1. Given a d-dimensional mean vector \( v \) and d\(\times d \) covariance matrix \( C \) (symmetric, pos def), generate \( N \) random sample points distributed according to a Gaussian with mean \( v \) and covariance \( C \). Recall that we discussed in class how to do this by first using \texttt{randn} in \texttt{matlab} to generate variables \( x \) with zero mean and covariance \( I \), and then doing a linear transformation \( y=Ax + b \) to transform them into new variables \( y \) with the desired mean and covariance.

2. For a set of 2D points generated as above, make a 2D plot of them in a graphics display, and then draw overlaid on that an elliptical contour associated with a Gaussian of mean \( v \) and covariance \( C \) to verify that your transformed points form a cluster with the correct center location and elliptical spread. We discussed doing this by first generating points on a 2D circle with radius 1.5 and then doing a linear transformation \( Ax+b \) to transform those points into new points lying on an elliptical boundary.

Note: as a sanity check, try all three cases we talked about in class. That is, try to generate and draw data/contours from: 1) circular Gaussian, 2) elliptical Gaussian with major axis lying along either the x or y coordinate axis, and 3) rotated elliptical Gaussian (major axis is oriented diagonally).

3. Given a set of \( N \) sample points from a Gaussian distribution with d-dimensional mean vector \( v \) and d\(\times d \) covariance matrix \( C \), compute the maximum likelihood estimates for \( v \) and \( C \) using the formulas from our class notes. If you generate the sample points using the routine in part 1 and some given mean \( v_0 \) and covariance \( C_0 \), and you then estimate the sample mean \( v \) and \( C \) using the maximum likelihood routine, you might expect \( v \) to be close to \( v_0 \) (distance-wise), and \( C \) to be similar to \( C_0 \) (similar axes and eigenvalues).

4. Given the specification of a mixture of Gaussians distribution with \( K \) Gaussian components, generate \( N \) sample points from that distribution. The values that are given to you are the mixing weights \( p_1, p_2, \ldots, p_K \), the mean vector of each component \( v_1, v_2, \ldots, v_K \), and the covariance matrix of each component \( C_1, C_2, \ldots, C_K \). Recall from our class the generative process for a mixture of Gaussians, which can easily be turned into an algorithm: for \( n=1 \text{ to } N \), generate sample point \( x_n \) by first choosing which of the \( K \) components it should come from (with the help of a uniform random number generator), and then using the routine from part 1 to generate a sample point from that component.

5. Given a set of \( N \) sample points from a mixture of Gaussians distribution with a known number of components \( K \), use the EM algorithm to estimate the values of the parameters of the mixture of Gaussians (that is, the \( K \) mixing weights, the \( K \) mean vectors, and the \( K \) covariance matrices). If you run this for 2-dimensional data points, you should be able to visualize the results of this step by plotting the data points overlaid with covariance ellipses for each Gaussian component. If you generate a sample set from part 4, and then estimate the parameters using EM, you should get back parameters that are “close” to the true parameters used to generate the data.