1) Construct random N-dimensional vectors $x^\mu$ for $\mu = 1, \ldots, P$ by choosing their components independently from a Gaussian distribution with zero mean and unit variance. Randomly assign $q^\mu = \pm 1$ for $\mu = 1, \ldots, P$ with equal probability for the two cases. Now build perceptrons to classify these data based on the a) Hebbian, b) Fisher linear discriminant, c) pseudo-inverse, d) maximum margin and e) learning method outlined in class. To implement the maximum margin classifier you can use the Matlab function $[w \kappa] = \text{MarginMax}(x, q)$ with accompanying functions in a folder that is posted on the course web site.

A) Plot performance of each method as a function of $P/N$ (for some reasonable $N$).

B) Examine the sensitivity of each classifier to noise by training on a set $x^\mu$, then classifying the noisy observations $(x^\mu + \varepsilon\eta)/\sqrt{1 + \varepsilon^2}$, where $\eta$ is generated independently from the same distribution you used for $x^\mu$. Repeat this for several trials (instantiations of $\eta$.) Compare the ability of the different methods to classify correctly despite this noise, and determine the range of $\varepsilon$ over which each can do this. Hypothesize based on your observations why some classifiers perform better than others.

2) For N-dimensional vectors generated as in problem #1, determine the probability distribution for their lengths, that is compute $p[|x| = r]$. Show that this distribution implies that all the points generated by a high-dimensional spherical Gaussian lie near the surface of a sphere.