A tutorial on a few Monte-Carlo Inference methods

Richard Yi Da Xu

University of Technology, Sydney

May 8, 2014

Some non-sampling solutions

In some applications, we are interested in obtaining the "best estimate" of the parameters for posterior distribution, i.e., Maximum a Posteriori (MAP):

$$\arg\max_{\theta} \log[p(X|\theta)p(\theta)] \tag{1}$$

Something about MAP

If lucky, can find $\arg\max_\theta \log[p(X|\theta)p(\theta)]$ analytically
When not, use numerical methods, such as Expectation-Maximization (EM) (**a separate talk** on why E-M converges)

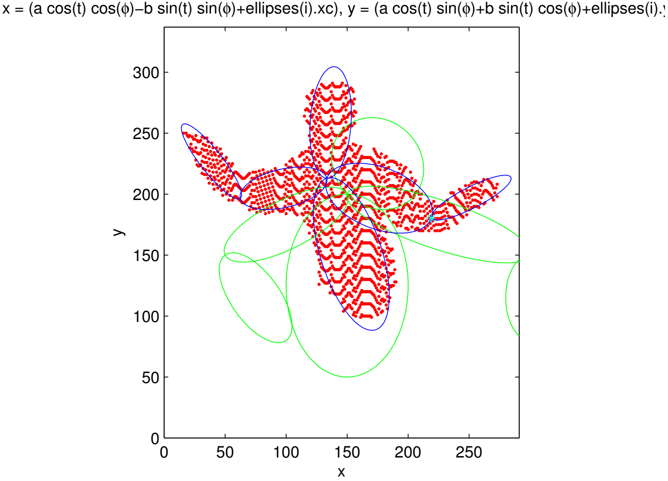
Given an initial parameter θ^1, we obtain a set of parameter estimate $\{\theta^1, \dots \theta^g, \theta^{g+1}, \dots\}$, such that:

$$\log[p(X|\theta^{g+1})p(\theta^{g+1})] \geq \log[p(X|\theta^g)p(\theta^g)] \tag{2}$$

An example relate to my research:

Mutiple Connected Ellipse Fittings

(Xu & Kemp, 2010 & 2013):



Posterior Inference

In many applications, we don't just want the "argmax" of θ, but we are interested in the posterior distribution.

Unfortunately posterior distributions $p(\theta|X)$ is often intractable. Some common approximation methods exist in inference.

- Variational Bayes - good starting point: chapter 10 of Bishop's textbook, and/or *"My ten-cents worth on Variational Bayes"*
- Monte-carlo - my choice (easy to do, but difficult to do-it-well)
- Convex optimization - have no experience with so far
- ...

Experiences with Non-Parametric Bayes

my experience in Non-Parametric Bayes (NPB), aka. Dirichlet Process alike methods. **"hot"** in the machine learning community

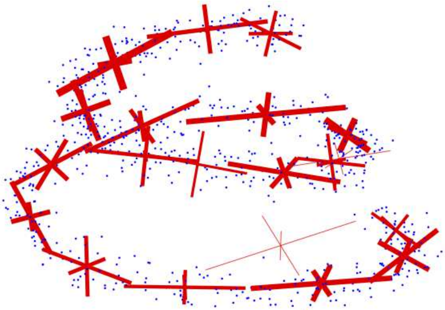
- Dirichlet Process Mixture Model / Chinese Restaurant Problem
- Hierarchical Dirichlet Process (HDP)
- HDP-Hidden Markov Model
- Indian Buffet Process
- ...

Complex posterior, need to learn/write sampler for them.

A quick notes on Dirichlet Process

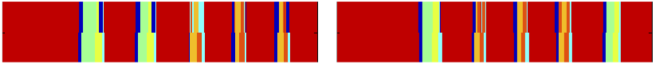
$$\begin{aligned} x_i|\theta_i &\sim F(\theta_i) \\ \theta_i|G &\sim G \\ G|\alpha, H &\sim \mathrm{DP}(\alpha, H) \end{aligned} \tag{3}$$

Figure: From (Rasmussen,1999): Infinite Gaussian Mixture Model



Our work

(Bargi & Xu & Piccardi, 2012): Online HDP-HMM:



Sampling techniques

- Gibbs sampling
- MetropolisHasting
- Rejection Sampling
- Importance Sampling
- Slice sampling
- ...

If you don't care about efficiency, then, use WinBUGS: www.mrc-bsu.cam.ac.uk/bugs/winbugs/

Today, we look at Sequential Monte Carlo, or Particle Filter (with two common simple techniques)

Put it in the relevant context of today's talk

Particle filter is useful for state space model.

Borrow from *"Speaker Localization and Tracking with a Microphone Array on a Mobile Robot Using von Mises Distribution and Particle Filtering"*:

The system state, i.e., the speaker azimuth, is defined via $\theta_k = tan^{-1}\left(\frac{y_k}{x_k}\right)$.

Measurement of the sound source state with M microphones, is $\mathbf{z}_k = \mathbf{h}_k(\theta_k, n_k)$

Importance sampling

To approximate the integral, but $p(z)$ is hard to sample.

$$\begin{aligned}
\mathrm{E}_{p(z)}[f(z)] &= \int f(z)p(z)dz \\
&= \int \underbrace{f(z)\frac{p(z)}{q(z)}}_{\text{new}\tilde{f}(z)} q(z)dz \\
&\approx \frac{1}{N}\sum_{n=1}^{N} f(z^i)\frac{p(z^i)}{q(z^i)}
\end{aligned} \tag{4}$$

Take Importance Sampling to higher dimensions, the importance weights are:

$$w_n(x_{1:n}) = \frac{\gamma(x_{1:n})}{q_n(x_{1:n})} \tag{5}$$

Hard to choose $q(.)$ in high-dimension

Solution : rewrite equation (5) in the following:

$$w_n(x_{1:n}) = \frac{\gamma(x_{1:n})}{q_n(x_n|x_{1:n-1})q_{n-1}(x_{1:n-1})} \times \frac{\gamma(x_{1:n-1})}{\gamma(x_{1:n-1})}$$

re-arrange:

$$w_n(x_{1:n}) = \frac{\gamma(x_{1:n})}{q(x_{1:n})} = w_{n-1}(x_{1:n-1}) \times \frac{\gamma(x_{1:n})}{\gamma(x_{1:n-1})q(x_n|x_{1:n-1})}$$

Revision on SMC (2)

Top-down:

$$w_n(x_{1:n}) = w_{n-1}(x_{1:n-1}) \frac{\gamma(x_{1:n})}{\gamma(x_{1:n-1}) q(x_n | x_{1:n-1})} \tag{6}$$

Bottom-up:

$$w_n(x_{1:n}) = w_1(x_1) \prod_{j=2}^{n} \frac{\gamma(x_{1:j})}{\gamma(x_{1:j-1}) q(x_j | x_{1:j-1})}$$

The two are equivalent

Just too easy to put it all in an algorithm:

The SIS algorithm:

$$\begin{aligned}
&\text{At dimension } n = 1\text{: For each particle } i \\
&\quad \text{Sample } x_1^i \sim q_1(x_1) \\
&\quad \text{Compute the weights } w_1^i \propto \frac{\gamma(x_1^i)}{q_1(x_1^i)} \\
&\text{At dimension } n \geq 2\text{: For each particle } i \\
&\quad \text{Sample } x_n^i \sim q_n(x_n | x_{1:n-1}^i) \\
&\quad \text{Compute the weights } w_n^i \propto w_{n-1}^i \frac{\gamma(x_{1:n}^i)}{\gamma(x_{1:n-1}^i) q(x_n^i | x_{1:n-1}^i)}
\end{aligned} \tag{7}$$

Particle Filter

Put this in a state-space setting, you have particle filter!
By changing n to t to reflect time sequentiality. In here, we assume that:

$$p(x_{1:t}|y_{1:t}) = \frac{p(x_{1:t}, y_{1:t})}{p(y_{1:t})} = \frac{\gamma_t(x_{1:t})}{\mathcal{Z}}$$

In here, we assume:

$$\begin{aligned}
\gamma_t(x_{1:t}) &= p(x_{1:t}, y_{1:t}) \\
&= p(y_t|x_{1:t}, y_{1:t-1})p(x_t|x_{1:t-1}, y_{1:t-1})\gamma_{t-1}(x_{1:t-1}) \\
&= p(y_t|x_t)p(x_t|x_{t-1})\gamma_{t-1}(x_{1:t-1})
\end{aligned} \tag{8}$$

Particle Filter

Divide by the proposal distribution $q(.)$, and do the same trick, this time, we use:

$$w_t(x_{1:t}) = \frac{\gamma_{(1:t)}}{q_{(1:t)}} = \frac{\gamma_{(1:t-1)}}{q_{(1:t-1)}} \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{q(x_t|x_{1:t-1})}$$

we can make a "reasonable" assumption that:

$$q(x_t|x_{1:t-1}) \equiv q(x_t|x_{t-1}, y_t) \tag{9}$$

Hence,

$$w_t(x_{1:t}) = w_{t-1}(x_{1:t-1}) \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{q(x_t|x_{t-1}, y_t)}$$

question is How are we going to choose $q(.)$ **a short answer** Choose $q(.)$ somehow from your dynamic model

Optimal proposal: $q(x_t|x_{k-1}, y_t) = p(x_t|x_{k-1}, y_t)$

Stated in [Doucet 1998], $q(x_t|x_{k-1}, y_t) = p(x_t|x_{k-1}, y_t)$ is optimal,then:

$$
\begin{aligned}
w_{(1:t)} &\propto w_{(1:t-1)} \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{p(x_t|x_{t-1}, y_t)} \\
&= w_{(1:t-1)} \times \frac{p(y_t|x_t)p(x_t|x_{t-1})p(y_t|x_{t-1})p(x_{t-1})}{p(y_t|x_t)p(x_t|x_{t-1})p(x_{t-1})} \\
&= w_{(1:t-1)} \times p(y_t|x_{t-1})
\end{aligned}
$$

However, $p(y_t|x_{t-1})$ is quite meaningless:

$$
w_{(1:t)} \propto w_{(1:t-1)} \times \int_{x_t} p(y_t|x_t)p(x_t|x_{t-1}) \tag{10}
$$

Two problem: (1) Difficult to sample from $p(x_t|x_{k-1}, y_t)$ and (2) integral is difficult to perform!

Main talk: sub-optimal methods

In this talk, I will present two "popular" sub-optimal sampling methods first:

- Bootstrap Particle Filter
- Auxiliary Particle Filter

Bootstrap Particle Filter

Sometimes calling it Condensational Filter. (Famous Michael Isard) Let $q(x_t|x_{k-1}, y_t) = p(x_t|x_{k-1})$, i.e., y_t does not participate in the proposal $q(.)$

$$
\begin{aligned}
w_{(1:t)} &\propto w_{(1:t-1)} \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{p(x_t|x_{t-1})} \\
&= w_{(1:t-1)} \times p(y_t|x_t)
\end{aligned}
\tag{11}
$$

- particles x_t^i are sampled from $p(.|x_{t-1})$, but are weighted by $p(y_t|x_t^i)$
- the danger is that x_t^i may receive close to zero weight if $p(y_t|x_t^i)$ is very small.

The Condensational Filter algorithm:

$$
\begin{aligned}
&\text{At time } t\\
&\text{For each particle } i:\\
&\quad \text{Sample } x_t^i \sim p(x_t|x_{t-1}^i) \qquad \big(\text{Or } x_1^i \sim p(x_1) \text{ when } t=1\big)\\
&\quad \text{Compute the weights } w_t^i \propto \pi_{t-1}^i p(y_t|x_t^i)\\
&\text{normalize weights } \pi_t^i = \frac{w_t^i}{\sum_{i=1}^N w_t^i}
\end{aligned}
\tag{12}
$$

Problem particle degeneracy occurs very quickly.
Solution break those big particle into smaller ones, from the "re-sampling" step. To determine if "big particles" exist, check effective particle size.
BTW re-sampling does not solve particle degeneracy problem altogether.

Introducing Re-Sampling

Re-sampling sometimes can be considered as jointly "sample" an index i^j to indicate which $x_{t-1}^{i^j}$ generated x_t^i, and x_t^i itself.

$$\begin{aligned} &x_t^i \sim q(x_t|x_{t-1}^i, y_t) \\ &\text{ becomes:} \\ &j \sim \pi_{t-1}(x_{1:t-1}) \\ &x_t^i \sim q(x_t|x_{t-1}^{i^j}, y_t) \end{aligned} \tag{13}$$

For each particle i at time t, you get (x_t^i, i^j).

Introducing Re-Sampling

Substituting N of the (x_t^i, i^j) into the following:

$$w_t(x_{1:t}) \propto \pi_{t-1}(x_{1:t-1}) \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{q(x_t|x_{t-1}, y_t)}$$

$$\begin{aligned} w_t^i(x_{1:t}) &\propto \pi_{(t-1)}^{i^j} \times \frac{p(y_t|x_t^i)p(x_t^i|x_{t-1}^{i^j})}{\pi_{(t-1)}^{i^j} q(x_t^i|x_{t-1}^{i^j}, y_t)} \\ &= \frac{p(y_t|x_t^i)p(x_t^i|x_{t-1}^{i^j})}{q(x_t^i|x_{t-1}^{i^j}, y_t)} \end{aligned}$$

In the bootstrap filter:

$$w_t^i(x_{1:t}) \propto p(y_t|x_t^i)$$

The Condensational Filter algorithm:

At time t

For each i:

Sample $j \sim \pi_{t-1}(x_{1:t-1})$ $\quad -$ choose an an ancestor

Sample $x_t^i \sim p(x_t|x_{t-1}^{ij})$ $\quad \Big($Or $x_1^i \sim p(x_1)$ when $t = 1\Big)$ $\quad$ (14)

Compute the weights $w_t^i \propto p(y_t|x_t^i)$

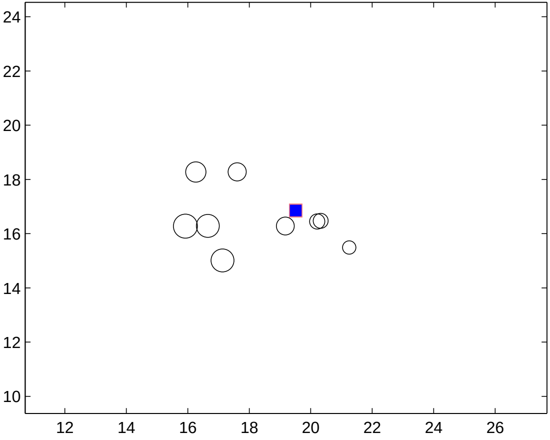
normalize weights $\pi_t^i = \dfrac{w_t^i}{\sum_{i=1}^N w_t^i}$

A little demo

- $p(x_t|x_{t-1}) = \mathcal{N}(Ax_{t-1} + B, Q)$
- $p(y_t|x_t) = \mathcal{N}(x_t, R)$

This is just for demo purpose, you can compute $p(x_t|y_{1:t})$ exactly using Kalman Filter!

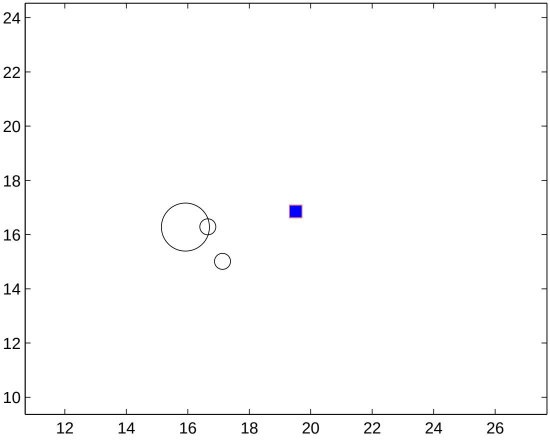
Representation for $p(x_{t-1}|y_{1:t-1})$



- Circles are weighted particle representation of $p(x_{t-1}|y_{1:t-1})$
- The blue square is y_t

Re-sampling

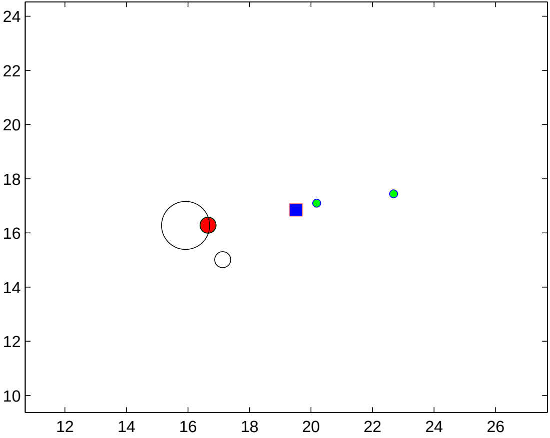
To sample $j \sim \pi_{t-1}(x_{1:t-1})$:



- Size of the circle indicates the number of times x^{i^j} was

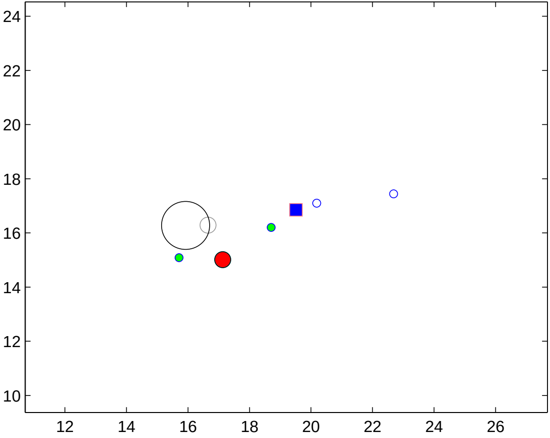
Transition demos

Sample $x_t^i \sim p(x_t | x_{t-1}^{i^j}) : \forall i^j = 1$



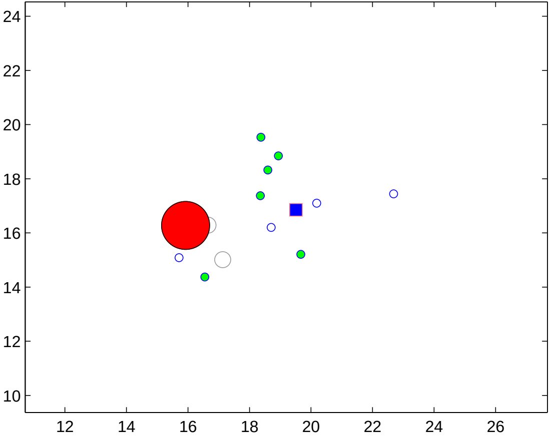
Transition demos

Sample $x_t^i \sim p(x_t | x_{t-1}^{i^j}) : \forall i^j = 2$



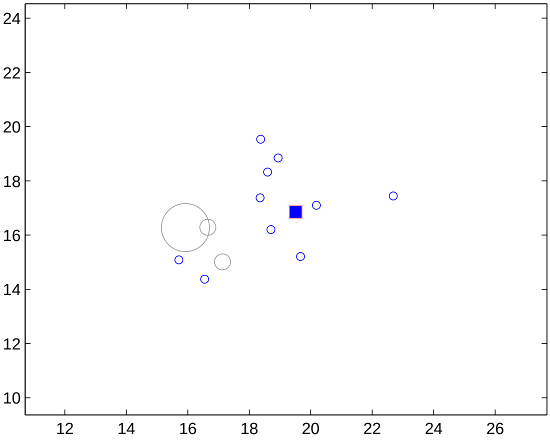
Transition demos

Sample $x_t^i \sim p(x_t | x_{t-1}^{i^j}) : \forall i^j = 3$



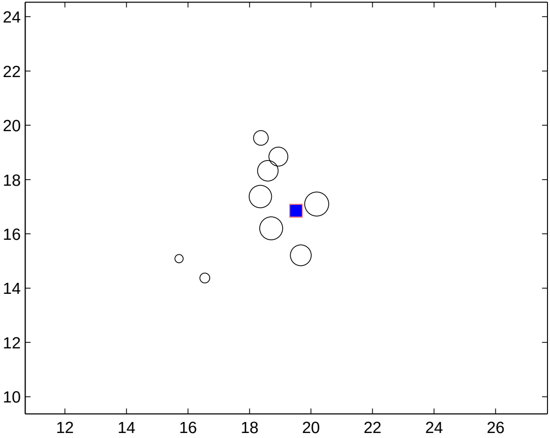
Transition demos

Here are the complete $\{x_t^i\}_1^N$ sampled.



After re-weighting

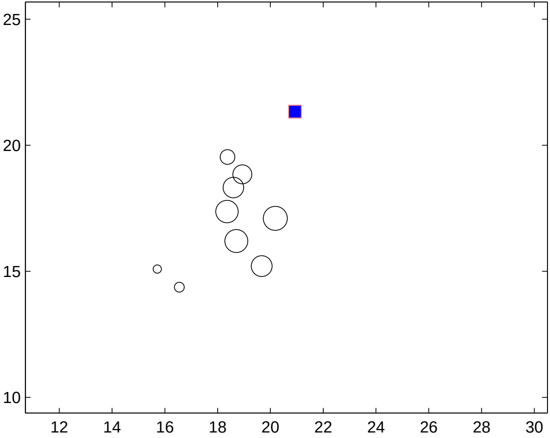
Compute the weights $w_t^i \propto p(y_t | x_t^i)$:



The above is the representation for $p(x_t | y_{1:t})$ Note that weights

Next t

So the recursion will repeat:



The above is the representation for $p(x_{t-1}|y_{1:t-1})$ in the next t:

Some cool things you can do just with Bootstrap Filter

For example, A Coupled two-states dynamic model:
To estimate $p(x^1_{1:t}, x^2_{1:t} | y^1_{1:t}, y^2_{1:t})$

$$
\begin{aligned}
& w^i_t(x^1_{1:t}, x^2_{1:t}) \propto \\
& = \frac{g_1(y^1_t|x^1_t) g_2(y^2_t|x^2_t) f_1(x^1_t|x^1_{t-1}, x^2_{t-1}) f_2(x^2_t|x^1_{t-1}, x^2_{t-1})}{q^1(x^1_t|y^1_t, x^1_{t-1}, x^2_{t-1}) q^2(x^2_t|y^2_t, x^1_{t-1}, x^2_{t-1})} \\
& \quad w^i_{t-1}(x^1_{1:t-1}, x^2_{1:t-1})
\end{aligned}
\tag{15}
$$

(leaving out the case of $t = 1$, and re-sampling step)

$$
\begin{aligned}
&\text{At time } t\text{:}\\
&\text{Sample } x_t^{1,(i)} \sim f_1(x_t^1 | x_{t-1}^{1,(i)}, x_{t-1}^{2,(i)})\\
&\text{Sample } x_t^{2,(i)} \sim f_2(x_t^2 | x_{t-1}^{1,(i)}, x_{t-1}^{2,(i)})\\
&\text{Compute the weights } w_t^{1,(i)} \propto \pi_{t-1}^{1,(i)} g_1(y_t^{1,(i)} | x_t^{1,(i)})\\
&\text{Compute the normalized weights } \pi_t^{1,(i)}\\
&\text{Compute the weights } w_t^{2,(i)} \propto \pi_{t-1}^{2,(i)} g_2(y_t^{2,(i)} | x_t^{2,(i)})\\
&\text{Compute the normalized weights } \pi_t^{2,(i)}
\end{aligned}
\tag{16}
$$

- **idea**: Let y_t also participates in the proposal.
- **how**: In bootstrap sampling, x_t^i is more likely to be generated from $x_{t-1}^{i^j}$ when the value of $\pi_{t-1}^{i^j}$ is high. **Then** , how about let's also give preference to those $x_{t-1}^{i^j}$ where their proposed $x^i \sim x_{t-1}^{i^j}$ can be weighted higher by $p(y_t|x^i)$ as well?
- **in my word**: Have a bit of scouting before sampling!

Auxiliary Particle Filter algorithm

$$\mu_t^i = \mathrm{E}_{x_t}[x_t | x_{t-1}^i], \text{ OR: } \mu_t^i \sim p(x_t | x_{t-1}^i) \tag{17}$$

At time t, for each particle i:

$$\begin{aligned}
&\text{Calculate } \mu_t^i \\
&\text{Compute the weights } w_t^i \propto p(y_t | \mu_t^i) \pi_{t-1}^i \\
&\text{Normalize } w_t^i \\
&\text{Sample } i^j \sim \{w_t^i\} \\
&\text{Sample } x_t^i \sim p(x_t | x_{t-1}^{i^j}) \\
&\text{Assign } w_t^i \propto \frac{p(y_t | x_t^i)}{p(y_t | \mu_t^{i_j})} \\
&\text{Normalize } w_t^i \to \pi_t^i
\end{aligned} \tag{18}$$

Why $w_t^i \propto \frac{p(y_t|x_t^i)}{p(y_t|\mu_t^{i^j})}$? The proposal

Looking at the proposal:

$$q(x_t^i, i^j|.) = \underbrace{q(x_t^i|i^j, x_{t-1}, y_{1:t})}_{\text{2: choose } x_t} \underbrace{q(i^j|x_{t-1}, y_{1:t})}_{\text{1: choose the index}} \tag{19}$$

From the algorithm of the previous page:

$$\begin{aligned} &\text{1st Step: choose the index: } q(i^j|x_{t-1}, y_{1:t}) \propto p(y_t|\mu_t^{i^j})\pi_{t-1}^{i^j} \\ &\text{2nd Step: choose the } x_t\text{: } q(x_t^i|i^j, x_{t-1}, y_{1:t}) \equiv p(x_t^i|x_{t-1}^{i^j}) \end{aligned} \tag{20}$$

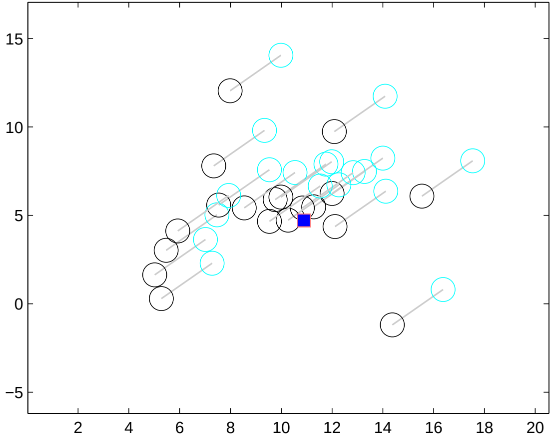
Why $w_t^i \propto \frac{p(y_t|x_t^i)}{p(y_t|\mu_t^{i_j})}$?

Substituting N of the (x^i, i^j) into the following:

$$w_t(x_{1:t}) \propto \pi_{t-1}(x_{1:t-1}) \times \frac{p(y_t|x_t)p(x_t|x_{t-1})}{q(x_t|x_{t-1}, y_t)}$$

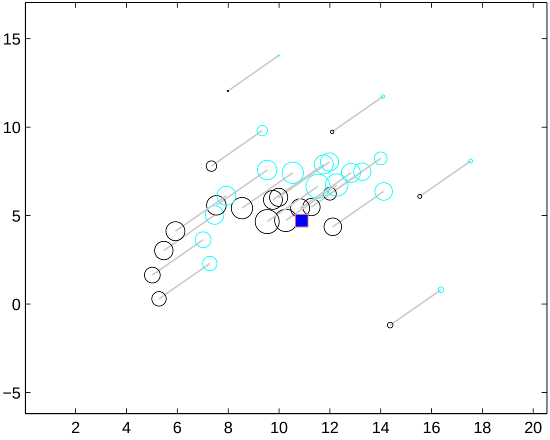
$$\begin{aligned} w_t^i(x_{1:t}) &\propto \pi_{t-1}^{i^j} \times \frac{p(y_t|x_t^i)p(x_t|x_{t-1}^{i^j})}{p(y_t|\mu_t^{i^j})\pi_{t-1}^{i^j}p(x_t^i|x_{t-1}^{i^j})} \\ &= \frac{p(y_t|x_t^i)}{p(y_t|\mu_t^{i^j})} \end{aligned}$$

Representation for $p(x_{t-1}|y_{1:t-1})$ and μ_t^i

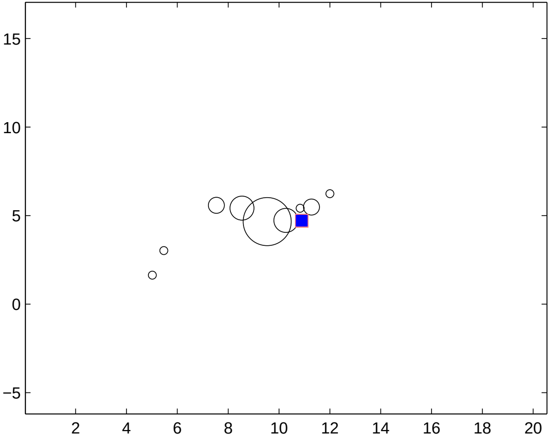


- Light blue circles are μ_t^i for each x_{t-1}^i

New weights: $\propto p(y_t|\mu_t^i)\pi_{t-1}^i$

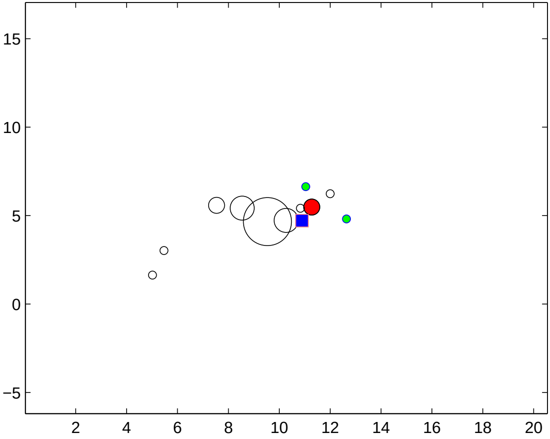


Re-sampling

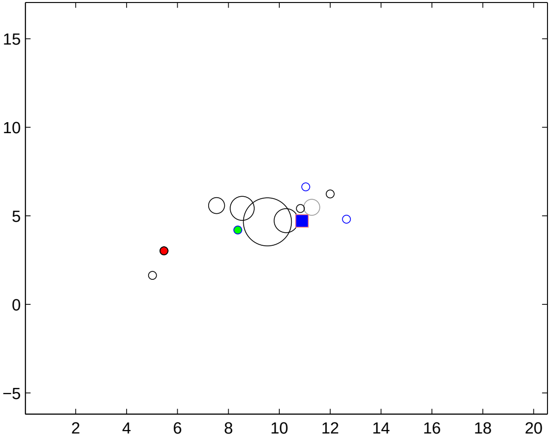


- Size of the circle indicates the number of times x_{t-1}^{ij} was selected.

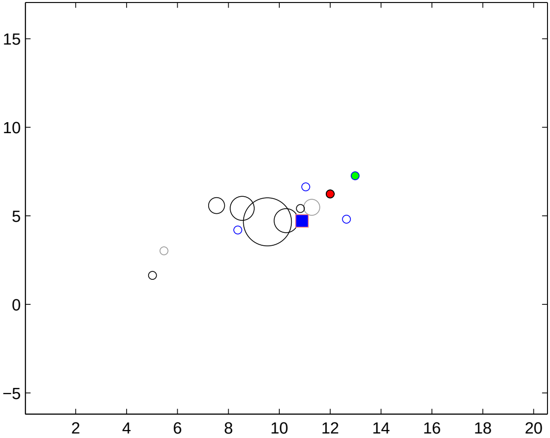
Transition demos



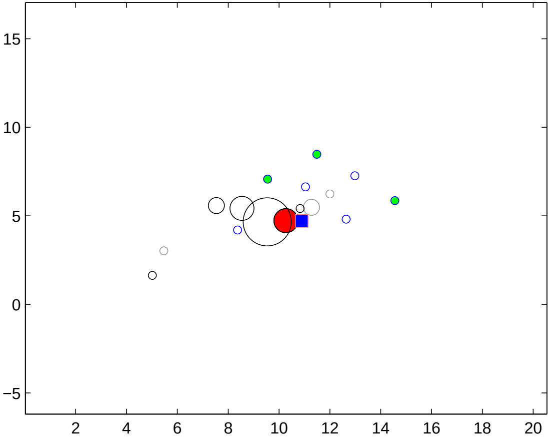
Transition demos



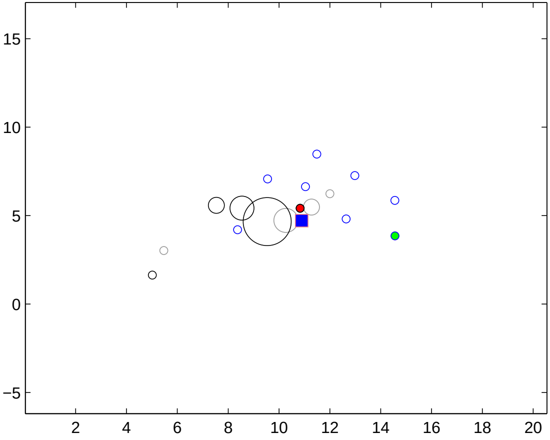
Transition demos



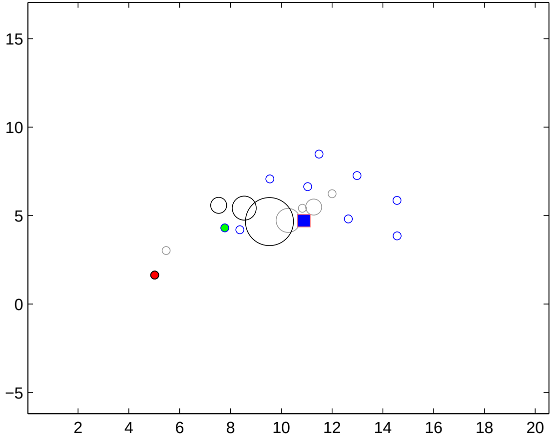
Transition demos



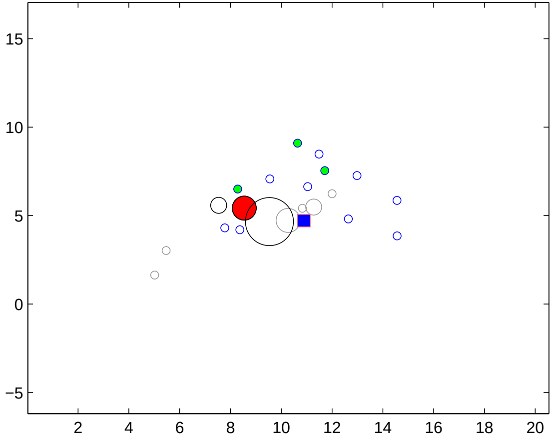
Transition demos



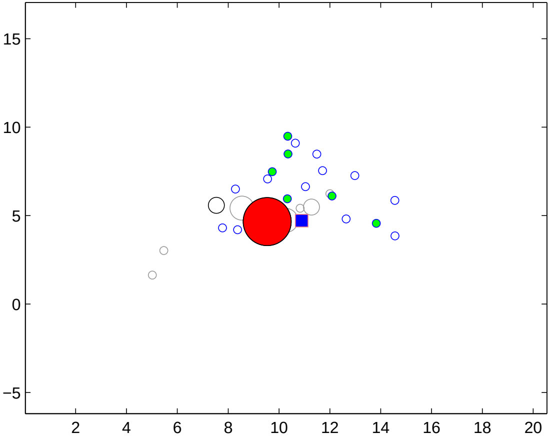
Transition demos



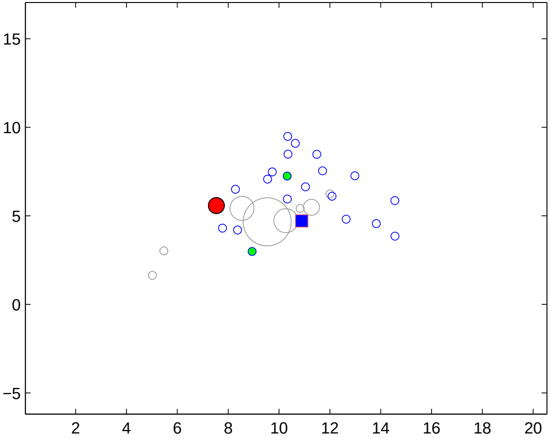
Transition demos



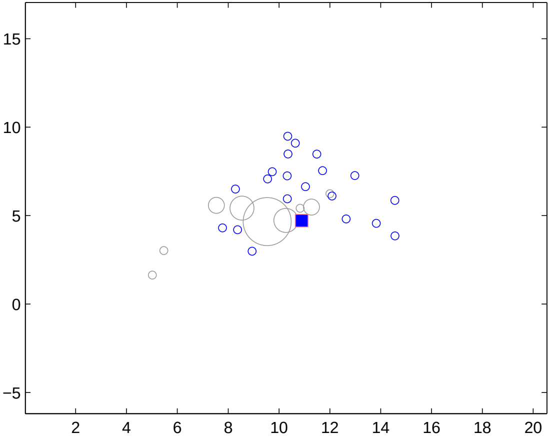
Transition demos



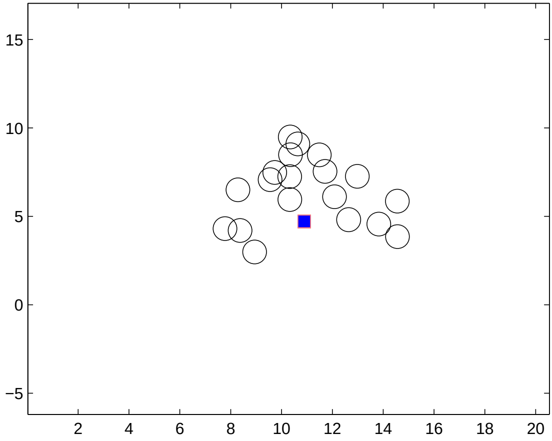
Transition demos



Transition demos

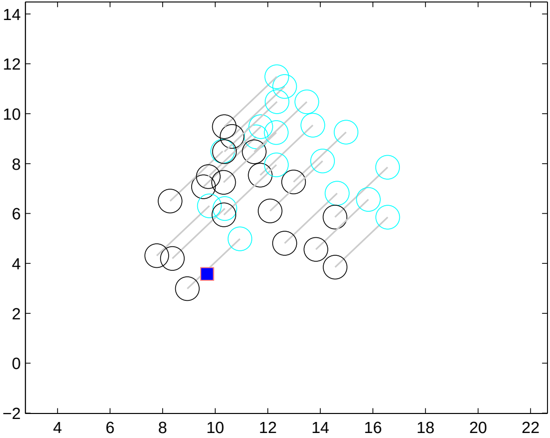


After re-weighting



The above is the representation for $p(x_t|y_{1:t})$ Note that weights are in log scale..

Next t



The above is the representation for $p(x_{t-1}|y_{1:t-1})$ in the next t:

Main References

- Arulampalam, M.S. and Maskell, S. and Gordon, N. and Clapp, T, A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing, 2002
- Pitt, M.K.; Shephard, N. (1999). "Filtering Via Simulation: Auxiliary Particle Filters". Journal of the American Statistical Association (American Statistical Association) 94 (446): 590591