Homework 5

1. Why is the simple Euclidean distance on noisy data a biased estimate of the true distance between two activity patterns?

A squared Euclidean distance is the inner product of the two difference vectors between the estimated activity patterns. As the vectors are noise, the expected value of a squared Euclidean distance is the true distance plus the expected value of the sums of squares of the random noise.

2. Do pair-wise classification between activity patterns using the provided pre-whitened estimates. That is, you need to do 10 total classifications, each finger against each finger. Report the average classification accuracies for each pair. Visualize the dissimilarity measure (% accuracy) in a 5x5 Representational Dissimilarity Matrix (RDM).

The pairwise average classification accuracies for each pair are:
0.9271 0.9896 0.9896 1.0000 0.8958 0.9635 0.9792 0.7448 0.9271 0.7708

The RDM is given in Figure 1a.

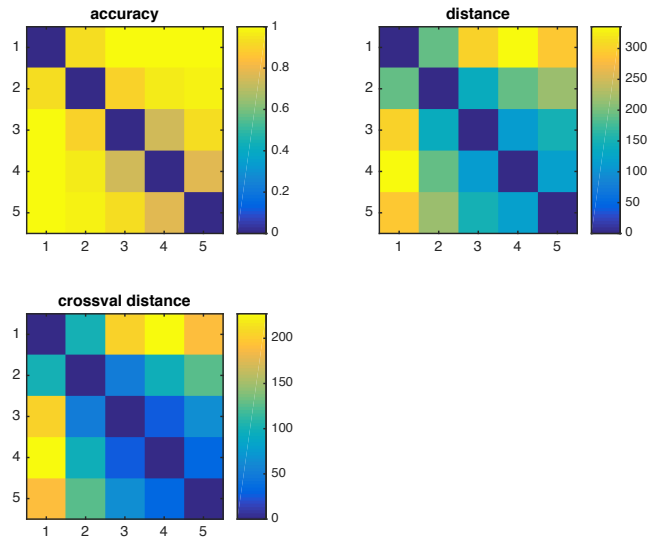

Figure 1. Representational dissimilarity matrices for (A) classification accuracy (B) Mahalanobis distance (i.e. Euclidean distance on prewhitened data) and (C) crossvalidated distance.

3. Average the activity pattern for each finger across runs. Calculate the 10 pairwise squared Euclidean distances between the mean activity patterns. Again, show your results using an RDM.
MATLAB Tip: The functions pdist and squareform can make your life much easier.
In a scatterplot, plot the squared Euclidean distances against % accuracy. Use different colours / symbols for the different subjects. What do you observe?

The 10 pairwise distances are
193.1167 303.6884 335.2598 290.2224 138.8106 192.4017 218.0015 110.7478 151.4682 119.6969

Distances are reported in Figure 1b.

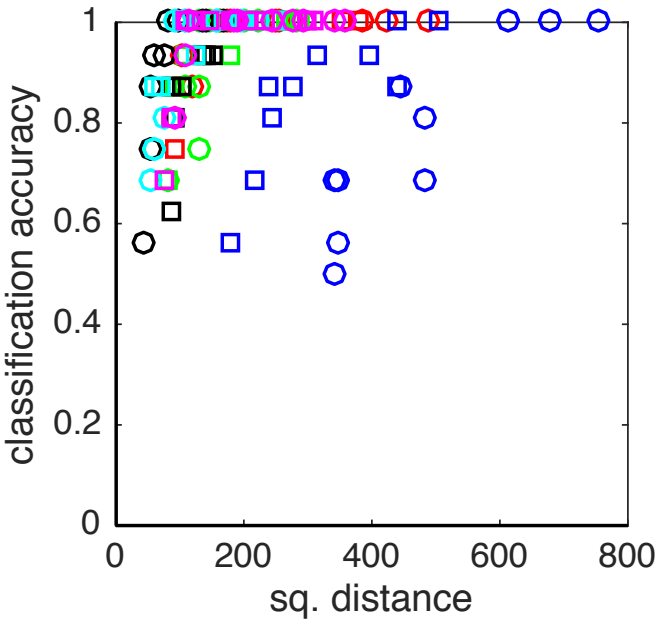

Figure 2. Squared Euclidean distances (x-axis) plotted against classification accuracy (y-axis).

We observe only a moderate relationship between distances between distances and accuracies. Classification accuracies saturate at 1 (100%). Also there are some subjects (blue) with very large distances (>200), but many of the classification accuracies are not very high.

4. Calculate the cross-validated squared Euclidean distance between each of the 10 finger pairs. Report your results in form of a RDM matrix. In a scatterplot, plot the cross validated squared Euclidean distances against the squared Euclidean distance. Use different colours / symbols for the different subjects. What do you observe?

The code for calculating the cv distances is given by:

```
function ldc=crossval_dist(data);
nPart = size(data,3);  % Partitions are the 3 dimension
nVox  = size(data,2);  % Number of voxels
nCond = size (data,1); % Number of conditons (should be 2)
part = [1:nPart];
for n=1:nPart
    trainIndx = find(part~=n);
    testIndx =  find(part==n);
    Mu_hat = mean(data(:,:,trainIndx),3); % Calculate means
    ldc(n)=diff(Mu_hat,1,1)*diff(data(:,:,testIndx),1,1)';
end;
ldc=mean(ldc);
```

The mean cross validated distances are
101.9622 203.1120 227.0220 190.8586 51.4064 98.5119 126.0337 27.0826
65.0024 32.4206

And they are reported as an RDM in Figure 1c.

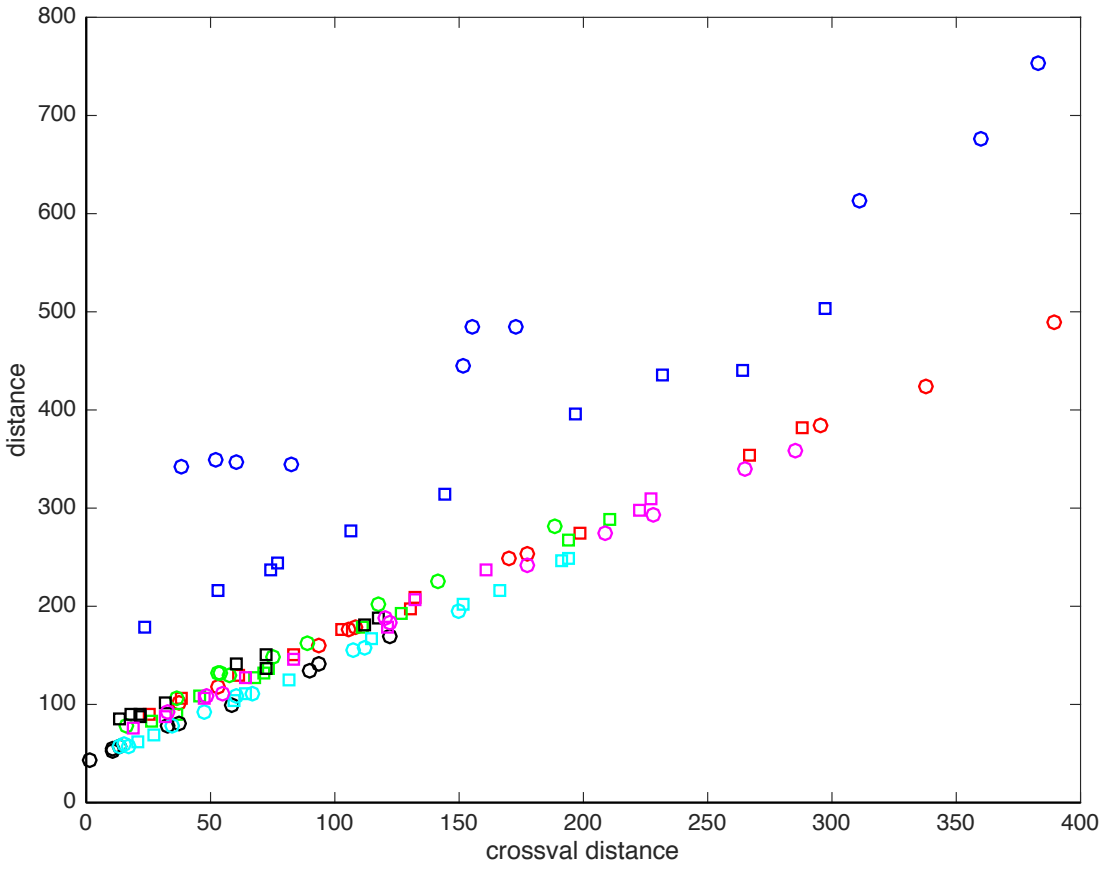

Figure 3: Accuracy against crossvalidated distance (LDC) against the normal distance.

The relationship within each subject between cv and normal distance is close to linear. However, in general the normal distance is the cv distance plus a constant (for most subjects ~50). For some subjects this constant is much higher (100-250). The constant offset is likely caused by the different noise level across subjects.

5. How stable is the RDM estimate across subjects? Calculate the correlation between the the vectors of 10 dissimilarities (%accuracies, sq. distance, sq. cross validated distance) for each pair of subjects. Report the average inter-subject correlation for each measure. Which measure yields the highest average correlation?

The average inter subject correlations across the 10 dissimilarities are:
accuracy : 0.717
normal dist : 0.888
crossval dist : 0.893

Thus, there is a big jump from classification accuracies to continuous distances, but not much improvement through crossvalidation.
In absence of knowing the ground truth - inter-subject correlation can serve as a proxy measure for the quality of the measure.

Overall code:

```
function varargout = homework5(what,varargin)
% Example matlab script to solve homework5
%
%
run=[]; % Run is a variable
switch(what)
    case 'calculate_distances'
        % Calculate the three different distance measures
        load('dataset_5.mat');
        data = CONTRA;

        % Get the pairwise classification accuracies
        for s=1:length(data)
            n=1;
            for a=1:5
                for b=a+1:5
                    T.acc(s,n)  = nn_classifier(data{s}([a b],:,:));
                    T.dist(s,n) = pdist(mean(data{s}([a b],:,:),3)).^2;
                    T.ldc(s,n)  = crossval_dist(data{s}([a b],:,:));
                    n=n+1;
                end;
            end;
        end;

        % Image scale the mean distances
        figure(1);
        subplot(2,2,1);
        imagesc(squareform(mean(T.acc))); colorbar;
        title('accuracy');
        subplot(2,2,2);
        imagesc(squareform(mean(T.dist))); colorbar;
        title('distance');
        subplot(2,2,3);
        imagesc(squareform(mean(T.ldc))); colorbar;
        title('crossval distance');


        % Plot the different measures against each other:
        color = {'ro','rs','bo','bs','go','gs','ko','ks','co','cs','mo','ms'};

        for s=1:12
            figure(2);
            plot(T.dist(s,:)',T.acc(s,:)',color{s});
            hold on;
            figure(3);
            plot(T.ldc(s,:)',T.dist(s,:)',color{s});
            hold on;
        end;
        figure(2);
```

```
        xlabel('sq. distance');
        ylabel('classification accuracy');
        drawline(0,'dir','horz');
        drawline(0.5);
        hold off;

        figure(3);
        xlabel('crossval distance');
        ylabel('distance');
        drawline(0,'dir','horz');
        drawline(0);
        hold off;

        varargout={T};
    case 'consistency'
        T=homework5('calculate_distances');
        R1=triu(corr(T.acc'),1);
        R1(R1==0)=NaN;
        R2=triu(corr(T.dist'),1);
        R2(R2==0)=NaN;
        R3=triu(corr(T.ldc'),1);
        R3(R3==0)=NaN;
        fprintf('consistency acc:%2.3f dist:%2.3f  ldc:%2.3f
\n',nanmean(R1(:)),nanmean(R2(:)),nanmean(R3(:)));

end;

function acc=nn_classifier(data);
nPart = size(data,3);  % Partitions are the 3 dimension
nVox  = size(data,2);
nCond = size (data,1); % Number of conditons
part = [1:nPart];
for n=1:nPart
    trainIndx = find(part~=n);
    testIndx =  find(part==n);
    Mu_hat = mean(data(:,:,trainIndx),3); % Calculate the training means
    % Now classify
    for c=1:nCond
        x = data(c,:,testIndx);  % This is the test pattern
        dist=x*x'-2*Mu_hat*x'+sum(Mu_hat.^2,2);
        [~,k(c,n)]=min(dist);  % Record the classification
    end;
end;
% Caluclate the % correct
correct=bsxfun(@eq,k,[1:nCond]');
acc = sum(correct(:))/numel(correct(:));

function ldc=crossval_dist(data);
nPart = size(data,3);  % Partitions are the 3 dimension
nVox  = size(data,2);  % Number of voxels
nCond = size (data,1); % Number of conditons (should be 2)
part = [1:nPart];
for n=1:nPart
    trainIndx = find(part~=n);
    testIndx =  find(part==n);
    Mu_hat = mean(data(:,:,trainIndx),3); % Calculate means
    ldc(n)=diff(Mu_hat,1,1)*diff(data(:,:,testIndx),1,1)';
end;
ldc=mean(ldc);
```