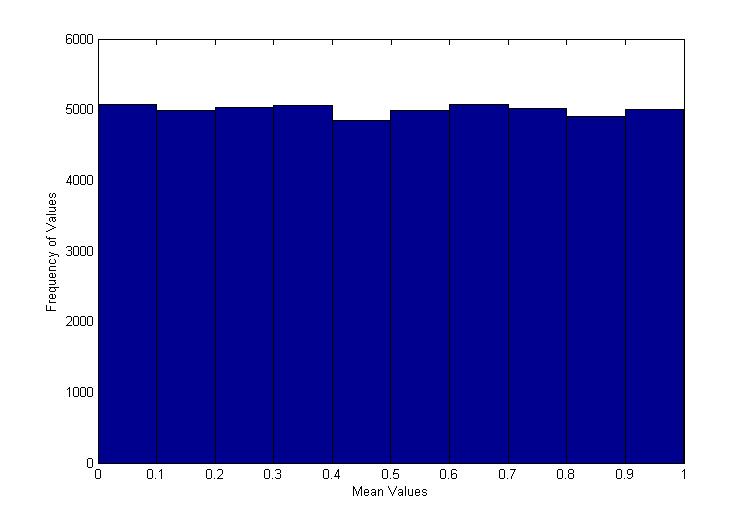
Assignment 1

First problem:

The function used is rand() and generated 50,000 random numbers in the range of 0-1, uniformly distributed between 10 bins.

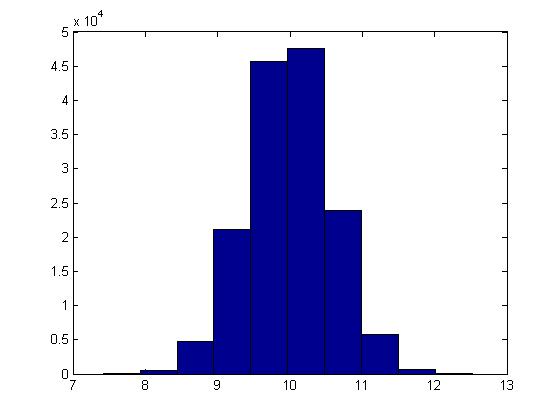


The code is as follows:

r = rand(1,50000);

hist(r)

Second Problem:



My histogram differs from a normal distribution curve in a variety of ways. For one thing, the distributions are not perfectly symmetrical around the mean. This can be attributed to normal distributions having equal mean, median and mode whereas in my histogram, the mean and median are 9.99 but the mode is 7.59. Another key feature of normal distribution is 68% of the area is within one standard deviation and 95% is within two standard deviation and considering that the standard deviation for this histogram is 0.57, it proved to be true.

Problem Three:

1. The dot product geometrically means the multiplication of the magnitudes and cosine angle. For example, if the problem is the dot product between A and B, then geometrically it is:
   1. (magnitude of A)(magnitude of B)(cosine angle between the two)
2. See attached ZIP file
3. The mean of the dot product should be zero although after many trials, the mean turned out to be -0.004 or 0.004 with a +/- 0.001 margin of error. The mean should be zero because normalized vectors have a mean of zero which means they are orthogonal. The dot product of vectors that are orthogonal also equal zero meaning the mean must equal zero as well.

E) The standard deviation would decrease the more bins are introduced because standard deviation is the statistic of how closely related numbers are to the mean of the normal distribution. The more bins that are measured, the lower the difference is between the bins that are centered around the mean. However, the standard deviation decreases at a much slower rate than the mean. If there are 10 bins, the SD is 0.1 but it is not until 1,000,000 bins is there a chance that it drops down to 0.09. The mean on the other hand is 0.0032 with 10 bins but 0.0023 with 1,000,000 bins.

Problem Four:

The results generates a value similar to pi, with errors that range in the hundredth to millionth value. The process of generating this value is generating random uniformly distributed values from -1 to 1 for each coordinate in (x,y) value. A circle is placed within a square with two unit sides, ranging from -1 to 1. Once the coordinate is generated, the program determines if it is within the circle that is centered at zero with an area of pi and radius of one. Finally the program compares the number of times the coordinates land within the circle compared to the total number of points and multiplies that value by four to create a value close to pi. The program runs a million coordinates to reduce the chance of error.