Baum-Welch training of the continuous-density hidden Markov model

Abstract The aim of this report is to implement the Baum –Welch training of the continuous-density hidden Markov model parameters. The computational process will proceed through three main steps, namely the estimation of the continuous output probability densities, the evaluation of the observation sequence probability and the training of the model parameters by adjusting the state transition matrix, the means and variances values. Moreover, the report will contain the state topology and will illustrate the probability density functions for each state of the model before and after the first iteration.

INTRODUCTION

The Hidden Markov Model describes the joint probability of a set of "hidden" and observed variables. It has some assumptions, such as the current observation independence from the previous observations and states, the model current state dependence only upon the previous state and the time independence of the state transitions [1, 2]. Furthermore, each state i of the model is characterised by a state transition probability distribution $\boldsymbol{A = \{ a_{ij} \}}$ (for 1 <= i, j <= N), a continuous Gaussian probability distribution $\boldsymbol{B = \{b_i(O_t) \} = \mathcal{N}(O_t, \mu_i, \Sigma_i)}$ and an initial state distribution $\boldsymbol{\pi = \{ \pi_i \}}$ (for 1<= i <=N).

The Markov Model is called "hidden" because, given the model $\boldsymbol{\lambda = (A, B, \pi)}$, and a sequence of observations $\boldsymbol{O = \{ O_t \}}$ (for 1<= t <=T), it cannot be exactly estimated which state sequence produced these observations. Nevertheless, by applying the Baum-Welch algorithm, it is possible to calculate the probability of the observation sequence $P(O|\lambda)$, which is used to estimate the occupation likelihoods. The latter performs a soft state assignment to the given observation sequence and determines the re-estimated means and variances values.

The Baum-Welch algorithm is used to find the unknown parameters of a hidden Markov model. Namely, it trains the model by re-estimating the state-transition and the output probabilities of the new models $\boldsymbol{\lambda^{\wedge} = (A^{\wedge}, B^{\wedge}, \pi^{\wedge})}$ through a forward and backward likelihoods computation.

The Baum-Welch training will require to perform three main steps for each iteration, such as:

1. The illustration of the probability density functions b_i for each state i and the computation of the output probability densities $b_i(O_t)$ for each state and each time frame.
2. The implementation of the forward-backward algorithm in order to evaluate the observation sequence probability $P(O|\lambda)$.
3. The re-estimation of model's parameters by computing the occupation and the transition likelihoods.

PDFs ESTIMATION AND REPRESENTATION

This report will consider a Markov model in which the observation is a probabilistic function of the state. Unlike the discrete HMM which holds discrete output probabilities, the continuous HMM's output pdfs can be represented as Gaussian distributions with mean μ_i and variance Σ_i :

$$\mathbf{b}_i(\mathbf{Ot}) = \frac{\mathbf{1}}{\sqrt{\mathbf{2\pi\Sigma(i)}}} \boldsymbol{exp}^{\frac{-(\boldsymbol{O}-\mu(\boldsymbol{i}))^2}{\mathbf{2\Sigma(i)}}}$$

The above equation generates a [N x T] matrix, which holds the observations' probability values of all given observation O_t at each state i (see Fig.2). The graphical representation of the output pdfs follows:

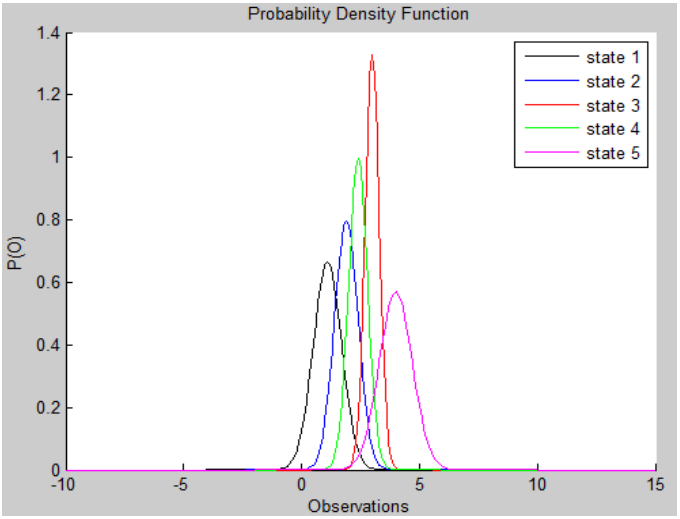


Fig. 1 The pdfs b_i for each of the five states have a Gaussian distribution determined by the above equation with μ = [1.1, 1.9, 3, 2.4, 4] and Σ = [0.36, 0.25, 0.09, 0.16, 0.49], whereas the O vector describes the axes of observations [-10 : 0.1 : 15].

Fig. 2 The output probability densities $b_i(Ot)$ for i = 1 .. 5 and t = 1 .. 8

0.5324	0.6290	0.0900	0.0074	0.0015	0.0636	7.0781e-08	1.0126e-08
0.0448	0.3884	0.5794	0.1080	0.0272	0.4839	1.0722e-0.6	1.2365e-07
2.2927e-13	1.4156e-07	0.0874	1.2579	1.0648	0.1800	4.9557e-06	1.4156e-07
1.1930e-04	0.0227	0.9667	0.4566	0.1350	0.9974	1.0321e-06	6.5981e-08
8.5080e-06	3.3511e-04	0.0299	0.1658	0.2966	0.0418	0.4416	0.3457

EVALUATION OF P (O | λ)

Given a single observation sequence O = [0.7, 1.3, 2.3, 2.9, 3.2, 2.4, 4.5, 4.7] of T=8 time frames and a model λ = (A, μ, Σ, π) with N = 5 number of states, the main purpose is to efficiently compute the probability of the observation sequence P(O | λ) produced by the model λ. The transitions between states are characterized by the state transition matrix A = { π_j, a_{ij}, η_i }, which can be described by the state topology drawn below:

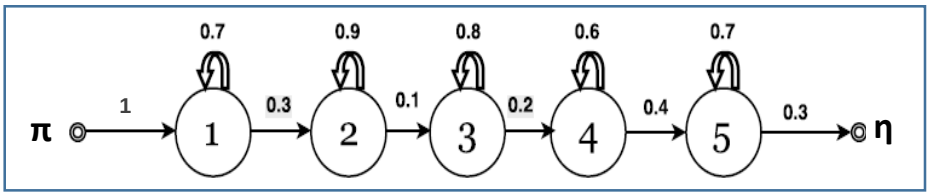
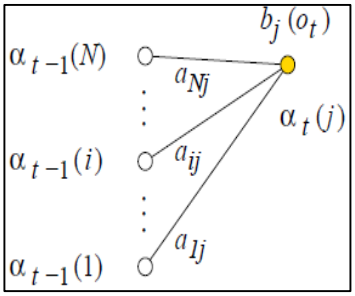


Fig. 3 The state topology of the initial HMM

The P(O | λ) value determines how well the initial model matches the given observation sequence. In order to compute the total probability of the observations, it is necessary to marginalize the state sequence, which is unknown, by summing over all possible states of all joint probabilities. This process could require a considerable amount of computation, unless a forward-backward procedure is implemented.

The forward procedure generates an $\alpha_t(i)$ matrix of size [TxN]. It estimates the probability of the partial observation sequence $b_j(O_t)$ until time *T* and the state a_{1j} ..a_{Nj} at time *t-1*.

The forward likelihood is calculated inductively in three steps:



1) Initialization: $\boldsymbol{\alpha_1(i) = \pi_i\, b_i(O_1)}$ *for* $1 \le i \le N$

2) Induction for t = [2 : T]: $\boldsymbol{\alpha_t(j) = [\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}]\, b_j(Ot)}$ *for* $1 \le i,j \le N$

3) Termination: $\boldsymbol{P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \eta_i}$ *for* $1 \le i \le N$

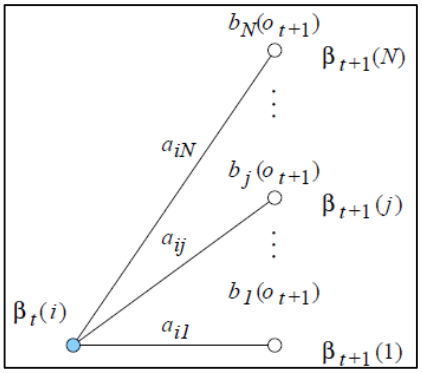
0.6516	0	0	0	0
0.2531	0.0568	0	0	0
0.0065	0.0440	8.1266e-04	0	0
7,4645e-06	0.0109	7.4854e-04	7.3353e-05	0
1.2053e-09	0.0020	2.4962e-04	1.1529e-05	2.4221e-06
1.9874e-11	6.0611e-04	5.7647e-05	7.5180e-05	5.4280e-07

2.6628e-20	1.6412e-05	1.3632e-05	3.2950e-15	2.6448e-06
3.7250e-30	2.9345e-07	1.5345e-06	1.1604e-18	2.2847e-07

Fig. 4 The trellis fragment depiction of the forward likelihood and the [TxN] matrix containing the forward likelihoods. Source of the trellis fragment: [4].

The backward procedure, instead, generates a $\beta_t(i)$ matrix of size [TxN]. It estimates the probability of the partial observation sequence $b_j(O_t)$ from the end to *t+1* and the state a_{1j} ..a_{Nj} at time *t+1*.

Also the backward likelihood is calculated inductively in three steps:



1) Initialization: $\boldsymbol{\beta_T(i) = \eta_i}$ *for* $1 \leq i, j \leq N$

2) Induction for t = [T-1 : 1]: $\boldsymbol{\beta_t(j) = \sum_{j=1}^{N} \beta_{t+1}(j)\, a_{ij}\ b_j(O_{t+1})}$

3) Termination: $\boldsymbol{P(O \mid \lambda) = \sum_{i=1}^{N} \beta_1(i) \pi_i\, b_i(O_1)}$

1.0519e-07	1.7741e-07	2.9703e-07	4.6180e-09	5.8916e-09
1.5035e-07	5.3649e-07	3.0695e-06	6.6856e-06	8.7964e-08
2.7525e-07	1.3790e-06	7.4540e-06	9.5455e-06	1.1519e-06
1.0655e-16	3.5069e-06	3.7336e-05	3.3549e-05	1.4262e-05
0	1.7410e-15	2.3688e-04	7.7975e-04	1.7277e-04
0	0	1.2061e-13	8.9550e-04	0.0022
0	0	0	0.0104	0.0259
0	0	0	0	0.3000

Fig. 5 The trellis fragment depiction of the backward likelihood and the [TxN] matrix containing the backward likelihoods $\beta_t(i)$. Source of the trellis fragment: [4]

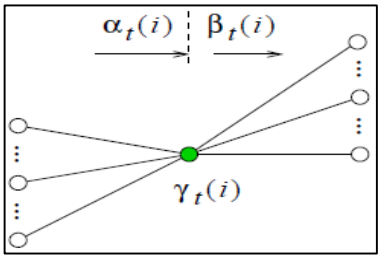
The forward and backward procedures are an equivalent way of computing P(O | λ) recursively, therefore in both cases the estimated value of the total probability of the observations was found to be $\mathbf{2.5439 * 10^{-5}}$.

RE-ESTIMATION OF MODEL'S PARAMETERS

The third, and by far the most important task, is the adjustment of the model's parameters so that the probability of the observation sequence given the model λ results locally maximised. As the state sequence is hidden, it is necessary to implement the Baum-Welch iterative procedure based on soft assignment of observations to states via the occupation and transition likelihoods estimation.

The occupation likelihood $\gamma_t(i)$ defines the probability of being in state *i* at time *t*. It can be expressed in terms of forward-backward variables:

$$\boldsymbol{\gamma_t(i) = \frac{\alpha_t(i)\, \beta_t(i)}{P(O|\lambda)}}$$



1	0	0	0	0
0.5552	0.4448	0	0	0
0.0260	0.8856	0.0884	0	0
1.1604e-14	0.5563	0.4078	0.0359	0
0	5.0719e-11	0.8627	0.1312	0.0061
0	0	1.0144e-10	0.9823	0.0177
0	0	0	4.9834e-10	1
0	0	0	0	1

Fig.6 The trellis fragment depiction of the occupation likelihood and the [TxN] occupation matrix $\gamma_t(i)$. Source of the trellis fragment: [3]

Once the occupation likelihood matrix is obtained, it is possible to estimate the new means and variances values for each state. In fact the re-estimated means $\mu_i^{\wedge}$ are defined by the joint probability of the observation sequence and the occupation likelihood at corresponding time frames, whereas the re-estimated variances $\Sigma_i^{\wedge}$ are defined by the joint probability between the occupation likelihood and the difference between the observation sequence and the new means.

$$\hat{\mu_i} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T}\gamma t(i)\, O_t}{\sum_{i=1}^{N}\sum_{t=1}^{T}\gamma t(i)} \qquad \hat{\Sigma_i} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T}\gamma t(i)(O_t-\mu_i)(O_t-\mu_i)'}{\sum_{i=1}^{N}\sum_{t=1}^{T}\gamma t(i)}$$

Re-estimated values:

$\mu_i^{\wedge}$ = [0.91, 2.02, 3.03, 2.46, 4.59] $\Sigma_i^{\wedge}$ = [0.33, 1.24, 7.23, 0.09, 21.32]

The adjusted means and variances values determine a different distribution of the output probability densities $b_i(O_t)$. In fact, it is possible to spot the difference if the new pdfs are plotted on the old pdfs:

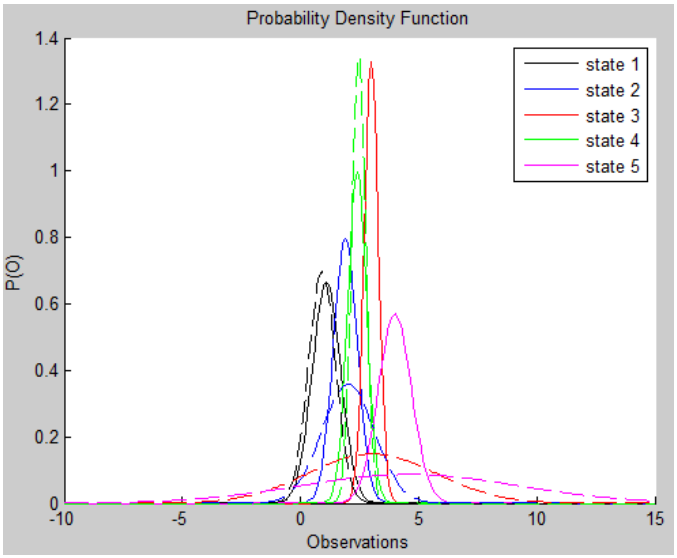


Fig. 7 The re-estimated pdfs b_i for each of the five states.

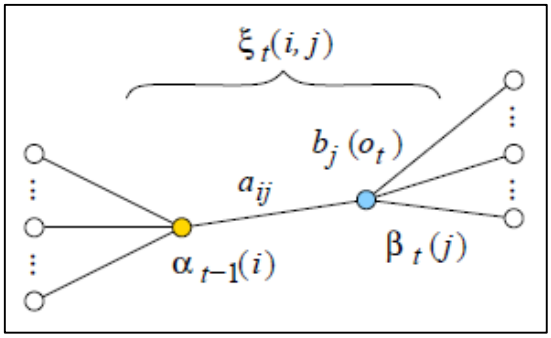


Fig.8 The trellis fragment depiction of the transition likelihood. Source: [3]

By evaluating the occupation likelihood $\gamma_t(i)$, it is possible to adjust only the output probabilities B = { $\mu_{i,}$, Σ_i}. The re-estimation of the state-transition matrix A = { a_{ij}}, instead, will require the computation of the transition likelihood $\xi_t(i, j)$, which defines the probability to be in state *i* at time *t-1* and in state *j* at time *t* (see App. Fig. 1). In fig. 7 is depicted the relationship between the terms in the below equations:

$$\xi_t(\mathbf{i},\mathbf{j}) = \frac{\alpha_{t-1}(\mathbf{i})a_{ij}b_j(O_t)\beta_t(\mathbf{j})}{\mathbf{P(O|\lambda)}} \qquad \hat{a_{ij}} = \frac{\sum_{t=2}^{T}\xi_t(\mathbf{i},\mathbf{j})}{\sum_{t=1}^{T}\gamma t(\mathbf{i})}$$

Now that both the state transition probability distribution and the output probability distribution are re-estimated, the first iteration of the Baum-Welch can be considered concluded. In fact, a new HMM, which locally maximises the probability of the observation sequence, was obtained.

CONCLUSION

The Baum-Welch training procedure is an iterative re-estimation procedure which leads the resulting probabilities to converge to satisfactory estimates. The final result of the re-estimation is called the maximum likelihood estimate of the HMM and it complies the following statement: $\mathbf{P(O|\lambda^{\wedge}) \geq P(O|\lambda)}$ [1,3]. Therefore, it is proved that, by using the $\lambda^{\wedge}$ model during each reiteration, the probability of O increases. Though the algorithm does not guarantee a global maximum because the forward-backward procedures offer only local maxima [1].

Thus, by analysing figure 7, it can be seen that the means of the new pdfs have changed slightly while most of the pdfs increased their spread, with a drastic change recorded for state 3 and state 5. In fact, the new estimates produced a model with a much smaller observation sequence probability ($0.7 * 10^{-8}$), contrary to what was stated before. This discrepancy may be caused by a bad initialisation of Markov Model or by the observation sequence vector's small size. In fact, if the training is iterated five times, the vector containing the observations sequence probabilities shows that the statement $P(O|\lambda^{\wedge}) \geq P(O|\lambda)$ is complied from the second iteration onward (P = [25.44 0.07 0.09 0.11 0.11]$*10^{-6}$).

In conclusion, the following report has implemented in practice the theory of hidden Markov Model in order to compute how well the initial model predicts the given observation sequence and to train the model's parameters, so that they could converge to satisfactory values of local maxima.

REFERENCES

[1]. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of IEEE, 1989.

[2]. P. Blunsom. Hidden Markov Models. Review, 2004.

[3]. P.Jackson, *Centre for Vision Speech & Signal Processing*, HMM part 3, Guildford: University of Surrey, 2015

[4]. P.Jackson, *Centre for Vision Speech & Signal Processing*, HMM part 4, Guildford: University of Surrey, 2015

APPENDICES

Figure 1 The transition likelihood $\xi_t(i, j)$ 3D matrix, with layers corresponding to 1:T time frames.

Time frame 1 -> val(:,:,1) =

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Time frame 2 -> val(:,:,2) =

0.5303	0.2273	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Time frame 3 -> val(:,:,3) =

0.0051	0.0022	0	0	0
0	0.4169	0.0463	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Time frame 4 -> val(:,:,4) =

0.0000	0.0000	0	0	0
0	0.0760	0.0084	0	0
0	0	0.0514	0.0128	0
0	0	0	0	0
0	0	0	0	0

Time frame 5 -> val(:,:,5) =

0	0	0	0	0
0	0.0000	0.0000	0	0
0	0	0.8342	0.2086	0
0	0	0	0.0013	0.0008
0	0	0	0	0

Time frame 6 -> val(:,:,6) =

0	0	0	0	0
0	0	0	0	0
0	0	0.0000	0.0000	0
0	0	0	0.0805	0.0537
0	0	0	0	0.0002

Time frame 7 -> val(:,:,7) =

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0.0000	0.0000
0	0	0	0	0.0041

Time frame 8 -> val(:,:,8) =

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	1.0000

MATLAB CODE:

```
close all
clear all

% A -> the state transition probability matrix

A = [0   1   0   0   0   0   0;
     0 0.7 0.3   0   0   0   0;
     0   0 0.9 0.1   0   0   0;
     0   0   0 0.8 0.2   0   0;
     0   0   0   0 0.6 0.4   0;
     0   0   0   0   0 0.7 0.3;
     0   0   0   0   0   0   0];


% Vectors of output probability densities

mean     = [1.1  1.9  3   2.4  4];
variance = [0.36 0.25 0.09 0.16 0.49];

% The observation sequence for training

O = [0.7  1.3  2.3  2.9  3.2  2.4  4.5  4.7 ];  % for t=1:T where T=8


% 1 % The output pdf for each state i is Gaussian

A1          = A(2:end-1,2:end-1); % a_ij
entries     = A(1,2:end-1);       % π_i
exits       = A(2:end-1,end);     % η_i



n = length(A1(1,:));
T = length(O);

C = {'k','b','r','g','m'}; % Colours of pdfs

x = -10:0.1:15;

Pf=0;
q = 1;
```

```
while (q <= 1)      % The loop could be extended to various iterations

% 1 % Plot of each state pdf

for t = 1:5;

    p1 = (-.5/variance(t)) * (x - mean(t)).^2 ;
    p2 = sqrt(2*pi*variance(t));
    f  = exp(p1)./p2;

    figure(q);
    hold on
    plot(x, f, 'color', C{t})
    title('Probability Density Function')
    xlabel ('Observations')
    ylabel('P(O)')
    legend({'state 1';'state 2';'state 3';'state 4';'state 5'})


end

% 2 % The output probability densities b_i(ot) for i=1..5 and t=1..T

for j = 1:T
    for i = 1:n

        p_1       = (-.5/variance(i)) * (O(j) - mean(i)).^2 ;
        p_2       = sqrt(2*pi*variance(i));
        b_i(i,j)  = exp(p_1)./p_2;

    end
end


% 3 % The forward likelihoods alfa_t(i) and overall likelihood of the
% observations P(O|λ).

% Initialization
for i = 1:n

    alfa(1,i) = b_i(i,1)*entries(i);

end


% Recursion
for t = 2:T
    for j = 1:n
        z = 0;
        for i = 1:n

            z = z+A1(i,j)*alfa(t-1,i);

        end
        alfa(t,j) = z*b_i(j,t);
    end
end

% Termination
```

```
for i = 1:n

    Pf = alfa(T,i)*exits(i)';

end


% 4 % The backward likelihoods beta_t(i)

% Initialization
beta(T,:) = exits';

% Recursion
for t = (T-1):-1:1
    for j = 1:n
        z = 0;
        for i = 1:n

           z = z+(A1(j,i)*beta(t+1,i)*b_i(i,t+1));

        end
        beta(t,j) = z;
    end
end

% Termination -> allows to check if the recursion gives the same result
% respect to forward likelihood

for i = 1:n

    Pb(1,i) = b_i(i,1)'*beta(1,i)*entries(1,i);

end


% 5 % The occupation likelihoods and the transition likelihoods

% Occupation Likelihoods

gamma   = alfa.*beta;
gamma   = gamma./Pf;

entries = gamma(1,:);


% Transition Likelihoods

for t = 2:T
    for j = 1:n
        for i = 1:n

            xi(i,j,t) = (alfa(t-1,i)*A1(i,j)*b_i(i,t)*beta(t,i))./Pf;

        end
    end
end

% 6 % Re-estimation of the means and variances
```

```
% Means
m = zeros(1,5);

for i = 1:n
    for t = 1:T

        m(i) = m(i) + gamma(t,i)*O(t);
    end
end


% Variances

v = zeros(1,5);

for i = 1:n
    for t = 1:T

        v(i) = v(i) + gamma(t,i)*(O(t)-m(i))*(O(t)-m(i))';
    end
end

% Output accumulators


bi = zeros(1,5);


for i = 1:n
    for t = 1:T

        bi(i) = bi(i) + gamma(t,i);
    end
end


ai = zeros(5,5);

for j = 1:n
    for i = 1:n
        for t = 2:T

            ai(i,j) = ai(i,j) + xi(i,j,t);

        end
    end
end


% Re-estimated values

mean = m./bi;

variance = v./bi;

[rows,columns] = size(ai);
denom          = repmat(bi', [1, columns]);
```

```
aij_es          = ai./denom;


% Normalisation of the re-estimated transition matrix "aij_es"

S       = sum(aij_es,2);
S       = repmat(S, [1,5]);
A1      = aij_es./S;


% 7 % Plot of each state pdf after the parameters re-estimation

for t= 1:5;

    p11 = (-.5/variance(t)) * (x - mean(t)).^2;
    p22 = sqrt(2*pi*variance(t));
    f1   = exp(p11) ./ p22;

    figure(q);
    hold on
    plot(x, f1, '--', 'color', C{t})

end

 P(q)=Pf;           % Stores the likelihood after each iteration
 q = q+1;

 end                % End of the while loop

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```