9. The Particle Filter

The *Particle Filter* (PF) is an approximation of the Bayesian state estimator for a general nonlinear system and general noise distributions. The basic idea is to approximately represent the state PDF by a (large) number of samples, which are called *particles*. Essentially, in a region where the PDF takes large values, there is a large number of particles. The basic recursive update mechanism is as follows: the particles are propagated through the process model; then, the particles are weighted according to the measurement likelihood; and, finally, a resampling step generates a new set of particles, again with equal weights.

9.1 Model

We consider the nonlinear discrete-time system

$$x(k) = q_{k-1}(x(k\text{–}1), v(k\text{–}1)), \quad k = 1, 2, \dots$$
$$z(k) = h_k(x(k), w(k)),$$

where $x(0), \{v(\cdot)\}$ and $\{w(\cdot)\}$ are independent and can be discrete or continuous random variables with known PDFs.

We introduce the PF for a system without input $u(k\text{–}1)$, but the discussion directly extends to a system with input (the known input $u(k\text{–}1)$ can be absorbed in the explicit time dependency of $q_{k-1}(\cdot)$).

9.2 Monte Carlo Sampling

Monte Carlo sampling refers to the basic technique of using a large number of samples (called particles) to approximate the PDF of a random variable. It is beneficial, for example, for computing an approximation of the PDF of a function of the random variable: instead of using change of variables, the function is simply evaluated at the particles. Monte Carlo sampling is the underlying idea that is used in the PF.

Discrete random variables

Consider a discrete random variable y, with a finite outcome space $\mathcal{Y} = \{1, 2, 3, \dots, \overline{Y}\}$, and an associated PDF $f_y(y)$. We draw N samples from this PDF: $\{y^1, y^2, \dots, y^N\}$, where samples are independent.

- For $n \in \{1, 2, \dots, N\}$, let

$$p_i^n := \begin{cases} 1 & \text{if } y^n = i \\ 0 & \text{otherwise}, \end{cases} \qquad i = 1, \dots, \overline{Y}$$

  that is, p_i^n indicates if $y^n = i$ (when $p_i^n = 1$) or not (when $p_i^n = 0$). In a more compact form, we write

$$p_i^n = \delta(y^n - i),$$

  where $\delta(\cdot)$ is the Kronecker delta function:

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise}. \end{cases}$$

Date compiled: May 14, 2013

- Let p_i denote the sample average of p_i^n over the N samples (i.e. the fraction of samples that take the value i):

$$p_i := \frac{1}{N}\sum_{n=1}^{N} p_i^n \,.$$

  By the *law of large numbers* (LLN) the sample average converges to the expected value of p_i^n; that is,

$$p_i \to \mathrm{E}[p_i^n] = \sum_{y^n=1}^{\bar{Y}} \delta(y^n - i) f_y(y^n) = f_y(i) \quad \text{for } N \to \infty.$$

- Conclusion: the PDF of a random variable can be approximated by repeated sampling of a distribution.

Now consider a function of $y \in \mathcal{Y}$: $x = g(y)$. We have already discussed in class how to construct $f_x(x)$ from $f_y(y)$ (change of variables). Now, we are interested in *approximating* $f_x(x)$ by a set of samples.

- Let $\mathcal{X} = g(\mathcal{Y})$. For each $j \in \mathcal{X}$, let $\mathcal{Y}_j = \{y_{j,i}\}$ be the set of all $y_{j,i} \in \mathcal{Y}$ such that $g(y_{j,i}) = j$.
- Let $x^n := g(y^n)$ and

$$r_j^n := \begin{cases} 1 & \text{if } x^n = g(y^n) = j \\ 0 & \text{otherwise} \end{cases} \quad = \delta(x^n - j).$$

- Let

$$r_j = \frac{1}{N}\sum_{n=1}^{N} r_j^n \,.$$

  By the LLN,

$$r_j \to \mathrm{E}[r_j^n] \quad \text{for } N \to \infty \quad \text{and} \quad \mathrm{E}[r_j^n] = \Pr(x^n = j) = \Pr(y^n \in \mathcal{Y}_j) = \sum_{y_{j,i} \in \mathcal{Y}_j} f_y(y_{j,i}) \,.$$

  But $\sum_{y_{j,i} \in \mathcal{Y}_j} f_y(y_{j,i}) = f_x(j)$ as proven in class (see section on change of variables).

- Hence,

$$f_x(j) \approx \frac{1}{N}\sum_{n=1}^{N} r_j^n = \frac{1}{N}\sum_{n=1}^{N} \delta(x^n - j),$$

  and we can approximate $f_x(x)$ by the samples $x^n = g(y^n)$. Instead of solving the change of variables, we simply need to evaluate the function $g(\cdot)$ at all samples y^n.

Extensions:

- The above is also true for joint discrete random variables:

$$f_x(\xi) \approx \frac{1}{N}\sum_{n=1}^{N} \delta(\xi - x^n)$$

  where now ξ, x^n are vectors, and $\delta(\cdot)$ refers to the vector version of the Kronecker delta (that is, δ is one if all entries of its vector argument are zero, otherwise δ is zero).

- We have only considered finite $\mathcal{X}$ and $\mathcal{Y}$, but the above holds for the infinite case as well.

Continuous random variables

Consider the continuous random variable y with PDF $f_y(y)$. Let $x = g(y)$. How can we approximate $f_x(x)$?

- Let $\{y^n\}$ be repeated samples from $f_y(y)$, and let $x^n := g(y^n)$. Fix Δx (bin size). Then

$$\Pr(\overline{x} \le x \le \overline{x} + \Delta x) \approx \frac{1}{N} \sum_{n=1}^{N} r_{\overline{x}}^n =: r_{\overline{x}}$$

where

$$r_{\overline{x}}^n = \begin{cases} 1 & \text{if } \overline{x} \le x^n \le \overline{x} + \Delta x \\ 0 & \text{otherwise}\,. \end{cases}$$

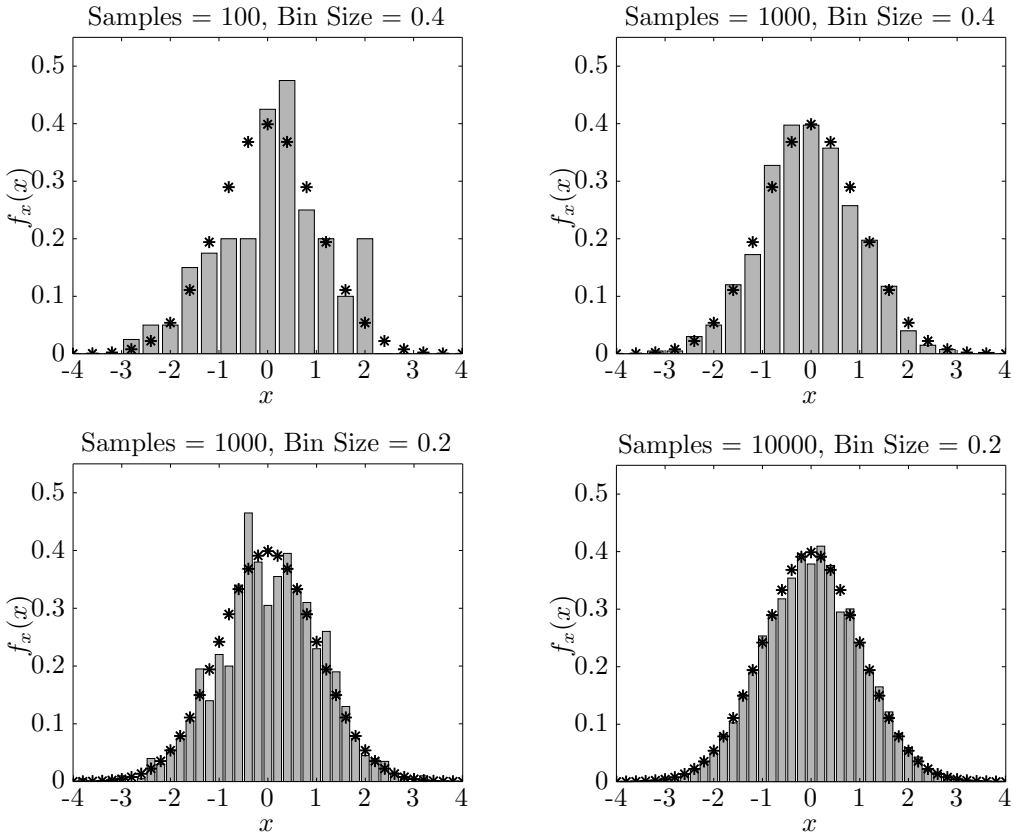
- Since (for small Δx and smooth $f_x(x)$)

$$\Pr(\overline{x} \le x \le \overline{x} + \Delta x) = \int_{\overline{x}}^{\overline{x}+\Delta x} f_x(x)\; dx \;\approx\; f_x(\xi)\Delta x \quad \text{for } \overline{x} \le \xi \le \overline{x} + \Delta x$$

we have

$$f_x(\xi) \approx \frac{r_{\overline{x}}}{\Delta x}, \quad \overline{x} \le \xi \le \overline{x} + \Delta x\,.$$

Example: We approximate a Gaussian distribution with zero mean and variance one by repeatedly drawing samples from this distribution. The graphs below show the result for different numbers of samples N and different bin sizes Δx. The exact value of the PDF is shown in black (asterisks), and the approximated value obtained from Monte Carlo sampling is shown as gray bars.



Typically, we use informal notation:

$$f_x(\xi) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - x^n)$$

with $\delta(\cdot)$ the Dirac delta pulse. In particular, note that

$$\int_{\overline{x}}^{\overline{x}+\Delta x} \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - x^n)\,d\xi = \frac{1}{N}\sum_{n=1}^{N} r_{\overline{x}}^n = r_{\overline{x}} \approx \Pr(\overline{x} \le x \le \overline{x} + \Delta x).$$

The advantage of this notation is that it is the same for discrete and continuous random variables (where $\delta(\cdot)$ is the Kronecker delta function or the Dirac delta pulse, respectively).

9.3 Recall Bayesian State Estimation

The Bayesian tracking algorithm (also called Bayesian state estimator) computes $f(x(k)|z(1{:}k))$. The objective of the PF is to *approximate* $f(x(k)|z(1{:}k))$.

We introduce auxiliary variables, just like we did for the Kalman filter. We define new random variables, $x_p(k)$, $x_m(k)$, and $z_m(k)$, and we distinguish again between the random variable $z(k)$ and the value $\mathbf{z}(k)$ that $z(k)$ takes (the actual measurement at time k).

Init: $x_m(0) := x(0)$

$$\left.\begin{array}{ll} \textbf{S1:} & x_p(k) := q_{k-1}(x_m(k{-}1), v(k{-}1)) \\ \textbf{S2:} & z_m(k) := h_k(x_p(k), w(k)) \\ & x_m(k) \text{ defined via its PDF} \\ & \quad f_{x_m(k)}(\xi) := f_{x_p(k)|z_m(k)}(\xi|\mathbf{z}(k)) \quad \forall \xi \end{array}\right\} \quad k = 1, 2, \ldots$$

Analogously to the proof for linear systems, we can show that for all ξ and $k = 1, 2, \ldots,$

$$f_{x_p(k)}(\xi) = f_{x(k)|z(1:k-1)}(\xi|\mathbf{z}(1{:}k{-}1))$$
$$f_{x_m(k)}(\xi) = f_{x(k)|z(1:k)}(\xi|\mathbf{z}(1{:}k))\,.$$

Using the short-hand notation:

$$f(x_p(k)) = f(x(k)|z(1{:}k{-}1))$$
$$f(x_m(k)) = f(x(k)|z(1{:}k))\,.$$

9.4 Prior Update

Given the PDF $f_{x_m(k-1)}(\xi)$ of $x_m(k{-}1)$, we construct the PDF $f_{x_p(k)}(\xi)$ of $x_p(k)$. We *approximate* $f_{x_m(k-1)}(\xi)$ and $f_{x_p(k)}(\xi)$ by Monte Carlo sampling:

- Let

  $$f_{x_m(k-1)}(\xi) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - x_m^n(k{-}1))$$

  where $\{x_m^n(k{-}1)\}$ are N particles that approximate the PDF of $x_m(k{-}1)$. These particles are Monte Carlo samples of $f_{x_m(k-1)}(\xi)$. We can therefore form an approximation of

  $$f_{x_p(k)}(\xi) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - x_p^n(k))$$

  where

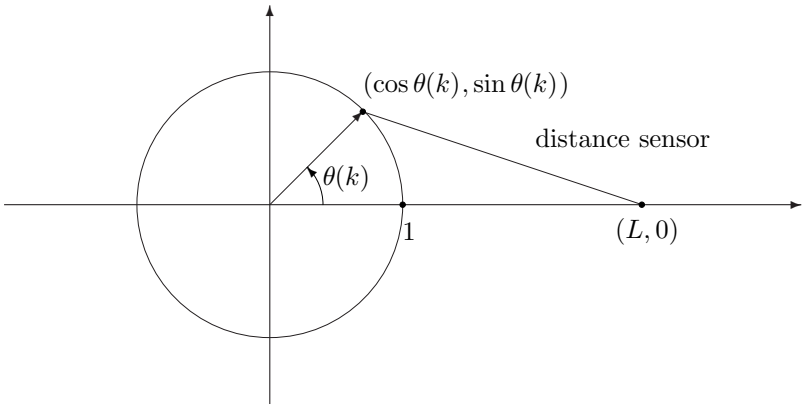
  $$x_p^n(k) := q_{k-1}(x_m^n(k{-}1), v^n(k{-}1)), \qquad \text{for } n = 1, 2, \ldots, N,$$

  and $v^n(k{-}1)$ are Monte Carlo samples of $f(v(k{-}1))$.

- This is straightforward and intuitive. We simply propagate N particles through the process dynamics. This is like a parallel simulation.
- Provided N is "large" *and* $\{x_m^n(k\text{–}1)\}$ is a "good" representation of $f_{x_m(k-1)}(\xi)$, $\{x_p^n(k)\}$ will be a "good" representation of $f_{x_p(k)}(\xi)$.

9.5 Example

We revisit an old example, slightly modified. We apply the PF prior update step and do not consider any measurements yet.



- $\theta(k)$: location of an object on a circle, $\theta(k) \in [0, 2\pi)$.
- Dynamics:

  $$\theta(k) = \text{mod}\big(\theta(k\text{–}1) + s(k\text{–}1),\, 2\pi\big)$$

  where $s(k\text{–}1)$ is a CRV uniformly distributed on $[-\overline{s}+b, \overline{s}+b]$, and b is a CRV (not time-dependent) uniformly distributed on $[-\overline{s}, \overline{s}]$. The modulo operation $\text{mod}(y, 2\pi)$ adds or subtracts multiples of 2π to the argument y such that the result is in the interval $[0, 2\pi)$.
- Introducing the states $x_1(k) := b$ and $x_2(k) := \theta(k)$, we can rewrite the dynamics in standard form:

  $$x_1(k) = x_1(k\text{–}1)$$
  $$x_2(k) = \text{mod}\big(x_2(k\text{–}1) + x_1(k\text{–}1) + v(k\text{–}1),\, 2\pi\big),$$

  where $x_1(0)$ is uniformly distributed on $[-\overline{s}, \overline{s}]$ and $v(k\text{–}1)$ is uniformly distributed on $[-\overline{s}, \overline{s}]$. The distribution of $x_2(0)$ reflects our initial knowledge of the object's location.
- Note that $x(k)$ is now a CRV. When we discussed the example in the context of Bayesian tracking, we assumed that the object moves in discrete steps, which allowed us to implement the Bayesian tracking algorithm directly. Here, we use a PF to track the objects location (as a first step, we consider the PF without measurements; next lecture, we add measurements).
- Implementation and simulation results:
  - Parameter values (known to the PF): the object's initial location $\theta(0) = \frac{\pi}{2}$, $\overline{s} = 0.01$.
  - We use $N = 1000$ particles. Each particle $x^n(k)$ is a vector with two elements: the first corresponding to the bias b and the second corresponding to the location $\theta(k)$.
  - The histogram of the object's location spreads out over time: because of the process noise, our knowledge of the object's location becomes less accurate. This is amplified when increasing $\overline{s}$.
  - The bias histogram does not change. Without any measurement, there is no way to infer information about the bias, and the process dynamics do not alter $x_1(k)$.
  - If we do not know $\theta(0)$ and initialize the PF with particles drawn from a uniform distribution on $[0, 2\pi)$, we basically have no idea where the object is.