Problem 1

```matlab
>> colordef none
>> load data
>> whos
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST1</td>
<td>1x45</td>
<td>360</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>ST2</td>
<td>1x11</td>
<td>88</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>ST3</td>
<td>1x64</td>
<td>512</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>ST4</td>
<td>1x70</td>
<td>560</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>ST5</td>
<td>1x51</td>
<td>408</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>ST6</td>
<td>1x110</td>
<td>880</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X1</td>
<td>43x2</td>
<td>688</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td>9x2</td>
<td>44</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X3</td>
<td>62x2</td>
<td>992</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X4</td>
<td>68x2</td>
<td>1088</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X5</td>
<td>49x2</td>
<td>784</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>X6</td>
<td>108x2</td>
<td>1728</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>hmm1</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>hmm2</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>hmm3</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>hmm4</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>hmm5</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>hmm6</td>
<td>1x1</td>
<td>1316</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>mu_a</td>
<td>2x1</td>
<td>16</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>mu_e</td>
<td>2x1</td>
<td>16</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>mu_i</td>
<td>2x1</td>
<td>16</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>mu_o</td>
<td>2x1</td>
<td>16</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>mu_y</td>
<td>2x1</td>
<td>16</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>var_a</td>
<td>2x2</td>
<td>32</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>var_e</td>
<td>2x2</td>
<td>32</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>var_i</td>
<td>2x2</td>
<td>32</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>var_o</td>
<td>2x2</td>
<td>32</td>
<td>double</td>
<td></td>
</tr>
<tr>
<td>var_y</td>
<td>2x2</td>
<td>32</td>
<td>double</td>
<td></td>
</tr>
</tbody>
</table>
\begin{verbatim}
>> [X, stateSeq] = genhmm(hmm1);
>> figure; plotseq(X, stateSeq);

\end{verbatim}
```matlab
>> clf; [X, stateSeq] = genhmm(hmm1); plotseq(X, stateSeq);
```
>> clf; [X, stateSeq] = genhmm(hmm1); plotseq(X, stateSeq);

![Figure 1](image1)

>> clf; [X, stateSeq] = genhmm(hmm1); plotseq(X, stateSeq);

![Figure 1](image2)
>> clf; [X, stateSeq] = genhmm(hmm1); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm1); plotseq(X, stateSeq);
\[
\text{[X, stateSeq]} = \text{genhmm(hmm2)};
\]
\[
\text{figure; plotseq(X, stateSeq)};
\]

\[
\text{figure; plotseq2(X, stateSeq, hmm2)};
\]
```matlab
>> [X, stateSeq] = genhmm(hmm2); figure; plotseq(X, stateSeq);
```

```
>> clf; [X, stateSeq] = genhmm(hmm2); plotseq(X, stateSeq);
```
>> clf; [X, stateSeq] = genhmm(hmm2); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm2); plotseq(X, stateSeq);
>> [X, stateSeq] = genhmm(hmm3);
>> figure; plotseq(X, stateSeq);

>> figure; plotseq2(X, stateSeq, hmm3);
>> clf; [X, stateSeq] = genhmm(hmm3); plotseq(X, stateSeq);

Figure 2

>> clf; [X, stateSeq] = genhmm(hmm3); plotseq(X, stateSeq);

Figure 2
\texttt{>> clf; [X, stateSeq] = genhmm(hmm3); plotseq(X, stateSeq);}
>> [X, stateSeq] = genhmm(hmm4);
>> figure; plotseq(X, stateSeq);

figure; plotseq2(X, stateSeq, hmm4);
>> clf; [X, stateSeq] = genhmm(hmm4); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm4); plotseq(X, stateSeq);
>> clf; [X, stateSeq] = genhmm(hmm4); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm4); plotseq(X, stateSeq);
>> [X, stateSeq] = genhmm(hmm5);
>> figure; plotseq(X, stateSeq);

>> figure; plotseq2(X, stateSeq, hmm5);
>> clf; [X, stateSeq] = genhmm(hmm5); plotseq(X, stateSeq);
>> clf; [X, stateSeq] = genhmm(hmm5); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm5); plotseq(X, stateSeq);
>> [X, stateSeq] = genhmm(hmm6);
>> figure; plotseq(X, stateSeq);

>> figure; plotseq2(X, stateSeq, hmm6);

> Figure 3
> Figure 4
>> clf; [X, stateSeq] = genhmm(hmm6); plotseq(X, stateSeq);
>> clf; [X, stateSeq] = genhmm(hmm6); plotseq(X, stateSeq);

>> clf; [X, stateSeq] = genhmm(hmm6); plotseq(X, stateSeq);
1. You can verify a transition matrix is valid by observing and analyzing the graphical outputs above of the sequences. If over a large data set, the calculated probabilities match the transitional matrix, then it is valid.

2. The different transition matrices inform the user how probable it is to transition to each state from one specific state, for all states in each model. If a probability is zero, then there is no transition between those two states.

3. If there was no final state, no sequence would be accepted by the HMM. If these HMM’s were being used as part of a speech recognizer, this means that no sounds would be recognized by this HMM with no final state.

4a. In the case of HMMs with plain Gaussian emission probabilities, the quantities that should be present in the complete parameter set include the mean and variance.

4b. If the model is ergodic with N states (including the initial and final), and represents data of dimension D, the total number of parameters in the complete parameter set is as follows:

\[(D \times 1) + (D \times D) + (N \times N)\] for the mean matrix, variance matrix, and transition matrix.

5. I would use an ergodic HMM to model words. A sound can occur in the beginning, middle, or end of a word or sentence with few restrictions.
1. The following expression can be used to compute the log of a sum given the logs of the sum’s arguments:

$$\log(a + b) = f(\log(a), \log(b)) = \log(a) + \log\left(1 + e^{(\log(b) - \log(a))}\right)$$

Demonstrate the validity of this expression.

There are two choices of using the previous expression:

$$\log(a + b) = \log(a) + \log\left(1 + e^{(\log(b) - \log(a))}\right)$$
$$\log(b + a) = \log(b) + \log\left(1 + e^{(\log(a) - \log(b))}\right)$$

If $\log(a) > \log(b)$ which version leads to the most precise implementation?

Since $e^{(\log(b) - \log(a))} = e^{(\log(b/a))} = b/a$

So $\log(a) + \log\left(1 + e^{(\log(b) - \log(a))}\right) = \log(a) + \log(1 + (b/a)) = \log(a * ((1 + (b/a)) = \log(a + b)$

The most precise implementation comes from the top equation if $\log(a) > \log(b)$ as per the example below.

```
>> log(.6)       >> log(.2)       >> log(.4)
ans =          ans =             ans =
-0.5108        -1.6094         -0.9163

Bottom equation
>> log(.4) - log(.2)
ans =
0.6931
>> exp(.6931)
ans =
1.9999
>> ans+1
ans =
2.9999
>> log(ans)
ans =
1.0986
>> ans+log(.2)
ans =
-0.5109
```

```
Top equation
>> log(.2) - log(.4)
ans =
-0.6931
>> exp(ans)
ans =
0.5000
>> ans+1
ans =
1.5000
>> log(ans)
ans =
0.4055
>> ans + log(.4)
ans =
-0.5108
```
2. Express the log version of the forward recursion. In addition to the arithmetic precision issues, what are the other computational advantages of the log version?

1. **Initialization**

   $$\alpha_1(i) = a_i b_i(x_i), \quad 2 \leq i \leq N - 1$$

   $$\log(\alpha_1(i)) = \log(a_i) + \log(b_i(x_i))$$

2. **Recursion**

   $$\alpha_{t+1}(j) = \left[ \sum_{i=2}^{N-1} \alpha_t(i) \cdot a_{ij} \right] b_j(x_{t+1}), \quad 1 \leq t \leq T, \quad 2 \leq j \leq N - 1$$

   $$\log(\alpha_{t+1}(j)) = \log \left( \left[ \sum_{i=2}^{N-1} \alpha_t(i) \cdot a_{ij} \right] b_j(x_{t+1}) \right)$$

3. **Termination**

   $$p(X | \Theta) = \left[ \sum_{t=2}^{N-1} \alpha_t(i) \cdot a_{iN} \right]$$

   $$\log(p(X | \Theta)) = \log \left( \left[ \sum_{t=2}^{N-1} \alpha_t(i) \cdot a_{iN} \right] \right)$$

We can accurately compute the probabilities of longer sequences because we avoid the buffer underflow problem by using the logarithmic version.
Problem 2

A. The additional quantities and assumptions we need to perform a true Bayesian classification rather than a Maximum Likelihood classification of the sequences is as follows:

We need the values on the righthand side of the equation.

\[ P(W | A) = \frac{P(A | W)P(W)}{P(A)} \]

Where \( P(W) \) is the probability that the word string \( W \) will be uttered, \( P(A | W) \) is the probability that when \( W \) was uttered the acoustic evidence \( A \) will be observed, and \( P(A) \) is the average probability that \( A \) will be observed, also called the evidence.

B. The additional condition to would be to require the value of the “evidence” to be constant across data updates makes the result of Bayesian classification equivalent to the result of ML classification.
Problem 3

>> NumModels = 6;
>> for i=1:NumModels
  for j=1:NumModels
    stri = num2str(i);
    strj = num2str(j);
    eval(['logProb(', stri, ',', strj, ') = logfwd(X', stri, ',hmm', strj, ');']);
  end;
end;

>> logProb
logProb =
1.0e+003 *
  -0.5594  -0.6212  -1.4398  -1.4211  -1.2945  -0.9932
  -0.1160  -0.1175  -0.1114  -0.2462  -0.1402
  -0.8265  -0.7878  -1.1802  -1.1543  -0.7410  -1.3291
  -0.8789  -0.8233  -0.8551  -0.8203  -1.1583  -1.0779
  -0.7765  -0.7609  -0.8116  -0.7889  -0.9978  -0.6035
  -1.3968  -1.3228  -1.9053  -1.8520  -2.1105  -2.0234

>> [val, indx] = max(logProb')
val =
1.0e+003 *
  -0.5594  -0.1114  -0.7410  -0.8203  -0.6035  -1.3228
indx =
  1      3      5      4      6      2

| Sequence | $\log P(X|\Theta_1)$ | $\log P(X|\Theta_2)$ | $\log P(X|\Theta_3)$ | $\log P(X|\Theta_4)$ | $\log P(X|\Theta_5)$ | $\log P(X|\Theta_6)$ | Most Likely Model |
|----------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|------------------|
| $X_1$    | -559.4               | -621.2               | -1439.8              | -1421.1              | -1294.5              | -993.2               | 1                |
| $X_2$    | -116.0               | -117.5               | -111.4               | -114.5               | -246.2               | -140.2               | 3                |
| $X_3$    | -826.5               | -787.8               | -1180.2              | -1154.3              | -741.0               | -1329.1              | 5                |
| $X_4$    | -878.9               | -823.3               | -855.1               | -820.3               | -1158.3              | -1077.9              | 4                |
| $X_5$    | -776.5               | -760.9               | -811.6               | -788.9               | -997.8               | -603.5               | 6                |
| $X_6$    | -1396.8              | -1322.8              | -1905.3              | -1852.0              | -2110.5              | -2023.4              | 2                |
Problem 4

A. The delta variable in the Viterbi algorithm is computed from the maximum likelihood along the best path. The alpha variable in the forward algorithm is computed from all possible paths.

B. Derive a log version of the Viterbi algorithm.

A. Initialization

\[
\delta_t(i) = a_{x_i} \cdot b_i(x_t), \quad 2 \leq i \leq N - 1
\]
\[
\psi_{t+1} = 0
\]

\[
\log (\delta_t(i)) = \log (a_{x_i}) + \log (b_i(x_t))
\]

B. Recursion

\[
\delta_{t+1}(j) = \max_{2 \leq i \leq N-1} \left\{ \delta_t(i) \cdot a_{x_{t+1}} \cdot b_j(x_{t+1}) \right\}, \quad 1 \leq t \leq T
\]
\[
\psi_{t+1} = \arg \max_{2 \leq j \leq N-1} \left[ \delta_t(j) \cdot a_{x_{t+1}} \right], \quad 1 \leq t \leq T
\]

\[
\log (\delta_{t+1}(j)) = \log (\max_{2 \leq i \leq N-1} \left[ \delta_t(i) \cdot a_{x_{t+1}} \right]) + \log (b_j(x_{t+1}))
\]

\[
\log (\psi_{t+1}) = \log (\arg \max_{2 \leq j \leq N-1} \left[ \delta_t(j) \cdot a_{x_{t+1}} \right])
\]

C. Termination

\[
p^*(X | \Theta) = \max_{2 \leq i \leq N-1} \left[ \delta_T(i) \cdot a_{x_T} \right]
\]
\[
q_T^* = \arg \max_{2 \leq i \leq N-1} \left[ \delta_T(i) \cdot a_{x_T} \right]
\]

\[
\log (p^*(X | \Theta)) = \log (\max_{2 \leq i \leq N-1} \left[ \delta_T(i) \cdot a_{x_T} \right])
\]

\[
\log (q_T^*) = \log (\arg \max_{2 \leq i \leq N-1} \left[ \delta_T(i) \cdot a_{x_T} \right])
\]

D. Backtracking

\[
Q^* = \{q_t^* \ldots, q_T^*\} \text{ so that } q_T^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, \ldots, 1
\]

\[
\log (Q^*) = \log (\{q_1^*, \ldots, q_T^*\})
\]
C.

\[
\text{figure;} \ [\text{STbest, bestProb]} = \text{logvit(X1,hmm1)}; \ \text{compseq(X1,ST1,STbest);}
\]

\[
\text{figure;} \ [\text{STbest, bestProb]} = \text{logvit(X2,hmm3)}; \ \text{compseq(X2,ST2,STbest);}
\]
>> figure; [STbest, bestProb] = logvit(X3,hmm5); compseq(X3,ST3,STbest);

>> figure; [STbest, bestProb] = logvit(X4,hmm5); compseq(X4,ST4,STbest);
\[
\begin{align*}
&\text{figure;} \quad \text{[STbest, bestProb] = logvit(X5,hmm6); compseq(X5,ST5,STbest);} \\
&\text{figure;} \quad \text{[STbest, bestProb] = logvit(X6,hmm2); compseq(X6,ST6,STbest);} \\
\end{align*}
\]
>> [val, indx] = max(bestProb')
val =
    1.0e+003 *
    -0.5594   -0.1114   -0.7410   -0.8203   -0.6035   -1.3228
indx =
    1     3     5     4     6     2

Likelihoods along the best path:

| Sequence | logP(X|\Theta_1) | logP(X|\Theta_2) | logP(X|\Theta_3) | logP(X|\Theta_4) | logP(X|\Theta_5) | logP(X|\Theta_6) | Most Likely Model |
|----------|----------------|----------------|----------------|----------------|----------------|----------------|-----------------|
| X_1      | -559.4         | -621.2         | -1439.8        | -1421.1        | -1294.5        | -993.2         | 1               |
| X_2      | -116.0         | -117.5         | -111.4         | -114.5         | -246.2         | -140.2         | 3               |
| X_3      | -826.5         | -787.8         | -1180.2        | -1154.3        | -741.0         | -1329.1        | 5               |
| X_4      | -878.9         | -823.3         | -855.1         | -820.3         | -1158.3        | -1077.9        | 4               |
| X_5      | -776.5         | -760.9         | -811.6         | -788.9         | -997.8         | -603.5         | 6               |
| X_6      | -1396.8        | -1322.8        | -1905.3        | -1852.0        | -2110.5        | -2023.4        | 2               |
D.

```matlab
>> NumModels = 6;
>> for i=1:NumModels
    for j=1:NumModels
        stri = num2str(i);
        strj = num2str(j);
        eval(['[ STbest, bestProb(', stri, ', strj, ',')] = logvit(X', stri, ',hmm', strj, ');']);
    end;
end;
>> diffProb = logProb - bestProb

diffProb =

1.0e+003 *

0.0001  0.0009  0.0000  0.0000  0.0000  0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
0.0002  0.0001  0.1676  0.1626  0.0000  0.0469
0.0000  0.0000  0.0000  0.0000  0.4624  0.0000
0.0003  0.0001  0.0000  0.0000  0.0000  0.0000
0.0001  0.0000  1.2140  1.2068  1.2882  0.0204
```

Difference between log-likelihoods along the best path:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>HMM1</th>
<th>HMM2</th>
<th>HMM3</th>
<th>HMM4</th>
<th>HMM5</th>
<th>HMM6</th>
</tr>
</thead>
<tbody>
<tr>
<td>X₁</td>
<td>0.1</td>
<td>0.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>X₂</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>X₃</td>
<td>0.2</td>
<td>0.1</td>
<td>167.6</td>
<td>162.6</td>
<td>0</td>
<td>46.9</td>
</tr>
<tr>
<td>X₄</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>462.4</td>
<td>0.0</td>
</tr>
<tr>
<td>X₅</td>
<td>0.3</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>X₆</td>
<td>0.1</td>
<td>0.0</td>
<td>1214.0</td>
<td>1206.8</td>
<td>1288.2</td>
<td>20.4</td>
</tr>
</tbody>
</table>

E. The likelihood along the best path is a good approximation of the real likelihood of a sequence given a model because the difference in the probabilities across the entire matrix are minimal.
Problem 5

In the previous homework assignment several algorithms were presented that were estimating parameters of Gaussian pdfs for a given set of training data. Suppose that one has a database containing several utterances of the imaginary word /aiy/, and that you want to train a HMM for this word. Suppose also that this database comes with a labeling of the data, i.e., some data structures that tell you were are the phoneme boundaries for each instance of the word.

A. I would use a left-right model for this situation. No sounds are repeated so there is no need for a loop-back in the model. I would use at least three states, for word /aiy/, one for each phone. Two additional states including the initial and final states may be added.

B. I would take the parameters from the data since it is all labeled already (probability of each phoneme).

C. I would run all of the data and keep adjusting the values in my HMM until the change with each iteration was smaller than a certain value using the Viterbi and forward algorithms as a learning algorithm.