivariate, or if we want to jointly sample from θ_t, α, β, etc.

Particle filter algorithm

First, a basic sketch of the particle filter algorithm. At each time point:

- Transition particles according to latent transition model $p(\theta_t \mid \theta_{t-1})$ to get θ_t.
- Calculate weights according to the observation model $w_t \propto p(y_t \mid \theta_t)$.
- Resample particles so that each has equal probability.

Building blocks of particle filters

To understand particle filters, first we need to backtrack and build up our understanding! Particle filters are built on a long history of other sampling procedures.

- Monte Carlo sampling
- Bayesian weighted bootstrap
- Resampling

Short review: Monte Carlo sampling

Let's begin with a short review of Monte Carlo sampling. I will start with a simpler problem than filtering, but it easily extends to more complicated problems.

- Goal: approximate some function $f(X)$ of a random variable, such as the mean $\mu = E(X)$ for some random variable $X \sim p(x)$
- Strategy: sample N values $X_i \sim p(x)$ for $i = 1, \ldots, N$
- Estimator: $\hat{\mu}_{MC} = \frac{1}{N} \sum_{i=1}^{N} X_i$

When sampling is useful

When do we even need Monte Carlo sampling?

- Sometimes calculating $f(X)$ analytically is difficult/impossible but sampling from $p(x)$ is easy.
- Example: X is a random variable where we first sample $Y \sim Pois(4)$, then if $Y > 4$, then we sample from $X \sim Gamma(4, 3)$ and if $Y < 4$ then we sample from $Exponential(1.2)$.
- Though we can still calculate $E(X)$ in this case...
- Easy to imagine a situation where sampling is easy, analytic calculations are hard!

Review: Bayesian weighted bootstrap

We are interested in sampling from posterior:
$h(\theta) = p(\theta \mid y) \propto p(\theta)L(\theta \mid y)$.

- Simulate values $\theta_1, \ldots, \theta_N$ from prior $p(\theta)$.
- Calculate w_i proportional to likelihood $L(\theta_i \mid y)$ (sometimes called importance weights).
- Normalize: $W_i = w_i / \sum_{i=1}^{N} w_i$.
- Sample $\theta^\star$ from $\theta_1, \ldots, \theta_N$ from $p(\theta)$ with probabilities $W_1, \ldots W_N$.
- Then $\theta^\star$ is approximately from $p(\theta \mid y)$.

Building blocks: Bayesian weighted bootstrap

Connecting back to state-space models...

- We may not be able to sample directly from $p(\theta_{1:t} \mid y_{1:t})$ with a complex state-space model.
- Example: high dimensional, non-linear, non-Gaussian.
- Even if we can, it may be computationally expensive!

Building blocks: Bayesian weighted bootstrap

Connecting back to state-space models...Interested in sampling from "posterior" $p(\theta_{1:t} \mid y_{1:t})$.

- Sample from "prior" $p(\theta_{1:t} \mid \theta_{1:t-1})$.
- Update based on "likelihood" $p(y_t \mid \theta_t)$.
- Resample.

Building blocks: Multinomial resampling

Different types of sampling/resampling exist. One common method used in particle filters is Multinomial resampling.

Multinomial resampling

Given a N vector z and an N vector of weights W, draw values of z *with replacement* to arrive at a new N vector $z^\star$.

- Same idea as in Bayesian bootstrap.
- Approximation of distribution is better with more draws N.

Multinomial resampling

Example: Generate $z \sim Pois(10)$

$$z = \begin{pmatrix} 13 & 10 & 21 & 10 \end{pmatrix}$$

$$w = \begin{pmatrix} 0.07 & 0.13 & 0.00 & 0.13 \end{pmatrix}$$

$$W = \begin{pmatrix} 0.21 & 0.395 & 0.00 & 0.395 \end{pmatrix}$$

After multinomial resampling:

$$z^\star = \begin{pmatrix} 10 & 13 & 10 & 10 \end{pmatrix}$$

We have discarded the unlikely value of 21.

Particle filter algorithm

Recursion in filtering

We can express $p(\theta_t \mid y_{1:t})$ recursively:

$$p(\theta_{1:t} \mid y_{1:t}) = p(\theta_{1:t-1} \mid y_{1:t-1}) \frac{p(y_t \mid \theta_t) p(\theta_t \mid \theta_{t-1})}{p(y_t \mid y_{1:t-1})}$$

with

$$p(y_t \mid y_{1:t-1}) = \int p(\theta_t \mid \theta_{t-1}) p(\theta_{t-1} \mid y_{1:t-1}) p(y_t \mid \theta_t) d\theta_{t-1:t}$$

Consequence: we can compute distributions sequentially!

Reminder: very similar to results from Kalman filter, but we are not assuming specific forms of distributions here, so we cannot get closed-form result on left-hand side.

Sequential sampling

- Sequential procedure is much more computationally efficient.
- Option 1: sample entire vector $p(\theta_{1:t})$ then calculate importance weights based on $y_{1:t}$.
- Easy to imagine many of these importance weights will be zero!

Resampling and sequential sampling

- Option 2: sample θ_1, calculate importance weights based on y_1.
- Resample: consequence is to *discard* unlikely values of θ_1.
- Transition to θ_2 based only on these likely values of θ_1.
- Rinse and repeat until we reach θ_t.
- Much more likely to end up with paths $\theta_{1:t}$ that are likely based on $y_{1:t}$!

Full algorithm: Initialize

Initialize

For particles $i = 1, \ldots, N$:

1. Sample $\theta_1^{(i)} \sim p(\theta_1)$
2. Calculate weights $w_1^{(i)} \propto p(y_1 \mid \theta_1^{(i)})$ and normalize weights $W_1^{(i)} = w_1^{(i)} / \sum_{i=1}^{N} w_1^{(i)}$

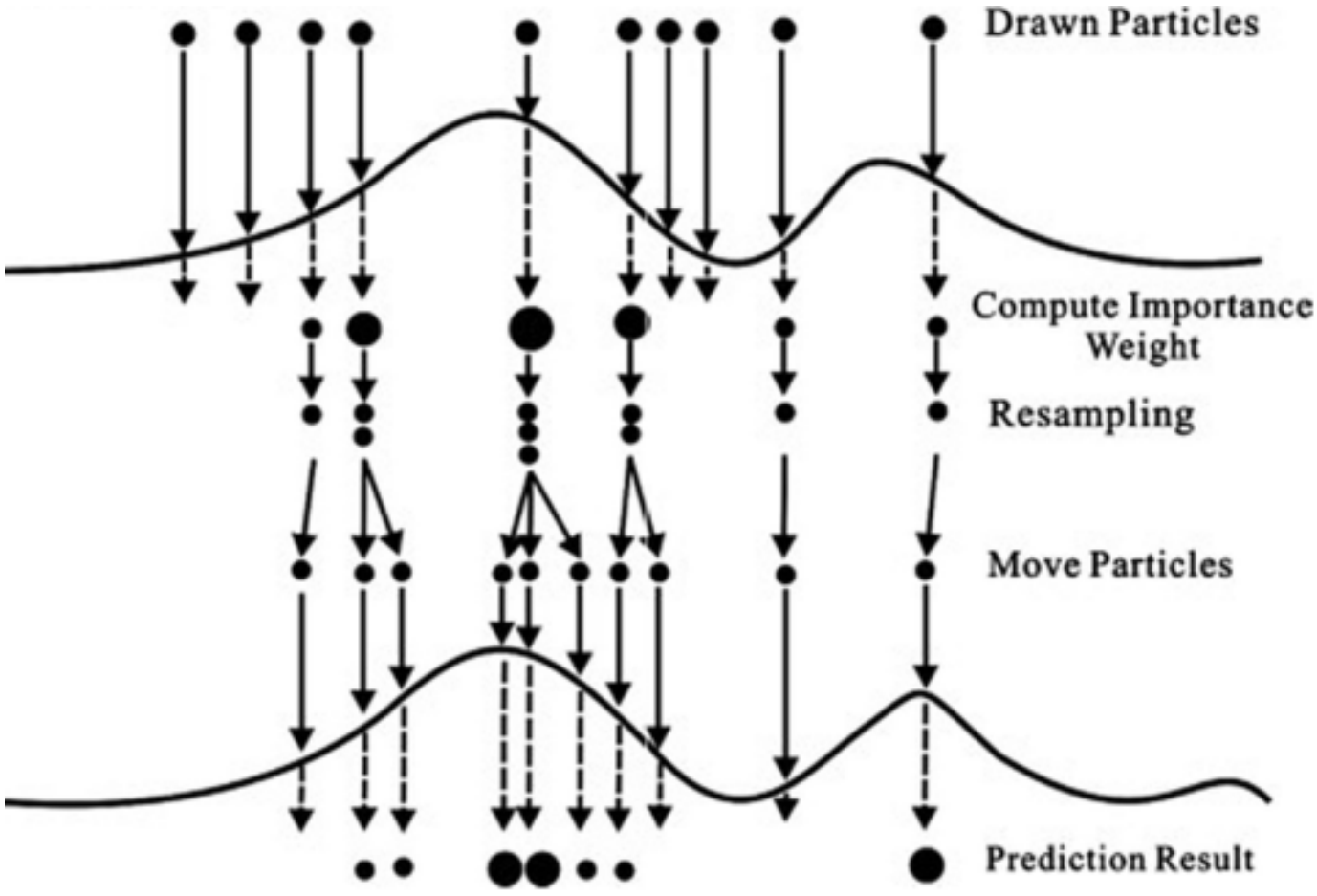
Full algorithm: Recursion

Recursion

For steps/time points $t = 1, \ldots, T$ and for particles $i = 1, \ldots, N$:

1. Resample: sample $\tilde{\theta}_{t-1}^{(i)}$ from $\theta_{t-1}^{(i)}$ with weights $W_{t-1}^{(i)}$.
2. Transition: simulate $\theta_t^{(i)} \sim p(\theta_t \mid \tilde{\theta}_{t-1}^{(i)})$.
3. Calculate weights: $w_t^{(i)} \propto p(y_t \mid \theta_t^{(i)})$ and get normalized weights $W_t^{(i)}$.

Particle filter algorithm



https://umbertopicchini.wordpress.com/2016/10/19/sequential-monte-carlo-bootstrap-filter/

Final step: inference

We now have N values of θ_t for each time point. Inference is straightforward. For example, we may be interested in inference on the posterior mean, $\mu = E(\theta_t \mid y_{1:t})$

We can use either re-sampled or weighted non-resampled values.

- $\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} \tilde{x}_t^{(i)}$
- $\hat{\mu} = \sum_{i=1}^{N} W_t^i x_t^{(i)}$

The weighted estimator is generally preferred, as the resampling step immediately injects additional randomness into the estimator, and thus can result in higher variance.

Terminology

The term particle filter is sometimes used interchangeably with sequential Monte Carlo (SMC).

- SMC methods are a general class of Monte Carlo methods that sample sequentially from a sequence of target probability densities $\{p_t(\theta_{1:t})\}$ of increasing dimension.
- Particle filters are a special case of SMC methods.
- They are also the most common use, and thus they are often conflated.
- However, SMC also includes other settings, and some filtering methods do not rely on a sequence of target distributions.

Extensions and modifications

SMC/Particle filter research

I have presented a vanilla version of SMC. SMC is a very active area of research! Some broad categories of modifications:

- Increase efficiency (lower variance)
- Specialized for certain domains
- Faster
- Resolve path degeneracy (explanation to follow)
- Algorithms for smoothing
- Algorithms for parameter estimation (particularly hard)
- Relaxing assumptions (such as Markov assumption)
- Many more!

Adaptive resampling

One way to reduce variance...

- Resampling removes particles with low weights, multiplies particles with high weights
- However, at the cost of immediately introducing more variance!
- Do we always want to resample?
- Proposal: choose a criteria, only resample if criteria is not met.

Effective sample size

Effective Sample Size (ESS)

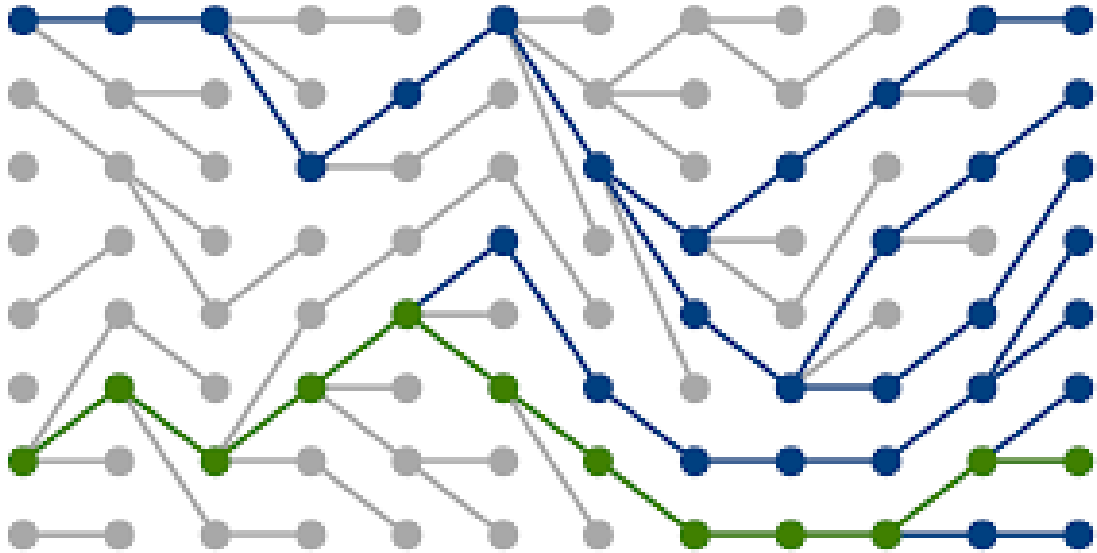
$$ESS_t = \left(\sum_{i=1}^{N} (W_t^i)^2 \right)^{-1}$$

Special cases:

- If a weight is zero, that particle is not contributing to the sampler.
- If all weights are $1/N$, $ESS_t = \left(\sum_{i=1}^{N} (W_t^i)^2 \right)^{-1} = N$

Resampling criteria: Only resample if $ESS < N/2$

Particle filter paths



The sequence of values that a particle takes over time can be called a path or trajectory.

Path degeneracy

Particle filter challenge: the *effective* number of particles may be smaller than the actual number of particles.

- Path degeneracy occurs when the algorithm degenerates into only exploring a small number of paths.
- Also called sample impoverishment: effective number of particles decreases over time.

Path Degeneracy

- Only exploring a small part of the model space.
- Particularly likely with high-dimensional spaces, or as length of time T increases.
- Inference is no longer reliable due to small effective sample size.
- Convergence to the right answer is no longer guaranteed!

Solving path degeneracy

- Resample-move methods uses an MCMC kernels to "jitter" the particle locations and thus to reduce degeneracy
- Block sampling: rather than sampling based on the just the current value θ_t, can sample based on (part of) the previous *path*
- Rao-Blackwellised (Marginalised) Particle Filters: if we have nice distributions (Normal), we can marginalize some components of the state process because they are analytically tractable, and only conduct filter on a lower-dimensional space.
- Use other resampling strategies (besides multinomial): residual, stratified, systematic resampling.

Particle Gibbs

Idea: add a reference trajectory.

- Run a complete "vanilla" SMC algorithm with N particles (for all time points).
- A single trajectory is sampled from these N trajectories using multinomial sampling with weights proportional to the likelihood of the observed data for each trajectory.

Particle Gibbs

Now, repeat the whole SMC procedure!

- At each time point, fix the last particle to be the reference trajectory.
- It will never be resampled away.
- Idea: reference trajectory guides the particles to a relevant region of the space.
- Can show that the reference trajectory results in valid draws from target distribution, regardless of number of particles!
- Problem: prone to path degeneracy.

Particle Gibbs with ancestor sampling (PGAS)

- When particles are resampled, this induces an ancestor-offpsring relationship.
- Instead of sampling particles, we can think about sampling ancestors.
- Can also resample the ancestor of the reference trajectory.
- Weight is proportional to the transition density $p(\theta_t \mid \theta_{t-1})$.
- Can show that we still successfully draw from target distribution.
- Substantial improvement in path degeneracy!

Bonus: Parameter estimation with PGAS

Up until now, have assumed model parameters are known. If we wanted to conduct inference on some model parameter α **and** $\theta_{1:T}$:

Parameter estimation algorithm

- Set arbitrary $\alpha[0]$ and $\theta_{1:T}[0]$

For $s \geq 1$:

- Draw $\theta_{1:T}[s]$ conditional on $\alpha[s-1]$ using PGAS algorithm (with $\theta_{1:T}[s-1]$ as reference trajectory)
- Draw $\alpha[s] \sim p(\alpha \mid \theta_{1:T}[s], y_{1:T})$

Gibbs sampler, but with entire SMC algorithm in the middle!

Summary: SMC/Particle Filters

What is sequential Monte Carlo?

- SMC is a sequential algorithm for sampling from a target distribution.
- Particularly suited to sequential Bayesian inference for nonlinear, non-Gaussian state-space models.
- When tackling the problem of filtering for a state-space model, synonymous with particle filters.

Summary: SMC/Particle Filters

Comparison to MCMC

- Easily parallelizable (compared to MCMC for example): except for normalization of weights, all operations only consider one particle at a time.
- In some settings, better properties for high dimensional, irregular, or multimodal settings.
- Sequential nature is tailored to settings with changing/updating information.

Summary: SMC/Particle Filters

Takeaways

- Almost all flavors can be seen as a combination of sampling and resampling steps.
- In addition to filtering, also suited to a wide variety of other Bayesian applications!
- Both strong theory and empirical evidence that these algorithms have good properties.

Want to see an example SMC in practice?
Hunter, Glickman, Campos (2022) Inferring medication adherence from time-varying health measures.
`https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.9351`

References I

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342, 2010.

J. K. Blitzstein and C. N. Morris. Probability for statistical science (draft textbook). Chapter 8.

J. Bulla. Application of hidden markov models and hidden semi-markov models to financial time series. *University Library of Munich, Germany, MPRA Paper*, 01 2006.

N. Chopin and O. Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer Nature, Switzerland, 2020.

A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. `https://www.stats.ox.ac.uk/~doucet/doucet_johansen_tutorialPF2011.pdf`.

References II

P. Fearnhead. Modern computational statistics: Alternatives to mcmc. https://www.maths.lancs.ac.uk/~fearnhea/GTP/GTP_Practicals.pdf.

N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin. On particle methods for parameters estimation in state-space models. *Statistical Science*, 30(3):328–351, 2015.

F. Lindsten, M. I. Jordan, and T. B. Schön. Particle gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15:2145–2184, 2014.

T. Rothenberg. State space models and the kalman filter. https://eml.berkeley.edu/~rothenbe/Fall2007/kalman.pdf, 2007.