Data mining

What do you do with LOTS of data?

General guidelines:

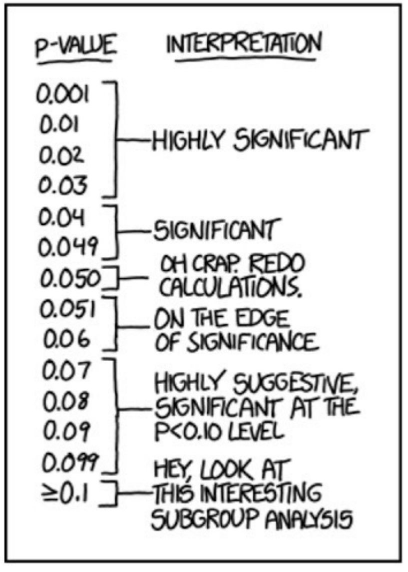
1. *Look* at your data
    - perform any/all preprocessing steps
    - use many visualizations, especially of raw, unaveraged data
        - scatterplots, heatplots, histograms
    - look at distributions, identify outliers
    - test hypotheses with encoding & decoding approaches
        (confusion matrices!)
    - look for trends and patterns (*unsupervised* methods)
        - dimensionality reduction, clustering

What do you do with LOTS of data?

General guidelines:

1. *Look* at your data
2. But don't look at *all* of your data

Inflation bias, a.k.a. "p-hacking"



https://imgs.xkcd.com/comics/p_values.png

What do you do with LOTS of data?

General guidelines:

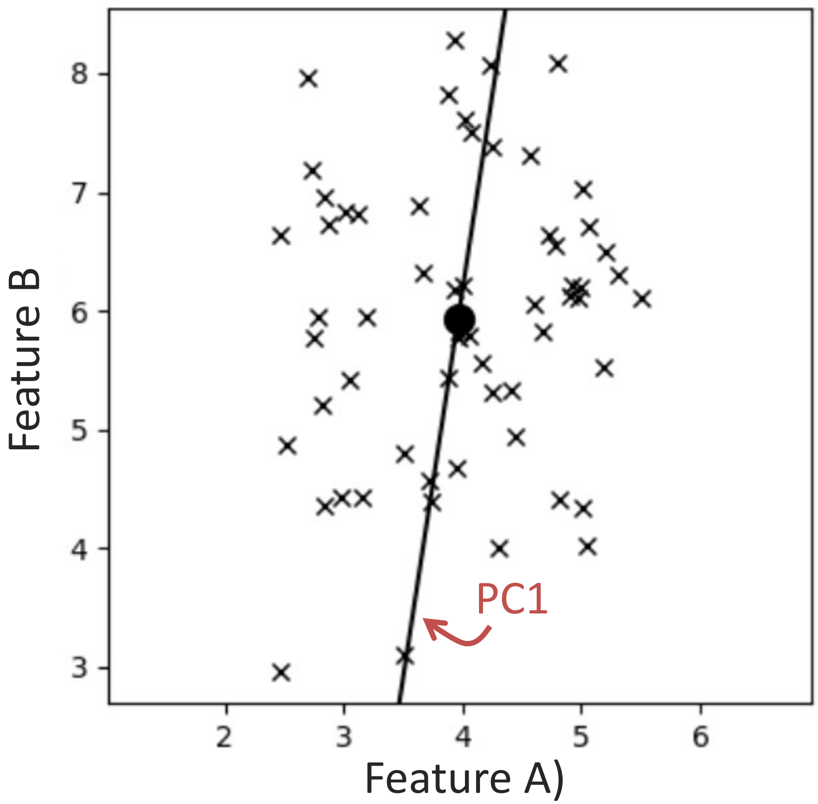
1. *Look* at your data
2. But don't look at *all* of your data

Best practices to avoid p-hacking

1. Create your own replication study by mining half your data and holding half out for test
2. Use visualizations to see patterns, do stats only at the end

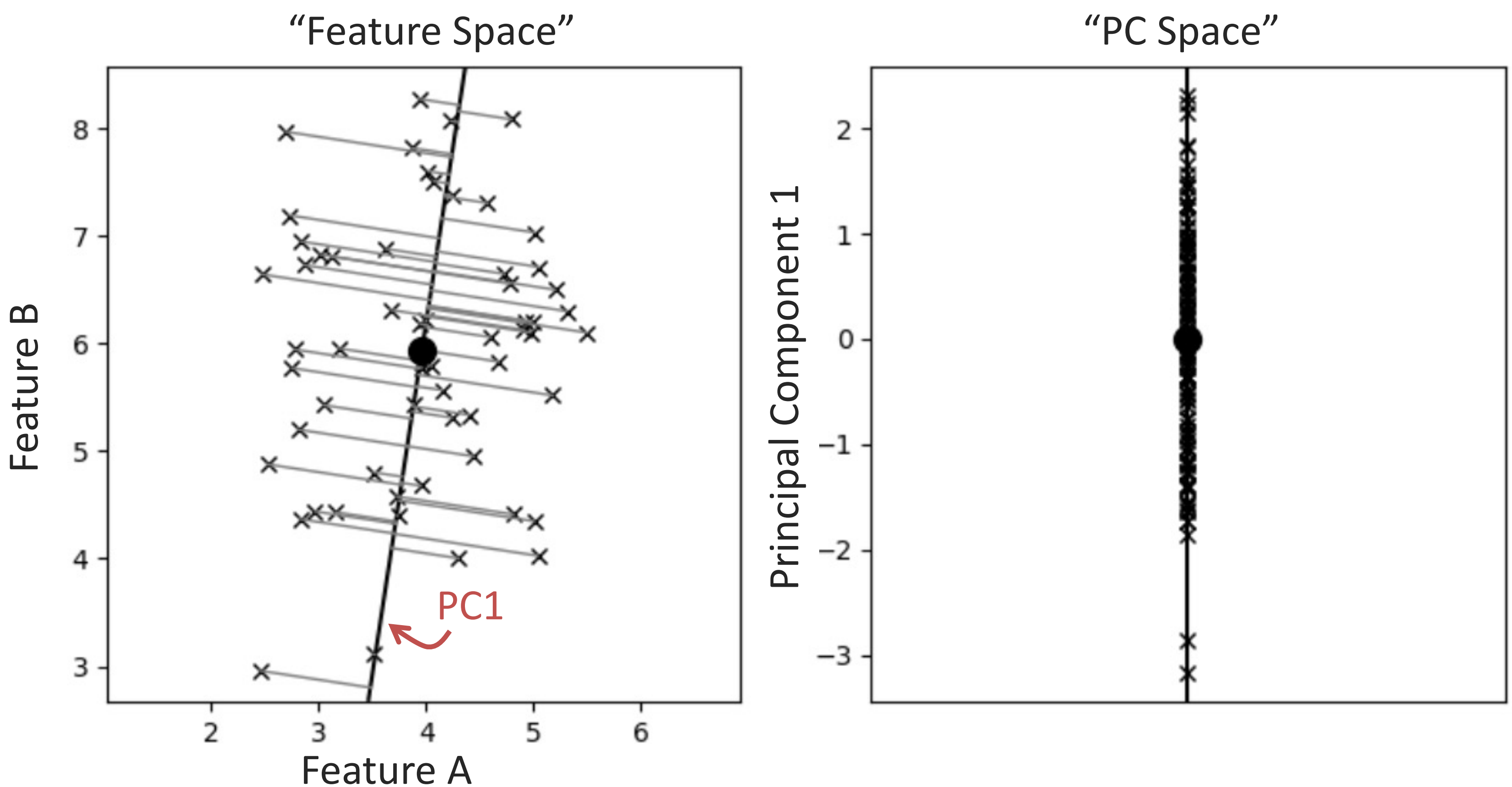
Finding patterns in data:
Dimensionality reduction

Principal component analysis (PCA)



Finding patterns in data: Dimensionality reduction

Principal component analysis (PCA)



PCA demo

Let’s create some fake data to see how PCA works:

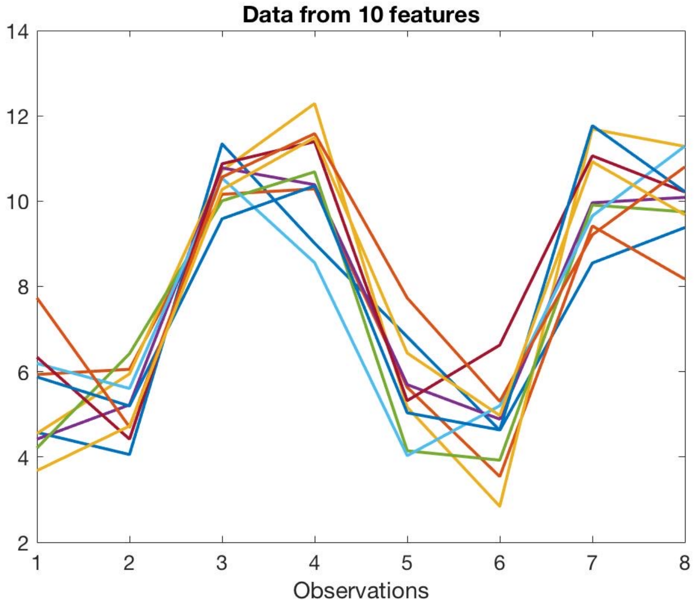
10 features (e.g. neurons)
8 observations (e.g. a measure in 8 different conditions)
All features reflect an underlying pattern:

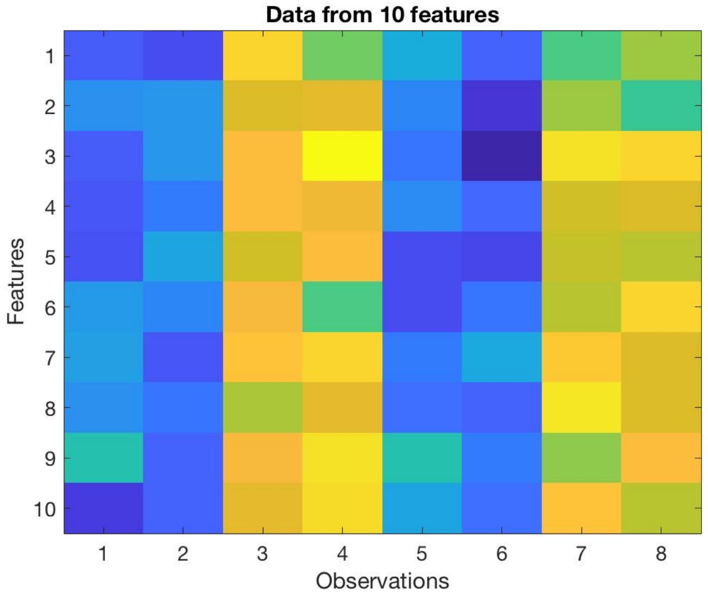
```
% Let's make some features that follow an underlying patterns
% We'll initialize the pattern as responses observed to 8 differnt
% conditions (of in 8 different observations)
pattern1 = [5 5 10 10 5 5 10 10];
var = 1;

% Next we'll make a population of 10 features that follow this pattern with
% some noise
pop1 = [];
% let's make 10 features per subpopulation
for k=1:10
    for j=1:8 %there are 8 observations
        noise = normrnd(0,var); %this will add random noise, drawn from a gaussian centered at 0 with standard deviation = var,
        pop1(k,j) = pattern1(j)+noise;
    end
end
```

PCA demo

Let's create some fake data to see how PCA works:

10 features (e.g. neurons)
8 observations (e.g. a measure in 8 different conditions)
All features reflect an underlying pattern:





PCA demo

Let's create some fake data to see how PCA works:

10 features (e.g. neurons)
8 observations (e.g. a measure in 8 different conditions)
All features reflect an underlying pattern:

In matlab:

```
% Now let's use PCA to see how it recovers patterns
[coeff,pcs,~,~,explained]=pca(pop1'); % note that the function that computes PCA has to take input matrices in the correct orientat
```

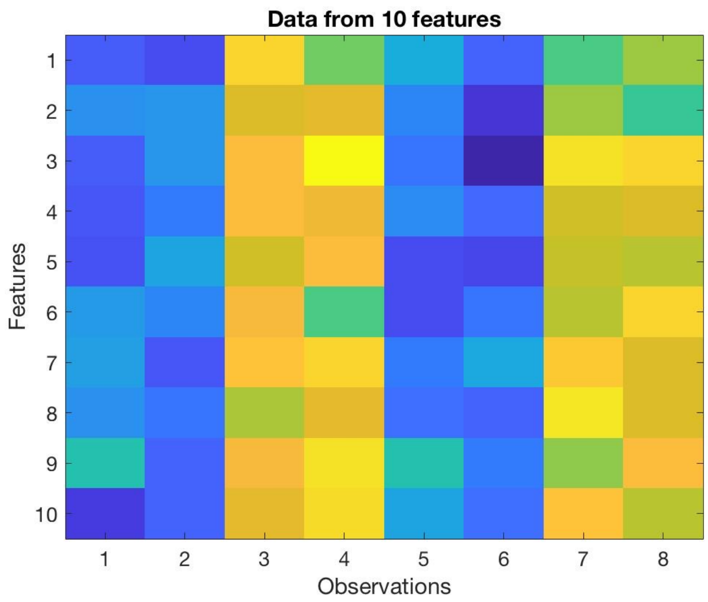
In R:

```
pop1_pca <- prcomp(pop1, center = FALSE) # note by default matlab "centers" data R does not
# so scale of values differs between matlab and R
```

PCA demo

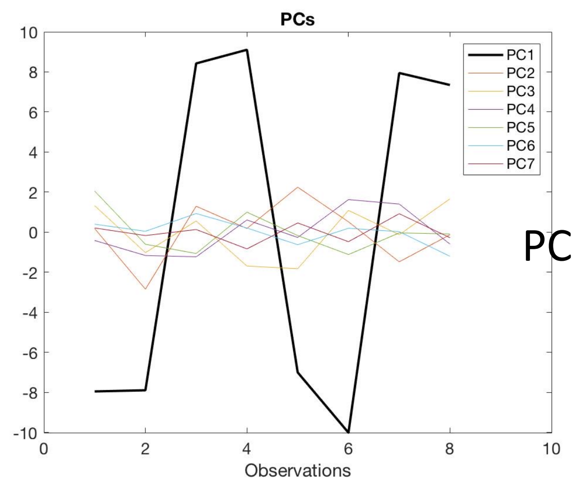
Let's create some fake data to see how PCA works:

10 features (e.g. neurons)
8 observations (e.g. a measure in 8 different conditions)
All features reflect an underlying pattern:



PCA can give you:

- **PCs**
- % variance explained by each PC
- feature loadings or "weights" for each PC

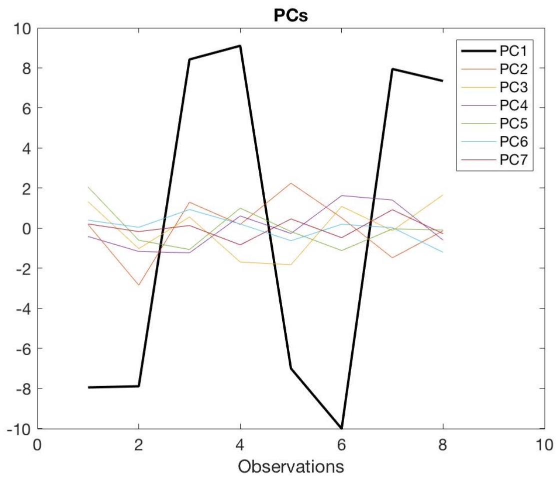


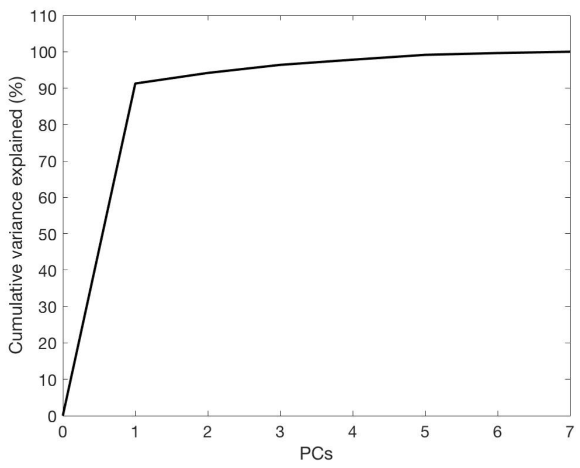
PC1

PCA demo

PCA can give you:

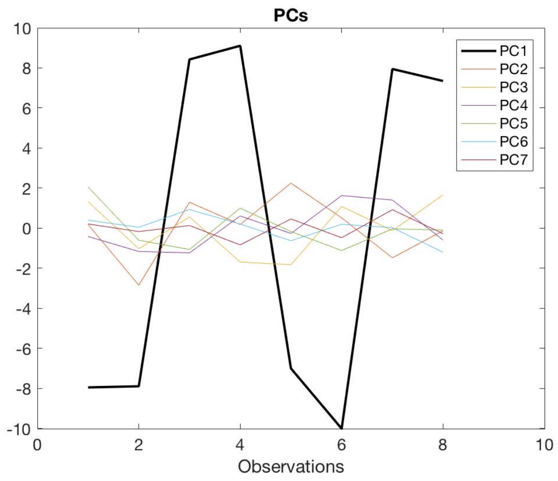
- PCs
- **% variance explained by each PC**
- feature loadings for each PC

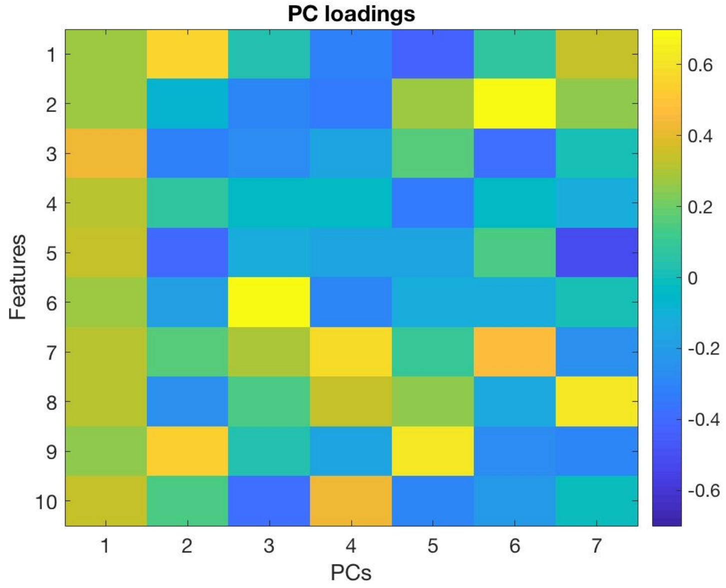




PCA demo

PCA can give you:
- PCs
- % variance explained by each PC
- **feature loadings for each PC**





- PC1 has high weights on each feature
- Other PCs weigh on only 1 or 2 features

PCA demo

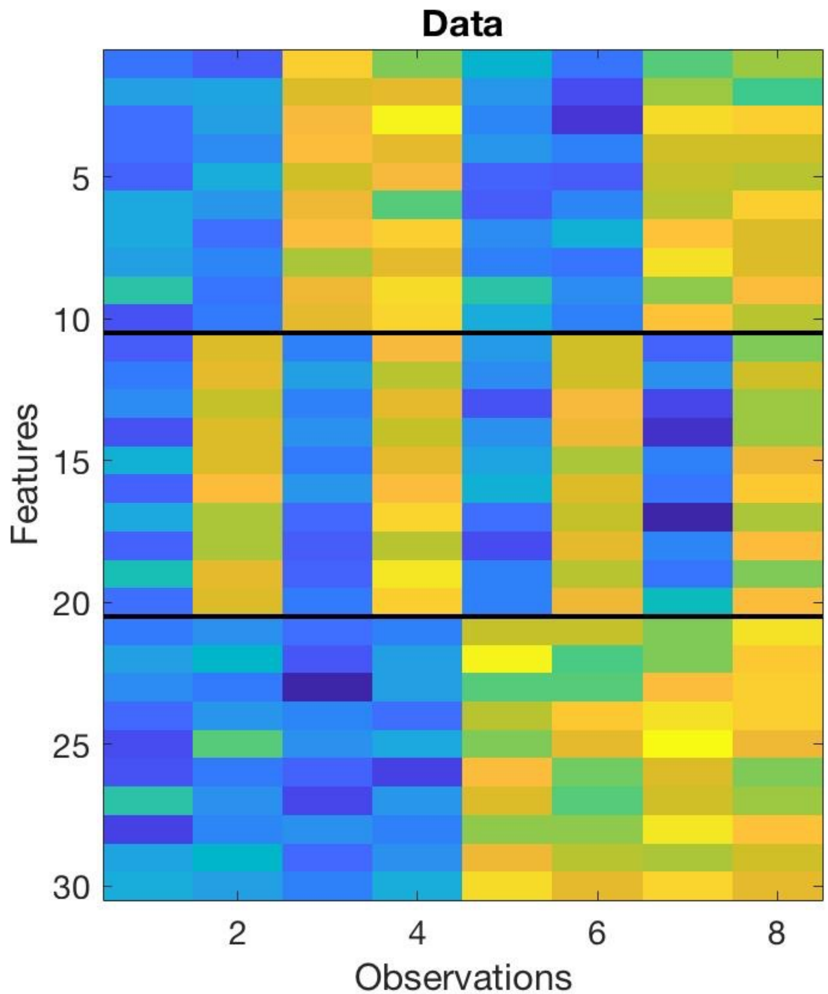
Let's add data….

30 features (e.g. neurons)
8 observations (e.g. measure in 8 conditions)
3 underlying patterns

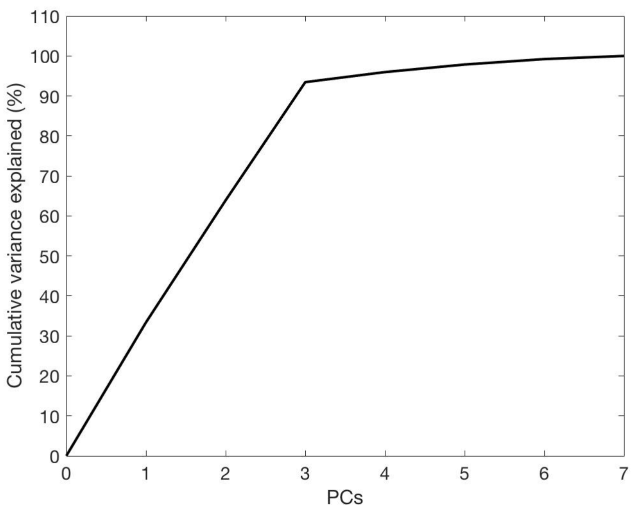
```
% Let's do it again with a more complex data set
% Additional patterns
pattern2 = [5 10 5 10 5 10 5 10];
pattern3 = [5 5 5 5 10 10 10 10];

% and create two more populations that follow different patterns
pop2 = []; pop3 = [];
for k = 1:10
    for j = 1:8
        noise = normrnd(0,var); % noise should be independent for this simulation
        pop2(k,j) = pattern2(j)+noise;
        noise = normrnd(0,var);
        pop3(k,j) = pattern3(j) + noise;
    end
end
pop = [pop1;pop2;pop3]; % Our full feature matrix is all of these subpopulations together
```

PCA demo

Let’s add data….

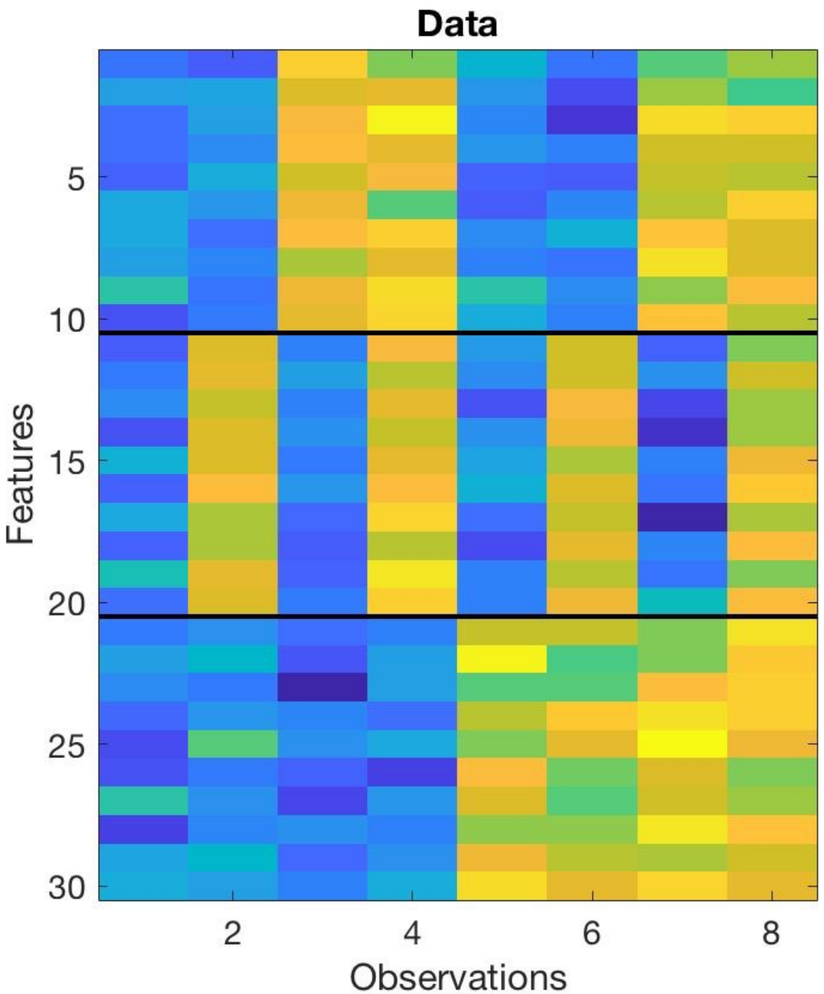


30 features (e.g. neurons)
8 observations (e.g. measure in 8 conditions)
3 underlying patterns

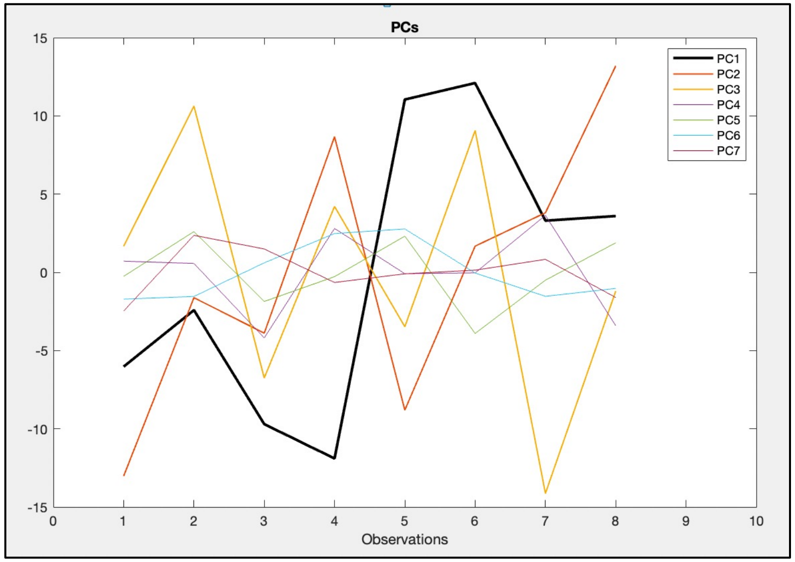


PCA demo

Let's add data….

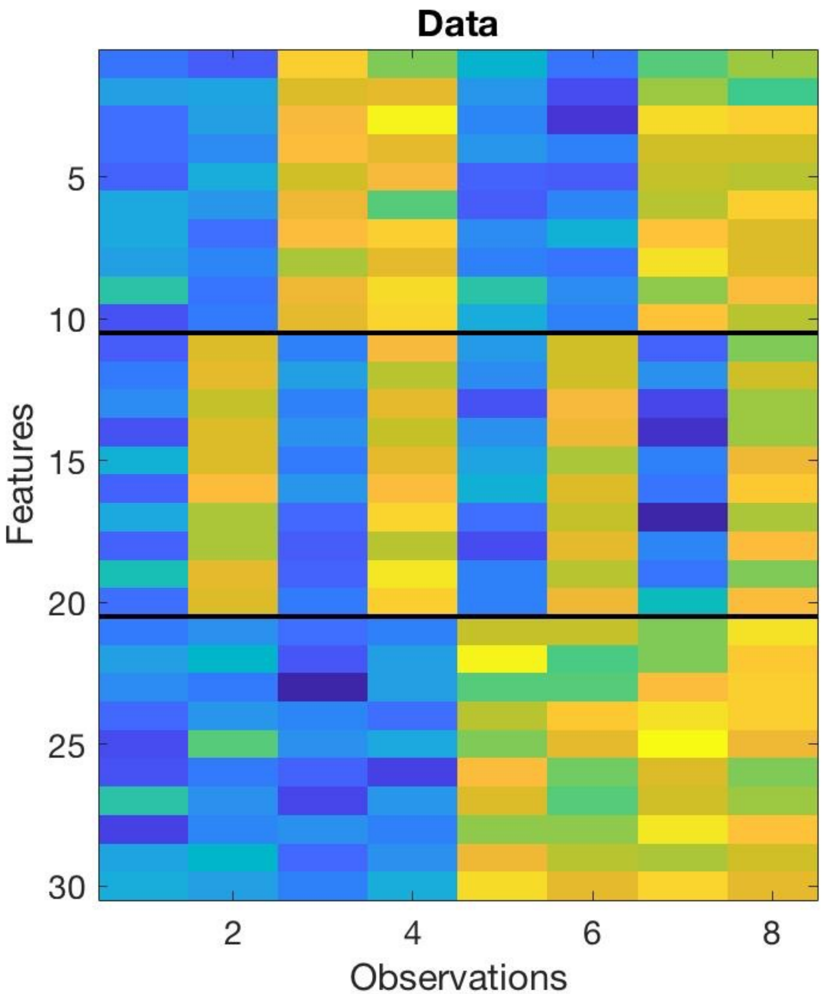


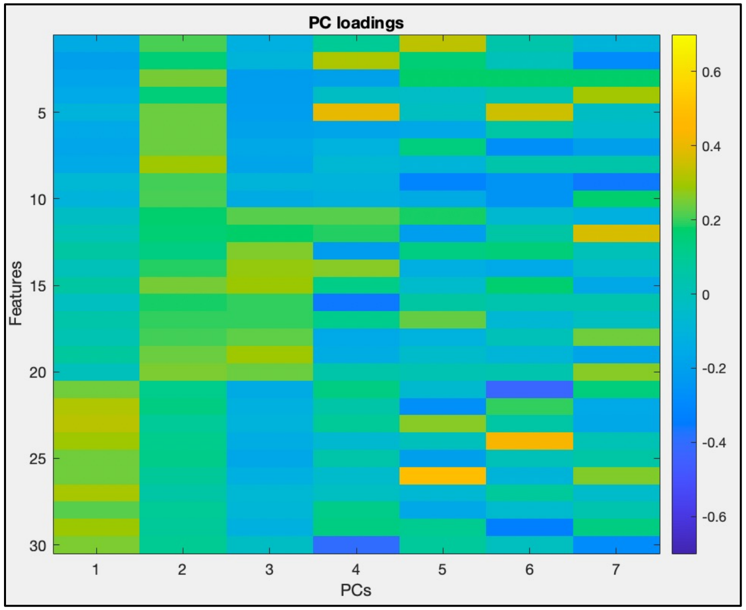
30 features (e.g. neurons)
8 observations (e.g. conditions
3 underlying patterns

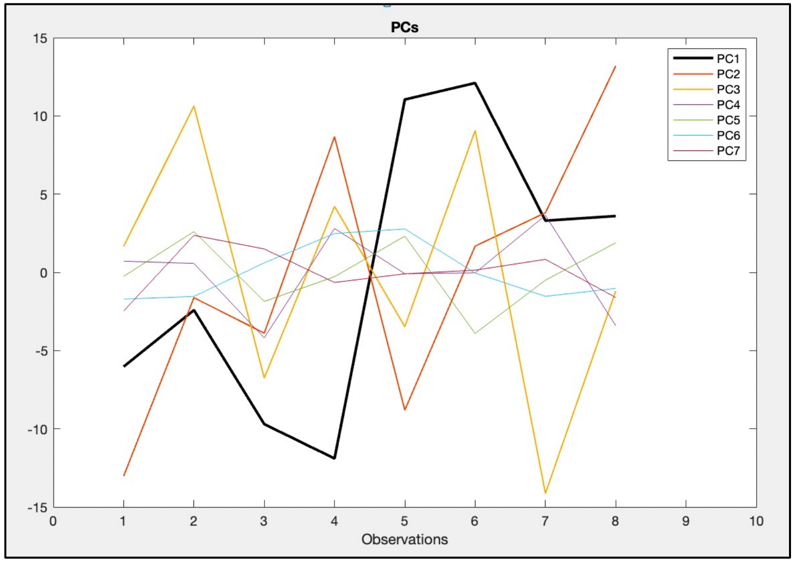


PCA demo

Let's add data….

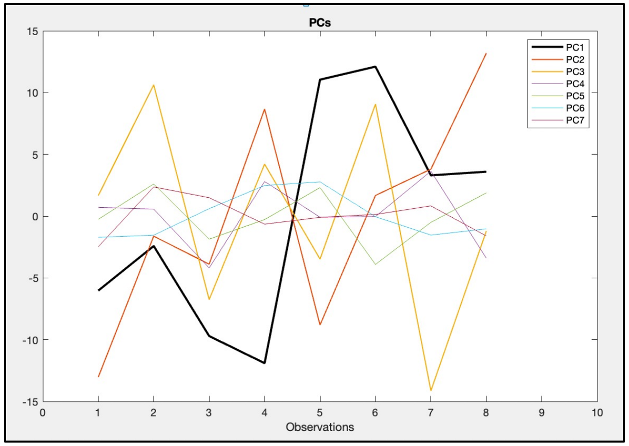


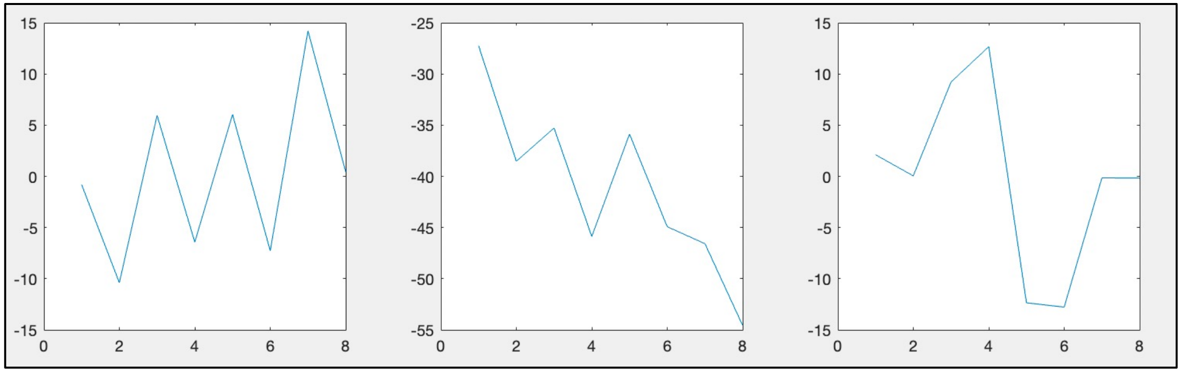




Finding patterns in data: Dimensionality reduction

Many algorithms
- PCA
- Singular value decomposition (SVD – closely related to PCA)
- Independent component analysis (ICA) - *similar to PCA, but finds components that have maximal statistical independence*

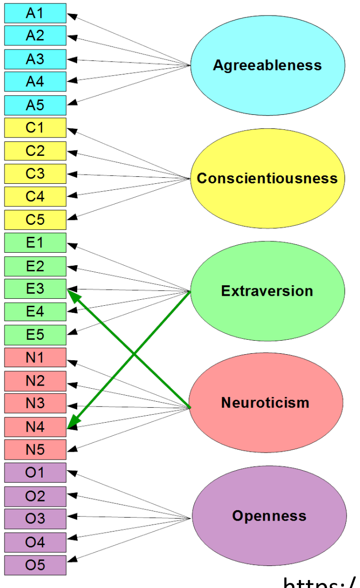




Finding patterns in data: Dimensionality reduction

Many algorithms

- PCA
- Singular value decomposition (SVD – closely related to PCA)
- Independent component analysis (ICA)
- Factor analysis – *reduces a large number of variables to a smaller number of ”factors” or latent variables by finding maximum common variance*

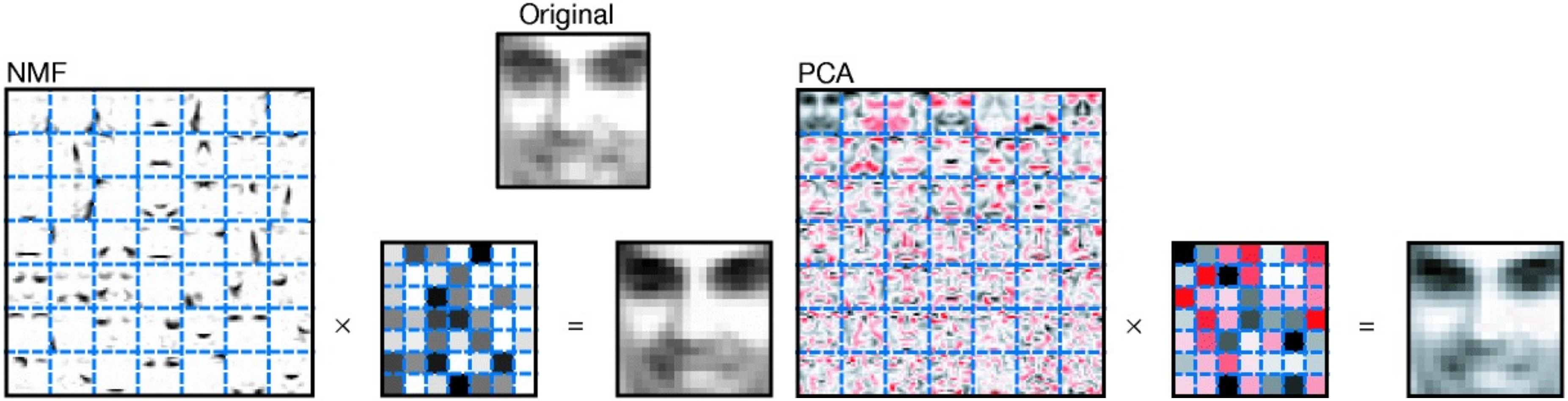


https://www.datacamp.com/blog/new-course-factor-analysis-in-r

Finding patterns in data: Dimensionality reduction

Many algorithms

- PCA
- Singular value decomposition (SVD – closely related to PCA)
- Independent component analysis (ICA)
- Factor analysis
- Non-negative matrix factorization (NMF) – *like PCA but constrains weights to be non-negative -> more interpretable in some instances*



Lee & Seung, 1999

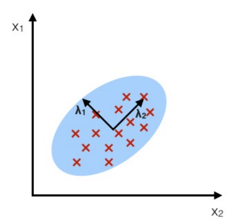
Finding patterns in data:
Dimensionality reduction

Many algorithms

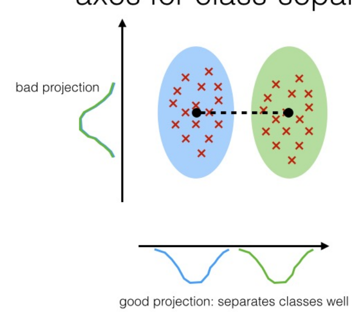
- PCA
- Singular value decomposition (SVD – closely related to PCA)
- Independent component analysis (ICA)
- Factor analysis
- Non-negative matrix factorization (NMF)
- demixed PCA (dPCA)
- Linear discriminant analysis (LDA)
- Bespoke statespace analyses

Methods that aim to find interpretable components (dPCA, LDA, bespoke statespace analyses)

PCA:
component axes that
maximize the variance



LDA:
maximizing the component
axes for class-separation

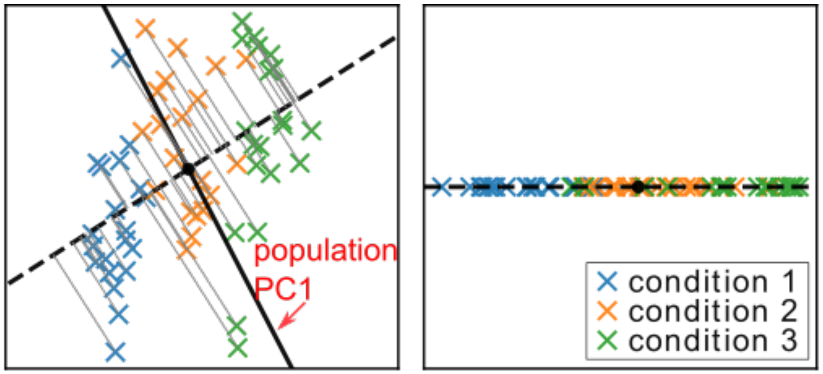


Finding patterns in data: Dimensionality reduction

Many algorithms

- PCA
- Singular value decomposition (SVD – closely related to PCA)
- Independent component analysis (ICA)
- Factor analysis
- Non-negative matrix factorization (NMF)
- demixed PCA (dPCA)
- Linear discriminant analysis (LDA)
- Bespoke statespace analyses

Methods that aim to find interpretable components



Homework #9

(first part)

HW9: Data mining

You have recorded pupil responses in a subject viewing different images.
The data are saved in *data.txt*, which includes 500 trials where each trial is 1200ms long

1. Plot the mean pupil response over all trials
2. Do PCA across trials (Hint: each PC should be1200 elements long, and there should be 500 of them). How much variance does the first PC account for? How many components account for >=90% of the variance?
3. Plot the first principal component.
4. Run k-means 10 times with k=2. For each run, plot the cluster centers you obtain.