Particle Learning for General Mixtures

Carlos M. Carvalho\textsuperscript{1}, Hedibert F. Lopes\textsuperscript{2}, Nicholas G. Polson\textsuperscript{2} and Matt A. Taddy \textsuperscript{2}

\textsuperscript{1}UT-Austin, \textsuperscript{2}University of Chicago
Presenter: Miao Liu

August 19, 2011
A general state-space model

\[ x_{t+1} = g(x_t, \theta) \]
\[ y_{t+1} = f(x_{t+1}, \theta) \]

The sequential inference of states and parameters involves computing the joint posterior distributions, \( p(x_s, \cdots x_{s'}, \theta | y^t) \), where \( y^t = (y_1, \cdots, y_t) \) is the set of observations up to time \( t \).

Specific problem includes filtering (when \( s = s' = t \)), smoothing (if \( s = 0 \) and \( s' = t \))

\( p(x_s, \cdots x_{s'}, \theta | y^t) \) can be computed in closed form only in very specific cases including linear Gaussian model and discrete hidden Markov Model.

For nonlinear dynamic systems the extend Kalman filter (EKF) and its variant can be employed, but the performance may be very poor.

Sequential Monte Carlo (particle) methods use a discrete representation of \( p(x_t, \theta | y^t) \) via

\[ p^N(x_t, \theta | y^t) = \frac{1}{N} \sum_{i=1}^N \delta^{(i)}(x_t, \theta) \]
Particle Filtering

The filtering density we are interested in is

\[ p(x_{t+1} \mid y^{t+1}) \propto p(y_{t+1} \mid x_{t+1})p(x_{t+1} \mid y^t) \]

where the state predictive is

\[ p(x_{t+1} \mid y^t) = \int p(x_{t+1} \mid x_t) dP(x_t \mid y^t) \]

Importance Sampling (IS)

- Assume you have at time \( t \)

  \[ \hat{p}(x_t \mid y^t) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_t^{(i)}}(x_t) \]

- By sampling \( \tilde{x}_{t+1}^{(i)} \sim p(x_{t+1} \mid x_t^{(i)}) \), then

  \[ \hat{p}(x_{t+1} \mid y^t) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\tilde{x}_t^{(i)}}(x_{t+1}) \]

- Our target at time \( t + 1 \) is

  \[ p(x_{t+1} \mid y^{t+1}) \propto p(y_{t+1} \mid x_{t+1})p(x_{t+1} \mid y^t) \]

  so by substituting \( \hat{p}(x_{t+1} \mid y^t) \) to \( p(x_{t+1} \mid y^t) \) we obtain

  \[ \tilde{p}(x_{t+1} \mid y^{t+1}) = \sum_{i=1}^{N} \omega_n^{(i)} \delta_{\tilde{x}_{t+1}^{(i)}}(x_{t+1}), \omega_n^{(i)} \propto p(y_{t+1} \mid \tilde{x}_{t+1}^{(i)}) \]
Resampling

- Through IS, we obtain a "weighted" approximation \( \tilde{p}(x_{t+1} \mid y^{t+1}) \) of \( p(x_{t+1} \mid y^{t+1}) \). This approximation is based on weighted samples from \( p(x_{t+1} \mid x_t) \).
- To obtain \( N \) samples \( x^{(i)}_{t+1} \) approximated distributed according to \( P(x_{t+1} \mid y^{t+1}) \), we just resample \( x^{(i)}_{t+1} \sim \tilde{p}(x_{t+1} \mid y^{t+1}) \) to obtain \( \hat{p}(x_{t+1} \mid y^{t+1}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x^{(i)}_{t+1}}(x_{t+1}) \)
- Particles with high weights are copied multiple times, particles with low weights die.
- In practice resampling is performed when the variance of the weights is large than a pre-specified threshold.
- BF is called "propagate-resample filter".

**BOOTSTRAP FILTER (BF)[Gordon, Salmond and Smith (1993)]**

1. Propagate. \( \{x^{(i)}_t\}_{i=1}^{N} \) to \( \{\tilde{x}^{(i)}_{t+1}\}_{i=1}^{N} \) via \( p(x_{t+1} \mid x_t) \)
2. Resample. \( \{x^{(i)}_{t+1}\}_{i=1}^{N} \) from \( \{\tilde{x}^{(i)}_{t+1}\}_{i=1}^{N} \) with weights \( \omega_{t+1} \propto p(y_{t+1} \mid \tilde{x}^{(i)}_{t+1}) \)
Auxiliary particle filter (APF) [Pitt and Shephard (1999)]

1. **Resample.** \( \{\tilde{x}_t^{(i)}\}_{i=1}^N \) from \( \{x_t^{(i)}\}_{i=1}^N \) weights \( \tilde{\omega}_t^{(i)} \propto p(y_{t+1} \mid g(x_t^{(i)})) \)

2. **Propagate.** \( \{\tilde{x}_t^{(i)}\}_{i=1}^N \) to \( \{\tilde{x}_{t+1}^{(i)}\}_{i=1}^N \) via \( p(x_{t+1} \mid \tilde{x}_t) \)

3. **Resample.** \( \{\tilde{x}_{t+1}^{(i)}\}_{i=1}^N \) with weights \( \omega_{t+1}^{(i)} \propto p(y_{t+1} \mid \tilde{x}_t^{(i)}) / p(y_{t+1} \mid g(\tilde{x}_t^{(i)})) \)

- The APF uses a different representation of the joint filtering distribution of \((x_t, x_{t+1})\) as

  \[
  p(x_t, x_{t+1} \mid y^{t+1}) \propto P(x_{t+1} \mid x_t, y^{t+1})p(x_t \mid y^{t+1}) \tag{1}
  \]

  \[
  = p(x_{t+1} \mid x_t, y^{t+1})p(y_{t+1} \mid x_t)p(x_t \mid y^t) \tag{2}
  \]

- APF is a "resample-propagate filter".

- Starting with a particle approximation of \( p(x_t \mid y^t) \), draws from \( p(x_t \mid y^{t+1}) \) are obtained by resampling the particles with weights proportional to \( p(y_{t+1} \mid x_t) \). These resampled particles are then propagated forward via \( p(x_{t+1} \mid x_t, y^{t+1}) \).

- The above representation is optimal, fully adaptive if both the predictive and propagation densities were available.

- Otherwise importance function \( p(y_{t+1} \mid g(x_t)) \) is proposed for the resampling steps.
The General PL Framework

- PL = APF + Parameter Estimation
- Define $Z_t$ as the "essential state vector" which may include state variables, auxiliary variables, sufficient statistics etc.
- $Z_t$ allows the computation of:
  
  (a) the posterior predictive $p(y_{t+1} \mid Z_t)$,
  
  (b) the posterior updating rule $p(Z_{t+1} \mid Z_t, y_{t+1})$
  
  (c) parameter learning via $p(\theta \mid Z_{t+1})$

Particle Learning

1. **Resample.** $\{\tilde{z}_t^{(i)}\}_{i=1}^N$ from $\{z_t^{(i)}\}_{i=1}^N$ weights $\tilde{w}_t^{(i)} \propto p(y_{t+1} \mid z_t^{(i)})$

2. **Propagate.** $\{\tilde{x}_t^{(i)}\}_{i=1}^N$ to $\{\tilde{x}_{t+1}^{(i)}\}_{i=1}^N$ via $p(x_{t+1} \mid \tilde{z}_t, y_{t+1})$ update sufficient statistics $s_{t+1}^i = S(\tilde{s}_t^{(i)}, x_{t+1}^{(i)}, y_{t+1})$

3. **Learn:** $p(\theta \mid y^{t+1}) \approx \frac{1}{N} \sum_{i=1}^N p(\theta \mid z_{t+1}^i)$
Example 1: Finite Mixture of Poisson Densities

\[ p(y_t) = \sum_{j=1}^{m} \pi_j Po(y_t; \theta_j) \]
\[ \pi \sim Dir(\gamma), \quad \theta_j \sim Ga(\alpha_j, \beta_j), \text{ for } j = 1, \ldots, m \]

- Define \( n_t \) the number of samples in each component, and sufficient states \( s_t = (s_{t,1}, \ldots, s_{t,m}) \) where \( s_{t,j} = \sum_{r=1}^{t} y_r \mathbf{1}_{\{k_r=j\}} \) for the conditional posterior given \( y^t \) and \( k^t \), we have essential state vector \( Z_t = \{k_t, s_t, n_t\} \)
- Predictive for \( y_{t+1} \)

\[
p(y_{t+1} | Z_t) = \sum_{k_{t+1}=j=1}^{m} \int \int p(y_{t+1}, k_{t+1} | \theta_{k_{t+1}}) p(\theta, \pi) d(\theta, \pi)
= \sum_{k_{t+1}=j=1}^{m} \frac{\Gamma(s_{t,j} + y_{t+1} + \alpha_j)}{\Gamma(s_{t,j} + \alpha_j)} \frac{(\beta_j + n_{t,j})^{s_{t,j}+\alpha_j}}{(\beta_j + n_{t,j} + 1)^{s_{t,j}+\alpha_j+y_{t+1}}} \frac{1}{y_{t+1}! \left(\sum_{i=1}^{m} \gamma_i + n_{t,i}\right)}
\]

- propagating \( k_{t+1} \) is done according to

\[
p(k_{t+1} = j | Z_t, y_{t+1}) \propto \frac{\Gamma(s_{t,j} + y_{t+1} + \alpha_j)}{\Gamma(s_{t,j} + \alpha_j)} \frac{(\beta_j + n_{t,j})^{s_{t,j}+\alpha_j}}{(\beta_j + n_{t,j} + 1)^{s_{t,j}+\alpha_j+y_{t+1}}} \frac{1}{y_{t+1}! \left(\sum_{i=1}^{m} \gamma_i + n_{t,i}\right)}
\]

- Given \( k_{t+1} Z_t \) is updated by the recursions \( s_{t+1,j} = s_{t,j} + \mathbb{1}_{k_{t+1}=j} \) and \( n_{t+1,j} = n_{t,j} + \mathbb{1}_{k_{t+1}=j} \), for \( j = 1, \ldots, m \).
(a) 4 mixture of Poisson

(b) MC study

Figure: Poisson Mixture Example.
Allocation: to obtain smoothed samples of $k^t$

- Draws from $p(k^t \mid y^t)$ can be obtained through the backwards update equation:

$$p(k^t \mid y^t) = \int p(k^t \mid Z_t, y^t) dP(Z_t \mid y^t) = \int \prod_{r=1}^{t} p(k_r \mid Z_r, y_r) dP(Z_t \mid y^t)$$

- We can directly approximate the $p(k^t \mid y^t)$ by sampling for each particle $Z_t^{(i)}$ with probability $p(k_r \mid Z_t, y_r) \propto p(y_r \mid k_r = j, Z_t)p(k_r = j \mid Z_t)$

- The conditional independence of the mixture models leads to the factorization provides an algorithm for posterior allocation that is of order $O(N)$

Marginal Likelihood

- Useful in Bayesian model comparison procedures, based either on Bayes factors or posterior model probabilities.
- Sequential marginal likelihood holds that $p(y^t) = \prod_{r=1}^{t} p(y_r \mid y^{r-1})$. The factors of this product can be estimated at each PL step as

$$p(y_t \mid y^{t-1}) = \int p(y_t \mid Z_{t-1}) dP(Z_{t-1}) \approx \frac{1}{N} \sum_{i=1}^{N} p(y_t \mid Z_{t-1}^{(i)})$$

- Given each new observation $y_t$, $p(y^t) = p(y^{t-1}) \sum_{i=1}^{N} p(y_t \mid Z_{t-1}^{(i)}) / N$
PL for Nonparametric mixture models

- Consider DP mixture model
  \[ y_t \sim k(y_t; \theta_{x_t}), x_t \sim G \text{ and } G \sim DP(\alpha, G_0) \] (3)

- Define \( n_t = (n_{t,1}, \cdots, n_{t,m_t}) \), the number of observations allocated to each unique component, \( s_t = (s_{t,1}, \cdots s_{t,m_t}) \) the conditional sufficient statistics, where \( m_t \) is the number of mixture component; \( \mathcal{Z}_t = (x_t, m_t, s_t, n_t) \) the state vector to be traced

- the uncertainty updating equation
  \[
p(s_{t+1}, n_{t+1} | y^{t+1}) \propto \int p(s_{t+1}, n_{t+1} | s_t, n_t, m_t, y_{t+1}) \]
  \[
p(y_{t+1} | s_t, n_t, m_t) dP(s_t, n_t, m_t | y^t),
\]
  where
  \[
p(y_{t+1} | n_t, s_t) = \int \int \int k(y_{t+1}; \theta)dG(\theta)dP(G|\theta^*_t, n_t)dP(\theta^*_t | n_t, s_t)
  \]
  \[
  = \int k(y_{t+1}; \theta) \left[ \int dG^t_0(\theta; \theta^*_t, n_t)dP(\theta^*_t | n_t, s_t) \right]
  \]
  \[
p(s_{t+1}, n_{t+1}, m_{t+1} | s_t, n_t, m_t, y_{t+1}) = p(x_{t+1} | s_t, n_t, m_t, y_{t+1})
  \]
  \[
  \propto \begin{cases} 
  n_{t,x_t+1} \int k(y_{t+1}; \theta^*_j)dP(\theta^*_j | s_{t,x_t+1}, n_{t,x_t+1}) & \text{for } x_{t+1} = 1, \cdots, m_t \\
  \alpha \int k(y_{t+1}; \theta)dG_0(\theta) & \text{if } x_{t+1} = m_t + 1
  \end{cases}
  \]
PL for DP mixture models

Given a set of particles \( \{ (n_t, s_t, m_t)^{(i)} \}_{i=1}^{N} \) which approximate \( p(n_t, s_t, m_t | y^t) \) and a new observation \( y_{t+1} \), the PL algorithm for DP mixture models updates the approximation to \( p(n_{t+1}, s_{t+1} | y^{t+1}) \) using the following algorithm:

**Algorithm 2: PL for DP mixture models**

1. **Resample:** Generate an index \( \xi \sim MN(\omega, N) \) where

   \[
   \omega(i) = \frac{p(y_{t+1} | (s_t, n_t, m_t)^{(i)})}{\sum_{i=1}^{N} p(y_{t+1} | (s_t, n_t, m_t)^{(i)})}
   \]

2. **Propagate:**

   \[
   k_{t+1} \sim p(x_{t+1} | (s_t, n_t, m_t)^{\xi(i)}, y_{t+1})
   \]

   \[ s_{t+1} = S(s_t, k_{t+1}, y_{t+1}). \text{ For } j \neq x_{t+1}, n_{t+1,j} = n_t,j. \]

   If \( k_{t+1} \leq m_t, n_{t+1,x_t} = n_t,x_t + 1 \) and \( m_{t+1} = m_t. \)

   Otherwise, \( m_{t+1} = m_t + 1 \) and \( n_t,m_{t+1} = 1; \)

3. **Estimate:**

   \[
   p(\mathbb{E}[f(y; G)] | y^t) = \frac{1}{N} p(y | (s_t, n_t, m_t)^{(i)})
   \]
Example 2: DP mixture of Multivariate Normals

\[
f(y_t; G) = \int \mathcal{N}(y_t | \mu_t, \Sigma_t) dG(\mu_t, \Sigma_t)
\]
\[
G \sim DP(\alpha, G_0(\mu, \Sigma))
\]

with \( G_0 = \mathcal{N}(\mu; \lambda, \Sigma/\kappa)\mathcal{W}(\Sigma^{-1}; \nu, \Omega) \)

- \( \mathcal{Z}_t = \{k_t, s_t, n_t, m_t\} \), where \( s_t \) is the conditional sufficient statistics for each unique mixture component, it is defined by
  \[
  \bar{y}_{t,j} = \sum_{r:k_r=j} y_r/n_{t,j} \quad \text{and} \quad S_{t,j} = \sum_{r:k_r=j} (y_r - \bar{y}_{t,j})(y_r - \bar{y}_{t,j})'
  \]
- The predictive density for sampling is
  \[
p(y_{t+1} | \mathcal{Z}_t) = \frac{\alpha}{\alpha + t} St(y_{t+1}; a_0, B_0, c_0) + \sum_{j=1}^{m_t} \frac{n_{t,j}}{\alpha + t} St(y_{t+1}; a_{t,j}, B_{t,j}, c_{t,j})
  \]
where the Student's t distribution are parametrized by
  \[
a_0 = \lambda, \quad B_0 = \frac{2(\kappa+1)}{\kappa c_0}, \quad c_0 = 2\nu - d + 1, \quad a_{t,j} = \frac{\kappa \lambda + n_{t,j} \bar{y}_{t,j}}{\kappa + n_{t,j}}, ...
  \]
- Propagate \( k_{t+1} \) according to
  \[
  \text{for } j = 1, \ldots, m_t, \quad p(k_{t+1} = j) \propto \frac{n_{t,j}}{\alpha + t} St(y_{t+1}; a_{t,j}, B_{t,j}, c_{t,j})
  \]
  \[
  \text{and } p(k_{t+1} = m_t + 1) \propto \frac{\alpha}{\alpha + t} St(y_{t+1}; a_0, B_0, c_0)
  \]
Figure 2: **DP-MVN Example.** Data and density estimates for PL fit with 1000 particles (Top) and each of ten PL fits with 500 particles (Bottom), to a random ordering of the 1000 observations of 2 dimensional data generated as in Section 4.2.
Figure 3: DP-MVN Example. Results from the simulation study in Section 4.2. The left plot shows average log posterior predictive score for validation sets of 100 observations, and the right plot shows the posterior averages for $m_T$, the total number of allocated mixture components. In each case, boxplots illustrate the distribution over ten repetitions of the algorithm. Red (light grey) boxplots correspond to PL, and the blue (dark grey) correspond to Gibbs.
Figure 4: **DP-MVN Example.** Data and marginal density estimates for the DP-MVN model fit to 12,500 observations of 25 dimensional data. The colors represent mean posterior estimates for each of ten PL fits, with 500 particles, to a random ordering of the data.
Latent Features Models via the Indian Buffet Process

Consider the infinite linear-Gaussian matrix factorization model of (Griffiths and Ghahramani (2006))

\[ Y^T = Z^T B + E \]  \hspace{1cm} (4)

\[ Y^T = \{ y'_1, y'_2, ..., y'_T \} \], where each \( y_t \in \mathbb{R}^p \), \( Z \in \{0, 1\}^{k \times T} \), \( B \in \mathbb{R}^{k \times p} \), \( E \in \mathbb{R}^{T \times p} \)

Assume \( e_{i,j} \sim \mathcal{N}(0, \sigma_e^2) \), \( b_{r,q} \sim \mathcal{N}(0, \sigma_b^2) \)

- Eqn (4) can be casted as a state-space model where each row of \( Z^T \) is a state and the transition \( p(z_{t+1} \mid Z^T) \) follows from the IBP generative process:
  \[ p(z_{t+1,n,j} \mid Z^T) = n \]

- Define the essential state vector \( \mathcal{Z}_t = (m_t, s_t, n_t) \), where \( m_t \) is the number of current latent features, \( n_t \) is the number of objects allocated to each of the currently available features, \( s_t \) is the set of conditional sufficient statistics for \( B \). Let \( m_* \) denote the number of potential new features per particle, \( m_*^{(i)} \sim \text{Po}(\alpha/t) \)

- The posterior predictive: \( p(y_{t+1} \mid \mathcal{Z}_t, m_*) \propto \sum_{z_{t+1} \in \mathcal{C}} p(y_{t+1} \mid z_{t+1}, s_t)p(z_{t+1} \mid \mathcal{Z}_t) \)

  where \( \mathcal{C} \) is defined by all \( 2^{m_t} \) possible configurations of \( z_{t+1} \) where the final \( m_* \) elements are fixed at one.

- Propagating \( Z_{t+1} \) forward: \( p(z_{t+1} \mid \mathcal{Z}_t, y_{t+1}) \propto p(y_{t+1} \mid z_{t+1}, s_t)p(z_{t+1} \mid \mathcal{Z}_t) \)
summary

- PL is essentially an auxiliary particle filter applied in settings where sufficient statistic exists.
- PL can provide on-line inference to a wide class of mixture models.
- Computation can be accelerated by utilizing parallel nature of PL.
- Caveat: Degeneracy is unavoidable: increasing the number of observations without simultaneously and exponentially increasing the number of particles necessarily leads to the degeneracy of the simulated sufficient statistic path. In the experiment of [Chopin etc. 2010], this degeneracy always occurs between 5,000 and 10,000 observations.