```
% Code For Problem 2

clear;
load 'q1_data.mat'

NUM_FOLDS = 4;

Z = [Z1;Z2];  % 2 Dimensional Representation Of Spikes

REDUCED_DIMENSION = size(Z,1);
RANGE_OF_GAUSSIANS = 8;

% Splitting the data into folds
Z_folded = mat2cell(Z, REDUCED_DIMENSION, ...
    repmat(NUM_DATA/NUM_FOLDS, 1, NUM_FOLDS));


% The Cross Validated Likelihood as a function of number of gaussians
likelihood = zeros(1, RANGE_OF_GAUSSIANS);

% Calculating Cross Validate Likelihood for each particular value of K
for k=1:RANGE_OF_GAUSSIANS
    K = k;s
    % Initializing Parameters
    params.mu = InitParams.mu(:,1:K);
    params.sigma = repmat(InitParams.Sigma, [1,1,K]);
    params.pi = repmat(1/K,1,K);

    % Initilizing likelihood computed by taking each fold as train data and
    % rest as test data
    fold_likelihood = zeros(1, NUM_FOLDS);

    % Computing likelihood for each fold by taking each fold as
    % train data and rest as test data
    for i=1:NUM_FOLDS

        %Separating Test Data and Train Data
        train_folds_nums = 1:NUM_FOLDS;
        test_fold_num = i;
        train_folds_nums(i) = [];
        train_data = [];
        for j=train_folds_nums
            train_data = [train_data Z_folded{j}];
        end
        test_data = Z_folded{test_fold_num};

        % Fitting the model on the test data
        [mu, sigma, ppi] = func_GMM(params, train_data);

        % Computing the likelihood on the test data
        this_params = params;
        this_params.mu = mu;
        this_params.pi = ppi;
        covvar = sigma;
        for j=1:size(test_data, 2)
            fold_likelihood(i) = fold_likelihood(i) + ...
                    boxed_term(test_data(:,j), this_params, covvar);
        end

%         This part is for computing likelihood on all data
%         for j=1:size(Z, 2)
%             fold_likelihood(i) = fold_likelihood(i) + ...
%                     boxed_term(Z(:,j), this_params, covvar);
%         end

    end

    likelihood(k) = -sum(fold_likelihood);
```

```
end

% Plotting Likelohood for Problem 2a
plot(likelihood, 'LineWidth',1.25);
xlabel('Number Of Gaussians');
ylabel('Likelihood');
title('Likelihood v/s Number Of Gaussians');


% Problem 2b. For plotting the ellipse arround the cluster centers
figure
for k=1:RANGE_OF_GAUSSIANS
    K = k;
    subplot(4,2,k);

    %Plot all the data points
    plot(Z1, Z2, '.');
    hold on

    % Initialization and Isolate training data
    params.mu = InitParams.mu(:,1:K);
    params.sigma = repmat(InitParams.Sigma, [1,1,K]);
    params.pi = repmat(1/K,1,K);

    train_data = [];
    for i=2:NUM_FOLDS
        train_data = [train_data  Z_folded{i}];
    end

    % Estimating the model
    [mu, sigma, ~] = func_GMM(params, train_data);

    % Plotting the cluster center and their spread
    for i=1:K
        this_point = mu(:,i);
        plot(this_point(1), this_point(2),'k.');
        hold on
        func_plotEllipse(this_point, sigma(:,:,i));
        hold on
    end
    xlabel('X(1)');
    ylabel('X(2)');
    title_str = sprintf('Clustering for %d Gaussians', K);
    title(title_str);
end

% Problem 2c. PLotting Cannonical Waveforms corresponding to 3 cluster
% centers
figure;
K = 3;
recovered_spikes = zeros(DIMENSION, K);
params.mu = InitParams.mu(:,1:K);
params.sigma = repmat(InitParams.Sigma, [1,1,K]);
params.pi = repmat(1/K,1,K);
[mu, sigma, ppi] = func_GMM(params, Z_folded{1});

% Use formula Xn = U_m * Z_n + mean_spike, for each of the clusters
U_m = fliplr(U(:,end-1:end));
for i =1:K
    recovered_spikes(:, i) = U_m * mu(:,i) + mean_spike;
end
plot(recovered_spikes(:, 1), 'r', 'LineWidth',1.25); hold on
plot(recovered_spikes(:, 2), 'g', 'LineWidth',1.25); hold on
plot(recovered_spikes(:, 3), 'b', 'LineWidth',1.25); hold on
xlabel('Time');
ylabel('Voltage');
legend('Cluster Center 1', 'Cluster Center 2', 'Cluster Center 3')
title('High D vectors from 2-D space');
```