CSCI 2950-P Homework 3: Inference & Learning for Temporal State Space Models

Brown University, Spring 2013

Homework due at 11:59pm on April 12, 2013

In this problem set, we explore inference and learning algorithms for the *switching state space model*, which is summarized by the graphical model of Fig. 1. Our observations are a sequence of real-valued vectors, $y_t \in \mathbb{R}^p$, which depend on corresponding latent state vectors $x_t \in \mathbb{R}^d$. The temporal evolution of the state and observation vectors is influenced by discrete "switch" variables $z_t \in \{1, 2, \ldots, K\}$ as follows:

$$p(y_t \mid x_t, z_t = k) = \text{Norm}(y_t \mid C_k x_t, R_k)$$
$$p(x_t \mid x_{t-1}, z_t = k) = \text{Norm}(x_t \mid A_k x_{t-1}, Q_k)$$

If $K = 1$ so that $z_t = 1$ for all times, we recover a standard linear Gaussian state space model. In this case, we omit the subscripts above and denote the model parameters by $\{A, C, Q, R\}$. More generally, the discrete switch variables evolve according to a Markov chain:

$$p(z_t \mid z_{t-1} = k) = \text{Cat}(z_t \mid \pi_k)$$

Thus, $\pi_{k\ell}$ is the probability that $z_t = \ell$ given $z_{t-1} = k$. We assume that z_1 is uniformly distributed across the K possible switch modes, and $p(x_1) = \text{Norm}(x_1 \mid 0, I_d)$.

In general, many different state space models may assign equal likelihood to a given dataset, due to generalized versions of the rotational ambiguities underlying PCA and factor analysis models. For all models we consider, we thus constrain $C = [I_p \ O]$, where O is a $p \times (d - p)$ matrix of zeros, and I_p is a $p \times p$ identity matrix. When $d = p$, we have $C = I_p$.

For some experiments, we define our state space model parameters to take one of a few canonical forms. The *constant position* model takes $d = p$, and

$$A = I_p, \quad Q = \sigma_x^2 I_p, \quad R = \sigma_y^2 I_p,$$

so that the observations are noisy estimates of an underlying random walk. The *constant velocity* model generates more smooth motions by taking $d = 2p$, and

$$A = \begin{bmatrix} I_p & I_p \\ O_p & I_p \end{bmatrix}, \quad Q = \begin{bmatrix} \sigma_x^2 I_p & O_p \\ O_p & \sigma_v^2 I_p \end{bmatrix}, \quad R = \sigma_y^2 I_p,$$

where O_p is a $p \times p$ matrix of zeros. More generally, we will use the expectation maximization (EM) algorithm to learn model parameters $\{A, Q, R\}$ from observation sequences.

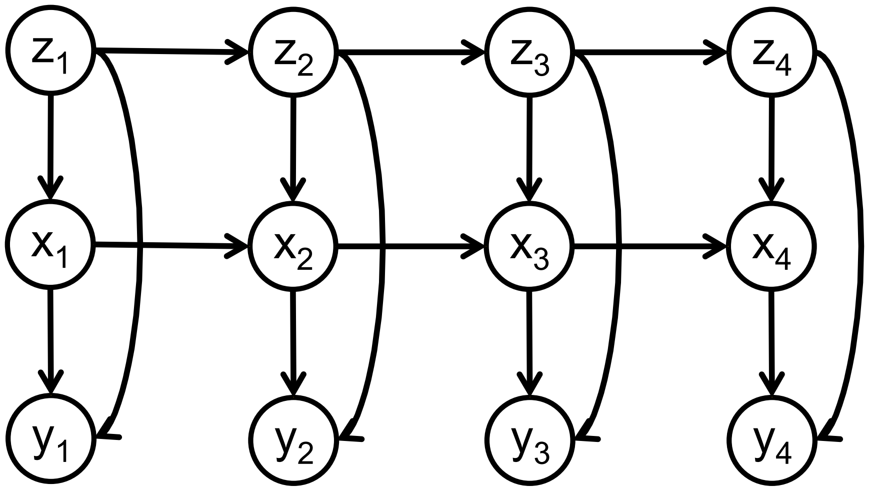

Figure 1: A directed graphical model representing a switching state space model. Discrete switch variables z_t evolve according to a Markov chain. At each time step, the switch z_t selects which state space model determines the continuous state x_t and observations y_t.

The following questions ask you to implement variants of the Kalman filter, Kalman smoother, and EM algorithm for linear-Gaussian state space models. For a compact reference to the required update equations, see *Parameter Estimation for Linear Dynamical Systems*, Ghahramani and Hinton, Technical Report CRG-TR-96-2, Feb. 1996. You can use results from this report, or other course readings, without proof.

Question 1: Tracking, Kalman Filters, & Particle Filters

Throughout this question we assume a non-switching state space model (K=1).

a) *Implement the Kalman filter, which recursively computes $p(x_t \mid y_1, \ldots, y_t)$ for any linear state space model, parameterized as above. Given a sequence of T observations, your code should produce a $d \times T$ matrix of posterior mean estimates, and a $d \times d \times T$ array of posterior covariance estimates.*

b) *Consider the $p = 1$-dimensional observation sequence we provide in* `track`. *It was sampled from a constant velocity model with $d = 2, \sigma_x^2 = \frac{1}{3}\sigma_v^2, \sigma_v^2 = 0.1^2, \sigma_y^2 = 20$. Using this correct model, apply your Kalman filter code. On one set of axes, plot the observation sequence, the posterior mean of the first (position) component of the state, and posterior confidence intervals. The confidence intervals should be determined as the posterior mean plus or minus two times the posterior standard deviation of the first state component.*

c) *Again consider the* `track` *data, and suppose that you incorrectly assumed it had been generated by a constant position model with $d = 1$, and σ_x^2, σ_y^2 as in part (b). Apply your Kalman filter code using this alternative model, and plot your results as in part (b).*

d) *Implement a basic "bootstrap" particle filter, which proposes particles from the state transition distribution $p(x_t \mid x_{t-1})$, reweights with the observation likelihood $p(y_t \mid x_t)$, and resamples weighted particles (with replacement) at each time step.*

e) Apply your particle filter implementation to the model and data from part (b), using 100 particles. At each time step, estimate the posterior mean as the weighted mean (before resampling) of the current particle locations. Repeat this experiment five times, and on a single set of axes, plot the five particle filter mean estimates together with the optimal Kalman filter mean estimates.

f) Repeat part (e) using 500 particles and 1000 particles. Discuss the robustness of the particle filter, compared to the Kalman filter.

g) We finish by exploring robustness to outliers. Again consider the `track` *data and independently at each time step t, with probability 0.1 replace the true observation by a sample $y_t \sim Norm(0, 40^2)$. Apply your Kalman filter to this corrupted data, with the (now inaccurate) original observation likelihood. Also apply a particle filter with 1000 particles, and an observation likelihood that correctly models the outlier process. Plot the corrupted observation data, as well as the mean estimates for both methods, and discuss.*

Question 2: Learning, Kalman Smoothers, & Expectation Maximization (EM)

Throughout this question we assume a non-switching state space model (K=1).

a) Implement a Kalman smoother which efficiently computes $p(x_t \mid y_1, \ldots, y_T)$ for any linear state space model. Use the Rauch-Tung-Striebel (RTS) smoothing algorithm (see Ghahramani and Hinton, or Chap. 15 of Jordan's textbook), which combines the Kalman filter from part 1(a) with a backwards smoothing pass.

b) Apply your Kalman smoother to the constant velocity model and `track` *data from part 1(b). On one set of axes, plot the observation sequence, the posterior mean of the first (position) component of the state, and posterior confidence intervals. Compare to the estimates produced by the Kalman filter.*

c) The marginal log-likelihood of an observation sequence y, integrating over states x for some fixed state space model parameters, can be written as follows:

$$\begin{aligned}\log p(y) &= \log p(y_1) + \sum_{t=1}^{T-1} \log p(y_{t+1} \mid y_1, \ldots, y_t) \\ &= \log \int_{\mathcal{X}_1} p(y_1 \mid x_1) p(x_1)\, dx_1 + \sum_{t=1}^{T-1} \log \int_{\mathcal{X}_t} p(y_{t+1} \mid x_t) p(x_t \mid y_1, \ldots, y_t)\, dx_t\end{aligned}$$

Use these expressions to efficiently evaluate the log-likelihood in closed form, based on the output of the Kalman filter. Write code implementing this log-likelihood computation.

d) Implement the EM algorithm for learning linear state space models. Follow Ghahramani and Hinton, except keep C fixed (as previously discussed) rather than re-estimating it in the M-step. Assume the first E-step is initialized with some set of valid state space model parameters. After each M-step, compute the log-likelihood bound from part (c); these bounds should monotonically increase. Hint: *You must use Kalman filter estimates* *based on the latest model parameters in evaluating this bound,* not *those from the previous E-step. To debug, create a simple model, generate synthetic data from that model, and verify that EM recovers similar parameters.*

e) *Consider the $p = 2$-dimensional observation sequence we provide in* `spiral`*, and use the EM algorithm to learn a $d = 2$-dimensional state space model. Initialize with a constant position model with $\sigma_x^2 = 1, \sigma_y^2 = 1$. Run the EM algorithm for 100 iterations, plot the marginal log-likelihood versus iteration, and report the learned parameters after the final iteration. Plot the observation sequence, and the output of the Kalman smoother after the final iteration, as overlaid 2-dimensional curves.*

f) *Repeat part (e) using the $p = 2$-dimensional observation sequence we provide in* `attractor`*.*

Question 3: Tracking in Switching State Space Models

We now consider the full switching state space model, and apply it to modeling the tracked motions of dancing honeybees. You can read more about this data at

`http://www.cc.gatech.edu/~borg/ijcv_psslds/`

Honeybees communicate via dances that can be approximated as switching among $K = 3$ behaviors: waggle, turn left, turn right. *Using training data in which behaviors z_t and bee poses y_t are both available, we will learn a switching state space model of bee motion.*

We provide three sequences of bee motion. In the questions below we analyze three train-test splits, where each split uses two training sequences and one test sequence. Our bee pose representation y_t is $p = 4$-dimensional: two position coordinates, and two orientation coordinates. For an orientation angle θ, the orientation coordinates are $cos(\theta), sin(\theta)$, and we approximate their behavior-specific conditional distributions as Gaussian to simplify learning and inference. To estimate an angle from inferred angle coordinates, first renormalize those coordinates to lie on the unit circle, and then apply an inverse-trigonometric function.

a) *Assuming training data in which the active behaviors z_t are observed, specify formulas for maximum likelihood estimation of the transition parameters π. Compute and report these ML estimates for each training split.*

b) *Suppose that your state space model training data is not a single sequence, but rather multiple sequences independently sampled from a common set of $\{A, C, Q, R\}$ model parameters. Specify an extension of the EM algorithm from problem 2 suitable for this case. Justify your approach, but a formal derivation of all steps is not required.* Hint: *During the E-step, the sequences are independent conditioned on the model parameters. The M-step should aggregate information across all sequences.*

c) *In general, bees stay in maneuver modes for many time steps before switching to other maneuver types. Extract "segments" corresponding to the contiguous time blocks where $z_t = k$ for $k = 1, 2, 3$, and store these as a set of training sequences for learning the corresponding state space model parameters $\{A_k, C_k, Q_k, R_k\}$.*

d) *Set $d = p = 4$, and use the EM algorithm from part (b) to learn a state space model for each of the three behavior modes, using the data from part (c). In each case, initialize* *using a constant position model with $\sigma_x^2 = 1, \sigma_y^2 = 1$. For each train-test split, plot the log-likelihood bounds after each iteration, and report the learned model parameters.*

e) *We now apply the learned state space model to track bee motion. Simulating the behavior of sometimes-inaccurate vision-based trackers, generate a noisy observation sequence $\tilde{y}$ which depends on the true poses y as follows:*

$$p(\tilde{y}_t \mid y_t) = 0.9 Norm(\tilde{y}_t \mid y_t, 0.1 I_4) + 0.1 Norm(\tilde{y}_t \mid 0, 5 I_4)$$

Derive an explicit formula for $p(\tilde{y}_t \mid x_t)$.

f) *Apply a "bootstrap" particle filter to bee tracking, where each particle represents a joint configuration (z_t, x_t) of the hidden variables at a given time point. For each train-test split, use switching state-space model parameters learned in parts (a) and (d), a noisy observation sequence generated as in part (e), and 1000 particles.*

g) *Analyze the results from part (f). For each sequence, plot (i) the inferred probability that each behavior is active over time, together with the true held-out behavior labels; (ii) the posterior mean of the bee's two-dimensional position over time, together with the true position; (iii) the posterior mean of the bee's orientation angle over time (after mapping angle coordinates to the unit circle), together with the true angle.*

h) Optional: *Repeat parts (d-g) with a higher state dimension of $d = 2p = 8$. Initialize with a constant velocity model with $\sigma_x^2 = 0.1, \sigma_v^2 = 1, \sigma_y^2 = 1$.*

Question 4: Gibbs Sampling for Switching State Space Models

While particle filters provide a natural approach for tracking in switching state space models, smoothed analysis of batch observation sequences is more challenging. One approach is to use a Gibbs sampler, which samples discrete modes z and continuous states x from their joint posterior given known model parameters. In this question, we derive (but do not implement) the Gibbs sampler update equations.

a) *Let $z_{\setminus t}$ denote all modes except z_t. Derive the form of $p(z_t \mid z_{\setminus t}, x, y)$, and give an explicit formula for the parameters of this K-dimensional categorical distribution.*

b) *Let $x_{\setminus t}$ denote all continuous states except x_t. Derive the form of $p(x_t \mid x_{\setminus t}, z, y)$, and give an explicit formula for this distribution as a member of some standard family of continuous probability densities.*

c) *The mixing rate of Gibbs samplers can often be improved by blocked resampling of groups of variables. Derive the form of $p(z_t, x_t \mid z_{\setminus t}, x_{\setminus t}, y)$. Suppose you have code available to sample from categorical distributions, and from multivariate normal distributions. Describe how you could use this code to draw a correct joint sample (z_t, x_t).*