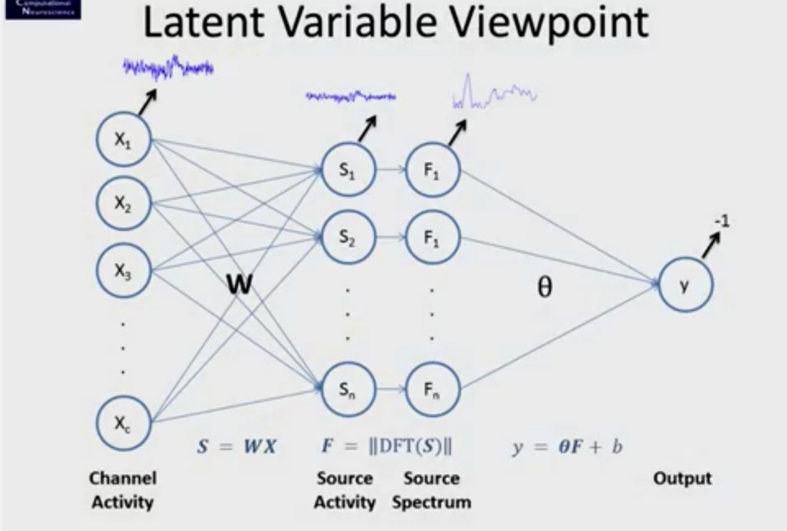
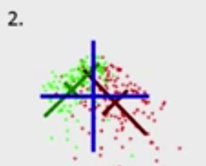
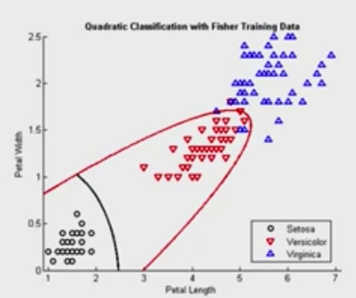
1. Outline
   1. Basics and examples
   2. The Spatial Filter Problem
   3. Common Spatial Patterns
   4. Alternatives and Extensions
2. Lecture 7.1 – Basics and Examples
   1. Oscillatory Processes
      1. Best Example: Cortical idle rhythms, e.g. occipital alpha, motor cortex alpha+beta
         1. There is general fallout with the amplitude vc. Frequency graph where the lower frequencies have higher power and the higher frequencies have lower power.
         2. Higher frequencies have more energy
         3. There is a lot of energy around the 10 Hz range
         4. Large group of neurons firing in sync when not activated
         5. These graphs tell us which parts of the brain are not being engaged
   2. Experimental Task
      1. The experiment consists of 160 trials (pause at ½ the experiment). Each trial begins with a letter (either L or R) displayed for 3s. The subject is instructed to subsequently imagine either a left-hand or a right-hand movement. Each trial ends with a blank screen displayed for 3.5s.
         1. If the person actually moved their body they would produce a ton of artifacts and such, causing you to not really know if the BCI was really working.
   3. Motor Cortex ERD/ERS
      1. Event-Related Synchronization / Desynchronization: attenuation of motoric idle rhythms in response to an event
      2. Average spectrogram for left-hand movement imagination in red + average spectrogram for right-hand movement imagination in green (160 trials each, stimulus at t=0)
      3. This is actually an average of many trials, further these are the channels with the other channels surrounding it subtract, which is actually a surface lapacian.
      4. Ok so essentially if you imagine movement of the left hand idle rhythms on the right side of the brain near C3 break down, vice versa for imagining trying to move the left.
      5. Alternative visualization of information content per time-frequency residual, same data.
         1. Shows that there is power in the 10 Hz band for several seconds
      6. The duration of the power intensification is not known
      7. The exactly frequency where this power increase will be seen is not known
3. Lecture 7.2 The Spatial Filter Problem
   1. Quantifying Oscillatory Processes
      1. Nonlinear operation in play, on *source* signals
         1. Somewhere in your pipeline there is a non-linear operation
         2. If you were to use and entirely linear method you would be doing ERP and you are trying to linearly map an this oscillating signal onto the output.
         3. Since the phases for each trial may not be the same, if you took the average of many trials you would essentially get zero because the none-phase locked waves would overlap and such.
      2. Necessary due to *shift* *indeterminacy* of source waveforms (no precise time/phase-locking, jitter ..)
      3. In oscillatory processes represented by determining the amplitude of source oscillations  
         , ,
         1. Three step mapping from raw channel data
         2. ***X***
            1. Matrix of raw channel data which has dimensions number of channels by number of time points
         3. ***W***
            1. Spatial filter
         4. ***S***
            1. Approximation of something Spatially filtered
            2. Called Source Time Courses
            3. Would look like 3 by 1000 time point matrix for one trial
         5. ***F***
            1. Spectral characteristics
            2. When we want to get to the spectrum
         6. *DFT*
            1. Discrete Fourier Transform
         7. *||DFT(****S****)||*
            1. Fourier transform on the source time point matrix, S, which is a linear movement
            2. Take the absolute value of it to get the magnitude of that oscillation, the power
         8. *y*
            1. The linear mapping onto the output
            2. If there was only one channel we would have a inner product of a vector and the channel weights ***θ*** and add a bias
         9. ***Θ***
            1. Linear Classifier
      4. Nonlinear operation, also discards phase information (If done on channels, source spectral properties cannot be recovered)
   2. Linear Variable Viewpoint  
      
      1. I have multiple channels
      2. The relationship between channel activity and the source activity is W
         1. So S1 depends on everything by some weights
      3. Then we do our frequency transform and get the frequency representation as power
      4. At the end we have our linear classifier which maps it onto a one-dimensional value (scalar) that could be -1 or +1
      5. What we cannot calculate though is W because LDA cannot optimize it because there is custom stuff between S and y
   3. How to Learn the Spatial Filter W?
      1. **Option A – No Learning**: use fixed *ad hoc* filters instead
         1. Performance not abysmal, but *far from optimal* – room for improvement.
         2. Common Average Reference
            1. We take a channel and subtract all the others and take the average and we just optimize the Θ.
            2. Does not work that well
         3. Bipolar Derivations
            1. Take this channel and minus that channel
         4. Surface Lapacian Derivations
            1. The channel minus the neighboring ones
      2. **Option B – Top-down**: Using neural-network like back-propagation / gradient descent (supervised learning)
         1. Inputs **X** are known, desired outputs y are known, spectral mapping in between is known
         2. For any (**W,*Θ***) can calculate the loss given known X and y, and update it.
         3. If you take some value for W and θ and you know the data and the labels you can optimize these two using gradient descent or something
            1. The problem with using gradient descent though is you will have lots of local minima in this parameter space and you would improve this if you had an update rule for it
         4. This is a multilayer neural network that you can use back-propagation to update this.
      3. **Option C – Bottom-up**: Without looking at the labels y, learn a good spatial filter **W** for the data (unsupervised learning)
         1. Criterion for a good spatial filter? Independent Component Analysis, Dictionary Learning, PCA
            1. Tons of methods you can apply
         2. We are not trying to look at the labels, we are completely ignoring task parameters, we are just looking at the raw data and trying to learn the W from it
         3. Maybe we can look at our data and capture the dominate structure and find the dominate sources and find spatial filters to extract these components.
         4. So you are hoping that the filters that pick up a lot of data contain the information you are looking for, however there is no guarantee of that
      4. **Option D – Both**:
         1. Perform a mixture of unsupervised and supervised learning
         2. Supervised ICA, Unsupervised pre-training + supervised fine-tuning,…
         3. First do an ICA to calculate **W**, then sort of somehow tune the component weight vector using a supervised criterion.
         4. Can use the label information around ***Θ*** to optimize W a little better.
      5. **Option E – Using Direct Observations:** Is there a way to observe **W** directly from data?
         1. If given an MR scan (or default image), can use e.g., Beamforming
      6. **Option F – Using Additional Assumptions**: These can make the problem solvable
         1. Powerful assumptions: The source activation in the time window of interest is *jointly Gaussian-distributed* whoa
         2. Make enough extra assumptions on the data that our problem of deducing W becomes analytically solvable
         3. The time series that is an oscillating signal that is jointly Gaussian distributed
            1. Source activations are Gaussian
            2. After you have mapped them on the channels they are even more Gaussian
4. Lecture 7.3 – Common Spatial Patterns
   1. Common Spatial Patterns
      1. Most popular algorithm in BCI field for learning spatial filters for oscillatory process
         1. Cannot use for ERP
      2. Assumptions
         1. Frequency band and time window are known
            1. If you do not know the frequency band then it does not work as well if you do not get it exactly right
         2. Band-passed signal is jointly Gaussian within the time window
            1. Over or under estimate and it will not work that well
         3. Source activity constellation differs between two classes
            1. Some how there is some difference between the two oscillations.
      3. Different source activities for a left-hand epoch vs. a right-hand epoch (band-passed to 7-30 Hz)
      4. Signal activation is scatter-plotted for channels C3 and C4
         1. Take a single try of a person taking a left hand movement and you just scatter plot the signal for two channels (C3 & C4 here)
         2. Take every sample in this two channel time series and plot the excursion for one channel on one axis and the excursion for the other channel on the other axis
            1. This gives you a scatter plot
            2. These are samples in one trial!
         3. We notice that there is higher variance of channel then the other
         4. The two channels are pretty much correlated
         5. Can’t really tell which class is which from just looking at it… not ideal
         6. The task now is to do a linear transform of this so we can directly measure the variance and get a lot of information on whether it is condition 1 or 2
      5. This algorithm designs a linear transform, a set of spatial filters, mapping the data into a new space where it is much more discriminative
      6. Goal: Design a pair of spatial filters (i.e., spatial transforms) such that the filtered signal’s variance is maximal for one class while minimal for the other
         1. The data turns to be mutually diagonal or perpendicular
      7. And vice versa
         1. If we look at the variance of one channel it is very small on one axis and large on the other, while the other channel is completely opposite
   2. Three Ways to Compute Spatial Filter
      1. Optimization Problem: Given a set of *t* trial segments , per-trial covariance matrices , and per-class average covariance matrices , optimize the spatial filter for class *c* as:  
          such that
         1. The signal is jointly-Gaussian so you can calculate the covariance matrix of the signal in one condition and you completely characterized it, you cannot get more information then this matrix
         2. So here for example there are two covariance matrices for class “-1” and class “+1” [ from eq above]
         3. The term where you take a spatial filter and you multiply before a covariance matrix and then you multiply the same spatial filter after the covariance matrix, applying that gives you a value that is the equivalent to taking a signal, apply a spatial filter and then taking the variance of the spatial filtered signal.
            1. This is a way to essentially look up from a covariance matrix what the variances in any direction in space are
         4. In short, in the equation above, we are trying to optimize ***w*** such that ***w*** is maximal so that the projected variance is maximal.
         5. The constraint on the right hand side of the equation says that if I take the average covariance matrix (the sum of both classes) and spatially filter it, we get 1. The average covariance should be 1. But looking at the covariance for one class it should be maximal and minimal for the other class.
         6. The reason there is a c in is because you have one optimization problem that gives you one axis and the other spatial filter solution gives you the other axis
      2. Generalized Eigenvalue Problem: Given per-class average covariance matrices , find the simultaneous diagonalizer ***V*** of and :  
         ,  
         ,  
         for diagonal and such that .
         1. This yields a generalized eigenvalue problem of the form
         2. The k smallest and largest eigenvalues in ***D*** correspond to the k leftmost/rightmost columns in ***V*** (spatial filters) that yield smallest (largest) variance in class -1 and simultaneously largest (smallest) variance in class +1
         3. Very easy in MATLAB:  
            >>[V,D] = eig(cov1, cov1+cov2)
            1. V here is actually your spatial filters
            2. D are you eigenvalues
         4. The proof as to why these works are tough
      3. Geometric Approach: A more intuitive approach is a three-step procedure:
         1. Determine a *whitening* transform ***U*** for the average of both covariance matrices (blue) using PCA  
            
            1. Red, Green are channel distributions
            2. Blue is the average
         2. Apply it to one of the matrices and calculate its principal components ***P*** (green)  
            
         3. The spatial filter operation ***W*** is to first whiten by ***U*** and then transform by P-1, i.e. so then   
            
      4. Step 1 is using PCA to find the principle components of each distribution we get the vectors for each of them. Step 2 we transform the data into the average distribution (scaling and rotating the other distributions in respect to the average distribution) coordinate system we get unit variance in all directions. We actually in the background have diagonalized the two distributions, however we still have nothing we can read out in respect to variance to classify, so step 3 we rotate it again into either the red or the green coordinate system, now we can directly read out the classification, such that this example green variance is low and red variance is high.
   3. Resulting Spatial Filters
      1. Produces well-adapted filters (left) and occasionally roughly dipolar filter inverses (right)
         1. The inverse is simply the forward projections
      2. Note that typically only filters for the k top and the k bottom eigenvalues are retained.
   4. CSP Prediction Function
      1. The CSP Prediction function amounts to:
         1. Spatial filtering
         2. Log-variance calculation
         3. Application of a linear (of non-linear) classifier
   5. Putting it all Together
      1. A CSP-based BCI typically operates on a band-pass filtered signal
      2. Choice of the frequency band is not trivial
      3. The online window length does not need to correspond to the training window length
      4. Progression is
         1. Raw Input Data
         2. Filter Graph (just a band-pass filter)
         3. Sliding window of the output of ***X***
         4. Get prediction function
5. Lecture 7.4 – Alternatives and Extensions
   1. Choosing the Classifier
      1. Feature space is low-dimensional (4-6) and distributions are well-behaved
      2. Simple linear classifiers perform well, LDA is hear to beat in practice (strong assumptions)
      3. Some groups prefer Quadratic Discriminant Analysis (QDA) or other classifiers
      4. What is really nice about CSP is it gives you useful features, the most informative spatial filter for one class and gives you another one for the other class, so with just the two features that come out of CSP, the log variance, you are well on your way to take a couple more features and include a couple more eigenvalues to work with, so we are left with low dimensional space.
   2. Alternatives to LDA
      1. Omitting the assumption of *condition-independent noise* yields Quadratic Discriminant Analysis (QDA)
         1. Surprisingly(?), QDA vary rarely performs better than LDA  
            
         2. QDA is much easier to destroy, just with one outlier
      2. Fitting multiple Gaussians for each condition instead of one yields Gaussian Mixture Models (GMMs)
         1. GMMs serve as a good “low anchor” in BCI benchmarks
         2. Easy to break with outliers
         3. Note that there is *no efficient procedure* to calculate the *globally optimal* GMM fit
         4. The number of Gaussians is usually not know in advance (unless given by sub-conditions), but can be estimated using Bayesian methods (e.g., VDPGM\*) or found via parameter search
      3. Important Type of Classifiers: Discriminative models (as opposed to Generative)
         1. Discussed in the next lecture
         2. Trying to optimize the Hyperplane
         3. Don’t try to optimize the data, just the classifier
   3. Alternatives and Extensions
      1. CSP is the most popular spatial filtering method in the BCI field for oscillations
      2. There exist >20 extensions addressing various limitations (frequency bands, time window, …)
      3. The most successful variants so far:
         1. Spectrally Weighted CSP (adaptive spectral bands)
         2. Filter-Bank CSP (multiple time/frequency windows and *feature combination*)
         3. Regularized CSPs (if too few training trials)
   4. Spectrally Weighted CSP (Spec-CSP)
      1. One of the best algorithms for learning the correct frequency bands (others: CSSP, CSSSP)
      2. Iterative algorithm that alternates between optimizing the spatial and spectral filter (block coordinate descent)
         1. Spatial filters are optimized using CSP
         2. Spectral filters are optimized using Person’s correlation coefficient
      3. Updating a spectral filter given spatially filtered data
   5. Resulting Spatio-Spectral Filters
      1. Adaptive filters for left vs. right hand movement imagination
   6. Spec-CSP Prediction Function
      1. For simplicity here implemented without any signal processing stage, using a temporal filter matrix (not very efficient):
         1. ***S***
            1. Spatial filters
         2. ***T***
            1. Temporal filters
            2. Temporal linear transform
            3. On the time side
            4. Different ways to optimize this matrix
         3. ***Θ***, *b*
            1. Linear classifier
      2. Can be used for any other CSP-like approach that requires a temporal filter  
         1. If you have adapted or learned frequency weights then you can use a somewhat more complex prediction function
   7. Regularized CSP Variants
      1. Add a regularization term and parameter that needs to be searched via grid search
      2. Compared methods
         1. Basic CSP
         2. Generic Learning CSP
         3. Spatially Regularized CSP
         4. Weighted Tikhonov-Regularized CSP
      3. 5 pathological data sets (from BCI competitions)
         1. If you have way too few trials to learn any parameters or there is lots of garbage happening in your recording then CSP can FAIL quite spectacularly
      4. All of these methods have a regularization parameter that is optimized using cross-validation
         1. They are all in the BCILAB toolbox
   8. Multi-Class Extensions
      1. Most CSP variants are inherently defined for two classes
      2. Not a problem – can solve CSP for pairs of classes, train pairwise classifiers, and determine most likely class by voting
      3. Possibilities: one-versus-rest, one-versus-one
      4. Note: Classifiers should preferably produce probabilistic outputs so that voting can be done as probabilistically evidence gathering
      5. A useful probability classifier is *logistical regression*
   9. Time Window Estimation
      1. The time window is a free parameter that depends on the task of interest
      2. Can be searched as a 2D parameter space (slow!)
      3. Can be chosen via heuristics (threshold the correlation coefficients – or use them as weighting)