1. *RBM:* This problem will consider the training of a simple RBM on a sample from the MINST. Specifically, consider the same test data set as Homework 3 (0's and 9's), but the data are converted to binary (1's and 0's): `minst0bw_train.dat` and `minst9bw_train.dat`.

   The following pseudo-code algorithm can be used to fit the RBM (note, this version considers mini-batches and only uses $k = 1$ Gibbs iterate, as is the standard for RBM training):

```
numV = 100   %number of minibatch training samples
m=784  %number of visible units
n=20   %number of hidden units
eta = 0.01 %learning rate
nepoch = 1000  %number of minibatch samples

load minst_train.dat   % note, this is the combined data

%Initial value for weights
w <-  N(0.1,1)
b <- 0
c <- -

err=zeros(nepoch,1);   %variable to save SSE for each epoch

for ne = 1:nepoch   %loop over the number of epochs

    trainbw <- sample of size numV from the training set "minst_train.dat"
    for iv = 1:numV    %loop over the number of images in the epoch
         v <- assign image from epoch training sample

         %sample initial hidden units; first calculate sigmoid probabilities (h)
         %         then sample (hs)

         h = 1./(1 + exp(-(w*v + c)))
         hs <- (0,1) depending on probability (usual discrete Bernoulli sampling)

         %calculate probabilities for "reconstructed" model visible units (note, we
         %       typically don't sample these, but use the probs

         vr= 1./(1 + exp(-(w'*hs + b))) %input reconstruction

         %calculate probabilities for model hidden unit "reconstruction"

         hr= 1./(1 + exp(-(w*vr + c))) %hidden reconstruction
```

```
        %Contrastive divergence

        dw = eta*(h*v' - hr*vr')
        w= w + dw

        db = eta*(v - vr)
        b = b + db

        dc = eta*(h - hr)
        c = c+ dc;

       err(ne) = err(ne) + sum((v-vr).^2);  %calculate running SSE for each epoch

    end
end
```

(a) Explore different choices for number of hidden units, learning rate, minibatch sample, number of epochs, etc. and report your results. What is your final model?

(b) With your final model, plot your error rate by epoch and discuss.

(c) With your final model, make a plot of your average weights (as images for each hidden unit).

(d) With your final model, sample 20 images at random and then "flip" 15% of the pixels (e.g., if they are 1's, change to 0's and vice versa). Use your fitted parameters to predict the "true" images associated with these flipped images. Also, report the true images.