1. Lecture 5.1 – Task
   1. Experimental Task
      1. Flanker **Task**: the experiment consists of a sequence of ca. 330 trials with inter-trial interval of 2s ± 1.5s
      2. At the beginning of each trial, an arrow is presented centrally (pointing either left or right)
      3. The arrow is flanked by congruent or incongruent “flanker” arrows:  
         
      4. The subject is asked to press left/right button, according to the central arrow, and makes frequent errors (25%)
         1. The flankers prime the person to make frequent errors
      5. We want to predict whether or not the person will make an error or not using BCI
   2. Consideration
      1. The peak ERP features discussed so far were chosen for a single channel of EEG
      2. **Problem**: with multiple channels all channels measure almost the same signal properties, thus little information gain to expect
         1. If we treat multiple channels like last time then we will not be utilizing all of the data possible because the peak latency, amplitude, and width will all be the same.
         2. Not using spatial filters, just random features
      3. **Idea**: Derive a spatial filter and use multiple channels to computationally focus on source processes of interest, then extract source signal features.
         1. Can we design a BCI that learns spatial filters which focus (in a computational manner) on a high fidelity time coarse from the channel data, then go on extract features of this source signal such as amplitude in the subsequent process.
         2. Also have processes that have more than two peaks
         3. We want to design and compute spatial features.
      4. How to design an optimal spatial filter for this task?
      5. **Idea**: Can be done implicitly by a linear classifier when applied to multiple channels.
         1. A linear classifier if applied correctly can actually learn an optimal spatial feature that can tell you how to weight the features properly
      6. Works only for source-signal features that are a *linear transform* of channel-signal features
      7. The classifier must produce the *same solution under rotation and scaling* (not all do, but e.g., LDA does)
2. Lecture 5.2 – Analysis Approach
   1. Approach
      1. Calibrate recording is band-pass filtered between 0.5Hz and 15Hz
         1. 0.5Hz lower edge removes drift
         2. 15 Hz upper edge leaves enough room for sharp ERP features
      2. Epochs are extracted for each trial and label is set to A for incorrect trials and B for correct trials
   2. Actual Data
      1. Time courses for all trials super-imposed (color-coded by class) – but here different task
      2. If you combine multiple samples you can remove high frequency noise
   3. Extracted Epochs
      1. Ok so now isolate one electrode at a time
      2. We further take an average of all green trials, then make two more signals from ± standard deviation
      3. We extracted this segment of data from the trials and we color red for A and green for B correct and incorrect trials respectively.
   4. Extracting linear features
      1. Then we just simply pick out what ever feature we want to look at and pick say three feature areas we want to look at
      2. For each trial segment, calculate signal mean in 3 time sub-windows ( 3-dimension feature vector because we extract three features!)
   5. Problem with LDA
      1. Multi-channel features are too high dimensional for LDA to handle with few trials!
   6. Fixing LDA
      1. Given trial segments (in vector form) in and ,  
         ,   
         ,
      2. Θ often high-dimensional but only few trials available
      3. Can use **regularized estimator instead**, here using ***shrinkage***
         1. Instead of , we use above:
         2. We have to do this to deal with degenerate covariance matrices
         3. It is a way to control the effective number of degrees of freedom in the parameters of the classifier
         4. Way to penalized overly complex models
         5. “Here are a bunch of models that I can learn, those over there are too complex, lets find one that is simple enough to capture the relevant data”
   7. Determining λ
      1. The regularization parameter is a free “tunable” parameter of the approach, depends on the data
      2. Can be found by parameter search (one cross-validation for each possible value) over a range like [0.0 0.1 0.2 … 0.9 1.0]
      3. Caveat: Parameter search can be *very* slow (10 possible values times 5 folds = 50x slower)
      4. Especially if nested inside an outer cross-validation
      5. Pick the one that gives the best performance
      6. Enter free parameters to trade off complexity and we search for the best value
      7. In the special case of shrinkage LDA, *λ* can be determined analytically or as the result of a convex optimization problem.
      8. Some further choices exist (e.g. empirical Bayes estimator, information criteria, …)
3. Lecture 5.3 – Review
   1. Resulting Spatial Filters
      1. Topographically mapped, the following filters emerge
         1. Ok, so we have an epoch that we split into 7 different feature vectors and we have a 32 channels or electrode and our time windows are consecutive and only last several seconds.
         2. A parameter vector θ which is 32 \* 7 in this case
         3. The rule for applying these weights is take your epoch and for every channel average the signal in the first window 250 to 300ms and that gives you across all those channels 32 numbers, this is the averages in this window for this one trial, and then linearly combine them according to this weight, then sum those all up. This gives you a single number, then do this for every other window. All of this is linear and what we end up getting is one vector times another vector and our inner product is essentially one number and all of those weights essentially are the vector that make up the Hyperplane! Hereby classifying our stuff…
   2. How good is This Approach?
      1. Source activation S can be recovered from sensor measurements by a linear mapping if (linear) volume conduction (forward mapping) is invertible
      2. Assuming a jointly Gaussian noise process and a noise distribution that is independent of the condition (A/B), LDA approximates the *optimal linear mapping*
         1. This means you must have jointly Gaussian noise process, you have jointly Gaussian error in both conditions, that means in the sources and in the channels.
         2. If you have one class that is more probable then the other than everything sort of messes up and this does not work
      3. Shrinkage LDA on these features yields state-of-the-art ERP performance!
         1. Without shrinkage LDA would not work
         2. With shrinkage it works great as long as there are no outliers!
         3. Ask your subject to not move
      4. Linear classifiers like LDA can operate implicitly on source ERPs, but:
         1. EEG variation is often *not* Gaussian
            1. Might prompt error messages
         2. Data variability *can* depend significantly on condition
            1. If the variability of your data depends on the condition and the subject is making an error, they have larger variance in the data then if they make no error, then the distributions do not have equal covariance and then the optimal mapping is not linear anymore
         3. For limited data samples, LDA is not necessarily optimal
         4. Results are only “mildly” interpretable…
4. Lecture 5.4 – Advance ERP Topics
   1. Equivalence under Linear Transforms
      1. **Note**: LDA on linear features yields the same result (but linearly transformed) with the same performance when applied to any (non-reductive) linear transformation of the data
         1. Principle Component Analysis, Independent Component Analysis, Non-adaptive Beamforming
         2. We cannot use when non-linear.. I think
         3. LDA when applied to linear filters you get the same performance even if you apply it to linearly transformed data. LDA would have just learned different weights, as long as you don’t through information away.
      2. **But**: These can be used to
         1. LDA assigns a weight to every component so you can easily say do not pay attention to this eye blink or muscle movement
         2. Better interpret or localization underlying sources of a classifier, e.g., artifact/non-artifact components
         3. Introduce location-dependent constraints or prior knowledge into the classifier
         4. Could say I don’t want to monitor anything from the occipital cortex because I don’t think my signal originates from here
   2. Other Linear Features [besides averages]
      1. There are other linear features you can use besides averages
         1. If you know that your data comes in the form of ripples
      2. Wavelet transforms of the source time course
      3. Allow you to design features adapted to intricate temporal characteristics of the signal (e.g. ripple, rebound, etc.)
      4. Can design generic features and employ feature-selection or sparse classification techniques (more later)
      5. If you know that your peak comes in other forms, then you can use inner product to do pattern matching of the wavelet features
         1. Pick features of your wavelet and use them as your features
      6. Only makes sense if you throw away many of the features and do simple pattern matching
   3. Non-Linear Features
      1. Extracting non-linear source signal features is not easy to get right on channel data
         1. The general class of features, in fact linear is a special case
         2. You want non-linear features that are features of the source time course because you think there maybe be something non-linear happening at the source, not the channel source, as opposed to non-linear features of the channels
         3. So if you do non-linear transform somewhere in your feature extractions somewhere on channels, and then apply classifier you have basically done it the wrong way because you did it too early
         4. The right way is to design the spatial filters first, which gets you to the force then to a non linear feature extraction then maybe apply a non-linear classifier to that
      2. In theory, non-linear classifiers could recover such source features, but *in practice* most fail to capture the necessary structure for the given amount of data
      3. Can be handled by a latent-variable model that represents source signals explicitly (more later) such as certain 3+ layer neural networks
      4. Examples: relative measures (e.g., amplitude ratios), effective connectivity, …
   4. Signal Detection Aspects
      1. ERP analysis often amounts to classifying a characteristic ERP vs. a non-ERP / background noise where class ratios are often very imbalanced (e.g., RSVP target detection tasks)
      2. In such cases other evaluation measures than miss-classification rate rates are needed
      3. A canonical example are different costs per failure type (e.g., high false negative costs) *if such costs are known*
         1. Use different criteria to measure these artifacts
      4. When you want to detect a signal in noise
   5. Signal Detection Aspects
      1. Linear analysis does not really work if 90% of the time you are in one state and only 10% in the other because let’s say that in the 90% state you are in your true state then well you would be measuring true 90% of the time and that’s not right
      2. A general-purpose measure is Area under Receiver Operator Characteristic (AUC or AUROC) – quantifies performance over all cost choices
         1. For when you are dealing with unbalanced classes
         2. If your model has a tunable threshold
      3. Can be approximated efficiently for given targets and associated predictions
      4. You can go from 0 to 100% false positive rate you will have a 100% to 0% true positive
         1. For 0% false positives you have 100% true positives
         2. If you have 50% false positive you have 50% true positive
         3. Useful for is your model outputs scalars and you want to determine the probability you have seen what you are looking for
   6. Impact on the Classifier Choice
      1. Most classifiers allow in principle for weighted cost structure, if known (e.g. LDA, logistic regression, Support Vector Machines)
      2. Caveat: Most classifiers assumer that the class ratio in the training data equals their prior probability on test data (e.g., logistic regression)
      3. Some classifiers can be directly trained to optimize the AUC criterion (e.g. boosting, SVMperf) and there are ways to use any binary classifier (active research topic)
      4. taco