Short course

A vademecum of statistical pattern recognition and machine learning

The Kalman filter. Particle filters

Massimo Piccardi
University of Technology, Sydney, Australia

Agenda

- State space models
- Recursive Bayesian estimation
- The Kalman filter
- Particle filters
- A mention to probabilistic data association
- Example papers
- References

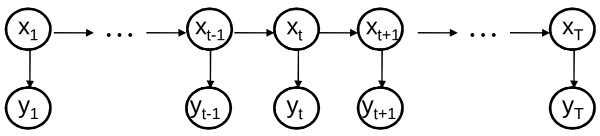
State space models

- In the hidden Markov model, the state variables are discrete random variables
- In many cases, we are instead interested in systems whose **state** is modelled as a **continuous, multivariate** random variable. Such systems are often referred to as *state(-)space models*

© Massimo Piccardi, UTS

State space models: probabilistic graphical model

- The graphical model is identical to that of the HMM, with the only difference that the latent states (henceforth noted as $x_{1:T}$) are each a continuous, multivariate random variable. We note the measurements as $y_{1:T}$



- On the side, when state space models are used in Automation, another layer of control input variables is typically present. Yet, it is not relevant for us

© Massimo Piccardi, UTS

State space models: generative model

- Again, the generative model associated with the graphical model is identical to that of the HMM:

$$p(y_{1:T}, x_{1:T}) = p(x_1)\prod_{t=2}^{T} p(x_t \mid x_{t-1})\prod_{t=1}^{T} p(y_t \mid x_t)$$

- Given that the measurements are also assumed to be continuous, multivariate random variables, all the factors above are pdf (e.g., Gaussian)
- The assumptions implied by this model are known as **Bayesian tracking hypotheses**. Variable x is the latent state of a target to be tracked

State and measurement equations

- An alternative representation for a state space model is given in terms of two equations:

$$x_t = f(x_{t-1}, v_{t-1})$$

state equation, or *system*, or *process*, *model*

$$y_t = h(x_t, n_t)$$

measurement equation, or *output*, or *emission, model*

x_t is the state r.v. at time t, $\in \Re^n$
v_t is the process noise r.v. at time t, $\in \Re^l$
y_t is the measurement r.v. at time t, $\in \Re^p$
n_t is the measurement noise r.v. at time t, $\in \Re^r$
(a value for x_1, v_1 is also needed)

Inference: recursive Bayesian estimation

- Like with HMM, one of the main problems is to infer the sequence of states from the sequence of measurements
- A focal difference is that the estimation is typically provided in a **recursive** way: at every new y_t, the estimate is updated up to x_t. This suits the requirements of real-time tracking
- There are two usual inference problems:
  - the *posterior filtering density*, $\mathbf{p(x_{1:t} \mid y_{1:t})}$
  - the *marginal filtering density*, $\mathbf{p(x_t \mid y_{1:t})}$, obviously a marginal of the above

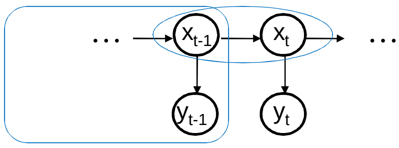
© Massimo Piccardi, UTS

Marginal filtering density

- How do we recursively estimate $\mathbf{p(x_t \mid y_{1:t})}$?
- Assume available: $p(x_{t-1} \mid y_{1:t-1})$
- $p(x_1)$, $p(y_t|x_t)$, and $p(x_t|x_{t-1})$ are given as the **model**
- Add x_t, marginalise x_{t-1}, add y_t, and normalise. Detailed steps in the following slides (we'll write "left to right" to show the order of the operations)

© Massimo Piccardi, UTS

Marginal filtering density



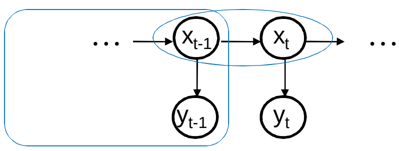
$$p(x_{t-1} \mid y_{1:t-1})\, p(x_t \mid x_{t-1}) = p(x_t, x_{t-1} \mid y_{1:t-1})$$

Easily proven with Markov property:

$$p(x_t, x_{t-1} \mid y_{1:t-1}) = p(x_t \mid x_{t-1}, y_{1:t-1})\, p(x_{t-1} \mid y_{1:t-1}) \quad \textit{Bayes}$$
$$= p(x_t \mid x_{t-1}, y_{1:t-1})\, p(x_{t-1} \mid y_{1:t-1}) \quad \textit{Markov}$$

© Massimo Piccardi, UTS 9

Marginal filtering density

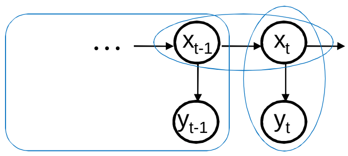


$$\int_{x_{t-1}} p(x_t, x_{t-1} \mid y_{1:t-1})\, dx_{t-1} = p(x_t \mid y_{1:t-1})$$

prediction

© Massimo Piccardi, UTS 10

Marginal filtering density

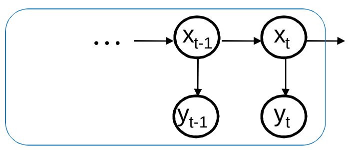


$$p(x_t \mid y_{1:t-1})\,p(y_t \mid x_t) = p(x_t, y_t \mid y_{1:t-1})$$

Easily proven with independence of observation given its state:

$$p(x_t, y_t \mid y_{1:t-1}) = p(y_t \mid x_t, y_{1:t-1})\,p(x_t \mid y_{1:t-1}) \quad \textit{Bayes}$$
$$= p(y_t \mid x_t)\,p(x_t \mid y_{1:t-1}) \quad \textit{independence}$$

Marginal filtering density



marginal filtering density ***likelihood*** ***prediction***

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

evidence (normalisation)

From Bayes inversion:

$$p(x_t \mid y_{1:t}) \equiv p(x_t \mid y_t, y_{1:t-1}) = p(x_t, y_t \mid, y_{1:t-1}) / p(y_t \mid y_{1:t-1}) \quad \textit{Bayes}$$

Marginal filtering density

- The evidence is, as usual, just the marginalisation of the numerator (over x_t in this case):

$$p(y_t \mid y_{1:t-1}) = \int_{x_t} p(x_t, y_t \mid y_{1:t-1})\, dx_t$$
$$= \int_{x_t} p(y_t \mid x_t)\, p(x_t \mid y_{1:t-1})\, dx_t$$

Marginal filtering density

- In summary:

$$p(x_t \mid y_{1:t-1}) = \int p(x_t \mid x_{t-1})\, p(x_{t-1} \mid y_{1:t-1})\, dx_{t-1}$$ prediction

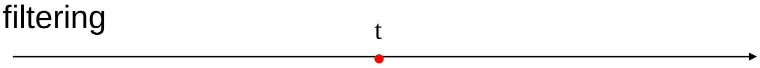
$$p(y_t, x_t \mid y_{1:t-1}) = p(y_t \mid x_t)\, p(x_t \mid y_{1:t-1})$$ add likelihood

$$p(y_t \mid y_{1:t-1}) = \int p(y_t \mid x_t)\, p(x_t \mid y_{1:t-1})\, dx_t$$ evidence

$$p(x_t \mid y_{1:t}) = \frac{p(y_t \mid x_t)\, p(x_t \mid y_{1:t-1})}{p(y_t \mid y_{1:t-1})}$$ Bayes inversion

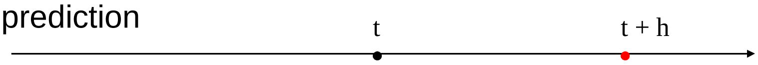
Other related marginals

- filtering



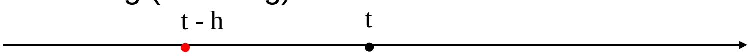
$\mathbf{p(x_t|y_{1:t})}$

- prediction



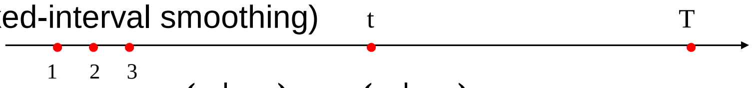
$\mathbf{p(x_{t+h}|y_{1:t})}$

- smoothing (fixed-lag)



$\mathbf{p(x_{t-h}|y_{1:t})}$

- offline/batch
(fixed-interval smoothing)



$\mathbf{p(x_1|y_{1:T}), \ldots\ p(x_T|y_{1:T})}$

© Massimo Piccardi, UTS 15

Posterior filtering density

- In brief:

$$p(x_{1:t} \mid y_{1:t-1}) \;= p(x_t \mid x_{t-1})\, p(x_{1:t-1} \mid y_{1:t-1}) \qquad \text{prediction}$$

$$p(y_t, x_{1:t} \mid y_{1:t-1}) = p(y_t \mid x_t)\, p(x_{1:t} \mid y_{1:t-1}) \qquad \text{add likelihood}$$

$$p(y_t \mid y_{1:t-1}) = \int p(y_t \mid x_t)\, p(x_{1:t} \mid y_{1:t-1})\, dx_{1:t} \qquad \text{evidence}$$

$$p(x_{1:t} \mid y_{1:t}) = \frac{p(y_t \mid x_t)\, p(x_{1:t} \mid y_{1:t-1})}{p(y_t \mid y_{1:t-1})} \qquad \text{Bayes inversion}$$

© Massimo Piccardi, UTS 16

The Kalman filter

- "Linear/Gaussian" assumptions: linear state and measurement models, Gaussian distributions for all random variables
- If $p(x_1)$ and noise distributions are assumed Gaussian, subsequent distributions remain Gaussian under linear transformations

  $X \sim N(\mu, \Sigma)$

  $Y = AX + K$

  $\rightarrow Y \sim N(A\mu + K, A\Sigma A^T)$

- An optimal solution exists (Swerling (1958), Kalman (1960) and Kalman and Bucy (1961))

© Massimo Piccardi, UTS 17

The Kalman filter

- The assumptions lead to the following equations (aka **linear dynamical system**):

$$x_t = Ax_{t-1} + v_{t-1}$$

$$y_t = Hx_t + n_t$$

x_t: state r.v. $\in R^n$
v_t: process noise r.v. $\in R^n$, independent, ~ **N(0,Q)**
A: state transition matrix (n x n)
yt: measurement r.v. $\in$ Rp
nt: measurement noise r.v. $\in$ Rp , independent, ~ **N(0,R)**
H: measurement matrix (p x n)
p(x1) = N(x1 | μ1, Σ1)

© Massimo Piccardi, UTS 18

Solution

- The target is the filtering distribution's pdf, $p(x_t \mid y_{1:t})$; however, since it is Gaussian, mean and covariance identify it fully
- In the classic literature on signal processing, the mean at time t is noted as $\hat{x}_t$, the covariance as P_t and called the *error covariance*
- The solution is obtained in two steps:
  - the **time update**, i.e. the application of the dynamics (system model): prediction $p(x_t \mid y_{1:t-1})$ its parameters are noted as $\hat{x}_t^-, P_t^-$ and called a-priori estimates
  - the **measurement update**, i.e. the use of the measurement (measurement model); a *correction* to the previous estimates yielding the a-posteriori estimates

Time update equations

- A-priori estimates:

$$\hat{x}_t^- = A\hat{x}_{t-1} \qquad \text{(mean)}$$

$$P_t^- = AP_{t-1}A^T + Q \qquad \text{(covariance)}$$

- Directly from linear transformation of Gaussian random variables
- Optimal estimate in the absence of measurement
- The process noise causes P to increase

Weighting

- A weighting matrix, K_t (n x p), called the *Kalman gain*:

$$K_t = P_t^- H^T \left(H P_t^- H^T + R \right)^{-1}$$

- We are just keeping on applying the rule for linear transformations
- K_t decides how much the a-priori estimates should be corrected by the k-th observation, y_t: the larger the measurement noise, R (uncertainty on the observation), the smaller the correction

© Massimo Piccardi, UTS

Measurement update equations

- A-posteriori estimates:

$$\hat{x}_t = \hat{x}_t^- + K_t \left(y_t - H\hat{x}_t^- \right) \quad \text{(mean)}$$

$$P_t = \left(I - K_t H \right) P_t^- \quad \text{(covariance)}$$

- $H\hat{x}_t^-$ is the predicted mean converted to the measurement space (predicted observation)
- $y_t - H\hat{x}_t^-$ is the difference between the actual and predicted observations (often called *innovation* or *measurement residual*)

© Massimo Piccardi, UTS

Kalman filter: parameters

- Historically, the parameters of the Kalman filter were not estimated from training sequences; rather, chosen based on some physical properties of the target
- Some canonical parametrisations for A are shown in the next few slides
- At large, we assume that the parameters are all time-invariant. However, many *adaptive Kalman filters* have been proposed in turn (adjusting the parameters based on the residual)

© Massimo Piccardi, UTS

Drift model

- State vector contains only the position of the target
- Constant position assumption – position changes only due to the effect of noise (*random walk*)
- **1-D example** (u: position)

$$x = [u] \qquad A = [1]$$

© Massimo Piccardi, UTS

Constant velocity model

- State vector contains position and velocity of the target
- Constant velocity assumption
- 1-D example (u: position; v: velocity)

$$x = \begin{bmatrix} u \\ v \end{bmatrix} \qquad A = \begin{bmatrix} 1 & \Delta_t \\ 0 & 1 \end{bmatrix}$$

© Massimo Piccardi, UTS

Constant acceleration model

- State vector contains position, velocity and acceleration of the target
- Constant acceleration assumption
- 1-D example (u: position; v: velocity; a: acceleration)

$$x = \begin{bmatrix} u \\ v \\ a \end{bmatrix} \qquad A = \begin{bmatrix} 1 & \Delta_t & 0 \\ 0 & 1 & \Delta_t \\ 0 & 0 & 1 \end{bmatrix}$$

© Massimo Piccardi, UTS

Periodic motion model

- State vector contains position and velocity of the target
- Periodic motion assumption: $d^2u/dt^2 = -cu$
- 1-D example (u: position; v: velocity)

$$x = \begin{bmatrix} u \\ v \end{bmatrix} \quad A = \begin{bmatrix} 1 & \Delta_t \\ -c\Delta_t & 1 \end{bmatrix}$$

© Massimo Piccardi, UTS

Example

- An example courtesy of Greg Welch and Gary Bishop, An Introduction to the Kalman Filter, ACM SIGGRAPH 2001 tutorial
- 2D data from a PC tablet
- A rich resource page at http://www.cs.unc.edu/~welch/kalman/
- A Java-based 1-D Kalman Filter Learning Tool

© Massimo Piccardi, UTS

Drift model

- Process model:

$$x = \begin{bmatrix} u_x \\ u_y \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad Q = \begin{bmatrix} Q_{xx} & 0 \\ 0 & Q_{yy} \end{bmatrix}$$

- Measurement model (y: measured position):

$$y = \begin{bmatrix} y_x \\ y_y \end{bmatrix} \qquad H = \begin{bmatrix} H_x & 0 \\ 0 & H_y \end{bmatrix} \qquad R = \begin{bmatrix} R_{xx} & 0 \\ 0 & R_{yy} \end{bmatrix}$$

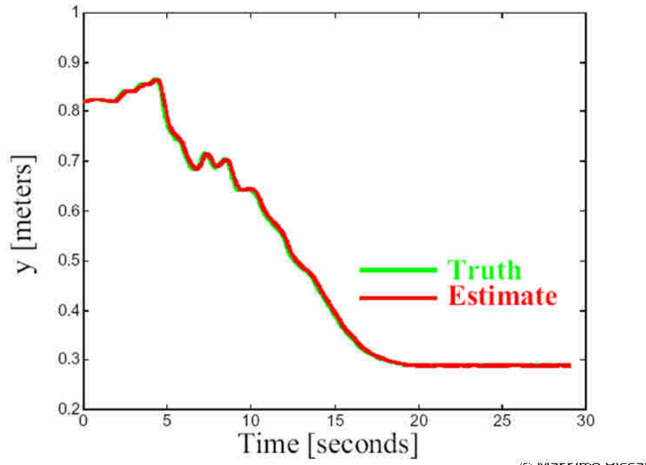
- Initialization:

$$\hat{x}_1 = H^{-1} y_1 \qquad P_0 = \begin{bmatrix} \varepsilon & 0 \\ 0 & \varepsilon \end{bmatrix}$$

© Massimo Piccardi, UTS

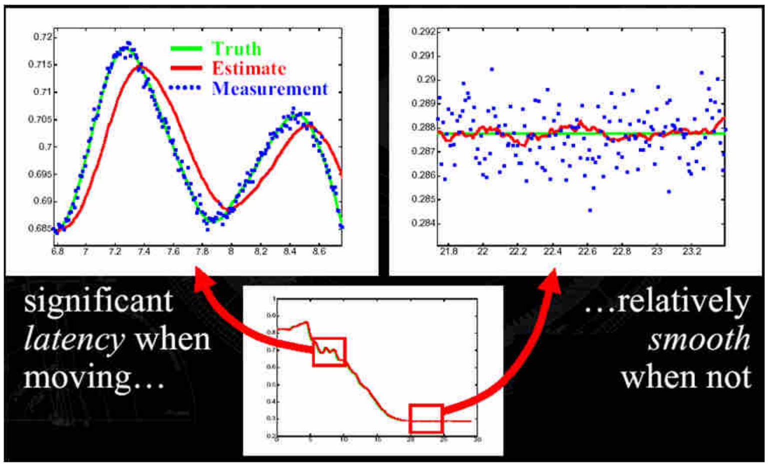
Results

- vertical (y) co-ordinate: first moving, then still



© Massimo Piccardi, UTS

Results: highlights



Constant velocity model

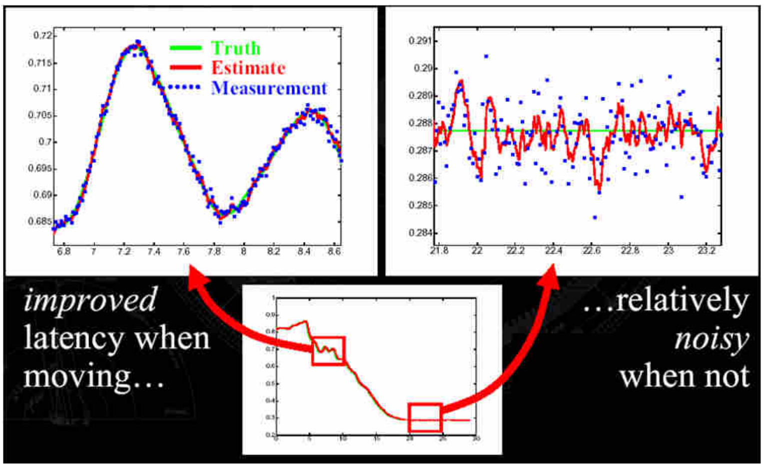
- Process model:

$$x = \begin{bmatrix} u_x \\ u_y \\ v_x \\ v_y \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Measurement model (same as before):

$$y = \begin{bmatrix} y_x \\ y_y \end{bmatrix} \qquad H = \begin{bmatrix} H_x & 0 & 0 & 0 \\ 0 & H_y & 0 & 0 \end{bmatrix}$$

Results: highlights



© Massimo Piccardi, UTS 33

Learning (MLE) of Kalman filter parameters

- Often the parameters of the Kalman filter, θ = {A, Q, H, R, μ_1, Σ_1) are chosen empirically
- Shumway and Stoffer (1982), Ghahramani and Hinton (1996) and Roweis and Ghahramani (1999) derived an EM algorithm to learn these parameters from one or more training sequences, $y_{1:T}$, such that likelihood $p(y_{1:T} \mid \theta)$ is maximised
- The needed expectations are computed with a forward-backward formula using the Kalman filter in the forward direction and Rauch's recursion in the backward direction

© Massimo Piccardi, UTS 34

Beyond linear/Gaussian

- The Gaussian modelling of distributions proper of the Kalman filter is restrictive is many cases
- If transformations are not linear, subsequent distributions would become non-Gaussian in any case
- Many other models have been proposed, the main of which are:
  - Extended Kalman filter
  - Grid filter
  - Unscented Kalman filter
  - Particle filters
- A nice punch line: the Kalman filter provides an exact solution for an approximate model; we may prefer an approximate solution for an exact model

© Massimo Piccardi, UTS

Particle filters

- In the following, we will focus on particle filters
- In these filters, various distributions are represented by weighted sets of samples (nicknamed *particles*)
- These techniques are more appropriately called ***sequential Monte Carlo*** methods
- The main target is again either the full posterior filtering density, $p(x_{1:t} \mid y_{1:t})$, or the marginal, $p(x_t \mid y_{1:t})$
- Distributions are not restricted to be Gaussian
- The model may be linear or not linear

© Massimo Piccardi, UTS

Monte Carlo methods

- Probabilistic methods where a distribution is simply represented by a set of its samples are commonly referred to as *Monte Carlo methods*
- These techniques were first adopted by physicists; the name was coined by N. C. Metropolis based on the gambling habits of a colleague's relative
- We see all the basic concepts in the following, and then extend them to particle filters (consequently called *sequential Monte Carlo* (SMC) methods)

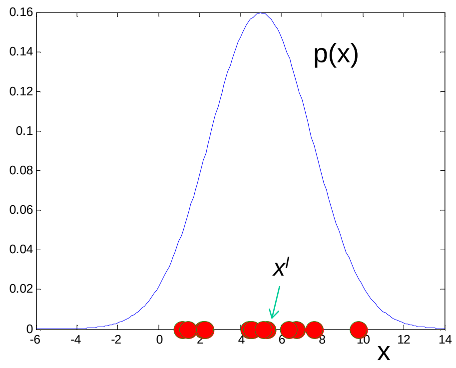
Distributions as sets of samples

- A generic density, p(x), can be approximated by a set of its samples, $\{x^l\}$, l=1…L:

$$p(x) \approx \frac{1}{L} \sum_{l=1}^{L} \delta\left(x - x^l\right)$$

- *Intuitively,* one can see that this density is normalised (integrates to 1 over x); each sample has a 1/L weight
- The $\{x^l\}$, l=1…L, are distinct with probability 1
- NB: this approximation requires us to be able to sample from p(x)

Example



- $L = 12$ samples from $p(x) = N(x \mid \mu = 5, \sigma = 2.5)$
- The radius of the samples is proportional to $1/L$

Notes Page

Importance sampling

- We may not be able to sample from p(x). In this case, an alternative sampling technique is offered by *importance sampling*
- In importance sampling, we sample from another distribution, q(x) (known as the *proposal distribution* or *importance density*)
- However, it is required that we can evaluate both p(x) and q(x) for any x
- For reasons of efficiency, q(x) should be as similar as possible to p(x)

Importance sampling

- Let us have a set of samples, $\{x^l\}_{l=1\ldots L}$, **from q(x)**
- For such samples to represent p(x), their weights cannot be uniform (1/L each), otherwise they would represent q(x)
- The weights must be proportionally increased where p(x) is denser than q(x), and vice versa
- Therefore:

$$p(x) \approx \frac{\sum_{l=1}^{L} \frac{p(x^l)}{q(x^l)} \delta(x - x^l)}{\sum_{l=1}^{L} \frac{p(x^l)}{q(x^l)}} = \sum_{l=1}^{L} w^l \delta(x - x^l)$$

© Massimo Piccardi, UTS

Importance sampling

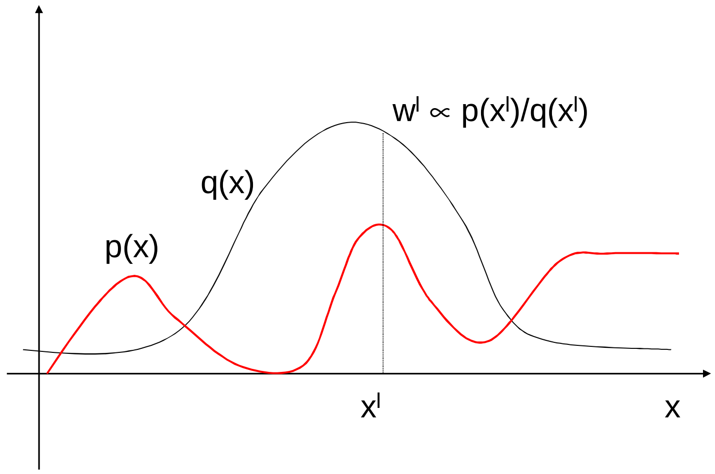
- In the previous slide, we have posed:

$$w^l = \frac{p(x^l)}{q(x^l)} \Bigg/ \sum_{l=1}^{L} \frac{p(x^l)}{q(x^l)}$$

- This definition encompasses also the case where q(x) is equal to p(x) itself; in such a case, w^l is simply 1/L
- The pair formed by the sample, x^l, and its weight, x^l , is known as *particle*

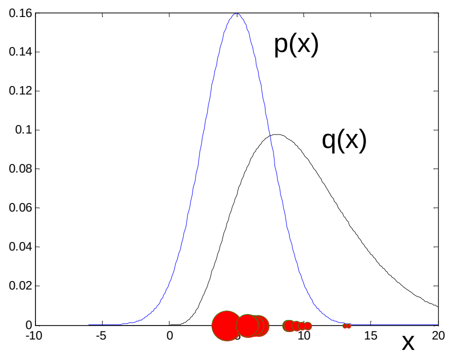
© Massimo Piccardi, UTS

Example



© Massimo Piccardi, UTS 43

Example



- L = 12 samples from proposal $q(x) = \mathcal{X}^2(x \mid k = 10)$
- Their radius is proportional to w^l: see how these weighted samples reflect the density of p, not q

Notes Page

© Massimo Piccardi, UTS 44

Expectations

An expectation based on the exact integral:

$$E[f(x)] = \int_x f(x)p(x)dx$$

can be estimated as (*Monte Carlo integration*):

$$E[f(x)] \approx \int_x f(x)\frac{1}{L}\sum_{l=1}^{L} \delta(x - x^l)dx \approx \frac{1}{L}\sum_{l=1}^{L} f(x^l)$$

where the x^l are L samples from p(x) (requiring that p(x) can be sampled)

Expectations and importance sampling

- If x is sampled from a proposal, q(x):

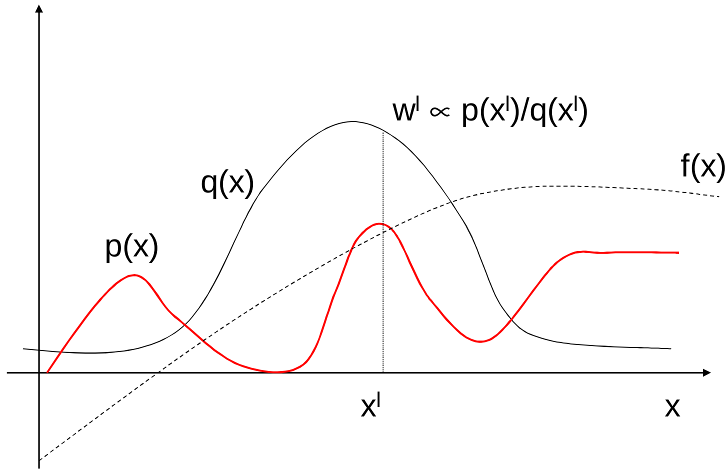
$$E[f(x)] = \int_X f(x)p(x)dx =$$

$$= \int_X f(x)\sum_{l=1}^{L} w^l \delta(x - x^l)dx \cong \sum_{l=1}^{L} w^l f(x^l)$$

where the x^l are sampled from q(x)

- For efficiency, q(x) should be large where product f(x)p(x) is large (having samples where f(x) or p(x) is close to zero would be a waste of samples)

Example



© Massimo Piccardi, UTS 47

Resampling

- A density p(x) expressed in sampled form via $\{x^l, w^l\}$, l:1…L, can be sampled at its turn: this is called *re-sampling*
- By drawing, for instance, L samples, x^{l*}, l:1…L, we obtain another approximation for p(x):

$$p(x) \approx \sum_{l=1}^{L} w^l \delta\left(x - x^l\right) \approx \frac{1}{L} \sum_{l=1}^{L} \delta\left(x - x^{l*}\right)$$

- The x^{l*} all have uniform weights, 1/L, and take value in the $\{x^l\}$ set; in the $\{x^{l*}\}$ set, some values may be repeated, some values of the original $\{x^l\}$ set may be missing

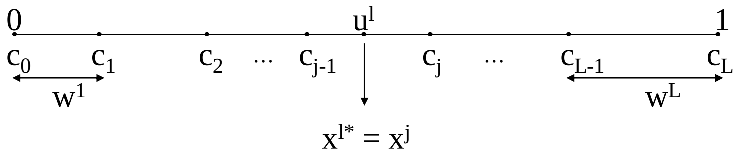
© Massimo Piccardi, UTS 48

Resampling

- Construct the *cdf* of the weighted set:

  $c_i = c_{i-1} + w^i \qquad c_0 = 0,\ i = 1 \ldots L$

- Sample a **uniform distribution in interval [0,1]** L times to obtain the u^l samples
- u^l falls in the j-th interval $\rightarrow x^{l*} = x^j$



- This is only one of the possible sampling schemes; see *systematic resampling* for greater efficiency

© Massimo Piccardi, UTS

Combining sampled and ordinary distributions

- Let us have **p(y,x)** where both y and x are continuous, multivariate random variables
- We factorise it as p(y|x)p(x), and represent p(x) in sampled form:

$$p(y,x) = p(y \mid x)p(x) \approx p(y \mid x)\sum_{l=1}^{L} w^l \delta(x - x^l) =$$

$$= \sum_{l=1}^{L} p(y \mid x) w^l \delta(x - x^l) = \sum_{l=1}^{L} p(y \mid x^l) w^l \delta(x - x^l) = \sum_{l=1}^{L} \tilde{w}^l \delta(x - x^l)$$

- The contribution of $p(y|x^l)$ can be incorporated into the weight, provided you keep in mind that it is a function of y; **such weights do not add up to 1**

© Massimo Piccardi, UTS

Combining sampled and ordinary distributions

- Let us now consider the marginal and the conditional
- The marginal, p(y), is just the sum of the weights:

$$p(y) = \int_x p(y,x)\,dx \approx \int_x \sum_{l=1}^{L} \tilde{w}^l \delta\left(x - x^l\right)dx = \sum_{l=1}^{L} \tilde{w}^l$$

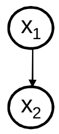
- The conditional, p(x|y), is just given by Bayes' theorem:

$$p(x \mid y) = \frac{p(y,x)}{p(y)} \approx \frac{\sum_{l=1}^{L} \tilde{w}^l \delta\left(x - x^l\right)}{\sum_{l=1}^{L} \tilde{w}^l} = \sum_{l=1}^{L} u^l \delta\left(x - x^l\right)$$

NB: weights u^l add up to 1!

Combining sampled and sampled distributions

- Let us have $\mathbf{p(x_2,x_1)}$ where both x_2 and x_1 are continuous, multivariate random variables
- We want to obtain a sample set from this joint density so as to represent it by a set of its samples
- We assume x_1 is the parent and x_2 its child, factorise as $p(x_2,x_1) = p(x_2|x_1)p(x_1)$ and use **ancestral sampling**:



$$x_1^l, x_2^l \equiv x_{1:2}^l \;:\; x_1^l \sim p(x_1),\; x_2^l \sim p\left(x_2 \mid x_1^l\right)$$

$$p(x_1, x_2) \approx \frac{1}{L}\sum_{l=1}^{L} \delta\left(x_1 - x_1^l\right)\delta\left(x_2 - x_2^l\right) \equiv \frac{1}{L}\sum_{l=1}^{L} \delta\left(x_{1:2} - x_{1:2}^l\right)$$

Combining sampled and sampled distributions

- The only variation w.r.t. usual ancestral sampling is that we assume that both sampling steps come from proposals, $q(x_1)$ and $q(x_2)$:

$$p(x_2, x_1) = p(x_2 \mid x_1) p(x_1) \approx$$

$$\approx \sum_{l=1}^{L} w_2^l \, \delta(x_2 - x_2^l) w_1^l \, \delta(x_1 - x_1^l) = \sum_{l=1}^{L} \tilde{w}^l \;\; \delta(x_{1:2} - x_{1:2}^l)$$

$$\tilde{w}^l = w_2^l \, w_1^l$$

© Massimo Piccardi, UTS 53

Combining sampled and sampled distributions

- NB: the new weights, $w^l_2 \, w^l_1$, do not add up to 1! If we want to normalise this approximated density, we'd need to divide the weights by their sum. However, if this is a partial step of a larger evaluation, we could just normalise at the end
- You can see very easily that marginal $p(x_2)$ is just!:

$$p(x_2) \approx \sum_{l=1}^{L} \tilde{w}^l \;\; \delta(x_2 - x_2^l)$$

© Massimo Piccardi, UTS 54

Particle filters

- Particle filters are based on sampled distributions and the properties presented so far
- Given that observations become available one by one, both the posterior filtering density, $p(x_{1:t} \mid y_{1:t})$, and its marginal, $p(x_t \mid y_{1:t})$, are estimated recursively
- The actual algorithms are many; here we'll see:
  - **Sequential Importance Sampling (SIS)**
  - **Sequential Importance Sampling with Resampling (SIS-R)**
  - **Sampling Importance Resampling (SIR)**

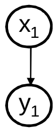
© Massimo Piccardi, UTS

SIS particle filter

- *Sequential importance sampling* (SIS) is the immediate sequential extension of importance sampling
- At every time frame t, L samples, x^l_t, are drawn out of a proposal distribution, $q(x^l_t \mid x^l_{t-1}, y_t)$. This proposal should take into account the previous sample and the current measurement
- The properties of sampled and ordinary distributions will allow us to estimate the posterior and its marginal recursively

© Massimo Piccardi, UTS

First step



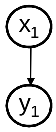
$$p(x_1) \approx \sum_{l=1}^{L} w'^{l}_{1}\, \delta\left(x_1 - x_1^l\right)$$

the samples are obtained from some $q(x_1 \mid y_1)$, or even just $q(x_1)$ (it's the initial "prediction"); we note their weights as w'^{l}_{1}

$$p(y_1, x_1) \approx \sum_{l=1}^{L} \left[w''^{l}_{1} = p\left(y_1 \mid x_1^l\right) w'^{l}_{1}\right] \delta\left(x_1 - x_1^l\right)$$

weights are then corrected by likelihood $p(y_1|x^l_1)$; we note such weights as w''^{l}_{1} and remind that they do not add up to 1 (they add up to $p(y_1)$)

First step (continued)

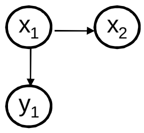


$$p(y_1) \approx \sum_{l=1}^{L} w''^{l}_{1}$$ we compute the evidence (just the sum of weights)

$$p(x_1 \mid y_1) \approx \sum_{l=1}^{L} \frac{w''^{l}_{1}}{\sum_{l=1}^{L} w''^{l}_{1}} \delta\left(x_1 - x_1^l\right) = \sum_{l=1}^{L} W_1^l\, \delta\left(x_1 - x_1^l\right)$$

the desired posterior at time t is computed by normalising the weights (weights W^l_1 obviously add up to 1 by construction)

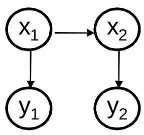
Second step



$$p(x_2, x_1 \mid y_1) = p(x_2 \mid x_1)\,p(x_1 \mid y_1) \approx \sum_{l=1}^{L} \left(w'^{l}_{2} = w^{l}_{2} W^{l}_{1}\right) \delta\left(x_{1:2} - x^{l}_{1:2}\right)$$

- prediction: we sample some $q(x_2|x^l_1, y_2)$, or $q(x_2|x^l_1)$ or even just $q(x_2)$, **once for each l** (see slides "combining sampled and sampled distributions"). We note the weights of the new samples as w^l_2 and we then combine them with the W^l_1 as $w'^l_2 = w^l_2\, W^l_1$
- pairs $\{x^l_1, x^l_2\} =: x^l_{1:2}$, $l = 1 \ldots L$, are a set of trajectories

Second step (continued)

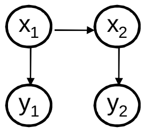


$$p(y_2, x_{1:2} \mid y_1) = p(y_2 \mid x_2)\,p(x_{1:2} \mid y_1) \approx$$

$$\approx \sum_{l=1}^{L} \left[w''^{l}_{2} = p\left(y_2 \mid x^{l}_{2}\right) w'^{l}_{2}\right] \delta\left(x_{1:2} - x^{l}_{1:2}\right)$$

weights are further corrected by likelihood $p(y_2|x^l_2)$

Second step (continued)



$$p(y_2 \mid y_1) \approx \sum_{l=1}^{L} w''^{l}_{2}$$ compute the evidence for normalisation

$$p(x_{1:2} \mid y_{1:2}) \approx \sum_{l=1}^{L} \frac{w''^{l}_{2}}{\sum_{l=1}^{L} w''^{l}_{2}} \delta\left(x_{1:2} - x^{l}_{1:2}\right) = \sum_{l=1}^{L} W^{l}_{2}\, \delta\left(x_{1:2} - x^{l}_{1:2}\right)$$

weights are normalised to add up to 1

© Massimo Piccardi, UTS 61

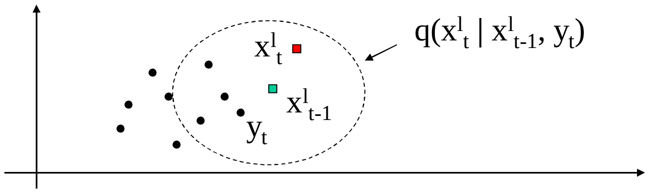
Generic step

	Recursive Bayesian estimation	**Sequential importance sampling**
prediction	$p(x_{1:t} \mid y_{1:t-1}) = p(x_t \mid x_{t-1}) p(x_{1:t-1} \mid y_{1:t-1})$	$p(x_{1:t} \mid y_{1:t-1}) \approx \sum_{l=1}^{L} w'^{l}_{t}\, \delta(x_{1:t} - x^{l}_{1:t})$
likelihood	$p(y_t, x_{1:t} \mid y_{1:t-1}) = p(y_t \mid x_t)\, p(x_{1:t} \mid y_{1:t-1})$	$p(y_t, x_{1:t} \mid y_{1:t-1}) \approx \sum_{l=1}^{L} w''^{l}_{t}\, \delta(x_{1:t} - x^{l}_{1:t})$
evidence	$p(y_t \mid y_{1:t-1}) = \int p(y_t \mid x_t)\, p(x_{1:t} \mid y_{1:t-1}) dx_{1:t}$	$p(y_t \mid y_{1:t-1}) \approx \sum_{l=1}^{L} w''^{l}_{t}$
Bayes inversion	$p(x_{1:t} \mid y_{1:t}) = \frac{p(y_t \mid x_t) p(x_{1:t} \mid y_{1:t-1})}{p(y_t \mid y_{1:t-1})}$	$p(x_{1:t} \mid y_{1:t}) \approx \sum_{l=1}^{L} W^{l}_{t}\, \delta(x_{1:t} - x^{l}_{1:t})$

© Massimo Piccardi, UTS 62

SIS particle filter: summary

- Sampling step:



- All the weight updates can be written like this in brief:

$$w_t^l = p(y_t \mid x_t^l)\frac{p(x_t^l \mid x_{t-1}^l)}{q(x_t^l \mid x_{t-1}^l, y_t)} W_{t-1}^l; \quad W_t^l = w_t^l \Big/ \sum_{l=1}^{L} w_t^l$$

- The marginal density is just!: $p(x_t \mid y_{1:t}) \approx \sum_{l=1}^{L} W_t^l \, \delta(x_t - x_t^l)$

SIS particle filter: algorithm

- SIS particle filter algorithm in a nutshell:

  At every time t,

  – draw the x^l_t from the current proposal distribution

  – update weights W^l_t based on formula

SIS particle filter: weight degeneracy

- Degeneracy of the weights: it has been proven that the variance of the W^l_t weights can only increase over time: after a few iterations, all but one particle will have weight $W^l_t = 0$
- Useful measure of degeneracy:

$$\hat{N}_{eff}^{-1} = \sum_{l=1}^{L} \left(W_t^l\right)^2$$

- Two practicable countermeasures:
  1. choice of a better proposal function that keeps more weights ≠ 0
  2. resampling

© Massimo Piccardi, UTS

Choice of proposal function

- It has been shown that the optimal proposal function to sample each x^l_t is:

$$p\left(x_t^l \mid x_{t-1}^l, y_t\right)$$

  since it minimises the degeneracy; yet, its use is not possible in most cases
- Often, the proposal function is taken as:

$$p\left(x_t^l \mid x_{t-1}^l\right)$$

  this simplifies the weight update formula greatly; yet, it ignores y_t in the sampling of x^l_t: it may sample away from useful directions

© Massimo Piccardi, UTS

Resampling

- Degeneracy can be monitored at every iteration; if > threshold, resampling is applied
- SIS particle filter *with resampling*:

  At every t,
  - draw the x^l_t from the current proposal distribution
  - update weights W^l_t based on formula
  - measure degeneracy: if > threshold, apply *resampling*

© Massimo Piccardi, UTS

SIR particle filter

- The *sampling importance resampling* (SIR) particle filter is a particle filter that uses $p(x^l_t \mid x^l_{t-1})$ as proposal and resamples at *every* time t
- SIR particle filter:

  At every t,
  - draw the x^l_t from $p(x^l_t \mid x^l_{t-1})$
  - compute weights W^l_t; formula simplifies because of proposal and previous weights being all equal to 1/L
  - resample

© Massimo Piccardi, UTS

SIR particle filter: issues

- The SIR particle filter uses a simple proposal distribution and avoids weight degeneracy by frequent resampling
- Yet, the proposal distribution ignores y_t in the sampling of x^l_t: it may sample away from useful directions
- Moreover, a very frequent resampling tends to create *sample impoverishment*: the samples with highest weights tend to be sampled more often and create x^l_t that are all equal; the degeneracy of the W^l_t weights is mollified at the cost of a degeneracy in the x^l_t samples

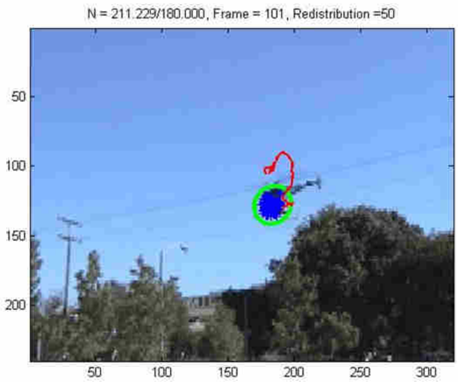
Other particle filters

- From Wikipedia (as of Apr 2012):
  - Auxiliary particle filter
  - Gaussian particle filter
  - Unscented particle filter
  - Gauss-Hermite particle filter
  - Cost Reference particle filter
  - Hierarchical/Scalable particle filter
  - Rao-Blackwellized particle filter
  - Rejection-sampling based optimal particle filter

Demo

Particle Filter Color Tracker from Matlab Central

Author: Sebastien Paris, 11 Dec 2007 (updated 28 Dec 2010)



Data association

- Data association is the process of assigning measurements to tracks (typically, the targets' predicted positions)
- Assumptions can be made about the number of targets (one, known etc)
- A matrix of costs can be established
- Association can be performed in a deterministic or probabilistic manner
- The most famous deterministic algorithm is the "Hungarian" algorithm (reviewed by Munkres in 1957)

Probabilistic data association (PDA)

- Single target, multiple observations: PDA filter (PDAF): in the update step, **all** observations are used for the update, weighted by their "membership" to the track:

$$p_i = dist_i \Big/ \sum_{i=1}^{N} dist_i$$

- Multiple targets: joint PDAF (JPDAF): avoids assigning the same observation more than once and adjusts weights jointly
- Many other algorithms: multiple hypothesis tracker (MHT), interacting multiple model PDAF (IMM-JPDAF), integrated PDA (IPDA) etc

Example papers

- Application: *tracking of visual objects*
- **Visual tracking with particle filters**:

  M. Isard, A. Blake, "CONDENSATION -- conditional density propagation for visual tracking," Int. J. Computer Vision, vol. 29, no. 1, pp. 5-28, 1998.

  3933 cites on Google Scholar as of April 2012
- **Multiple-target tracking and data association with EM**:

  H. Tao, H. S. Sawhney, R. Kumar, "Object Tracking with Bayesian Estimation of Dynamic Layer Representations," IEEE Trans. on Pattern Anal. and Machine Intell., vol. 24, no. 1, pp. 75-89, 2002.

References

Recursive Bayesian estimation:

- M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayesian Tracking," IEEE Trans. on Signal Processing, 50(2):174-188, February 2002
- S. Dablemont, Université catholique de Louvain, Introduction to particle filters, 2006 (slide set)
- K. Murphy, A tutorial on dynamic Bayesian networks, 2002 (slide set)

Kalman and particle filters:

- Materials from Greg Welch and Gary Bishop, University of North Carolina at Chapel Hill:
  - The Kalman Filter, http://www.cs.unc.edu/~welch/kalman/ (online resource)
  - An Introduction to the Kalman Filter, STC Lecture Series (slides)
  - An Introduction to the Kalman Filter, SIGGRAPH 2001 (slides and paper)
  - An Introduction to the Kalman Filter, TR95-041, University of North Carolina at Chapel Hill (tech rep)

References (2)

- Peter S. Maybeck, Stochastic models, estimation, and control. Volume 1, Academic Press, 1979
- Siome Klein Goldenstein, "A gentle Introduction to predictive filters," Revista de Informática Teórica e Aplicada - RITA, Volume 11, vol. 11, no. 1, 2004
- M S Arulampalam, S Maskell, N Gordon, and T Clapp, idem
- Arnaud Doucet, Adam M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," UBC Tech. Rep., 2008

Probabilistic data association:

- R. Collins, Intro to data association methods, CSE598G Vision-Based Tracking, Penn State University, 2006
- Kirubarajan, T, Bar-Shalom, Y., "Probabilistic data association techniques for target tracking in clutter," Proceedings of the IEEE, vol. 92, no. 3, Mar 2004, pp. 536 - 557
- I. Cox, "A Review of Statistical Data Association Techniques for Motion Correspondence," Int'l J. Computer Vision, vol. 10, no. 1, pp. 53-65, 1993
- Rasmussen, C., Hager, G.D., "Probabilistic Data Association Methods for Tracking Complex Visual Objects," PAMI(23), No. 6, June 2001, pp. 560-576