CPSC 340:
Machine Learning and Data Mining

Sparse Matrix Factorization

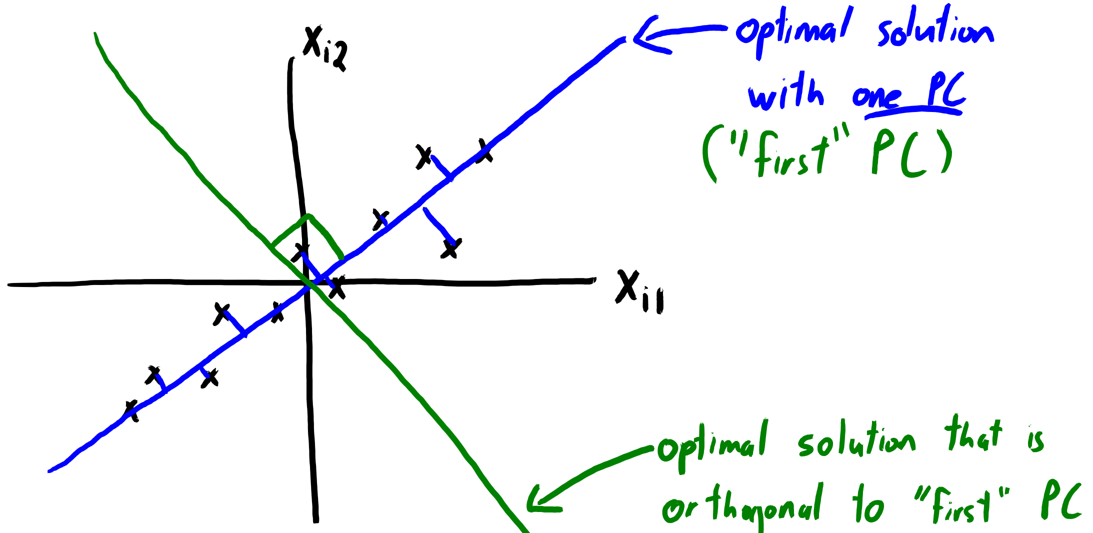
Fall 2018

Last Time: PCA with Orthogonal/Sequential Basis

- When k = 1, PCA has a scaling problem.
- When k > 1, have scaling, rotation, and label switching.
  - Standard fix: use normalized orthogonal rows W_c of 'W'.

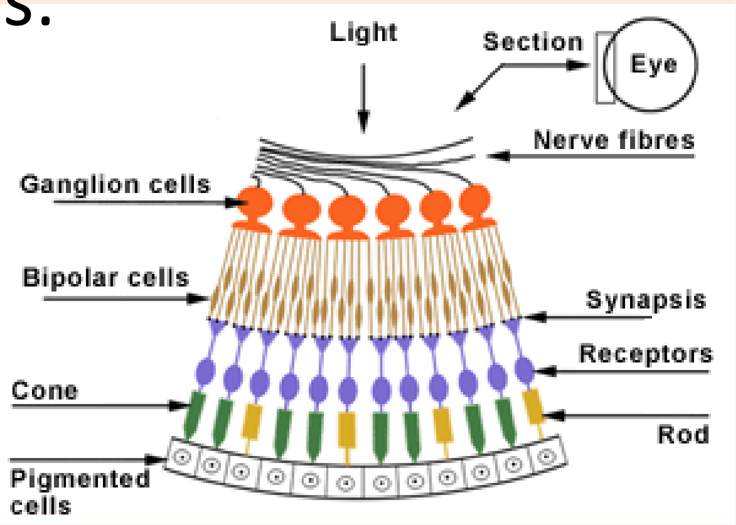
$$\|w_c\| = 1 \quad \text{and} \quad w_c^T w_{c'} = 0 \quad \text{for } c' \neq c$$

  - And fit the rows in order:
    - First row "explains the most variance" or "reduces error the most".

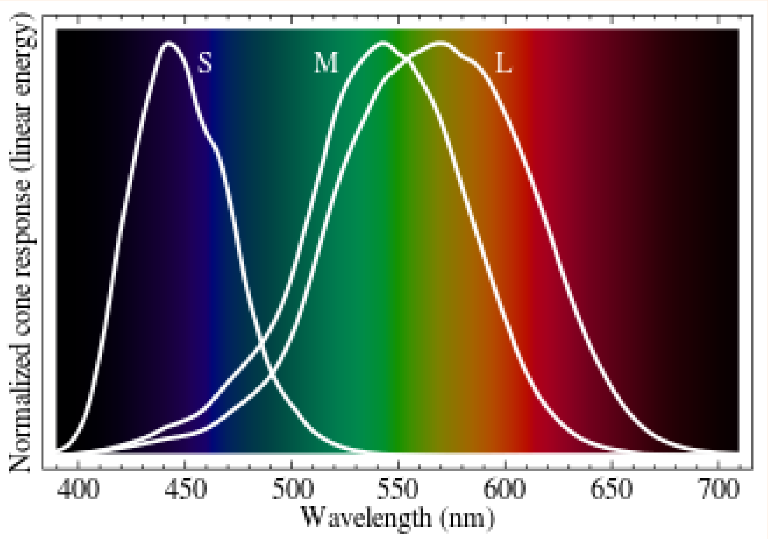


Colour Opponency in the Human Eye

- Classic model of the eye is with 4 photoreceptors:
  - Rods (more sensitive to brightness).
  - L-Cones (most sensitive to red).
  - M-Cones (most sensitive to green).
  - S-Cones (most sensitive to blue).



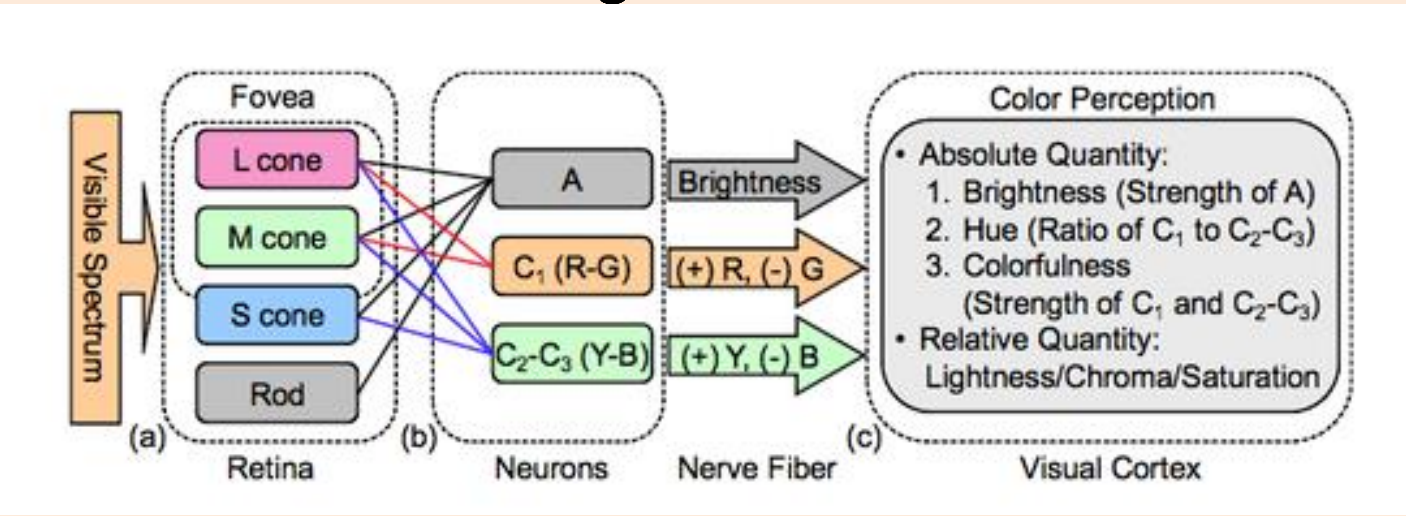
- Two problems with this system:
  - Not orthogonal.
    - High correlation in particular between red/green.
  - We have 4 receptors for 3 colours.

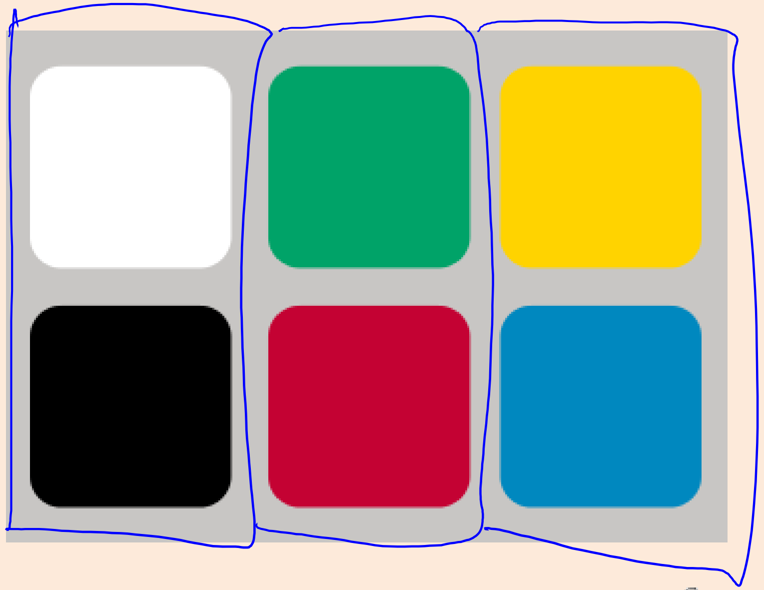


http://oneminuteastronomer.com/astro-course-day-5/
https://en.wikipedia.org/wiki/Color_visio

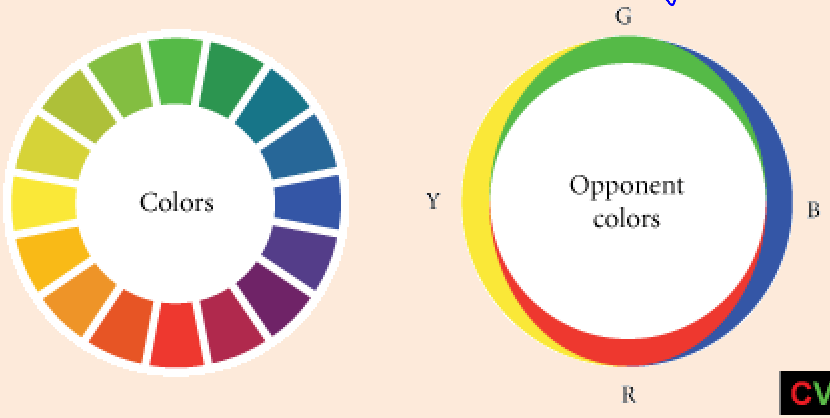
Colour Opponency in the Human Eye

- Bipolar and ganglion cells seem to code using "opponent colors":
  - 3-variable orthogonal basis:





- This is similar to PCA (d = 4, k = 3).



http://oneminuteastronomer.com/astro-course-day-5/
https://en.wikipedia.org/wiki/Color_visio
http://5sensesnews.blogspot.ca/

Colour Opponency Representation

For this pixel, eye gets 4 signals

Can represent 4 original values with these 3 z_i values and matrix 'W'

$\simeq w_1$

First row of W (First PC)

Analogous to means in k-means.

brightness

$+w_2$

Second row (4x1)

red/green

$+w_3$

Third row (4x1)

blue/yellow

Choosing ‘k’ by “Variance Explained”

- Common to choose ‘k’ based on variance of the x_{ij}.

$$\underbrace{Var(x_{ij}) = E[(x_{ij}}_{\text{definition of variance}} - \underbrace{\mu_{ij}}_{\text{assumed to be zero}})^2] = \underbrace{E[x_{ij}^2] = \frac{1}{nd}}_{\text{definition of expectation}} \underbrace{\sum_{i=1}^{n}\sum_{j=1}^{d} x_{ij}^2 = \frac{1}{nd}\|X\|_F^2}_{\text{Frobenius norm}}$$

  - For a given ‘k’ we compute (variance of errors)/(variance of x_{ij}):

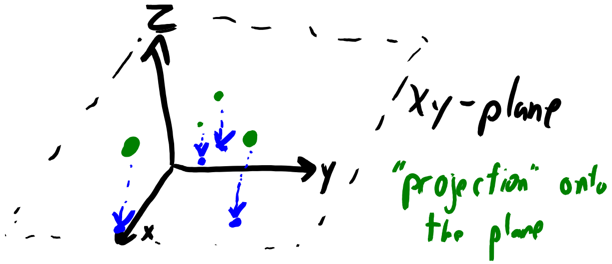
$$\frac{\|ZW - X\|_F^2}{\|X\|_F^2}$$

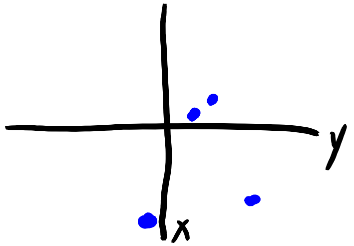
centered version

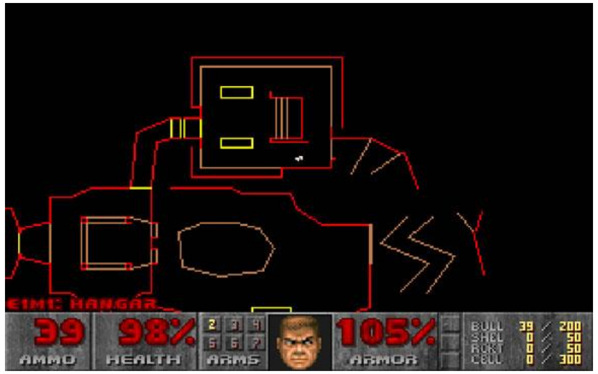
  - Gives a number between 0 (k=d) and 1 (k=0), giving “variance remaining”.
    - If you want to “explain 90% of variance”, choose smallest ‘k’ where ratio is < 0.10.

"Variance Explained" in the Doom Map

- Recall the Doom latent-factor model (where map ignores height):







- Interpretation of "variance remaining" formula:

$$\frac{\|ZW - X\|_F^2}{\|X\|_F^2}$$

$\|ZW - X\|_F^2$ ← Variance in z-dimension (variance in x- and y-dimensions fully captured by overhead map)

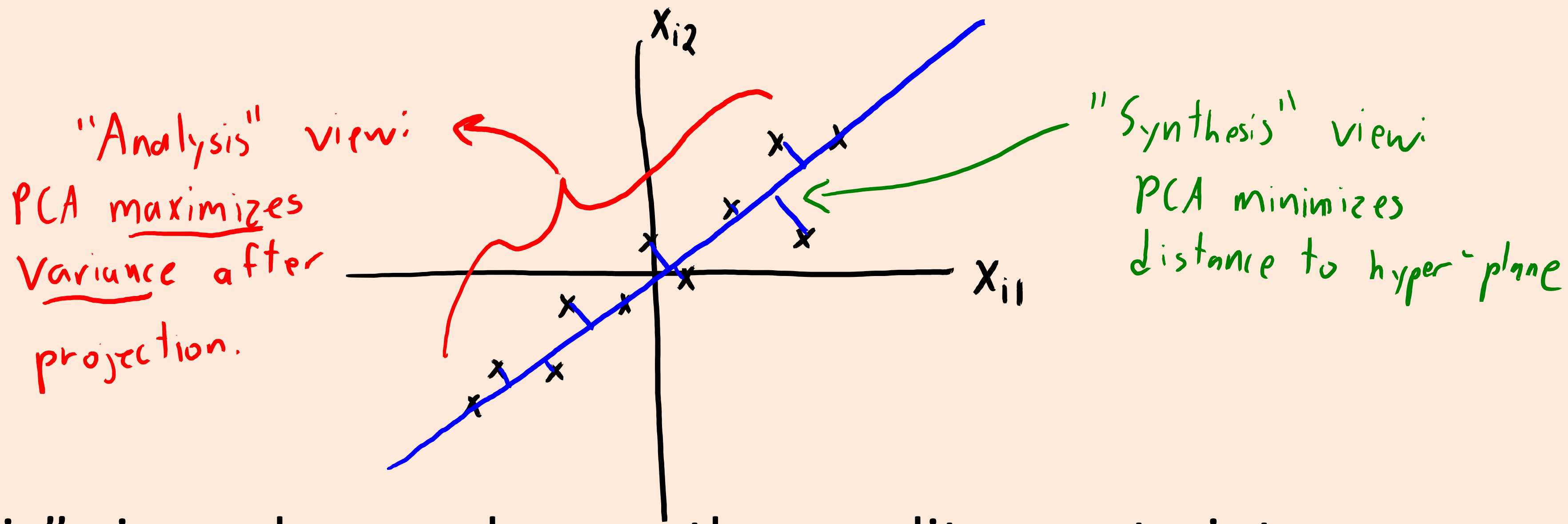
$\|X\|_F^2$ ← Variance of character in 3-dimensions

- If we had a 3D map the "variance remaining" would be 0.

https://en.wikipedia.org/wiki/Doom_(1993_video_game)
https://forum.minetest.net/viewtopic.php?f=5&t=9666

"Synthesis" View vs. "Analysis" View

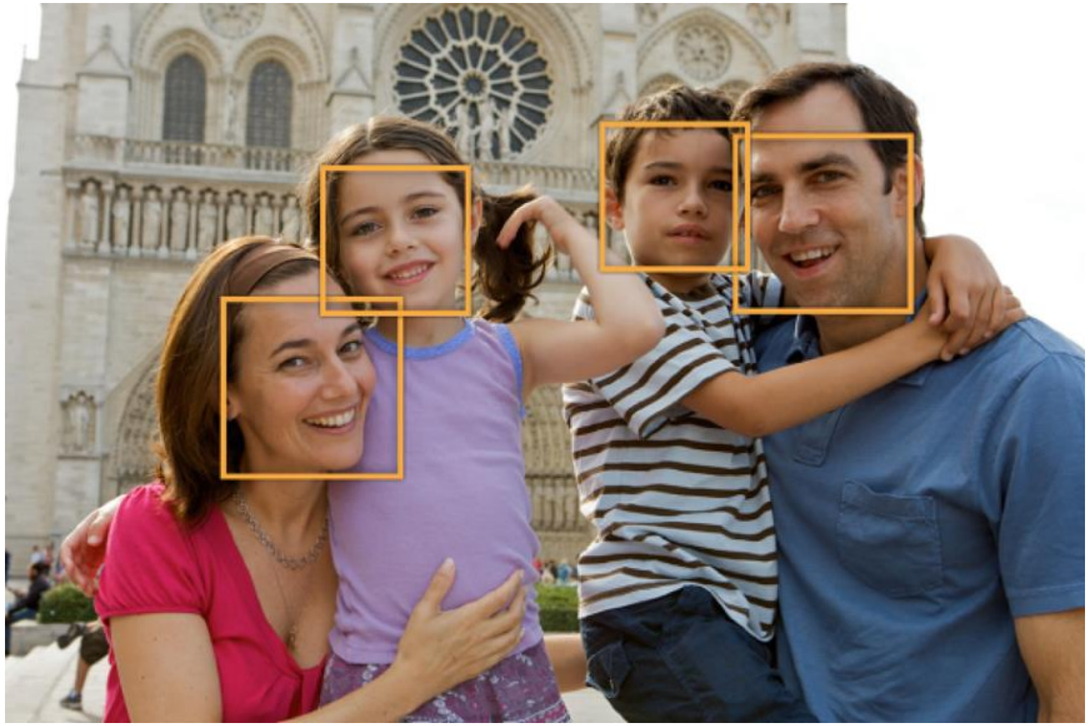
- We said that PCA finds hyper-plane minimizing distance to data x_i.
  - This is the "synthesis" view of PCA (connects to k-means and least squares).



- "Analysis" view when we have orthogonality constraints:
  - PCA finds hyper-plane maximizing variance in z_i space.
  - You pick W to "explain as much variance in the data" as possible.

Application: Face Detection

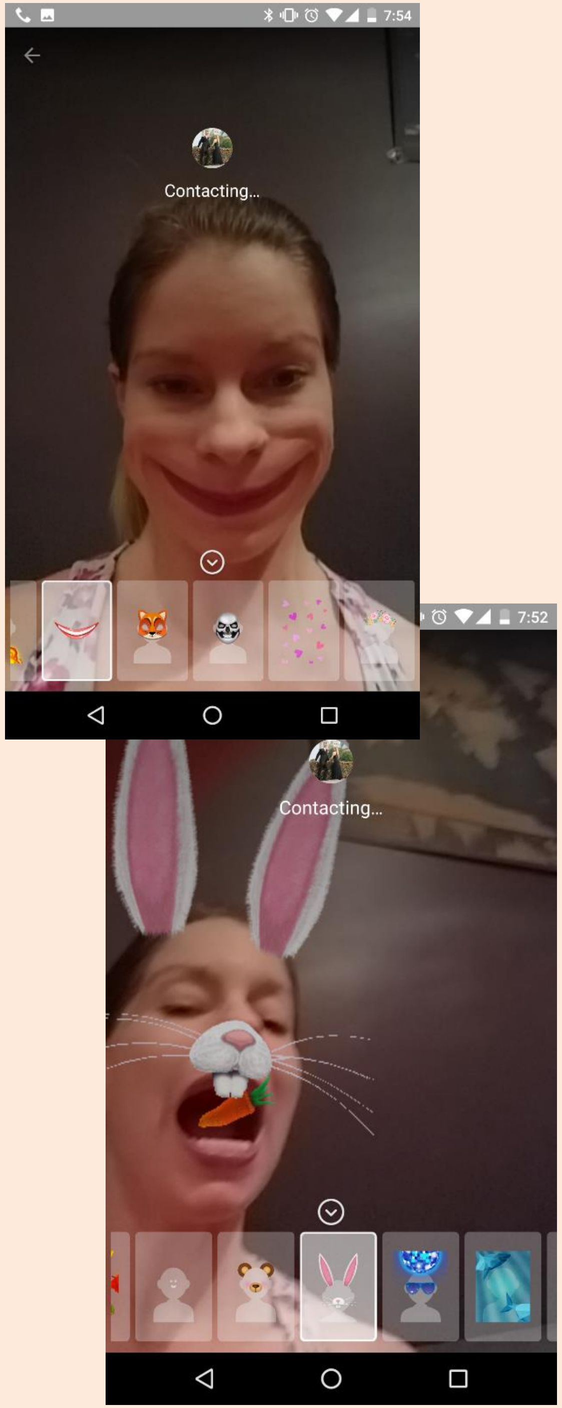
- Consider problem of face detection:

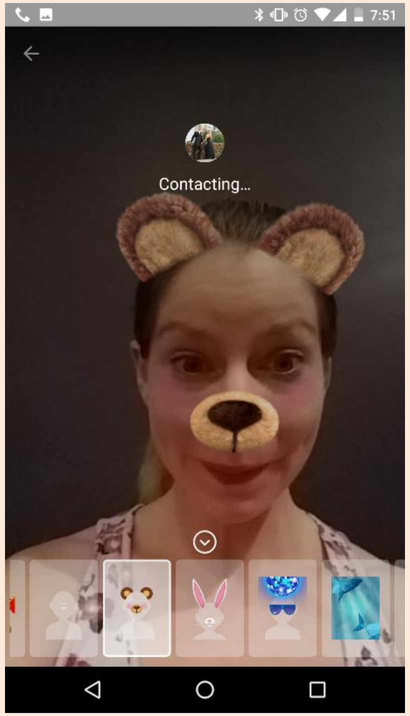


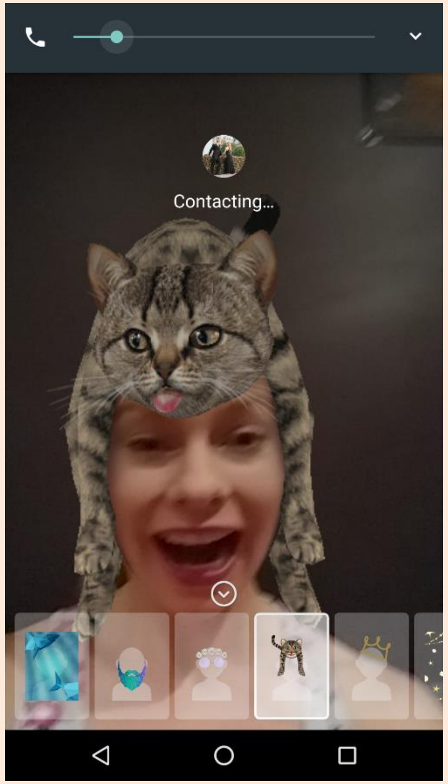
- Classic methods use “eigenfaces” as basis:
  - PCA applied to images of faces.

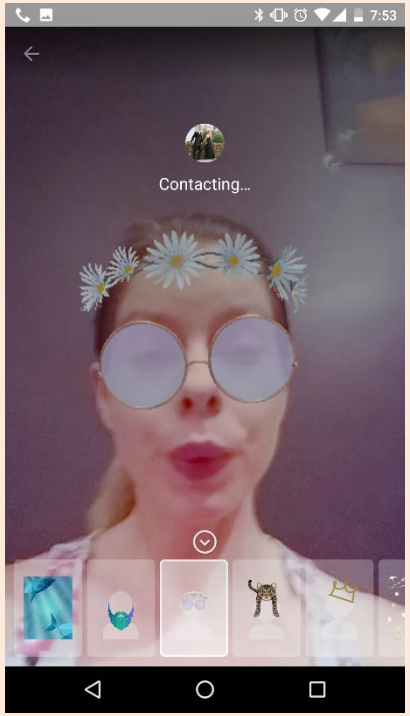
https://developer.apple.com/library/content/documentation/GraphicsImaging/Conceptual/CoreImaging/ci_detect_faces/ci_detect_faces.html

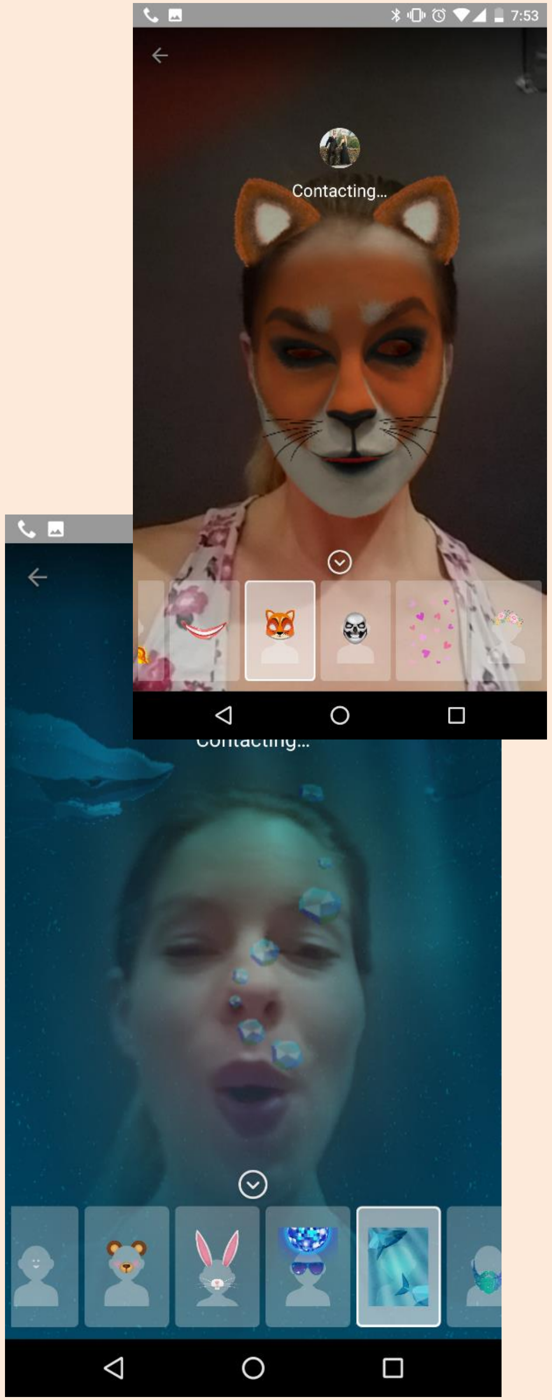
Application: Face Detection











Eigenfaces

- Collect a bunch of images of faces under different conditions:

Each row of X will be pixels in <u>one</u> image:

$X =$

If have 'n' images that are 'm' by 'm' then X is 'n' by m^2.

Eigenfaces

Compute <u>mean</u> μ_j of each column.

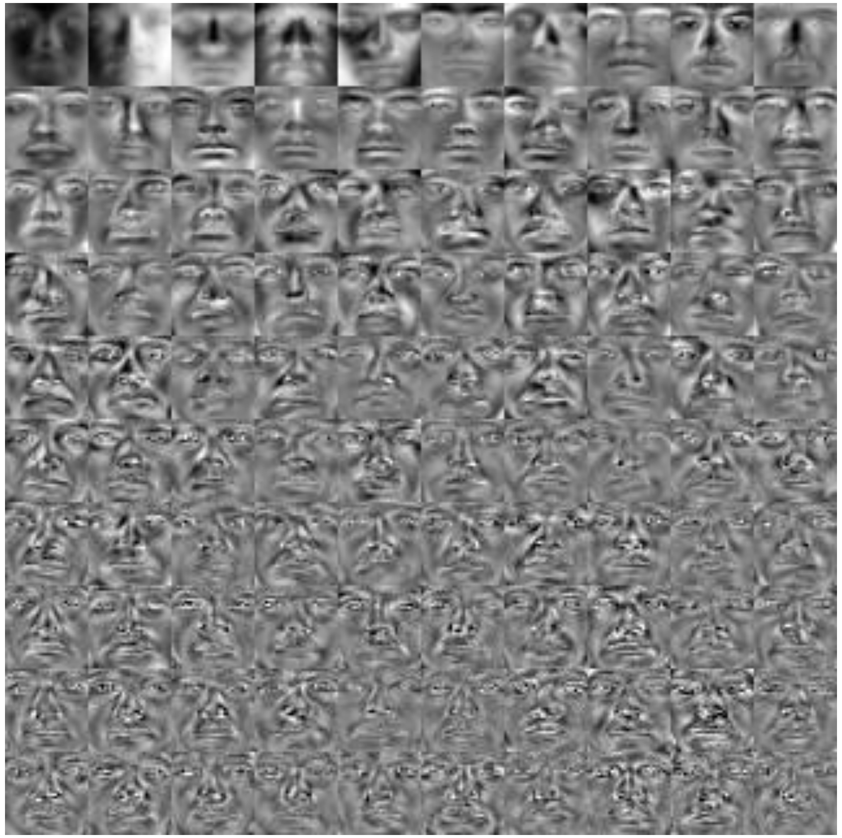
Replace each x_{ij} by $x_{ij} - \mu_j$

Each row of X will be pixels in <u>one</u> image:

$$X = \begin{bmatrix} \text{———} & x_1 - \mu & \text{———} \\ \text{———} & x_2 - \mu & \text{———} \\ & \vdots & \\ \text{———} & x_n - \mu & \text{———} \end{bmatrix}$$

Eigenfaces

Compute top 'k' PCs on centered data:



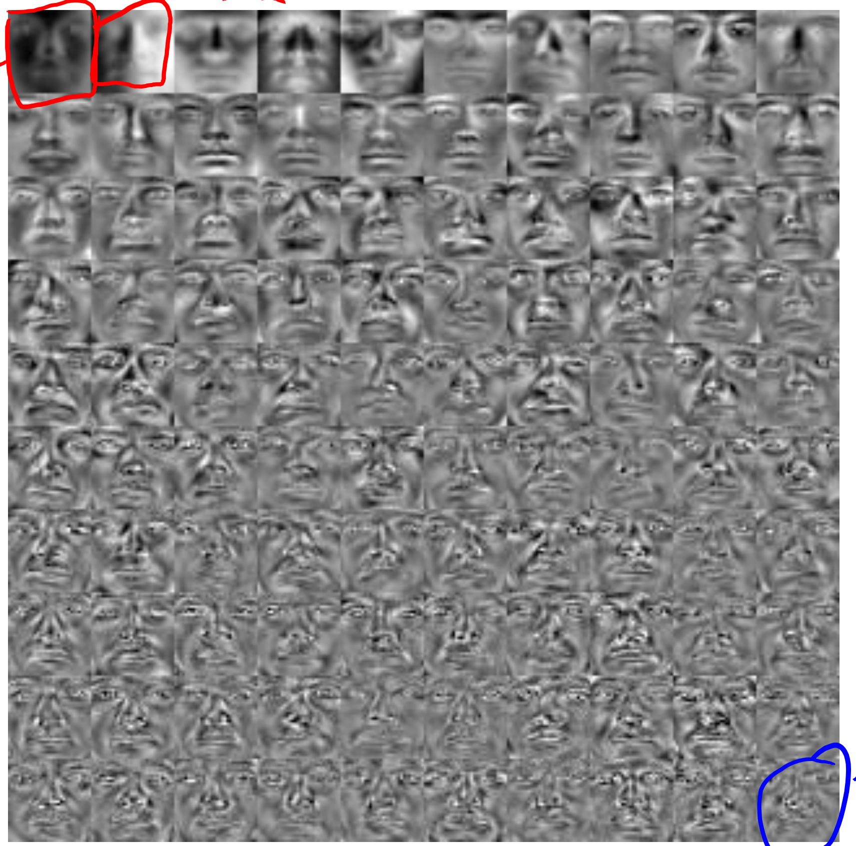
Each row of X will be pixels in <u>one</u> image:

$$X = \begin{bmatrix} \text{---} & x_1 - \mu & \text{---} \\ \text{---} & x_2 - \mu & \text{---} \\ & \vdots & \\ \text{---} & x_n - \mu & \text{---} \end{bmatrix}$$

Eigenfaces

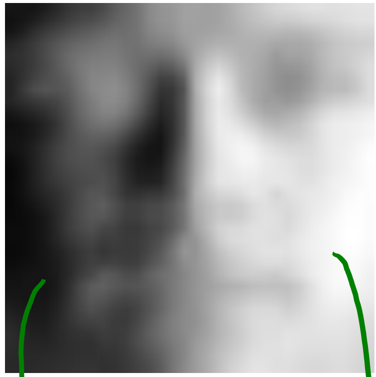
Compute top 'k' PCs on centered data:

PC2



PC1

Note that these are "signed" images.



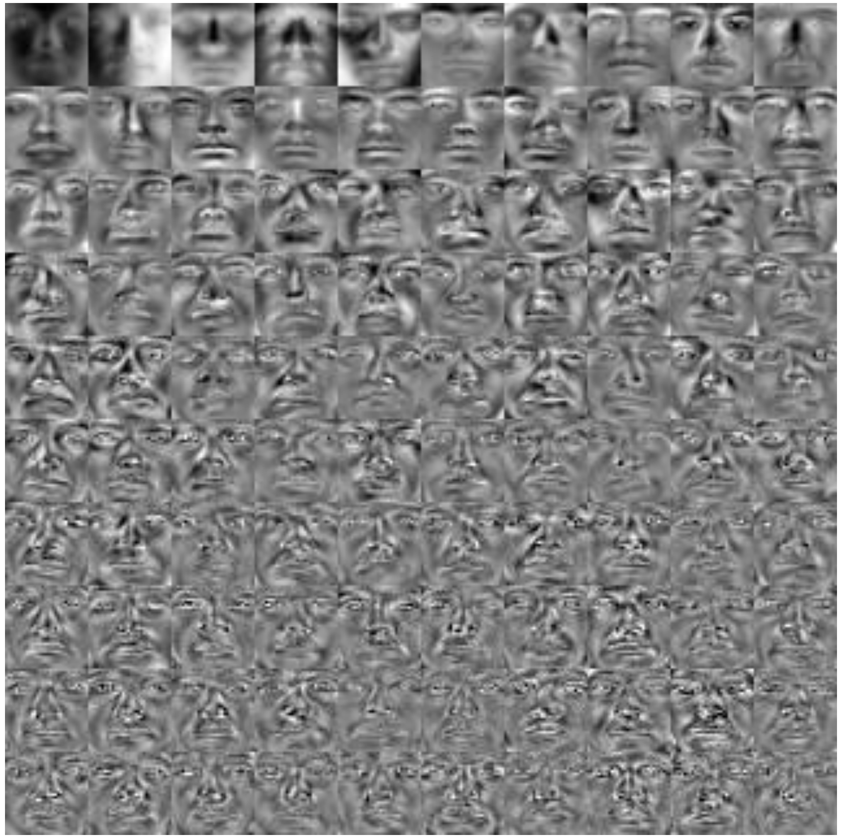
"gray" represents values close to 0.

"dark" represents negative values

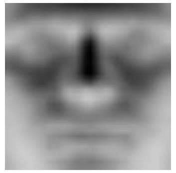
"bright" represents positive values

Eigenfaces

Compute top 'k' PCs on centered data:



"Eigenface" representation:

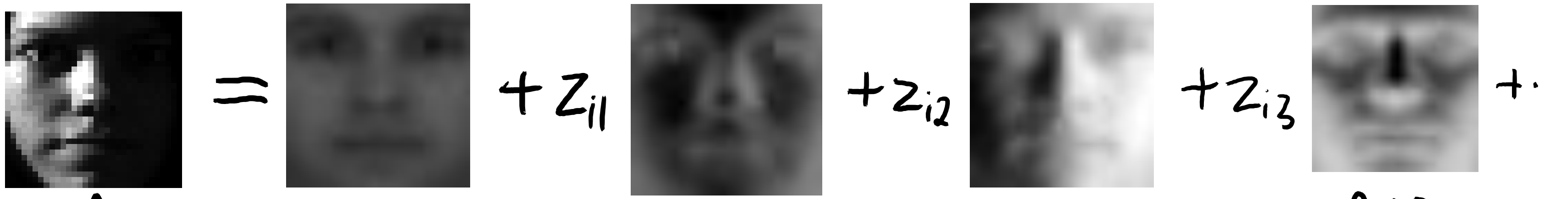
 =  $+ z_{i1}$  $+ z_{i2}$  $+ z_{i3}$  $+ \dots$

$\hat{x}_i$ $\quad$ μ $\quad$ PC1 (first <u>row</u> of W) $\quad$ PC2 $\quad$ PC3

Eigenfaces

100 of the original faces:

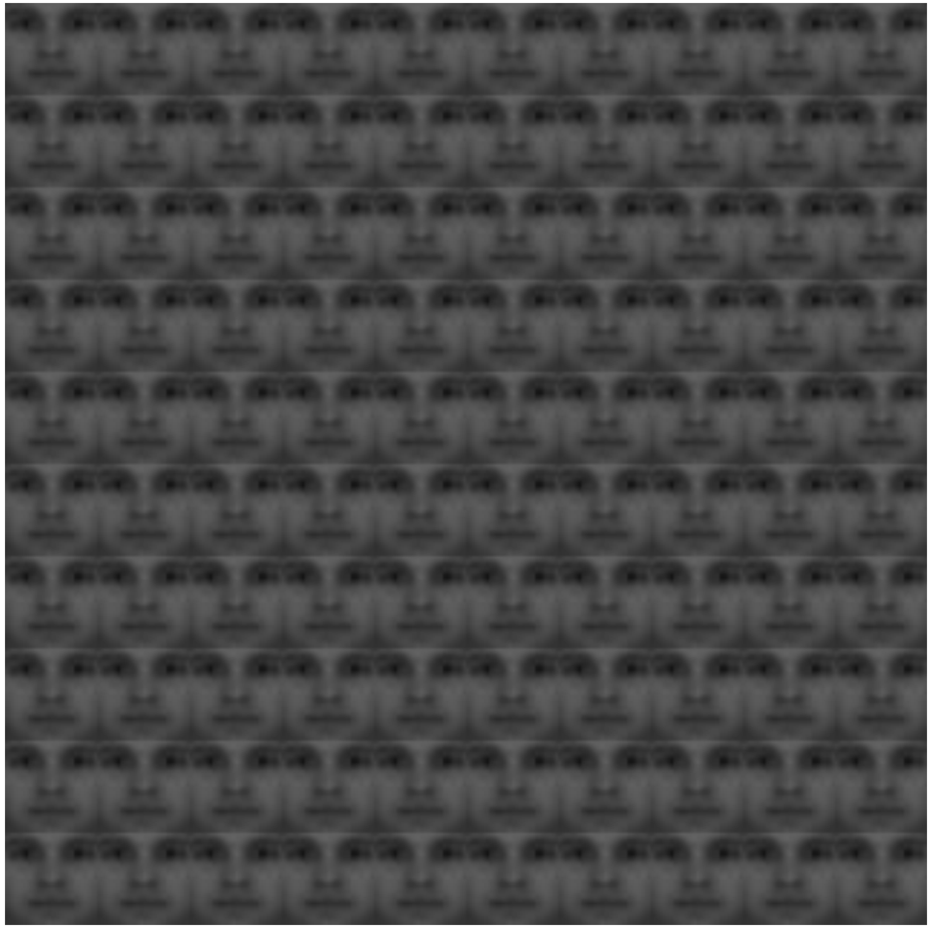
"Eigenface" representation:



$\hat{x}_i$ μ PC1 (first row of W) PC2 PC3

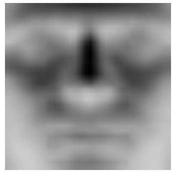
Eigenfaces

Reconstruction with $k = 0$



Variance explained: 0%

"Eigenface" representation:

 =  $+ z_{i1}$  $+ z_{i2}$  $+ z_{i3}$  $+ \dots$

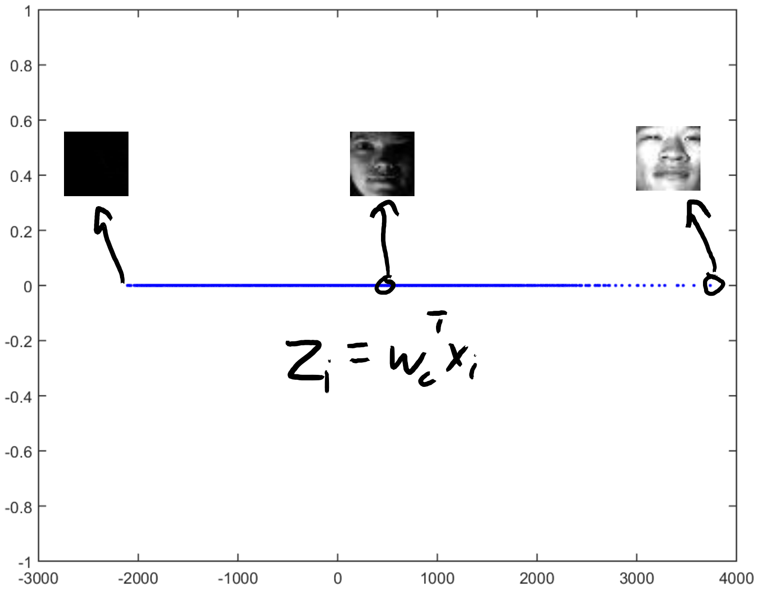
$\hat{x}_i$ $\quad$ μ $\quad$ PC1 (first row of W) $\quad$ PC2 $\quad$ PC3

Eigenfaces

Reconstruction with $k=1$

Variance explained: 36%

PCA Visualization:



"Eigenface" representation:

$$\boxed{\hat{x}_i = \mu + z_{i1}\,\text{PC1}} + z_{i2}\,\text{PC2} + z_{i3}\,\text{PC3} + \ldots$$

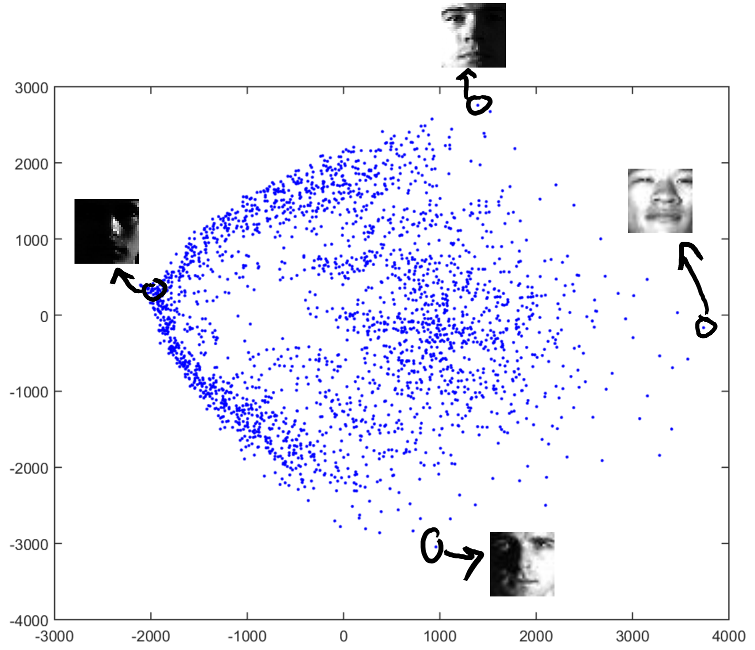
PC1 (first row of W)

Eigenfaces

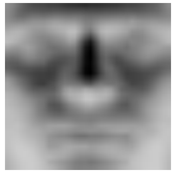
Reconstruction with $k=2$

Variance explained: 71%

PCA Visualization:



"Eigenface" representation

 =  $+ z_{i1}$  $+ z_{i2}$  $+ z_{i3}$  $+ \cdots$

$\hat{x}_i$ μ PC1 PC2 PC3

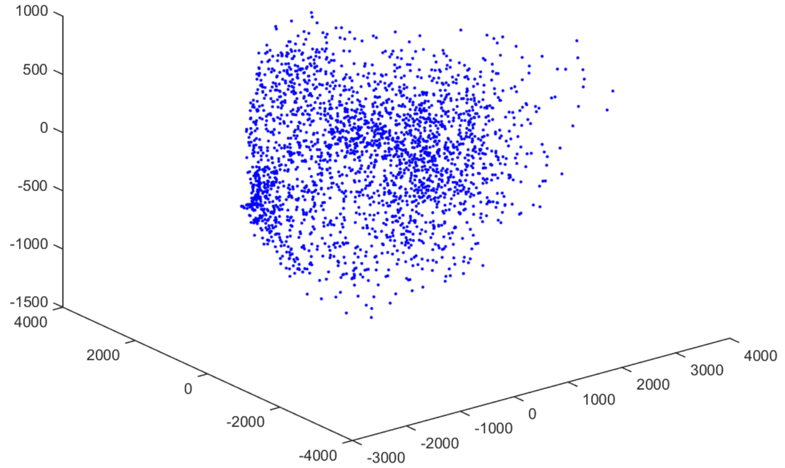
(first row of W)

Eigenfaces

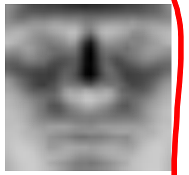
Reconstruction with $k=3$

Variance explained: 76%

PCA Visualization:



"Eigenface" representation:

 =  $+ z_{i1}$  $+ z_{i2}$  $+ z_{i3}$  + ...

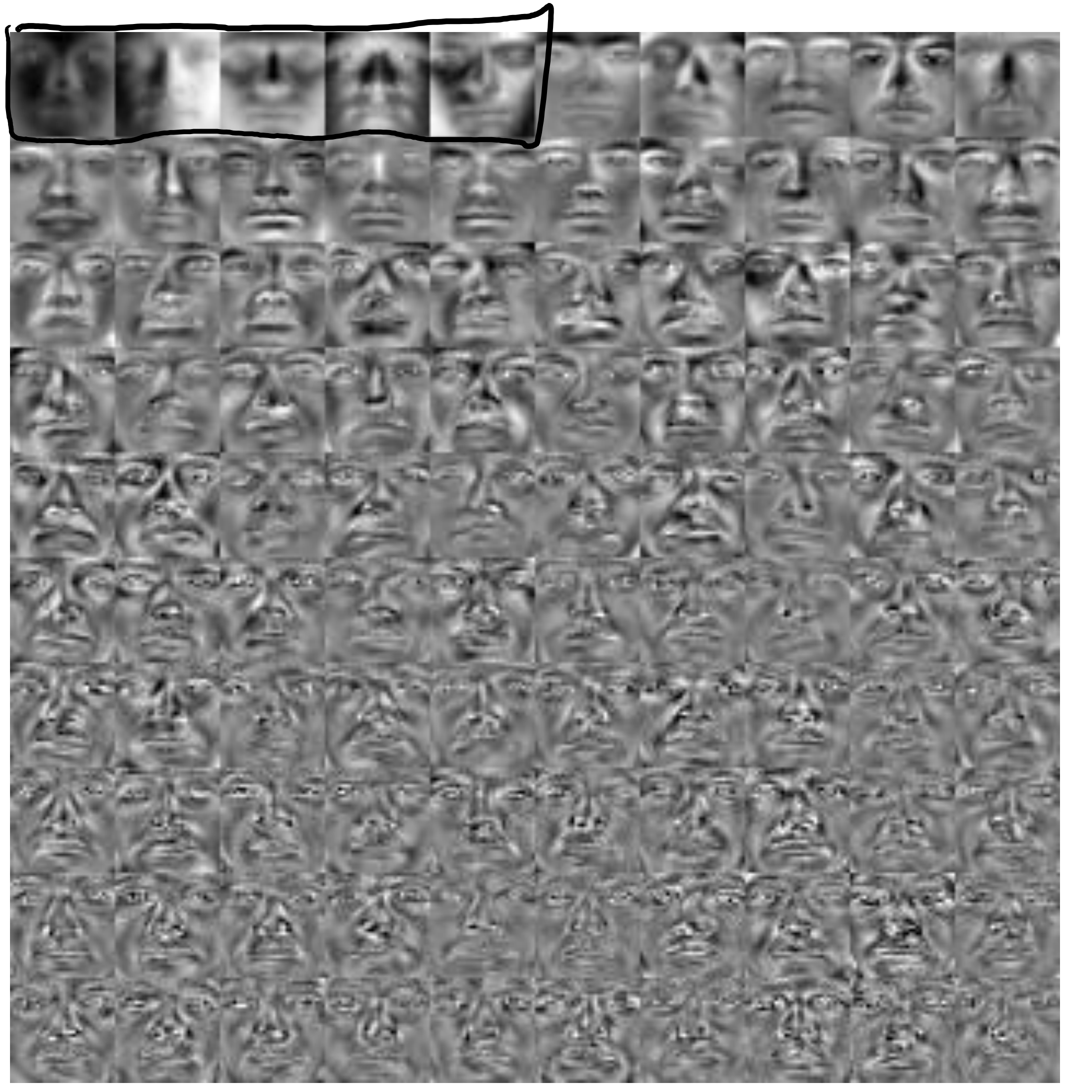
$\hat{x}_i$ μ PC1 PC2 PC3

(first row of W)

Eigenfaces

Reconstruction with $k=5$

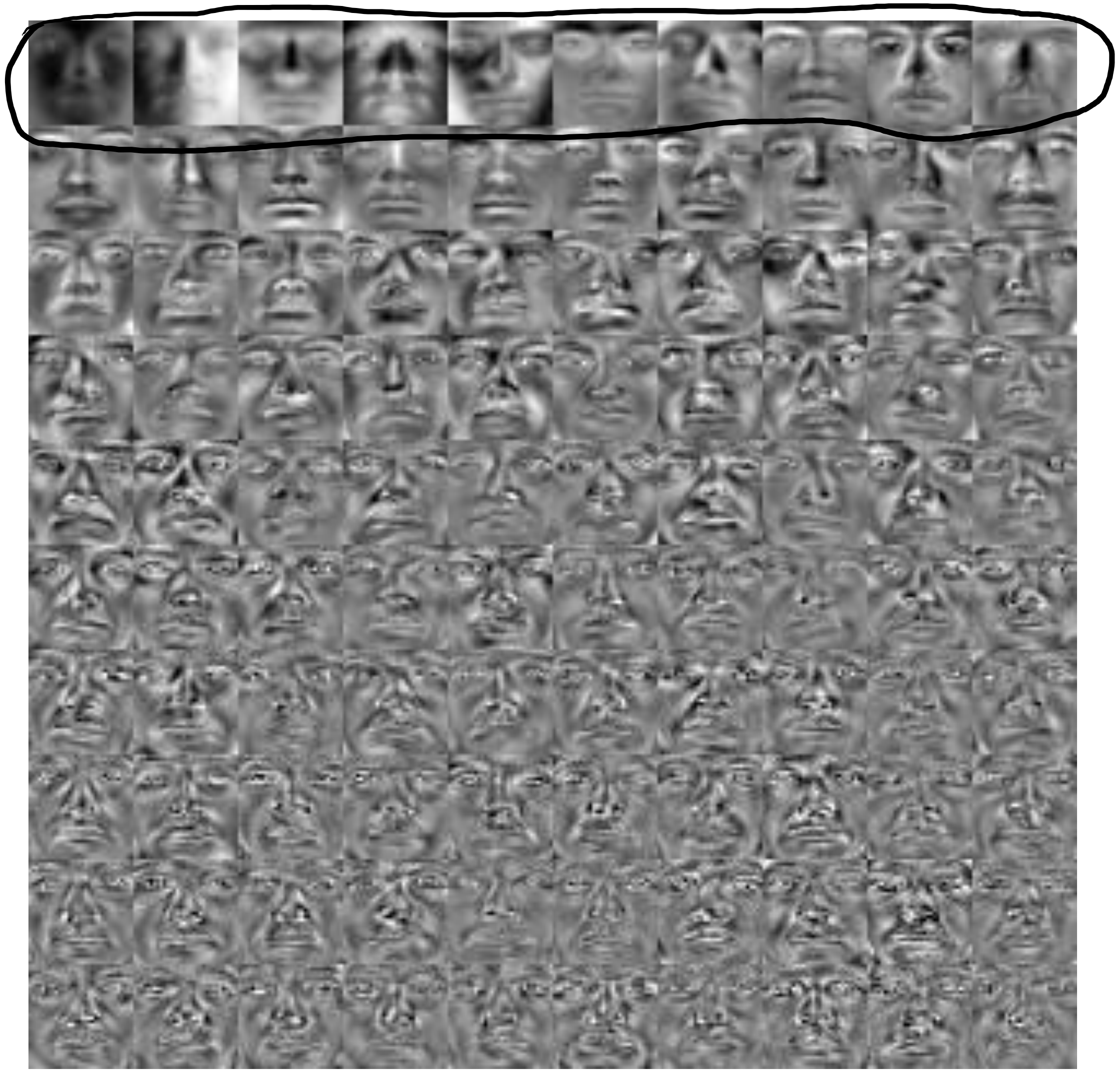
Variance explained: 80%



Eigenfaces

Reconstruction with $k=10$

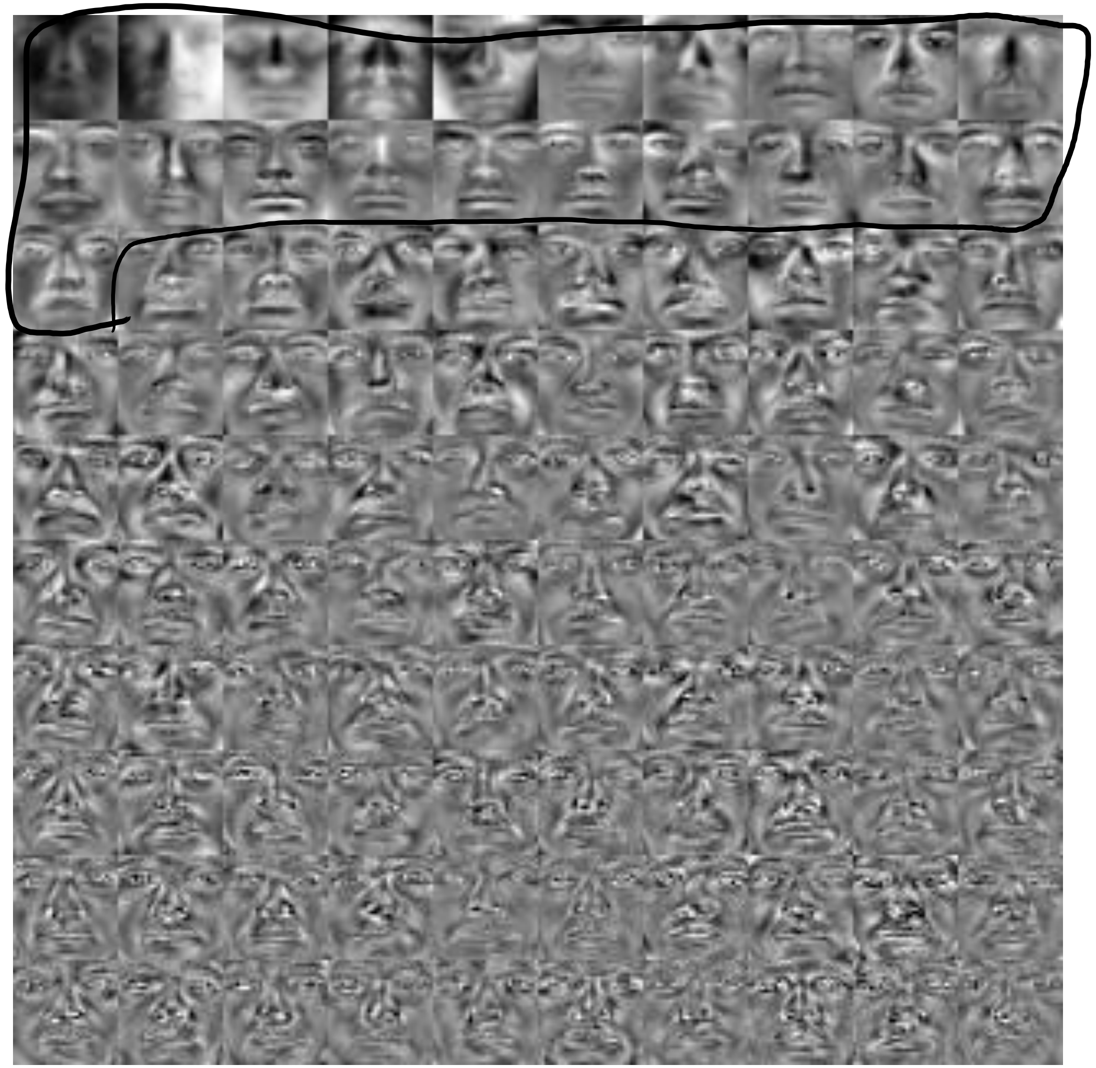
Variance explained: 85%



Eigenfaces

Reconstruction with k=21

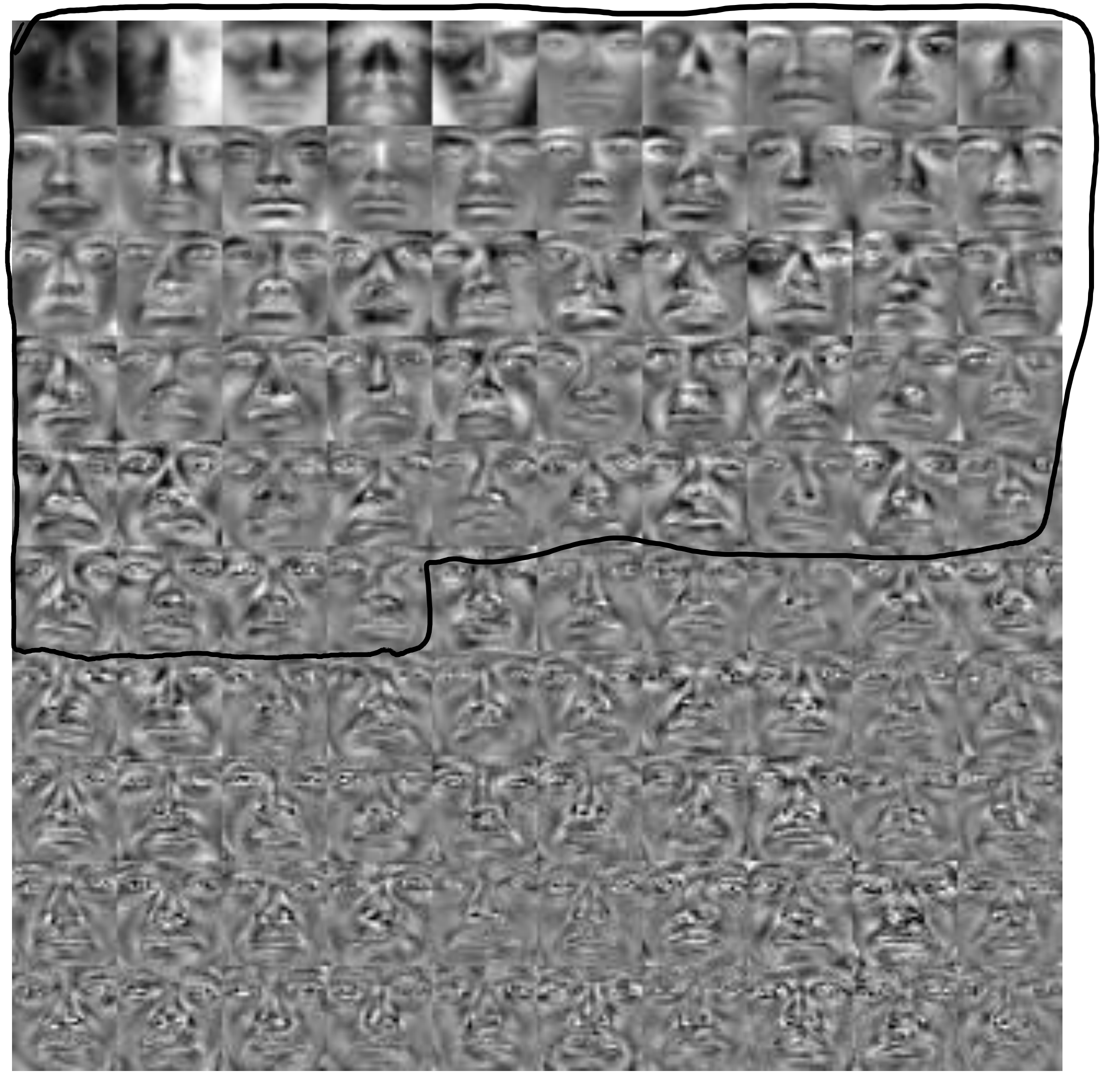
Variance explained: 90%



Eigenfaces

Reconstruction with $k = 54$

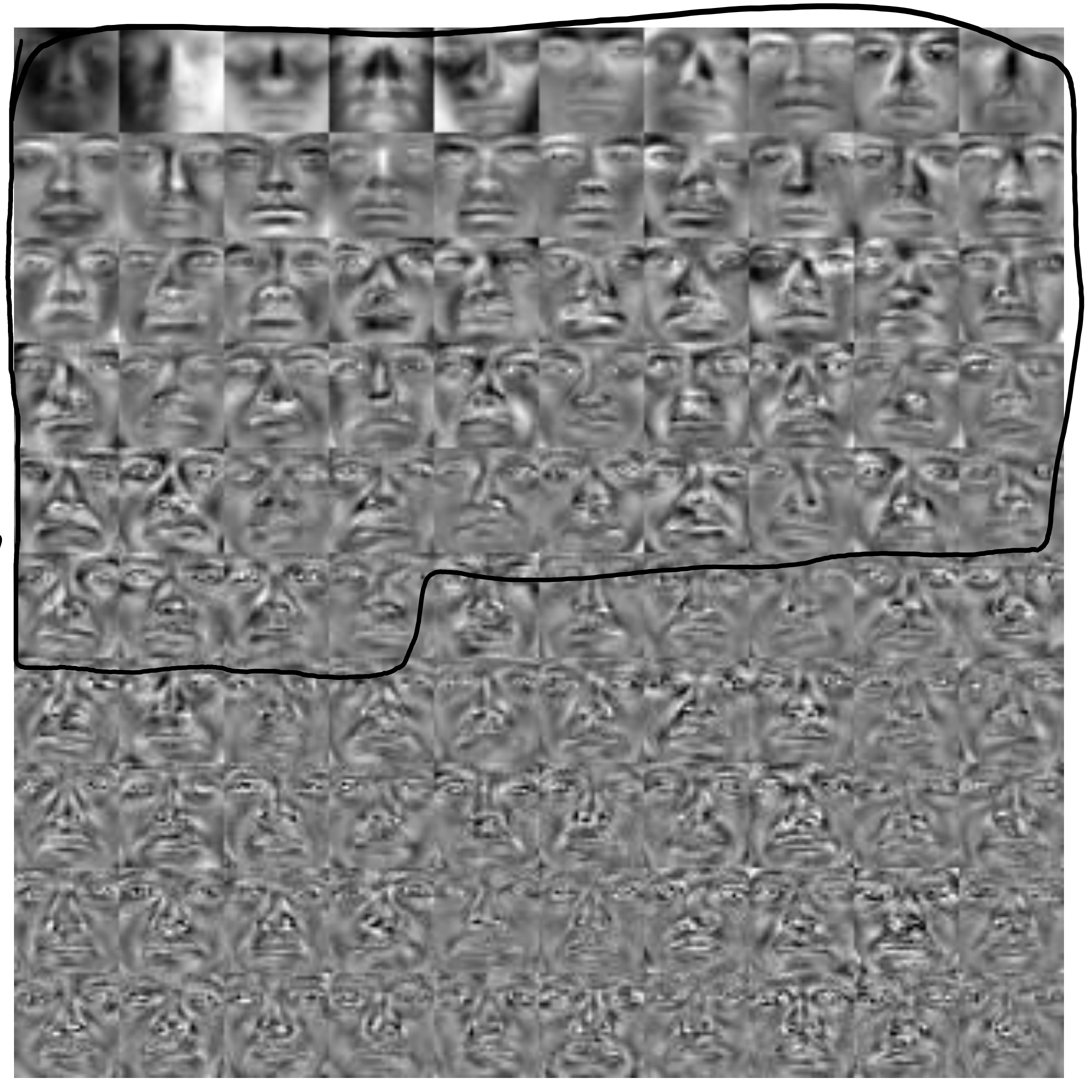
Variance explained: 95%



Eigenfaces

Original Images again:

Plus these "eigenfaces" and the mean.



We can replace 1024 x_i values by 54 z_i values

VQ vs. PCA vs. NMF

- But how *should* we represent faces?
  - Vector quantization (k-means).
    - Replace face by the average face in a cluster.
    - 'Grandmother cell': one neuron = one face.
    - Can't distinguish between people in the same cluster (only 'k' possible faces).
    - Almost certainly not true: too few neurons.

$$\hat{x}_i = z_{i1} * w_1 + z_{i2} * w_2 + z_{i3} * w_3 + z_{i4} * w_4 + z_{i5} * w_5 + z_{i6} * w_6$$

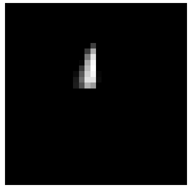
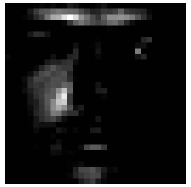
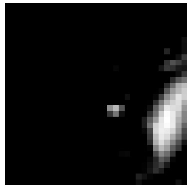
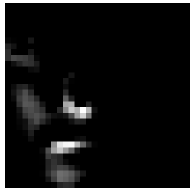
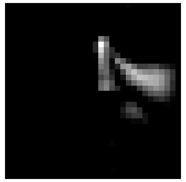
VQ vs. PCA vs. NMF

- But how *should* we represent faces?
  - Vector quantization (k-means).
  - PCA (orthogonal basis).
    - Global average plus linear combination of “eigenfaces”.
    - “Distributed representation”.
      - Coded by pattern of group of neurons: can represent infinite number of faces by changing z_i.
    - But “eigenfaces” are not intuitive ingredients for faces.
      - PCA tends to use positive/negative cancelling bases.

$\hat{x}_i = \mu + z_{i1} * w_1 + z_{i2} * w_2 + z_{i3} * w_3 + z_{i4} * w_4 + z_{i5} * w_5$

VQ vs. PCA vs. NMF

- But how *should* we represent faces?
  - Vector quantization (k-means).
  - PCA (orthogonal basis).
  - NMF (non-negative matrix factorization):
    - Instead of orthogonality/ordering in W, require W and Z to be non-negativity.
    - Example of "sparse coding":
      - The z_i are sparse so each face is coded by a small number of neurons.
      - The w_c are sparse so neurons tend to be "parts" of the object.

$=$ $4.2 *$  $+ 0 *$  $+ 0 *$  $+ 3.3 *$  $+ 0 *$  $+ \dots$

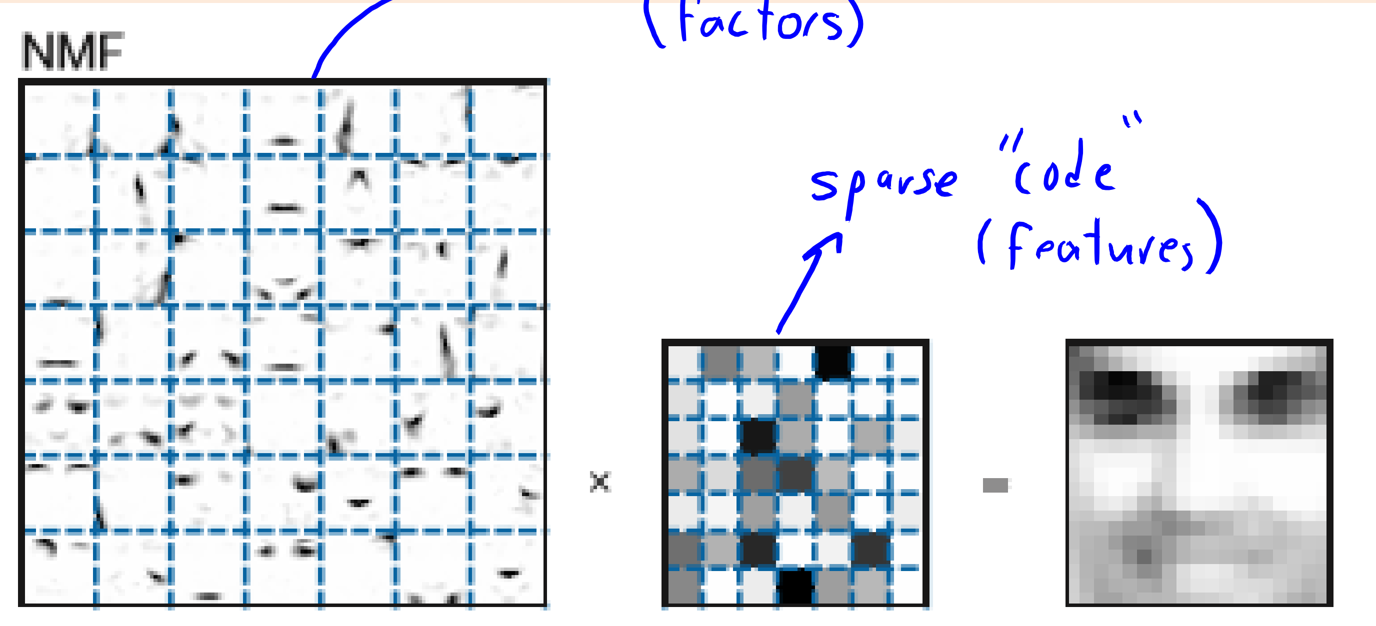
$$\hat{x}_i = z_{i1} * w_1 + z_{i2} * w_2 + z_{i3} * w_3 + z_{i4} * w_4 + z_{i5} * w_5$$

Representing Faces

- Why sparse coding?
  - "Parts" are intuitive, and brains seem to use sparse representation.
  - Energy efficiency if using sparse code.
  - Increase number of concepts you can memorize?
    - Some evidence in fruit fly olfactory system.

Sparse "dictionary" (factors)

sparse "code" (features)



http://www.columbia.edu/~jwp2128/Teaching/W4721/papers/nmf_nature.pdf

Warm-up to NMF: Non-Negative Least Squares

- Consider our usual least squares problem:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^Tx_i - y_i)^2$$

- But assume y_i and elements of x_i are non-negative:
  - Could be sizes (‘height’, ‘milk’, ‘km’) or counts (‘vicodin’, ‘likes’, ‘retweets’).
- Assume we want elements of ‘w’ to be non-negative, too:
  - No physical interpretation to negative weights.
  - If x_{ij} is amount of product you produce, what does $w_j < 0$ mean?

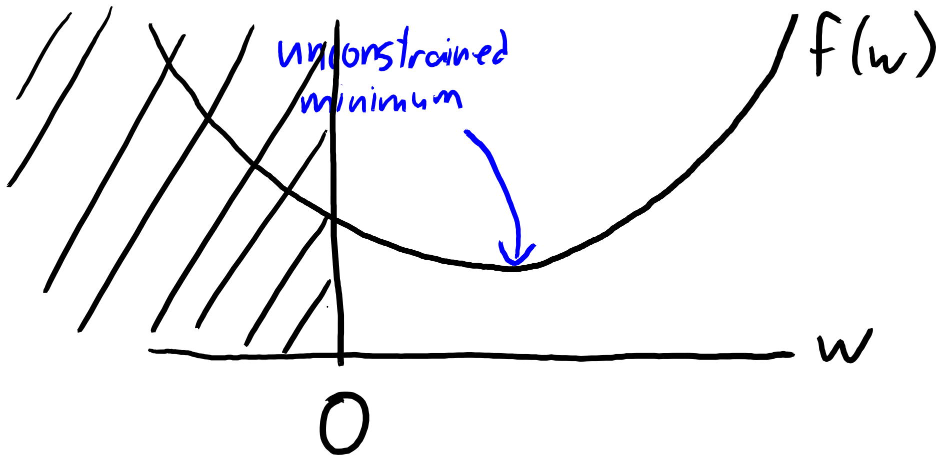
- Non-negativity leads to sparsity...

Sparsity and Non-Negative Least Squares

- Consider 1D non-negative least squares objective:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w x_i - y_i)^2 \quad \text{with} \quad w \geq 0$$

- Plotting the (constrained) objective function:



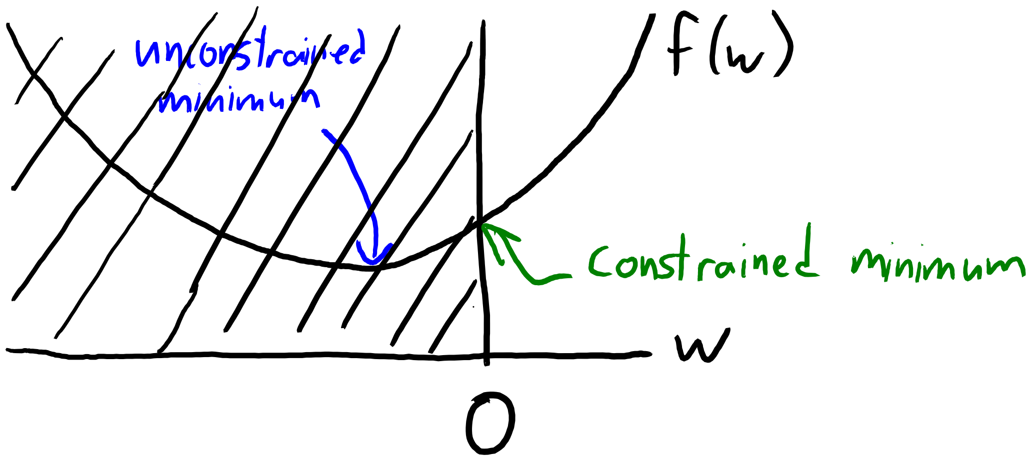
- In this case, non-negative solution is least squares solution.

Sparsity and Non-Negative Least Squares

- Consider 1D non-negative least squares objective:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w x_i - y_i)^2 \quad \text{with} \quad w \geq 0$$

- Plotting the (constrained) objective function:



- In this case, non-negative solution is w = 0.

Sparsity and Non-Negativity

- Similar to L1-regularization, non-negativity leads to sparsity.
  - Also regularizes: w_j are smaller since can't "cancel" negative values.
  - Sparsity leads to cheaper predictions and often to more interpretability.
    - Non-negative weights are often also more interpretable.

- How can we minimize f(w) with non-negative constraints?
  - Naive approach: solve least squares problem, set negative w_j to 0.

$$\text{Compute } w = (X^TX)\backslash(X^Ty)$$

$$\text{Set } w_j = \max\{0, w_j\}$$

  - This is correct when d = 1.
  - Can be worse than setting w = 0 when d ≥ 2.

Sparsity and Non-Negativity

- Similar to L1-regularization, non-negativity leads to sparsity.
  - Also regularizes: w_j are smaller since can't "cancel" out negative values.
- How can we minimize f(w) with non-negative constraints?
  - A correct approach is projected gradient algorithm:
    - Run a gradient descent iteration:

$$w^{t+1/2} = w^t - \alpha^t \nabla f(w^t)$$

    - After each step, set negative values to 0.

$$w_j^{t+1} = \max\{0, w_j^{t+1/2}\}$$

    - Repeat.

Sparsity and Non-Negativity

- Similar to L1-regularization, non-negativity leads to sparsity.
  - Also regularizes: w_j are smaller since can't "cancel" out negative values.
- How can we minimize f(w) with non-negative constraints?
  - A correct approach is projected gradient algorithm:

$$w^{t+1/2} = w^t - \alpha^t \nabla f(w^t) \qquad\qquad w_j^{t+1} = \max\{0, w_j^{t+1/2}\}$$

  - Similar properties to gradient descent:
    - Guaranteed decrease of 'f' if α_t is small enough.
    - Reaches local minimum under weak assumptions (global minimum for convex 'f').
      - Least squares objective is still convex when restricted to non-negative variables.
    - Generalizations allow things like L1-regularization instead of non-negativity.

("findMinL1")

Projected-Gradient for NMF

- Back to the non-negative matrix factorization (NMF) objective:

$$f(W,Z) = \sum_{i=1}^{n}\sum_{j=1}^{d}(\langle w^j, z_i\rangle - x_{ij})^2$$

with $w_{cj} \geq 0$ and $z_{ij} \geq 0$

  - Different ways to use projected gradient:
    - Alternate between projected gradient steps on 'W' and on 'Z'.
    - Or run projected gradient on both at once.
    - Or sample a random 'i' and 'j' and do stochastic projected gradient.

Set $z_i^{t+1} = z_i^t - \alpha^t \nabla_{z_i} f(W,Z)$ and $(w^j)^{t+1} = (w^j)^t - \alpha^t \nabla_{w^j} f(W,Z)$ for selected i and j

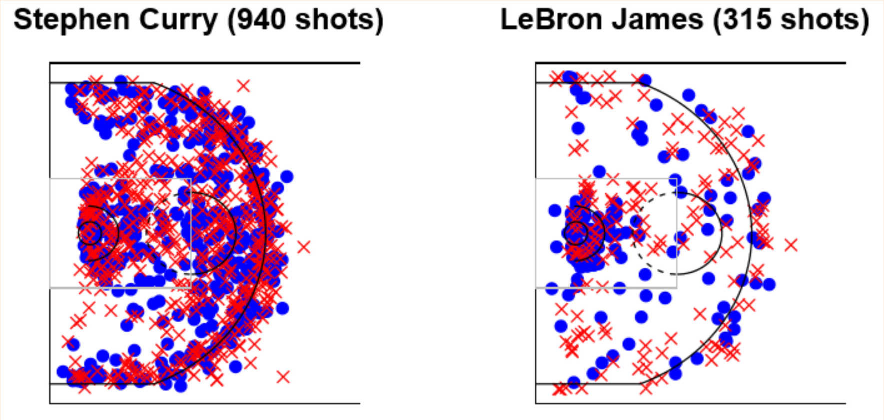
(keep other values of W and Z fixed)

Then set negative values to 0.

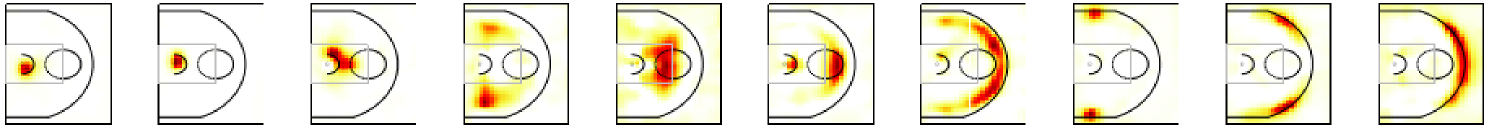
  - Non-convex and (unlike PCA) is sensitive to initialization.
    - Hard to find the global optimum.
    - Typically use random initialization.

Application: Sports Analytics

- NBA shot charts:



- NMF (using "KL divergence" loss with k=10 and smoothed data).
  - Negative values would not make sense here.

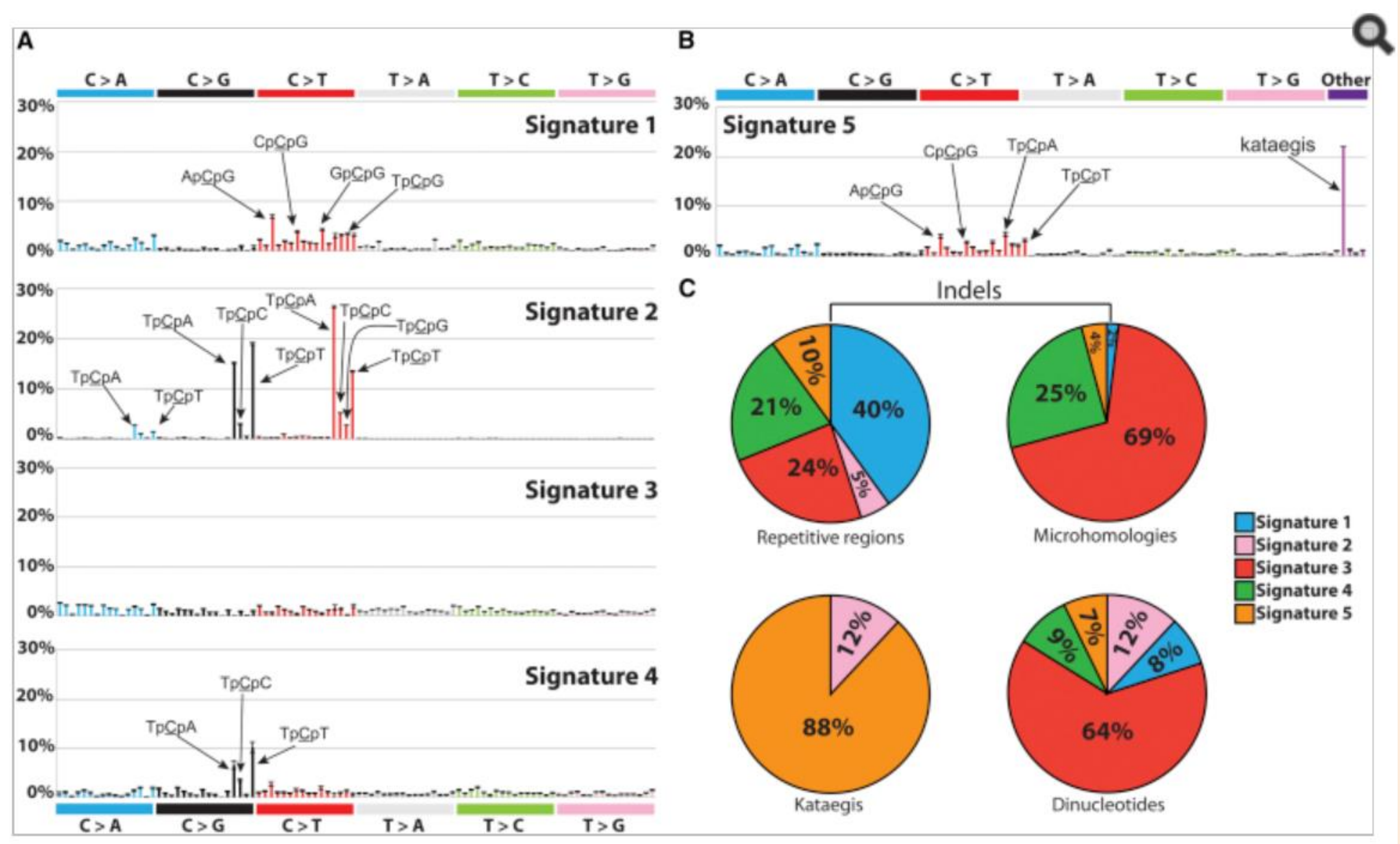


LeBron James	0.21	0.16	0.12	0.09	0.04	0.07	0.00	0.07	0.08	0.17
Brook Lopez	0.06	0.27	0.43	0.09	0.01	0.03	0.08	0.03	0.00	0.01
Tyson Chandler	0.26	0.65	0.03	0.00	0.01	0.02	0.01	0.01	0.02	0.01
Marc Gasol	0.19	0.02	0.17	0.01	0.33	0.25	0.00	0.01	0.00	0.03
Tony Parker	0.12	0.22	0.17	0.07	0.21	0.07	0.08	0.06	0.00	0.00
Kyrie Irving	0.13	0.10	0.09	0.13	0.16	0.02	0.13	0.00	0.10	0.14
Stephen Curry	0.08	0.03	0.07	0.01	0.10	0.08	0.22	0.05	0.10	0.24
James Harden	0.34	0.00	0.11	0.00	0.03	0.02	0.13	0.00	0.11	0.26
Steve Novak	0.00	0.01	0.00	0.02	0.00	0.00	0.01	0.27	0.35	0.34

http://jmlr.org/proceedings/papers/v32/miller14.pdf

Application: Cancer "Signatures"

- What are common sets of mutations in different cancers?
  - May lead to new treatment options.



https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3588146/

Summary

- Biological motivation for orthogonal and/or sparse latent factors.
- Choosing ‘k’:
  - We can choose ‘k’ to explain “percentage of variance” in the data.
- Non-negative matrix factorization leads to sparse LFM.
- Non-negativity constraints lead to sparse solution.
  - Projected gradient adds constraints to gradient descent.
  - Non-orthogonal LFMs make sense in many applications.
- L1-regularization leads to other sparse LFMs.

- Next time: the million-dollar NetFlix challenge.

Proof: "Synthesis" View = "Analysis" View ($WW^T = I$)

- The variance of the z_{ij} (maximized in "analysis" view):

$$\frac{1}{nk}\sum_{i=1}^{n}\|z_i - \mu_z\|^2 = \frac{1}{nk}\sum_{i=1}^{n}\|Wx_i\|^2 \quad (\mu_z = 0 \text{ and } z_i = Wx_i \text{ if } \|w_c\| = 1 \text{ and } w_c^T w_{c'} = 0)$$

$$= \frac{1}{nk}\sum_{i=1}^{n} x_i^T W^T W x_i = \frac{1}{nk}\sum_{i=1}^{n} Tr(x_i^T W^T W x_i) = \frac{1}{nk}\sum_{i=1}^{n} Tr(W^T W x_i x_i^T)$$

"cyclic" property of trace

$$= \frac{1}{nk} Tr(W^T W \underbrace{\sum_{i=1}^{n} x_i x_i^T}_{X^T X}) = \frac{1}{nk} Tr(W^T W X^T X)$$

linearity of trace

- The distance to the hyper-plane (minimized in "synthesis" view):

$\|A\|_F^2 = Tr(A^T A)$

$$\|ZW - X\|_F^2 = \|XW^T W - X\|_F^2 = Tr((XW^TW - X)^T(XW^TW - X))$$

$= XW^T$

$$= Tr(W^T W X^T X W^T W) - 2Tr(W^T W X^T X) + Tr(X^T X)$$

$$= Tr(W^T \underbrace{W W^T}_{I} W X^T X) - 2Tr(W^T W X^T X) + Tr(X^T X)$$

$$= -Tr(W^T W X^T X) + (\text{constant})$$

Solved by same 'W'

Canonical Correlation Analysis (CCA)

- Suppose we have two matrices, ‘X’ and ‘Y’.
- Want to find matrices W_X and W_Y that maximize correlation.
  - “What are the latent factors in common between these datasets?”
- Define the correlation matrices:

$$\Sigma_{xx} = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T \qquad \Sigma_{yy} = \frac{1}{n}\sum_{i=1}^{n} y_i y_i^T \qquad \Sigma_{xy} = \frac{1}{n}\sum_{i=1}^{n} x_i y_i^T$$

- Canonical correlation analysis (CCA) maximizes

$$Tr(W_Y^T W_x \Sigma_{xx}^{-1/2} \Sigma_{xy} \Sigma^{-1/2})$$

  - Subject to W_X and W_Y having orthogonal rows.
- Computationally, equivalent to PCA with a different matrix.
  - Using the “analysis” view that PCA maximizes $Tr(W^TWX^TX)$.

Kernel PCA

- From the "analysis" view (with orthogonal PCs) PCA maximizes:

$$Tr(W^TWX^TX)$$

- It can be shown that the solution has the form (see here):

$$\underbrace{W}_{k \times d} = \underbrace{U}_{k \times n}\underbrace{X}_{n \times d}$$

- Re-parameterizing in terms of 'U' gives a kernelized PCA:

$$Tr(X^TU^TUXX^TX) = Tr(U^TU\underbrace{XX^T}_{K}\underbrace{XX^T}_{K})$$

- It's hard to initially center data in 'Z' space,
  but you can form the centered kernel matrix (see here).

Probabilistic PCA

- With zero-mean ("centered") data, in PCA we assume that

$$x_i \approx W^T z_i$$

- In probabilistic PCA we assume that

$$x_i \sim \mathcal{N}(W^T z_i, \sigma^2 I) \qquad z_i \sim \mathcal{N}(0, I)$$

- Integrating over 'Z' the marginal likelihood given 'W' is Gaussian,

$$x_i \mid W \sim \mathcal{N}(0, W^T W + \sigma^2 I)$$

- Regular PCA is obtained as the limit of σ^2 going to 0.

Generalizations of Probabilistic PCA

- Probabilistic PCA model:

$$x_i \mid W \sim \mathcal{N}(0, W^TW + \sigma^2 I)$$

- Why do we need a probabilistic interpretation?

- Shows that PCA fits a Gaussian with restricted covariance.
  - Hope is that $W^TW + \sigma^2 I$ is a good approximation of X^TX.

- Gives precise connection between PCA and factor analysis.

Factor Analysis

- Factor analysis is a method for discovering latent factors.
- Historical applications are measures of intelligence and personality.

Trait	Description
Openness	Being curious, original, intellectual, creative, and open to new ideas.
Conscientiousness	Being organized, systematic, punctual, achievement-oriented, and dependable.
Extraversion	Being outgoing, talkative, sociable, and enjoying social situations.
Agreeableness	Being affable, tolerant, sensitive, trusting, kind, and warm.
Neuroticism	Being anxious, irritable, temperamental, and moody.

- A standard tool and widely-used across science and engineering.

https://new.edu/resources/big-5-personality-traits

PCA vs. Factor Analysis

- PCA and FA both write the matrix ‘X’ as

$$X \approx ZW$$

- PCA and FA are both based on a Gaussian assumption.

- Are PCA and FA the same?
  - Both are more than 100 years old.
  - People are still arguing about whether they are the same:
    - Doesn’t help that some packages run PCA when you call their FA method.

Google

pca vs. factor analysis

All Images Videos News Maps More Search tools

About 358,000 results (0.17 seconds)

[PDF] Principal Component Analysis versus Exploratory Factor ...
www2.sas.com/proceedings/sugi30/203-30.pdf
by DD Suhr - Cited by 118 - Related articles
1. Paper 203-30. **Principal Component Analysis vs.** Exploratory **Factor Analysis.** Diana D. Suhr, Ph.D. University of Northern Colorado. Abstract. **Principal** ...

pca - What are the differences between Factor Analysis and ...
stats.stackexchange.com/.../what-are-the-differences-between-**factor-anal**...
Aug 12, 2010 - **Principal Component Analysis** (PCA) and Common **Factor Analysis** (CFA) ..... differently one has to interpret the strength of loadings in PCA vs.

What are the differences between principal components ...
support.minitab.com/...**factor-analysis**/differences-between-**pca-and-facto**...
Principal Components Analysis and **Factor Analysis** are similar because both procedures are used to simplify the structure of a set of variables. However, the ...

[PDF] Principal Components Analysis - UNT
https://www.unt.edu/rss/class/.../Principal%20Components%20**Analysis**.p...
PCA vs. Factor Analysis. • It is easy to make the mistake in assuming that these are the same techniques, though in some ways exploratory factor analysis and ...

Factor analysis versus Principal Components Analysis (PCA)
psych.wisc.edu/henriques/**pca**.html
Jun 19, 2010 - **Factor analysis versus PCA.** These techniques are typically used to analyze groups of correlated variables representing one or more common ...

[PDF] Principal Component Analysis and Factor Analysis
www.stats.ox.ac.uk/~ripley/MultAnal_HT2007/PC-FA.pdf
where D is diagonal with non-negative and decreasing values and U and V ..... **Factor analysis** and **PCA** are often confused, and indeed SPSS has **PCA** as.

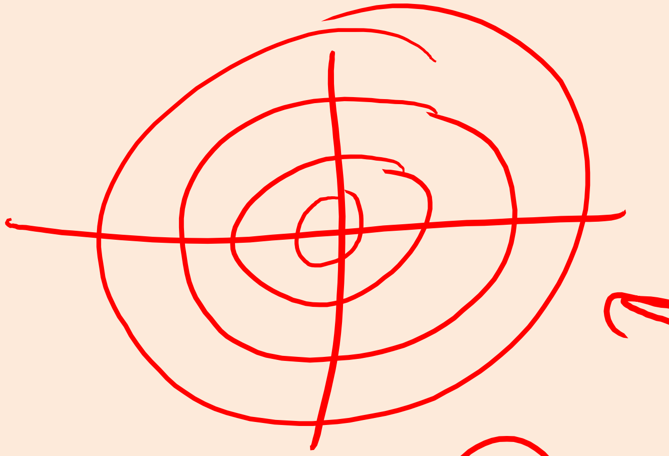
How can I decide between using principal components ...
https://www.researchgate.net/.../How_can_I_decide_between_using_prin...
Factor analysis (FA) is a group of statistical methods used to understand and simplify patterns ... Retrieved from http://pareonline.net/getvn.asp?v=10&n=7 ... **Principal component analysis** (PCA) is a method of factor extraction (the second step ...

[PDF] Exploratory Factor Analysis and Principal Component An...
www.lesahoffman.com/948/948_Lecture2_EFA_**PCA**.pdf
2 very different schools of thought on exploratory **factor analysis** (EFA) vs. principal components **analysis** (PCA): ➢ EFA and PCA are TWO ENTIRELY ...

Factor analysis - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/**Factor_analysis**
Jump to Exploratory **factor analysis versus** principal components ... - [edit]. See also: **Principal component analysis** and Exploratory **factor analysis**.

[PDF] The Truth about PCA and Factor Analysis
www.stat.cmu.edu/~cshalizi/350/lectures/13/lecture-13.pdf
Sep 28, 2009 - nents and **factor analysis**, we'll wrap up by looking at their uses and

PCA vs. Factor Analysis

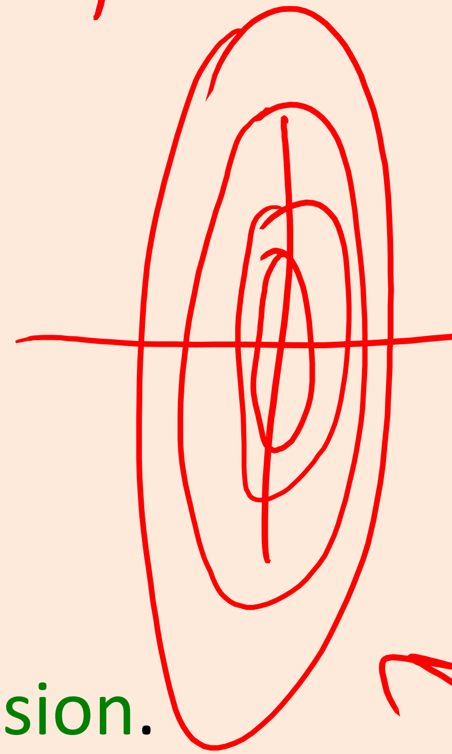


- In probabilistic PCA we assume:

$$x_i \sim \mathcal{N}(W^T z_i, \sigma^2 I)$$

- In FA we assume for a diagonal matrix D that:

$$x_i \sim \mathcal{N}(W^T z_i, D)$$

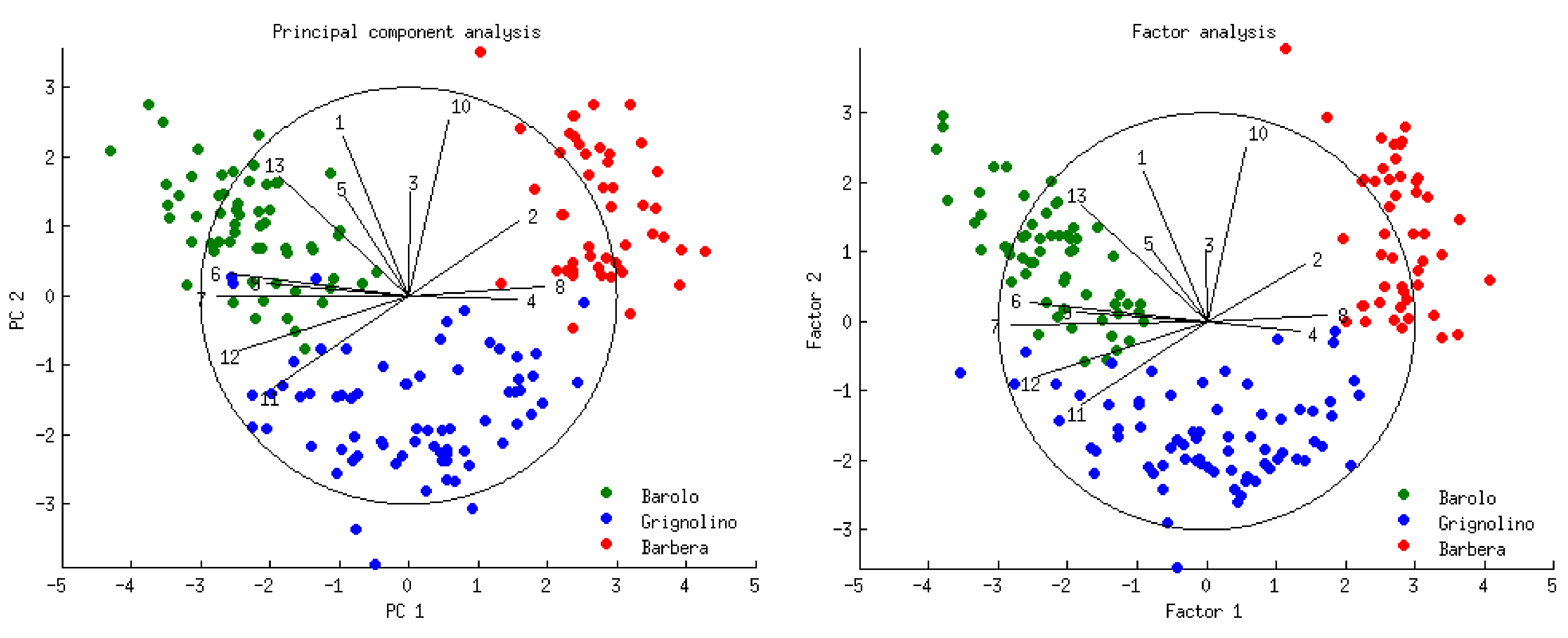


- The posterior in this case is: $x_i \mid W \sim \mathcal{N}(0, W^T W + D)$

- The difference is you have a noise variance for each dimension.
  - FA has extra degrees of freedom.

PCA vs. Factor Analysis

- In practice there often isn't a huge difference:



http://stats.stackexchange.com/questions/1576/what-are-the-differences-between-factor-analysis-and-principal-component-analysi

Factor Analysis Discussion

- Differences with PCA:
  - Unlike PCA, FA is not affected by scaling individual features.
  - But unlike PCA, it's affected by rotation of the data.
  - No nice "SVD" approach for FA, you can get different local optima.

- Similar to PCA, FA is invariant to rotation of 'W'.
  - So as with PCA you can't interpret multiple factors as being unique.

Motivation for ICA

- Factor analysis has found an enormous number of applications.
  - People really want to find the “hidden factors” that make up their data.

- But PCA and FA can’t identify the factors.

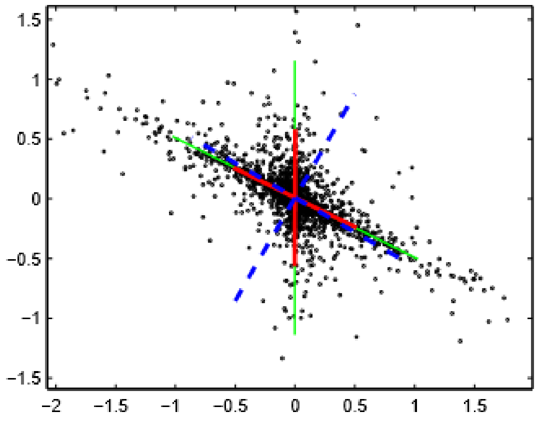


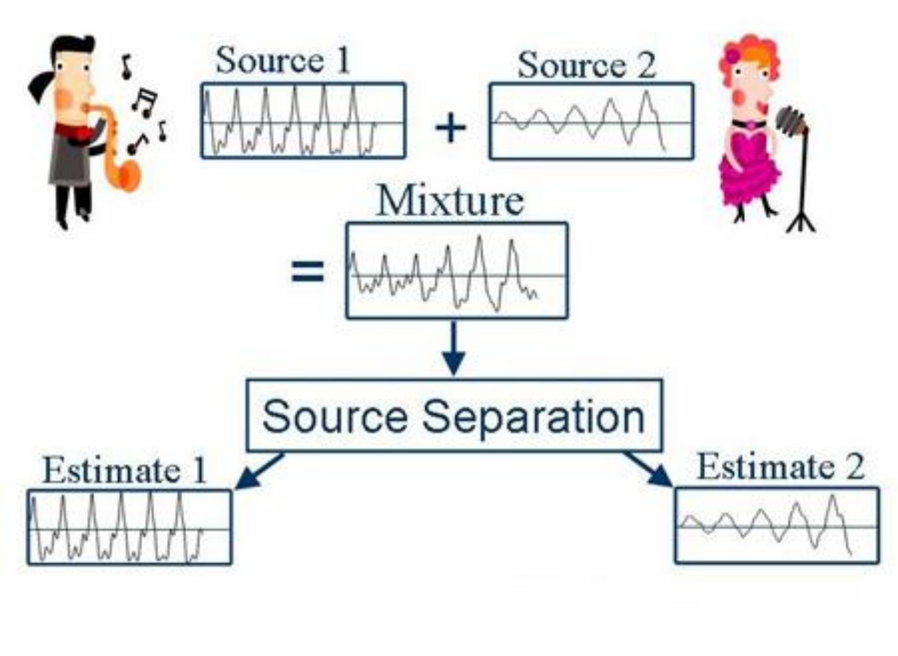
Figure : Latent data is sampled from the prior $p(x_i) \propto \exp(-5\sqrt{|x_i|})$ with the mixing matrix $\mathbf{A}$ shown in green to create the observed two dimensional vectors $\mathbf{y} = \mathbf{A}\mathbf{x}$. The red lines are the mixing matrix estimated by `ica.m` based on the observations. For comparison, PCA produces the blue (dashed) components. Note that the components have been scaled to improve visualisation. As expected, PCA finds the orthogonal directions of maximal variation. ICA however, correctly estimates the directions in which the components were independently generated.

Motivation for ICA

- Factor analysis has found an enormous number of applications.
  - People really want to find the "hidden factors" that make up their data.
- But PCA and FA can't identify the factors.
  - We can rotate W and obtain the same model.
- Independent component analysis (ICA) is a more recent approach.
  - Around 30 years old instead of > 100.
  - Under certain assumptions it can identify factors.
- The canonical application of ICA is blind source separation.

Blind Source Separation

- Input to blind source separation:
  - Multiple microphones recording multiple sources.



- Each microphone gets different mixture of the sources.
  - Goal is reconstruct sources (factors) from the measurements.

http://music.eecs.northwestern.edu/research.php

Independent Component Analysis Applications

- ICA is replacing PCA and FA in many applications:

Some ICA applications are listed below:[1]

- optical Imaging of neurons[17]
- neuronal spike sorting[18]
- face recognition[19]
- modeling receptive fields of primary visual neurons[20]
- predicting stock market prices[21]
- mobile phone communications [22]
- color based detection of the ripeness of tomatoes[23]
- removing artifacts, such as eye blinks, from EEG data.[24]

- Recent work shows that ICA can often resolve direction of causality.

https://en.wikipedia.org/wiki/Independent_component_analysis#Applications

Limitations of Matrix Factorization

- ICA is a matrix factorization method like PCA/FA,

$$X = ZW$$

- Let's assume that $X = ZW$ for a "true" W with $k = d$.
  - Different from PCA where we assume $k \leq d$.

- There are only 3 issues stopping us from finding "true" W.

3 Sources of Matrix Factorization Non-Uniquness

- Label switching: get same model if we permute rows of W.
  - We can exchange row 1 and 2 of W (and same columns of Z).
  - Not a problem because we don't care about order of factors.

- Scaling: get same model if you scale a row.
  - If we mutiply row 1 of W by α, could multiply column 1 of Z by $1/\alpha$.
  - Can't identify sign/scale, but might hope to identify direction.

- Rotation: get same model if we rotate W.
  - Rotations correspond to orthogonal matrices Q, such matrices have $Q^TQ = I$.
  - If we rotate W with Q, then we have $(QW)^TQW = W^TQ^TQW = W^TW$.

- If we could address rotation, we could identify the "true" directions.

A Unique Gaussian Property

- Consider an independent prior on each latent features z_c.
  - E.g., in PPCA and FA we use N(0,1) for each z_c.

- If prior p(z) is independent and rotation-invariant (p(Qz) = p(z)), then it must be Gaussian (only Gaussians have this property).

- The (non-intuitive) magic behind ICA:
  - If the priors are all non-Gaussian, it isn’t rotationally symmetric.
  - In this case, we can identify factors W (up to permutations and scalings).

PCA vs. ICA

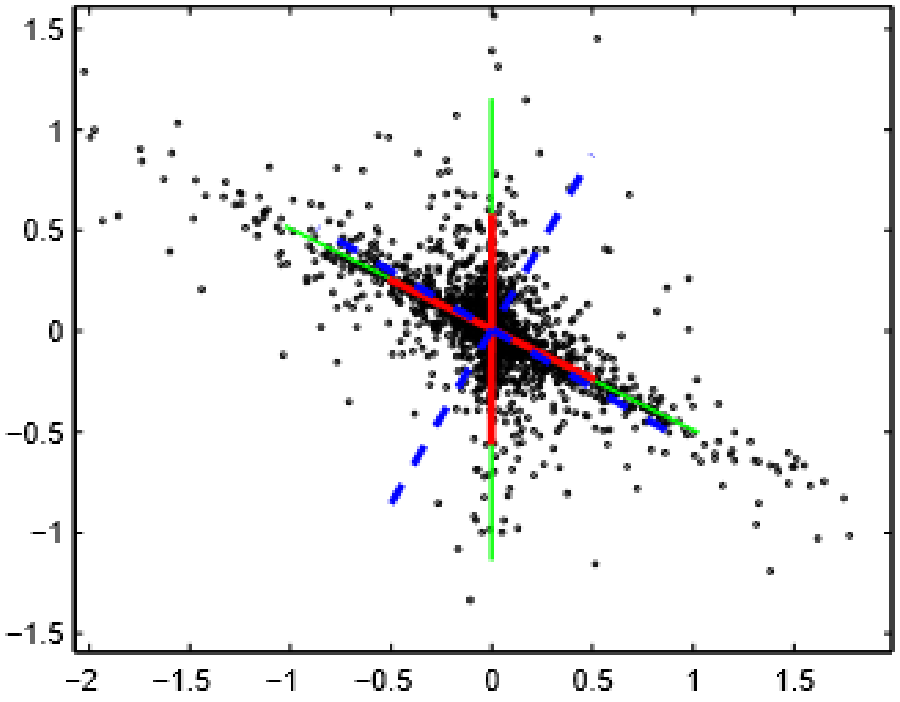


Figure : Latent data is sampled from the prior $p(x_i) \propto \exp(-5\sqrt{|x_i|})$ with the mixing matrix $\mathbf{A}$ shown in green to create the observed two dimensional vectors $\mathbf{y} = \mathbf{A}\mathbf{x}$. The red lines are the mixing matrix estimated by `ica.m` based on the observations. For comparison, PCA produces the blue (dashed) components. Note that the components have been scaled to improve visualisation. As expected, PCA finds the orthogonal directions of maximal variation. ICA however, correctly estimates the directions in which the components were independently generated.

http://www.inf.ed.ac.uk/teaching/courses/pmr/lectures/ica.pdf

Independent Component Analysis

- In ICA we approximate X with ZW,
  assuming $p(z_{ic})$ are non-Gaussian.

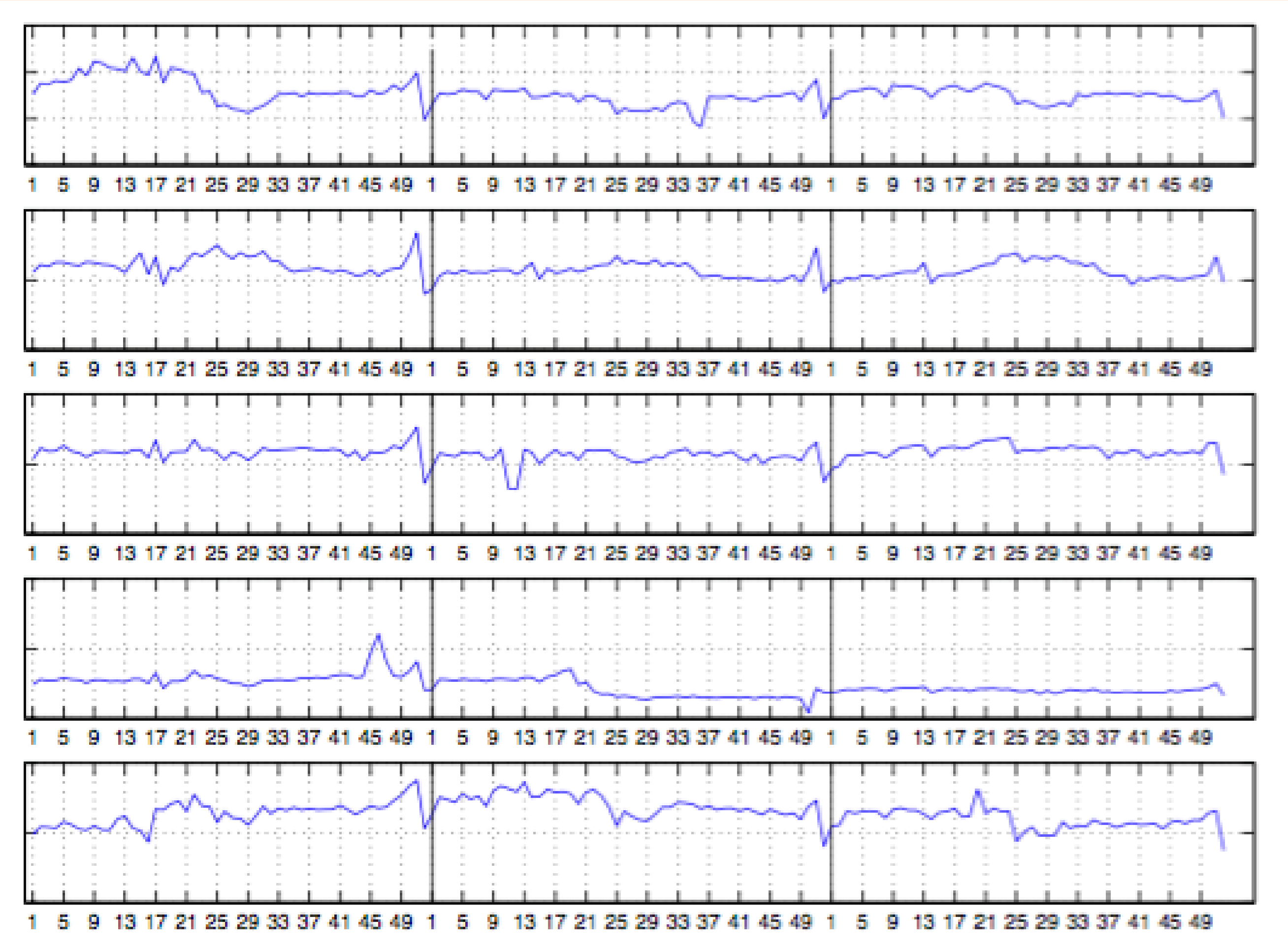
- Usually we "center" and "whiten" the data before applying ICA.

- There are several penalties that encourage non-Gaussianity:
  - Penalize low kurtosis, since kurtosis is minimized by Gaussians.
  - Penalize high entropy, since entropy is maximized by Gaussians.

- The fastICA is a popular method maximizing kurtosis.

ICA on Retail Purchase Data

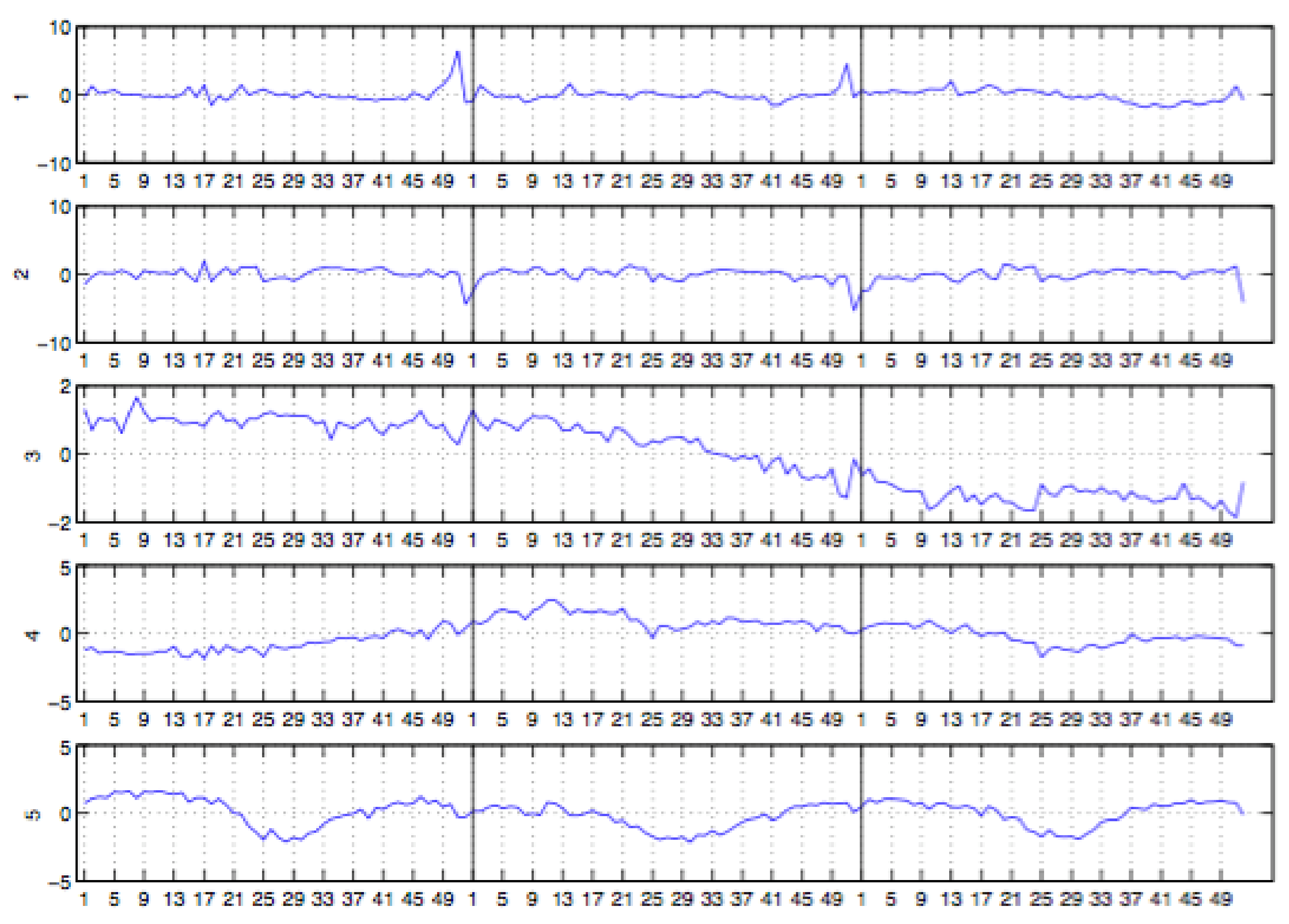
- Cash flow from 5 stores over 3 years:



http://www.stat.ucla.edu/~yuille/courses/Stat161-261-Spring14/HyvO00-icatut.pdf

ICA on Retail Purchase Data

- Factors found using ICA:



http://www.stat.ucla.edu/~yuille/courses/Stat161-261-Spring14/HyvO00-icatut.pdf