In classification problems, we often encounter hybrid probability distributions. For instance, a measurement \( X = x \) out of the uncountable set \( X = \mathbb{R}^d \) may be made when an object \( T = t \) out of a finite list \( T = \{ t_0, \ldots, t_{K-1} \} \) is observed (the capital letter denotes the random variable, and the lower case letter denotes its values). Then the joint, cumulative distribution of random variables \( X \) and \( T \) is defined as the probability
\[
P_{X,T}(x, t) = \mathbb{P}[X \leq x \text{ and } T = t]
\]
where
\[
X \leq x \quad \text{stands for} \quad X_1 \leq x_1 \quad \text{and} \quad \ldots \quad \text{and} \quad X_d \leq x_d .
\]
The derivative of \( P_{X,T} \) with respect to \( x \) is called the joint distribution of \( X \) and \( T \):
\[
p_{X,T}(x, t) = \frac{\partial^d P_{X,T}(x, t)}{\partial x_1 \cdots \partial x_d}.
\]
Typically, the subscripts \( X, T \) are omitted for simplicity, and the joint distribution is simply written as \( p(x, t) \). From its definition, we see that
\[
\int_a^b p(x, t) \, dx = \mathbb{P}[a < X \leq b \text{ and } T = t] .
\]

In the following problems, let \( d = 2 \) and \( K = 3 \), and let \( X \) and \( T \) have the following joint distribution:
\[
p(x, t) = \begin{cases} 
\mu_0 a_{11} a_{21} & \text{for } 0 \leq x_1 \leq a_{11} \text{ and } 0 \leq x_2 \leq a_{21} \text{ and } t = 0, 1, 2 \\
0 & \text{otherwise}
\end{cases}
\]
(1)
where \( \mu_0 + \mu_1 + \mu_2 = 1 \).

All your calculations should be done for the general expression (1) above. Your plots and diagrams, on the other hand, should refer to the following numerical values:
\[
A = \begin{bmatrix} a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 \\ 2 & 1 & 2 \end{bmatrix} \quad \text{and} \quad \mu = \begin{bmatrix} \mu_0 & \mu_1 & \mu_2 \end{bmatrix} = \begin{bmatrix} 0.3 & 0.5 & 0.2 \end{bmatrix} .
\]
(2)

Figure 1 plots this joint probability. This is a piecewise-constant function from \( \mathbb{R}^2 \times \{ 0, 1, 2 \} \rightarrow \mathbb{R} \), so its graph lives in four dimensions. To display it, the function values are written into the appropriate rectangles of the domain.

(a) Show that equation (1) defines a probability distribution. The values in it are obviously nonnegative, so the only thing to check is that
\[
\sum_{t \in T} \int_X p(x, t) \, dx = 1 .
\]
(b) Write an expression in the style of equation (1) for the marginal probabilities \( p(x) \) and \( p(t) \). Show a plot of these distributions for the numerical values given in (2) above. Label your plot similarly to what is done in Figure 1. Write all numerical values as ratios of mutually prime integers, as done in Figure 1.

(c) Do the same for the conditional probabilities \( p(x \mid t) \) and \( p(t \mid x) \). To make sure that your plots are correct, verify that

\[
\int_X p(x \mid t) \, dx = 1 \quad \text{and} \quad \sum_T p(t \mid x) = 1.
\]

2. The file data.mat available on the homework web page contains a vector \texttt{xtrain} of observations, the corresponding vector \texttt{ttrain} of target values, and a vector \texttt{xtest} of observations for testing. All target values are assumed to be drawn from the probability distribution (1.60) in Bishop. The polynomial in equation (1.1) of Bishop has \( M = 3 \), and the values of \( x \) are between 0 and 1. Write Matlab code that implements the predictor (1.64) in Bishop and test it on \texttt{xtest}. Specifically:

(a) Hand in a printout of a function with header

function \([w, \sigma, ttest] = mle(xtrain, ttrain, M, xtest)\)

that implements your estimator. Do not show the rest of your code.

The only twist here is how to maximize (1.62) in Bishop. This is a least squares problem. See \texttt{polyfit} in Matlab. The function \texttt{polyval} may save you a few minutes of programming as well. Note that this assignment does not ask for the coefficients of the polynomial, so it is up to you whether you follow Bishop’s convention (1.1) or Matlab’s to store these.

(b) Show a plot in the style of Bishop’s Figure 1.2. The curve in this figure should be your maximum-likelihood polynomial. Dots should represent the points \((xtrain, ttrain)\). Also add error bars for the points \((xtest, ttest)\) using the Matlab function \texttt{errorbar}, with the argument \texttt{E} equal to your estimate of the standard deviation of noise. Set the \texttt{LineStyle} property to \texttt{none} for the error bars.

(c) Show the same plot for \( M = 2 \), and then for \( M = 9 \). Make sure you show which plot is which.