Sequential Monte Carlo tings

Shannon Williams

May 19, 2021

1 State-space models

A *state-space model* is a time series model consisting of two discrete time processes $\{X_t\} \coloneqq (X_t)_{t\geq 0}$ and $\{Y_t\} \coloneqq (Y_t)_{t\geq 0}$ taking values in $\mathcal{X}$ and $\mathcal{Y}$, respectively. The model is specified via a parameter vector $\theta \in \Theta$ and a set of densities defining the joint density of the processes:

$$p_0^\theta(x_0) \prod_{t=0}^{T} f_t^\theta\left(y_t \middle| x_t\right) \prod_{t=1}^{T} p_t^\theta\left(x_t \middle| x_{t-1}\right).$$

This describes a generative probabilistic model: X_0 is drawn according to the initial density $p_0^\theta(x_0)$, each X_t is then drawn conditionally on the $X_{t-1} = x_{t=1}$ according to $p_t^\theta\left(x_t \middle| x_{t-1}\right)$, and then each Y_t is drawn conditionally on $X_t = x_t$ according to f_t^θ.

An informal definition of a state-space model is as a Markov chain observed with noise.

1.1 Sequential analysis

Process $\{Y_t\}$ is observed while $\{X_t\}$ is not, and the objective is to derive the distribution of certain X_t's conditional on certain components of $\{Y_t\}$'s. Traditionally this is done without taking into account the parameter uncertainty (θ is assumed to be known and fixed).

Sequential analysis refers to drawing inference on the process X_t sequentially in time at each t, given realised observations $y_{0:t}$ of $Y_{0:t}$ collected up to t. *Filtering* refers to the task of deriving the distribution of X_t conditional on $Y_{0:t} = y_{0:t}$, $t = 0, 1, \ldots$.

1.2 Stochastic volatility models in finance

We are interested in modelling the volatility of log-returns,

$$Y_t = \log\left(\frac{p_t}{p_{t-1}}\right),$$

where p_t is the price of an asset at time t. A standard approach is ARCH (auto-regressive conditional heteroscedastic) or GARCH (generalised ARCH) modelling, where the variance of Y_t is a deterministic function of past data $Y_{0:t-1}$. A different approach takes the volatility to be a stochastic process.

A basic *stochastic volatility model* is

$$Y_t | X_t = x_t \sim \mathcal{N}\left(0, \exp(x_t)\right),$$

where $\{X_t\}$ is an auto-regressive process (define AR processes here?):

$$X_t - \mu = \rho(X_{t-1} - \mu) + U_t,$$

where $U_t \sim \mathcal{N}\left(0, \sigma^2\right)$ and $\theta \coloneqq \left(\mu, \rho, \sigma^2\right)$. Imposing $|\rho| < 1$ ensures that $\{X_t\}$ is a stationary processes. In practice we might expect $\rho \approxeq 1$ as financial data often exhibit volatility clustering (volatility remains high or low for long periods of time).

Variants of the stochastic volatility model:

- $Y_t|X_t$ may have heavier-than-Gaussian tails, e.g. follow a student distribution.
- Account for a leverage effect by assuming the Guassian noises of $Y_t|X_t$ and $X_t|X_{t-1}$ are correlated.
- Introduce skewness by taking $Y_t = \alpha X_t + \exp(X_t/2)V_t$.

The volatility might be assumed to be an AR process of order $k > 1$. In which case, we can retain the basic structure of a state-space model by increasing the dimension of $\{X_t\}$. If $\mathcal{X} = \mathbb{R}^2$ then

$$X_t - \begin{pmatrix}\mu \\ \mu\end{pmatrix} = \begin{pmatrix}\rho_1 & \rho_2 \\ 1 & 0\end{pmatrix}\left(X_{t-1} - \begin{pmatrix}\mu \\ \mu\end{pmatrix}) + \begin{pmatrix}U_t \\ 0\end{pmatrix}\right),$$

where $U_t \sim \mathcal{N}\left(0, \sigma^2\right)$. Then $X_t(1)$ is an AR process of order 2,

$$X_t(1) - \mu = \rho_1(X_{t-1}(1) - \mu) + \rho_2(X_{t-2}(1) - \mu) + U_t,$$

and we can take $\mathrm{Var}(Y_t|X_t) = \exp\left(X_t(1)\right)$.

2 Feynman-Kac models

Start with a *Markov probability law* defined on a state space $\mathcal{X}$ with initial distribution $\mathbb{M}_)$ and transition kernels $M_{1:T}$:

$$\mathbb{M}_T(dx_{0:T}) = \mathbb{M}_0(dx_0)\prod_{t=1}^{T} M_t(x_{t-1}, dx_t).$$

Consider a sequence of *potential functions* $G_0 : \mathcal{X} \to \mathbb{R}^+$ and $G_t : \mathcal{X}^2 \to \mathbb{R}^+$, $t = 1, \ldots, T$. For $t = 0, \ldots, T$, a sequence of Feynman-Kac models is given by probability measures on $\left(\mathcal{X}^{t+1}, \mathcal{B}(\mathcal{X})^{t+1}\right)$, obtained as the following changes of measure from $\mathbb{M}_t$:

$$\mathbb{Q}_t(dx_{0:t}) \coloneqq \frac{1}{L_t} G_0(x_0)\left\{\prod_{s=1}^{t} G_s(x_{s-1}, x_s)\right\}\mathbb{M}_t(dx_{0:t}),$$

where L_t is the normalising constant needed for $\mathbb{Q}_t$ to be a probability measure.

We refer to T, $\mathbb{M}_0$, G_0 and $M_t(x_{t-1}, dx_t)$, $G_t(x_{t-1}, x_t)$, $t = 1, \ldots, T$ as the components of the Feynman-Kac model.

2.1 Feynman-Kac formalisms of a state-space model

Consider a state-space model with initial distribution $\mathbb{P}_0(dx_0)$, signal transition kernels $P_t(x_{t-1}, dx_t)$, and observation densities $f_t(y_t|x_t)$. We define its *bootstrap Feynman-Kac formalism* to be the Feynman-Kac model with the components:

$$\mathbb{M}_0(dx_0) = \mathbb{P}_0(dx_0), \qquad G_0(x_0) = f_0(y_0|x_0), \tag{1}$$
$$M_t(x_{t-1}, dx_t) = P_t(x_{t-1}, dx_t), \quad G_t(x_{t-1}, x_t) = f_t(y_t|x_t). \tag{2}$$

Then

$$\mathbb{Q}_{t-1}(dx_{0:t}) = \mathbb{P}_t\left(X_{0:t} \in dx_{0:t} \,|Y_{0:t-1} = y_{0:t-1}\right), \tag{3}$$
$$\mathbb{Q}_t(dx_{0:t}) = \mathbb{P}_t\left(X_{0:t} \in dx_{0:t} \,|Y_{0:t} = y_{0:t}\right), \tag{4}$$
$$L_t = p_t(Y_{0:t}), \tag{5}$$
$$l_t \coloneqq \frac{L_t}{L_{t-1}} = p_t\left(y_t|\, y_{0:t-1}\right). \tag{6}$$

The potential functions G_t for $t \geq 1$ depend only on x_t, and G_t depends implicitly on the (fixed) datapoint y_t.

3 Resampling

Resampling: the action of drawing randomly from a weighted sample, so as to obtain an unweighted sample. We can view resampling as a random weight importance sampling technique.

Suppose we have the following particle approximation of measure $\mathbb{Q}_0(dx_0)$:

$$\mathbb{Q}_0^N(dx_0) = \sum_{n=1}^N W_0^N \delta_{X_0^n}, \qquad , X_0^n \sim \mathbb{M}_0, \qquad W_0^n = \frac{w_0(X_0^n)}{\sum_{m=1}^N w_0(X_0^m)},$$

obtained through importance sampling based on the proposal $\mathbb{M}_0$ and weight function $w_0 \propto d\mathbb{Q}_0/d\mathbb{M}_0$. We want to use this to approximate the extended probability measure

$$(\mathbb{Q}_0 M_1)(dx_{0:1}) = \mathbb{Q}_0(dx_0) M_1(x_0, dx_1). \tag{7}$$

Importance resampling uses a two-step approximation: first replace $\mathbb{Q}_0$ by $\mathbb{Q}_0^N$ in (7),

$$\mathbb{Q}_0^N(dx_0) M_1(x_0, dx_1) = \sum_{n=1}^N W_0^n M_1(X_0^n, dx_1) \delta_{X_0^n}(dx_0), \tag{8}$$

and then sample N times from this intermediate approximation to form

$$\frac{1}{N}\sum_{n=1}^N \delta_{\tilde{X}_{0:1}^n}, \qquad \tilde{X}_{0:1}^n \sim \mathbb{Q}_0^N(dx_0) M_1(x_0, dx_1).$$

The simplest way to sample from (8): for $n = 1, \ldots, N$, sample independently the pairs $(A_1^n, \tilde{X}_{0:1}^n)$ as

$$A_1^n \sim \mathcal{M}(W_0^{1:N}), \qquad \tilde{X}_{0:1}^N = \left(X_0^{A_1^n}, X_1^n\right), \qquad X_1^n \sim M_1\left(X_0^{A_1^n}, dx_1\right),$$

where $\mathcal{M}(W_0^{1:N})$ is the multinomial distribution that generates value n with probability W_0^n, for $n = 1, \ldots, N$.

3.1 Multinomial resampling

Multinomial resampling describes an efficient algorithm for simulating the *ancestor* variables A^n from the multinomial distribution $\mathcal{M}(W^{1:N})$.

Use the inverse CDF transformation method: generate N uniform variates U^m, $m = 1, \ldots, N$, and set A^m according to

$$C^{n-1} \leq U^m \leq C^n \quad \Longleftrightarrow \quad A^m = n,$$

where the C^n's are the cumulative weights:

$$C^0 = 0, \quad C^n = \sum_{l=1}^{n} W^l, \quad n = 1, \ldots, N.$$

Each individual simulation requires $\mathcal{O}(N)$ comparisons to be made, so $\mathcal{O}\left(N^2\right)$ comparisons should be performed to generate N draws. However, if the U^m are ordered in a preliminary step we obtain an algorithm with $\mathcal{O}(N)$ complexity.

Algorithm 1: Computing the inverse of the multinomial CDF $x \to \sum_{n=1}^{N} W^n \mathbb{I}\{n \leq x\}$.

Input: Normalised weights $W^{1:N}$ and ordered uniform points $0 < U^{(1)} < \cdots < U^{(N)} < 1$.
Output: Ordered ancestor variables $1 \leq A^1 \leq \cdots \leq A^N \leq N$ in {1, . . . , N}.
Initialise $s \leftarrow W^1$, $m \leftarrow 1$.
for $n = 1, \ldots, N$ **do**
  while $s < U^{(n)}$ **do**
    $m \leftarrow m + 1$.
    $s \leftarrow s + W^m$.
  end
  $A^n \leftarrow m$.
end

Note the A^n are not i.i.d. draws from $\mathcal{M}(W^{1:N})$ but correspond to the order statistics of a vector of N draws from this distribution.

To generate the *uniform spacings* $0 < U^{(1)} < \cdots < U^{(N)} < 1$, we use an $\mathcal{O}(N)$ algorithm.

The overall approach for multinomial resampling is then:

Can be interpreted as a random weight importance technique with random weight $\mathcal{W}^n$ such that

- $\mathcal{W}^n$ is integer-valued and represents the number of offsprings of particle n;
- $\mathbb{E}[W^n] = NW^n$;

Algorithm 2: Generation of uniform spacings.

Input: An integer N
. **Output:** An ordered sequence $0 < U^{(1)} < \cdots < U^{(N)} < 1$ in $(0, 1)$.
Initisalise $S^0 \leftarrow 0$.
for *n = 1, . . . , N+1* **do**
  Draw $E^n \sim \mathcal{E}(1)$.
  $S^n \leftarrow S^{n-1} + E^n$.
end
for *n = 1, . . . , N* **do**
  $U^{(n)} \leftarrow S^n / S^{N+1}$.
end

Algorithm 3: Multinomial resampling.

Input: Normalised weights $W^{1:N}$ such that $\sum_{n=1}^N W^n = 1$ and $W^n \geq 0$, $n = 1, \ldots, N$.
Output: N draws from $\mathcal{M}(W^{1:n})$.
Obtain $U^{(1:N)}$ via Algorithm 2.
Obtain $A^{1:N}$ via Algorithm 1.

- $\sum_{n=1}^N \mathcal{W}^n = N$.

Any method that generates ancestor variables $A^{1:N}$ such that the number of copies $\mathcal{W}^n = \sum_{m=1}^n \mathbb{I}\{A^m = n\}$ fulfilles the above three properites will be an *unbiased resampling scheme*, satisfying

$$\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \varphi\left(X^{A^n}\right) \middle| X^{1:N}\right] = \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \mathcal{W}^n \varphi(X^n) \middle| X^{1:N}\right] = \sum_{n=1}^N W^n \varphi(X^n).$$

That is, it generates an unweighted sample that provides estimates with the same expectation (but increased variance) as the original weighted sample.

3.2 Systematic resampling

A strategy for variance reduction is to replace the i.i.d. uniforms by values that cover $[0, 1]$ more regularly. A *systematic resampling approach* is similar to Algorithm 1, specialised to the structure of the $U^{(n)}$'s. We simulate a single uniform random variable $U \sim \mathcal{U}([0, 1])$ and take $U^{(n)} = (n - 1 + U)/N$.

Algorithm 4: Systematic resampling approach for obtaining ancestor variables $A^{1:N}$.

Input: Normalised weights $W^{1:N}$ and $U \in [0, 1]$.
Output: N random indices $A^{1:N}$ taking values in $1 : N$.
Initialise $s \leftarrow U$ and $m \leftarrow 1$.
Compute the cumulative weights $v^n := \sum_{m=1}^n NW^m$, $n = 1, \ldots, N$.
for $n = 1, \ldots, N$ **do**
  while $v^m < s$ **do**
    Update $m \leftarrow m + 1$, $A^n \leftarrow m$, $s \leftarrow s + 1$.
  end
end

This is unbiased: $\mathbb{E}[\mathcal{W}^n] = NW^n$.

Compared to other variance reduction schemes (not discussed here), systematic resampling is often recommended in practice as it is fast and tends to (empirically) work better than other schemes in that it yields lower-variance estimates.

4 The bootstrap filter for state-space models

Consider a state space model $\{(X_t, Y_t)\}$ with initial law $\mathbb{P}_0$, transition kenels $P_t(x_{t-1}, dx_t)$ and observation densities $f_t(y_t|\, x_t)$. Consider the basic Feynman-Kac representation of the model (1), (2), for $t \geq 1$.

Algorithm 5: The bootstrap filter for state-space models with adaptive resampling (resampling only when the ESS is too low – reduces computational time of the algorithm).

Input: A state-space model such that we can simulate from $\mathbb{P}_0(dx_0)$ and from $P_t(x_{t-1}, dx_t)$ for each x_{t-1} and t, and compute the function $f_t(y_t|\, x_t)$ point-wise, a number of particles N, a choice of a resampling scheme, and an ESS threshold $\mathrm{ESS}_{\min}$.

Output: $\hat{w}_0^n, \ldots, \hat{w}_{T-1}^n$, $X_1^n, \ldots, X_T^n$, $w_0^n, \ldots, w_T^n$, $W_0^n, \ldots, W_T^n$, for $n = 1, \ldots, N$.

All operations involving index n must be performed for $n = 1, \ldots, N$.

Simulate $X_0^n \sim \mathbb{M}_0(dx_0)$.

$w_0^n \leftarrow G_0(X_0^n)$.

$W_0^n \leftarrow w_0^n / \sum_{m=1}^M w_0^m$.

for $t = 1, \ldots, T$ **do**

  if $\mathrm{ESS}(W_{t-1}^{1:N}) < \mathrm{ESS}_{\min}$ **then**

    Resample $A_t^{1:N}$ with weights $W_{t-1}^{1:N}$.

    $\hat{w}_{t-1}^n \leftarrow 1$.

  end

  else

    $A_n^t \leftarrow n$.

    $\hat{w}_{t-1}^n \leftarrow w_{t-1}^n$.

  end

  Draw $X_t^n \sim P_t\left(X_{t-1}^{A_t^n}\middle|\, dx_t\right)$.

  $w_t^n \leftarrow \hat{w}_{t-1}^n f_t(y_t|\, X_t^n)$.

  $W_t^n \leftarrow w_t^n / \sum_{m=1}^N w_t^m$.

end

At each time $t = 1, \ldots, T$, we generate simulations from the law of the Markov chain $\{X_t\}_{t\geq 1}$, weight these simulations according to how "compatible" they are with the datapoint Y_t, and resample if necessary. We obtain the approximations:

$$\frac{1}{\sum_{n=1}^N \hat{w}_{t-1}^n} \sum_{n=1}^N \hat{w}_{t-1}^n \varphi(X_t^n) \approx \mathbb{E}\left[\varphi(X_t)\,|Y_{0:t-1} = y_{0:t-1}\right]$$

$$\sum_{n=1}^N W_t^n \varphi(X_t^n) \approx \mathbb{E}\left[\varphi(X_t)\,|Y_{0:t} = y_{0:t}\right]$$

$$\ell_t^N \approx p_t(y_t|\, y_{0:t-1})$$

$$L_t^N = \prod_{s=1}^t \ell_s^N \approx p(y_{0:t})$$

where

$$\ell_t^N = \begin{cases} \frac{1}{N}\sum_{n=1}^N w_t^n & \text{if resampling occurred at time } t, \\ \frac{\sum_{n=1}^N w_t^n}{\sum_{n=1}^N w_{t-1}^n} & \text{otherwise} \end{cases}$$

Note $\approx$ means the LHS approximates the RHS: that is, the approximation error tends to zero as $N \to \infty$ with rate $\mathcal{O}\left(N^{-\frac{1}{2}}\right)$ (the standard Monte Carlo rate).

The bootstrap filter is very simple and widely applicable with few requirements. However, if samples particles X_t "blindly" from $P_t(x_{t-1}, dx_t)$ without any guarantee that these simulated particles will be compatible with the datapoint y_t (i.e. have non-negligible weights).