1

Inference and estimation in probabilistic time series models

David Barber, A. Taylan Cemgil and Silvia Chiappa

1.1 Time series

The term 'time series' refers to data that can be represented as a sequence. This includes for example financial data in which the sequence index indicates time, and genetic data (e.g. $ACATGC\ldots$) in which the sequence index has no temporal meaning. In this tutorial we give an overview of discrete-time probabilistic models, which are the subject of most chapters in this book, with continuous-time models being discussed separately in Chapters 4, 6, 11 and 17. Throughout our focus is on the basic algorithmic issues underlying time series, rather than on surveying the wide field of applications.

Defining a probabilistic model of a time series $y_{1:T} \equiv y_1, \ldots, y_T$ requires the specification of a joint distribution $p(y_{1:T})$.[1] In general, specifying all independent entries of $p(y_{1:T})$ is infeasible without making some statistical independence assumptions. For example, in the case of binary data, $y_t \in \{0, 1\}$, the joint distribution contains maximally $2^T - 1$ independent entries. Therefore, for time series of more than a few time steps, we need to introduce simplifications in order to ensure tractability. One way to introduce statistical independence is to use the probability of a conditioned on observed b

$$p(a|b) = \frac{p(a, b)}{p(b)}.$$

Replacing a with y_T and b with $y_{1:T-1}$ and rearranging we obtain $p(y_{1:T}) = p(y_T|y_{1:T-1})p(y_{1:T-1})$. Similarly, we can decompose $p(y_{1:T-1}) = p(y_{T-1}|y_{1:T-2})p(y_{1:T-2})$. By repeated application, we can then express the joint distribution as[2]

$$p(y_{1:T}) = \prod_{t=1}^{T} p(y_t|y_{1:t-1}).$$

This factorisation is consistent with the causal nature of time, since each factor represents a generative model of a variable conditioned on its past. To make the specification simpler, we can impose conditional independence by dropping variables in each factor conditioning set. For example, by imposing $p(y_t|y_{1:t-1}) = p(y_t|y_{t-m:t-1})$ we obtain the mth-order Markov model discussed in Section 1.2.

[1] To simplify the notation, throughout the tutorial we use lowercase to indicate both a random variable and its realisation.

[2] We use the convention that $y_{1:t-1} = \emptyset$ if $t < 2$. More generally, one may write $p_t(y_t|y_{1:t-1})$, as we generally have a different distribution at each time step. However, for notational simplicity we generally omit the time index.

Figure 1.1 Belief network representations of two time series models. (a) First-order Markov model $p(y_{1:4}) = p(y_4|y_3)p(y_3|y_2)p(y_2|y_1)p(y_1)$. (b) Second-order Markov model $p(y_{1:4}) = p(y_4|y_3, y_2)p(y_3|y_2, y_1)p(y_2|y_1)p(y_1)$.

A useful way to express statistical independence assumptions is to use a belief network graphical model which is a directed acyclic graph[3] representing the joint distribution

$$p(y_{1:N}) = \prod_{i=1}^{N} p(y_i|\text{pa}\,(y_i))\,,$$

where $\text{pa}\,(y_i)$ denotes the parents of y_i, that is the variables with a directed link to y_i. By limiting the parental set of each variable we can reduce the burden of specification. In Fig. 1.1 we give two examples of belief networks corresponding to a first- and second-order Markov model respectively, see Section 1.2. For the model $p(y_{1:4})$ in Fig. 1.1(a) and binary variables $y_t \in \{1, 2\}$ we need to specify only $1 + 2 + 2 + 2 = 7$ entries[4], compared to $2^4 - 1 = 15$ entries in the case that no independence assumptions are made.

Inference

Inference is the task of using a distribution to answer questions of interest. For example, given a set of observations $y_{1:T}$, a common inference problem in time series analysis is the use of the posterior distribution $p(y_{T+1}|y_{1:T})$ for the prediction of an unseen future variable y_{T+1}. One of the challenges in time series modelling is to develop computationally efficient algorithms for computing such posterior distributions by exploiting the independence assumptions of the model.

Estimation

Estimation is the task of determining a parameter θ of a model based on observations $y_{1:T}$. This can be considered as a form of inference in which we wish to compute $p(\theta|y_{1:T})$. Specifically, if $p(\theta)$ is a distribution quantifying our beliefs in the parameter values before having seen the data, we can use Bayes' rule to combine this prior with the observations to form a posterior distribution

$$\underbrace{p(\theta|y_{1:T})}_{\text{posterior}} = \frac{\underbrace{p(y_{1:T}|\theta)}_{\text{likelihood}}\underbrace{p(\theta)}_{\text{prior}}}{\underbrace{p(y_{1:T})}_{\text{marginal likelihood}}}.$$

The posterior distribution is often summarised by the maximum a posteriori (MAP) point estimate, given by the mode

$$\theta^{\text{MAP}} = \underset{\theta}{\text{argmax}}\; p(y_{1:T}|\theta)p(\theta).$$

[3] A directed graph is acyclic if, by following the direction of the arrows, a node will never be visited more than once.

[4] For example we need one specification for $p(y_1 = 1)$, with $p(y_1 = 2) = 1 - p(y_1 = 1)$ determined by normalisation. Similarly, we need two states for $p(y_2|y_1)$.

It can be computationally more convenient to use the log posterior,

$$\theta^{\text{MAP}} = \underset{\theta}{\text{argmax}} \log \left(p(y_{1:T}|\theta) p(\theta) \right),$$

where the equivalence follows from the monotonicity of the log function. When using a 'flat prior' $p(\theta) = \text{const.}$, the MAP solution coincides with the maximum likelihood (ML) solution

$$\theta^{\text{ML}} = \underset{\theta}{\text{argmax}}\, p(y_{1:T}|\theta) = \underset{\theta}{\text{argmax}} \log p(y_{1:T}|\theta).$$

In the following sections we introduce some popular time series models and describe associated inference and parameter estimation routines.

1.2 Markov models

Markov models (or Markov chains) are of fundamental importance and underpin many time series models [21]. In an mth order Markov model the joint distribution factorises as

$$p(y_{1:T}) = \prod_{t=1}^{T} p(y_t|y_{t-m:t-1}),$$

expressing the fact that only the previous m observations $y_{t-m:t-1}$ directly influence y_t. In a time-homogeneous model, the transition probabilities $p(y_t|y_{t-m:t-1})$ are time-independent.

1.2.1 Estimation in discrete Markov models

In a time-homogeneous first-order Markov model with discrete scalar observations $y_t \in \{1, \ldots, S\}$, the transition from y_{t-1} to y_t can be parameterised using a matrix θ, that is

$$\theta_{ji} \equiv p(y_t = j|y_{t-1} = i, \theta), \quad i, j \in \{1, \ldots, S\}.$$

Given observations $y_{1:T}$, maximum likelihood sets this matrix according to

$$\theta^{ML} = \underset{\theta}{\text{argmax}} \log p(y_{1:T}|\theta) = \underset{\theta}{\text{argmax}} \sum_t \log p(y_t|y_{t-1}, \theta).$$

Under the probability constraints $0 \leq \theta_{ji} \leq 1$ and $\sum_j \theta_{ji} = 1$, the optimal solution is given by the intuitive setting

$$\theta_{ji}^{ML} = \frac{n_{ji}}{T-1},$$

where n_{ji} is the number of transitions from i to j observed in $y_{1:T}$.

Alternatively, a Bayesian treatment would compute the parameter posterior distribution

$$p(\theta|y_{1:T}) \propto p(\theta) p(y_{1:T}|\theta) = p(\theta) \prod_{i,j} \theta_{ji}^{n_{ji}}.$$

In this case a convenient prior for θ is a Dirichlet distribution on each column $\theta_{:i}$ with hyperparameter vector $\alpha_{:i}$

$$p(\theta) = \prod_i \mathcal{DI}(\theta_{:i}|\alpha_{:i}) = \prod_i \frac{1}{Z(\alpha_{:i})} \prod_j \theta_{ji}^{\alpha_{ji}-1},$$

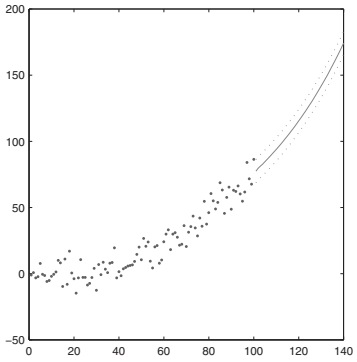


Figure 1.2 Maximum likelihood fit of a third-order AR model. The horizontal axis represents time, whilst the vertical axis the value of the time series. The dots represent the 100 observations $y_{1:100}$. The solid line indicates the mean predictions $\langle y \rangle_t, t > 100$, and the dashed lines $\langle y \rangle_t \pm \sqrt{r}$.

where $Z(\alpha_{:i}) = \int_0^1 \prod_j \theta_{ij}^{\alpha_{ji}-1} d\theta$. The convenience of this 'conjugate' prior is that it gives a parameter posterior that is also a Dirichlet distribution [15]

$$p(\theta|y_{1:T}) = \prod_i \mathcal{D}\mathcal{I}(\theta_{:i}|\alpha_{:i} + n_{:i}).$$

This Bayesian approach differs from maximum likelihood in that it treats the parameters as random variables and yields distributional information. This is motivated from the understanding that for a finite number of observations there is not necessarily a 'single best' parameter estimate, but rather a distribution of parameters weighted both by how well they fit the data and how well they match our prior assumptions.

1.2.2 Autoregressive models

A widely used Markov model of continuous scalar observations is the autoregressive (AR) model [2, 4]. An mth-order AR model assumes that y_t is a noisy linear combination of the previous m observations, that is

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_m y_{t-m} + \epsilon_t,$$

where $a_{1:m}$ are called the AR coefficients, and ϵ_t is an independent noise term commonly assumed to be zero-mean Gaussian with variance r (indicated with $\mathcal{N}(\epsilon_t|0, r)$). A so-called *generative form* for the AR model with Gaussian noise is given by[5]

$$p(y_{1:T}|y_{1:m}) = \prod_{t=m+1}^{T} p(y_t|y_{t-m:t-1}), \qquad p(y_t|y_{t-m:t-1}) = \mathcal{N}\Big(y_t \Big| \sum_{i=1}^{m} a_i y_{t-i}, r\Big).$$

Given observations $y_{1:T}$, the maximum likelihood estimate for the parameters $a_{1:m}$ and r is obtained by maximising with respect to a and r the log likelihood

$$\log p(y_{1:T}|y_{1:m}) = -\frac{1}{2r} \sum_{t=m+1}^{T} \Big(y_t - \sum_{i=1}^{m} a_i y_{t-i}\Big)^2 - \frac{T-m}{2} \log(2\pi r).$$

The optimal $a_{1:m}$ are given by solving the linear system

$$\sum_i a_i \sum_{t=m+1}^{T} y_{t-i} y_{t-j} = \sum_{t=m+1}^{T} y_t y_{t-j} \qquad \forall j.$$

[5] Note that the first m variables are not modelled.

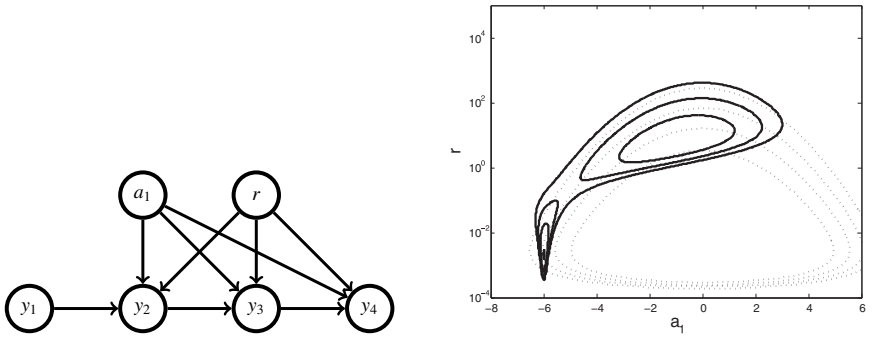


Figure 1.3 (a) Belief network representation of a first-order AR model with parameters $\theta = (a_1, r)$ (first four time steps). (b) Parameter prior $p(a_1, r)$ (light grey, dotted) and posterior $p(a_1, r|y_1 = 1, y_2 = -6)$ (black). The posterior describes two plausible explanations of the data: (i) the noise r is low and $a_1 \approx -6$, (ii) the noise r is high with a set of possible values for a_1 centred around zero.

which is readily solved using Gaussian elimination. The linear system has a Toeplitz form that can be more efficiently solved, if required, using the Levinson-Durbin method [9]. The optimal variance is then given by

$$r = \frac{1}{T-m} \sum_{t=m+1}^{T} \Big(y_t - \sum_{i=1}^{m} a_i y_{t-i} \Big)^2 .$$

The case in which y_t is multivariate can be handled by assuming that a_i is a matrix and ϵ_t is a vector. This generalisation is known as vector autoregression.

Example 1 We illustrate with a simple example how AR models can be used to estimate trends underlying time series data. A third-order AR model was fit to the set of 100 observations shown in Fig. 1.2 using maximum likelihood. A prediction for the mean $\langle y \rangle_t$ was then recursively generated as

$$\langle y \rangle_t = \begin{cases} \sum_{i=1}^{3} a_i \langle y \rangle_{t-i} & \text{for} \quad t > 100, \\ y_t & \text{for} \quad t \leq 100 \,. \end{cases}$$

As we can see (solid line in Fig. 1.2), the predicted means for time $t > 100$ capture an underlying trend in the time series.

Example 2 In a MAP and Bayesian approach, a prior on the AR coefficients can be used to define physical constraints (if any) or to regularise the system. Similarly, a prior on the variance r can be used to specify any knowledge about or constraint on the noise. As an example, consider a Bayesian approach to a first-order AR model in which the following Gaussian prior for a_1 and inverse Gamma prior for r are defined:

$$p(a_1) = \mathcal{N}(a_1 | 0, q) \,,$$
$$p(r) = \mathcal{IG}(r|\nu, \nu/\beta) = \exp\left[-(\nu+1)\log r - \frac{\nu}{\beta r} - \log\Gamma(\nu) + \nu \log \frac{\nu}{\beta} \right].$$

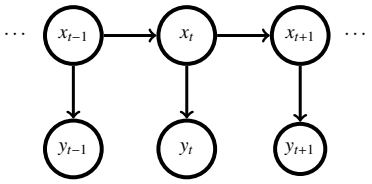


Figure 1.4 A first-order latent Markov model. In a hidden Markov model the latent variables $x_{1:T}$ are discrete and the observed variables $y_{1:T}$ can be either discrete or continuous.

Assuming that a_1 and r are a priori independent, the joint distribution is given by

$$p(a_1, r|y_{1:T}) \propto p(a_1)p(r)\prod_{t=2}^{T} p(y_t|y_{t-1}, a_1, r).$$

The belief network representation of this model is given in Fig. 1.3(a). For a numerical example, consider $T = 2$ and observations and hyperparameters given by

$$y_1 = 1, \quad y_2 = -6, \quad q = 1.2, \quad \nu = 0.4, \quad \beta = 100.$$

The parameter posterior, Fig. 1.3(b), takes the form

$$p(a_1, r|y_{1:2}) \propto \exp\left[-\left(\frac{\nu}{\beta} + \frac{y_2^2}{2}\right)\frac{1}{r} + y_1 y_2 \frac{a_1}{r} - \frac{1}{2}\left(\frac{y_1^2}{r} + \frac{1}{q}\right)a_1^2 - (\nu + 3/2)\log r\right],$$

As we can see, the posterior is multimodal, with each mode corresponding to a different interpretation: (i) The regression coefficient a_1 is approximately -6 and the noise is low. This solution gives a small prediction error. (ii) Since the prior for a_1 has zero mean, an alternative interpretation is that a_1 is centred around zero and the noise is high.

From this example we can make the following observations:

- Point estimates such as ML or MAP are not always representative of the solution.
- Even very simple models can lead to complicated posterior distributions.
- Variables that are independent *a priori* may become dependent *a posteriori*.
- Ambiguous data usually leads to a multimodal parameter posterior, with each mode corresponding to one plausible explanation.

1.3 Latent Markov models

In a latent Markov model, the observations $y_{1:T}$ are generated by a set of unobserved or 'latent' variables $x_{1:T}$. Typically, the latent variables are first-order Markovian and each observed variable y_t is independent from all other variables given x_t. The joint distribution thus factorises as[6]

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

where $p(x_t|x_{t-1})$ is called the 'transition' model and $p(y_t|x_t)$ the 'emission' model. The belief network representation of this latent Markov model is given in Fig. 1.4.

[6]This general form is also known as a state space model.

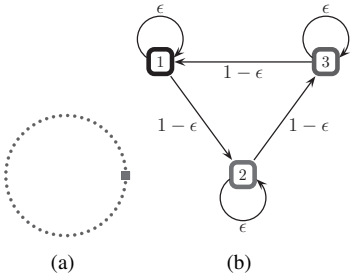


Figure 1.5 (a) Robot (square) moving sporadically with probability $1-\epsilon$ counter-clockwise in a circular corridor one location at a time. Small circles denote the S possible locations. (b) The state transition diagram for a corridor with $S = 3$ possible locations.

1.3.1 Discrete state latent Markov models

A well-known latent Markov model is the hidden Markov model[7] (HMM) [23] in which x_t is a scalar discrete variable ($x_t \in \{1, \ldots, S\}$).

Example Consider the following toy tracking problem. A robot is moving around a circular corridor and at any time occupies one of S possible locations. At each time step t, the robot stays where it is with probability ϵ, or moves to the next point in a counter-clockwise direction with probability $1-\epsilon$. This scenario, illustrated in Fig. 1.5, can be conveniently represented by an $S \times S$ matrix A with elements $A_{ji} = p(x_t = j | x_{t-1} = i)$. For example, for $S = 3$, we have

$$A = \epsilon \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + (1-\epsilon) \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}. \tag{1.1}$$

At each time step t, the robot sensors measure its position, obtaining either the correct location with probability w or a uniformly random location with probability $1-w$. This can be expressed formally as

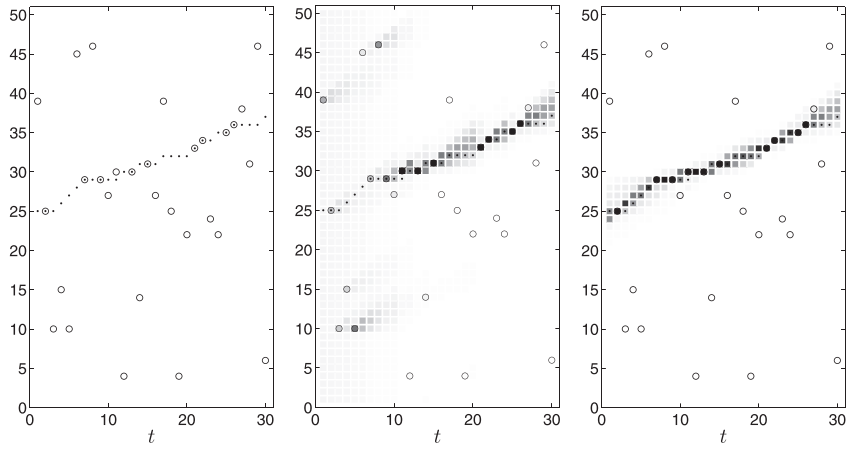
$$y_t | x_t \sim w\delta(y_t - x_t) + (1-w)\mathcal{U}(y_t | 1, \ldots, S),$$

where δ is the Kronecker delta function and $\mathcal{U}(y|1, \ldots, S)$ denotes the uniform distribution over the set of possible locations. We may parameterise $p(y_t|x_t)$ using an $S \times S$ matrix C with elements $C_{ui} = p(y_t = u | x_t = i)$. For $S = 3$, we have

$$C = w \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \frac{(1-w)}{3} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

A typical realisation $y_{1:T}$ from the process defined by this HMM with $S = 50$, $\epsilon = 0.5$, $T = 30$ and $w = 0.3$ is depicted in Fig. 1.6(a). We are interested in inferring the true locations of the robot from the noisy measured locations $y_{1:T}$. At each time t, the true location can be inferred from the so-called 'filtered' posterior $p(x_t|y_{1:t})$ (Fig. 1.6(b)), which uses measurements up to t; or from the so-called 'smoothed' posterior $p(x_t|y_{1:T})$ (Fig. 1.6(c)), which uses both past and future observations and is therefore generally more accurate. These posterior marginals are obtained using the efficient inference routines outlined in Section 1.4.

[7] Some authors use the terms hidden Markov model and state space model as synonymous [4]. We use the term HMM in a more restricted sense to refer to a latent Markov model where $x_{1:T}$ are discrete. The observations $y_{1:T}$ can be discrete or continuous.



(a) (b) (c)

Figure 1.6 Filtering and smoothing for robot tracking using a HMM with $S = 50$. (a) A realisation from the HMM example described in the text. The dots indicate the true latent locations of the robot, whilst the open circles indicate the noisy measured locations. (b) The squares indicate the filtering distribution at each time step t, $p(x_t|y_{1:t})$. This probability is proportional to the grey level with black corresponding to 1 and white to 0. Note that the posterior for the first time steps is multimodal, therefore the true position cannot be accurately estimated. (c) The squares indicate the smoothing distribution at each time step t, $p(x_t|y_{1:T} = y_{1:T})$. Note that, for $t < T$, we estimate the position retrospectively and the uncertainty is significantly lower when compared to the filtered estimates.

1.3.2 Continuous state latent Markov models

In continuous state latent Markov models, x_t is a multivariate continuous variable, $x_t \in \mathbb{R}^H$. For high-dimensional continuous x_t, the set of models for which operations such as filtering and smoothing are computationally tractable is severely limited. Within this tractable class, the linear dynamical system plays a special role, and is essentially the continuous analogue of the HMM.

Linear dynamical systems

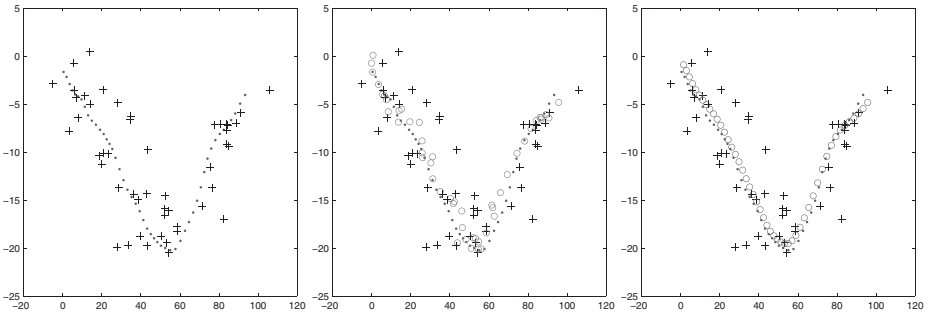
A linear dynamical system (LDS) on variables $x_{1:T}$, $y_{1:T}$ has the following form:

$$x_t = Ax_{t-1} + \bar{x}_t + \epsilon_t^x, \quad \epsilon_t^x \sim \mathcal{N}(\epsilon_t^x|0, Q), \quad x_1 \sim \mathcal{N}(x_1|\mu, P),$$
$$y_t = Cx_t + \bar{y}_t + \epsilon_t^y, \quad \epsilon_t^y \sim \mathcal{N}\left(\epsilon_t^y|0, R\right),$$

with transition matrix A and emission matrix C. The terms $\bar{x}_t$, $\bar{y}_t$ are often defined as $\bar{x}_t = Bz_t$ and $\bar{y}_t = Dz_t$, where z_t is a known input that can be used to control the system. The complete parameter set is therefore $\{A, B, C, D, Q, R, \mu, P\}$. As a generative model, the LDS is defined as

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t|Ax_{t-1} + \bar{x}_t, Q), \quad p(y_t|x_t) = \mathcal{N}(y_t|Cx_t + \bar{y}_t, R).$$

Example As an example scenario that can be modelled using an LDS, consider a moving object with position, velocity and instantaneous acceleration at time t given respectively by q_t, v_t and a_t. A discrete time description of the object dynamics is given by Newton's laws (see for example [11])



(a) (b) (c)

Figure 1.7 Tracking an object undergoing Newtonian dynamics in a two-dimensional space, Eq. (1.2). (a) The dots indicate the true latent positions of the object at each time t, $q_{1,t}$ (horizontal axis) and $q_{2,t}$ (vertical axis) (the time label is not shown). The crosses indicate the noisy observations of the latent positions. (b) The circles indicate the mean of the filtered latent positions $\int q_t p(q_t|y_{1:t})\mathrm{d}q_t$. (c) The circles indicate the mean of the smoothed latent positions $\int q_t p(q_t|y_{1:T})\mathrm{d}q_t$.

$$\underbrace{\begin{pmatrix} q_t \\ v_t \end{pmatrix}}_{x_t} = \underbrace{\begin{pmatrix} I & T_s I \\ 0 & I \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} q_{t-1} \\ v_{t-1} \end{pmatrix}}_{x_{t-1}} + \underbrace{\begin{pmatrix} \frac{1}{2}T_s^2 I \\ T_s I \end{pmatrix}}_{B} \underbrace{a_t}_{a_t}, \tag{1.2}$$

where I is the 3×3 identity matrix and T_s is the sampling period. In tracking applications, we are interested in inferring the true position q_t and velocity v_t of the object from limited noisy information. For example, in the case that we observe only the noise-corrupted positions, we may write

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

where $\bar{a}$ is the acceleration mean, $Q = B\Sigma B^{\mathsf{T}}$ with Σ being the acceleration covariance, $C = (I\ 0)$, and R is the covariance of the position noise. We can then track the position and velocity of the object using the filtered density $p(x_t|y_{1:t})$. An example with two-dimensional positions is shown in Fig. 1.7.

AR model as an LDS

Many popular time series models can be cast into a LDS form. For example, the AR model of Section 1.2.2 can be formulated as

$$\underbrace{\begin{pmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-m+1} \end{pmatrix}}_{x_t} = \underbrace{\begin{pmatrix} a_1 & a_2 & \dots & a_m \\ 1 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} y_{t-1} \\ y_{t-2} \\ \vdots \\ y_{t-m} \end{pmatrix}}_{x_{t-1}} + \underbrace{\begin{pmatrix} \epsilon_t \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{\epsilon_t^x},$$

$$y_t = \underbrace{\begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}}_{C} x_t + \epsilon_t^y,$$

where $\epsilon_t^x \sim \mathcal{N}(\epsilon_t^x|0, \text{diag}(r, 0, \ldots, 0))$, $\epsilon_t^y \sim \mathcal{N}\left(\epsilon_t^y|0, 0\right)$, the initial mean μ is set to the first m observations, and $P = 0$. This shows how to transform an mth-order Markov model into a constrained first-order latent Markov model. Many other related AR models and extensions can also be cast as a latent Markov model. This is therefore a very general class of models for which inference is of particular interest.

1.4 Inference in latent Markov models

In this section we derive computationally efficient methods for computing posterior distributions in latent Markov models. We assume throughout that x_t is discrete, though the recursions hold more generally on replacing summation with integration for those components of x_t that are continuous.

1.4.1 Filtering $p(x_t|y_{1:t})$

In filtering,[8] the aim is to compute the distribution of the latent variable x_t given all observations up to time t. This can be expressed as

$$p(x_t|y_{1:t}) = p(x_t, y_{1:t})/p(y_{1:t}).$$

The normalising term $p(y_{1:t})$ is the likelihood, see Section 1.4.2, and $\alpha(x_t) \equiv p(x_t, y_{1:t})$ can be computed by a 'forward' recursion

$$\begin{aligned}\alpha(x_t) &= p(y_t|x_t, \cancel{y_{1:t-1}})p(x_t, y_{1:t-1}) \\ &= p(y_t|x_t)\sum_{x_{t-1}} p(x_t|x_{t-1}, \cancel{y_{1:t-1}})p(x_{t-1}, y_{1:t-1}) \\ &= p(y_t|x_t)\sum_{x_{t-1}} p(x_t|x_{t-1})\alpha(x_{t-1}),\end{aligned} \tag{1.3}$$

where the cancellations follow from the conditional independence assumptions of the model. The recursion is initialised with $\alpha(x_1) = p(y_1|x_1)p(x_1)$. To avoid numerical over/underflow problems, it is advisable to work with $\log \alpha(x_t)$. If only the conditional distribution $p(x_t|y_{1:t})$ is required (not the joint $p(x_t, y_{1:t})$) a numerical alternative to using the logarithm is to form a recursion for $p(x_t|y_{1:t})$ directly by normalising $\alpha(x_t)$ at each stage.

1.4.2 The likelihood

The likelihood can be computed using the filtered results as

$$p(y_{1:t}) = \sum_{x_t} \alpha(x_t), \quad \text{and} \quad p(y_{1:T}) = \sum_{x_T} \alpha(x_T).$$

Maximum likelihood parameter learning can be carried out by the expectation maximisation algorithm, known in the HMM context as the Baum-Welch algorithm [23], see also Section 1.5.1.

[8] The term 'filtering' is somewhat a misnomer since in signal processing this term is reserved for a convolution operation. However, for linear systems, it turns out that state estimation is a linear function of past observations and can indeed be computed by a convolution, partially justifying the use of the term.

1.4.3 Smoothing $p(x_{1:T}|y_{1:T})$

The smoothing distribution is the joint distribution $p(x_{1:T}|y_{1:T})$. Typically we are interested in marginals such as $p(x_t|y_{1:T})$, which gives an estimate of x_t based on all observations. There are two main approaches to computing $p(x_t|y_{1:T})$, namely the parallel and the sequential methods described below.

Parallel smoothing $p(x_t|y_{1:T})$

In parallel smoothing, the posterior $\gamma(x_t) \equiv p(x_t|y_{1:T})$ is separated into contributions from the past and future

$$\gamma(x_t) \propto \underbrace{p(x_t, y_{1:t})}_{\text{past}} \underbrace{p(y_{t+1:T}|x_t, \cancel{y_{1:t}})}_{\text{future}} = \alpha(x_t)\beta(x_t). \tag{1.4}$$

The term $\alpha(x_t)$ is obtained from the filtering recursion (1.3). The terms $\beta(x_t)$ can be obtained by the following 'backward' recursion

$$\begin{aligned}\beta(x_t) &= \sum_{x_{t+1}} p(y_{t+1}|\cancel{y_{t+2:T}}, \cancel{x_t}, x_{t+1})p(y_{t+2:T}, x_{t+1}|x_t) \\ &= \sum_{x_{t+1}} p(y_{t+1}|x_{t+1})p(y_{t+2:T}|, \cancel{x_t}, x_{t+1})p(x_{t+1}|x_t) \\ &= \sum_{x_{t+1}} p(y_{t+1}|x_{t+1})p(x_{t+1}|x_t)\beta(x_{t+1}),\end{aligned} \tag{1.5}$$

with $\beta(x_T) = 1$. As for filtering, working in log space for β is recommended to avoid numerical difficulties.[9] The $\alpha - \beta$ recursions are independent and may therefore be run in parallel. These recursions also are called the *forward-backward algorithm*.

Sequential smoothing $p(x_t|y_{1:T})$

In sequential smoothing, we form a direct recursion for the smoothed posterior as

$$\gamma(x_t) = \sum_{x_{t+1}} p(x_t, x_{t+1}|y_{1:T}) = \sum_{x_{t+1}} p(x_t|x_{t+1}, y_{1:t}, \cancel{y_{t+1:T}})\gamma(x_{t+1}), \tag{1.6}$$

with $\gamma(x_T) \propto \alpha(x_T)$. The term $p(x_t|x_{t+1}, y_{1:t})$ is computed from filtering using

$$p(x_t|x_{t+1}, y_{1:t}) \propto p(x_{t+1}|x_t)\alpha(x_t),$$

where the proportionality constant is found by normalisation. The procedure is sequential since we need to complete the α recursions before starting the γ recursions. This technique is also termed the *Rauch–Tung–Striebel* smoother[10] and is a so-called correction smoother since it 'corrects' the filtered results. Interestingly, this correction process uses only filtered information. That is, once the filtered results have been computed, the observations $y_{1:T}$ are no longer needed. One can also view the γ recursion as a form of dynamics reversal, as if we were reversing the direction of the hidden-to-hidden arrows in the model.

[9] If only posterior distributions are required, one can also perform local normalisation at each stage, since only the relative magnitude of the components of β is of importance.

[10] It is most common to use this terminology for the continuous latent variable case.

Computing the pairwise marginal $p(x_t, x_{t+1}|y_{1:T})$

To implement algorithms for parameter learning, we often require terms such as $p(x_t, x_{t+1}|y_{1:T})$, see Section 1.5.1. These can be obtained from the sequential approach using

$$p(x_t, x_{t+1}|y_{1:T}) = p(x_t|x_{t+1}, y_{1:t})p(x_{t+1}|y_{1:T}),$$

or from the parallel approach using

$$p(x_t, x_{t+1}|y_{1:T}) \propto \beta(x_{t+1})p(y_{t+1}|x_{t+1})p(x_{t+1}|, x_t)\alpha(x_t). \tag{1.7}$$

1.4.4 Prediction $p(y_{t+1}|y_{1:t})$

Prediction is the problem of computing the posterior density $p(y_\tau|y_{1:t})$ for any $\tau > t$. For example, the distribution of the next observation may be found using

$$p(y_{t+1}|y_{1:t}) = \sum_{x_{t+1}} p(y_{t+1}|x_{t+1})p(x_{t+1}|y_{1:t}) = \sum_{x_{t+1}} p(y_{t+1}|x_{t+1}) \sum_{x_t} p(x_{t+1}|x_t)p(x_t|y_{1:t}).$$

1.4.5 Interpolation

Interpolation is the problem of estimating a set of missing observations given past and future data. This can be achieved using

$$p(y_\tau|y_{1:\tau-1}, y_{\tau+1:T}) \propto \sum_{x_\tau} p(y_\tau|x_\tau)p(x_\tau|y_{1:\tau-1})p(y_{\tau+1:T}|x_\tau).$$

1.4.6 Most likely latent trajectory

The most likely latent trajectory that explains the observations is given by

$$x^*_{1:T} = \operatorname*{argmax}_{x_{1:T}} p(x_{1:T}|y_{1:T}).$$

In the literature $x^*_{1:T}$ is also called the *Viterbi path*. Since $y_{1:T}$ is known, $x^*_{1:T}$ is equivalent to $\operatorname*{argmax}_{x_{1:T}} p(x_{1:T}, y_{1:T})$. By defining $\delta(x_t) \equiv \max_{x_{1:t-1}} p(x_{1:t}, y_{1:t})$, the most likely trajectory can be obtained with the following algorithm:

$$\delta(x_1) = p(x_1, y_1), \quad \delta(x_t) = p(y_t|x_t, y_{t-k+1:t}) \max_{x_{t-1}} p(x_t|x_{t-1})\delta(x_{t-1}) \quad \text{for } t = 2, \ldots, T,$$
$$\psi(x_t) = \operatorname*{argmax}_{x_{t-1}} p(x_t|x_{t-1})\delta(x_{t-1}) \quad \text{for } t = 2, \ldots, T,$$
$$x^*_T = \operatorname*{argmax}_{x_T} \delta(x_T), \quad x^*_t = \psi(x^*_{t+1}) \quad \text{for } t = T-1, \ldots, 1,$$

where the recursion for $\delta(x_t)$ is obtained analogously to the recursion (1.3) by replacing the sum with the max operator.

1.4.7 Inference in the linear dynamical system

Inference in the LDS has a long history and widespread applications ranging from tracking and control of ballistic projectiles to decoding brain signals.[11] The filtered and smoothed

[11] The LDS and associated filtering algorithm was proposed by Kalman in the late 1950s [14] based on least squares estimates. It is interesting to note that the method also appeared almost concurrently in the Russian literature, in a form that is surprisingly similar to the modern approach in terms of Bayes recursions [25]. Even earlier in the 1880s, Thiele defined the LDS and associated filtering and smoothing recursions [16].

posterior marginals, can be computed through conditioning and marginalisation of Gaussian distributions. The key results required for algebraic manipulation of Gaussians are stated below.

Gaussian conditioning, marginalisation and linear transformation

A multivariate Gaussian distribution is defined in the so-called moment form by

$$p(x) = \mathcal{N}(x|\mu, \Sigma) \equiv \frac{1}{\sqrt{\det 2\pi\Sigma}} e^{-\frac{1}{2}(x-\mu)^\mathsf{T}\Sigma^{-1}(x-\mu)},$$

where μ is the mean vector, and Σ is the covariance matrix.

Consider a vector z partitioned into two subvectors x and y,

$$z = \begin{pmatrix} x \\ y \end{pmatrix},$$

and a Gaussian distribution $\mathcal{N}(z|\mu, \Sigma)$ with corresponding partitioned mean and covariance

$$\mu = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{pmatrix}, \quad \Sigma_{yx} \equiv \Sigma_{xy}^\mathsf{T}.$$

The distribution of x conditioned on y is then given by

$$p(x|y) = \mathcal{N}\left(x\middle|\mu_x + \Sigma_{xy}\Sigma_{yy}^{-1}\left(y - \mu_y\right), \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}\right), \tag{1.8}$$

whilst the marginal distribution of x is given by $p(x) = \mathcal{N}(x|\mu_x, \Sigma_{xx})$.

A linear transformation $y = Ax$ of a Gaussian random variable x, with $p(x) = \mathcal{N}(x|\mu, \Sigma)$, is Gaussian with $p(y) = \mathcal{N}\left(y|A\mu, A\Sigma A^\mathsf{T}\right)$.

Filtering: predictor-corrector method

For continuous x, the analogue of recursion (1.3) is[12]

$$p(x_t|y_{1:t}) \propto p(x_t, y_t|y_{1:t-1}) = p(y_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}). \tag{1.9}$$

Since Gaussians are closed under multiplication and integration, the filtered distribution is also a Gaussian. This means that we may represent $p(x_t|y_{1:t}) = \mathcal{N}(x_t|f_t, F_t)$ and the filtered recursion translates into update formulae for the mean f_t and covariance F_t. One can derive these updates by carrying out the integration in Eq. (1.9). However, this is tedious and a shortcut is to use the linear transformation and conditioning results above. Specifically, let $\langle x|y\rangle$ denote expectation with respect to a distribution $p(x|y)$, and let $\Delta x \equiv x - \langle x\rangle$. By using the transition and emission models

$$x_t = Ax_{t-1} + \bar{x}_t + \epsilon_t^x, \qquad y_t = Cx_t + \bar{y}_t + \epsilon_t^y,$$

we obtain

$$\langle x_t|y_{1:t-1}\rangle = Af_{t-1} + \bar{x}_t, \quad \langle y_t|y_{1:t-1}\rangle = C\langle x_t|y_{1:t-1}\rangle + \bar{y}_t,$$

[12] With $\int_x$ we indicate the integral with respect to the variable x.

Algorithm 1.1 LDS forward pass. Compute the filtered posteriors $p(x_t|y_{1:t}) \equiv \mathcal{N}(f_t, F_t)$ for a LDS with parameters $\theta_t = A_t, C_t, Q_t, R_t, \bar{x}_t, \bar{y}_t$. The log-likelihood $L = \log p(y_{1:T})$ is also returned.

$\{f_1, F_1, p_1\} = \text{LDSFORWARD}(0, 0, y_1; \theta_1)$
$L \leftarrow \log p_1$
for $t \leftarrow 2, T$ **do**
  $\{f_t, F_t, p_t\} = \text{LDSFORWARD}(f_{t-1}, F_{t-1}, y_t; \theta_t)$
  $L \leftarrow L + \log p_t$
end for
function LDSFORWARD $(f, F, y; \theta)$
$\mu_x \leftarrow Af + \bar{x}$
$\mu_y \leftarrow C\mu_x + \bar{y}$
$\Sigma_{xx} \leftarrow AFA^\mathsf{T} + Q$
$\Sigma_{yy} \leftarrow C\Sigma_{xx}C^\mathsf{T} + R$
$\Sigma_{yx} \leftarrow C\Sigma_{xx}$ — $\Sigma_{yx}^\mathsf{T}\Sigma_{yy}^{-1}$ is termed the Kalman gain matrix
$f' \leftarrow \mu_x + \Sigma_{yx}^\mathsf{T}\Sigma_{yy}^{-1}\left(y - \mu_y\right)$ — updated mean
$F' \leftarrow \Sigma_{xx} - \Sigma_{yx}^\mathsf{T}\Sigma_{yy}^{-1}\Sigma_{yx}$ — updated covariance
$p' \leftarrow \exp\left(-\frac{1}{2}\left(y - \mu_y\right)^\mathsf{T}\Sigma_{yy}^{-1}\left(y - \mu_y\right)\right) / \sqrt{\det 2\pi\Sigma_{yy}}$ — likelihood contribution
return f', F', p'

and

$$\left\langle \Delta x_t \Delta x_t^\mathsf{T} | y_{1:t-1} \right\rangle = AF_{t-1}A^\mathsf{T} + Q, \quad \left\langle \Delta y_t \Delta x_t^\mathsf{T} | y_{1:t-1} \right\rangle = C\left(AF_{t-1}A^\mathsf{T} + Q\right),$$
$$\left\langle \Delta y_t \Delta y_t^\mathsf{T} | y_{1:t-1} \right\rangle = C\left(AF_{t-1}A^\mathsf{T} + Q\right)C^\mathsf{T} + R.$$

By conditioning $p(x_t, y_t|y_{1:t-1})$ on y_t using the formula (1.8), we obtain a Gaussian distribution with mean f_t and covariance F_t given by

$$f_t = \langle x_t|y_{1:t-1}\rangle + \left\langle \Delta x_t \Delta y_t^\mathsf{T} | y_{1:t-1} \right\rangle \left\langle \Delta y_t \Delta y_t^\mathsf{T} | y_{1:t-1} \right\rangle^{-1} \left(y_t - \langle y_t|y_{1:t-1}\rangle\right),$$
$$F_t = \left\langle \Delta x_t \Delta x_t^\mathsf{T} | y_{1:t-1} \right\rangle - \left\langle \Delta x_t \Delta y_t^\mathsf{T} | y_{1:t-1} \right\rangle \left\langle \Delta y_t \Delta y_t^\mathsf{T} | y_{1:t-1} \right\rangle^{-1} \left\langle \Delta y_t \Delta x_t^\mathsf{T} | y_{1:t-1} \right\rangle.$$

The resulting recursion is summarised in Algorithm 1.1, generalised to time-dependent noise means $\bar{x}_t, \bar{y}_t$ and time-dependent transition and emission noise covariances.

Algebraically the updates generate a symmetric covariance F_t although, numerically, symmetry can be lost. This can be corrected by either including an additional symmetrisation step, or by parameterising the covariance using a square root approach [22]. A detailed discussion regarding the numerical stability of various representations is given in [26].

Smoothing: Rauch–Tung–Striebel/correction method

For reasons of numerical stability, the most common approach to smoothing is based on the sequential approach, for which the continuous analogue of Eq. (1.6) is

$$p(x_t|y_{1:T}) \propto \int_{x_{t+1}} p(x_t|y_{1:t}, \cancel{y_{t+1:T}}, x_{t+1}) p(x_{t+1}|y_{1:T}).$$

Due to the closure properties of Gaussians, we may assume $p(x_t|y_{1:T}) = \mathcal{N}(x_t|g_t, G_t)$ and our task is to derive update formulae for the mean g_t and covariance G_t. Rather than

long-handed integration, as in the derivation of the filtering updates, we can make use of some algebraic shortcuts. We note that $p(x_t|y_{1:t}, x_{t+1})$ can be found by first computing $p(x_t, x_{t+1}|y_{1:t})$ using the linear transition model, and then conditioning $p(x_t, x_{t+1}|y_{1:t})$ on x_{t+1}. Given that $p(x_t|y_{1:t})$ has mean f_t and covariance F_t, we obtain

$$\langle x_{t+1}|y_{1:t}\rangle = Af_t, \quad \left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}|y_{1:t}\right\rangle = F_t A^\mathsf{T}, \quad \left\langle \Delta x_{t+1}\Delta x_{t+1}^\mathsf{T}|y_{1:t}\right\rangle = AF_tA^\mathsf{T} + Q.$$

Therefore $p(x_t|y_{1:t}, x_{t+1})$ has mean

$$\langle x_t\rangle + \left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}\right\rangle \left\langle \Delta x_{t+1}\Delta x_{t+1}^\mathsf{T}\right\rangle^{-1} (x_{t+1} - \langle x_{t+1}\rangle) \tag{1.10}$$

and covariance

$$\overleftarrow{\Sigma}_t \equiv \left\langle \Delta x_t \Delta x_t^\mathsf{T}\right\rangle - \left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}\right\rangle \left\langle \Delta x_{t+1}\Delta x_{t+1}^\mathsf{T}\right\rangle^{-1} \left\langle \Delta x_{t+1}\Delta x_t^\mathsf{T}\right\rangle, \tag{1.11}$$

where the averages are conditioned on the observations $y_{1:t}$. Equations (1.10) and (1.11) are equivalent to a reverse-time linear system

$$x_t = \overleftarrow{A}_t x_{t+1} + \overleftarrow{m}_t + \overleftarrow{\eta}_t,$$

where

$$\overleftarrow{A}_t \equiv \left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}\right\rangle \left\langle \Delta x_{t+1}\Delta x_{t+1}^\mathsf{T}\right\rangle^{-1}, \quad \overleftarrow{m}_t \equiv \langle x_t\rangle - \left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}\right\rangle \left\langle \Delta x_{t+1}\Delta x_{t+1}^\mathsf{T}\right\rangle^{-1} \langle x_{t+1}\rangle,$$

and $\overleftarrow{\eta}_t \sim \mathcal{N}\left(\overleftarrow{\eta}_t|0, \overleftarrow{\Sigma}_t\right)$. The statistics of $p(x_t|y_{1:T})$ then follow from the linear transformation

$$g_t = \overleftarrow{A}_t g_{t+1} + \overleftarrow{m}_t, \quad G_t = \overleftarrow{A}_t G_{t+1} \overleftarrow{A}_t^\mathsf{T} + \overleftarrow{\Sigma}_t.$$

The recursion is summarised in Algorithm 1.2. The cross moment, which is often required for learning, is easily obtained as follows:

$$\left\langle \Delta x_t \Delta x_{t+1}^\mathsf{T}|y_{1:T}\right\rangle = \overleftarrow{A}_t G_{t+1} \Rightarrow \left\langle x_t x_{t+1}^\mathsf{T}|y_{1:T}\right\rangle = \overleftarrow{A}_t G_{t+1} + g_t g_{t+1}^\mathsf{T}.$$

1.4.8 Non-linear latent Markov models

The HMM and LDS are the two main tractable workhorses of probabilistic time series modelling. However, they lie at opposite ends of the modelling spectrum: the HMM assumes fully discrete latent variables, whilst the LDS assumes fully continuous latent variables restricted to linear updates under Gaussian noise. In practice one often encounters more complex scenarios: models requiring both continuous and discrete latent variables, continuous non-linear transitions, hierarchical models with tied parameters, etc. For such cases exact inference is typically computationally intractable and approximations are required. This forms a rich area of research, and the topic of several chapters of this book. Below we give a brief overview of some classical deterministic and stochastic approximate inference techniques that have been used in the time series context.

1.5 Deterministic approximate inference

In many deterministic approximate inference methods, a computationally intractable distribution is approximated with a tractable one by optimising an objective function. For example, one may assume a family of tractable distributions $q(x|\theta_q)$, parameterised by θ_q, and

Algorithm 1.2 LDS backward pass. Compute the smoothed posteriors $p(x_t|y_{1:T})$. This requires the filtered results from Algorithm 1.1.

$G_T \leftarrow F_T, g_T \leftarrow f_T$
for $t \leftarrow T-1, 1$ **do**
  $\{g_t, G_t\} = \text{LDSBACKWARD}(g_{t+1}, G_{t+1}, f_t, F_t; \theta_t)$
end for
function LDSBACKWARD $(g, G, f, F; \theta)$
$\mu_x \leftarrow Af + \bar{x}$
$\Sigma_{x'x'} \leftarrow AFA^\mathsf{T} + Q$
$\Sigma_{x'x} \leftarrow AF$ — statistics of $p(x_t, x_{t+1}|y_{1:t})$
$\overleftarrow{\Sigma} \leftarrow F - \Sigma_{x'x}^\mathsf{T}\Sigma_{x'x'}^{-1}\Sigma_{x'x}$ — dynamics reversal $p(x_t|x_{t+1}, y_{1:t})$
$\overleftarrow{\mathrm{A}} \leftarrow \Sigma_{x'x}^\mathsf{T}\Sigma_{x'x'}^{-1}$
$\overleftarrow{m} \leftarrow f - \overleftarrow{\mathrm{A}}\mu_x$
$g' \leftarrow \overleftarrow{\mathrm{A}}g + \overleftarrow{m}$ — backward propagation
$G' \leftarrow \overleftarrow{\mathrm{A}}G\overleftarrow{\mathrm{A}}^\mathsf{T} + \overleftarrow{\Sigma}$
return g', G'

find the best approximation to an intractable distribution $p(x)$ by minimising the Kullback–Leibler (KL) divergence $\mathrm{KL}\big(q(x|\theta_q)|p(x)\big) = \big\langle \log(q(x|\theta_q)/p(x))\big\rangle_{q(x|\theta_q)}$ with respect to θ_q. The optimal $q(x|\theta_q)$ is then used to answer inference questions. In the Bayesian context, parameter learning is also a form of inference problem which is intractable for most models of interest. Below we describe a popular procedure for approximating the parameter posterior based on minimising a KL divergence.

1.5.1 Variational Bayes

In Bayesian procedures, one doesn't seek a 'single best' parameter estimate θ, but rather a posterior distribution over θ given by

$$p(\theta|y_{1:T}) = \frac{p(y_{1:T}|\theta)p(\theta)}{p(y_{1:T})}.$$

In latent variable models, the marginal likelihood $p(y_{1:T})$ is given by

$$p(y_{1:T}) = \int_\theta \int_{x_{1:T}} p(x_{1:T}, y_{1:T}|\theta)p(\theta).$$

In practice, computing the integral over both θ and $x_{1:T}$ can be difficult. The idea in variational Bayes (VB) (see, for example, [27]) is to seek an approximation

$$p(x_{1:T}, \theta|y_{1:T}) \approx q(x_{1:T}, \theta|y_{1:T}),$$

where the distribution q is restricted to the form

$$q(x_{1:T}, \theta|y_{1:T}) = q(x_{1:T}|y_{1:T})q(\theta|y_{1:T}).$$

The best distribution q in this class can be obtained by minimising the KL divergence[13]

$$\mathrm{KL}(q(x_{1:T})q(\theta)|p(x_{1:T},\theta|y_{1:T})) = \left\langle \log q(x_{1:T})\right\rangle_{q(x_{1:T})} + \left\langle \log q(\theta)\right\rangle_{q(\theta)} - \left\langle \log \frac{p(y_{1:T}|x_{1:T},\theta)p(x_{1:T}|\theta)p(\theta)}{p(y_{1:T})}\right\rangle_{q(x_{1:T})q(\theta)} .$$

The non-negativity of the divergence results in a lower bound on the marginal likelihood

$$\log p(y_{1:T}) \geq -\left\langle \log q(x_{1:T})\right\rangle_{q(x_{1:T})} - \left\langle \log q(\theta)\right\rangle_{q(\theta)} + \left\langle \log p(y_{1:T}|x_{1:T},\theta)p(x_{1:T}|\theta)p(\theta)\right\rangle_{q(x_{1:T})q(\theta)} .$$

In many cases of interest, this lower bound is computationally tractable. Minimising the KL divergence with respect to $q(x_{1:T})$ and $q(\theta)$ is equivalent to maximising the lower bound, which can be achieved by iterating the following numerical updates to convergence

1. $q(x_{1:T})^{new} \propto \exp\left\langle \log p(y_{1:T}|x_{1:T},\theta)p(x_{1:T}|\theta)\right\rangle_{q(\theta)}$
2. $q(\theta)^{new} \propto p(\theta)\exp\left\langle \log p(y_{1:T}|x_{1:T},\theta)\right\rangle_{q(x_{1:T})}$.

If we seek a point approximation $q(\theta) = \delta(\theta - \theta^*)$, the above simplifies to

1. $q(x_{1:T})^{new} \propto p(y_{1:T}|x_{1:T},\theta)p(x_{1:T}|\theta)$
2. $\theta^{new} = \underset{\theta}{\mathrm{argmax}} \left\{\left\langle \log p(y_{1:T}|x_{1:T},\theta)\right\rangle_{q(x_{1:T})} + \log p(\theta)\right\}$,

giving the penalised expectation maximisation (EM) algorithm [5]. For latent Markov models,

$$\left\langle \log p(y_{1:T}|x_{1:T},\theta)\right\rangle_{q(x_{1:T})} = \sum_t \left\langle \log p(y_t|x_t,\theta)\right\rangle_{q(x_t)} + \sum_t \left\langle \log p(x_t|x_{t-1},\theta)\right\rangle_{q(x_{t-1},x_t)}$$

so that the EM algorithm requires smoothed single and pairwise expectations [23].

1.5.2 Assumed density filtering

For more complex latent Markov models than the ones described in the previous sections, the filtering recursion (1.9) is in general numerically intractable. For continuous x_t and non-linear-Gaussian transition $p(x_t|x_{t-1})$ the integral over x_{t-1} may be difficult, or give rise to a distribution that is not in the same distributional family as $p(x_{t-1}|y_{1:t-1})$. In such cases, a useful approximation can be obtained with the assumed density filtering (ADF) method, in which the distribution obtained from the filtering recursion is projected back to a chosen family [1].

More specifically, assume that we are given an approximation $q(x_{t-1}|y_{1:t-1})$ to $p(x_{t-1}|y_{1:t-1})$, where $q(x_{t-1}|y_{1:t-1})$ is a distribution chosen for its numerical tractability (a Gaussian for example). Using the filtering recursion (1.9), we obtain an approximation for the filtered distribution at t

$$\tilde{q}(x_t|y_{1:t}) \propto \int_{x_{t-1}} p(y_t|x_t)p(x_t|x_{t-1})q(x_{t-1}|y_{1:t-1}).$$

[13] To simplify the subsequent expressions, we omit conditioning on the observations in the approximating distribution.

However, in general, $\tilde{q}(x_t|y_{1:t})$ will not be in the same family as $q(x_{t-1}|y_{1:t-1})$. To deal with this we project $\tilde{q}$ to the family q using

$$q(x_t|y_{1:t}) = \underset{q(x_t|y_{1:t})}{\text{argmin}}\, \text{KL}(\tilde{q}(x_t|y_{1:t})|q(x_t|y_{1:t}))\,.$$

For q in the exponential family, this corresponds to matching the moments of $q(x_t|y_{1:t})$ to those of $\tilde{q}(x_t|y_{1:t})$. Assumed density filtering is a widely employed approximation method and also forms part of other methods, such as approximate smoothing methods. For example, ADF is employed as part of the expectation correction method for approximate smoothing in the switching LDS (see Chapter 8 of this book). Furthermore, many approximation methods are based on ADF-style approaches. Below, we provide one example of an ADF-style approximation method for a Poisson model.

Example In this example, we discuss a model for tracking the number of objects in a given region based on noisy observations. Similar types of models appear in applications such as population dynamics (immigration) and multi-object tracking (see Chapters 3 and 11).

Suppose that, over time, objects of a specific type appear and disappear in a given region. At time step $t-1$, there are s_{t-1} objects in the region. At the next time step t, each of the s_{t-1} objects survives in the region independently of other objects with probability π_{sur}. We denote with $\bar{s}_t$ the number of surviving objects. Additionally, v_t new objects arrive with rate b, independent of existing objects, so that the number of objects present in the region at time step t becomes $s_t = \bar{s}_t + v_t$. By indicating with $\mathcal{BI}$ and $\mathcal{PO}$ the Binomial and Poisson distribution respectively, the specific survive–birth process is given by

$$\begin{aligned} &\text{Survive} && \bar{s}_t|s_{t-1} \sim \mathcal{BI}(\bar{s}_t|s_{t-1}, \pi_{sur}) = \binom{s_{t-1}}{\bar{s}_t} \pi_{sur}^{\bar{s}_t}(1-\pi_{sur})^{s_{t-1}-\bar{s}_t}, \\ &\text{Birth} && s_t = \bar{s}_t + v_t, \qquad v_t \sim \mathcal{PO}(v_t|b) = \frac{b^{v_t}}{v_t!}e^{-b}. \end{aligned}$$

Due to errors, each of the s_t objects is detected only with probability π_{det}, meaning that some objects remain possibly undetected. We denote with $\hat{s}_t$ the number of detected objects among the s_t objects. On the other hand, there is a number e_t of spurious objects that are detected (with rate c), so that we actually observe $y_t = \hat{s}_t + e_t$ objects. The specific detect–observe process is given by

$$\begin{aligned} &\text{Detect} && \hat{s}_t|s_t \sim \mathcal{BI}(\hat{s}_t|s_t, \pi_{det}), \\ &\text{Observe in clutter} && y_t = \hat{s}_t + e_t, \qquad e_t \sim \mathcal{PO}(e_t|c). \end{aligned}$$

The belief network representation of this model is given in Fig. 1.8.

The inferential goal is to estimate the true number of objects s_t from the filtered posterior $p(s_t|y_{1:t})$. Unfortunately, the filtered posterior is not a distribution in any standard form and becomes increasingly difficult to represent as we proceed in time. To deal with this, we make use of an ADF-style approach to obtain a Poisson approximation to the filtered posterior at each time step.

If we assume that $p(s_{t-1}|y_{1:t-1})$ is Poisson, then a natural way to compute $p(s_t|y_{1:t})$ would be to use

$$p(s_t|y_{1:t}) \propto p(y_t|s_t)p(s_t|y_{1:t-1}).$$

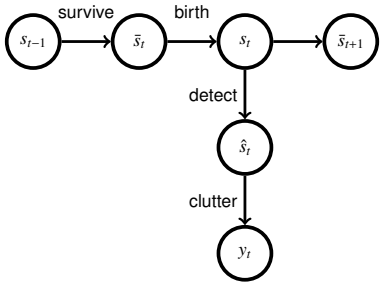


Figure 1.8 Belief network representation of the counting model. The goal is to compute the filtering density $p(s_t|y_{1:t})$. Nodes $\hat{s}_{t-1}$, y_{t-1} are omitted for clarity.

The first term $p(y_t|s_t)$ is obtained as the sum of a Binomial and a Poisson random variable. The second term $p(s_t|y_{1:t-1})$ may be computed recursively from $p(s_{t-1}|y_{1:t-1})$ and is Poisson distributed (see below). Performing ADF moment matching of the non-standard $p(s_t|y_{1:t})$ to a Poisson distribution is however not straightforward.

A simpler alternative approach (and not generally equivalent to fitting the best Poisson distribution to $p(s_t|y_{1:t})$ in the minimal KL divergence sense) is to project $p(\hat{s}_t|y_{1:t})$ to a Poisson distribution using moment matching and then form

$$p(s_t|y_{1:t}) = \sum_{\hat{s}_t} p(s_t|\hat{s}_t, y_{1:t-1}) p(\hat{s}_t|y_{1:t}) = \sum_{\hat{s}_t} \frac{p(s_t, \hat{s}_t, y_{1:t-1})}{p(\hat{s}_t, y_{1:t-1})} p(\hat{s}_t|y_{1:t})$$
$$= p(s_t|y_{1:t-1}) \sum_{\hat{s}_t} \frac{p(\hat{s}_t|s_t)}{p(\hat{s}_t|y_{1:t-1})} p(\hat{s}_t|y_{1:t}), \tag{1.12}$$

which, as we will see, is also Poisson distributed. Before proceeding with explaining the recursion, we state two useful results for Poisson random variables. Let s and e be Poisson random variables with respective intensities λ and ν. Then

superposition The sum $y = s + e$ is Poisson distributed with intensity $\lambda + \nu$.

conditioning The distribution of s conditioned on e is given by $p(s|e) = \mathcal{BI}(s|e, \lambda/\nu)$.

Using these results we can derive a recursion as follows. At time $t-1$ we assume $p(s_{t-1}|y_{1:t-1}) = \mathcal{PO}(s_{t-1}|\lambda_{t-1|t-1})$. This gives

$$p(\bar{s}_t|y_{1:t-1}) = \sum_{s_{t-1}} p(\bar{s}_t|s_{t-1}) p(s_{t-1}|y_{1:t-1}) = \mathcal{PO}(\bar{s}_t|\pi_{sur}\lambda_{t-1|t-1}),$$

where we have used the following general result derived from the conditioning property:

$$\sum_n \mathcal{BI}(m|n, \pi)\mathcal{PO}(n|\lambda) = \mathcal{PO}(m|\pi\lambda)\,.$$

From the birth process and using the superposition property we obtain

$$p(s_t|y_{1:t-1}) = \mathcal{PO}(s_t|\lambda_{t|t-1}), \qquad \lambda_{t|t-1} = b + \pi_{sur}\lambda_{t-1|t-1}\,.$$

This gives

$$p(\hat{s}_t|y_{1:t-1}) = \sum_{s_t} p(\hat{s}_t|s_t) p(s_t|y_{1:t-1}) = \mathcal{PO}(\hat{s}_t|\pi_{det}\lambda_{t|t-1})\,.$$

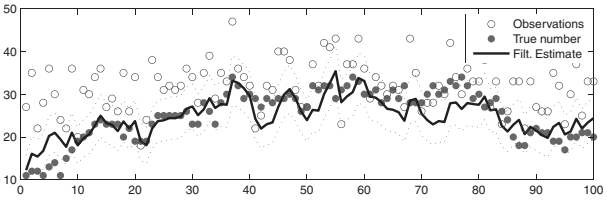


Figure 1.9 Assumed density filtering for object tracking. The horizontal axis denotes the time index and the vertical axis the number of objects. The dotted lines represent one standard deviation of the filtered posterior.

From the observe in clutter process and using the superposition property we obtain the predictive distribution

$$p(y_t|y_{1:t-1}) = \mathcal{PO}(y_t|\lambda_{t|t-1}\pi_{det} + c)\,.$$

The posterior distribution of the number of detected objects is therefore

$$p(\hat{s}_t|y_{1:t}) = \mathcal{BI}(\hat{s}_t|y_t, \lambda_{t|t-1}\pi_{det}/(\lambda_{t|t-1}\pi_{det} + c))\,.$$

A well-known Poisson approximation to the Binomial distribution based on moment matching has intensity

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}}\, \mathrm{KL}(\mathcal{BI}(s|y,\pi)|\mathcal{PO}(s|\lambda)) = y\pi,$$

so that

$$p(\hat{s}_t|y_{1:t}) \approx \mathcal{PO}(\hat{s}_t|\gamma), \qquad \gamma \equiv y_t\lambda_{t|t-1}\pi_{det}/(\lambda_{t|t-1}\pi_{det} + c)\,.$$

Using Eq. (1.12) with the Poisson approximation to $p(\hat{s}_t|y_{1:t})$, we obtain

$$\begin{aligned} p(s_t|y_{1:t}) &\approx \mathcal{PO}(s_t|\lambda_{t|t-1}) \sum_{\hat{s}_t} \frac{\mathcal{BI}(\hat{s}_t|s_t,\pi_{det})}{\mathcal{PO}(\hat{s}_t|\pi_{\mathrm{det}}\lambda_{t|t-1})}\mathcal{PO}(\hat{s}_t|\gamma) \\ &\propto \frac{(\lambda_{t|t-1}(1-\pi_{det}))^{s_t}}{s_t!} \underbrace{\sum_{\hat{s}_t} \binom{s_t}{\hat{s}_t} \left(\frac{\gamma}{\lambda_{t|t-1}(1-\pi_{det})}\right)^{\hat{s}_t}}_{\left(1+\frac{\gamma}{\lambda_{t|t-1}(1-\pi_{det})}\right)^{s_t}}, \end{aligned}$$

so that

$$p(s_t|y_{1:t}) \approx \mathcal{PO}(s_t|\lambda_{t|t}), \qquad \lambda_{t|t} = (1-\pi_{det})\lambda_{t|t-1} + y_t\frac{\pi_{det}\lambda_{t|t-1}}{c + \pi_{det}\lambda_{t|t-1}}\,.$$

Intuitively, the first term in $\lambda_{t|t}$ corresponds to the undetected objects, whilst the second term is the Poisson approximation to the Binomial posterior that results from observing the sum of two Poisson random variables with intensities c and $\pi_{det}\lambda_{t|t-1}$. At time $t = 1$, we initialise the intensity $\lambda_{1|0}$ to the birth intensity.

In Fig. 1.9, we show the results of the filtering recursion on data generated from the model. As we can see, the tracking performance is good even though the filter involves an approximation.

This technique is closely related to the *Poissonisation* method used heavily in probabilistic analysis of algorithms [20]. In Chapters 3 and 11, an extension called the probability hypothesis density (PHD) filter to multi-object tracking is considered. Instead of tracking a scalar intensity, an intensity function over the whole space is approximated. The PHD filter combines ADF with approximate inference methods such as sequential Monte Carlo (see Section 1.6).

1.5.3 Expectation propagation

In this section, we review another powerful deterministic approximation technique called expectation propagation (EP) [19]. We present here EP in the context of approximating the posterior $p(x_t|y_{1:T})$ of a continuous latent Markov model $p(x_{1:T}, y_{1:T})$, see also Chapter 7. According to Eqs. (1.4) and (1.7), the exact single and pairwise marginals have the form

$$p(x_t|y_{1:T}) \propto \alpha(x_t)\beta(x_t),$$
$$p(x_t, x_{t+1}|y_{1:T}) \propto \alpha(x_t)p(y_{t+1}|x_{t+1})p(x_{t+1}|, x_t)\beta(x_{t+1}).$$

Starting from these equations, we can retrieve the recursions (1.3)–(1.5) for α and β by requiring that the single marginal is consistent with the pairwise marginal, that is

$$p(x_{t+1}|y_{1:T}) = \int_{x_t} p(x_t, x_{t+1}|y_{1:T}),$$
$$\alpha(x_{t+1})\beta(x_{t+1}) \propto \int_{x_t} \alpha(x_t)p(y_{t+1}|x_{t+1})p(x_{t+1}|, x_t)\beta(x_{t+1}).$$

Cancelling $\beta(x_{t+1})$ from both sides we immediately retrieve the standard α recursion. One may derive the β recursion similarly by integrating the pairwise marginal over x_{t+1}. For complex situations, the resulting $\alpha(x_{t+1})$ is not in the same family as $\alpha(x_t)$, giving rise to representational difficulties. As in ADF, we therefore project $\alpha(x_{t+1})$ back to a chosen family. Whilst this is reasonably well defined, since $\alpha(x_{t+1})$ represents a filtered distribution, it is unclear how to project $\beta(x_t)$ to a chosen family since $\beta(x_t)$ is not a distribution in x_t. In EP, this problem is resolved by first defining

$$\tilde{q}(x_{t+1}) \propto \int_{x_t} \alpha(x_t)p(y_{t+1}|x_{t+1})p(x_{t+1}|, x_t)\beta(x_{t+1}),$$

and then iterating the following updates to convergence

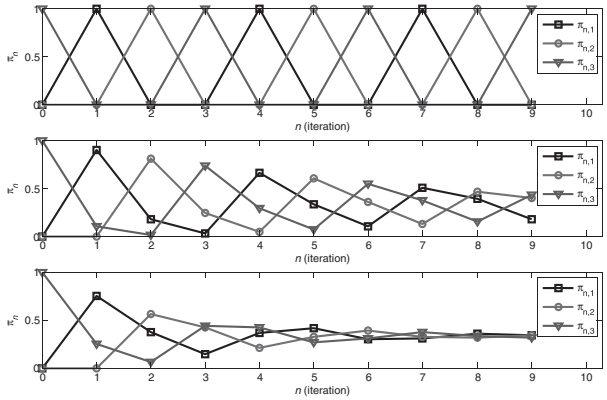
$$\alpha(x_{t+1}) = \underset{\alpha(x_{t+1})}{\operatorname{argmin}} \mathrm{KL}\left(\tilde{q}(x_{t+1})|\frac{1}{Z_{t+1}}\alpha(x_{t+1})\beta(x_{t+1})\right),$$
$$\beta(x_t) = \underset{\beta(x_t)}{\operatorname{argmin}} \mathrm{KL}\left(\tilde{q}(x_t)|\frac{1}{Z_t}\alpha(x_t)\beta(x_t)\right),$$

where Z_t and Z_{t+1} are normalisation constants. In exponential family approximations, these updates correspond to matching the moments of $\tilde{q}$ to the moments of $\alpha(x)\beta(x)$.

1.6 Monte Carlo inference

Many inference problems such as filtering and smoothing can be considered as computing expectations with respect to a (posterior) distribution. A general numerical method for approximating the expectation $\mathsf{E}_\pi[\varphi(x)] = \int_x \varphi(x)\pi(x)$ of a function φ of a random variable x is given by sampling. Consider a procedure that draws samples from a multivariate distribution $\hat{\pi}(x^1, \ldots, x^N)$. For $\mathcal{X} = \left\{x^1, \ldots, x^N\right\}$, the random variable

$$\bar{E}_{\mathcal{X},N} \equiv \frac{\varphi(x^1) + \cdots + \varphi(x^N)}{N}$$



$\epsilon = 0$: Periodic chain that fails to converge to a stationary distribution.

$\epsilon = 0.1$: Chain that converges to the stationary (uniform) distribution.

$\epsilon = 0.25$: Chain that converges to the stationary (uniform) distribution more quickly than for $\epsilon = 0.1$.

Figure 1.10 Convergence to the stationary distribution. For $\epsilon = 0$, the state transition diagram would be disconnected, hence the chain fails to be irreducible and therefore to converge.

has expectation

$$\mathsf{E}_{\hat{\pi}}\left[\bar{E}_{\mathcal{X},N}\right] = \frac{1}{N}\sum_n \mathsf{E}_{\hat{\pi}}\left[\phi(x^n)\right].$$

If the marginal distribution of each x^n is equal to the target distribution

$$\hat{\pi}(x^n) = \pi(x^n), \qquad n = 1, \ldots, N$$

then

$$\mathsf{E}_{\hat{\pi}}\left[\bar{E}_{\mathcal{X},N}\right] = \mathsf{E}_{\pi}\left[\varphi(x)\right],$$

that is $\bar{E}_{\mathcal{X},N}$ is an unbiased estimator of $\mathsf{E}_{\pi}\left[\varphi(x)\right]$. If, in addition, the samples are independent

$$\hat{\pi}(x^1, \ldots, x^N) = \prod_{n=1}^{N} \hat{\pi}(x^n),$$

and $\mathsf{E}_{\pi}\left[\varphi(x)\right]$ and $\mathsf{V}_{\pi}[\varphi(x)]$ are finite, the central limit theorem guarantees that, for sufficiently large N, $\bar{E}_{\mathcal{X},N}$ is Gaussian distributed with mean and covariance

$$\mathsf{E}_{\pi}\left[\varphi(x)\right], \qquad \frac{\mathsf{V}_{\pi}[\varphi(x)]}{N}.$$

That is the variance of the estimator drops with increasing N. These results have important practical consequences: If we have a procedure that draws *iid* samples $x^1, \ldots, x^N$ from π, then the sample average $\bar{E}_{\mathcal{X},N}$ converges rapidly to the exact expectation $\mathsf{E}_{\pi}\left[\varphi(x)\right]$ as N increases and provides a 'noisy' but unbiased estimator for any finite N. For large N the error behaves as $N^{-1/2}$ and is independent of the dimensionality of x. The key difficulty, however, is in generating independent samples from the target distribution π. Below we discuss various Monte Carlo methods that strive to provide unbiased samples and vary in the degree to which they generate independent samples.

1.6.1 Markov chain Monte Carlo

In Markov chain Monte Carlo (MCMC) methods, samples from a desired complex distribution $\pi(x)$ are approximated with samples from a simpler distribution defined by a specially constructed time-homogeneous Markov chain. Given an initial state x^1, a set of samples $x^2, \ldots, x^N$ from the chain are obtained by iteratively drawing from the transition distribution ('kernel') $K(x^n|x^{n-1})$.[14] The distribution $\pi_n(x^n)$ satisfies

$$\pi_n(x^n) = \int_{x^{n-1}} K(x^n|x^{n-1})\pi_{n-1}(x^{n-1}),$$

which we compactly write as $\pi_n = K\pi_{n-1}$. The theory of Markov chains characterises the convergence of the sequence $\pi_1, \pi_2, \ldots$ If the sequence converges to a distribution π, then π (called the stationary distribution) satisfies $\pi = K\pi$. For an ergodic chain, namely irreducible and aperiodic,[15] there exists a unique stationary distribution to which the sequence converges, irrespective of the initial state x^1. To illustrate the idea, consider the Markov chain of Section 1.3.1 in which the robot moves freely under the transition model defined in Eq. (1.1), repeated here for convenience

$$K = \epsilon \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + (1-\epsilon) \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

The robot starts at cell 3, i.e., $\pi_1 = (\pi_{11}, \pi_{12}, \pi_{13}) = (0, 0, 1)^\top$. In Fig. 1.10, we plot the cell probabilities of $\pi_n = K^{n-1}\pi_1$ as n increases for various choices of ϵ. Provided $0 < \epsilon \leq 1$, all chains converge to the uniform distribution $\pi = (1/3, 1/3, 1/3)$, however, with differing convergence rates.

This discussion suggests that, if we can design a transition kernel K such that the associated Markov chain is ergodic and has the target distribution π as its stationary distribution, at least in principle we can generate samples from the Markov chain that eventually will tend to be from π. After ignoring the initial 'burn in' part of the generated path as the sequence moves to the stationary distribution, the subsequent part can be used to estimate expectations under π. Notice, however, that the samples generated will typically be dependent and therefore the variance of the estimate may not scale inversely with the number of samples from the chain.

Metropolis–Hastings

Designing a transition kernel K for a given target π is straightforward via the approach proposed by Metropolis [18] and later generalised by Hastings [12]. Suppose that we are given a target density $\pi = \phi/Z$, where Z is a (possibly unknown) normalisation constant. The Metropolis–Hastings (MH) algorithm uses a proposal density $q(x|x')$ for generating a candidate sample x, which is accepted with probability $0 \leq \alpha(x|x') \leq 1$ defined as

$$\alpha(x|x') = \min\left\{1, \frac{q(x'|x)\pi(x)}{q(x|x')\pi(x')}\right\},$$

[14] Note that the sample index is conceptually different from the time index in a time series model; here n is the iteration number of the sampling algorithm.

[15] For finite state Markov chains, irreducibility means that each state can be visited starting from any other, while aperiodicity means that each state can be visited at any iteration n larger than some fixed number.

Algorithm 1.3 Metropolis–Hastings

Initialise x^1 arbitrarily.
for $n = 2, 3 \ldots$ **do**
  Propose a candidate: $x^{\text{cand}} \sim q(x^{\text{cand}}|x^{n-1})$.
  Compute acceptance probability:

$$\alpha(x^{\text{cand}}|x^{n-1}) = \min\left\{1, \frac{q(x^{n-1}|x^{\text{cand}})\pi(x^{\text{cand}})}{q(x^{\text{cand}}|x^{n-1})\pi(x^{n-1})}\right\}.$$

  Sample from uniform distribution: $u \sim \mathcal{U}(u|[0, 1])$.
  if $u < \alpha$ **then**
    Accept candidate: $x^n \leftarrow x^{\text{cand}}$.
  else
    Reject candidate: $x^n \leftarrow x^{n-1}$.
  end if
end for

and rejected with probability $0 \leq \rho(x') \leq 1$ defined as

$$\rho(x') = \int (1 - \alpha(x|x'))q(x|x')\mathrm{d}x.$$

The MH transition kernel K has the following form

$$K(x|x') = q(x|x')\alpha(x|x') + \delta(x - x')\rho(x').$$

This kernel satisfies the *detailed balance* property

$$\begin{aligned}
K(x|x')\pi(x') &= (q(x|x')\alpha(x|x') + \delta(x - x')\rho(x'))\pi(x') \\
&= q(x|x')\min\{1, \frac{q(x'|x)\pi(x)}{q(x|x')\pi(x')}\}\pi(x') + \delta(x - x')\rho(x')\pi(x') \\
&= \min\{q(x|x')\pi(x'), q(x'|x)\pi(x)\} + \delta(x - x')\rho(x')\pi(x') \\
&= q(x'|x)\min\{\frac{q(x|x')\pi(x')}{q(x'|x)\pi(x)}, 1\}\pi(x) + \delta(x' - x)\rho(x)\pi(x) \\
&= K(x'|x)\pi(x).
\end{aligned}$$

By integrating both sides over x', we obtain

$$\pi(x) = \int K(x|x')\pi(x')\mathrm{d}x',$$

and therefore π is a stationary distribution of K. Note that to compute the acceptance probability α we only need to evaluate ϕ, since the normalisation constant Z cancels out. For a given target π and proposal $q(x'|x)$ we now have a procedure for sampling from a Markov chain with stationary distribution π. The procedure is detailed in Algorithm 1.3.

Gibbs sampling

The Gibbs sampler [10, 17] is a MCMC method which is suitable for sampling a multivariate random variable $x = (x_1, \ldots, x_D)$ with joint distribution $p(x)$. Gibbs sampling proceeds by partitioning the set of variables x into a chosen variable x_i and the rest $x = (x_i, x_{-i})$. The

Algorithm 1.4 Gibbs sampler

1: Initialise $x^1 = (x_1^1, \ldots, x_D^1)$ arbitrarily.
2: **for** $n = 2, 3 \ldots$ **do**
3: $x_1^n \sim p(x_1^n | x_2^{n-1}, x_3^{n-1}, \ldots, x_D^{n-1})$.
4: $x_2^n \sim p(x_2^n | x_1^n, x_3^{n-1}, \ldots, x_D^{n-1})$.
$\vdots$
5: $x_D^n \sim p(x_D^n | x_1^n, x_2^n, \ldots, x_{D-1}^n)$.
6: **end for**

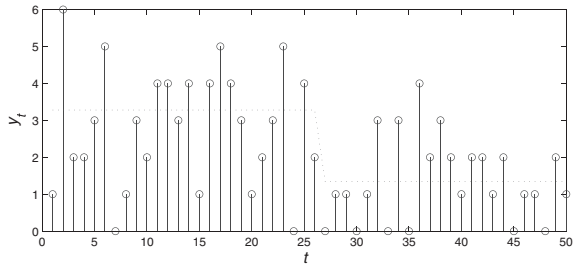


Figure 1.11 A typical realisation from the changepoint model. The time index is indicated by t and the number of counts by y_t. The true intensities are shown with a dotted line: at time step $\tau = 26$, the intensity drops from $\lambda_1 = 3.2$ to $\lambda_2 = 1.2$.

assumption is that the conditional distributions $p(x_i|x_{-i})$ are tractable. One then proceeds coordinate-wise by sampling from the conditionals as in Algorithm 1.4.

Gibbs sampling can be viewed as MH sampling with the proposal

$$q(x|x') = p(x_i|x'_{-i})\delta\left(x_{-i} - x'_{-i}\right).$$

Using this proposal results in a MH acceptance probability of 1, so that every candidate sample is accepted. Dealing with evidence (variables in known states) is straightforward – one sets the evidential variables into their states and samples from the remaining variables.

Example: Gibbs sampling for a changepoint model

We illustrate the Gibbs sampler on a changepoint model for count data [13]. In this model, at each time t we observe the count of an event y_t. All the counts up to an unknown time τ are *iid* realisations from a Poisson distribution with intensity λ_1. From time $\tau + 1$ to T, the counts come from a Poisson distribution with intensity λ_2. We assume that the changepoint τ is uniformly distributed over $1, \ldots, T$ and that the intensities λ_1, λ_2 are Gamma distributed

$$\mathcal{G}(\lambda_i|a, b) = \frac{1}{\Gamma(a)} b^a \lambda_i^{a-1} e^{-b\lambda_i}, \qquad i = 1, 2\,.$$

This leads to the following generative model

$$\begin{aligned} \tau &\sim \mathcal{U}(\tau|1, \ldots, T), \quad \lambda_i \sim \mathcal{G}(\lambda_i|a, b), \qquad i = 1, 2, \\ y_t &\sim \begin{cases} \mathcal{PO}(y_t|\lambda_1) & 1 \le t \le \tau, \\ \mathcal{PO}(y_t|\lambda_2) & \tau < t \le T. \end{cases} \end{aligned}$$

A typical draw from this model is shown in Fig. 1.11. The inferential goal is to compute the posterior distribution $p(\lambda_1, \lambda_2, \tau|y_{1:T})$ of the intensities and changepoint given the count

Algorithm 1.5 A Gibbs sampler for the changepoint model

1: Initialise λ_2^1, τ^1.
2: **for** $n = 2, 3 \ldots$ **do**
3: $\quad \lambda_1^n \sim p(\lambda_1^n | \tau^{n-1}, y_{1:T}) = \mathcal{G}(a + \sum_{t=1}^{\tau^{n-1}} y_t, \tau^{n-1} + b)$.
4: $\quad \lambda_2^n \sim p(\lambda_2^n | \tau^{n-1}, y_{1:T}) = \mathcal{G}(a + \sum_{t=\tau^{n-1}+1}^{T} y_t, T - \tau^{n-1} + b)$.
5: $\quad \tau^n \sim p(\tau^n | \lambda_1^n, \lambda_2^n, y_{1:T})$.
6: **end for**

data. In this problem this posterior is actually tractable and serves to assess the quality of the Gibbs sampling approximation.

To implement Gibbs sampling we need to compute the distribution of each variable, conditioned on the rest. These conditionals can be conveniently derived by writing the log of the joint distribution of all variables and collecting terms that depend only on the free variable. The log of the joint distribution is given by

$$\log p(y_{1:T}, \lambda_1, \lambda_2, \tau) = \log \left(p(\lambda_1) p(\lambda_2) p(\tau) \prod_{t=1}^{\tau} p(y_t|\lambda_1) \prod_{t=\tau+1}^{T} p(y_t|\lambda_2) \right).$$

This gives

$$\log p(\lambda_1 | \tau, \cancel{\lambda_2}, y_{1:T}) = \left(a + \sum_{t=1}^{\tau} y_t - 1 \right) \log \lambda_1 - (\tau + b)\lambda_1 + \text{const.},$$

$$\log p(\tau | \lambda_1, \lambda_2, y_{1:T}) = \sum_{t=1}^{\tau} y_t \log \lambda_1 + \sum_{t=\tau+1}^{T} y_t \log \lambda_2 + \tau (\lambda_2 - \lambda_1) + \text{const.},$$

and similarly for $\log p(\lambda_1 | \tau, y_{1:T})$, so that both $p(\lambda_1 | \tau, y_{1:T})$ and $p(\lambda_2 | \tau, y_{1:T})$ are Gamma distributions. Sampling from these conditional distributions is straightforward, see Algorithm 1.5. Samples from the obtained posterior distribution are plotted in Fig. 1.12. For this particularly simple problem, Gibbs sampling works well, with the estimated sample marginal estimates of λ and τ close to the values we expect based on the known parameters used to generate the data.

1.6.2 Sequential Monte Carlo

The MCMC techniques described above are batch algorithms that require the availability of all data records. These techniques are therefore unsuitable when the data needs to be processed sequentially and can be prohibitive for long time series. In such cases, it is desirable to use alternative methods which process the data sequentially and take a constant time per observation. In this context, sequential Monte Carlo (SMC) techniques [6, 8] have proved useful in many applications. These methods are based on importance sampling/resampling which we review below.

Importance sampling

Suppose that we are interested in computing the expectation $\mathsf{E}_p[\varphi(x)]$ with respect to a distribution $p(x) = \phi(x)/Z$, where the non-negative function $\phi(x)$ is known but the overall normalisation constant Z is assumed to be computationally intractable. In importance sampling (IS), instead of sampling from the target distribution $p(x)$, we sample from a

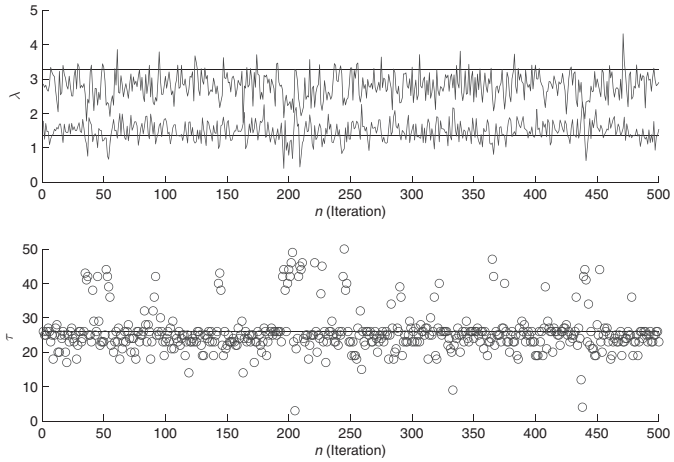


Figure 1.12 Gibbs samples from the posterior of the changepoint model vs. sample iteration. True values are shown with a horizontal line. (top) Intensities λ_1 and λ_2. (bottom) Changepoint index τ.

tractable distribution $q(x)$ and reweight the obtained samples to form an unbiased estimator of $\mathsf{E}_p[\varphi(x)]$. IS is based on the realisation that we can write the expectation with respect to p as a ratio of expectations with respect to q, that is

$$\mathsf{E}_p[\varphi(x)] = \frac{1}{Z}\int \varphi(x)\frac{\phi(x)}{q(x)}q(x) = \frac{\mathsf{E}_q[\varphi(x)W(x)]}{Z} = \frac{\mathsf{E}_q[\varphi(x)W(x)]}{\mathsf{E}_q[W(x)]},$$

where $W(x) \equiv \phi(x)/q(x)$ is called the *weight function*. Thus $\mathsf{E}_p[\varphi(x)]$ can be approximated using samples $x^1, \ldots, x^N$ from q as

$$\frac{\sum_{i=1}^N W^i\varphi(x^i)/N}{\sum_{i=1}^N W^i/N},$$

where $W^i \equiv W(x^i)$. The samples $x^1, \ldots, x^N$ are also known as 'particles'. Using normalised weights $w^i \equiv W^i/\sum_{i'=1}^N W^{i'}$, we can write the approximation as

$$\sum_{i=1}^N w^i\varphi(x^i).$$

An example for a bimodal distribution $p(x)$ and unimodal distribution $q(x)$ is given in Fig. 1.13, showing how the weights compensate for the mismatch between q and p.

1.6.3 Resampling

Unless the IS distribution $q(x)$ is close to the target distribution $p(x)$, the normalised weights will typically have significant mass in only a single component. This issue can be partially addressed using resampling. Given a weighted particle system $\sum_{i=1}^N w^i\delta(x-x^i)$, resampling is the term for a set of methods for generating randomly a reweighted particle system of the form $\frac{1}{M}\sum_{i=1}^N n_i\delta(x-x^i)$. Specifically, a resampling algorithm returns an occupancy vector $n_1, \ldots, n_N$ which satisfies $n_i \in \{0, 1, 2, \ldots, M\}$, $\sum_i n_i = M$. For the resampling algorithm to produce an unbiased estimator of the original system $\sum_{i=1}^N w^i\delta(x-x^i)$ we require

$$\mathsf{E}\Big[\frac{1}{M}\sum_{i=1}^N n_i\delta(x-x^i)\Big] = \sum_{i=1}^N \frac{1}{M}\mathsf{E}[n_i]\delta(x-x^i)\,.$$

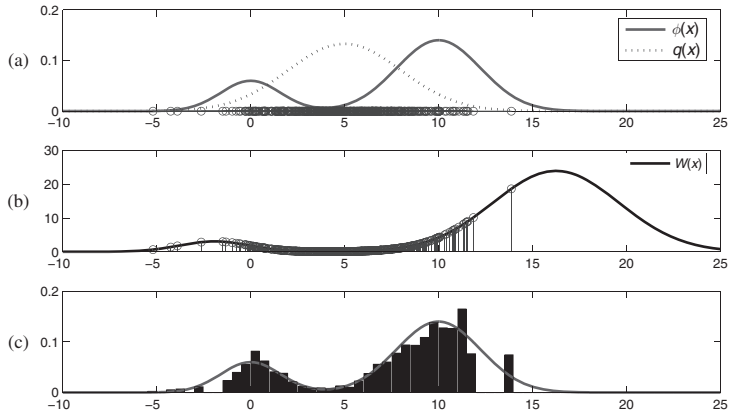


Figure 1.13 Importance sampling. (a) The solid curve denotes the unnormalised target distribution $\phi(x)$ and the dashed curve the tractable IS distribution $q(x)$. Samples from $q(x)$ are assumed straightforward to generate and are plotted on the axis. (b) To account for the fact that the samples are from q and not from the target p, we need to reweight the samples. The IS distribution q generates too many samples where p has low mass, and too few where p has high mass. The samples in these regions are reweighted accordingly. (c) Binning the weighted samples from q, we obtain an approximation to p such that averages with respect to this approximation will be close to averages with respect to p.

Hence, provided $\mathsf{E}[n_i] = Mw^i$, expectations carried out using the resampled particles will be unbiased. It is typical (though not necessary) to set $M = N$. Intuitively, resampling is a randomised pruning algorithm in which we discard particles with low weight. Unlike a deterministic pruning algorithm, the random but unbiased nature of resampling ensures an asymptotically consistent algorithm. For a discussion and comparison of resampling schemes in the context of SMC see [3, 8].

1.6.4 Sequential importance sampling

We now apply IS to the latent Markov models of Section 1.3. The resulting sequential IS methods are also known as particle filters. The goal is to estimate the posterior

$$p(x_{1:t}|y_{1:t}) = \underbrace{p(y_{1:t}|x_{1:t})p(x_{1:t})}_{\phi(x_{1:t})} / \underbrace{p(y_{1:t})}_{Z_t},$$

where we assume that the normalisation term Z_t is intractable. At each time t, we have an importance distribution $q_t(x_{1:t})$, from which we draw samples $x^i_{1:t}$ with corresponding importance weights

$$W^i_t = \phi(x^i_{1:t})/q_t(x^i_{1:t}).$$

Without loss of generality, we can construct q sequentially

$$q_t(x_{1:t}) = q_t(x_t|x_{1:t-1})q_t(x_{1:t-1}).$$

In particle filtering, one chooses a distribution q that only updates the current x_t and leaves previous samples unaffected. This is achieved using

$$q_t(x_{1:t}) = q_t(x_t|x_{1:t-1})q_{t-1}(x_{1:t-1})\,.$$

The weight function $W_t(x_{1:t})$ then admits a recursive formulation

$$W_t(x_{1:t}) = \frac{\phi(x_{1:t})}{q_t(x_{1:t})} = \frac{p(y_t|x_t)p(x_t|x_{t-1})\prod_{\tau=1}^{t-1} p(y_\tau|x_\tau)p(x_\tau|x_{\tau-1})}{q_t(x_t|x_{1:t-1})\prod_{\tau=1}^{t-1} q_\tau(x_\tau|x_{1:\tau-1})}$$
$$= \underbrace{\frac{p(y_t|x_t)p(x_t|x_{t-1})}{q_t(x_t|x_{1:t-1})}}_{v_t} W_{t-1}(x_{1:t-1}),$$

where v_t is called the incremental weight. Particle filtering algorithms differ in their choices for $q_t(x_t|x_{1:t-1})$. The optimal choice (in terms of reducing the variance of the weights) is the one step filtering distribution [7]

$$q_t(x_t|x_{1:t-1}) = p(x_t|x_{t-1}, y_t).$$

However, sampling from this distribution is difficult in practice, and simpler distributions are therefore employed. The *bootstrap* filter uses the transition

$$q_t(x_t|x_{1:t-1}) = p(x_t|x_{t-1}),$$

for which the incremental weight becomes $v_t = p(y_t|x_t)$. In this case, the IS distribution does not make any use of the recent observation and therefore has the tendency to lose track of the high probability regions of the posterior. Indeed, it can be shown that the variance of the importance weights for the bootstrap filter increases in an unbounded fashion [7, 17] so that, after a few time steps, the particle set typically loses track of the exact posterior mode. A crucial extra step to make the algorithm work is resampling, which prunes branches with low weights and keeps the particle set located in high probability regions. It can be shown that, although the particles become dependent due to resampling, the estimations are still consistent and converge to the true values as the number of particles increases to infinity.

A generic particle filter is given in Algorithm 1.6 and in Fig. 1.14 we illustrate the dynamics of the algorithm in a tracking scenario. At time step $t-1$ each 'parent' particle generates offspring candidates x_t from the IS distribution. The complete set of offspring is then weighted and resampled to generate a set of particles at time t. In the figure parent particles are linked to their surviving offspring.

1.7 Discussion and summary

Probabilistic time series models enable us to reason in a consistent way about temporal events under uncertainty. The probabilistic framework is particularly appealing for its conceptual clarity, and the use of a graphical model representation simplifies the development of the models and associated inference algorithms. The Markov independence assumption, which states that only a limited memory of the past is needed for understanding the present, plays an important role in time series models. This assumption reduces the burden in model specification and simplifies the computation of quantities of interest.

We reviewed several classical probabilistic Markovian models such AR models, hidden Markov models and linear dynamical systems, for which inference is tractable. We then discussed some of the main approximate approaches for the case of intractable inference, namely deterministic methods such as variational techniques and assumed density filtering and stochastic methods such as Monte Carlo sampling.

Many real-world time series problems are highly specialised and require novel models. The probabilistic approach, coupled with a graphical representation, facilitates the

Algorithm 1.6 Particle filter

for $i = 1, \ldots, N$ **do**
Compute the IS distribution: $q_t(x_t|x^i_{1:t-1})$.
Generate offsprings: $\hat{x}^i_t \sim q_t(x_t|x^i_{1:t-1})$.
Evaluate importance weights

$$v^i_t = \frac{p(y_t|\hat{x}^i_t)p(\hat{x}^i_t|x^i_{t-1})}{q_t(\hat{x}^i_t|x^i_{1:t-1})}, \qquad W^i_t = v^i_t W^i_{t-1}.$$

end for
if Not Resample **then**
Extend particles: $x^i_{1:t} = (x^i_{1:t-1}, \hat{x}^i_t)$, $i = 1, \ldots, N$.
else
Normalise importance weights: $\tilde{Z}_t \leftarrow \sum_j W^j_t, \qquad \tilde{\mathbf{w}}_t \leftarrow (W^1_t, \ldots, W^N_t)/\tilde{Z}_t$.
Generate associations: $(a(1), \ldots, a(N)) \leftarrow \text{Resample}(\tilde{\mathbf{w}}_t)$.
Discard or keep particles and reset weights

$$x^i_{0:t} \leftarrow (x^{a(i)}_{0:t-1}, \hat{x}^{a(i)}_t), \qquad W^i_t \leftarrow \tilde{Z}_t/N, \qquad i = 1, \ldots, N.$$

end if

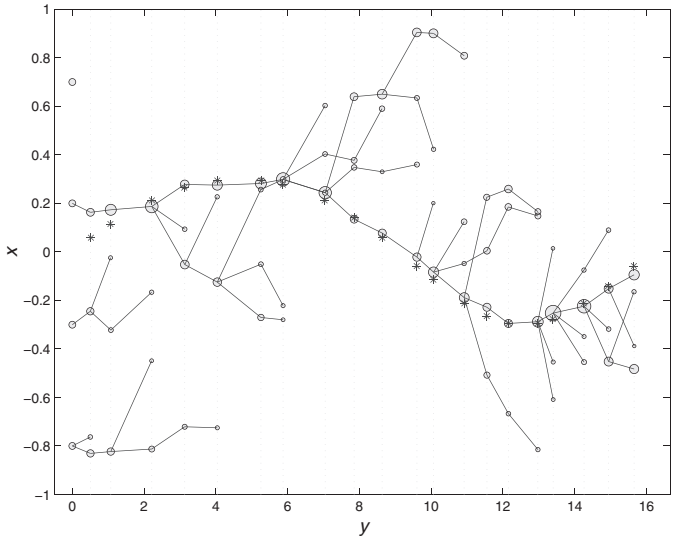


Figure 1.14 Illustration of the dynamics of a particle filter with $N = 4$ particles. The underlying latent Markov model corresponds to an object moving with positive velocity on the real line. The vertical axis corresponds to the latent log-velocities x and the horizontal axis to the observed noisy positions y: the underlying velocities of the process are shown as '*', while the observed positions are shown by dotted vertical lines. The nodes of the tree correspond to the particle positions and the sizes are proportional to normalised weights $\tilde{w}^{(i)}$.

development of tailored models and helps to reason about the computational complexity of their implementation. The field is currently very active, with many novel developments in modelling and inference, several of which are discussed in the remainder of this book.

Acknowledgments Silvia Chiappa would like to thank the European Commission for supporting her research through a Marie Curie Intra European Fellowship.

Bibliography

[1] B. D. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[2] G. Box, G. M. Jenkins and G. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, 1994.

[3] O. Cappé, R. Douc and E. Moulines. Comparison of resampling schemes for particle filtering. In *4th International Symposium on Image and Signal Processing and Analysis*, pages 64–69, 2005.

[4] O. Cappé, E. Moulines and T. Rydén. *Inference in Hidden Markov Models*. Springer-Verlag, 2005.

[5] A. Dempster, N. Laird and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, **39**(1):1–38, 1977.

[6] A. Doucet, N. de Freitas and N. J. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[7] A. Doucet, S. Godsill and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, **10**(3):197–208, 2000.

[8] A. Doucet and A. M. Johansen. A Tutorial on particle filtering and smoothing: fifteen years later. *Handbook of Nonlinear Filtering.* Oxford University Press, 2010.

[9] J. Durbin. The fitting of time series models. *Rev. Inst. Int. Stat.*, 28, pages 233–243, 1960.

[10] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. In M. A. Fischler and O. Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 564–584. Kaufmann, 1987.

[11] F. Gustafsson. *Adaptive filtering and change detection*. John Wiley & Sons, 2000.

[12] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**:97–109, 1970.

[13] A. M. Johansen, L. Evers and N. Whiteley. *Monte Carlo Methods*, Lecture Notes, Department of Mathematics, Bristol University, 2008.

[14] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME-Journal of Basic Engineering*, 35–45, 1960.

[15] S. Kotz, N. Balakrishnan and N. L. Johnson. *Continuous Multivariate Distributions*, volume 1, Models and Applications. John Wiley & Sons, 2000.

[16] S. L. Lauritzen. *Thiele: Pioneer in Statistics*. Oxford University Press, 2002.

[17] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2004.

[18] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, **44**(247):335–341, 1949.

[19] T. Minka. Expectation Propagation for approximate Bayesian inference. PhD thesis, MIT, 2001.

[20] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[21] J. R. Norris. *Markov Chains*. Cambridge University Press, 1997.

[22] P. Park and T. Kailath. New square-root smoothing algorithms. *IEEE Transactions on Automatic Control*, **41**:727–732, 1996.

[23] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognation. *Proceedings of the IEEE*, **77**(2):257–286, 1989.

[24] H. Rauch, F. Tung and C. Striebel. Maximum likelihood estimates of linear dynamic systems. *American Institute of Aeronautics and Astronautics Journal*, **3**(8):1445–1450, 1965.

[25] R. L. Stratonovich. Application of the theory of Markoff processes in optimal signal discrimination. *Radio Engineering and Electronic Physics*, **5**(11):1–19, 1960. Translated from Russian.

[26] M. Verhaegen and P. Van Dooren. Numerical Aspects of Different Implementations. *IEEE Transactions On Automatic Control*, **31**(10):907–917, 1986.

[27] M. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, **1**:1–305, 2008.

Contributors

David Barber, Department of Computer Science, University College London
A. Taylan Cemgil, Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey
Silvia Chiappa, Statistical Laboratory, Centre for Mathematical Sciences, University of Cambridge