CPSC 340: Machine Learning and Data Mining

More PCA

Summer 2021

In This Lecture

1. How to Learn PCA:
   - Sequential Fitting and SVD
   - Alternating Minimization
2. Eigenfaces

Coming Up Next

PCA OBJECTIVE FUNCTION

PCA Objective Function

- In PCA we minimize the squared error of the approximation:

$$f(W,Z) = \sum_{i=1}^{n} \|\underbrace{\overset{\hat{x}_i}{W^T z_i}}_{\text{approximation}} - \underbrace{x_i}_{\text{example 'i'}}\|^2 \qquad \hat{x}_i \approx W^T z_i$$

$\underset{k \times d}{W}$

- This is equivalent to the k-means objective:
  - Think of z_i as one-hot encoding of $\hat{y}_i$

- But in PCA, z_i can be any real number.
  - We approximate x_i as a linear combination of all factors.

PCA Objective Function

- In PCA we minimize the squared error of the approximation:

$$f(W,Z) = \sum_{i=1}^{n} \| \underbrace{W^T z_i}_{\text{approximation}} - \underbrace{x_i}_{\text{example 'i'}} \|^2 = \sum_{i=1}^{n} \sum_{j=1}^{d} (\underbrace{\langle w^j, z_i \rangle}_{\text{approximation}} - \underbrace{x_{ij}}_{\text{feature 'j' of example 'i'}})^2$$

- We can also view this as solving 'd' regression problems:
  - Each w^j is a model predicting column x^j from the features of z_i. model.fit(Z, x^j)
    - The output "y_i" = each feature of x_i.
  - Unlike in regression: learn the features of z_i.

Principal Component Analysis (PCA)

- The 3 different ways to write the PCA objective function:

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

$$= \sum_{i=1}^{n}\|W^T z_i - x_i\|^2 \quad \text{(approximating } x_i \text{ by } W^T z_i\text{)}$$

$$= \|ZW - X\|_F^2 \quad \text{(approximating } X \text{ by } ZW\text{)}$$

Digression: Data Centering (Important)

- In PCA, we assume that the data X is "centered".
  - Each column of X has a mean of zero.

- It's easy to center the data:

$$\text{Set } \mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \quad (\text{mean of column 'j'})$$

$$\text{Replace each } x_{ij} \text{ with } (x_{ij} - \mu_j)$$

- There are PCA variations that estimate "bias in each coordinate".
  - In basic model this is equivalent to centering the data.

Coming Up Next

NON-UNIQUENESS OF PCA AND SPANS OF FACTORS

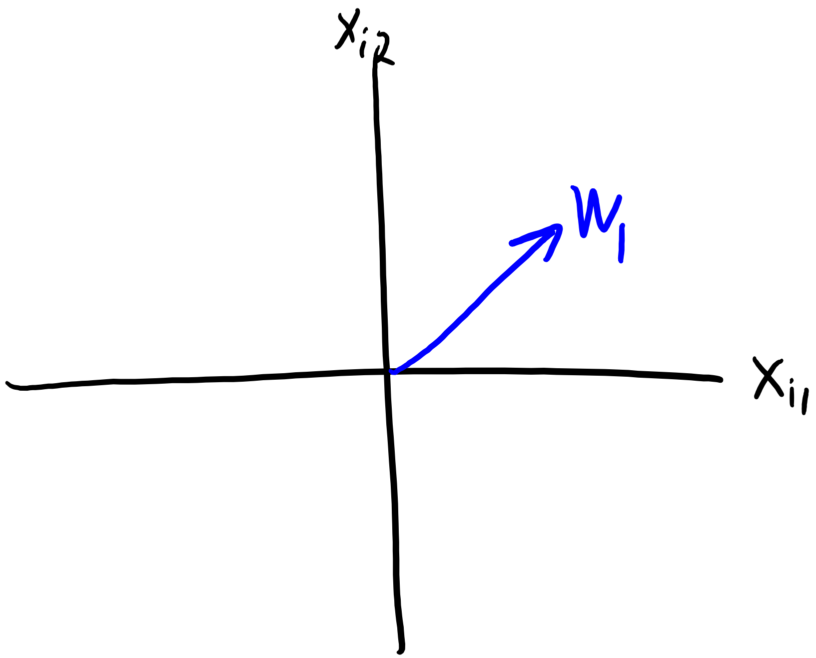
Non-Uniqueness of PCA

- Unlike k-means, we can efficiently find global optima of f(W,Z).
  - Algorithms coming later.
- Unfortunately, PCA never has a unique global optimum.
  - Several different sources of non-uniqueness (coming up soon)
- To understand these, we'll use "span" from linear algebra.
  - Helps explain the geometry of PCA.
  - Some global optima may be better than others. (coming up soon)

$$W, Z \in \underset{W,Z}{\operatorname{argmin}} f(W,Z)$$

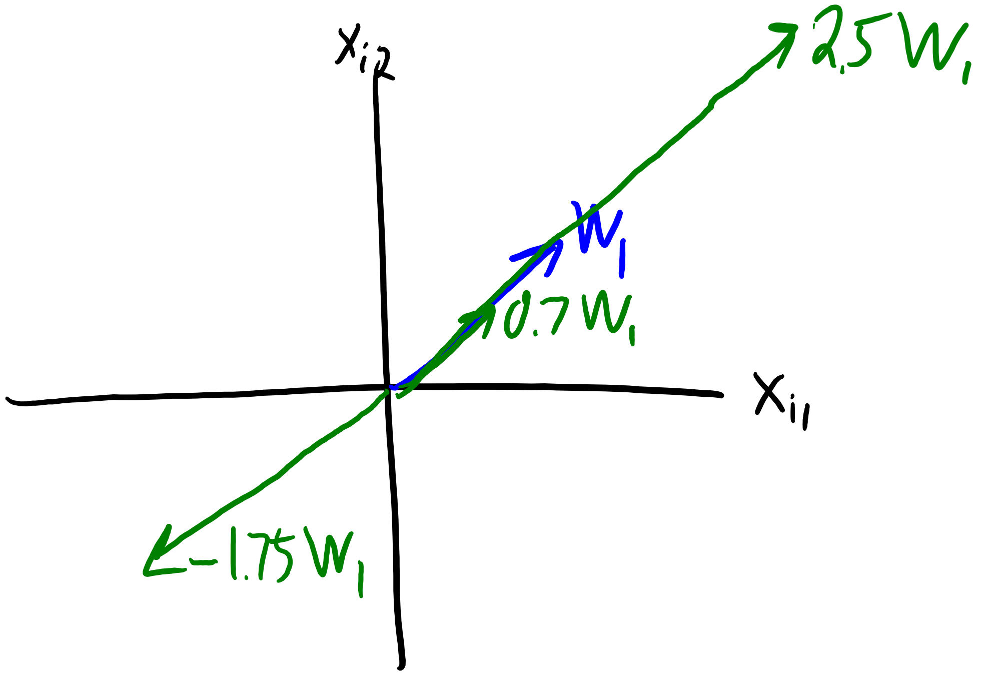
Span of 1 Vector

- Consider a single vector w_1 (k=1).



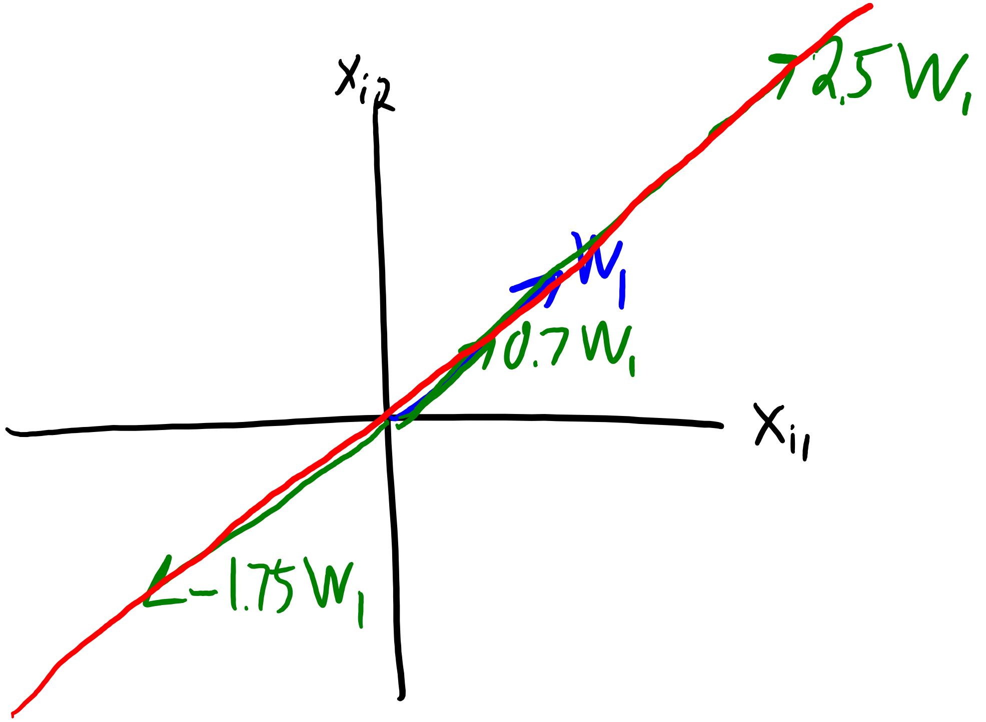
Span of 1 Vector

- Consider a single vector w_1 (k=1).
- The span(w_1) is all vectors of the form $z_i w_1$ for a scalar z_i.



Span of 1 Vector

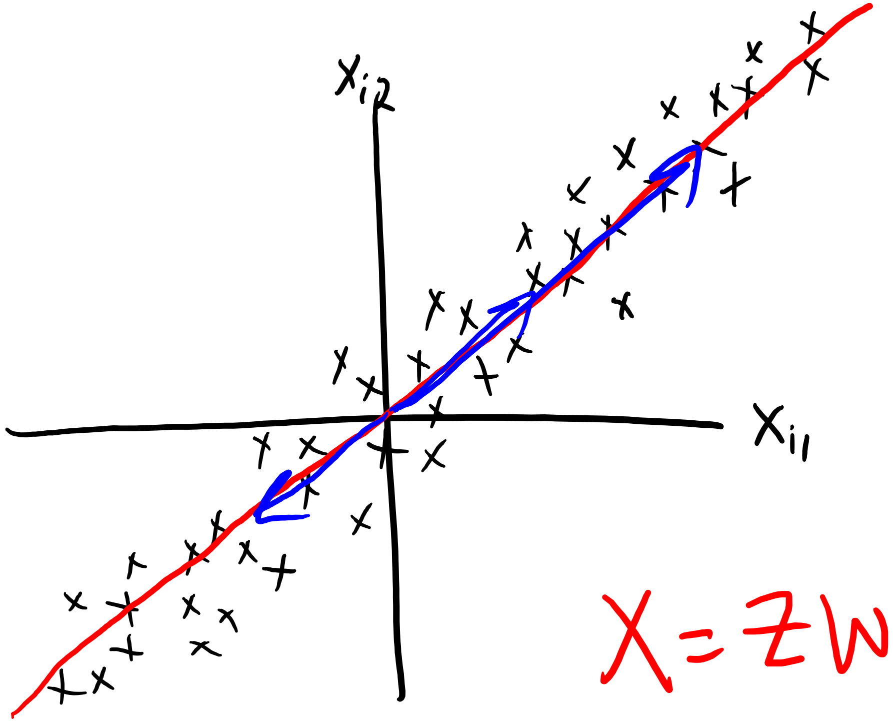
- Consider a single vector w_1 (k=1).
- The span(w_1) is all vectors of the form $z_i w_1$ for a scalar z_i.



- If $w_1 \neq 0$, this forms a line.

Span of 1 Vector

- Span of many different vectors gives same line.
  - Mathematically: αw_1 defines the same line as w_1 for any scalar $\alpha \neq 0$.



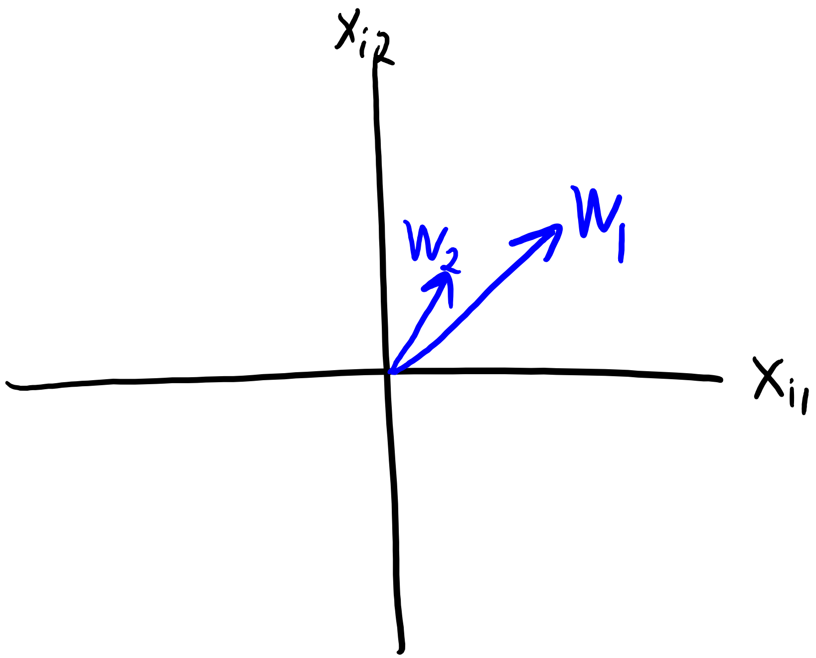
$$X = ZW = \frac{1}{\cancel{\alpha}} Z \cdot \cancel{\alpha} W$$

  - PCA solution is non-unique: (solution * scalar) is still PCA solution.
    - If (W,Z) is a solution, then $(\alpha W,(1/\alpha)Z)$ is also a solution.

$$\|(\alpha W)(\tfrac{1}{\alpha}Z) - X\|_F^2 = \|WZ - X\|_F^2$$

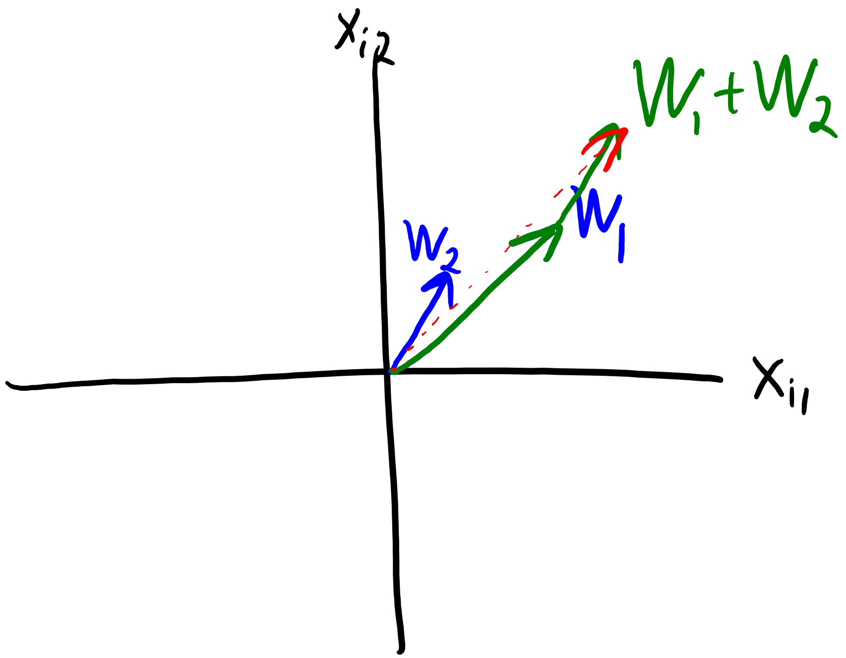
Span of 2 Vectors

- Consider two vectors w_1 and w_2 ($k=2$).



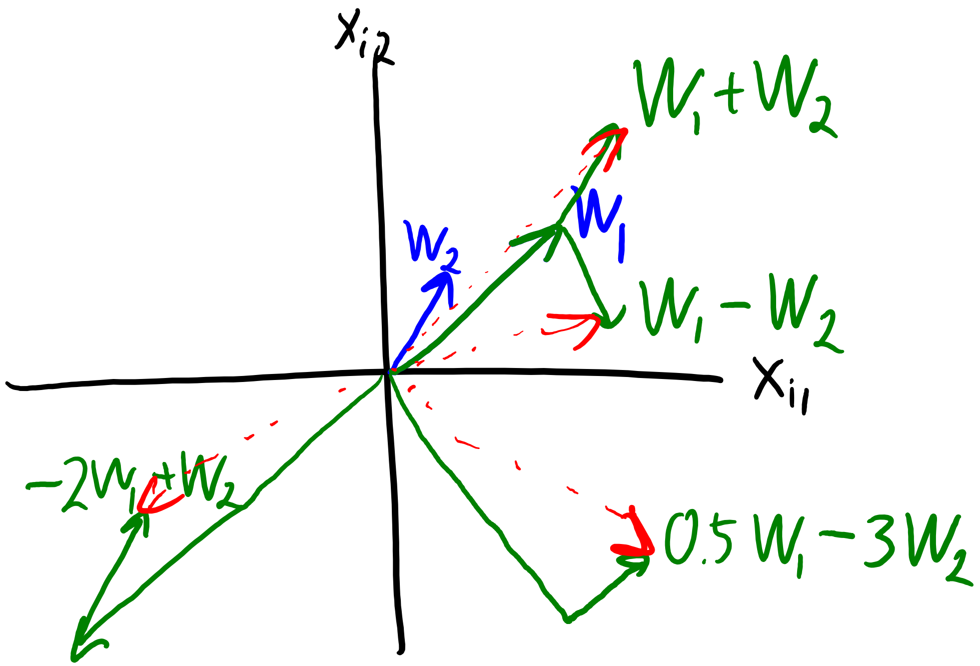
Span of 2 Vectors

- Consider two vectors w_1 and w_2 (k=2).
  - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for any scalars z_{i1} and z_{i2}.



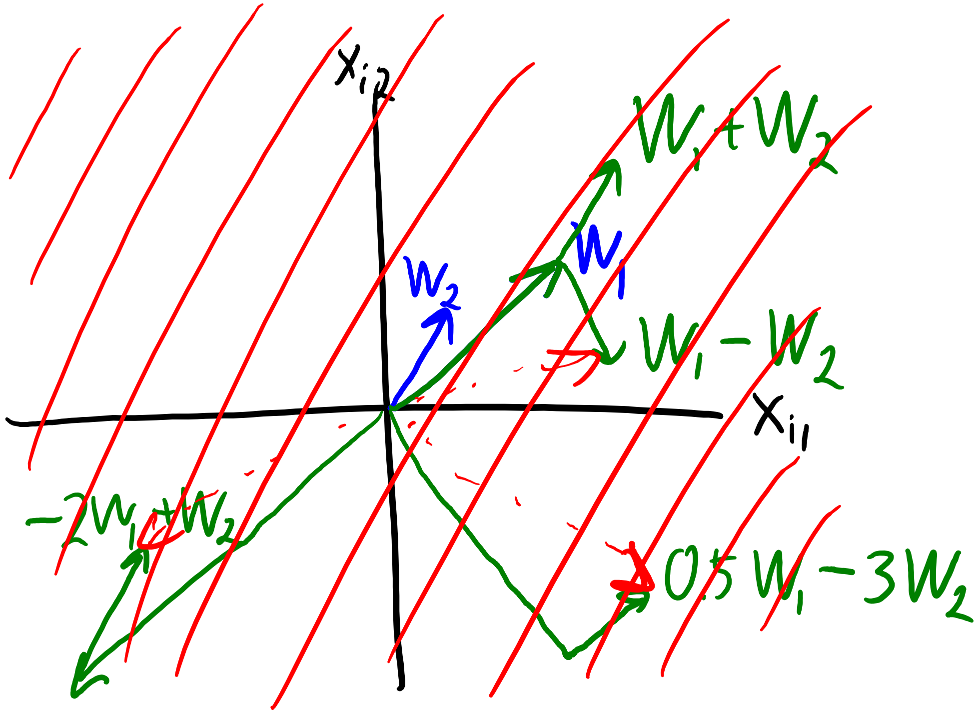
Span of 2 Vectors

- Consider two vectors w_1 and w_2 (k=2).
  - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for any scalars z_{i1} and z_{i2}.



Span of 2 Vectors

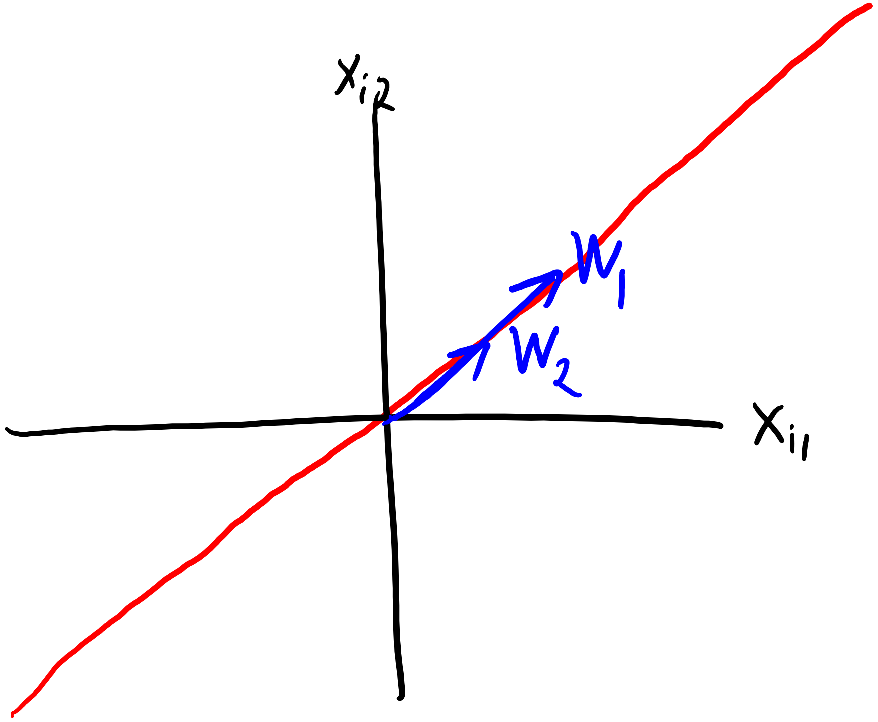
- Consider two vectors w_1 and w_2 ($k=2$).
  - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for any scalars z_{i1} and z_{i2}.



  - For most non-zero 2d vectors, span(w_1,w_2) is a plane.
    - In the case of two vectors in R^2, the plane will be *all* of R^2.

Span of 2 Vectors

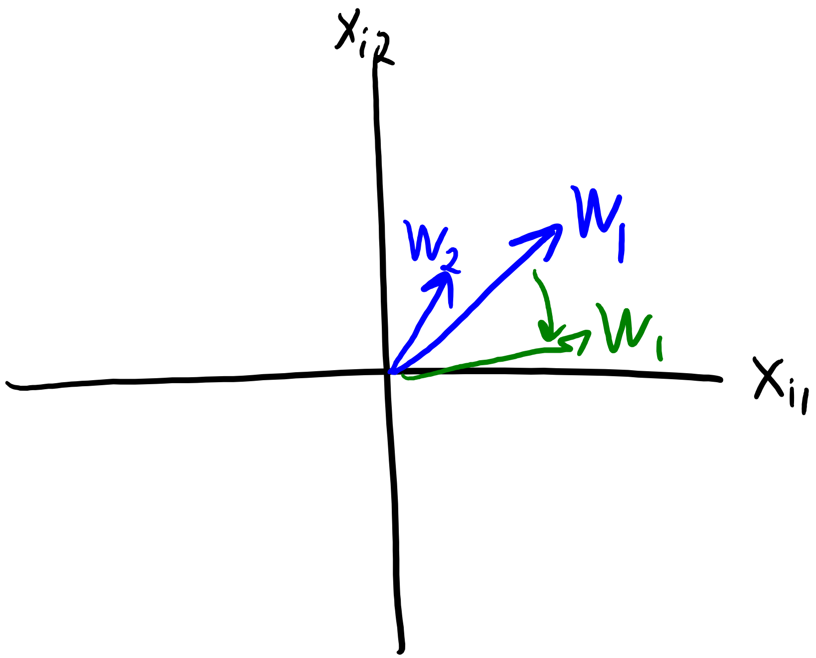
- Consider two vectors w_1 and w_2 (k=2).
  - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for any scalars z_{i1} and z_{i2}.



  - For most non-zero 2d vectors, span(w_1,w_2) is plane.
    - Exception is if w_2 is in span of w_1 ("collinear"), then span(w_1,w_2) is just a line.

Span of 2 Vectors

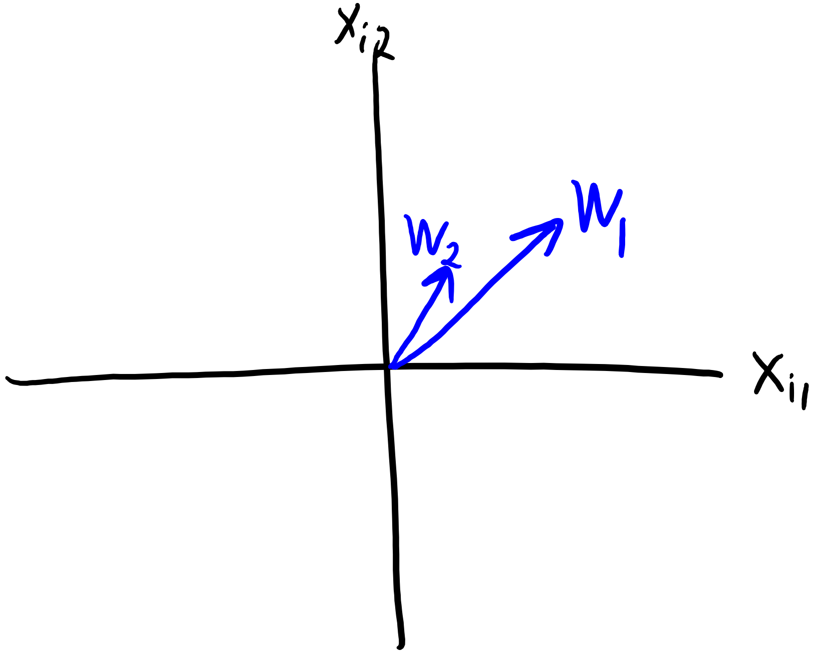
- Consider two vectors w_1 and w_2 (k=2).
  - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for any scalars z_{i1} and z_{i2}.



  - New issues for PCA (k >= 2):
    - We have label switching: span(w_1,w_2) = span(w_2,w_1).
    - We can rotate factors within the plane (if not rotated to be collinear).

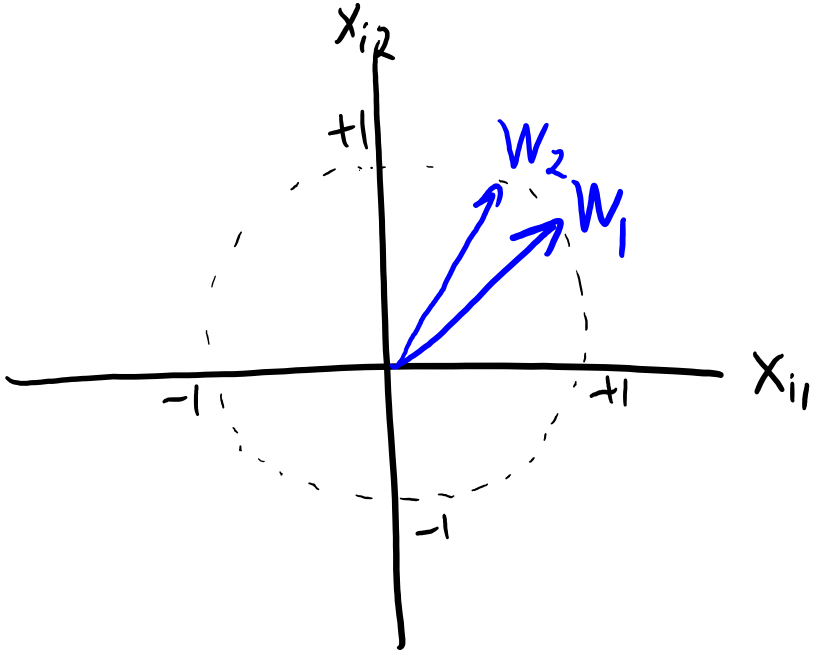
Span of 2 Vectors

- 2 tricks to make vectors defining a plane "more unique":
  - Normalization: enforce that $||w_1|| = 1$ and $||w_2|| = 1$.



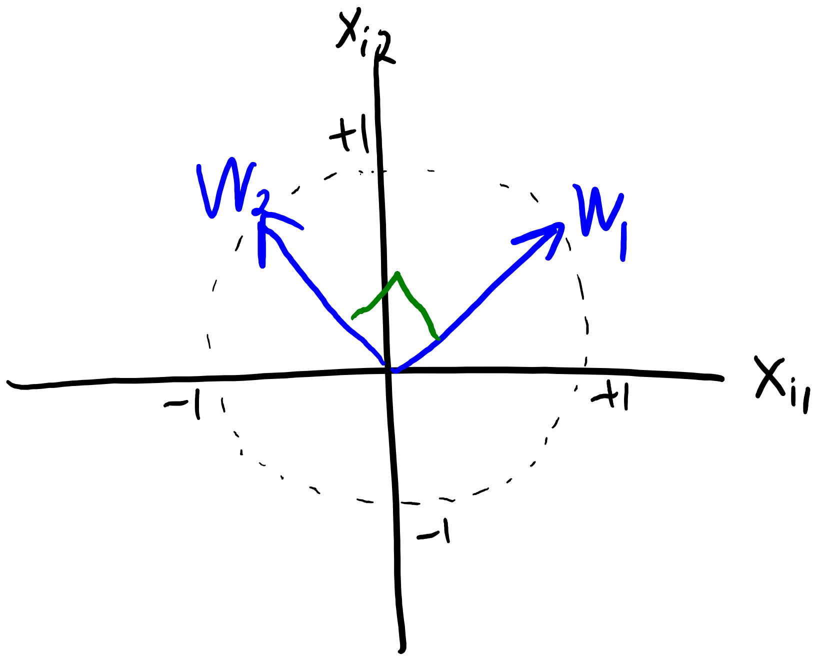
Span of 2 Vectors

- 2 tricks to make vectors defining a plane “more unique”:
  - Normalization: enforce that $||w_1|| = 1$ and $||w_2|| = 1$.



Span of 2 Vectors

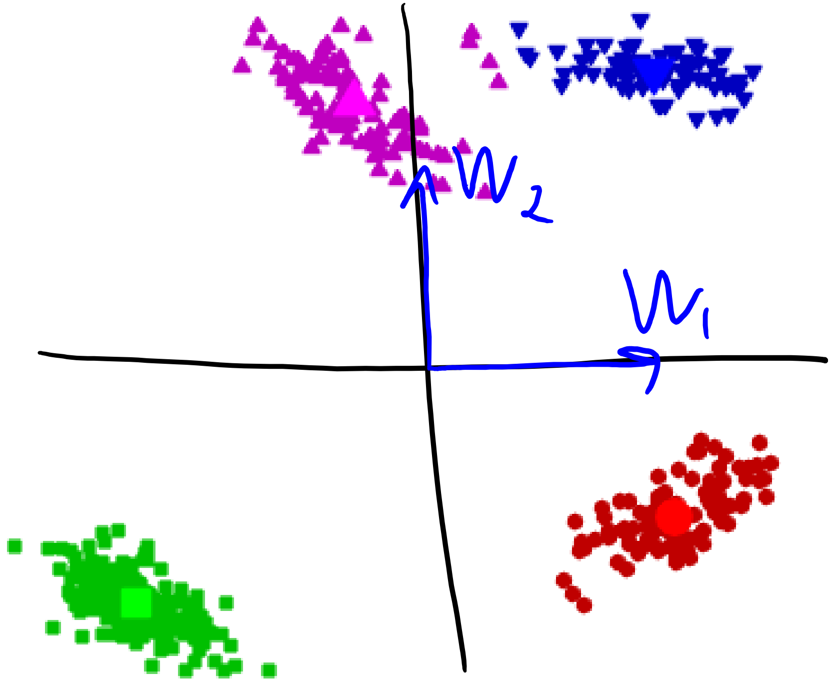
- 2 tricks to make vectors defining a plane "more unique":
  - Normalization: enforce that $||w_1|| = 1$ and $||w_2|| = 1$.
  - Orthogonality: enforce that $w_1^T w_2 = 0$ ("perpendicular").



  - Can't grow/shrink vectors (though I can still reflect).
  - Can't rotate one vector (but I can still rotate *both*).

Digression: PCA only makes sense for k ≤ d

- Remember our clustering dataset with 4 clusters:



- It doesn't make sense to use PCA with k=4 on this dataset.
  - Only need two vectors [1 0] and [0 1] to exactly represent all 2d points.
    - With k=2, set Z=X and W=I to get X=ZW exactly.

Span in Higher Dimensions

- In higher-dimensional spaces:
  - Span of 1 non-zero vector w_1 is a line.
  - Span of 2 non-zero vectors w_1 and w_2 is a plane (if not collinear).
    - Can be visualized as a 2D plot.
  - Span of 3 non-zeros vectors $\{w_1, w_2, w_3\}$ is a 3d space (if not "coplanar").
  - …

$$\begin{array}{cc} k & \begin{bmatrix} - w_1 - \\ - w_2 - \\ \\ \end{bmatrix} \\ & d \end{array}$$

- This is how the W matrix in PCA defines lines, planes, spaces, etc.
  - Each time we increase 'k', we add an extra "dimension" to the "subspace".

Making PCA Unique

- We've identified several reasons that optimal W is non-unique:
  - Multiply any w_c by any non-zero scalar.
  - Rotate any w_c almost arbitrarily within the span.
  - Switch any w_c with any other $w_{c'}$.

$k \leftarrow$ # components $d \leftarrow$ # features

- PCA implementations add constraints to make solution unique:
  - Normalization: enforce that $||w_c|| = 1$.
  - Orthogonality: enforce that $w_c^T w_{c'} = 0$ for all $c \neq c'$.
  - Sequential fitting: First fit w_1 ("first principal component") giving a line.
    - Then fit w_2 given w_1 ("second principal component") giving a plane.
    - Then we fit w_3 given w_1 and w_2 ("third principal component") giving a space.

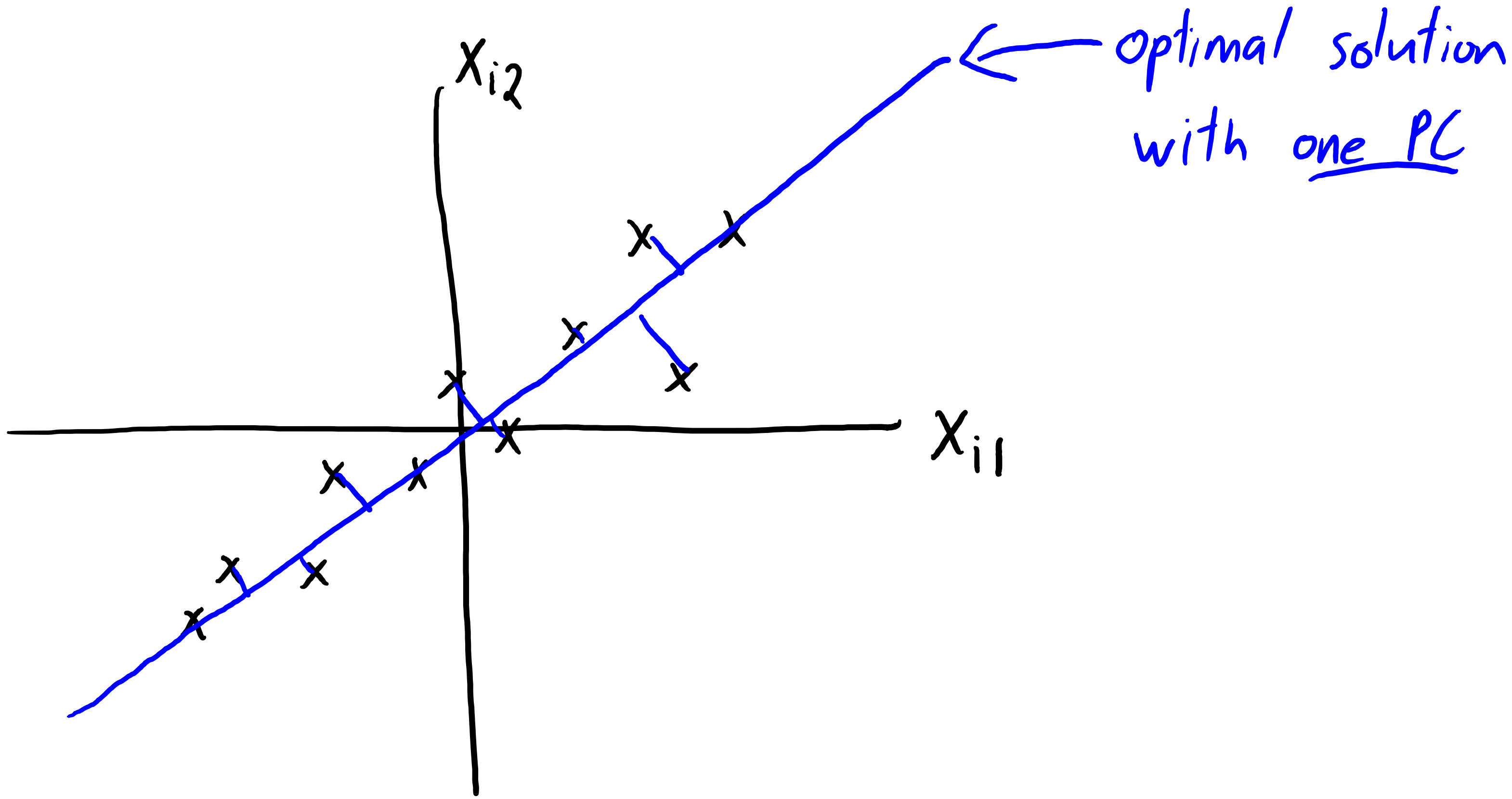
$$\begin{bmatrix} - w_1 - \\ - w_2 - \\ - w_3 - \end{bmatrix} \text{ orthogonal.}$$

$$\underbrace{W}_{k \times d} \underbrace{W^T}_{d \times k} = \underbrace{I}_{k \times k}$$

Coming Up Next

SEQUENTIAL FITTING AND SVD

Basis, Orthogonality, Sequential Fitting

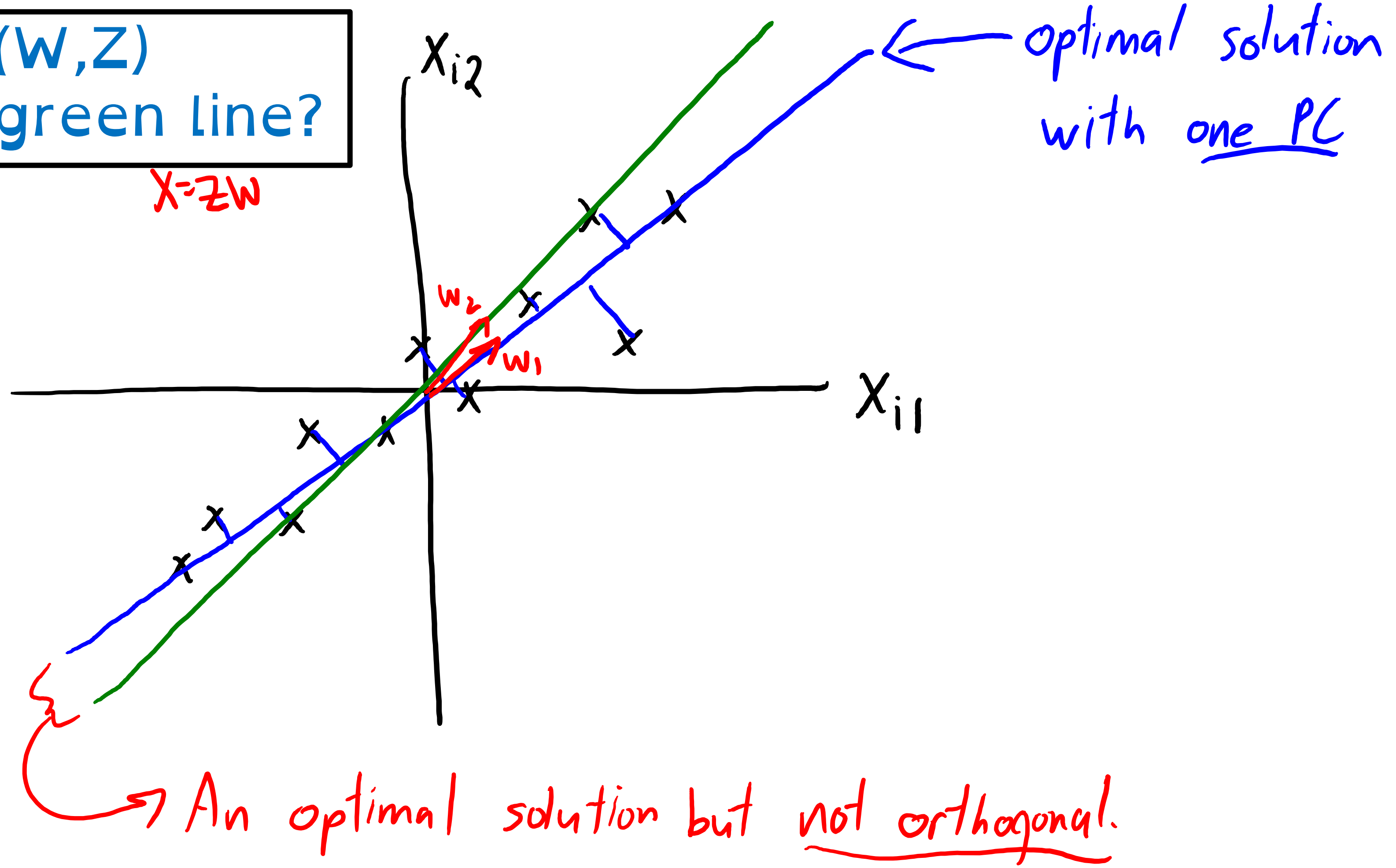


Basis, Orthogonality, Sequential Fitting

Q: What is f(W,Z)
when we add the green line?

=0

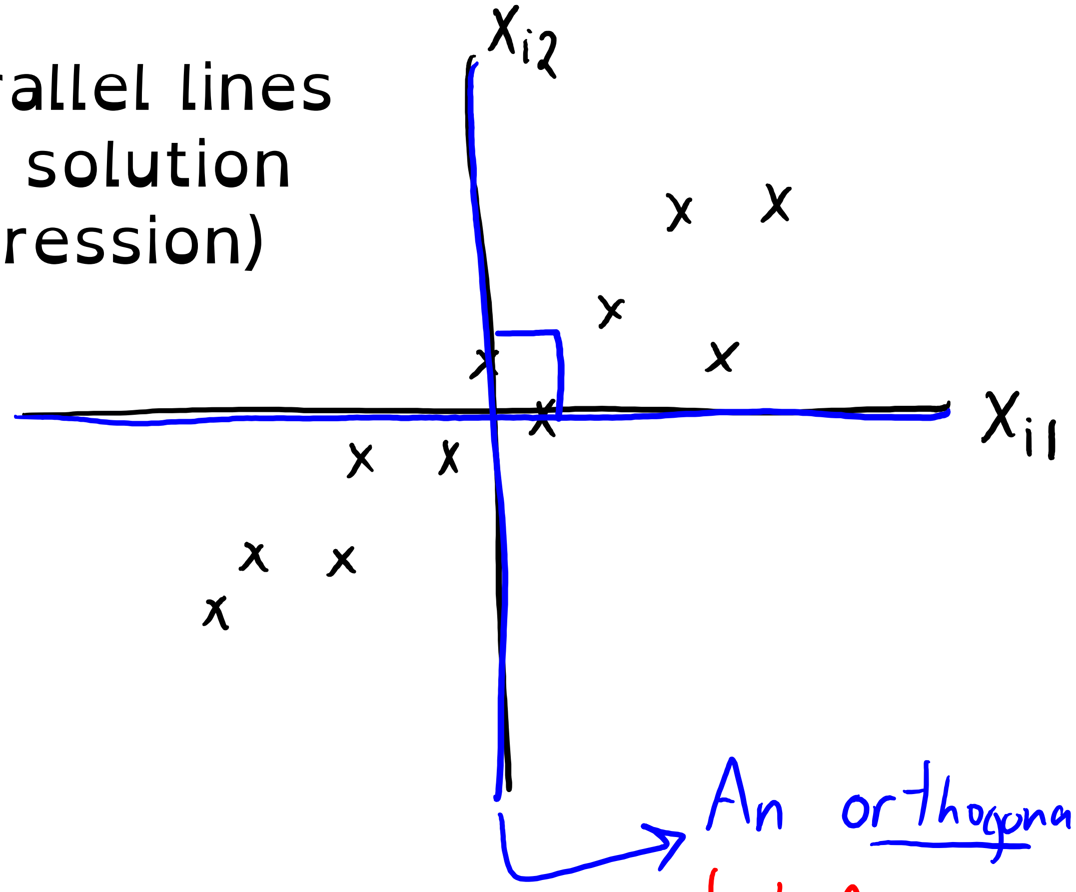
X=ZW



(both PCs give similar information)

Basis, Orthogonality, Sequential Fitting

Any two non-parallel lines give an optimal solution (lossless compression)

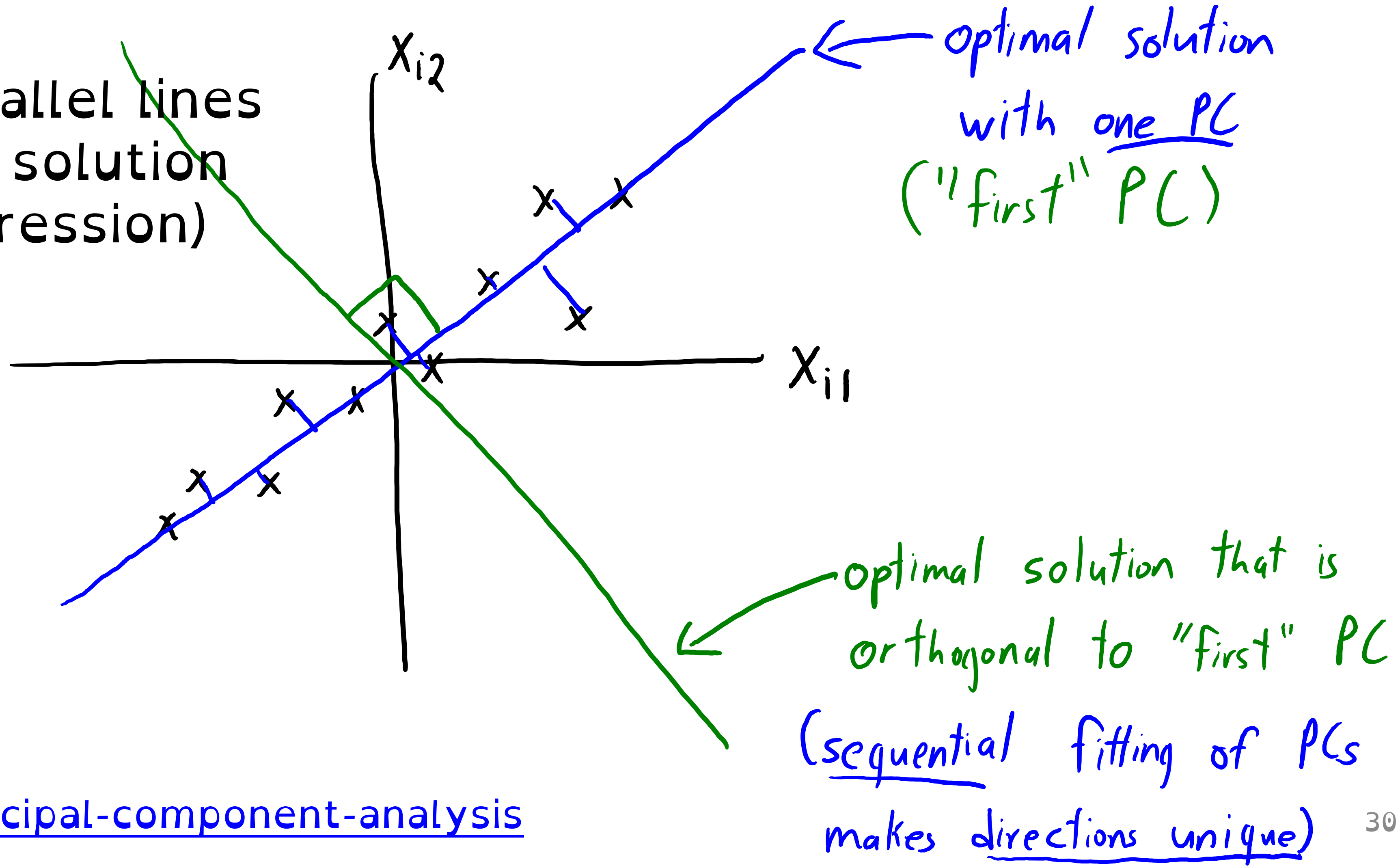


An orthogonal solution (PCs are not redundant)

but PCs have nothing to do with data

Basis, Orthogonality, Sequential Fitting

Any two non-parallel lines give an optimal solution (lossless compression)



http://setosa.io/ev/principal-component-analysis

PCA Computation: SVD

- How do we fit with normalization/orthogonality/sequential-fitting?
  - It can be done with the "singular value decomposition" (SVD).
  - Take CPSC 302 or MATH 307

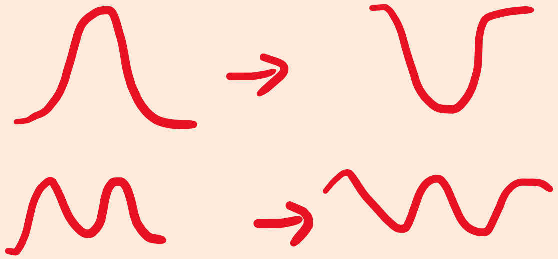
- 4 lines of Python code:
  - `mu = np.mean(X,axis=0)`
  - `X -= mu`
  - `U, s, Vh = np.linalg.svd(X)`
  - `W = Vh[:k, :]`

- Computing Z is cheaper now:

$$Z = XW^T(\underbrace{WW^T}_{})^{-1} = XW^T$$

$$WW^T = \begin{bmatrix} — w_1 — \\ — w_2 — \\ \vdots \\ — w_k — \end{bmatrix} \begin{bmatrix} | & | & & | \\ w_1^T & w_2^T & \cdots & w_k^T \\ | & | & & | \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & 0 \\ 0 & & \ddots & & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix} = I$$



Coming Up Next

ALTERNATING MINIMIZATION

PCA Computation

- With linear regression, we had the normal equations
  – But we also could do it with gradient descent, SGD, etc.
- With PCA we have the SVD
  – But we can also do it with gradient descent, SGD, etc.

- Gradient-based methods don't enforce uniqueness
  – Sensitive to initialization, don't enforce normalization, orthogonality, ordered PCs.
  – But you can do this in post-processing if you want.
- Why would we want this? We can use our tricks from Part 3 of the course:
  – We can do things like "robust" PCA (A6), "regularized" PCA, "sparse" PCA, "binary" PCA.
  – We can fit huge datasets where SVD is too expensive.

PCA Computation: Alternating Minimization

- With centered data, the PCA objective is:

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

- In k-means we optimized this with alternating minimization:
  - Fix "cluster assignments" Z and find the optimal "means" W.
  - Fix "means" W and find the optimal "cluster assignments" Z.

- Converges to a local optimum.
  - But may not find a global optimum (sensitive to initialization).

PCA Computation: Alternating Minimization

- With centered data, the PCA objective is:

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

- In PCA we can also use alternating minimization:
  - Fix "scores" Z, find optimal "factors" W.
  - Fix "factors" W, find optimal "scores" Z.

- Converges to a local optimum.
  - Which will be a global optimum (if we randomly initialize W and Z).

PCA Computation: Alternating Minimization

- With centered data, the PCA objective is:

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

- Alternating minimization steps:
  - If we fix Z, this is a quadratic function of W (least squares column-wise):

$$\nabla_W f(W,Z) = Z^TZW - Z^TX \quad \text{so} \quad W = (\underline{Z^TZ})^{-1}(Z^TX)$$

(writing gradient as a matrix)

  - If we fix W, this is a quadratic function of Z (transpose due to dimensions):

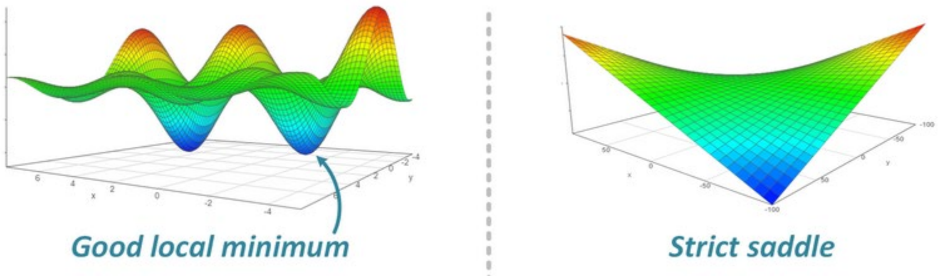
$$\nabla_Z f(W,Z) = ZWW^T - XW^T \quad \text{so} \quad Z = XW^T(\underline{WW^T})^{-1}$$

These are usually invertible since $k \ll n$ and $k \ll d$

PCA Computation: Alternating Minimization

- With centered data, the PCA objective is:

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



- This objective is not jointly convex in W and Z.
  - You will find different W and Z depending on the initialization.
    - For example, if you initialize with all $w_c = 0$, then they will stay at zero.

  - But it's possible to show that all "stable" local optima are global optima.
    - You will converge to a global optimum in practice if you initialize randomly.
      - Randomization means you don't start on one of the unstable non-global critical points.
    - E.g., sample each initial z_{ij} from a normal distribution.

http://www.offconvex.org/2018/11/07/optimization-beyond-landscape/

PCA Computation: Stochastic Gradient

- For big X matrices, you can also use stochastic gradient:

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

On each iteration, pick a random example 'i' and feature 'j':

→ Set w^j to $w^j - \alpha^t \nabla_{w^j} f(w^j, z, x_{ij})$

→ Set z_i to $z_i - \alpha^t \nabla_{z_i} f(w^j, z_i, x_{ij})$

- Other variables stay the same, cost per iteration is only O(k).

PCA Computation: Prediction

- At the end of training, the "model" is the μ_j and the W matrix.
  - PCA is parametric.
- PCA prediction phase:
  - Given new data $\tilde{X}$, we can use μ_j and W this to form $\tilde{Z}$:

$\tilde{X}$: t×d, W: k×d

1. Center: replace each $\tilde{x}_{ij}$ with $(\tilde{x}_{ij} - \mu_j)$

μ_j: means of training data

2. Find $\tilde{Z}$ minimizing squared error:

$$\underset{t\times k}{\tilde{Z}} = \underset{t\times d}{\tilde{X}} \underbrace{\underset{d\times k}{W^T} (\underbrace{\underset{k\times d}{W}\,\underset{d\times k}{W^T}}_{k\times k})^{-1}}_{\text{(could just store this } d\times k \text{ matrix)}}$$

PCA Computation: Prediction

- At the end of training, the "model" is the μ_j and the W matrix.
  - PCA is parametric.
- PCA prediction phase:
  - Given new data $\tilde{X}$, we can use μ_j and W this to form $\tilde{Z}$:
  - The "reconstruction error" is how close approximation is to $\tilde{X}$:

$$\|\underbrace{\tilde{Z}W}_{\hat{X}} - \tilde{X}\|_F^2$$

centered version

  - Our "error" from replacing the x_i with the z_i and W.

Choosing ‘k’ by “Variance Explained”

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

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

definition of variance

Sample mean to estimate $E[\cdot]$

Frobenius norm

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

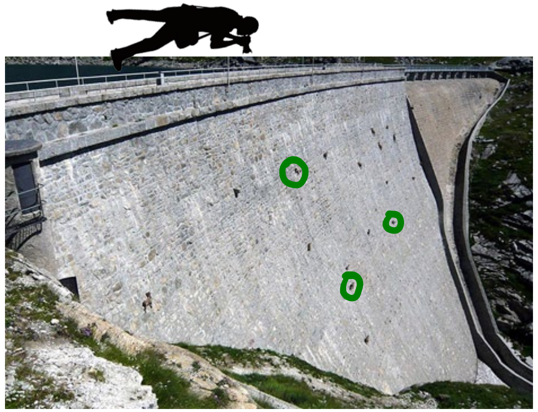
$$\frac{\|ZW - X\|_F^2}{\|X\|_F^2}$$

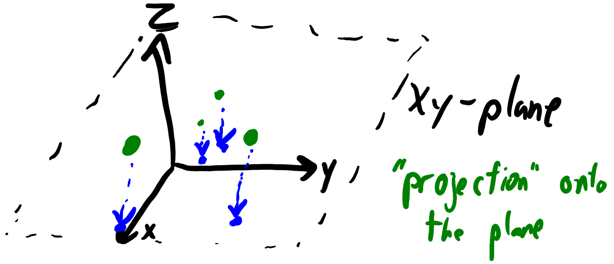
centered version

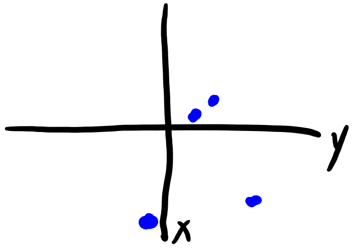
  - Gives a number between 0 (k=d) and 1 (k=0), giving “variance remaining”.
    - If you want to “explain 90% of variance”, choose smallest ‘k’ where ratio is < 0.10.

"Variance Explained" in the Goat Situation

- Recall: Crazy goats:







- Interpretation of "variance remaining" formula:

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

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

$\|X\|_F^2$ ← Variance of goat pos. in 3-dimensions

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

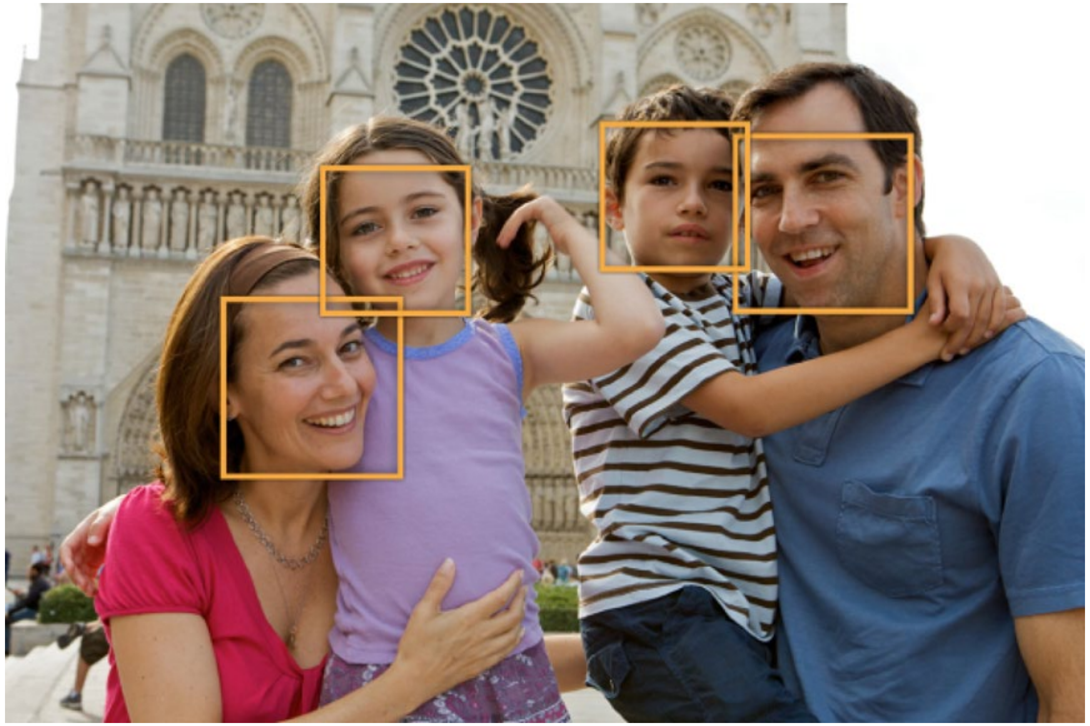
https://en.wikipedia.org/wiki/Doom_(1993_video_game)
https://forum.minetest.net/viewtopic.php?f=5&t=9666

Coming Up Next

EIGENFACES

Application: Face Detection

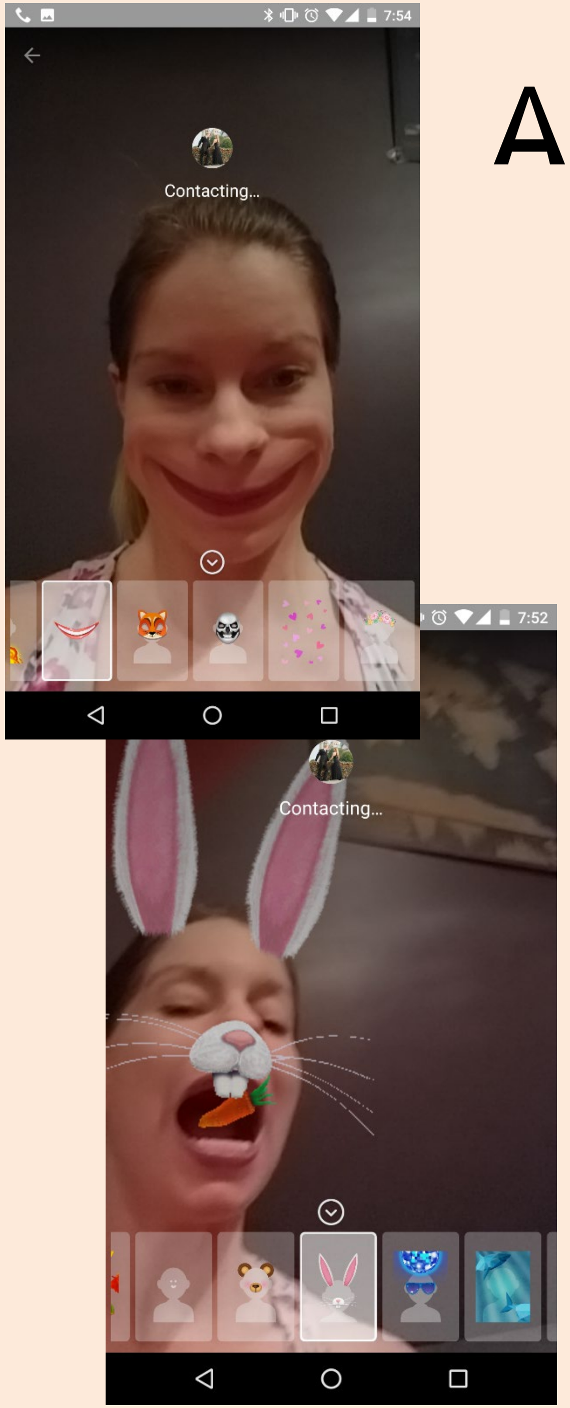
- Consider problem of face detection:

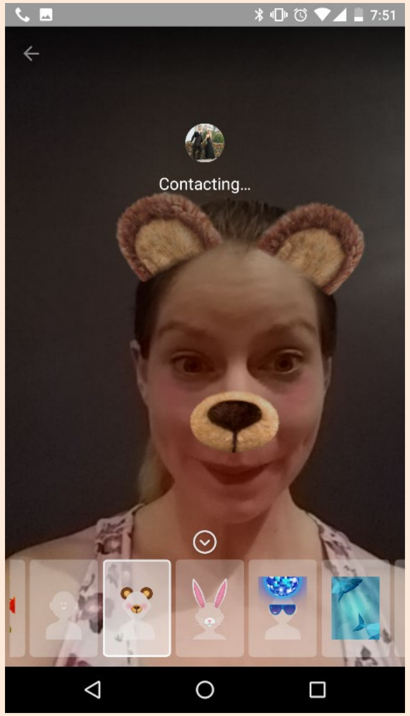


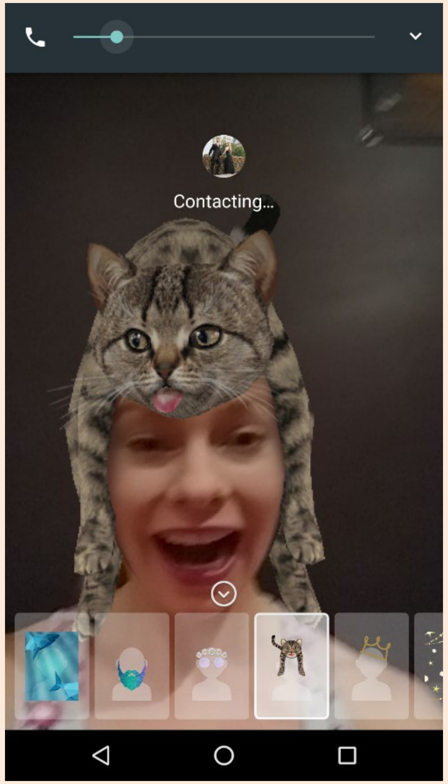
- Classic methods use “eigenfaces” as basis:
  - PCA applied to images of faces.

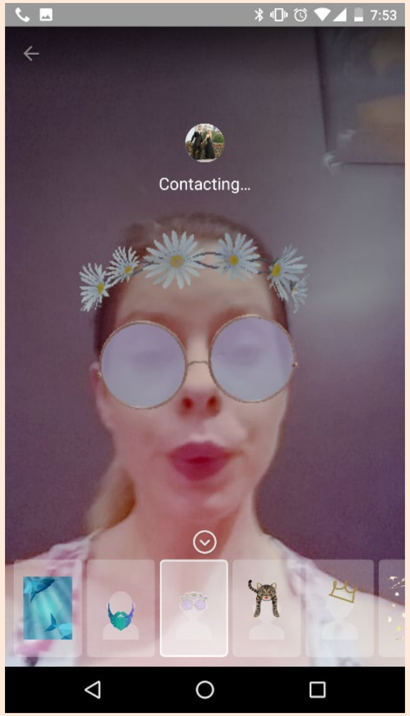
https://developer.apple.com/library/content/documentation/GraphicsImaging/Conceptual/CoreImaging/ci_detect_faces/ci_detect_faces.html

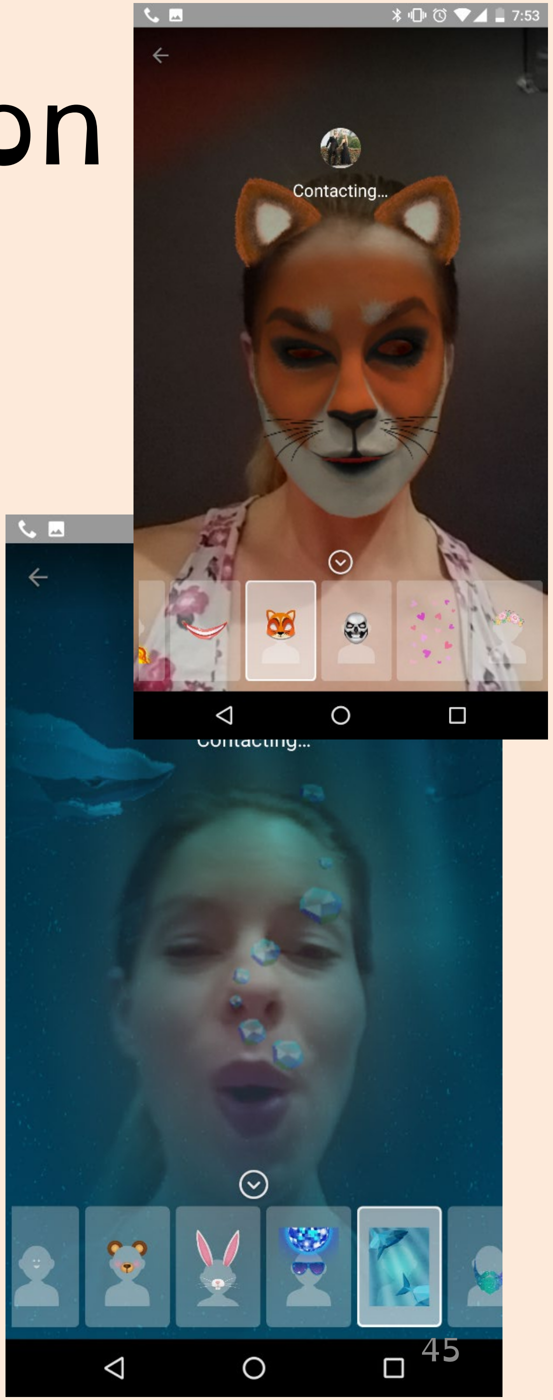
Application: Face Detection











Eigenfaces

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

Each row of X will be pixels in one 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.

“canonical face” according to data

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

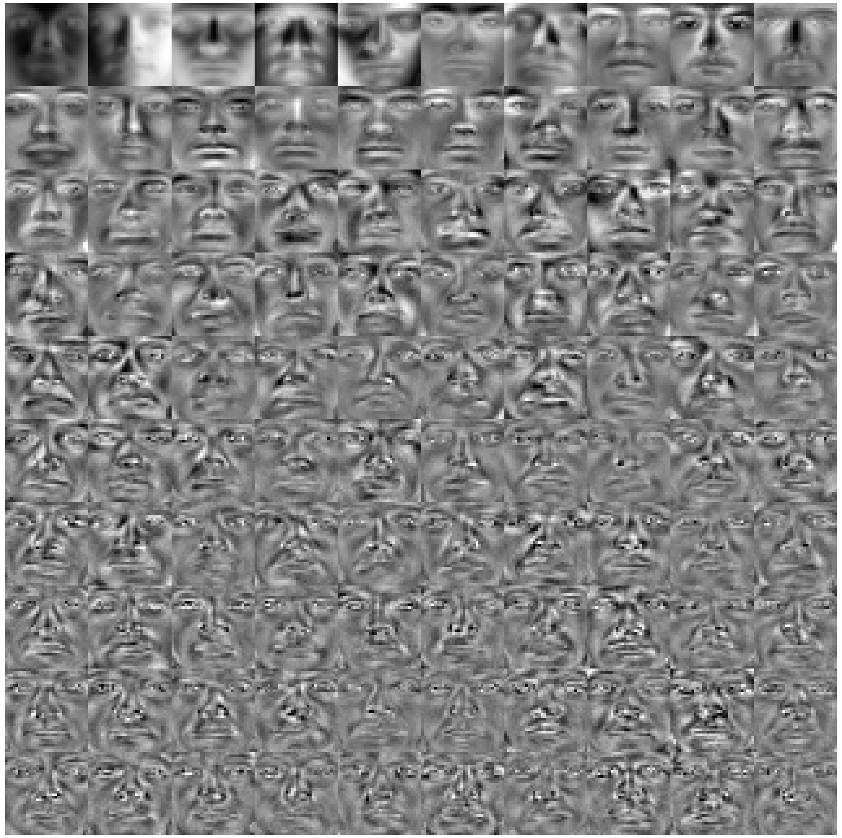
Each row of X will be pixels in <u>one</u> image:

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

Eigenfaces

$$\underset{k \times d}{W} = \begin{bmatrix} \text{---} & w_1 & \text{---} \\ \text{---} & v_2 & \text{---} \\ & \vdots & \end{bmatrix}$$

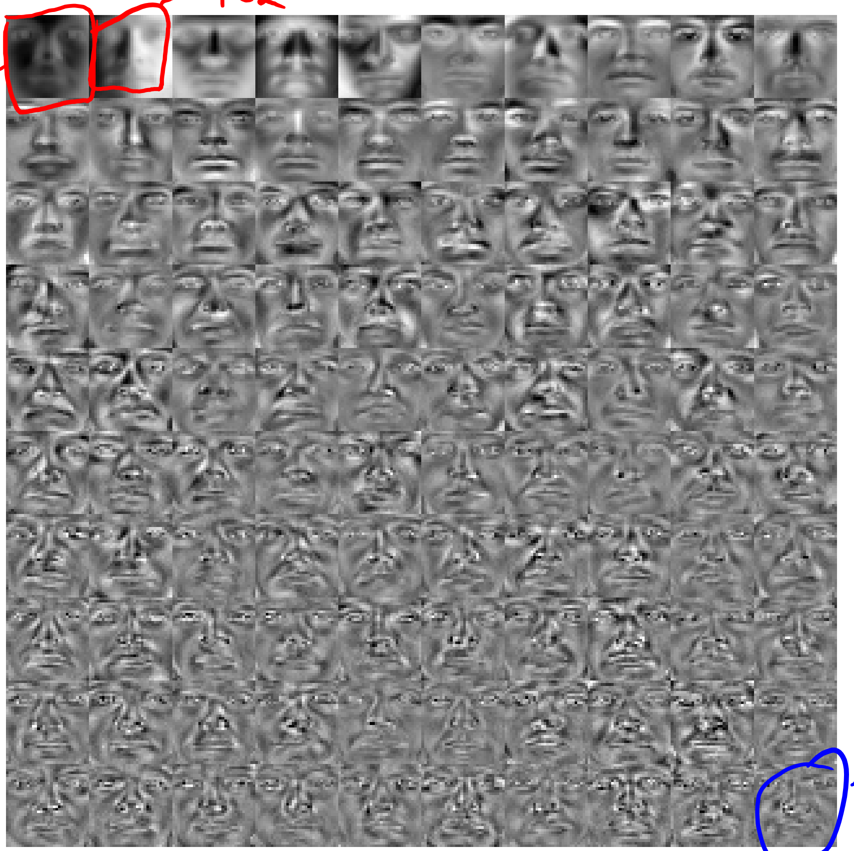
Compute top 'k' PCs on centered data: Each row of X will be pixels in <u>one</u> image:



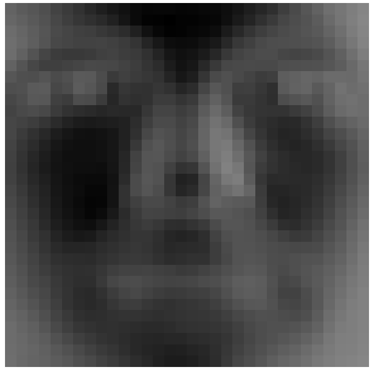
$$X = \begin{bmatrix} \text{---} & x_1 - \mu & \text{---} \\ \text{---} & x_2 - \mu & \text{---} \\ & \vdots & \\ \text{---} & x_n - \mu & \text{---} \end{bmatrix}$$

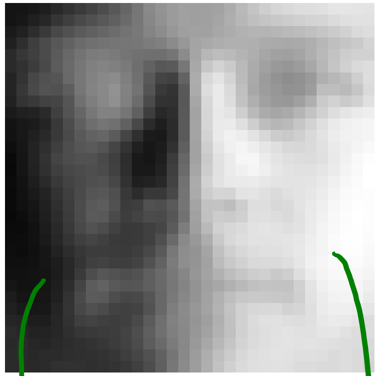
Eigenfaces

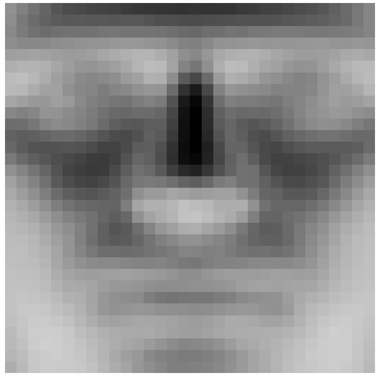
Compute top 'k' PCs on centered data:



Note that these are "signed" images.







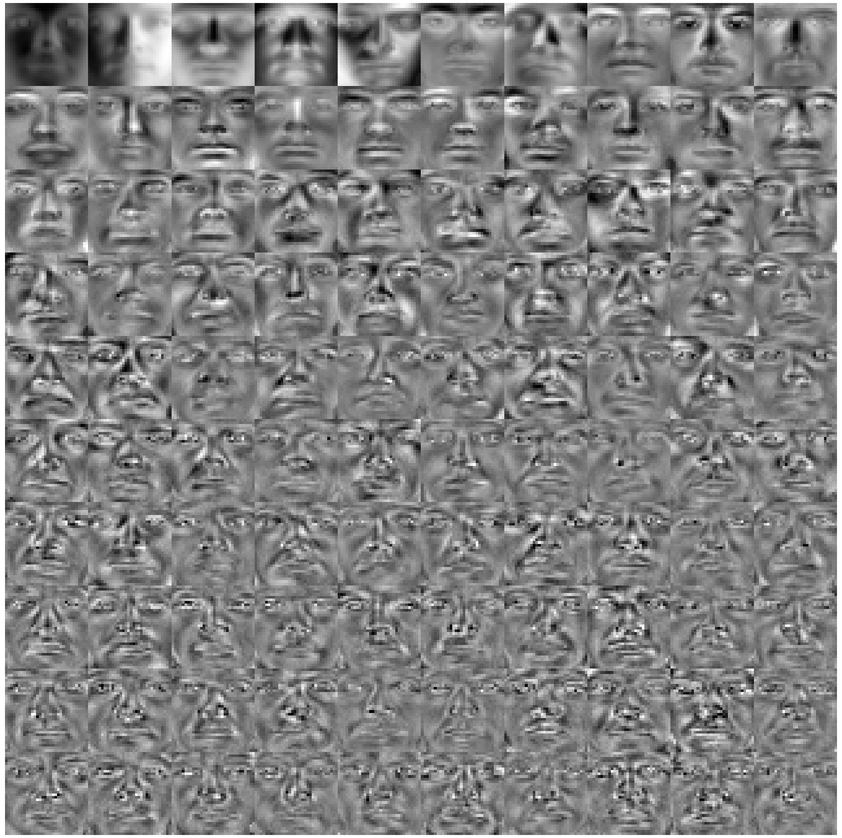
"gray" represents values close to 0.

"dark" represents negative values

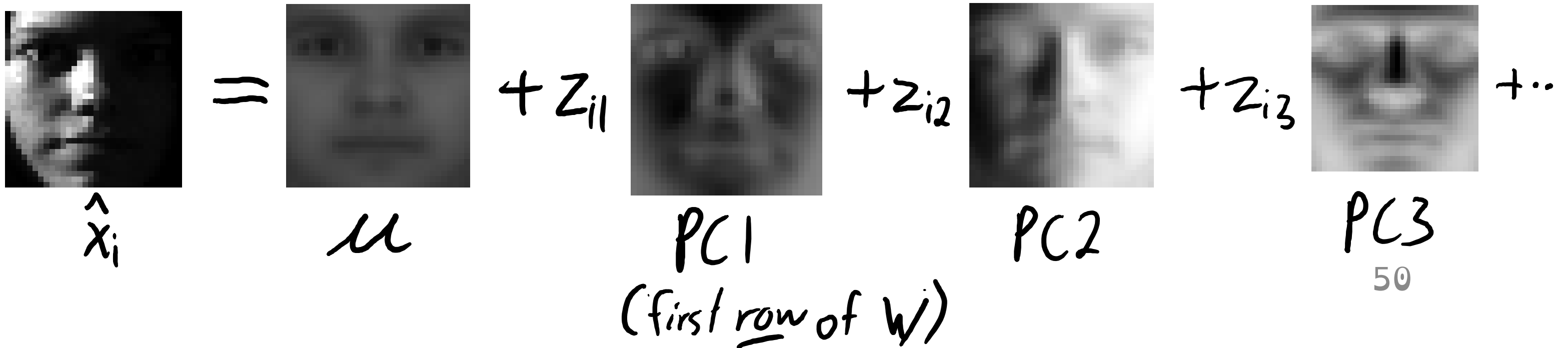
"bright" represents positive values

Eigenfaces

Compute top 'k' PCs on centered data:



"Eigenface" representation:



Eigenfaces

100 of the original faces:

"Eigenface" representation:

$\hat{x}_i$

$=$

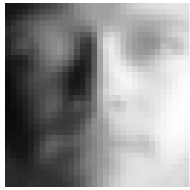
μ

$+ z_{i1}$

PC1

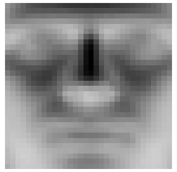
(first row of W)

$+ z_{i2}$



PC2

$+ z_{i3}$

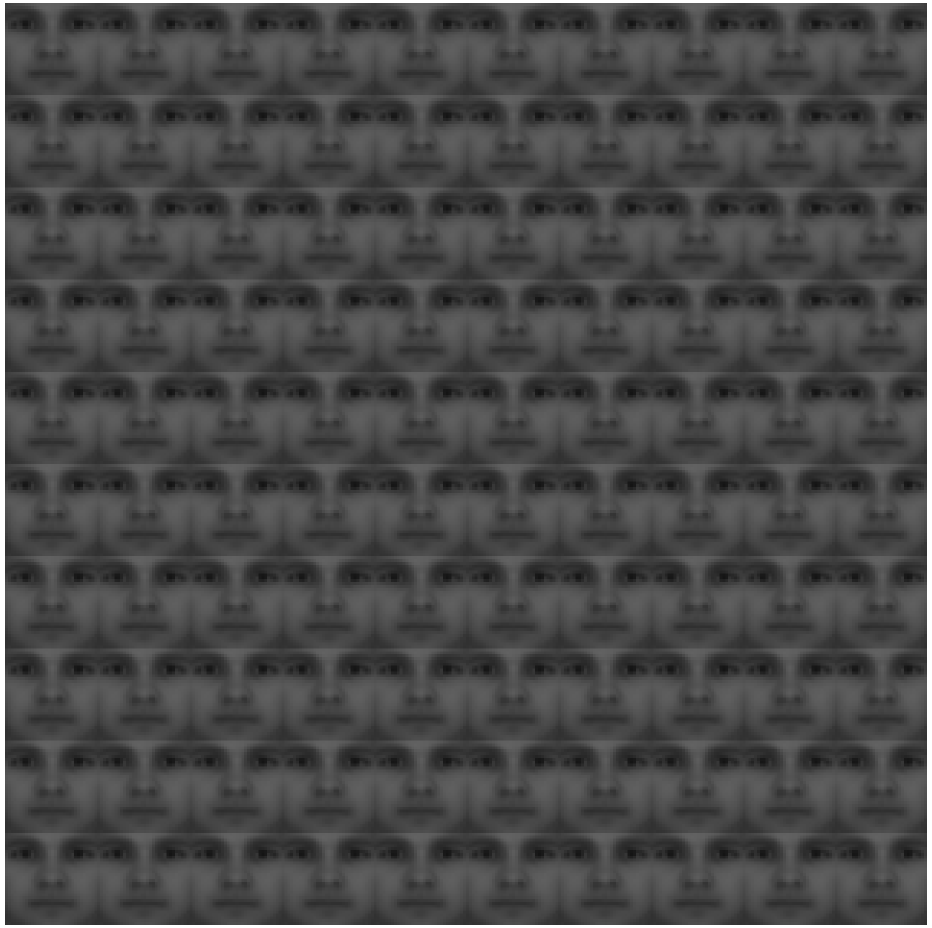


PC3

$+ \dots$

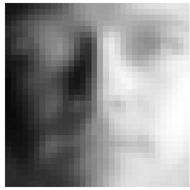
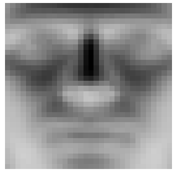
Eigenfaces

Reconstruction with $k = 0$



Variance explained: 0%

"Eigenface" representation:

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

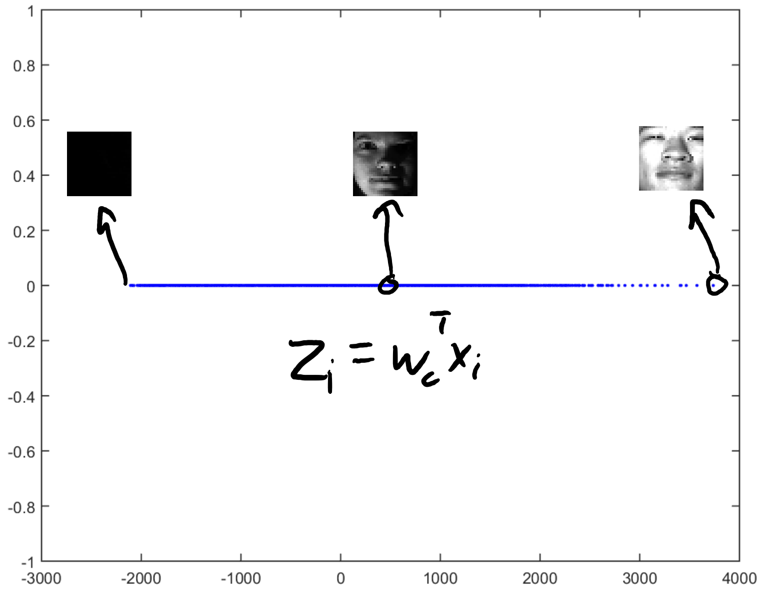
$\hat{x}_i$ $\qquad$ μ $\qquad$ PC1 (first row of W) $\qquad$ PC2 $\qquad$ PC3

Eigenfaces

Reconstruction with $k=1$

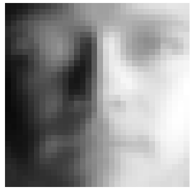
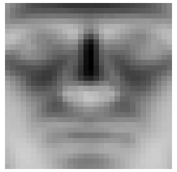
Variance explained: 36%

PCA Visualization:



$z_i = w_c^T x_i$

"Eigenface" representation:

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

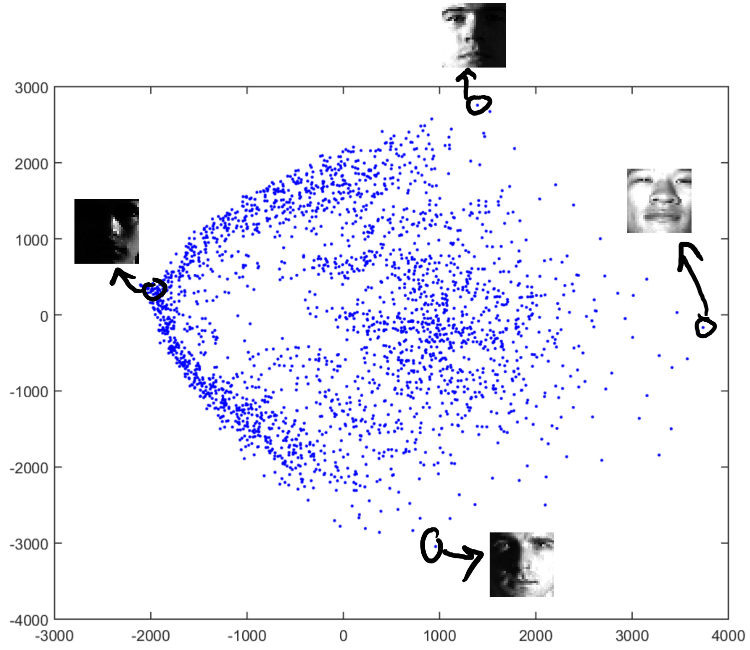
$\hat{x}_i$ $\quad \mu \quad$ PC1 (first row of W) $\quad$ PC2 $\quad$ PC3

Eigenfaces

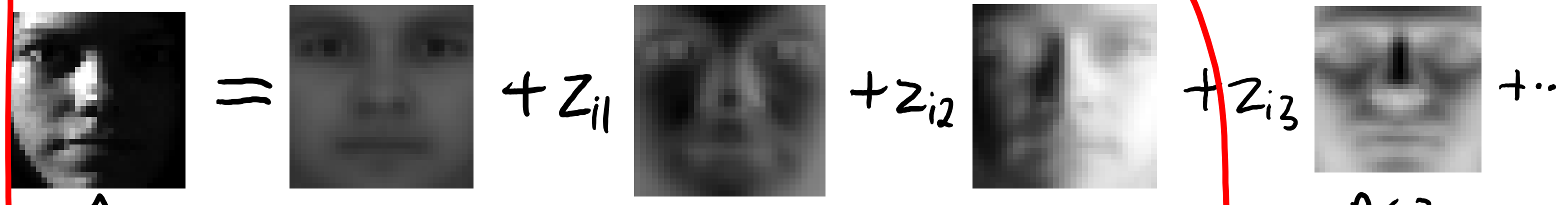
Reconstruction with $k=2$

Variance explained: 71%

PCA Visualization:



"Eigenface" representation:

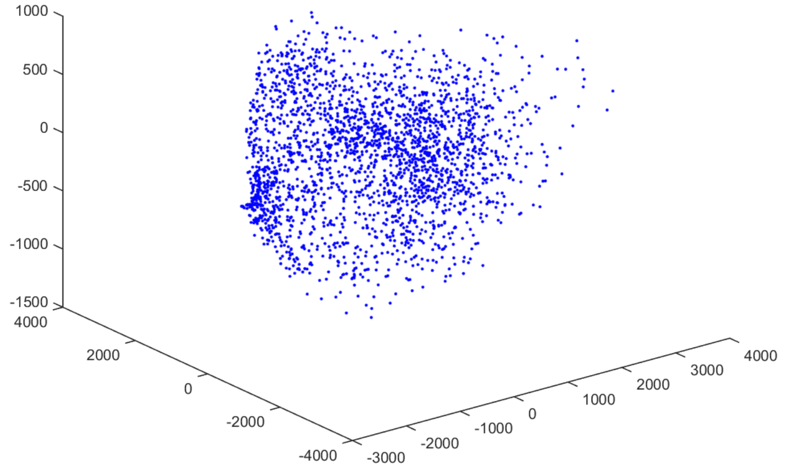


Eigenfaces

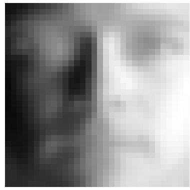
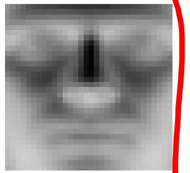
Reconstruction with $k=3$

Variance explained: 76%

PCA Visualization:



"Eigenface" representation:

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

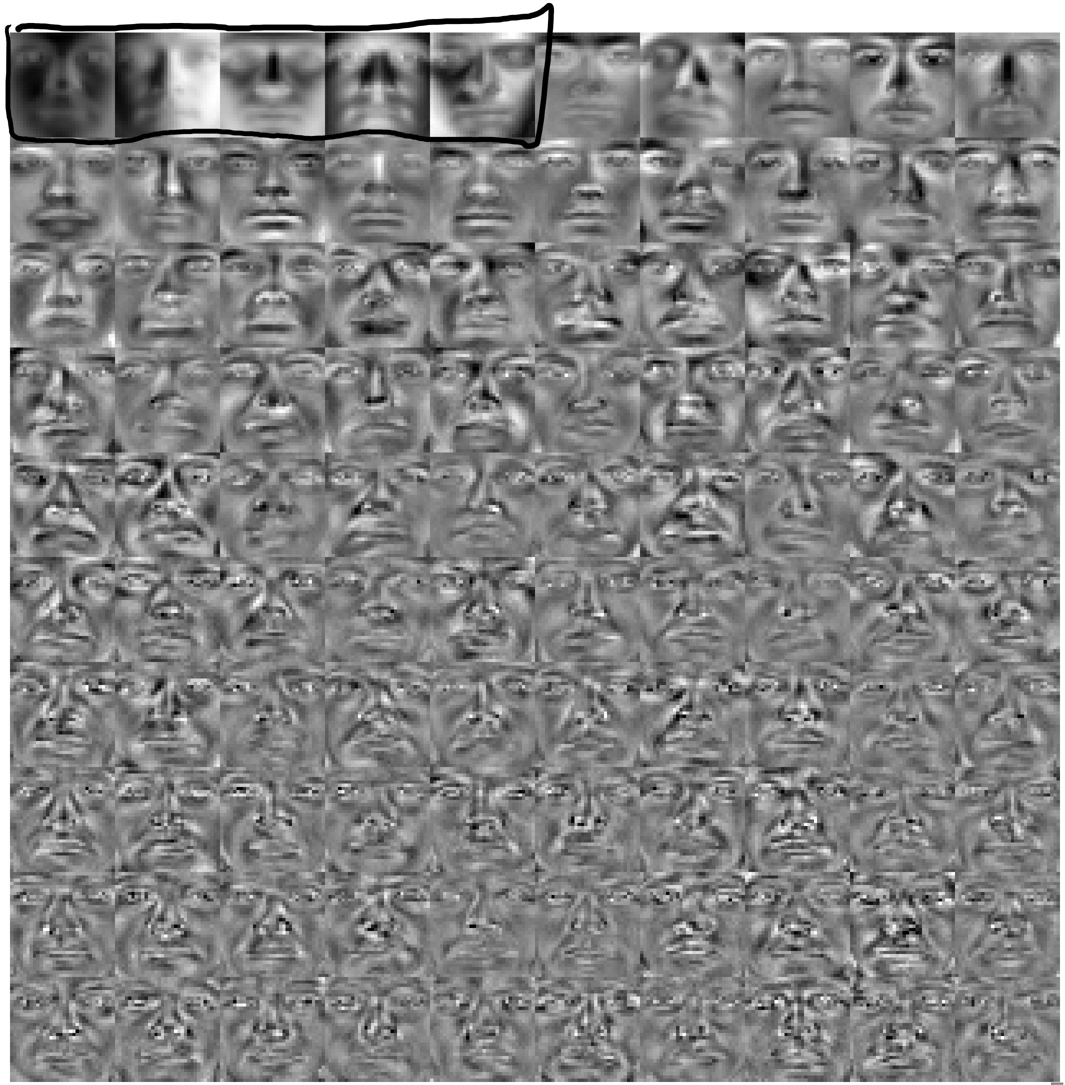
$\hat{x}_i$ $\qquad \mu \qquad$ PC1 $\qquad$ PC2 $\qquad$ PC3

(first row of W)

Eigenfaces

Reconstruction with $k=5$

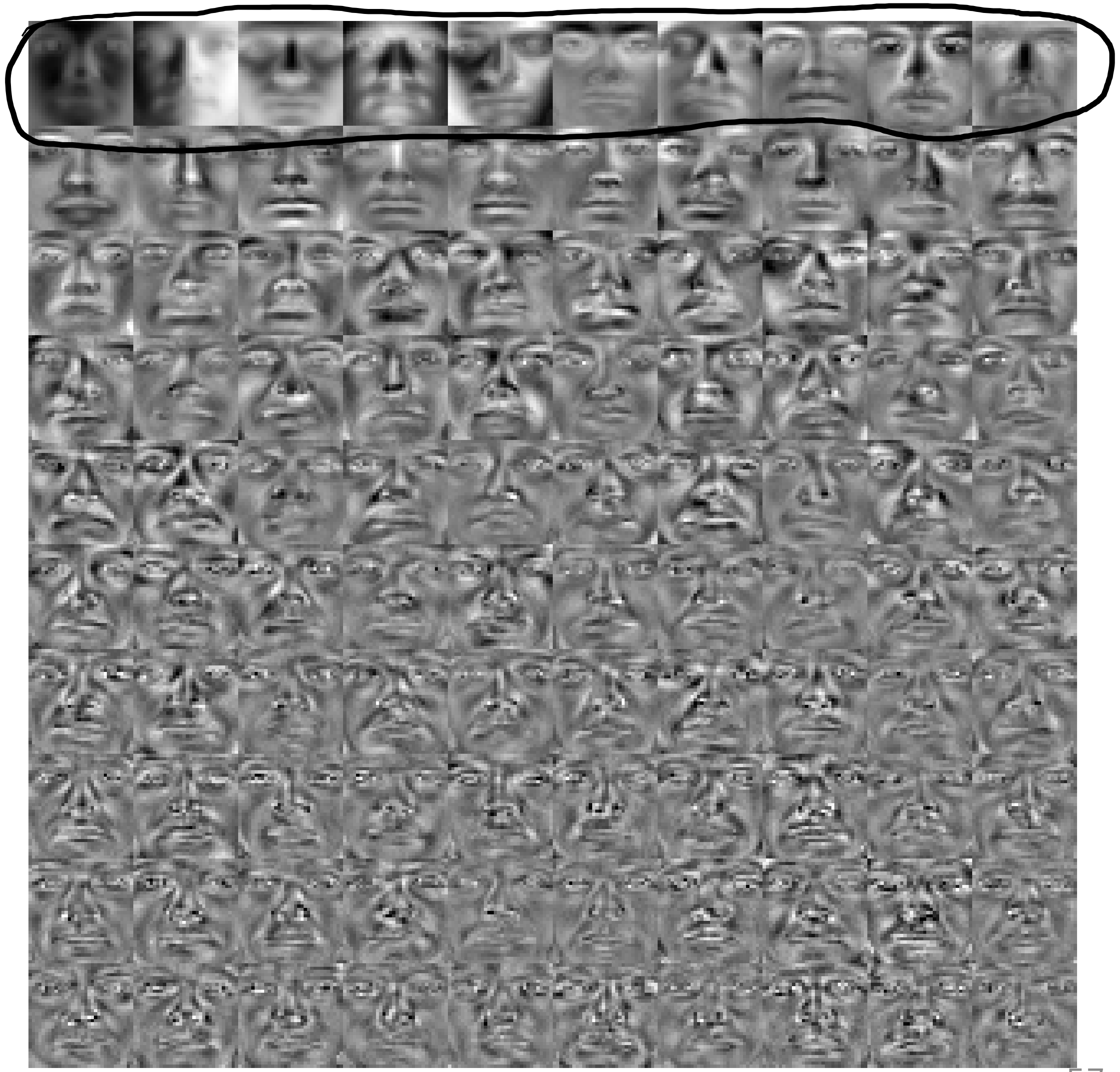
Variance explained: 80%



Eigenfaces

Reconstruction with k=10

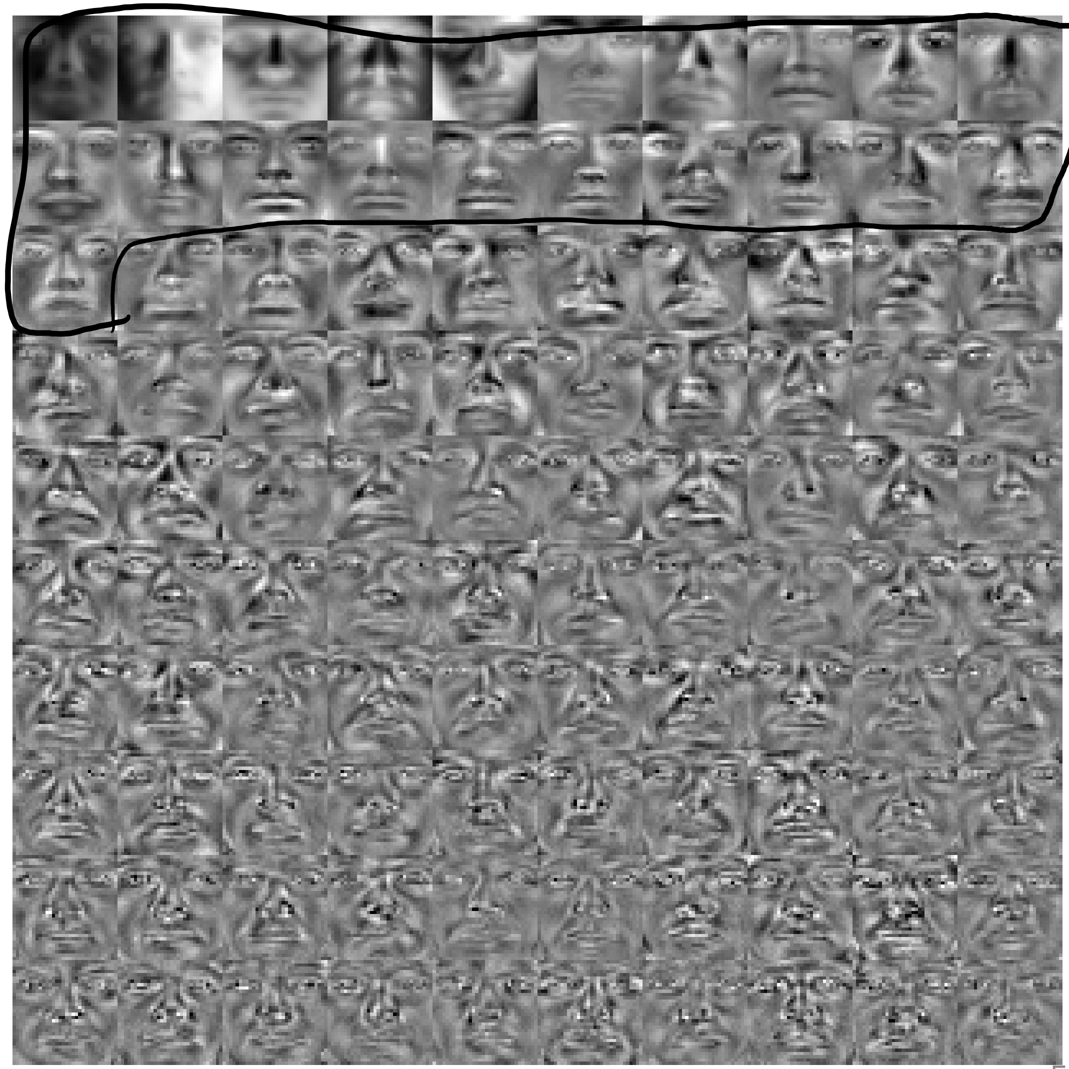
Variance explained: 85%



Eigenfaces

Reconstruction with k=21

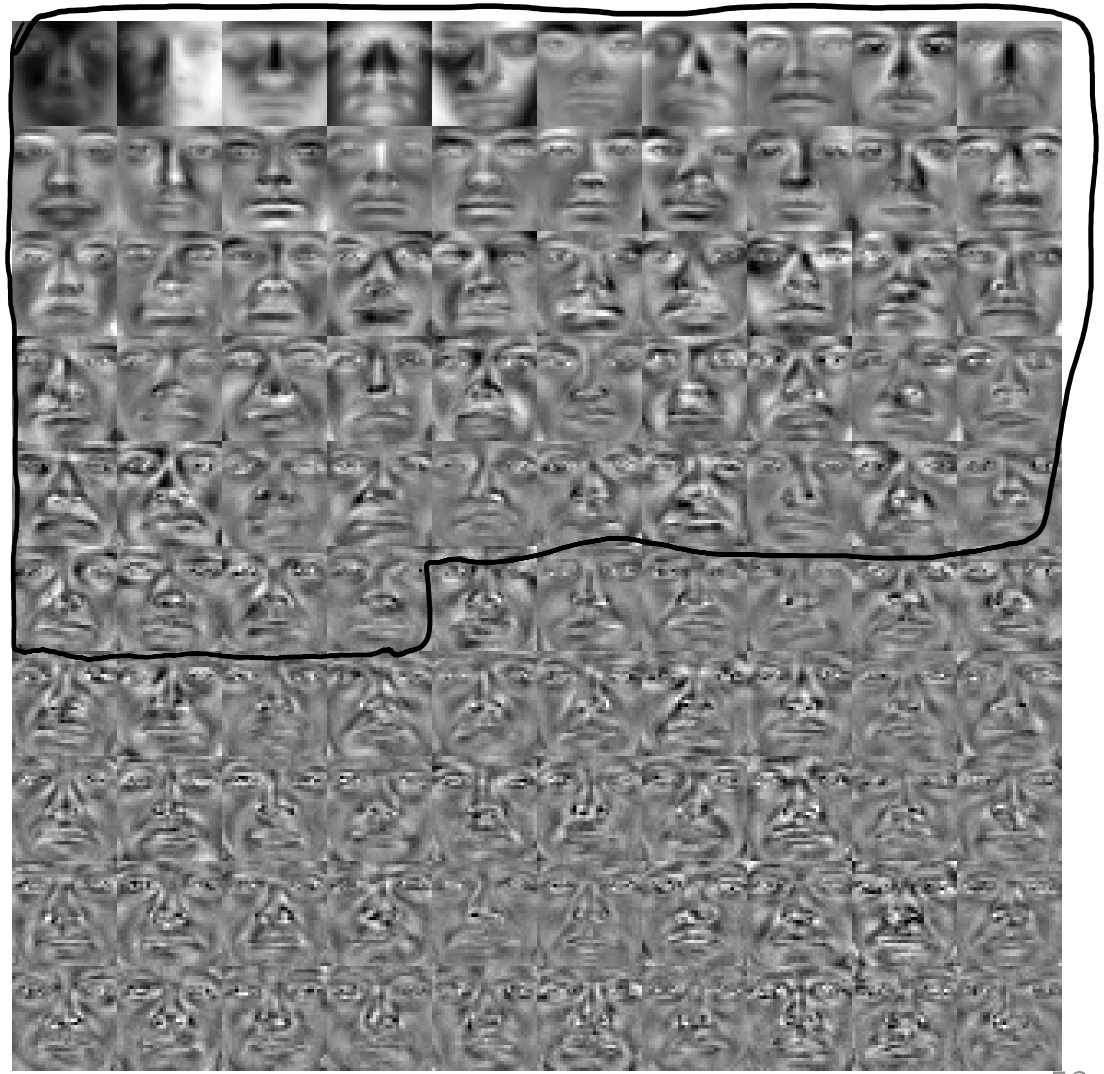
Variance explained: 90%



Eigenfaces

Reconstruction with $k = 54$

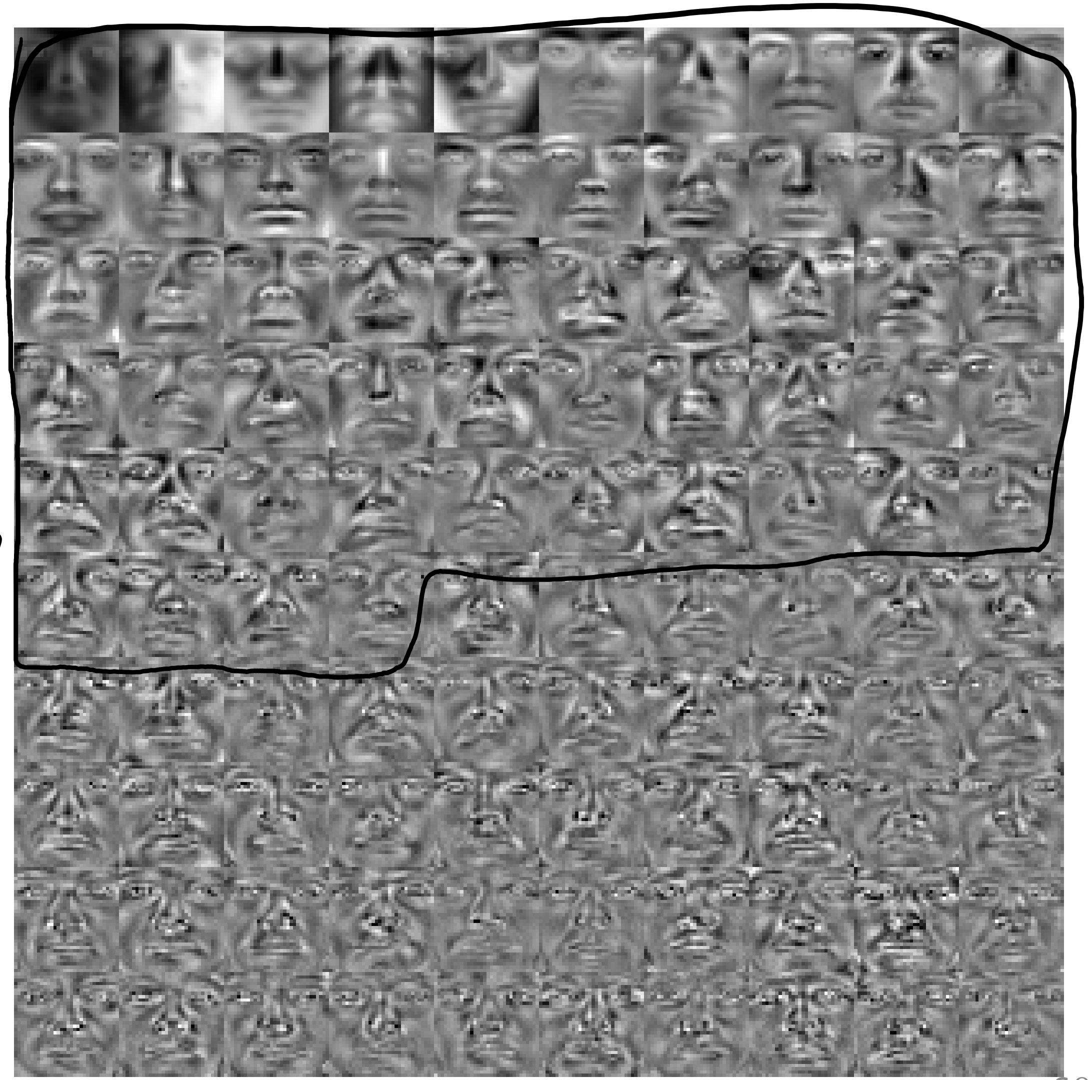
Variance explained: 95%



Eigenfaces

Original Images again:

Plus these "eigenfaces" and the mean.



We can replace 1024 x_i values by 54 z_i values

Summary

- PCA objective:
  - Minimizes squared error between elements of X and elements of ZW.
- Choosing 'k':
  - We can choose 'k' to explain "percentage of variance" in the data.
- PCA non-uniqueness:
  - Due to scaling, rotation, and label switching.
- Orthogonal basis and sequential fitting of PCs (via SVD):
  - Leads to non-redundant PCs with unique directions.
- Alternating minimization and stochastic gradient:
  - Iterative algorithms for minimizing PCA objective.

- Next time: cancer signatures and NBA shot charts.

Review Questions

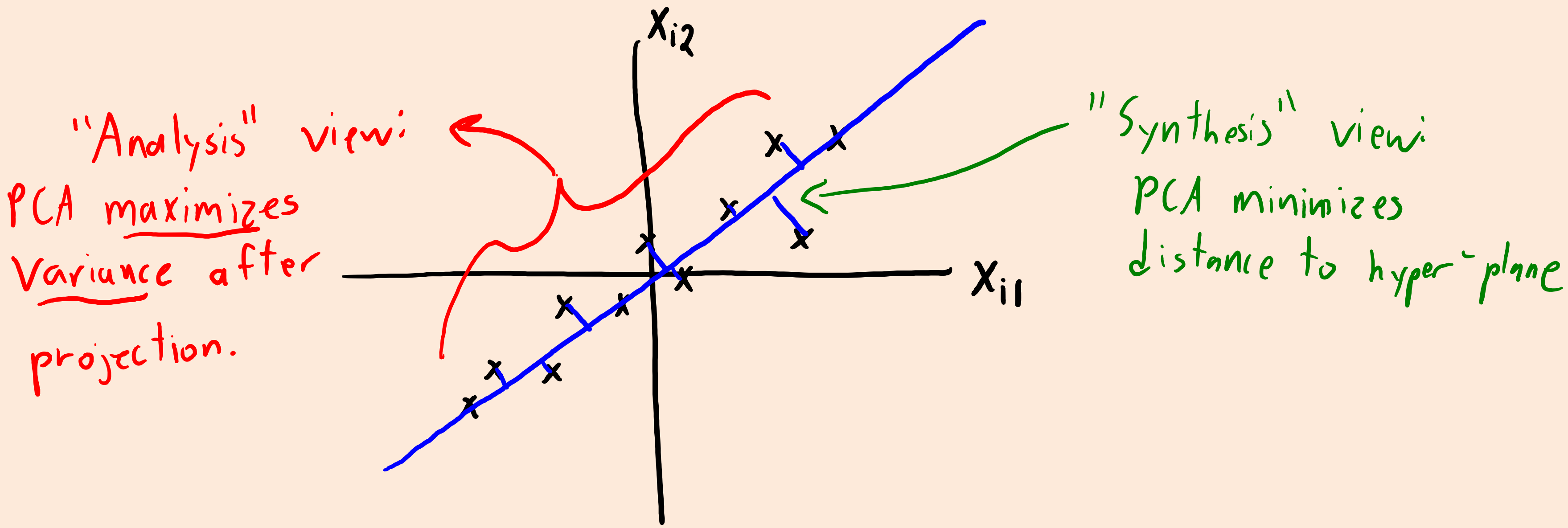
- Q1: How is PCA's objective function similar to k-means clustering's objective function?
- Q2: What makes PCA solutions non-unique?
- Q3: Why don't normalization and orthogonality guarantee the uniqueness of PCA solutions?
- Q4: How are "variance remaining" and "variance explained" related?

Making PCA Unique

- PCA implementations add constraints to make solution unique:
  - Normalization: we enforce that $||w_c|| = 1$.
  - Orthogonality: we enforce that $w_c^T w_{c'} = 0$ for all $c \neq c'$.
  - Sequential fitting: We first fit w_1 ("first principal component") giving a line.
    - Then fit w_2 given w_1 ("second principal component") giving a plane.
    - Then we fit w_3 given w_1 and w_2 ("third principal component") giving a space.
    - …
- Even with all this, the solution is only unique up to sign changes:
  - I can still replace any w_c by $-w_c$:
    - $-w_c$ is normalized, is orthogonal to the other $w_{c'}$, and spans the same space.
  - Possible fix: require that first non-zero element of each w_c is positive.
  - And this is assuming you don't have repeated singular values.
    - In that case you can rotate the repeated ones within the same plane.

"Synthesis" View vs. "Analysis" View

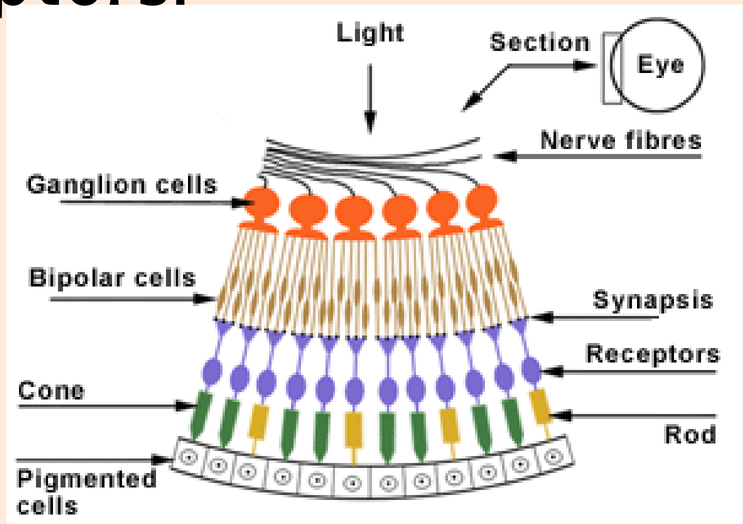
- We said that PCA finds hyper-plane minimizing distance to data x_i.
  - This is the "synthesis" view of PCA (connects to k-means and least squares).



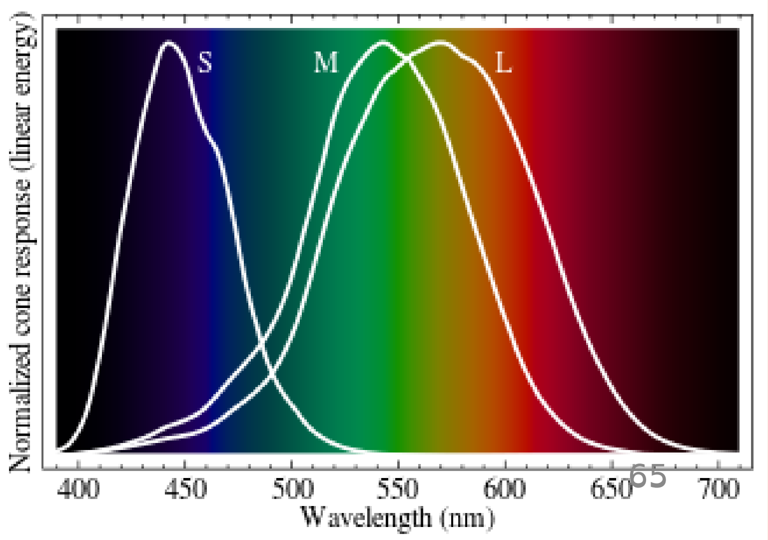
- "Analysis" view when we have orthogonality constraints:
  - PCA finds hyper-plane maximizing variance in z_i space.
  - You pick W to "explain as much variance in the data" as possible.

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).



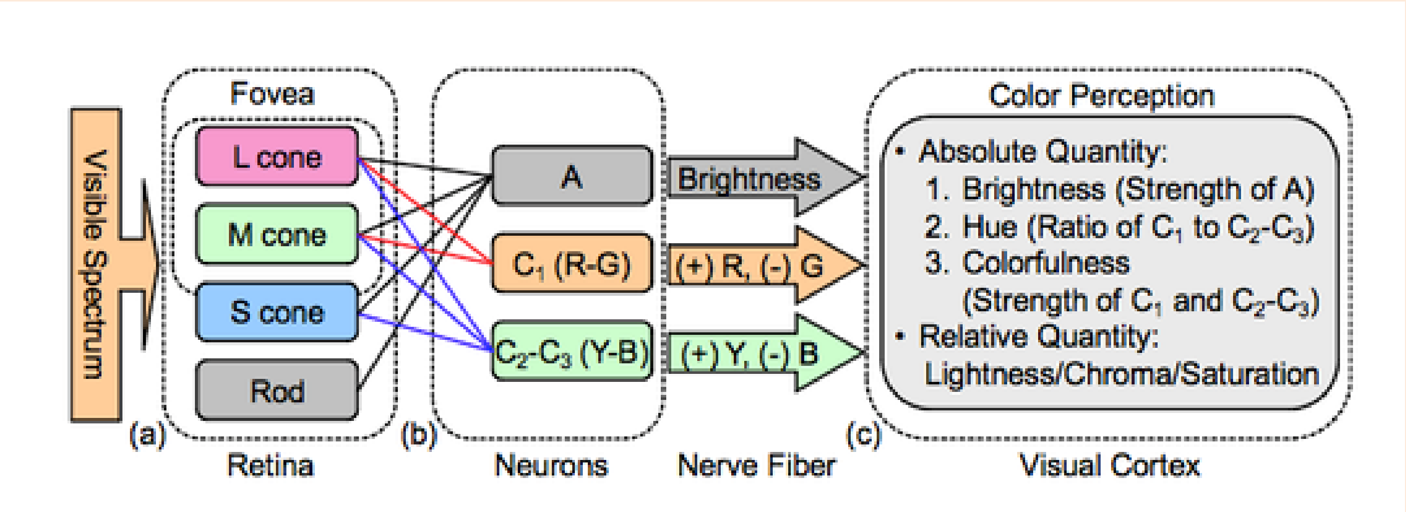
- Two problems with this system:
  - Not orthogonal.
    - High correlation in particular between red/green.
  - We have 4 receptors for 3 colours.

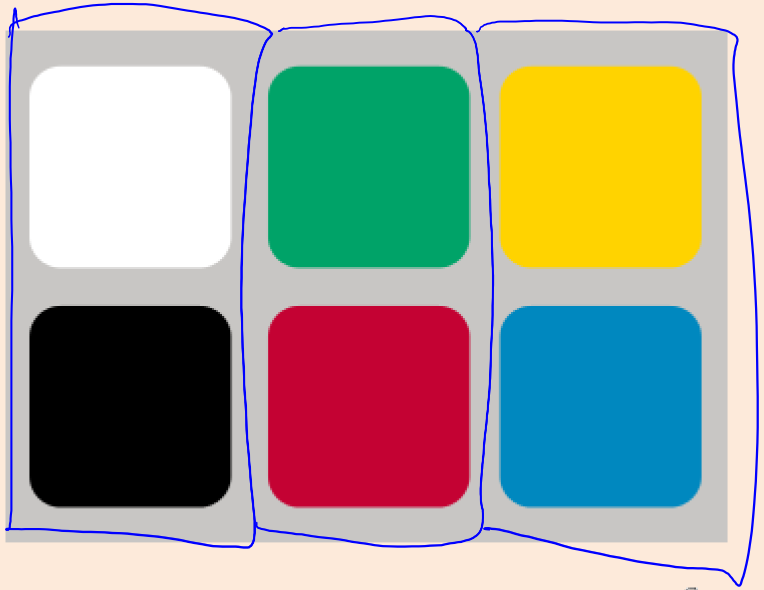


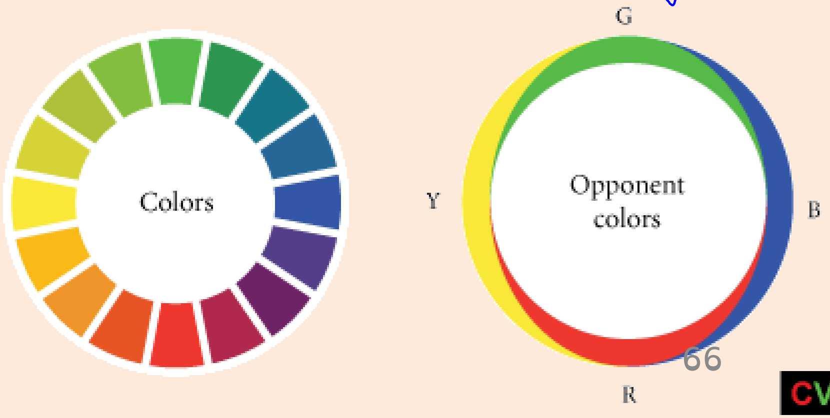
http://oneminuteastronomer.com/astro-course-day-5/
https://en.wikipedia.org/wiki/Color_visio

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 ($d = 4$, $k = 3$).

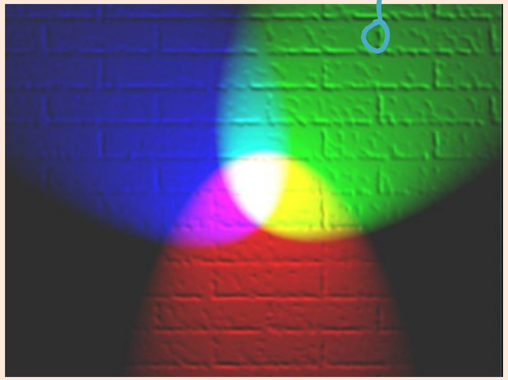
http://oneminuteastronomer.com/astro-course-day-5/
https://en.wikipedia.org/wiki/Color_visio
http://5sensesnews.blogspot.ca/

CVRL

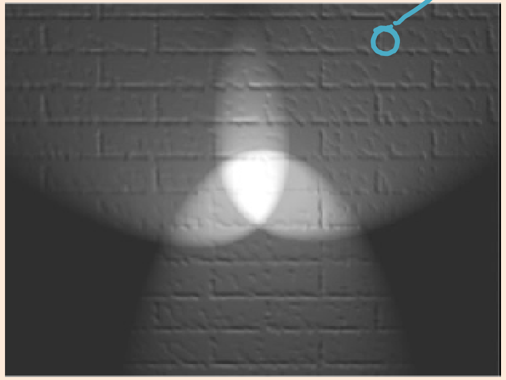
Colour Opponency Representation

For this pixel, eye gets 4 signals

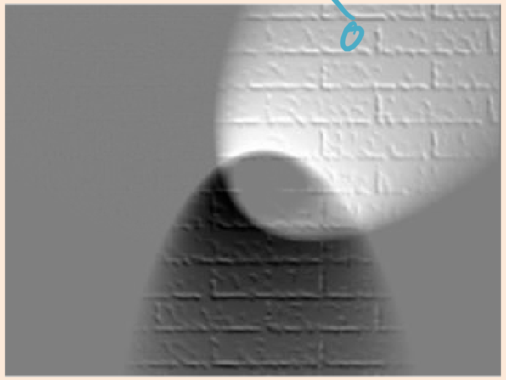
Can represent 4 original values with these 3 z_i values and matrix 'W'



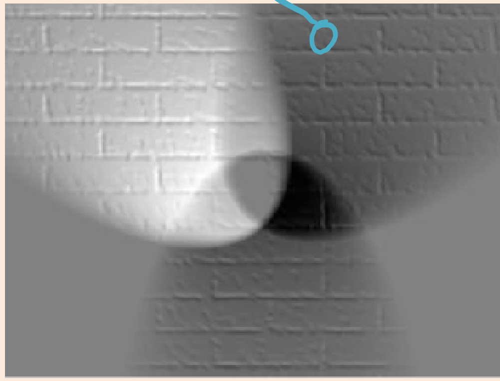
$=$ w_1



$+w_2$



$+w_3$



First row of W (First PC)

Analogous to means in k-means.

brightness

Second row (4x1)

red/green

Third row (4x1)

blue/yellow

https://en.wikipedia.org/wiki/RGB_color_model