Homework 4

Background:
Listen to the two first lectures of the Multivariate analysis section: Classification and Spatial prewhitening.

Data homework:
The data set comes from the same experiment as the data set for homework 1 and 2. The data contains time series from 328 voxels from the *right* primary motor cortex from one subject that moved one finger at a time for 8.1s. Each of the 10 fingers was repeated 3 times per run, and 8 runs (each containing 123 images) were collected. Each image took 2.7s took acquire. The details of the methods can be found in (Diedrichsen et al., 2013, Ejaz et al., 2015). The file dataset_4.mat contains the following variables:

Y	984x328	Time time-series data (984 measurement points) from 328 voxels from right motor cortex.
XYZ	328 x 3	Location of the 328 voxels in x,y,z space in mm.
Xtask	123x10x8	The task-based regressors for Left and Right fingers. The first 5 columns are for D1-D5 of the left hand, column 6-10 are for D6-D10 of the right hand. Each trial is already convolved with the hemodynamic response function.
Xintercept	123x1x8	Intercepts for the runs
Xhpf	123x5x8	High-pass filter regressors for each run
run	984x1	indicator which run the time point belongs to.

1. Implement a nearest-neighbor classifier to distinguish between the 5 fingers of the left hand

a. Estimate the beta-weights for the 10 fingers. *Before* estimating the betas, remove the low frequency trends with the high-pass filter regressors, as done for homework 2.2 (see model answer in doubt).

b. Classify between the five fingers of the contralateral, left hand (first 5 rows). Implement a multi-class leave-one-out cross validated classifier as an independent function. The function should take the beta estimates for the left hand fingers for all voxels and runs as an input. It should then loop over all runs, each time using that run as a test set and the remaining runs as a training set. You then estimate the mean patterns for the 5 fingers across the 7 runs of the training set (μ_i) and then classify the 5 patterns of the left-out run depending on which one of the patterns from the training data set they are closest to. Perform these classification independent of each other (that is sometimes 2 or 3 patterns may be classified as “thumb”). Use the Euclidean distance: If x is the test pattern, and μ_i are the training pattern for the ith finger, then assign x to the finger for which $dist = (x-\mu_i)(x-\mu_i)^T$ minimal. The function should return the percentage accuracy. Overall you made 8*5 = 40 classifications. Report that classification accuracy. What is guessing baseline (random chance)?

c. Repeat the procedure for the 5 fingers of the ipsilateral, right hand. What do you find?

2. Derive a significance value from a randomization test

For each hand, repeat the classification procedure 1000 times, each time shuffling the patterns in each run independently. This simulates the null-hypothesis that for this region and this subject, there are no differences between patterns associated with the fingers. Call your classification routine with this shuffled data - building up a Null-distribution for classification accuracy. Is the classification for left and right hand significant? What is the p-value based on your null-distribution?

Advanced question: Why can we not approximate the distribution of number of correct classifications as a binomial distribution with p=1/5 and N = number(Conditions) * number(Runs)?

3. Prewhiten the data spatially using the residuals from 1-level regression

a. During the estimation of the betas, calculate the residuals time-series for each run and voxel. Estimate the voxel-by-voxel covariance matrix for each run (this should be a 328 x 328 matrix). Plot the co-variance matrix (or better the correlation matrix) for each run and compare them visually.

b. Calculate the average covariance matrix across runs (Σ) and regularize it by using a combination of a the diagonal version of this matrix (preserve the diagonal values, but set all off-diagonals to 0), and the original version of the matrix.
$\Sigma_{reg} = 0.1 diag(\Sigma) + 0.9 * \Sigma$. Prewhiten the estimated beta weights from all runs using this matrix. If the matrix of the task-related beta weights **B** for a run is a 10x328 matrix, and Σ_{reg} is your regularized 328x328 covariance matrix, you can prewhiten the data by calculating:
$\tilde{\mathbf{B}} = \mathbf{B}\Sigma_{reg}^{-1/2}$
Repeat the classification algorithm from (1) using prewhitened data and also redo and report the randomization. What do you observe?