TTIC 31170: Robot Learning and Estimation (Spring 2021)

Problem Set #1

Due Date: April 15, 2021

1 Bayes Mean-Square Error Estimator [20 pts]

Consider the case in which we are interested in a estimating a continuous, multivariate random variable $\boldsymbol{x} \in \mathbb{R}^n$ based upon noisy continuous, vector-valued observations. Suppose that the prior $p(\boldsymbol{x})$ over the latent random variable is known.

(a) **[4pts]** Derive an expression for the estimator that minimizes the mean-square error

$$\hat{\boldsymbol{x}}_{\text{BLS}}(\boldsymbol{z}) = \arg\min_{\boldsymbol{f}(\cdot)} \mathbb{E}_{\boldsymbol{x}|\boldsymbol{z}}\left[\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{z})\|_2^2\right] \tag{1}$$

where $\mathbb{E}_{\boldsymbol{x}|\boldsymbol{z}}[\cdot]$ denotes expectation with respect to $p(\boldsymbol{x} \mid \boldsymbol{z})$.[1]

(b) **[6pts]** Prove that an estimator $\hat{\boldsymbol{x}}(\cdot)$ minimizes the mean-square error (Eqn. 1) if and only if the associated estimation error $\boldsymbol{e}(\boldsymbol{x}, \boldsymbol{z}) = \hat{\boldsymbol{x}}(\boldsymbol{z}) - \boldsymbol{x}$ is orthogonal to any (vector-valued) function of the data $\boldsymbol{g}(\boldsymbol{z})$, i.e.,

$$\mathbb{E}_{\boldsymbol{x}}\left[(\hat{\boldsymbol{x}}(\boldsymbol{z}) - \boldsymbol{x})\boldsymbol{g}^\top(\boldsymbol{z})\right] = \boldsymbol{0} \tag{2}$$

(c) **[4pts]** Letting Σ_{BLS} be the error covariance of the estimator derived in (a), show that $\Sigma_{\text{BLS}} \leq \Sigma_e$, where Σ_e is the error covariance associated with any estimator $\hat{\boldsymbol{x}}(\cdot)$. **Hint**: use the orthogonality specified in Eqn. 2.

(d) **[2pts]** Derive the expression for the estimator that minimizes the weighted mean-square error

$$\hat{\boldsymbol{x}}_{\text{WLS}}(\boldsymbol{z}) = \arg\min_{\boldsymbol{f}(\cdot)} \int_{-\infty}^{\infty} (\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{z}))^\top S(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{z}))p(\boldsymbol{x} \mid \boldsymbol{z})d\boldsymbol{x},$$

where S is a positive definite matrix.

(e) **[4pts]** Derive the expression for the estimator that minimizes the expected loss $\mathbb{E}[L(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{z}))]$, where the loss has the form

$$L(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{z})) = \begin{cases} 0 & \text{if } \|\boldsymbol{f}(\cdot) - \boldsymbol{x}\| < \epsilon \\ 1 & \text{else} \end{cases}$$

where you can assume that $\epsilon \to 0$.

[1] Note that this is equivalent to minimizing the expectation with respect to $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})$, as we effectively can minimize the above quantity for every possible $\boldsymbol{z}$, i.e., optimize the bracketed term in arg min $\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\|(x) - \boldsymbol{f}(\boldsymbol{z})\|_2^2\, p(\boldsymbol{x} \mid \boldsymbol{z})d\boldsymbol{x}\right] d\boldsymbol{z}$ for every $\boldsymbol{z}$.

2 Linear Estimators [30 pts]

The optimal mean-square error estimator is not guaranteed to be linear in the observations. In such cases, we may prefer a linear estimator for computational reasons.

Consider a system with a latent state represented by a deterministic (but unknown) continuous quantity x. Suppose that you have two observations of x corrupted by additive noise,

$$z_1 = x + v_1$$
$$z_2 = x + v_2,$$

where $v_1 \sim \mathcal{N}(0, \sigma_1^2)$ and $v_2 \sim \mathcal{N}(0, \sigma_2^2)$ are Gaussian random variables with zero-mean and variance σ_1^2 and σ_2^2, respectively.

Your task is to derive an estimator $\hat{x} = k_1 z_1 + k_2 z_2$ for x that is linear in the observations z_1 and z_2.

(a) **[6pts]** Assuming that the additive noises are independent, i.e., $\mathbb{E}(v_1 v_2) = 0$:

(i) Derive a constraint on k_1 and k_2 that ensures that the estimator is unbiased.

(ii) Derive the settings for k_1 and k_2 that minimize the mean-square error $\mathbb{E}(\tilde{x}^2)$, where $\tilde{x} = x - \hat{x}$. Can you provide intuition for these values for k_1 and k_2?

(iii) What is the resulting mean-square estimation error and how does it compare to the corresponding error that would result from using either z_1 or z_2 to estimate x?

(b) **[4pts]** Repeat the above derivation for the case that the noise terms are correlated, i.e., $\mathbb{E}[v_1 v_2] = \rho \sigma_1 \sigma_2$, where ρ is the correlation coefficient ($|\rho| \leq 1$).

(c) **[4pts]** More generally, suppose that we are interested in a estimating a continuous, vector-valued quantity $\boldsymbol{x} \in \mathbb{R}^n$ based upon continuous, vector-valued observations $\boldsymbol{z} \in \mathbb{R}^m$ ($m > n$) that are linear in $\boldsymbol{x}$

$$\boldsymbol{z} = H\boldsymbol{x} + \boldsymbol{v},$$

where $\boldsymbol{v}$ denotes additive random noise.

(i) Assuming that $\boldsymbol{x}$ is a non-random constant, derive the expression for the *least-squares estimator* $\hat{\boldsymbol{x}}$ that minimizes the sum of squares of the measurement residual $z_i - \hat{z}_i$, i.e.,

$$\hat{\boldsymbol{x}}(\cdot) = \arg\min_{\boldsymbol{f}(\cdot)} \|\boldsymbol{z} - H\boldsymbol{f}(\boldsymbol{z})\|_2^2.$$

(ii) Derive the expression for the weighted least-squares estimator that minimizes the weighed sum of squares

$$\hat{\boldsymbol{x}}(\cdot) = \arg\min_{\boldsymbol{f}(\cdot)} \|\boldsymbol{z} - H\boldsymbol{f}(\boldsymbol{z})\|_R^2$$

where R is a symmetric, positive definite matrix that specifies the weights associated with deviations.

(d) **[4pts]** Now, suppose that you have a discrete-time sequence of observations of the form $z_t = x + v_t$ for $t \in \{1, \ldots, T\}$, where v_t is a zero-mean Gaussian white sequence (i.e., elements of the sequence are uncorrelated, $\mathbb{E}[v_s v_t] = 0 \; \forall s \neq t$).

(i) Derive the expression for the minimum variance unbiased estimator $\hat{x}(z^T)$.

(ii) The above filter requires storing the entire sequence of observations. In the interest of having an estimator with constant memory requirements, derive a recursive estimator $\hat{x}_t$ in terms of the current measurement z_t and previous estimate $\hat{x}_{t-1}$.

(e) **[12pts]** Consider the case in which $\boldsymbol{x}$ is not deterministic, but rather is a multivariate stochastic process with linear, potentially time-variant dynamics

$$\boldsymbol{x}_k = A_k \boldsymbol{x}_{k-1} + \boldsymbol{w}_{k-1},$$

where $\boldsymbol{w}_{k-1}$ is a zero-mean white sequence. We have access to noisy vector-valued observations that are linear in the latent state

$$\boldsymbol{z}_k = H_k \boldsymbol{x}_k + \boldsymbol{v}_k,$$

where $\boldsymbol{v}_k$ is zero-mean noise with covariance R. For memory and computational efficiency, we would like to find a linear, recursive estimator of the form

$$\hat{\boldsymbol{x}}_k^+ = K_k^- \hat{\boldsymbol{x}}_k^- + K_k \boldsymbol{z}_k, \tag{3}$$

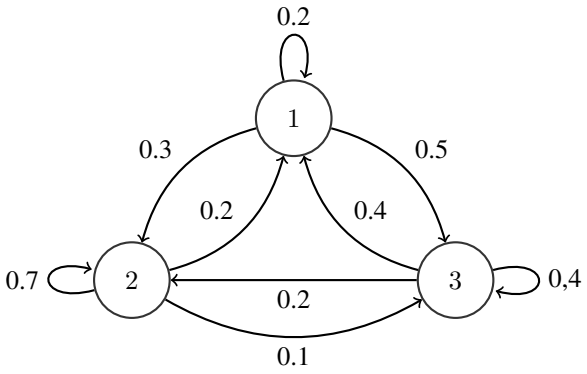
where $\hat{\boldsymbol{x}}_k^-$ and $\hat{\boldsymbol{x}}_k^+$ denote the estimates before and after incorporating the latest observation $\boldsymbol{z}_k$, respectively. The terms K_k^- and K_k are time-varying matrices that weigh the contributions of the previous estimate $\hat{\boldsymbol{x}}_k^-$ (which incorporates all previous observations $\boldsymbol{z}^{k-1}$) and the latest observation $\boldsymbol{z}_k$.

(i) Assuming that $\hat{\boldsymbol{x}}_k^-$ is unbiased (i.e., $\mathbb{E}[\tilde{\boldsymbol{x}}_k^-] = \mathbf{0}$), derive an expression for K_k^- in terms of K_k that ensures that the estimator remains unbiased. **Hint**: Find a recursive formulation for the estimation error $\tilde{\boldsymbol{x}}_k^+$ in terms of $\boldsymbol{x}_k$ and $\tilde{\boldsymbol{x}}_k^-$.

(ii) What is the resulting expression for the estimator (Eqn. 3)? In a few sentences, explain what it is doing.

(iii) Derive an expression for the estimator covariance $\Sigma_k^+ = \mathbb{E}[\tilde{\boldsymbol{x}}_k^+ {\tilde{\boldsymbol{x}}_k^+}^\top]$ in terms of Σ_k^-. **Hint**: Remember that the noise is white and uncorrelated with the state.

(iv) Up to this point, we have derived everything in terms of K_k as well as known parameters, but we haven't decided on the best setting for K_k. We would like to find a setting for K_k that minimizes the weighted mean-square error $J_k = \mathbb{E}[\tilde{\boldsymbol{x}}_k^+ S {\tilde{\boldsymbol{x}}_k^+}^\top]$, where $S \geq 0$. In fact, you can show that the optimal setting for the gain matrix is independent of S, so for simplicity let's assume that $S = I$, in which case $J_k = \text{trace}[\Sigma_k^+]$. Derive the expression for the optimal gain matrix. **Hint**: For matrices A and B, where B is symmetric, $\frac{\partial}{\partial A}\text{trace}[ABA^\top] = 2AB$.



		X_t		
		1	2	3
Z_t	1	0.6	0.1	0.2
	2	0.1	0.7	0.3
	3	0.3	0.2	0.5

Figure 1: State transition diagram.

3 Hidden Markov Models: Filtering and Smoothing [20 pts]

Consider a hidden Markov model consisting of a discrete-valued, latent process $X_t \in \{1, 2, 3\}$ and observations $Z_t \in \{1, 2, 3\}$. Figure 1 presents the state transition diagram with the transition and observation (emission) $P(Z_t \mid X_t)$ probabilities. The distribution over the initial state is uniform.

Assume that we observed the sequence $Z^T = \{2, 3, 1\}$.

(a) **[6pts]** Determine the maximum a posteriori (MAP) state estimate at each step using filtering, i.e., $X_t^* = \arg\max_{X_t} P(X_t \mid Z^t)$.

(b) **[6pts]** Determine the maximum a posteriori (MAP) state estimate at each step using smoothing, i.e., $X_t^* = \arg\max_{X_t} P(X_t \mid Z^T)$.

(c) **[8pts]** Determine the MAP state sequence, i.e., $\{X_1, X_2, X_3\}^*$, using the Viterbi algorithm. Compare the result with the sequence of MAP states from filtering and smoothing.

$$\{X_1, X_2, X_3\}^* = \arg\max_{X_1, X_2, X_3} p(X_1, X_2, X_3 \mid Z_1, Z_2, Z_3)$$

4 Hidden Markov Models: Robot Localization [40 pts]

Assume that we have a wheeled robot that moves about in a small environment that consists of a collection of 16 rooms arranged in a 4×4 grid. Each room is painted either red, green, blue, yellow, or black. The robot is free to enter all but the four rooms that are used as storage, which are painted black. Figure 2 depicts this simple environment.

At each time step, the robot chooses to either stay where it is or move up, down, left, or right (it can not move diagonally). These decisions are made at random and don't depend upon where the robot was previously. If the robot attempts to move to a storage room or outside the confines of the environment, it will draw another action until it chooses one that is valid. When the robot takes the action, it observes the color of the resulting room using its camera. Unfortunately, the white balance is off and the camera may yield an inaccurate observation of the color.

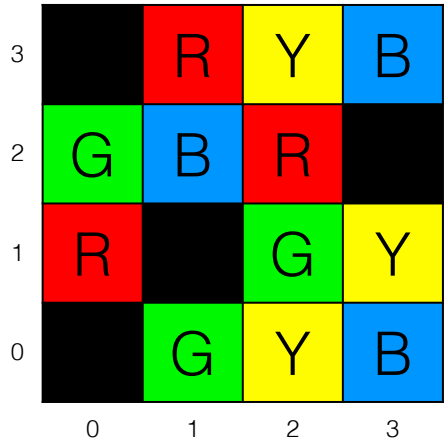


Figure 2: The robot is free to occupy any cell in the 4×4 grid except those colored black.

For this problem, the state is the robot's position in the environment, (X, Y) where $X \in \{0, 1, 2, 3\}$ and $Y \in \{0, 1, 2, 3\}$, and the output is the observed color, $Z \in \{r, g, b, y\}$. For example, a ten step random walk through the environment of the robot might look like:

$$
\begin{aligned}
t = 0: &\quad (1, 3), r \\
t = 1: &\quad (2, 3), y \\
t = 2: &\quad (3, 3), b \\
t = 3: &\quad (3, 3), g \\
t = 4: &\quad (2, 3), r \\
t = 5: &\quad (2, 3), y \\
t = 6: &\quad (2, 3), y \\
t = 7: &\quad (2, 3), y \\
t = 8: &\quad (3, 3), b \\
t = 9: &\quad (3, 3), b
\end{aligned}
$$

In this example, the robot starts in the room at $(1, 3)$ where it correctly observes red, moves to the room at $(2, 3)$ and correctly observes yellow, moves to the room at $(3, 3)$ and first correctly observes blue, but after staying in the same room, observes its color as green. Note that it may be advantageous to represent each cell by a number $\{0, 1, \ldots, 15\}$ and use that as the state rather than the coordinate.

We would like to develop an algorithm that estimates the maximum a posteriori (MAP) hidden (latent) trajectory of the robot based upon the sequence of room color observations. As we've seen in lecture, we can model this estimation problem as a hidden Markov model, since the robot's motion obeys the first-order Markov property and the observation likelihood is only a function of the robot's current state. We can then use the forward-backward and Viterbi algorithms to determine the MAP estimates of the robot's state or sequence of states, respectively.

(a) **[20pts]** Formulating this problem as an HMM requires an estimate of the transition, observation, and prior probabilities. While we don't know these, we can estimate them from observation data using the Baum-Welch algorithm. Doing so requires an initial estimate for the HMM parameters. Assume a uniform prior over the initial states; that the robot stays in the current state with likelihood 0.2 and the remaining probability is evenly distributed among valid adjacent states; and that the camera observes the correct color with probability

0.7, with the remaining probability distributed evenly among the other three colors. Starting with this estimate, iterate through successive HMM models, using the (log-)likelihood of the data to determine convergence (i.e., when the change is below a user-defined threshold.) Tip: The $\alpha_t(i)$ and $\beta_t(i)$ terms often become very small. To avoid numerical errors, you should normalize the vector $\boldsymbol{\alpha}_t$ of $\alpha_t(i)$ values at each time step, and use the same normalization factor for the corresponding $\boldsymbol{\beta}_t$ vector

$$C_t = \sum_{i=1}^{n} \alpha_t(i)$$
$$\bar{\alpha}_t(i) = \alpha_t(i)/C_t$$
$$\bar{\beta}_t(i) = \beta(i)/C_t$$

As part of this problem set, we are providing you with 200 training sequences (random walks) of state-observation pairs, with each sequence consisting of 201 time steps. These sequences are contained in the text file `randomwalk.train.txt`, with sequences separated by a period. Note that you should only be using the observations. We are providing the states so that you can sanity-check your parameter estimates with those determined using knowledge of the true state (be sure to normalize).

(i) Implement the Baum-Welch algorithm as part of the `train` function defined in `HMM.py`, which provides the structure for an HMM class implementation. The `train` function takes as input a set of observation sequences and estimates the transition, emission, and state prior probabilities for the HMM.

Included with this problem set are the file `DataSet.py`, which provides the structure for parsing the sequence data, as well as the file `TrainHMM.py`, which reads in the training data (via the `DataSet` class), calls `HMM.train`, and saves the resulting model as `trained-model.pkl`. The function can be called as follows:

```
$ python TrainHMM.py trainingdata.txt
```

Feel free modify these files or to write your own.

(ii) Plot the (log-)likelihood of the observations (i.e., the (log-)likelihood of the data) as a function of iteration number.

(b) **[20pts]** Having identified the likelihoods that define the HMM representation, the next task is to write code that determines the most likely sequence of states (robot locations) based upon the corresponding sequence of observed room colors.

Included as part of this problem set is a second text file `randomwalk.test.txt` that includes 200 sequences of state-observation pairs, with each sequence consisting of 201 time steps. The locations are provided simply for verification—you should only use the sequence of room color observations.

(i) Implement the Viterbi algorithm as part of the `viterbi` function within the `HMM` class. The problem set includes a Python file `RunViterbi.py` that loads in the test data as well as the model generated above, and then calls `HMM.viterbi` on the observations. The code ban be called as:

```
$ python RunViterbi.py testingdata.txt trained-model.pkl
```

Note that `trained-model.pkl` is optional and with out it, the function will use the initial model described in Part (a). It may be useful to compare the accuracy when using your learned model to the default model.

(ii) Using the data available in the file `randomwalk.test.txt`, measure the number of times that the ground-truth state is different from the state in your MAP sequence.

What to hand in: All the code necessary to (a) train the HMM and (b) determine the maximum likelihood state sequence, the data likelihood plot, and the MAP sequence error count.

5 Time and Collaboration Accounting

(a) **[0.5pts]** Did you work with anyone on this problem set? If so, who?

(b) **[0.5pts]** How long (in hours) did you spend on this problem set (exclusive of going through lecture notes, doing readings, etc.)?