CPSC 340:
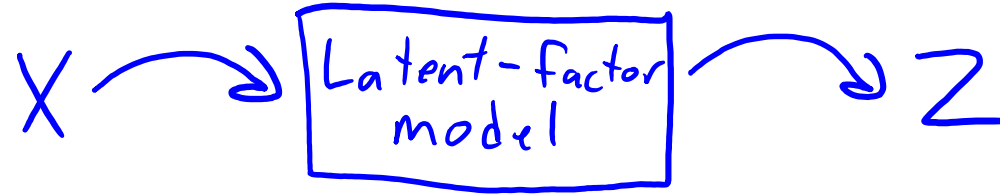# Machine Learning and Data Mining

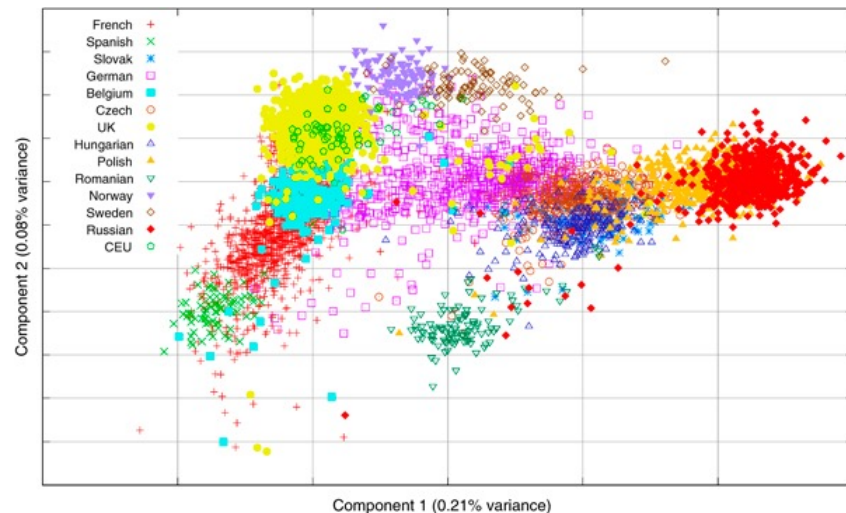More PCA

Last Time: Latent-Factor Models

- Latent-factor models take input data 'X' and output 'Z':

$$X \longrightarrow \boxed{\text{Latent-factor model}} \longrightarrow Z$$

  – Usually, 'Z' has fewer features than 'X'.

- Uses: dimensionality reduction, visualization, factor discovery.

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.

Last Time: Principal Component Analysis

- Principal component analysis (PCA) is a linear latent-factor model:
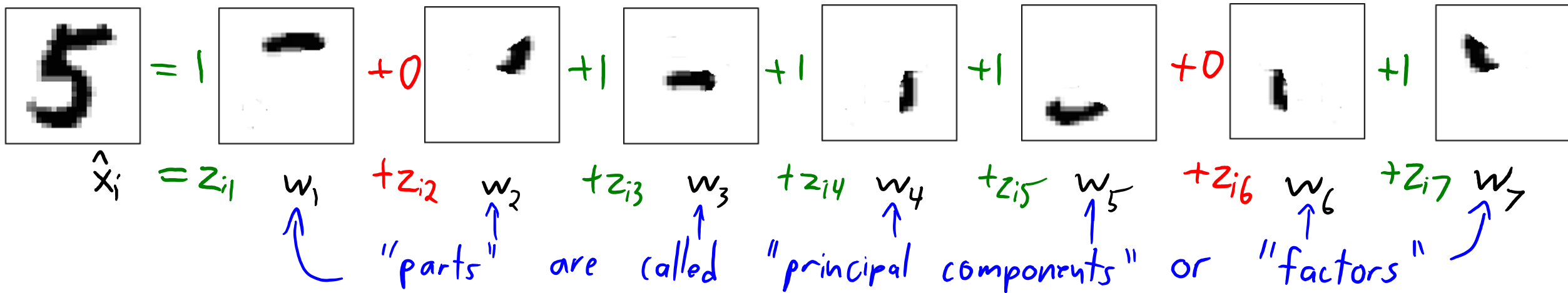 - These models "factorize" matrix X into matrices Z and W:

$$X \underset{n \times d}{\approx} \underset{n \times k}{Z} \underset{k \times d}{W} \qquad x_i \approx W^\top z_i \qquad x_{ij} \approx \langle w^j, z_i \rangle$$

  - We can think of rows w_c of W as 'k' fixed "part" (used in all examples).
  - z_i is the "part weights" for example x_i: "how much of each part w_c to use".

$$\hat{x}_i = z_{i1} \, w_1 + z_{i2} \, w_2 + z_{i3} \, w_3 + z_{i4} \, w_4 + z_{i5} \, w_5 + z_{i6} \, w_6 + z_{i7} \, w_7$$

"parts" are called "principal components" or "factors"

Top-10 ML Algorithms

1. Decision trees
2. Naïve Bayes classification
3. Ordinary least squares regression
4. Logistic regression
5. Support vector machines
6. Ensemble methods
7. Clustering algorithms
8. Principal component analysis
9. Singular value decomposition
10. Independent component analysis (bonus)

The 10 Algorithms Machine Learning Engineers Need to Know

Aug 2016
KDnuggets Gold Blog

Next Topic: PCA Loss Function and Prediction

PCA Objective Function

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

$$f(W, z) = \sum_{i=1}^{n} || W^T z_i - x_i ||^2$$

$\underbrace{W^T z_i}_{\text{approximation}}$ $\underbrace{x_i}_{\text{example } i}$

- This is equivalent to the k-means objective:
  - In k-means z_i only has a single '1' value and other entries are zero.

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

PCA Objective Function

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

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

approximation example 'i'   approximation   feature 'j' of example 'i'

- We can also view this as solving 'd' regression problems:
  - Each w^j is trying to predict column 'x^j' from the low-dimensional representation z_i.
    - The output "y_i" we try to predict here is actually the features "x_i".
  - Unlike in regression we are also learning the features z_i.

Principal Component Analysis (PCA)

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

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

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

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

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:

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

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

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

Digression: Data Centering (Important)

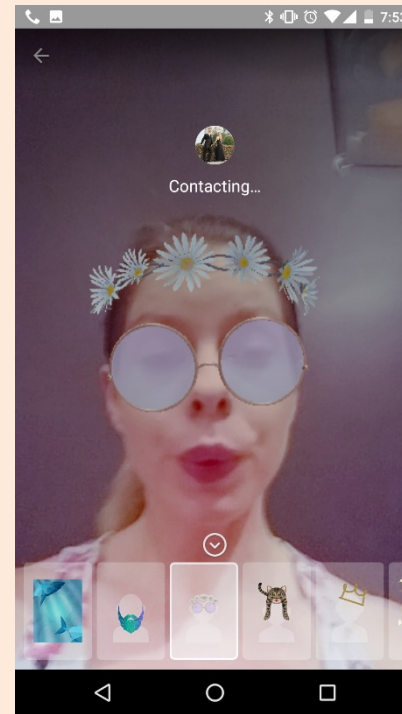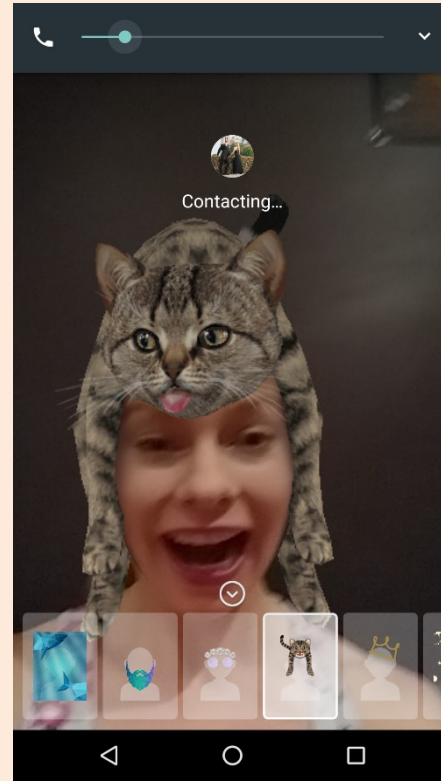$$x_i \approx W^\top z_i$$
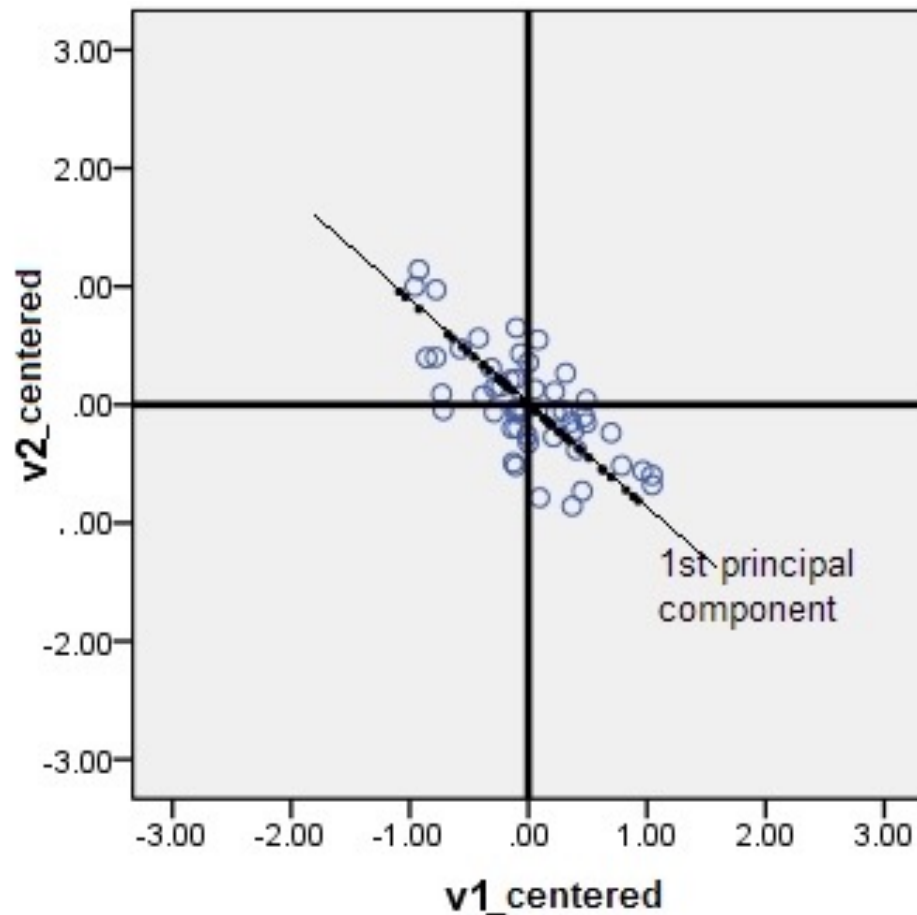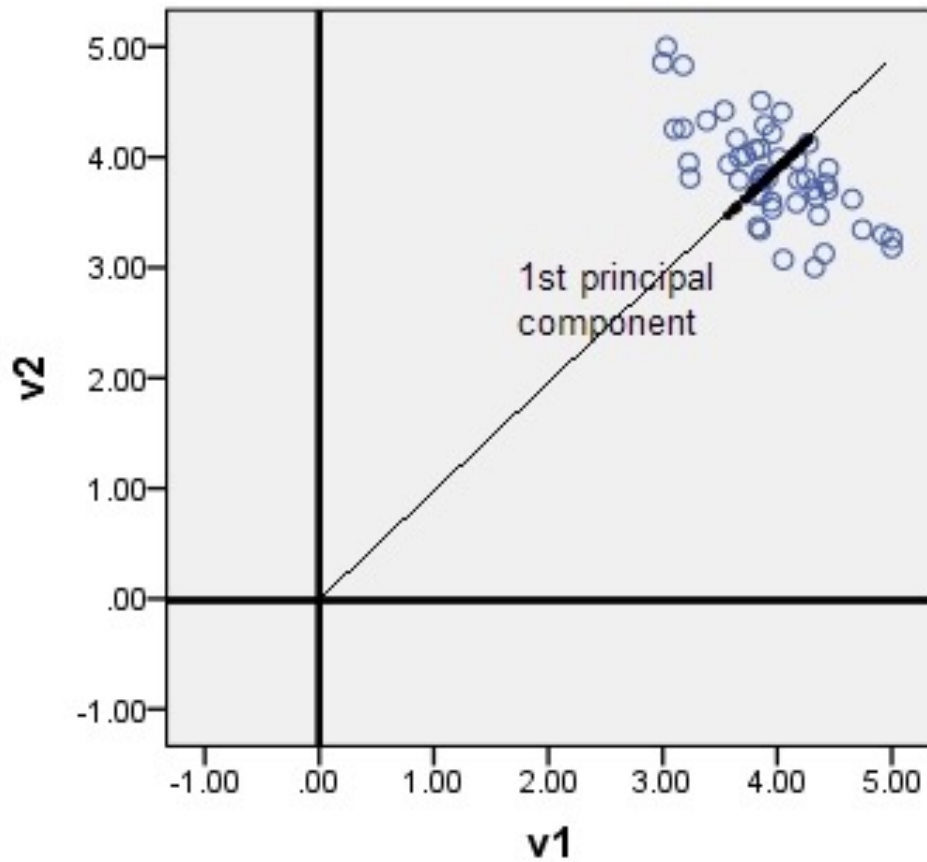
Next Topic: Eigenfaces

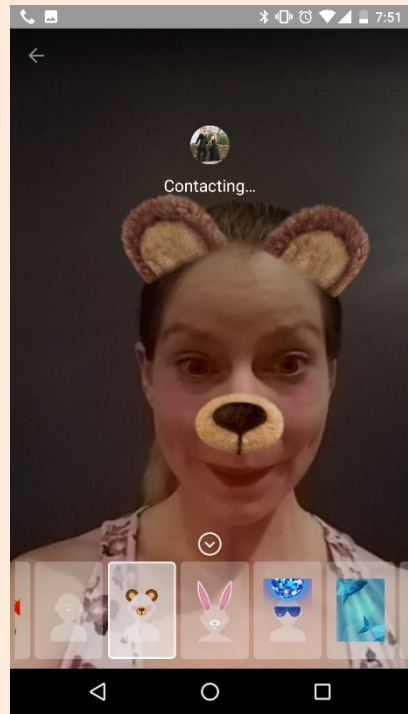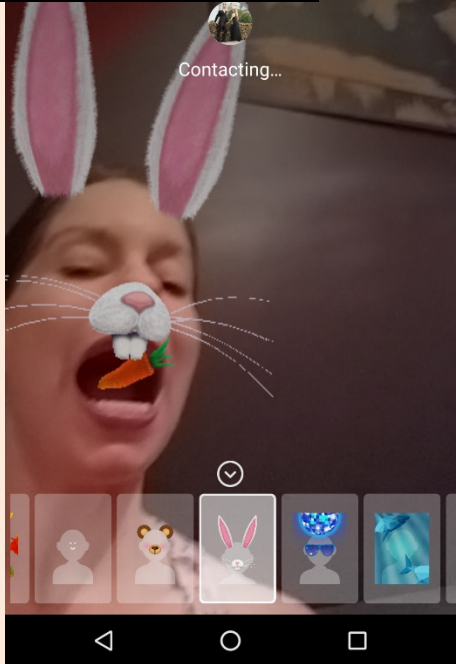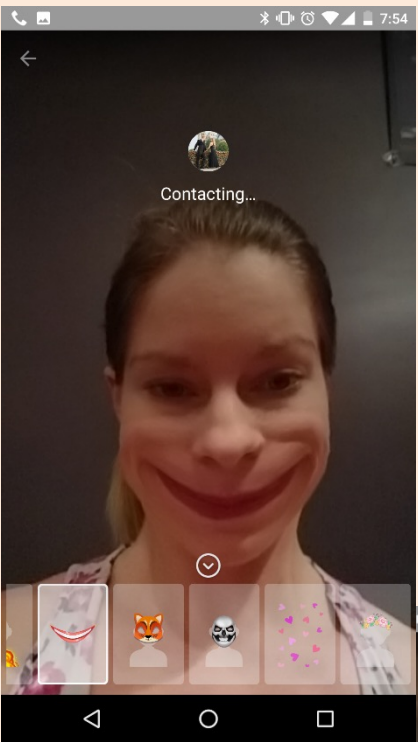# Application: Face Detection

- Consider the problem of face detection:
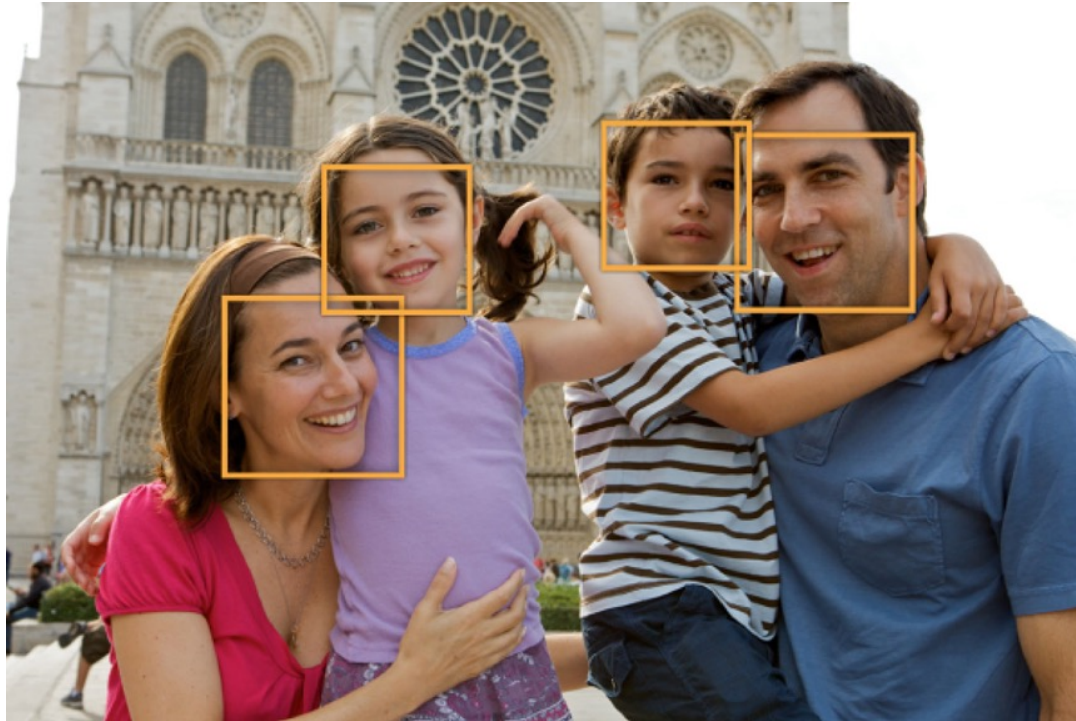
- Classic methods use "eigenfaces" as basis:
  - PCA applied to images of faces.

Application: Face Detection
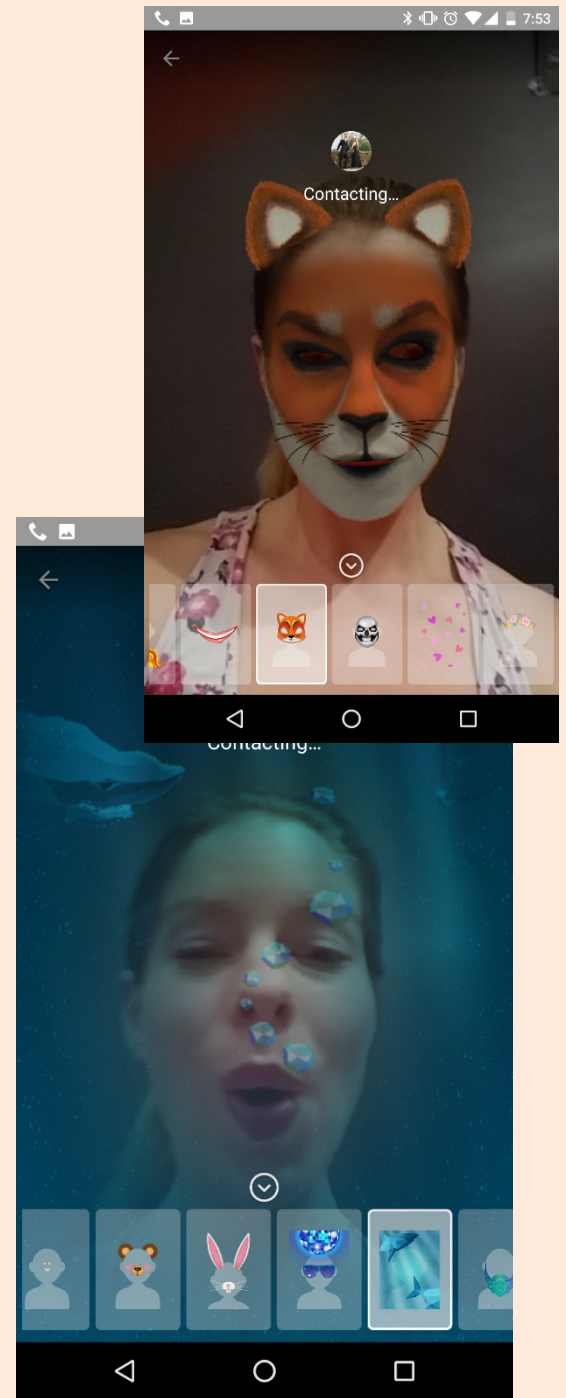
Eigenfaces

- 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

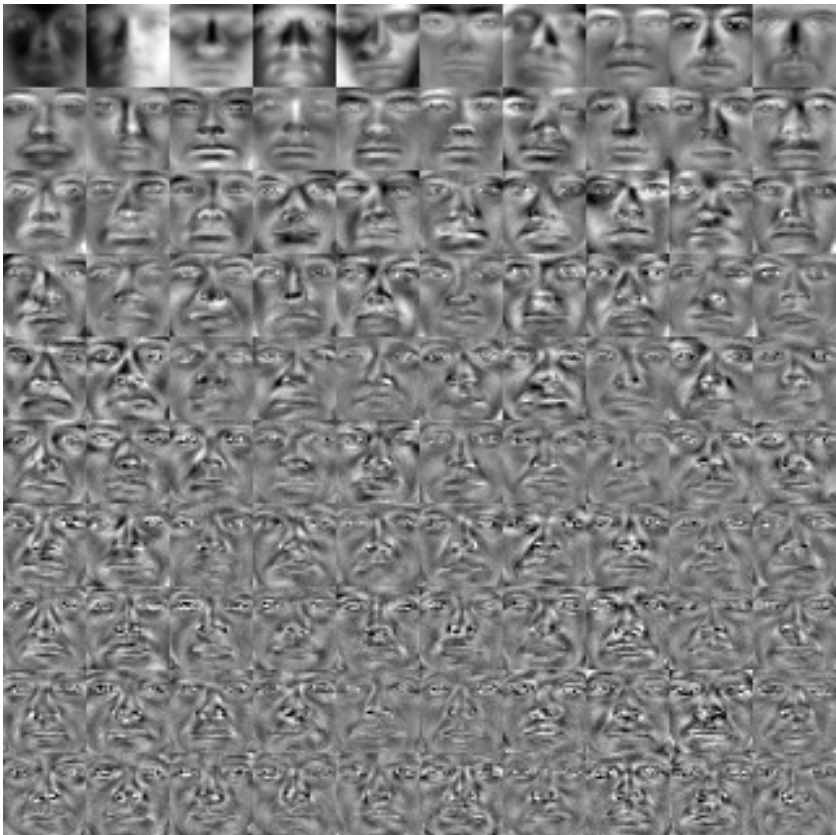Compute _mean_ μ_j of each column.

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

Each row of X will be pixels in _one_ image:

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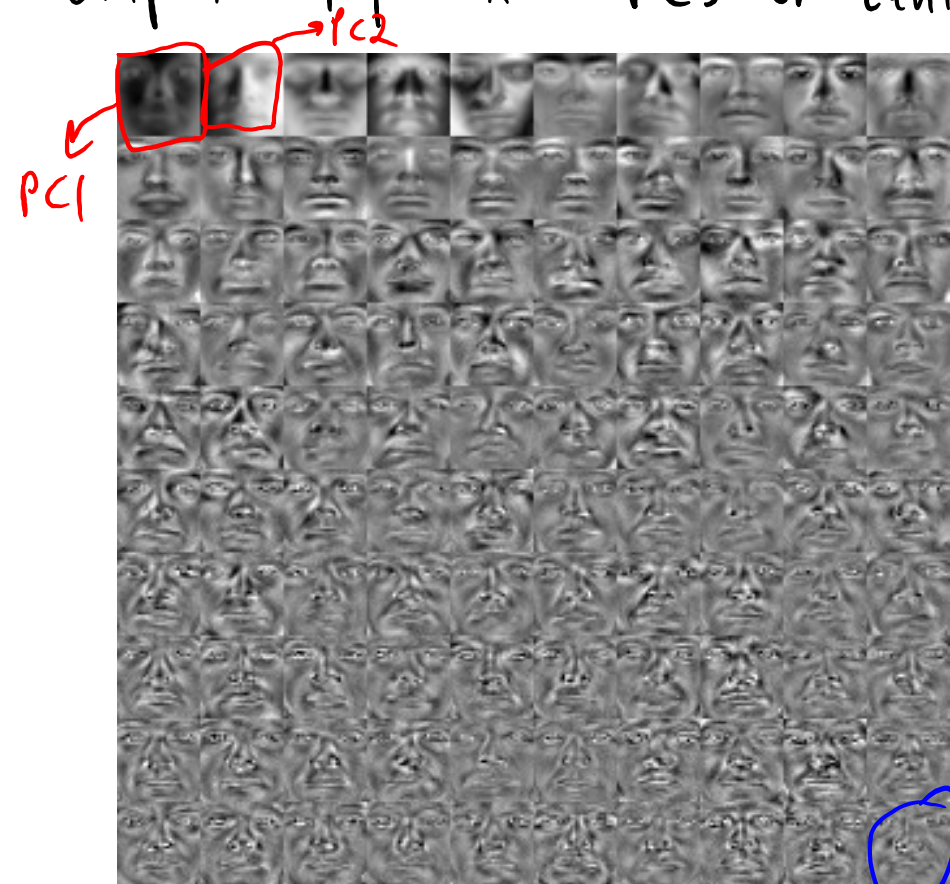
Eigenfaces

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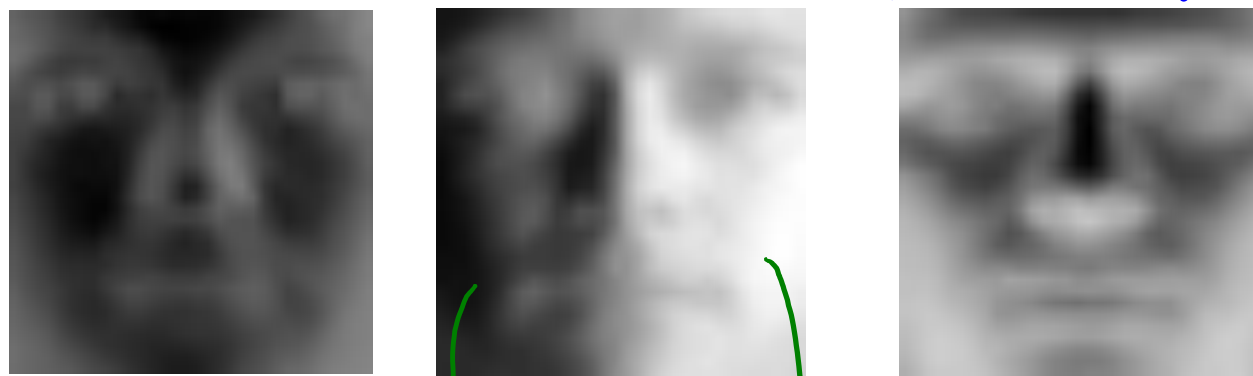

$$X = \begin{bmatrix} & \text{---} & x_1 - \mu & \text{---} & \\ & \text{---} & x_2 - \mu & \text{---} & \\ & & \vdots & & \\ & \text{---} & x_n - \mu & \text{---} & \end{bmatrix}$$

Eigenfaces

Compute top 'k' PCs on centered data:

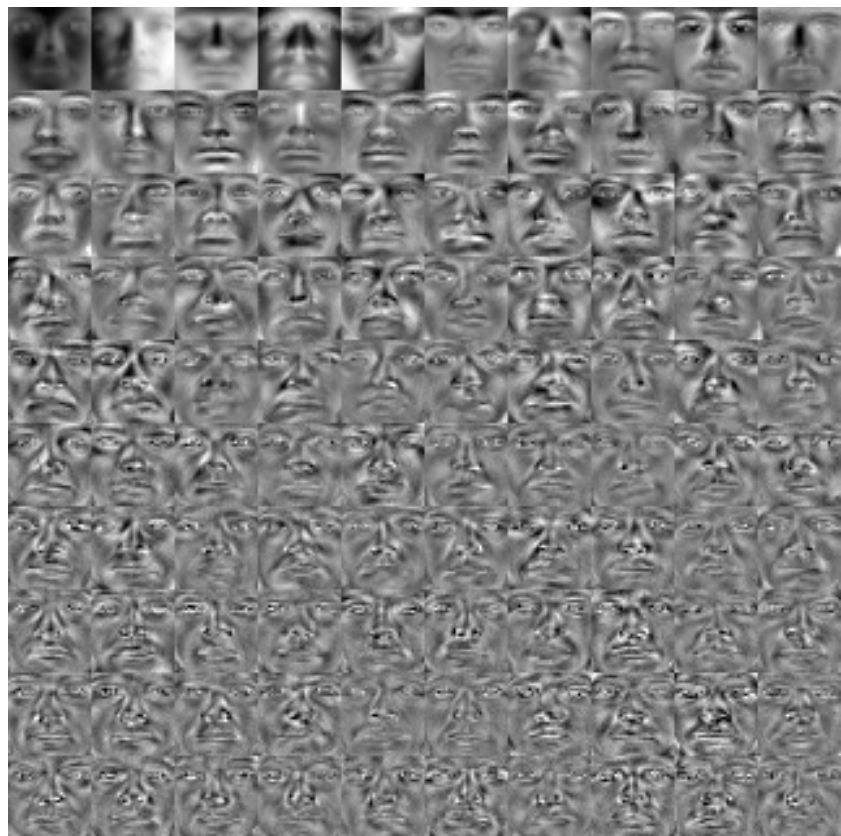

PC1
PC2

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:

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

PC1 (first row of W)

Eigenfaces

106 of the original faces:

"Eigenface" representation:

$$\hat{x}_i = \mu + z_{i1} \cdot PC1 + z_{i2} \cdot PC2 + z_{i3} \cdot PC3 + \cdots$$

(first row of W)

Eigenfaces

Reconstruction with $k = 0$

Variance explained: 0%

"Eigenface" representation:

$\hat{x}_i$ μ $+z_{i1}$ PC1 $+z_{i2}$ PC2 $+z_{i3}$ PC3 $+\cdots$

(first row of W)

Eigenfaces

Reconstruction with $K=1$

Variance explained: 36%

PCA Visualization:

$$z_i = w_c^T x_i$$

"Eigenface" representation:

$$\hat{x}_i = \mu + z_{i1} \, PC1 + z_{i2} \, PC2 + z_{i3} \, PC3 + \cdots$$
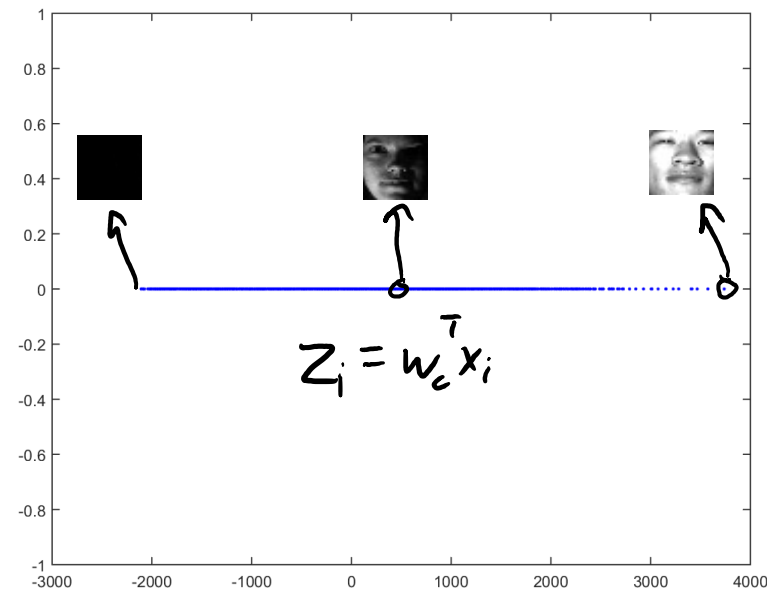
(first row of W)

Eigenfaces

Reconstruction with $k=2$

Variance explained: 71%

PCA Visualization:

"Eigenface" representation:

$$\hat{x}_i = \mu + z_{i1}\,PC1 + z_{i2}\,PC2 + z_{i3}\,PC3 + \cdots$$
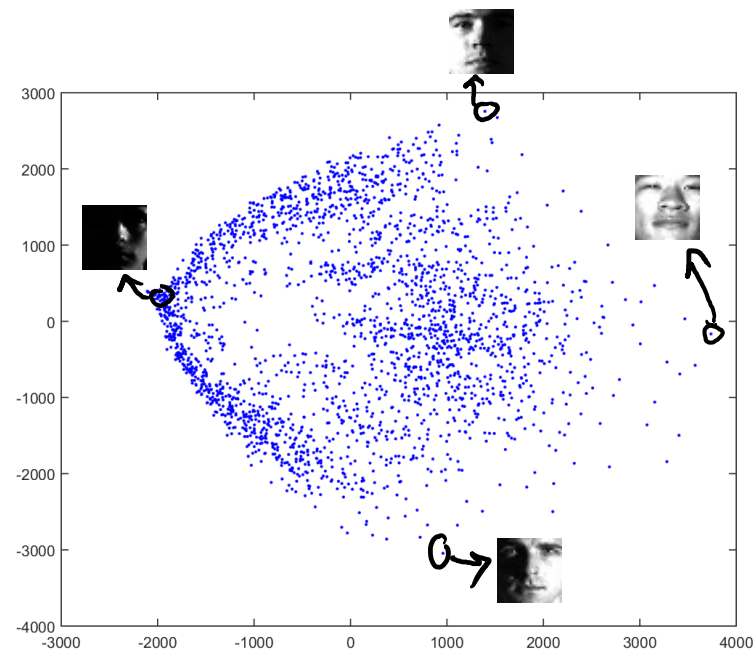
(first row of W)

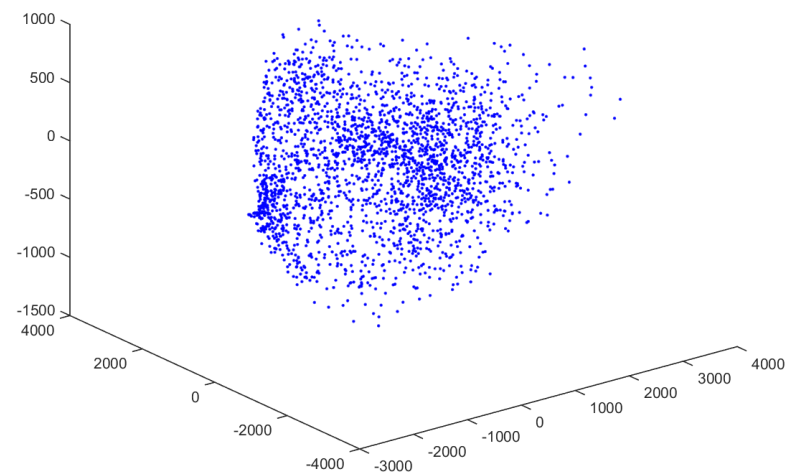# Eigenfaces

Reconstruction with k=3

Variance explained: 76%

PCA Visualization:

"Eigenface" representation:

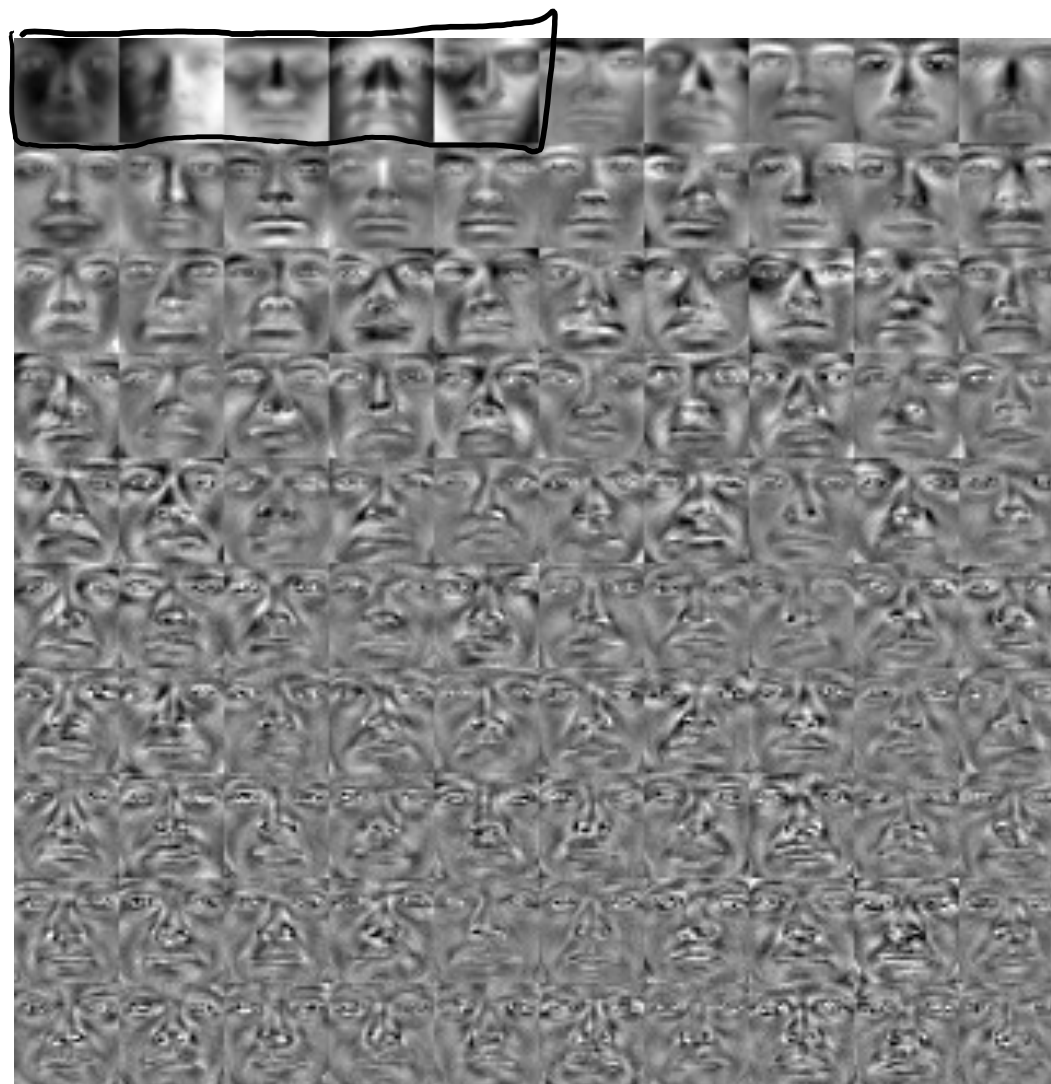$$\hat{x}_i = \mu + z_{i1} \, PC1 + z_{i2} \, PC2 + z_{i3} \, PC3 + \cdots$$

(first row of W)

Eigenfaces

Reconstruction with k=5
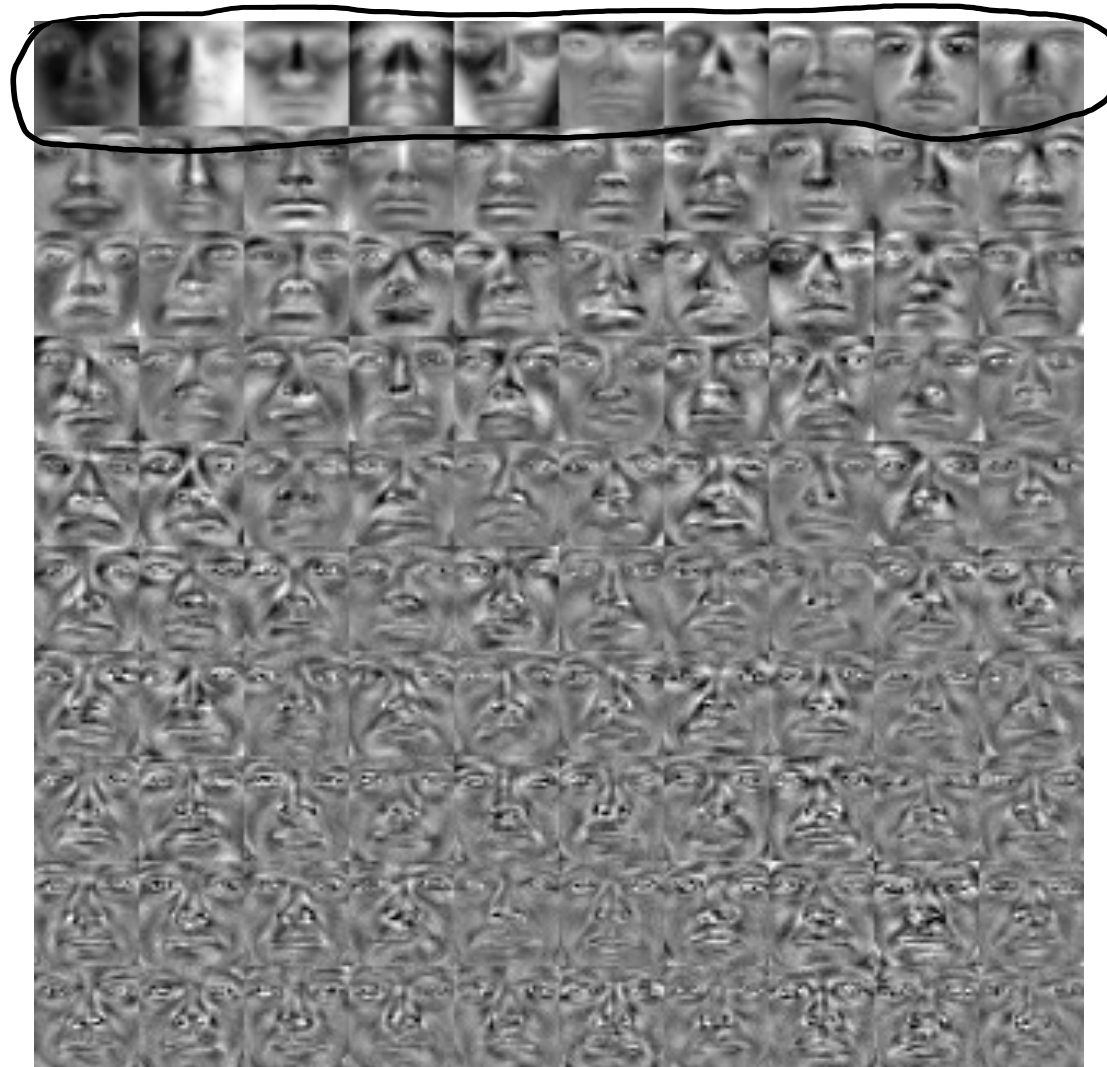

Variance explained: 80%

Eigenfaces

Reconstruction with K=10
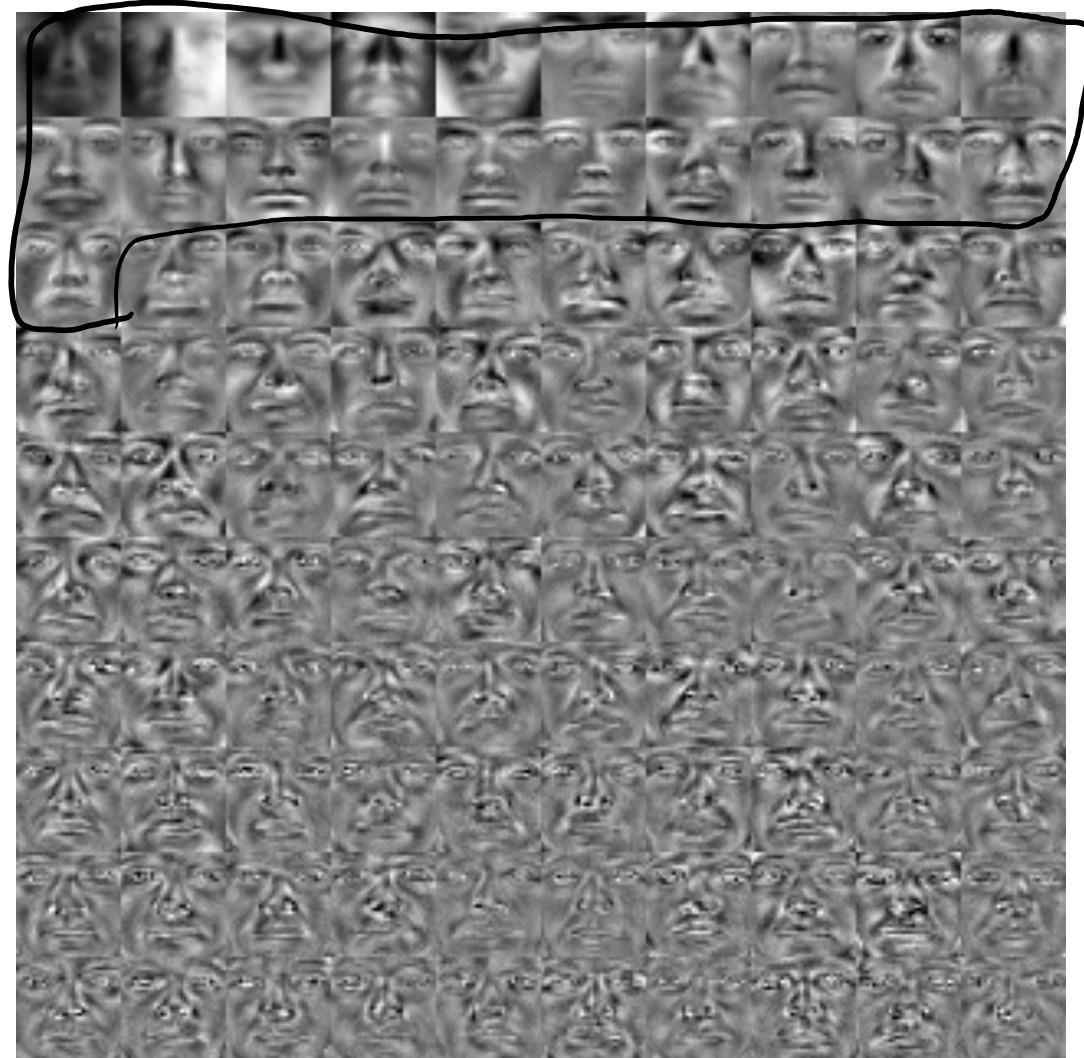

Variance explained: 85%

Eigenfaces

Reconstruction with K=21

Variance explained: 90%

Eigenfaces

Reconstruction with K=54

Variance explained: 95%
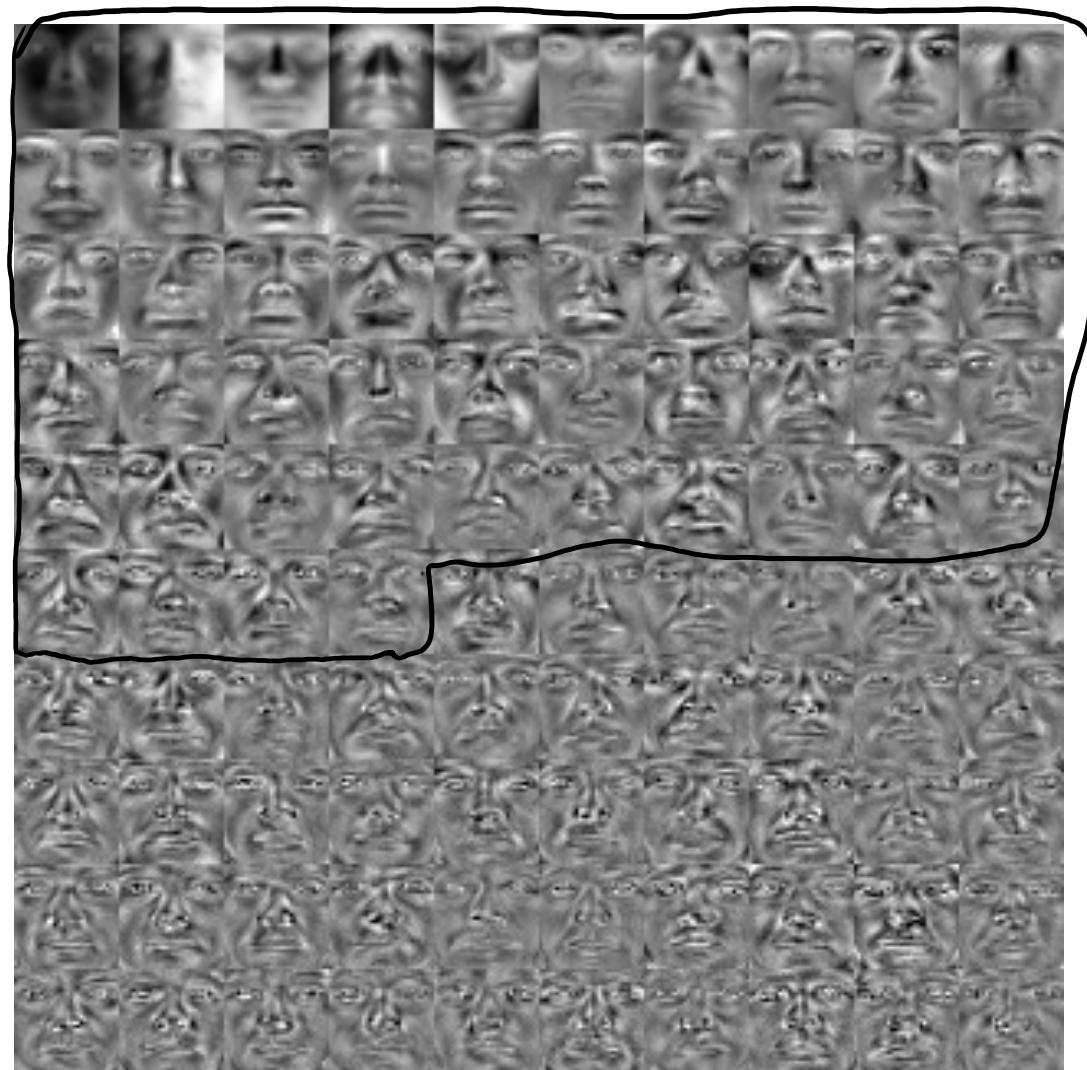
Eigenfaces

Original Images again:

Plus these "eigenfaces" and the mean.
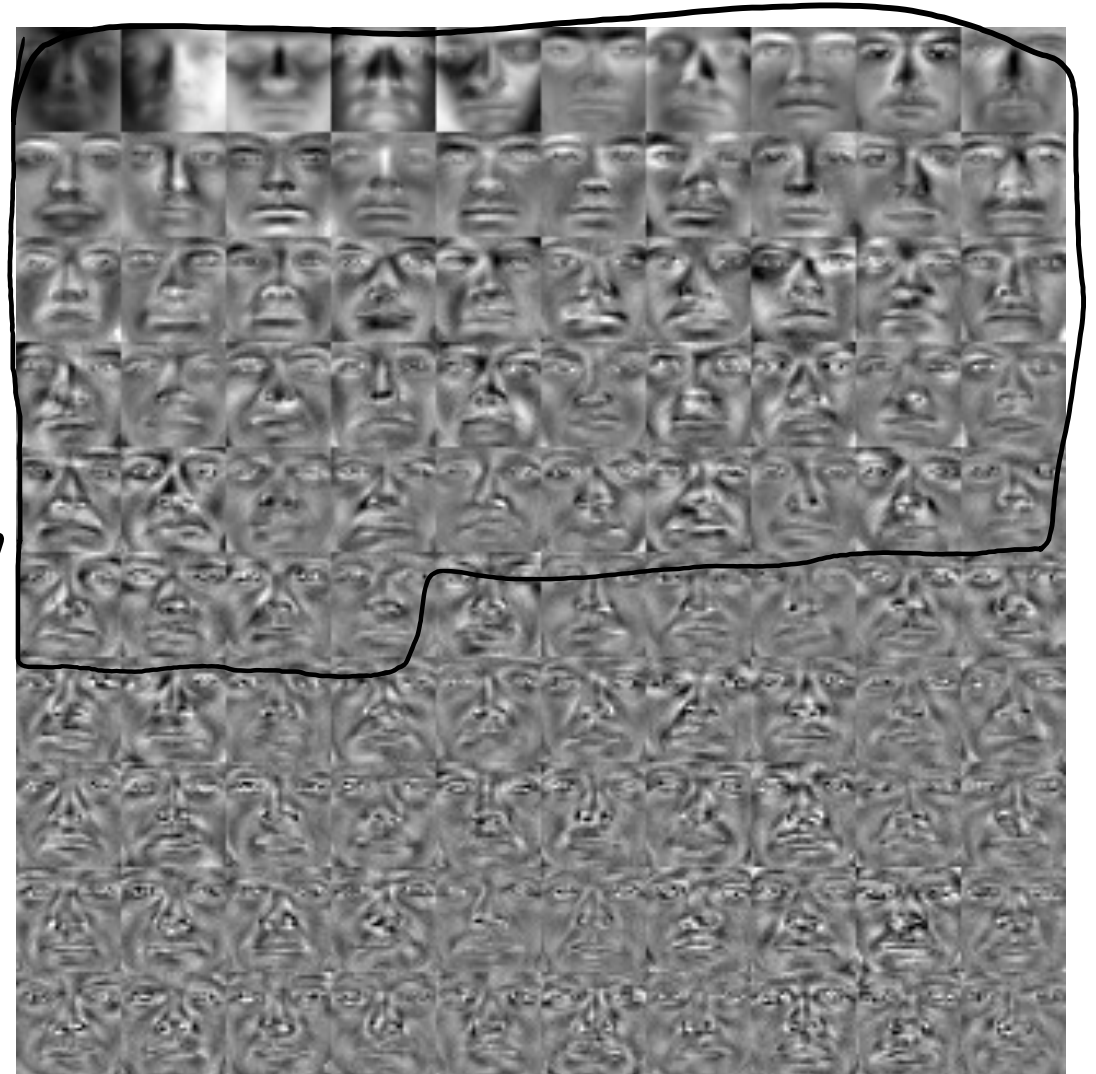
We can replace 1024 x_i values by 54 z_i values

Next Topic: Non-Uniqueness of PCA

Non-Uniqueness of PCA

- Unlike k-means, we can efficiently find global optima of f(W,Z).
  - Algorithms coming later.

- Unfortunately, there never exists a unique global optimum.
  - There are actually several different sources of non-uniqueness.

- To understand these, we'll need idea of "span" from linear algebra.
  - This also helps explain the geometry of PCA.
  - We'll also see that some global optima may be better than others.

Span of 1 Vector

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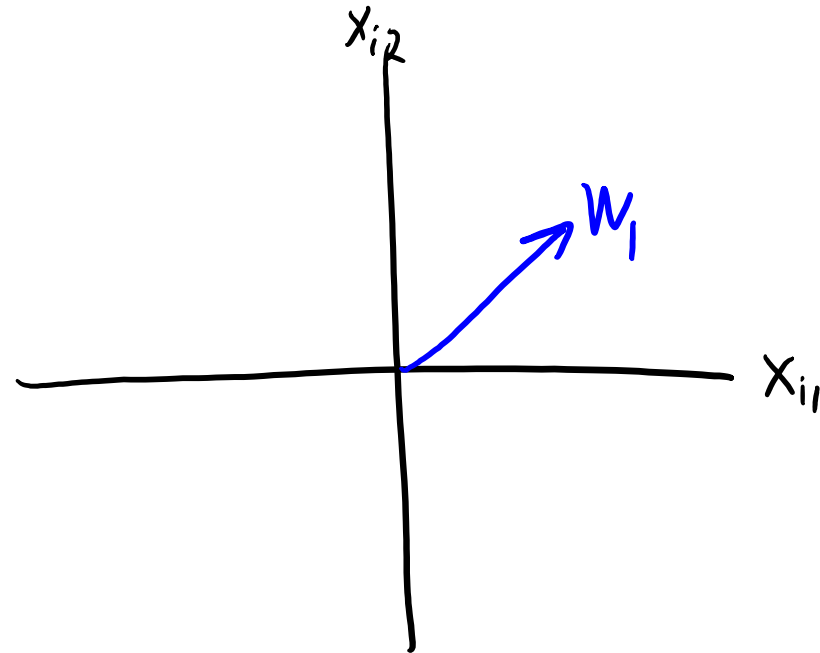
Span of 1 Vector

- 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.
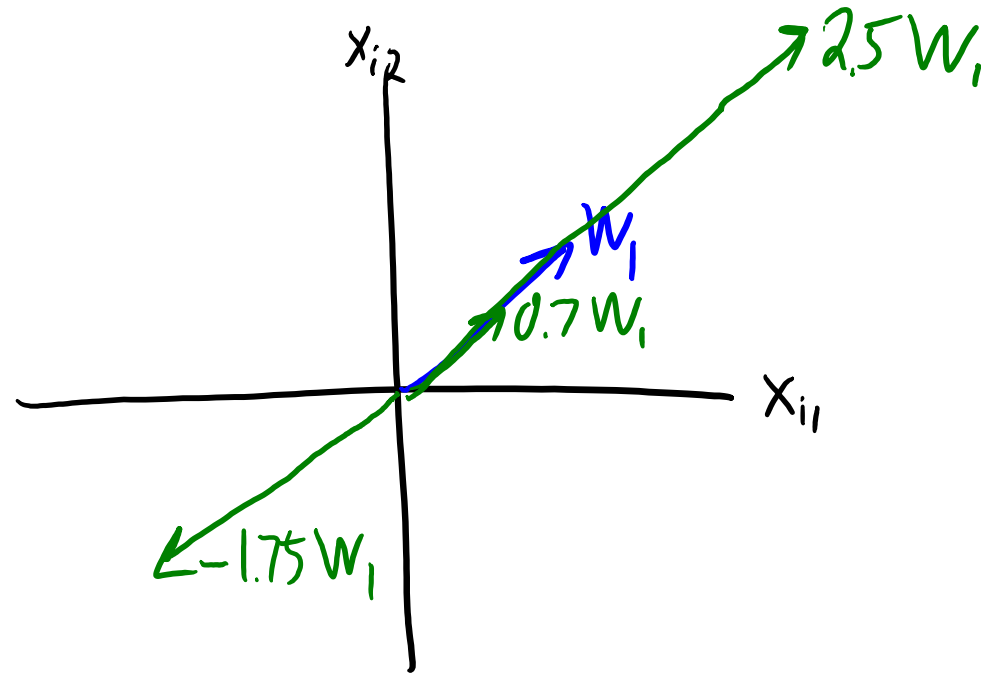
Span of 1 Vector

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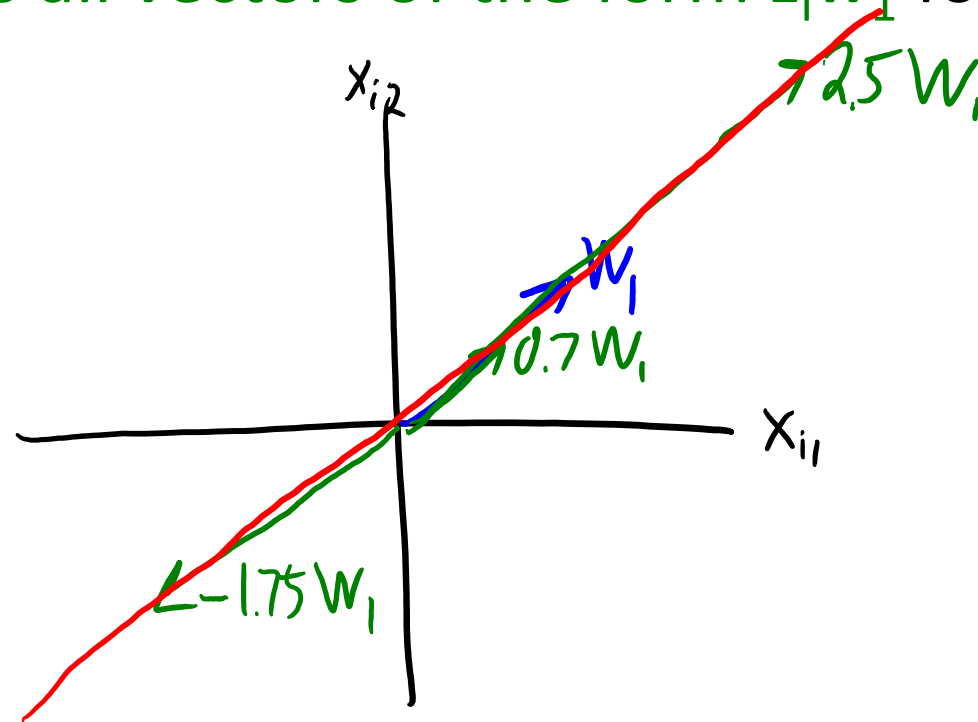
- If $w_1 \neq 0$, this forms a line.

Span of 1 Vector

- But note that the "span" of many different vectors gives same line.
  - Mathematically: αw_1 defines the same line as w_1 for any scalar $\alpha \neq 0$.
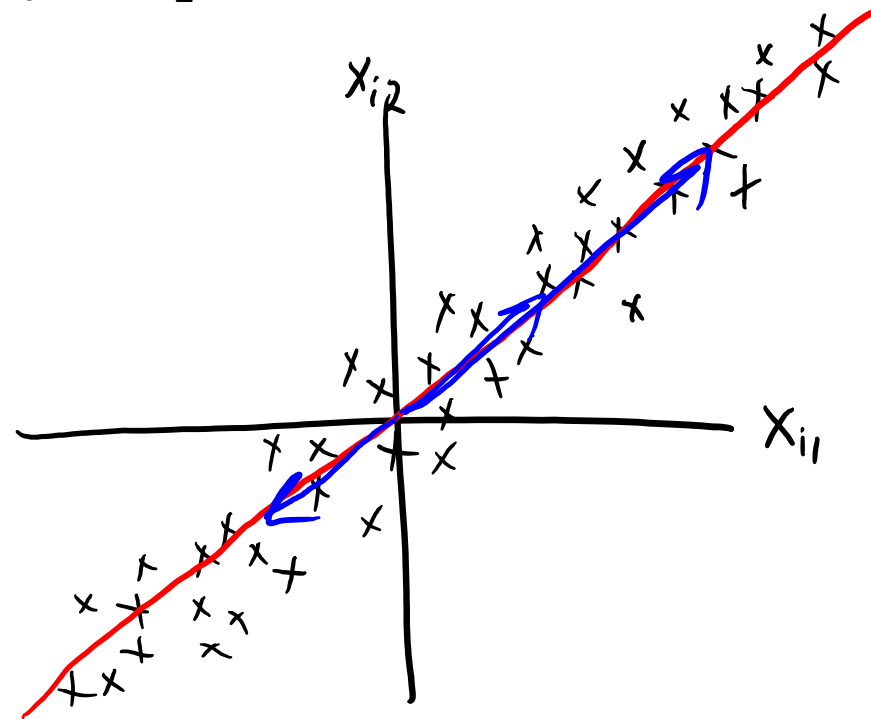
  - PCA solution can only be defined up to scalar multiplication.
    - If (W,Z) is a solution, then (αW,(1/α)Z) is also a solution. $\| (\alpha W)(\frac{1}{\alpha}Z) - X \|_F^2 = \| WZ - X \|_F^2$

Span of 2 Vectors
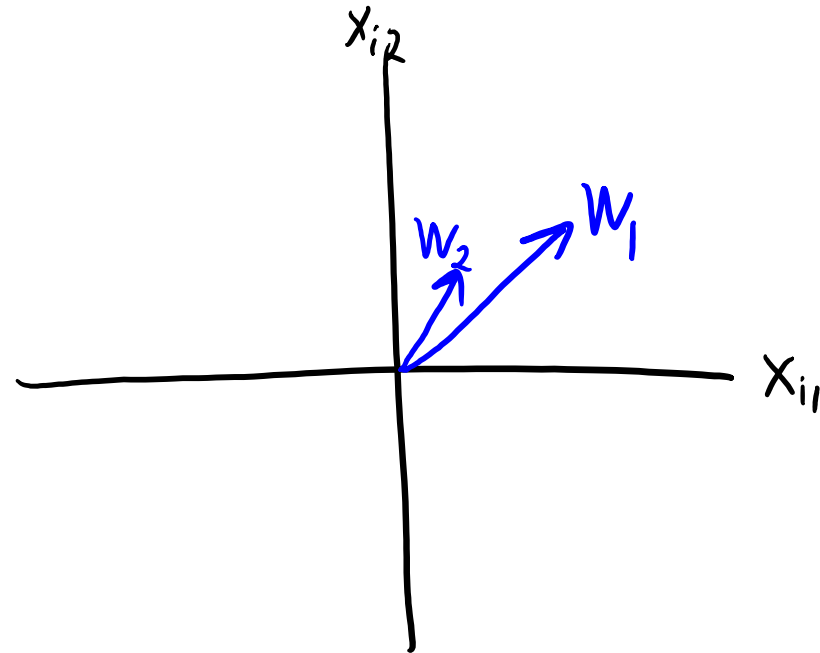
- Consider two vector w_1 and w_2 (k=2).

Span of 2 Vectors

- Consider two vector 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 a scalars z_{i1} and z_{i2}.
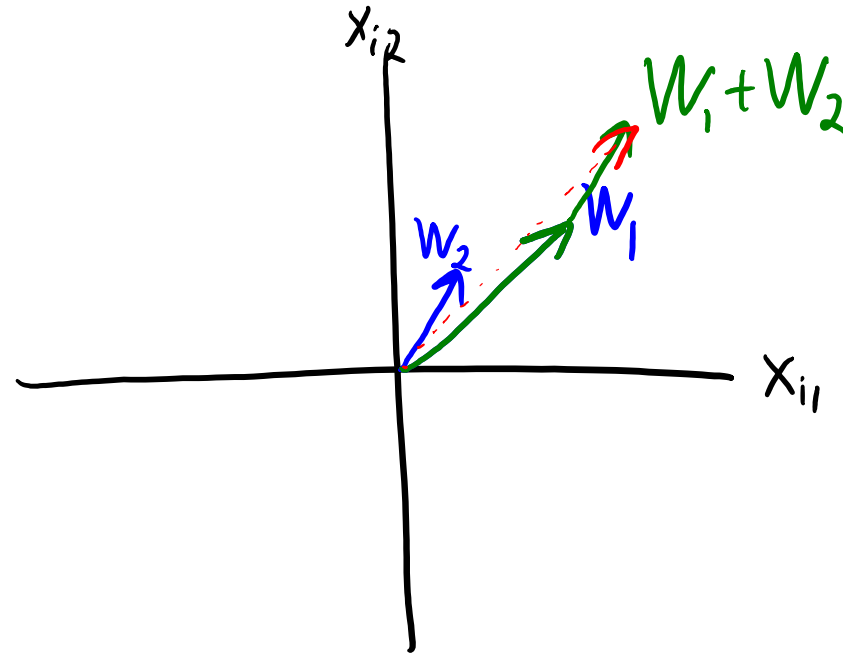
Span of 2 Vectors

- Consider two vector 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 a scalars z_{i1} and z_{i2}.

Span of 2 Vectors

- Consider two vector 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 a scalars z_{i1} and z_{i2}.
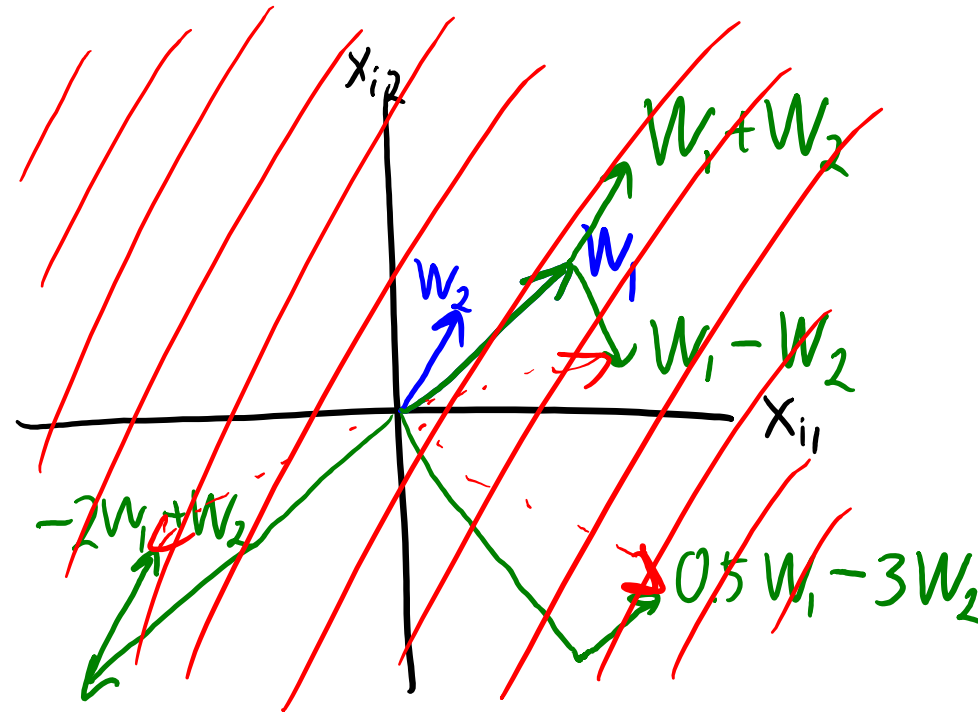
  - For most non-zero 2d vectors, span(w_1,w_2) is a plane.
    - In the case of two vectors in R^2, the plane will be *all* of R^2.

Span of 2 Vectors

- Consider two vector 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 a scalars z_{i1} and z_{i2}.
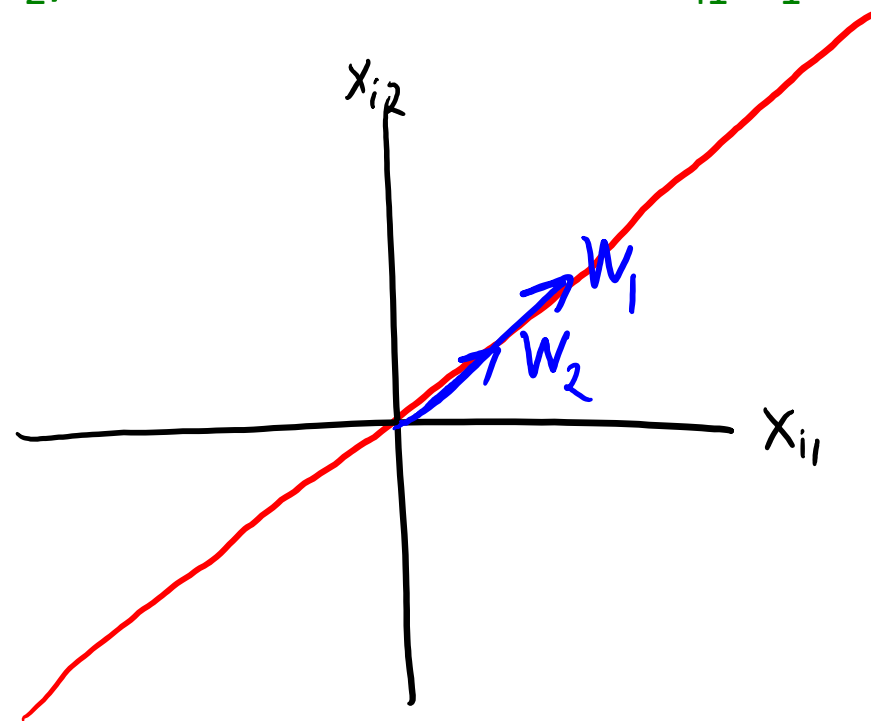
  - For most non-zero 2d vectors, span(w_1,w_2) is plane.
    - Exception is if w_2 is in span of w_1 ("collinear"), then span(w_1,w_2) is just a line.

Span of 2 Vectors

- Consider two vector w_1 and w_2 (k=2).
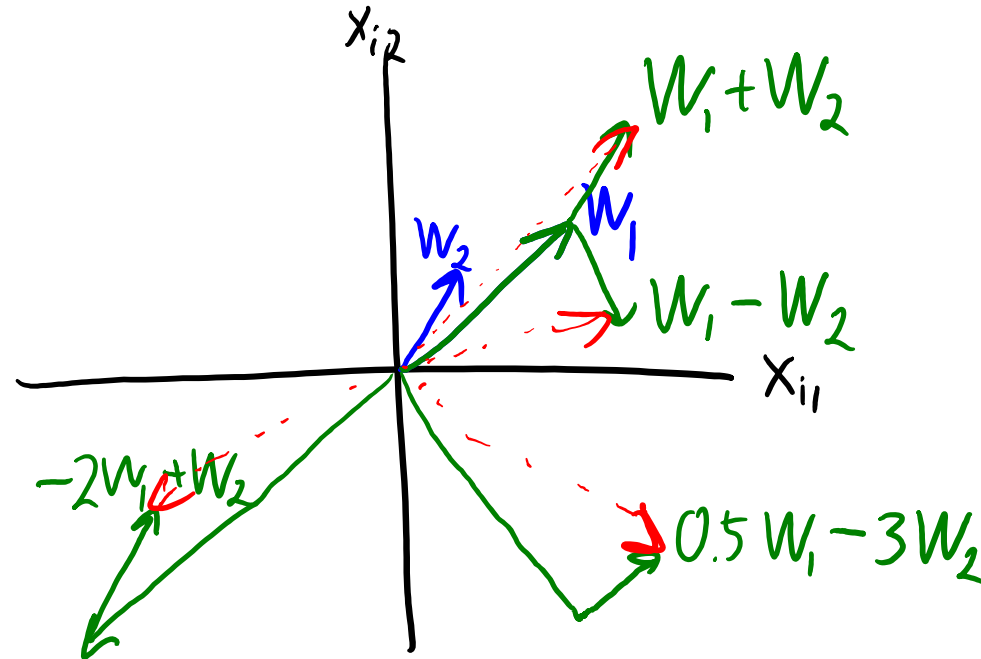 - The span(w_1,w_2) is all vectors of form $z_{i1}w_1 + z_{i2}w_2$ for a scalars z_{i1} and z_{i2}.
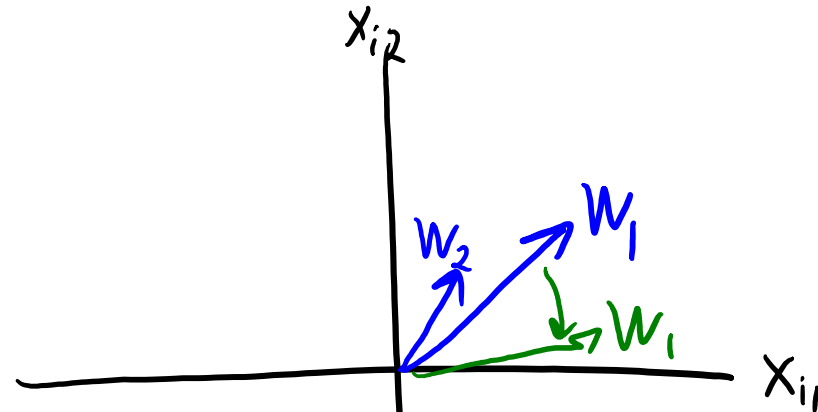
  - New issues for PCA (k >= 2):
    - We have label switching: span(w_1,w_2) = span(w_2,w_1).
    - We can rotate factors within the plane (if not rotated to be collinear).

Span of 2 Vectors

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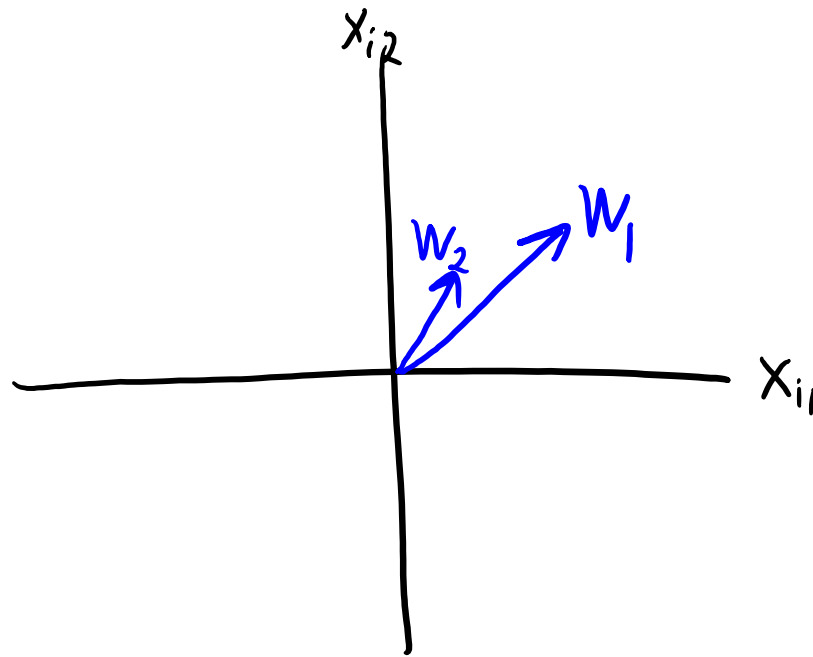
Span of 2 Vectors

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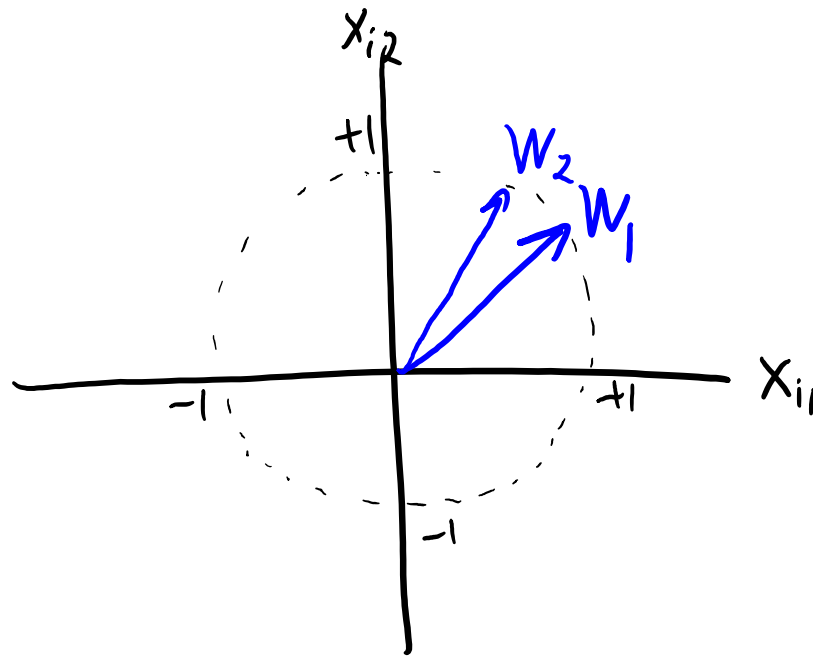
Span of 2 Vectors

- 2 tricks to make vectors defining a plane "more unique":
  - Normalization: enforce that $||w_1|| = 1$ and $||w_2|| = 1$.
  - Orthogonality: enforce that $w_1^{\mathsf{T}} w_2 = 0$ ("perpendicular").
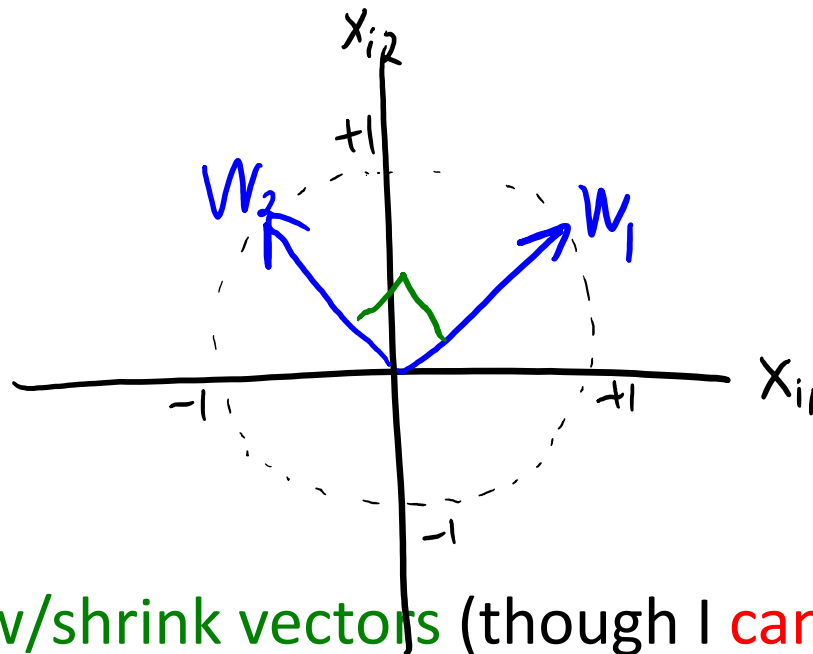
  - Now I can't grow/shrink vectors (though I can still reflect).
  - Now I can't rotate one vector (but I can still rotate *both*).

Digression: PCA only makes sense for k ≤ d

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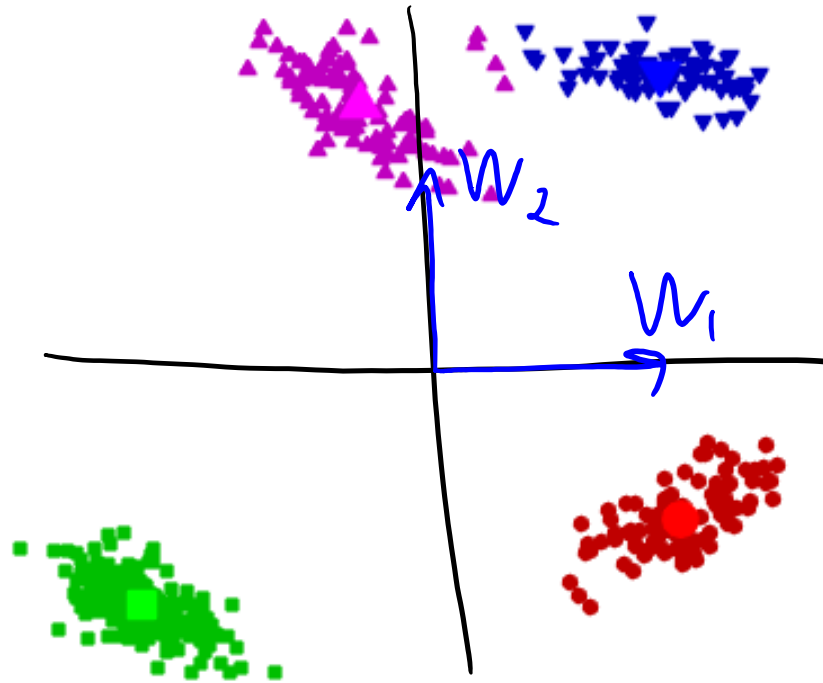
- It doesn't make sense to use PCA with k=4 on this dataset.
  - We only need two vectors [1 0] and [0 1] to exactly represent all 2d points.
    - With k=2, I could set Z=X and W=I to get X=ZW exactly.

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).
  - Span of 3 non-zeros vectors $\{w_1, w_2, w_3\}$ is a 3d space (if not "coplanar").
  - ...

- 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 (More) Unique

- We've identified several reasons that optimal W is non-unique:
  - I can multiply any w_c by any non-zero α.
  - I can rotate any w_c almost arbitrarily within the span.
  - I can switch any w_c with any other $w_{c'}$.

- PCA implementations add constraints to make solution unique:
  - Normalization: we enforce that $||w_c|| = 1$.
  - Orthogonality: we enforce that $w_c^\top w_{c'} = 0$ 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.

Basis, Orthogonality, Sequential Fitting
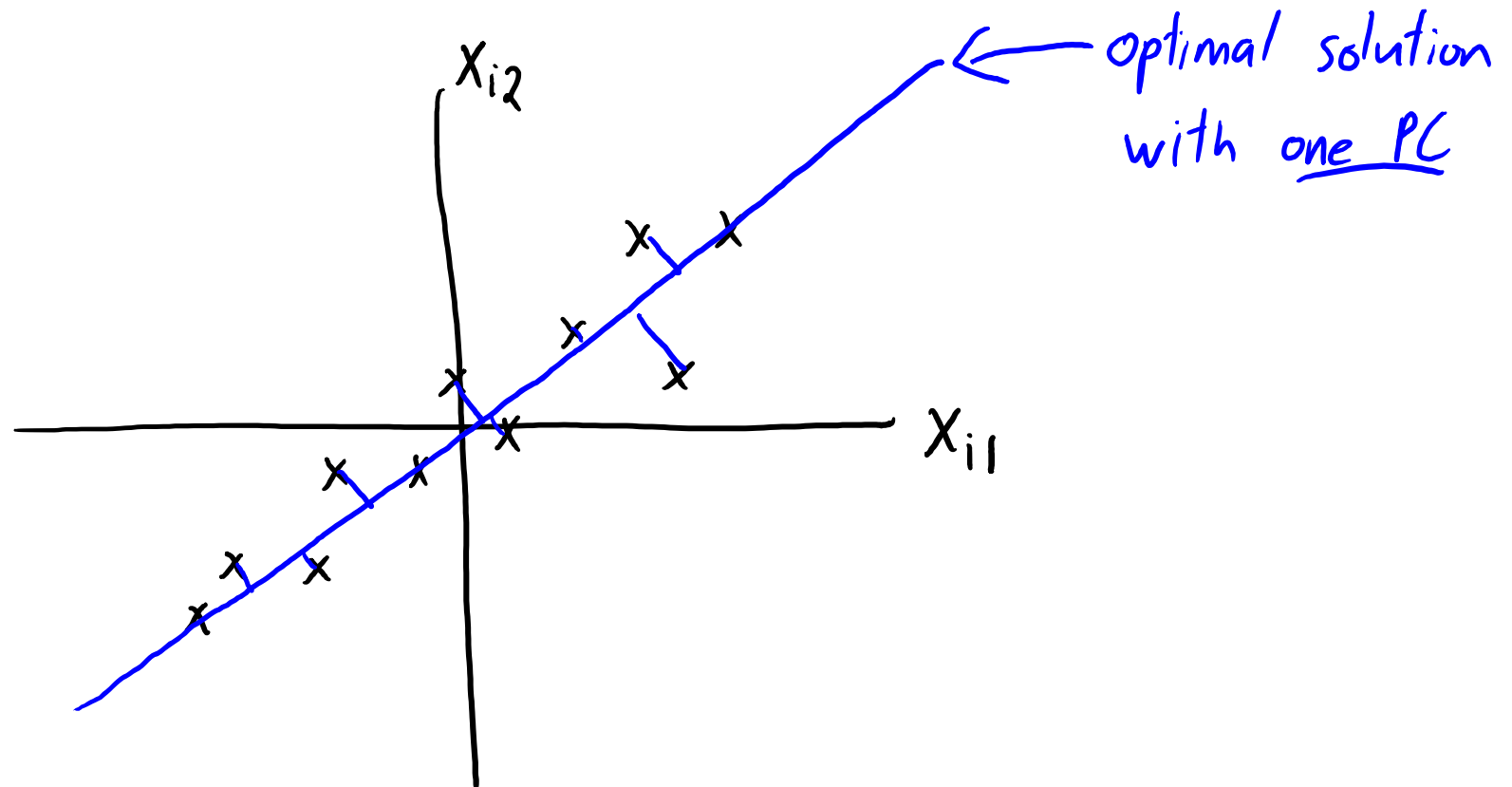
Basis, Orthogonality, Sequential Fitting

Any non-parallel line
gives optimal solution
to second PC (when d=2)

I can get 0 error
on every data point.

X_{i2}

X_{i1}

optimal solution
with one PC

An optimal solution but not orthogonal.

(both PCs give similar information)

Basis, Orthogonality, Sequential Fitting

Any non-parrallel line
gives optimal solution
to second PC (when $d=2$)

I can get 0 error
on every data point.

X_{i2}

X_{i1}
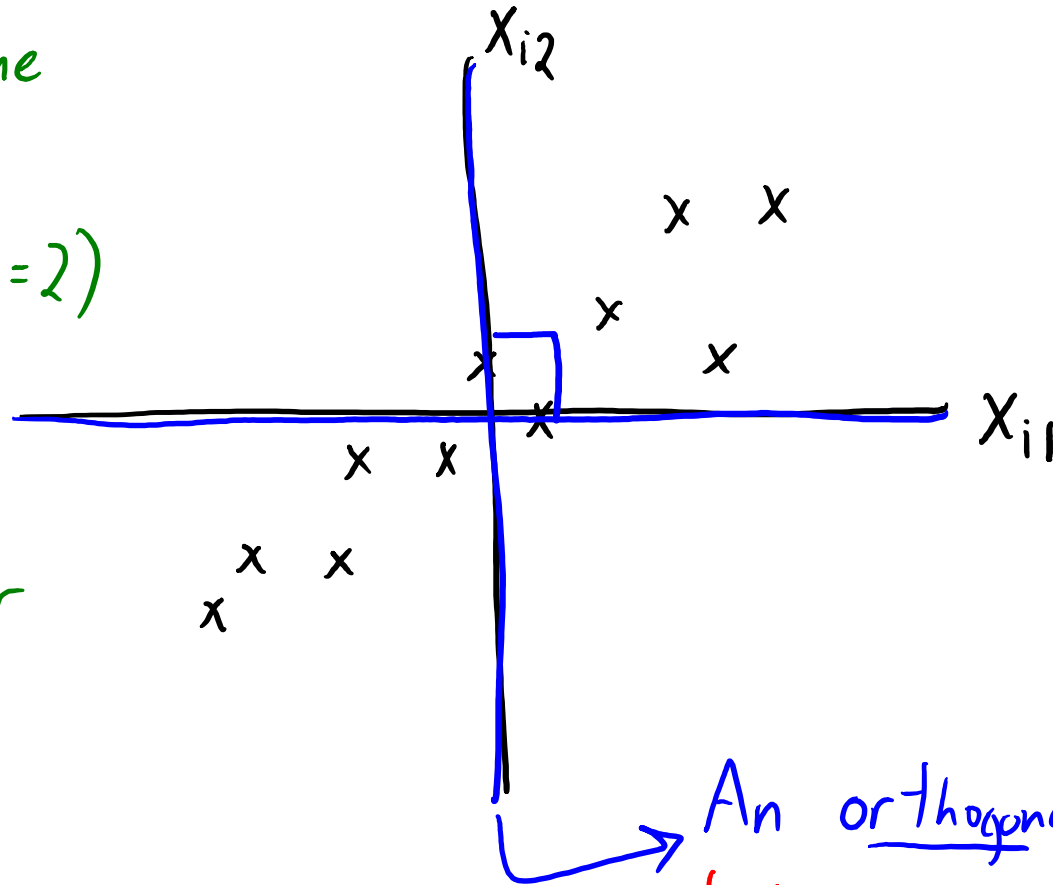
An orthogonal solution (PCs are not redundant)
but PCs have nothing to do with data

Basis, Orthogonality, Sequential Fitting

optimal solution
with one PC
("first" PC)

optimal solution that is
orthogonal to "first" PC
(sequential fitting of PCs
makes directions unique)

http://setosa.io/ev/principal-component-analysis
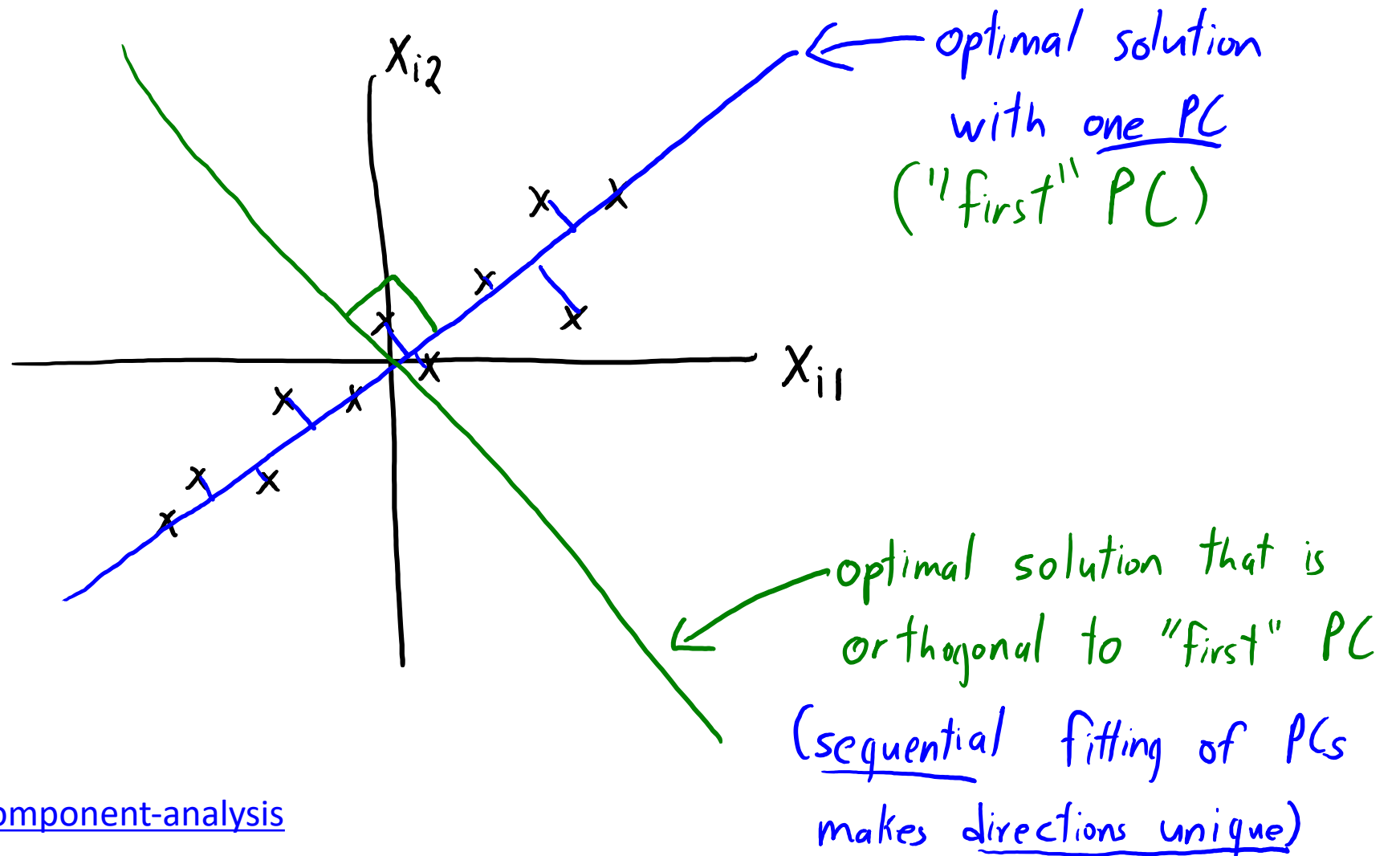
Next Topic: Optimizing the PCA Objective

Finding the First Principal Axis

Let $\mathbf{z}_1 \in \mathbb{R}^N$ be the coefficients associated with $\mathbf{w}_1$ across all the N data points

We first find $\mathbf{w}_1$ and $\mathbf{z}_1$. We assume that $\mathbf{w}_i$ are unit vectors.

$$
\mathcal{L}(\mathbf{w}_1, \mathbf{z}_1) = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - z_{n,1}\mathbf{w}_1)^{\mathrm{T}}(\mathbf{x}_n - z_{n,1}\mathbf{w}_1)
$$

$$
= \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_n - 2z_{n,1}\mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1 + z_{n,1}^2 \mathbf{w}_1^{\mathrm{T}}\mathbf{w}_1)
$$

Thus, we get

$$
z_{n,1} = \frac{\mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1}{\mathbf{w}_1^{\mathrm{T}}\mathbf{w}_1} = \mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1
$$

Finding the First Principal Axis

Back-substituting $z_{n,1}$ into $\mathcal{L}(\mathbf{w}_1, \mathbf{z}_1)$, we get

$$
\mathcal{L}(\mathbf{w}_1, \mathbf{z}_1) = -\frac{1}{N}\sum_{n=1}^{N}\mathbf{w}_1^{\mathrm{T}}\mathbf{x}_n\mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1 + \mathrm{const}
$$
$$
= -\mathbf{w}_1^{\mathrm{T}}\mathbf{\Sigma}\mathbf{w}_1 + \mathrm{const}
$$

Here, $\mathbf{\Sigma}$ is the empirical covariance matrix

To minimize $-\mathbf{w}_1^{\mathrm{T}}\mathbf{\Sigma}\mathbf{w}_1$, we add a constraint $\mathbf{w}_1^{\mathrm{T}}\mathbf{w}_1 = 1$ to prevent trival solution

Finding the First Principal Axis

The constrained optimization problem is as follows:

$$\mathcal{L}(\mathbf{w}_1, \mathbf{z}_1) = -\mathbf{w}_1^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{w}_1 + \lambda(\mathbf{w}_1^{\mathrm{T}}\mathbf{w}_1 - 1)$$

We further get $-2\boldsymbol{\Sigma}\mathbf{w}_1 + 2\lambda\mathbf{w}_1 = 0 \Leftrightarrow \boldsymbol{\Sigma}\mathbf{w}_1 = \lambda\mathbf{w}_1$

To minimize $-\mathbf{w}_1^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{w}_1 = -\lambda$, $\mathbf{w}_1$ must be the first eigenvector of $\boldsymbol{\Sigma}$ with the largest eigen value

We can then find $\mathbf{z}_2 \in \mathbb{R}^N$ and $\mathbf{w}_2 \in \mathbb{R}^D$ after subtracting $\mathbf{z}_1\mathbf{w}_1^{\mathrm{T}}$ from $\mathbf{X}$

This process can be repeated and we get $\mathbf{W}$ whose rows (for our course) are the top K eigenvectors of the empirical covariance matrix $\boldsymbol{\Sigma}$

Maximizing the Variance of the Projected Data

Let $\mathbf{z}_1 \in \mathbb{R}^N$ be the coefficients associated with $\mathbf{w}_1$ across all the N data points

$$\mathbb{E}(Z_{n,1}) = \mathbb{E}(\mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1) = (\mathbb{E}(\mathbf{x}_n))^{\mathrm{T}}\mathbf{w}_1 = 0$$

$$\mathbb{V}(Z_{n,1}) = \frac{1}{N}\sum_{n=1}^{N}\mathbf{w}_1^{\mathrm{T}}\mathbf{x}_n\mathbf{x}_n^{\mathrm{T}}\mathbf{w}_1 = \mathbf{w}_1^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{w}_1 = \lambda_1$$
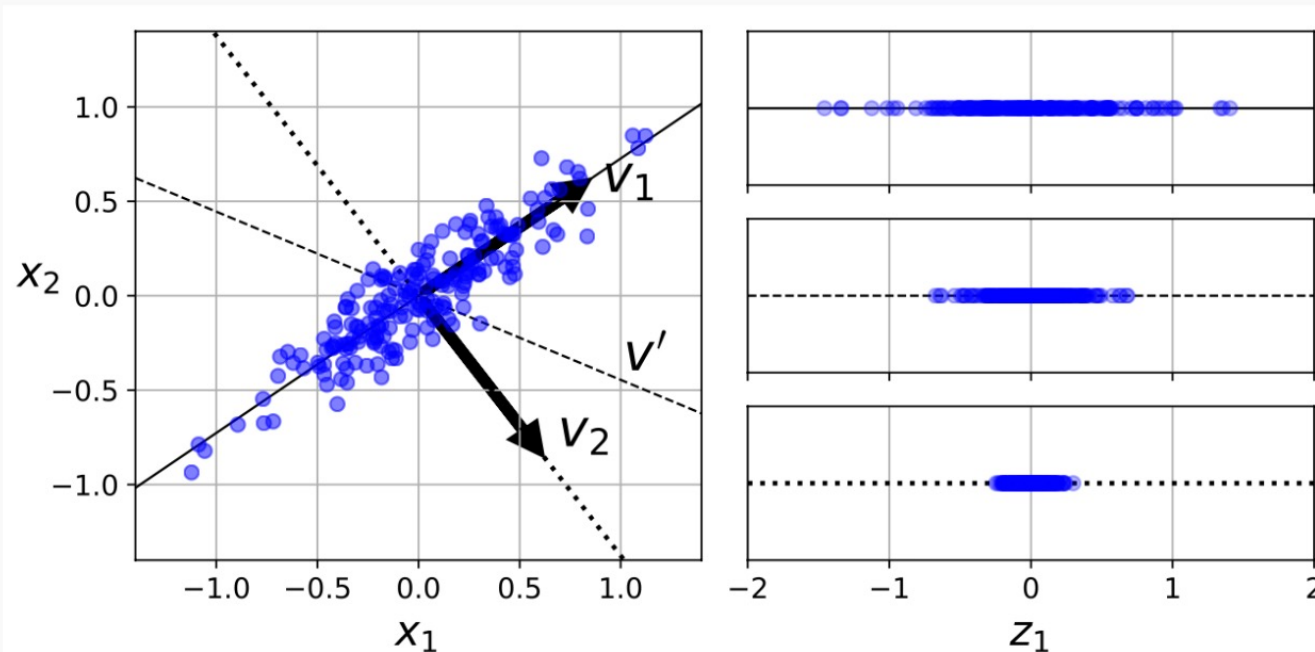
PCA Workflow
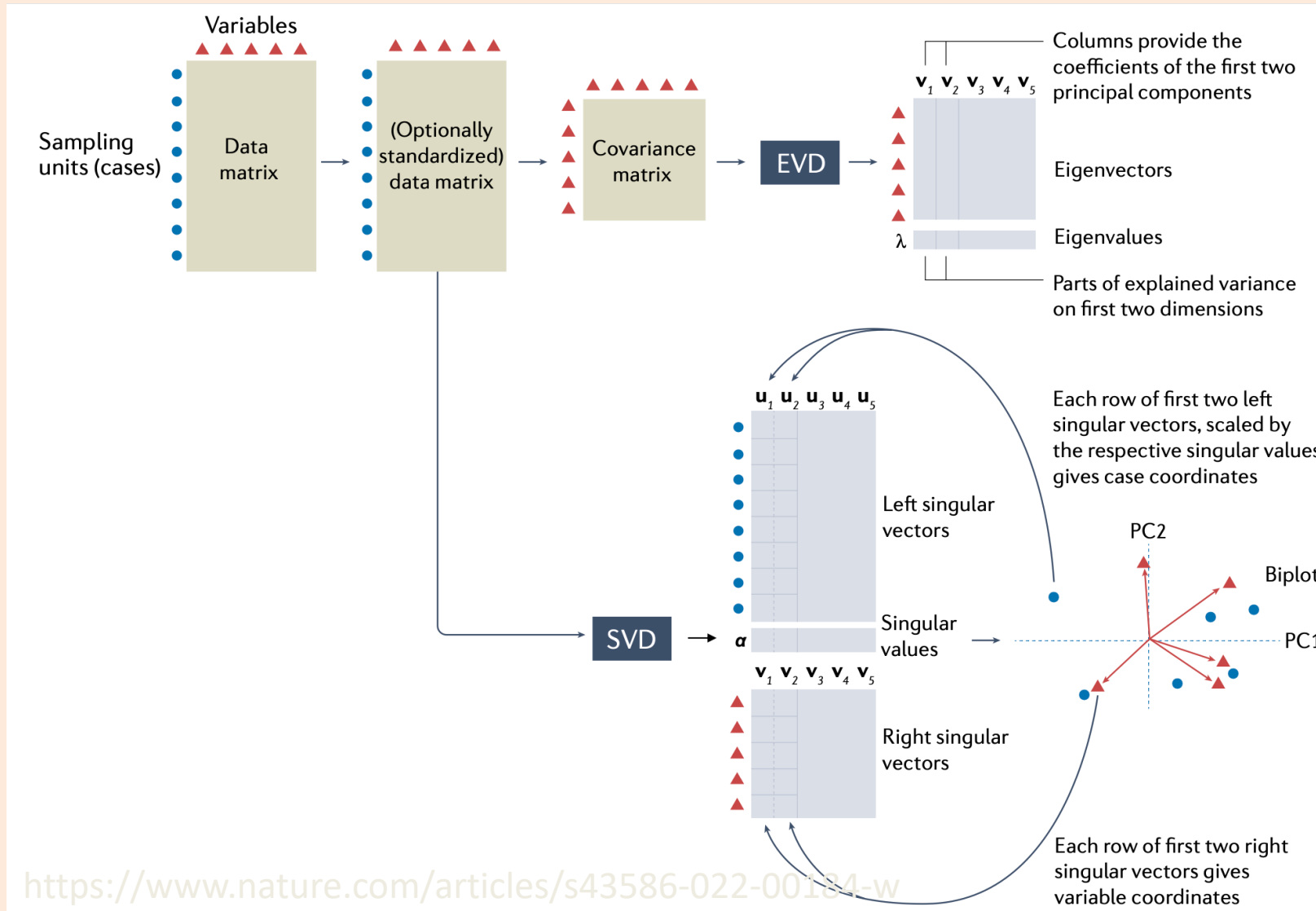
PCA Computation: SVD

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

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

- Computing $\tilde{Z}$ is cheaper now:

$$\tilde{Z} = \tilde{X} W^\top (WW^\top)^{-1} = \tilde{X} W^\top$$

$$WW^\top = \begin{bmatrix} - w_1 - \\ - w_2 - \\ \vdots \\ - w_K - \end{bmatrix} \begin{bmatrix} | & | & & | \\ w_1^\top & w_2^\top & \cdots & w_K^\top \\ | & | & & | \end{bmatrix}$$

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

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 to form $\tilde{Z}$:

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

$\longrightarrow$ means of Training data

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

$$\tilde{Z} = \tilde{X} W^T (WW^T)^{-1}$$

(could just store this $d \times k$ matrix)

Choosing 'k' by "Variance Explained"

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

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

definition of variance

assumed to be **zero**

definition of expectation

Frobenius norm

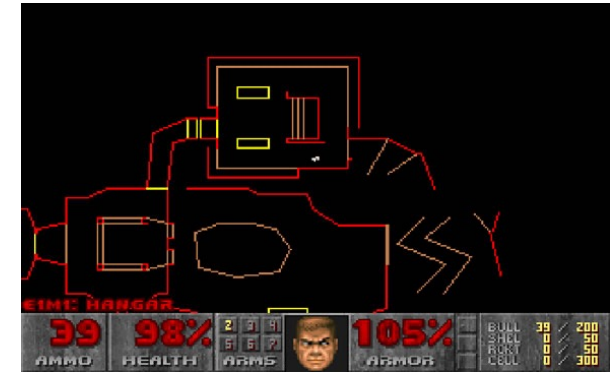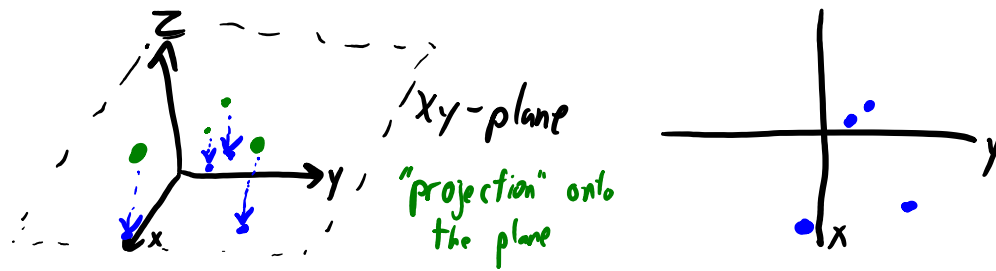– For a given 'k' we compute (variance of errors)/(variance of x_{ij}):

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

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:

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

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

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

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

Summary

- PCA objective:
  - Minimizes squared error between elements of X and elements of ZW.
- Eigenfaces
- 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.
- Choosing 'k':
  - We can choose 'k' to explain "percentage of variance" in the data.

- Next time: cancer signatures and NBA shot charts.