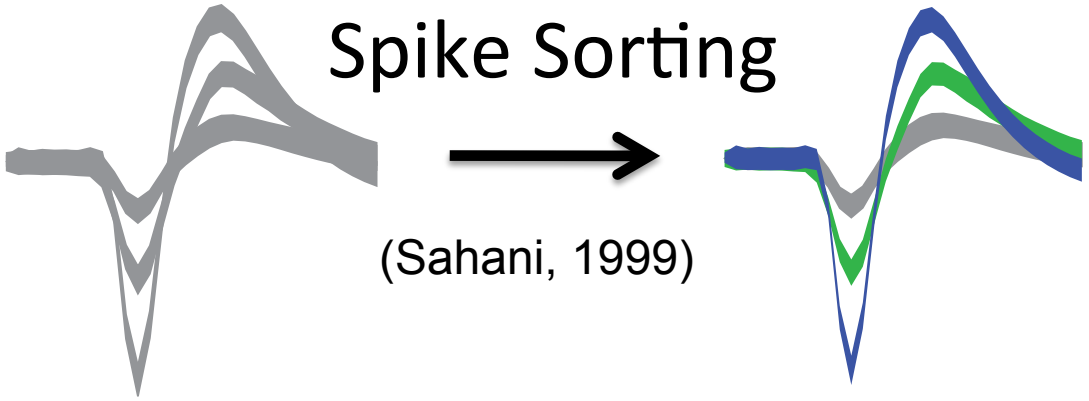
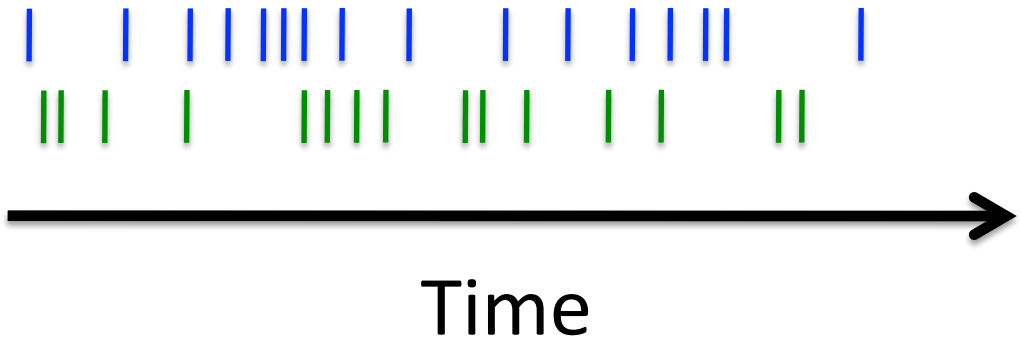
Lecture 14 – Spike Sorting

- Principal components of spiking waveforms
- Kmeans, Mixture of Gaussians spike clusters
- Feature selection

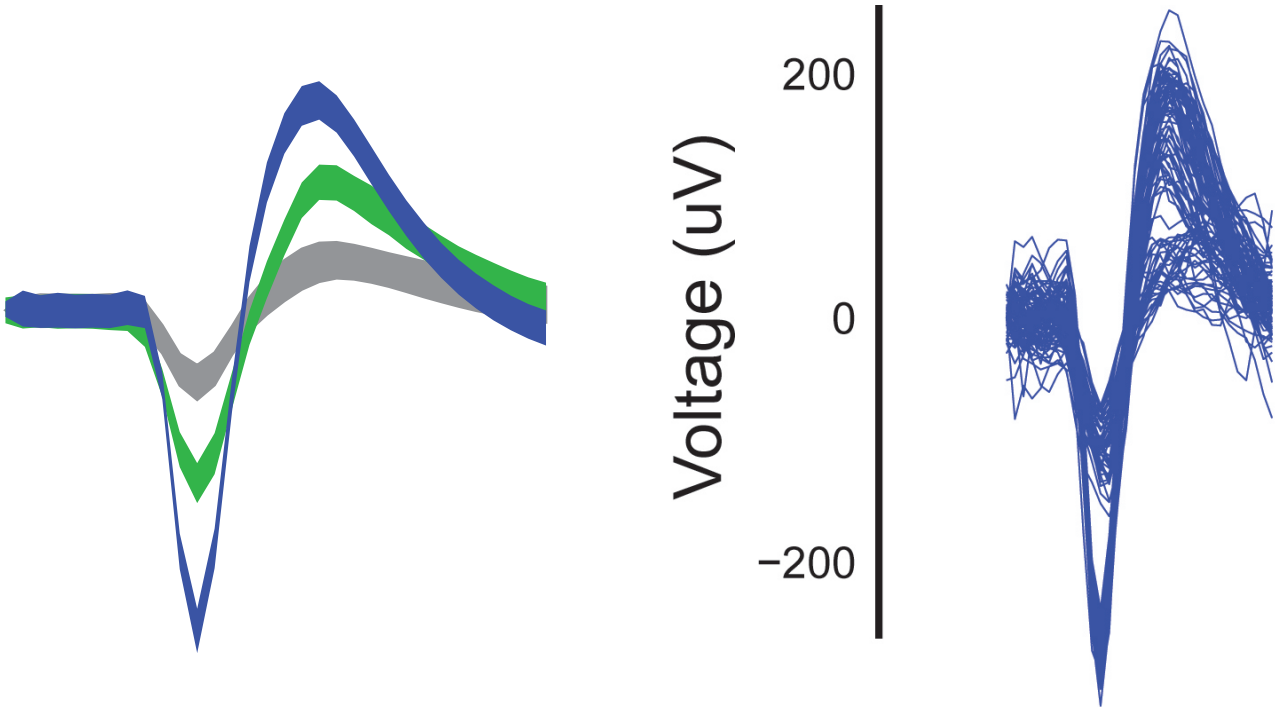
Spike Sorting

Action Potentials and Firing Rate



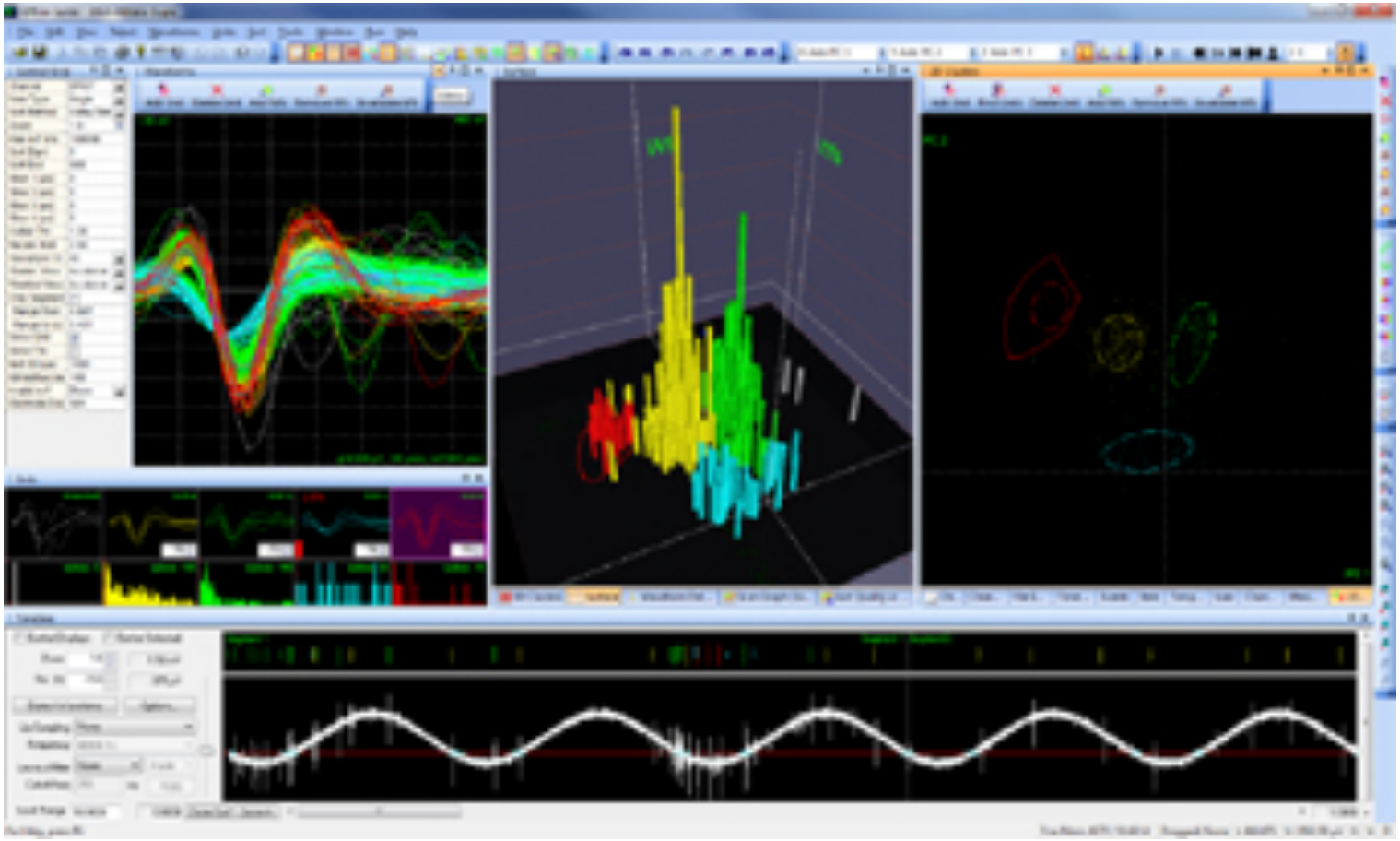


Spike Sorting



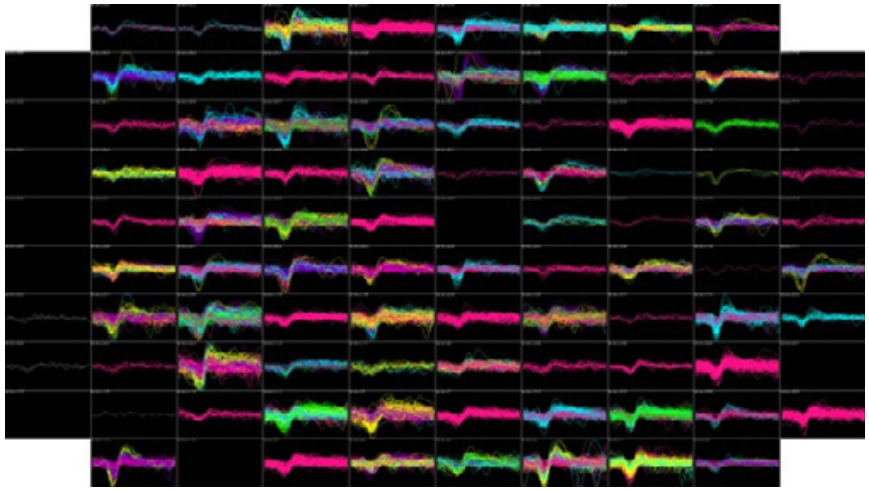
Manual Spike Sorting

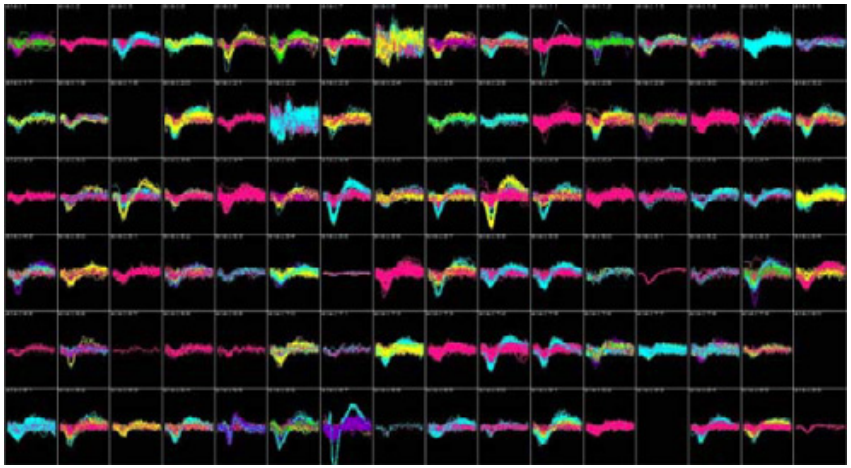
- Plexon Offline Spike Sorter



Results

- Normann et al., 2009

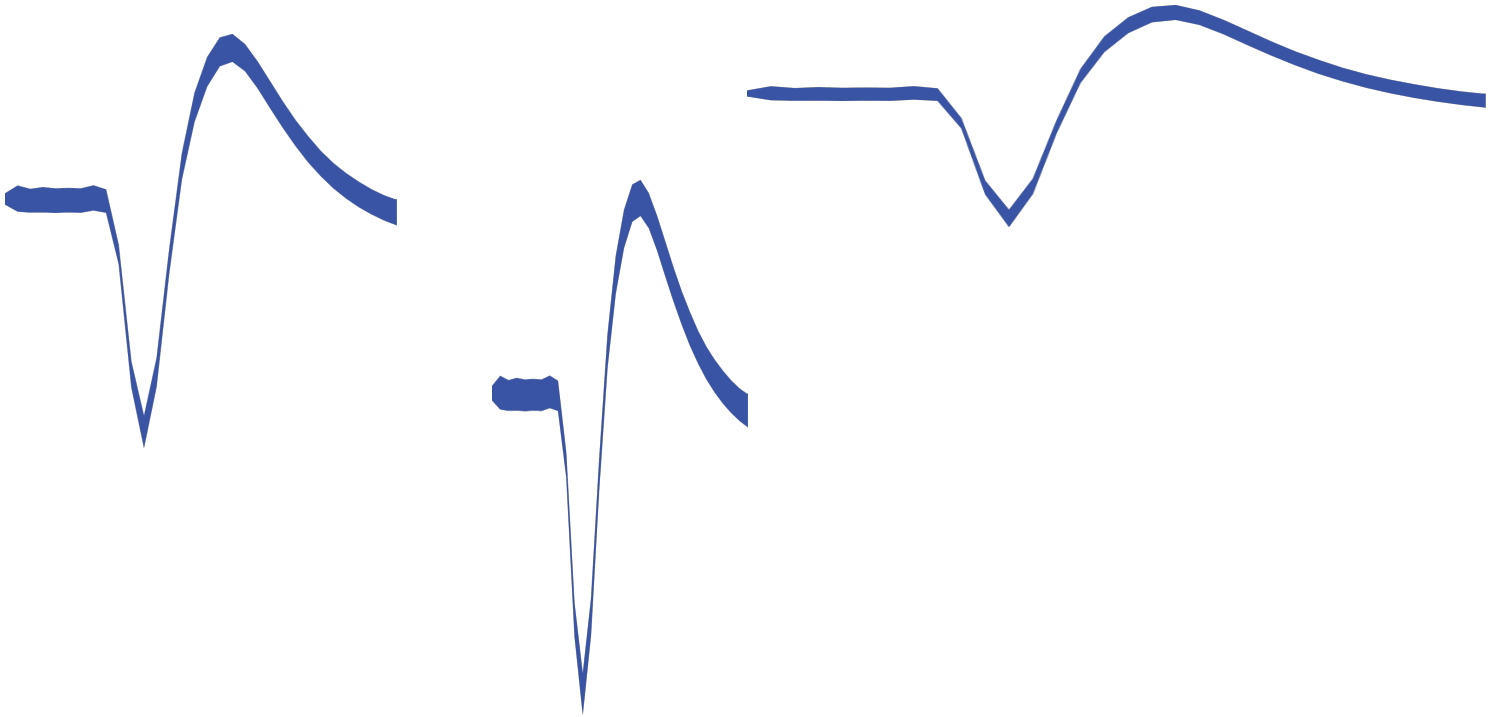




Matlab Bit

Automated Spike Sorting

- Can get a wide variety of heights, widths.
- Can be triphasic.



Principal Components

- One event: a neuron spikes
- Lots of redundant measurements of that event (32 samples of spike snippets)

- One event: reach to the right
- Lots of redundant measurements of that event from recording from 200 relevant neurons in motor cortex

Dataset of many spikes

- Spikes = [spike events, 32 samples];
- Want to choose first “principal component” **u** such that $\mathbf{u}^T\mathbf{x}$ gives me the best single number for differentiating between the different spikes
- Spikes have things in common, so there’s a characteristic covariance of the data, S, describing how individual spikes deviate from the mean spike.

Dataset of many spikes

- Try picking a vector **u** that amplifies/multiplies the data's covariance
- But not a dumb solution like **u** = infinity
- So maximize…

$$\vec{u}^T S\vec{u} + \lambda(1 - \vec{u}^T\vec{u})$$

Solving…

- Take derivative wrt **u** and set to 0

$$\frac{\partial}{\partial \vec{u}}\left(\vec{u}^T S\vec{u} + \lambda(1 - \vec{u}^T\vec{u})\right) = 0$$

$$2S\vec{u} - 2\lambda\vec{u}) = 0$$

$$\frac{\partial}{\partial \vec{x}}\vec{x}^T A\vec{x} = 2A\vec{x}$$

$$S\vec{u} = \lambda\vec{u}$$

Definition of an eigenvector
(Matlab command “eig”)

Multiple Principal Components

- Covariance matrix is 32 x 32 samples
- Have 32 possible eigenvectors, eigenvalues
- Biggest one corresponds to dimension with the most variance, and so on
- Can decompose spikes into principal component vectors, and a multiplier for those vectors

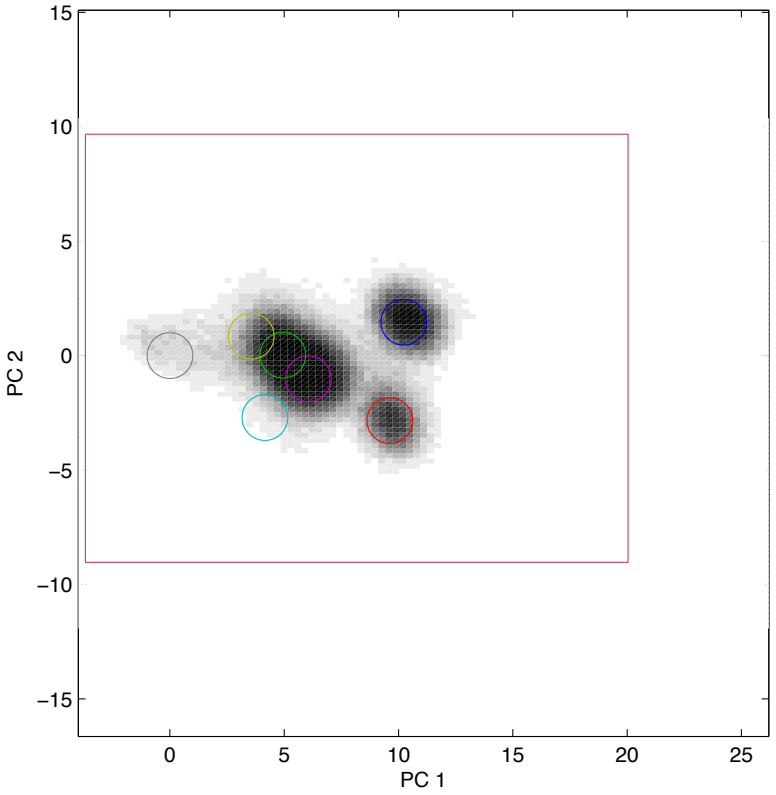
Matlab Bit

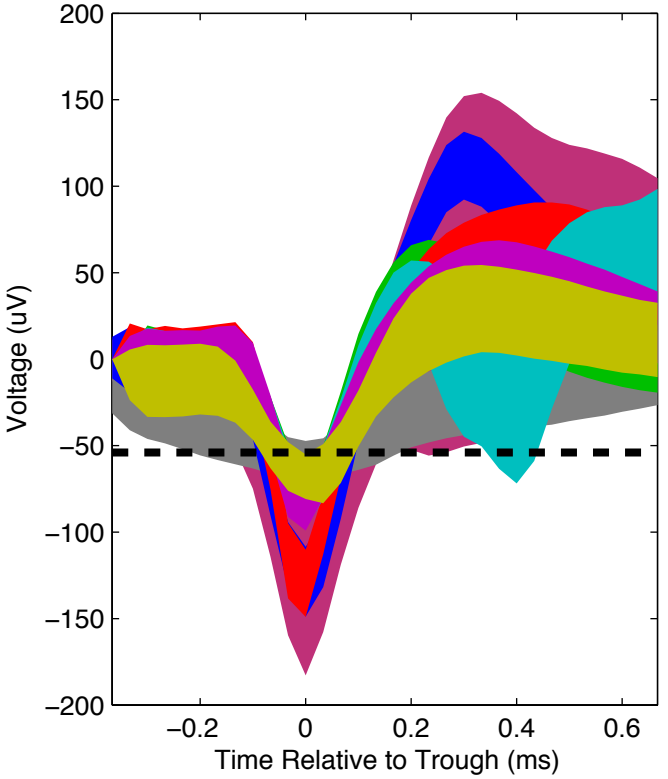
Code

```
load spikes
plot(spikes(1:1000,:)','b');

stdsp=std(spikes);
meansp=mean(spikes);
[n m] = size(spikes);

spikesnorm = (spikes - repmat(meansp,[n 1])) ./ repmat(stdsp,[n 1]);

[coeff,score,latent]=princomp(spikesnorm);

cumsum(latent)./sum(latent)
%top six components give you 90% of the variance

plot(coeff(:,1))% Prototypical spike, first principle component
plot(coeff(:,2))% Looks important

%What is the same thing
plot(score(1,:));
plot(spikesnorm(1,:)*coeff);

%How do you get the data back?
plot(score(1,:)*coeff')
plot(spikesnorm(1,:))

%How do you make it look like a spike again?
plot(spikesnorm.*stdsp+meansp);
plot(spikes(1,:));

%Now do it with only the top four components
plot((score(1,1:4)*coeff(:,1:4)').*stdsp+meansp,'b--')
hold on
plot(spikes(1,:));

%Why is this useful?
plot(score(1:1000,1),score(1:1000,2),'.');

%Looks like there's a nice division at 1.5
ind1=(find(score(:,1)>1.5));
ind2=(find(score(:,1)<1.5));


%Now we've sorted out two units
plot(spikes(ind1(1:100),:)','b')
hold on
plot(spikes(ind2(1:100),:)','g')
```

Results

- From a Utah array electrode





Automatic Clustering?

- Want groups where intragroup distances are small compared to distances to points outside the cluster

- Easy way: K-means
- Harder way: Mixture of Gaussians (using Expectation Maximization to fit the Gaussians)

K-Means

- Decide that there will be K clusters

$$k = 1, 2, 3 \ldots K$$

- Each datapoint n has a 0 or 1 for membership in cluster k

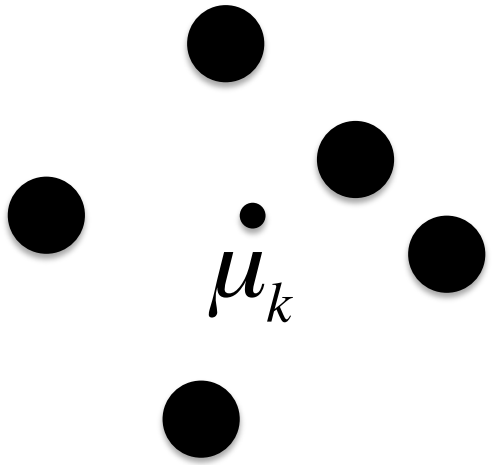
$$r_{nk} \in \{0, 1\}$$

- r_{nk} = 1 for assigned cluster, r_{nk}=0 otherwise

K-means

- Each cluster k has a centroid μ_k
- Choose r_{nk} to minimize distance of each point to the centroid of it's assigned cluster
- Minimize:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| \vec{x}_n - \mu_k \right\|^2$$



Iterative Process

1) Pick k random data points to be your first set of centroids
2) Assign each datapoint to nearest centroid

$$k = \arg\min_j \left\| \vec{x}_n - \mu_j \right\|^2$$

3) Recalculate the centroids and repeat

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

4) Stop when no one changes cluster membership or you run out of clock cycles

Matlab Bit

Code

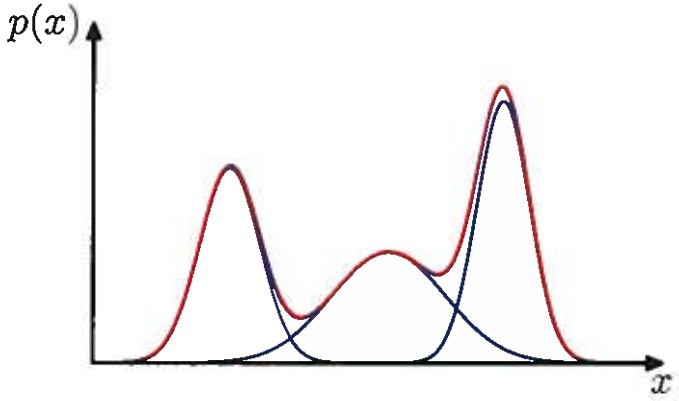
```
clusternum=3;
colors=['bgrmc'];
[ind,centroid]=kmeans(score(:,1:2),clusternum);
close all
hold on
for i=1:clusternum
   plotind=find(ind==i);
   plot(score(plotind,1),score(plotind,2),[colors(i) '.'])
   plot(centroid(i,1),centroid(i,2),[colors(i) 'x']);
end
```

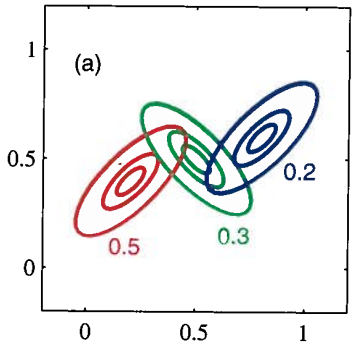
Problems with Centroids

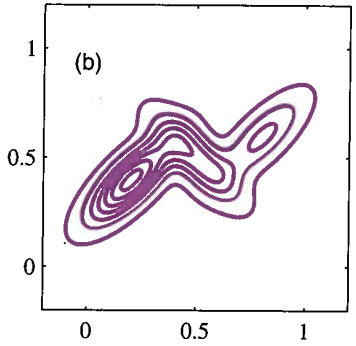
- Single units have very little “mass” in their clusters.
- Kmeans ignores low firing rate clusters

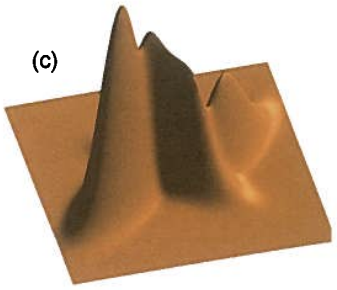
Mixture of Gaussians

- Can make all kinds of complex topography by fitting with multiple Gaussian distributions
- Can do this in any number of dimensions



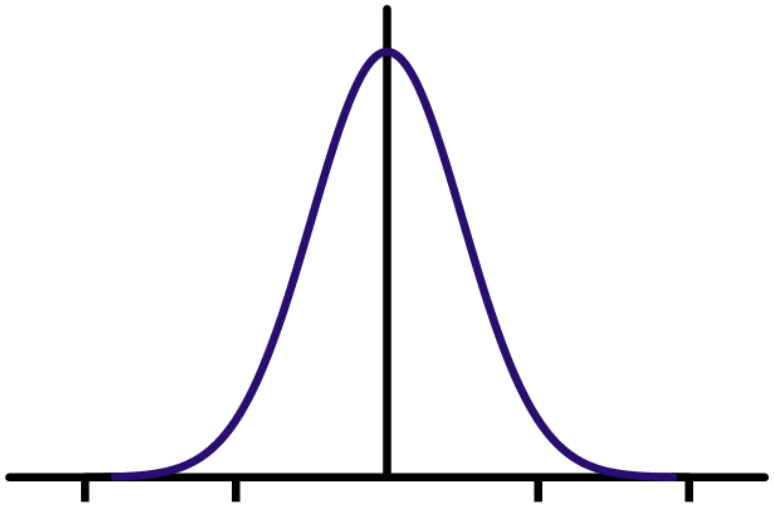


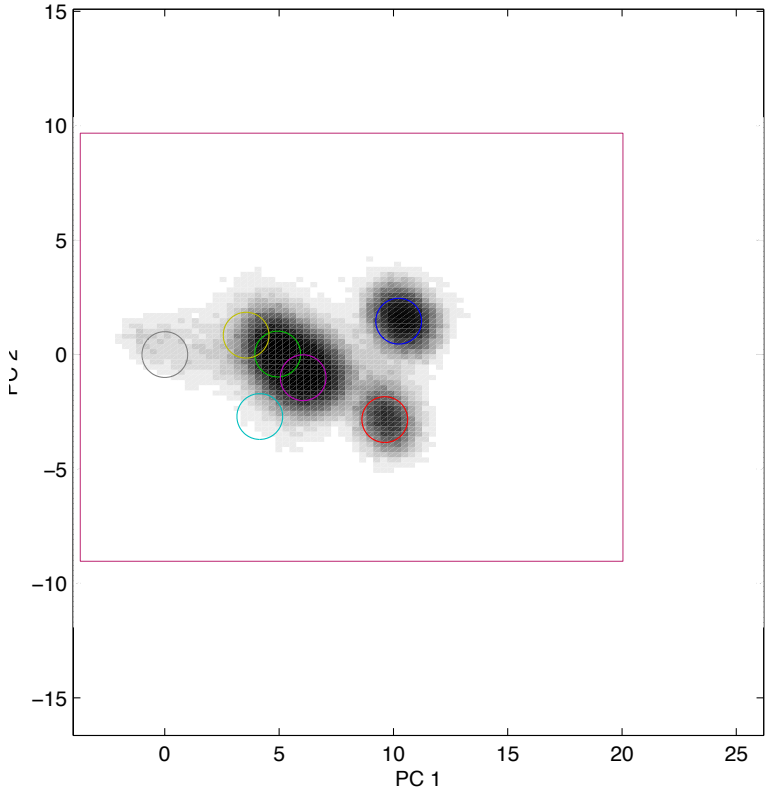




Mixture of Gaussians

- Fit gaussian "mountains" on each cluster





Math

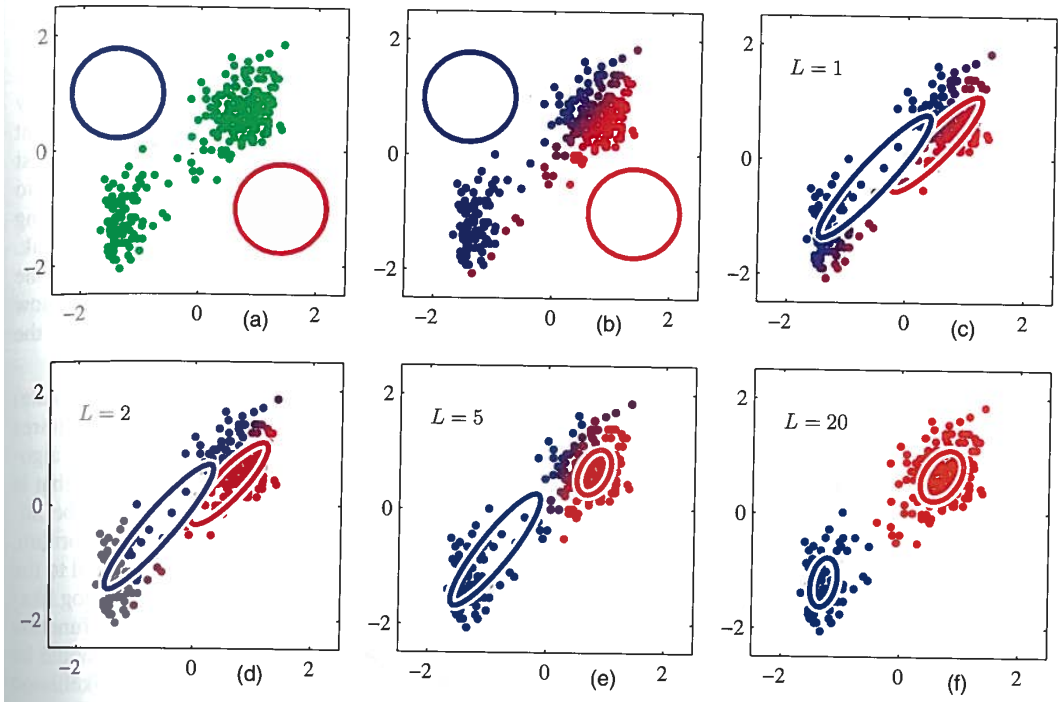
- Described in text as a probability distribution
- Have K gaussians in the model with their own mean and covariance

$$p(x) = \sum_{k=1}^{K} \pi_k N(\vec{x} \mid \mu_k, \Sigma_k)$$

- "Mixing component" π_k
- We interpret individual gaussians means as unit centroids

Fitting the Gaussians

- Usual method is “Expectation Maximization”
- Iterative process similar to K Means
- Based on “maximum likelihood” (next class)



Matlab Bit

At Decoding Yet?

- "Decoding" is extracting information from neural signals
  - How should the hand move?
  - Is there a seizure occuring?
  - What letter do you want to type?
- Finished pre-processing our data
- Still need to pick **Feature Set** to decode
  - Power in one or more frequency bands
  - Mean firing rate of neurons

Code

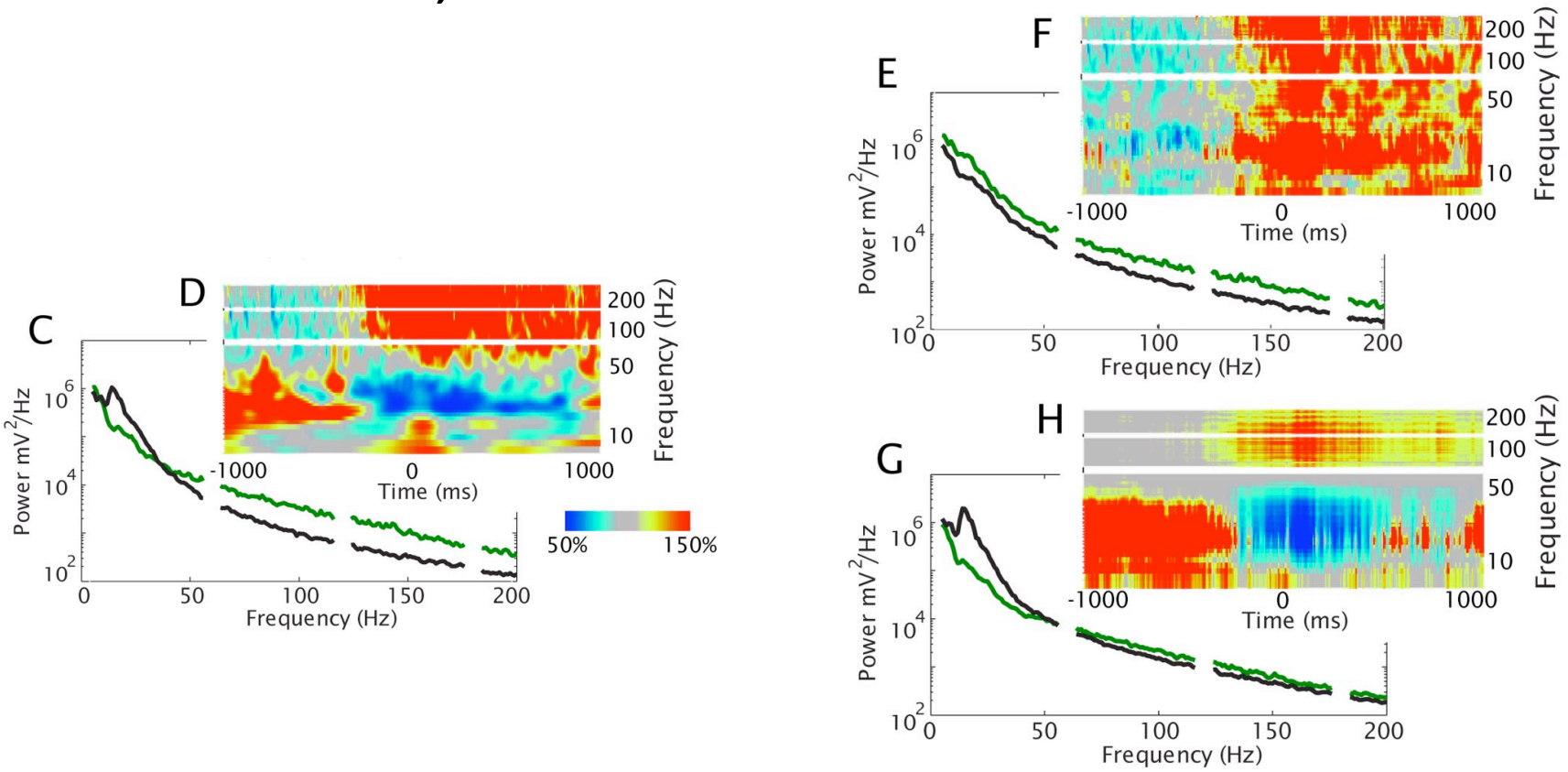
```
%Mixture of gaussians
clusternum=3
options = statset('Display','final');
obj = gmdistribution.fit(score(:,1:2),clusternum,'Options',options);
scatter(score(1:10000,1),score(1:10000,2),10,'.');
hold on
h = ezcontour(@(x,y)pdf(obj,[x y]),[-15 15],[-15 15]);
scatter(obj.mu(:,1),obj.mu(:,2),1000,'ro','LineWidth',2)

close
obj = gmdistribution.fit(score(:,1:4),clusternum,'Options',options);
scatter(score(1:10000,1),score(1:10000,2),10,'.');
hold on
scatter(obj.mu(:,1),obj.mu(:,2),1000,'ro','LineWidth',2)

clusternum=8;
close
obj = gmdistribution.fit(score(:,1:2),clusternum,'Options',options);
scatter(score(1:10000,1),score(1:10000,2),10,'.');
hold on
scatter(obj.mu(:,1),obj.mu(:,2),1000,'ro','LineWidth',2)
```

Feature Selection

- Most dramatic improves in algorithm performance don't come from fancier algorithms, they come from picking better features to feed into your algorithm

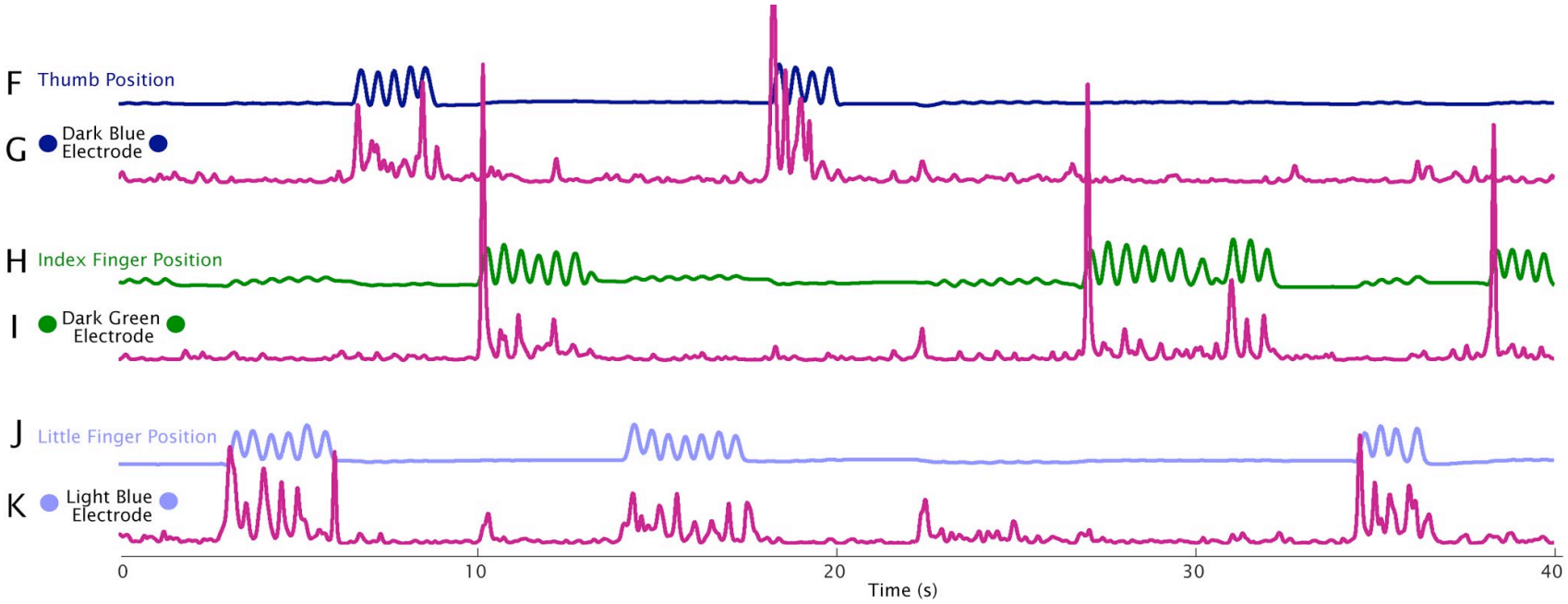
Principle Components with ECoG

- Miller et al., 2009



Principle Components with ECoG

- Miller et al., 2009



Feature Selection

- Why not try every vaguely possible feature?
- More prone to overfitting

Overfitting

- With enough parameters, you can fit every point of your training dataset perfectly

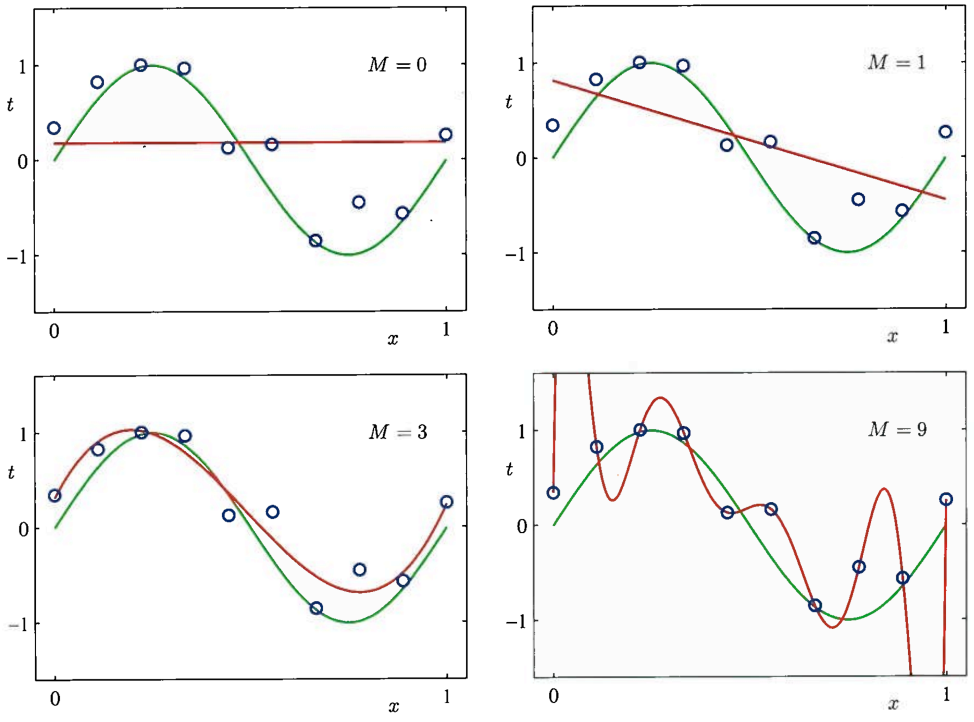


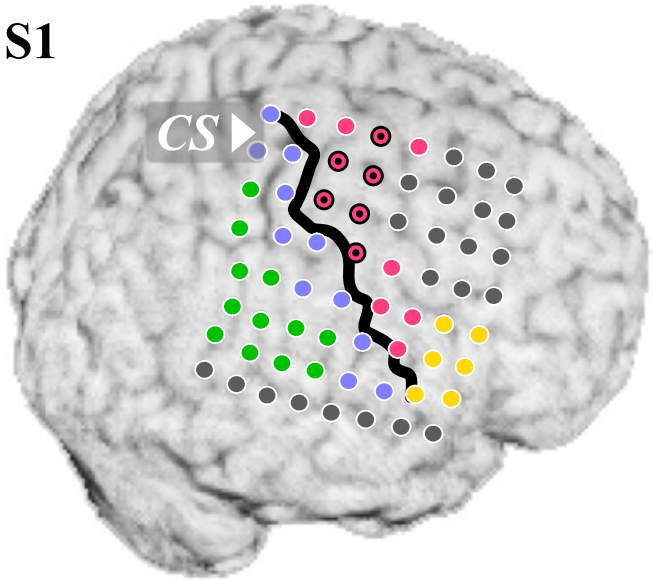
Figure 1.4 Plots of polynomials having various orders M, shown as red curves, fitted to the data set shown in Figure 1.2.

Overfitting and Feature Selection

- Machine learning
  - Training set: data you use to train an algorithm to extract information
  - Test set: data that you hold out during that process to validate your model
  - Approximating novel data in real life
- If you have enough potential features, some were correlated with your signals by chance
  - Creates amazing performance on training data
  - You end up fooling yourself when you forgot not to look at any test data during feature selection

Example Selection Process

- Pistohl et al., 2011
- Decoding pinch vs power grip
- Pick relevant channels
- Pick frequency bands



Example Selection Process

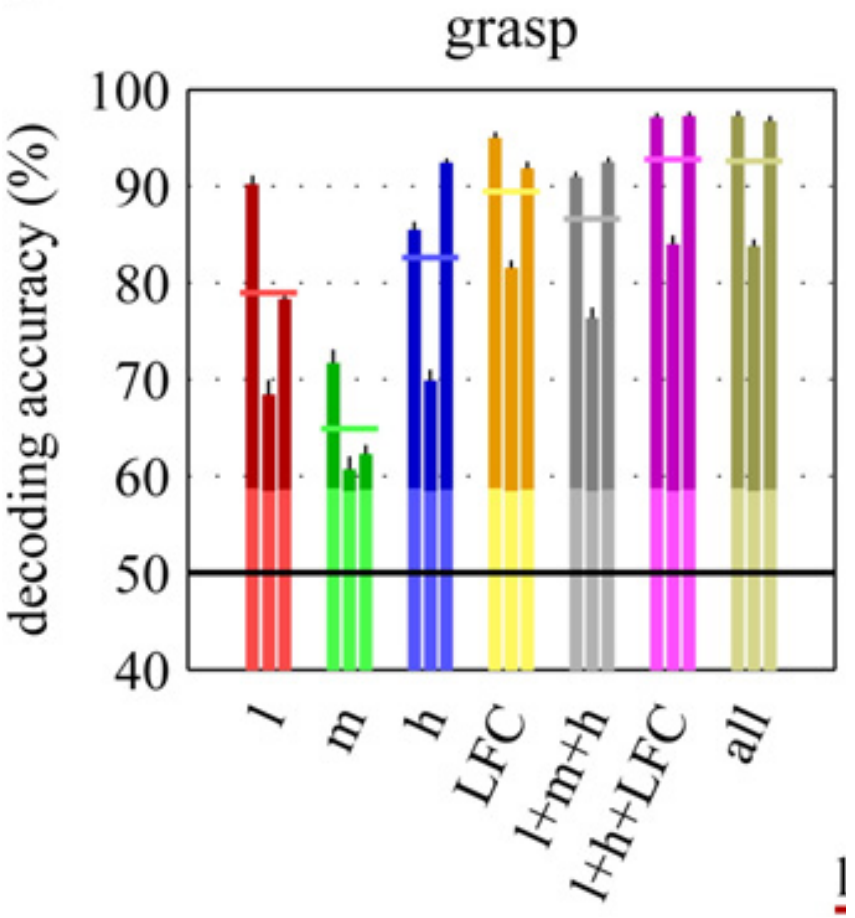
- Grabbed four numbers per channel based on initial inspection of spectrograms
  - Raw low pass filtered voltage
  - 2-6 Hz Low frequency band (delta)
  - 14-46 Hz Mid frequency band (beta)
  - 54-114 Hz High frequency bad (gamma)
  - Used all the channels on the motor side of central sulcus
  - Used cross validation

Cross validation

- Because datasets are small
- "10 fold cross validation" means:
  - Train on 90% of data, test on 10%
  - Repeat for 10 sets of 10%
  - Average the results
- Single-trial cross validation
  - Hold out only one trial for testing, train on everything else
  - Repeat for each trial

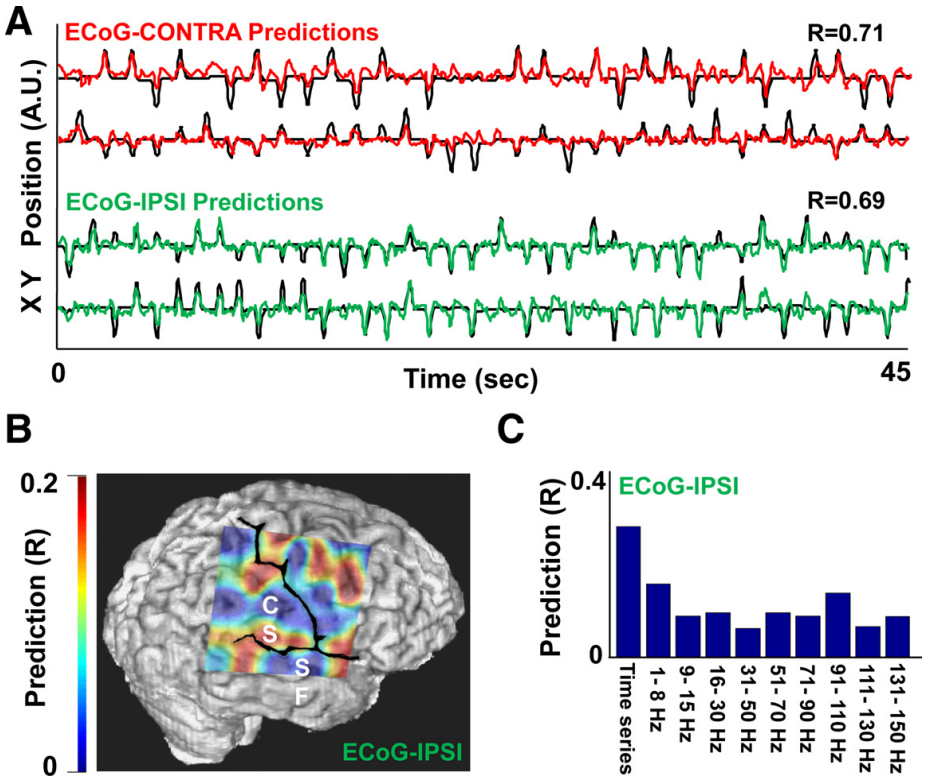
Pistohl et al., Results

- How well did each feature predict 1 of 2 grasp types?



Local Motor Potential

- Every reason to believe in evoked motor potentials, but fraught with peril



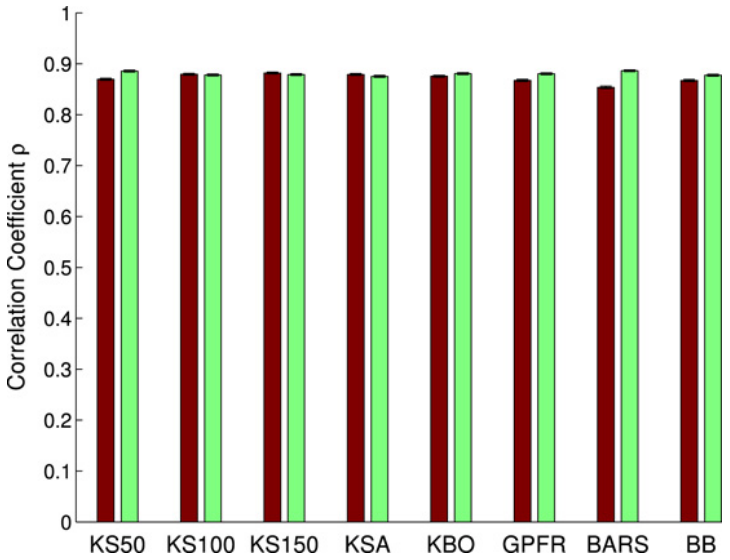
(Ganguly et al. 2009)

Final Word of Caution

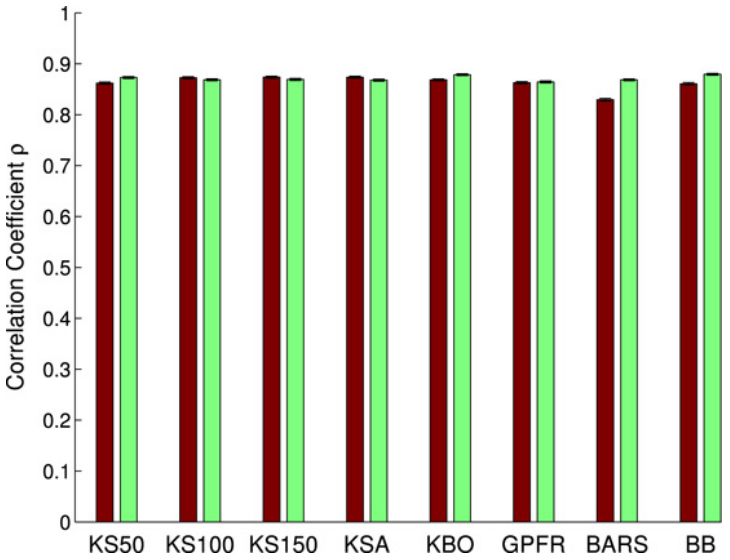
- Bonferroni, Benjamini-Hochberg corrections
- Common to check if many features significantly correlate with what you're trying to extract
- One t-test, significant at $p<0.05$
- 1000 tests?
  - New $p < 0.05/1000$
- Rank p values from smallest to largest, find largest k such that $p(k) < 0.05*k/1000$
  - Gives you percent of null hypotheses rejected

Spiking Feature Sets

- Estimate firing rate from spiking events
- This is hard to screw up
- Convolving with 50 ms wide Gaussian is fine (and even binning at 50 increments works)



(c) Horizontal hand position, L2006B.



(d) Vertical hand position, L2006B.

(Cunningham et al., 2009)

Spiking Bands

- Stark and Abeles, 2007
- SU = Sorted Single Units
- MSP = Threshold Crossing events
- LFP < 300 Hz
- MUA = Band pass filter between 300-6000 Hz