S&DS 265 / 565

Introductory Machine Learning

Unsupervised Learning: PCA

October 11

Yale

Checkpoint

- Assn 2 due Thursday
- Midterm one week from today, in class
- Practice midterms (and solutions) posted on Canvas
- Review sessions to be scheduled

Bagging and Random Forests

- Grow many trees and average their predictions
- Trees are grown deep, to have low bias
- To “decorrelate” the predictors and reduce variance:
  - Grow each tree on a bootstrap sample (bag)
  - For RF: Choose from andom subset of predictors at each step

Using the ensemble of trees

To make a prediction at a new point x using ensemble of trees $T_1, T_2, \ldots, T_B$:

Regression: Average $\widehat{f}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x)$

Classification: Majority vote of the individual trees

Random Forests Algorithm

1. For $b = 1$ to B:
   - **(a)** Draw a bootstrap sample Z^* of size n from the training data
   - **(b)** Grow a random-forest tree T_b to the bootstrapped data, recursively repeating following steps, until minimum node size reached:
     - **i.** Select m variables at random from the p variables
     - **ii.** Pick the best variable/split-point among the m
     - **iii.** Split the node into two children nodes
2. Output the ensemble of trees $\{T_b\}_{b=1}^{B}$.

Bootstrap samples

- We have n data points
- Randomly sample exactly n of them *with replacement*
- Contains about $\frac{2}{3}$ of the data, with duplicates
- This is a "bag" of n data points

Chance a particular example does not appear in sample:

$$\left(1 - \frac{1}{n}\right)^n \longrightarrow \frac{1}{e} \approx 0.37$$

https://inferentialthinking.com/chapters/13/2/Bootstrap.html

Out-of-Bag error estimation

Each bagged tree uses about 2/3 of all observations (with repeats).

The remaining data—*out-of-bag* (OOB) observations—can be put to good use.

Recall: Out-of-Bag error estimation

	Bagging iteration					OOB Est.
Obs	1	2	3	...	B	
1	OOB	train	train	...	train	$\widehat{y}_1$
2	train	OOB	train	...	train	$\widehat{y}_2$
3	train	train	OOB	...	train	$\widehat{y}_3$
4	OOB	train	train	...	OOB	$\widehat{y}_4$
...	...	...	...	...	...	...
n	train	train	OOB	...	train	$\widehat{y}_n$

Out-of-Bag error estimation

What does this table mean?

- Consider bags $\mathcal{B}_i \subset \{1, 2, \ldots, B\}$ where (x_i, y_i) does *not* appear
- We average over those trees to form a prediction:

$$\widehat{y}_i = \frac{1}{|\mathcal{B}_i|} \sum_{b \in \mathcal{B}_i} T_b(x_i)$$

- OOB error estimate is then $\frac{1}{n} \sum_i (y_i - \widehat{y}_i)^2$

Demo code

Bagging

A quick little demo of the bagging operations

In [1]:
```
import numpy as np
```

In [2]:
```
n=15

# random response vector
y = np.round(np.random.normal(size=n), 3)

# add a little noise
yhat = np.round(y + .01* np.random.normal(size=n), 3)
```

In [3]:
```
# Choose a random bag of data, by sampling with replacement
bag = np.random.choice(range(n), replace=True, size=n)
bag
```

Out[3]:
```
array([12,  0, 14,  7,  0,  2,  5,  6,  6, 11,  2,  2,  1,  3,  2])
```

Unsupervised Learning

Supervised learning is about being able to predict out Y from predictors $X_1, X_2, \ldots, X_p$

Unsupervised learning deals with data that do not have labels Y

We are not trying to predict anything—What are we trying to learn?

Unsupervised Learning

Supervised learning is about being able to predict out Y from predictors $X_1, X_2, \ldots, X_p$

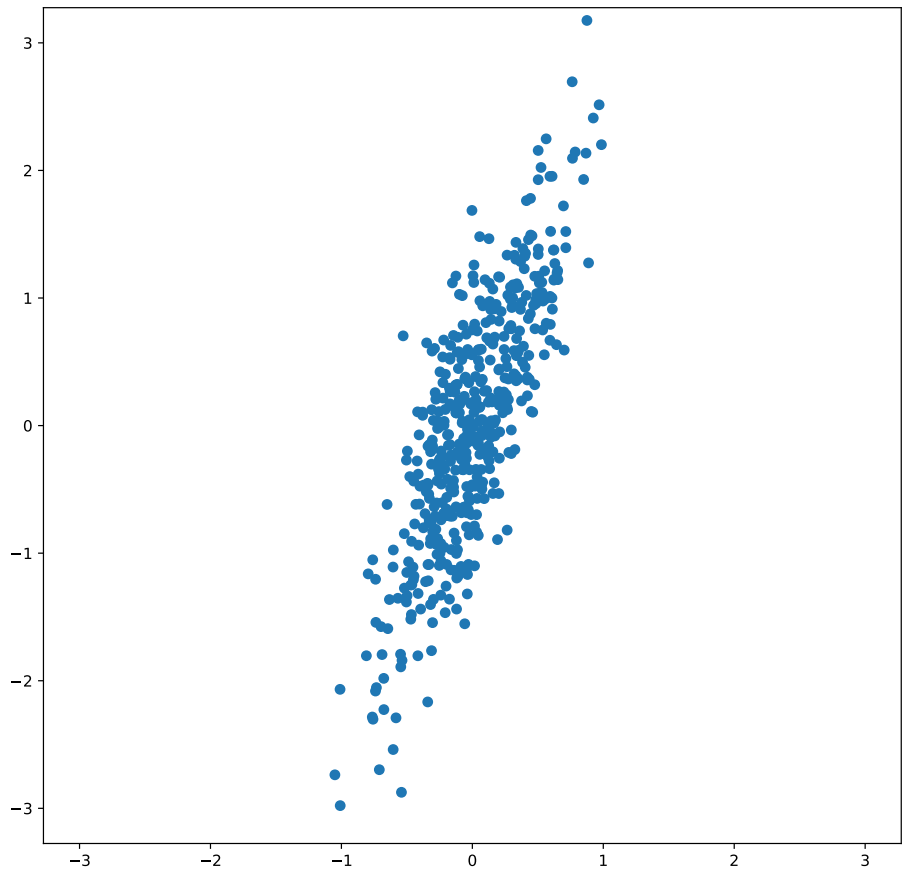
Unsupervised learning deals with data that do not have labels Y

We are not trying to predict anything—What are we trying to learn?

- Are there interesting ways to visualize/summarize the data?
- Are there natural subgroups in the data?
- What are the important types of variation in the data?

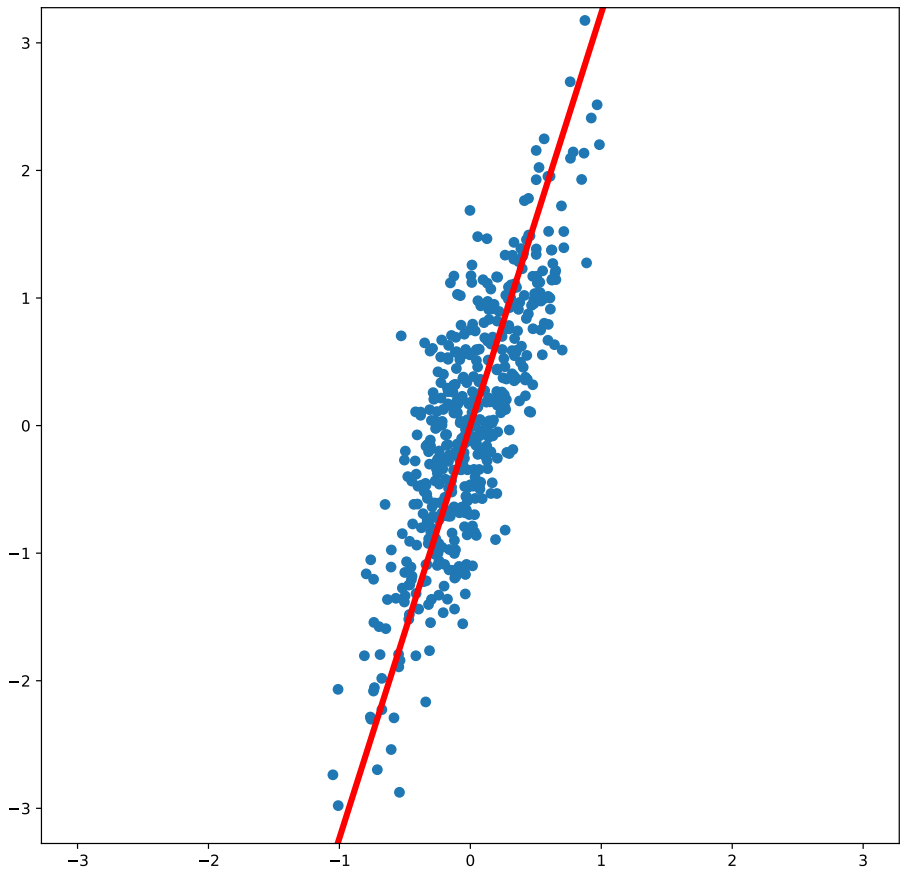
Principal Component Analysis (PCA)

PCA finds the directions of greatest variability in the data.

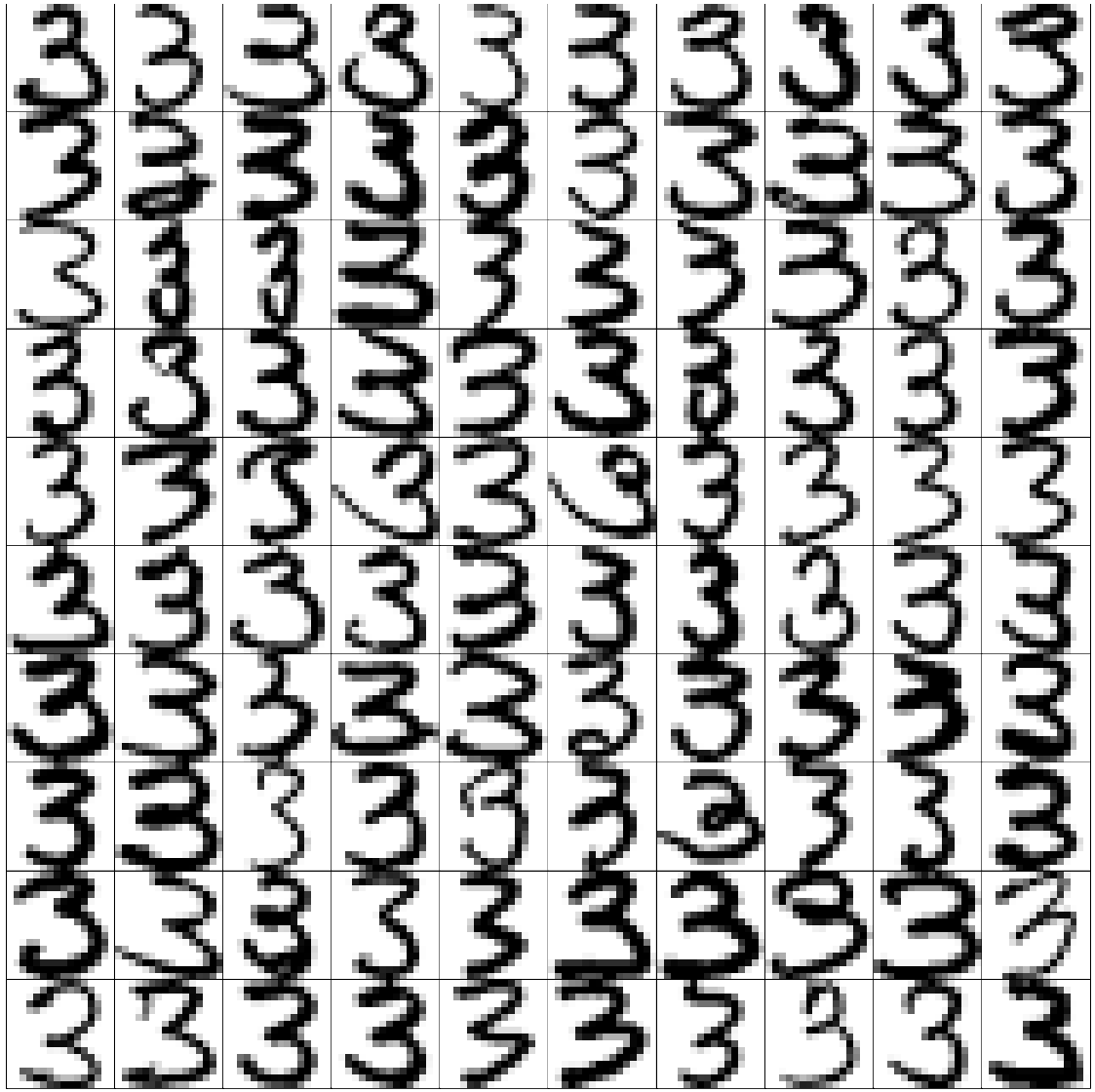


Principal Component Analysis (PCA)

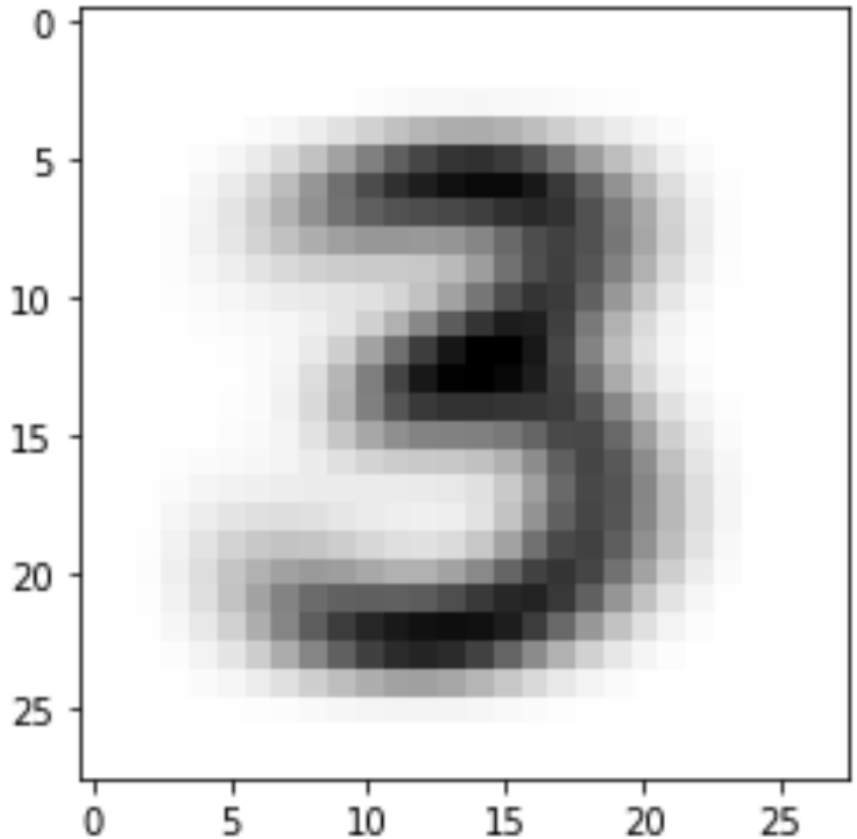
PCA finds the directions of greatest variability in the data.



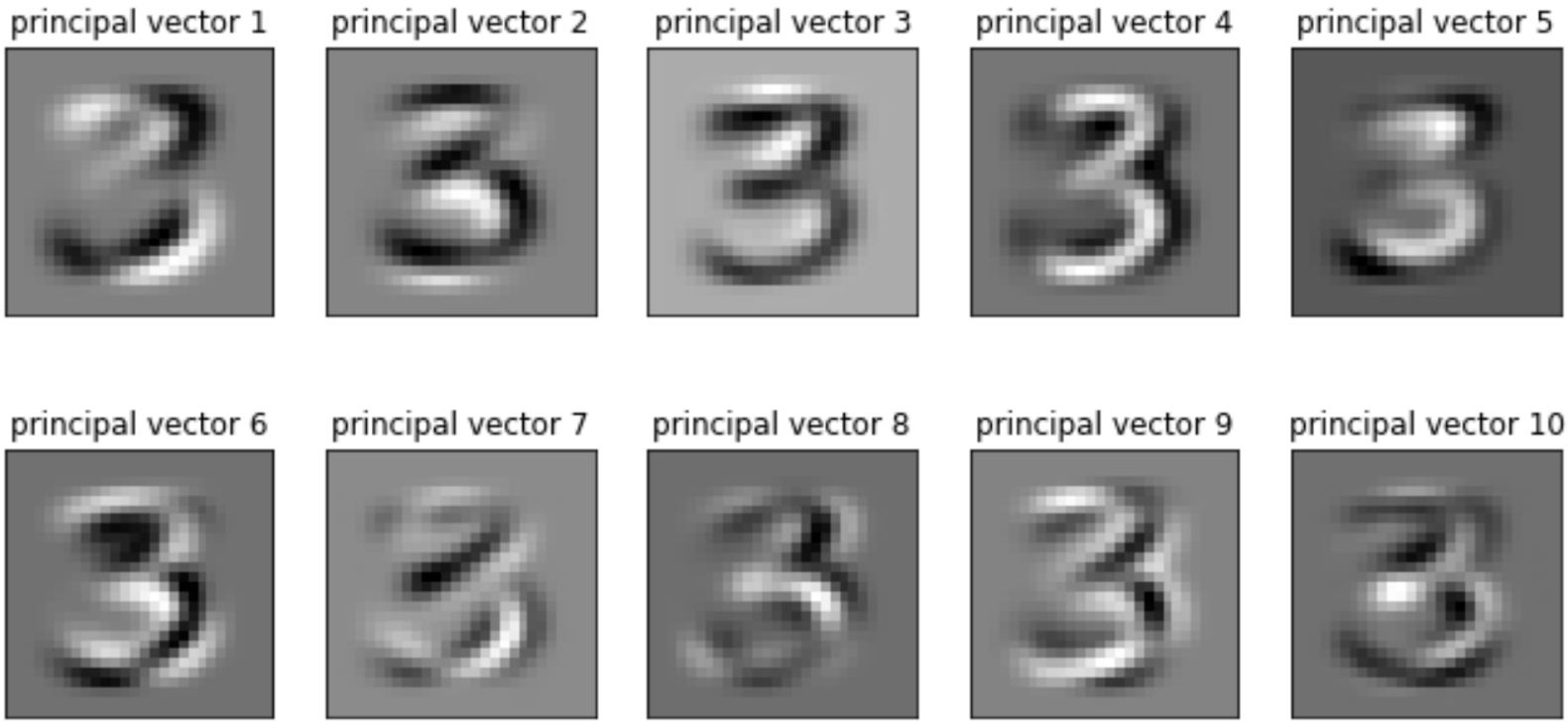
Handwritten Digits (3s)



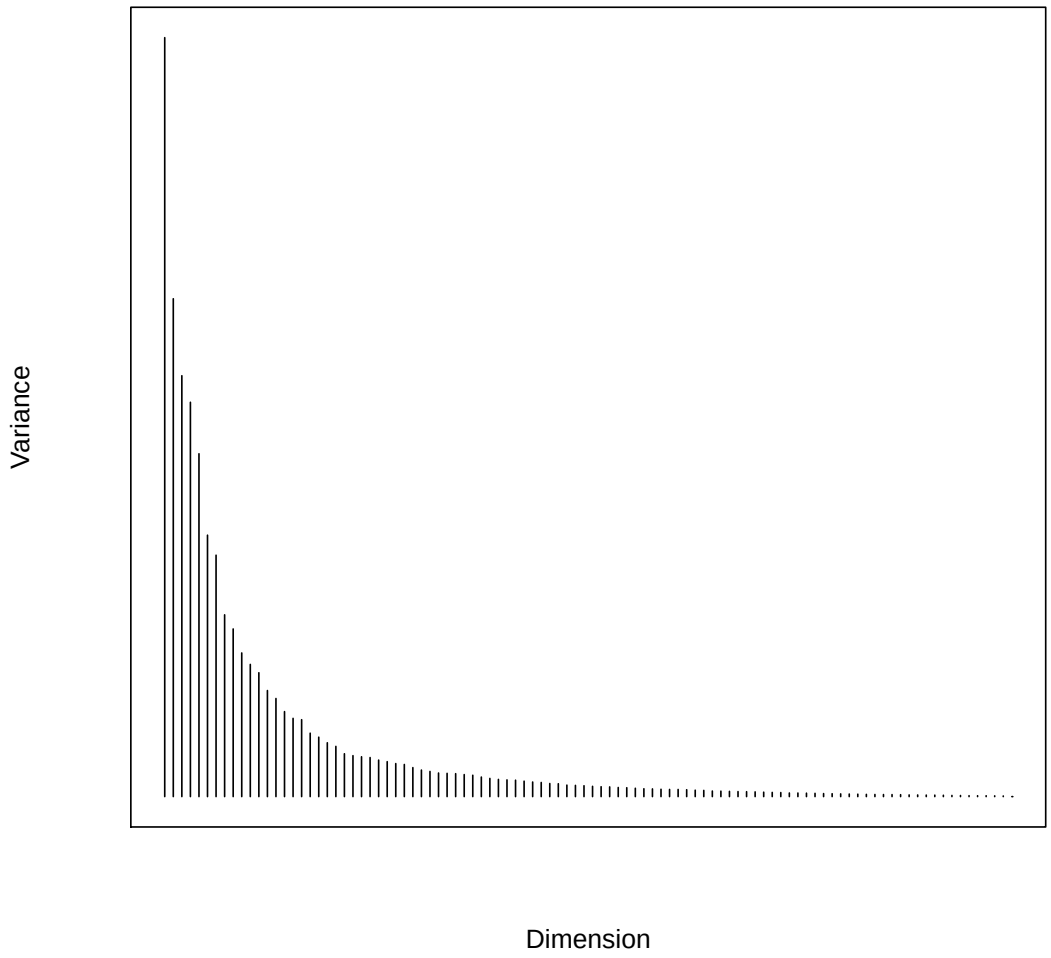
Handwritten Digits (3s) – Average



Handwritten Digits (3s) – Principal vectors



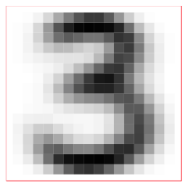
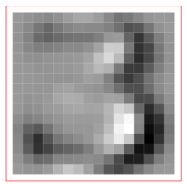
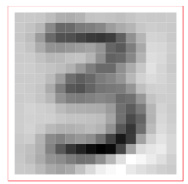
Handwritten Digits (3s) – PCA variance



Handwritten Digits (3s)

$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

$$= \bar{x} + \lambda_1 \cdot v_1 + \lambda_2 \cdot v_2.$$

Approximation

The way to read this is:

$$\text{approximation}(x) = \overline{x} + \lambda_1(x) \cdot v_1 + \lambda_2(x) \cdot v_2$$

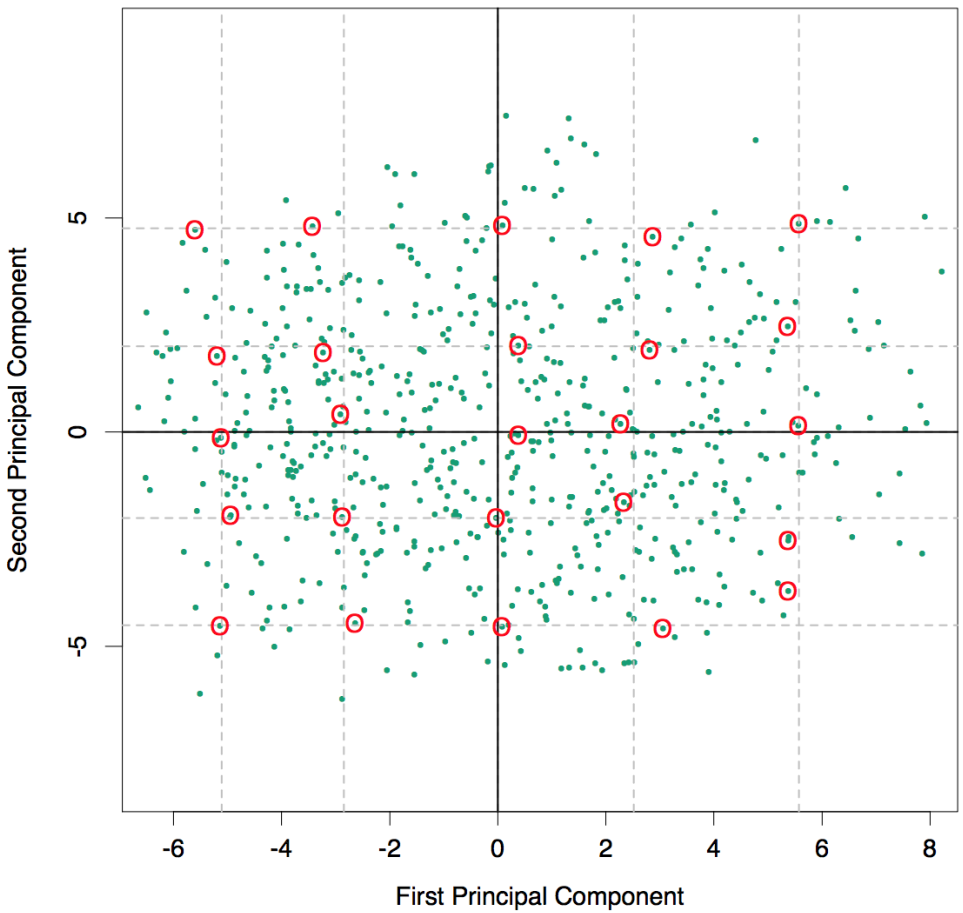
where

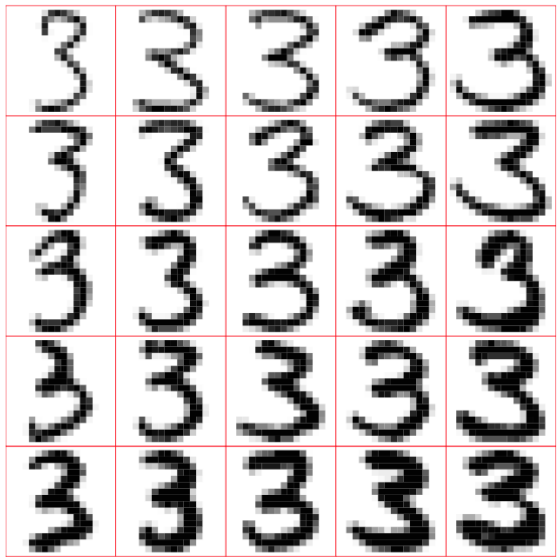
$$\begin{aligned} \overline{x} &: \quad \text{average 3} \\ v_i &: \quad \text{the } i\text{-th principal vector} \\ \lambda_i(x) &: \quad \text{the } i\text{-th principal component for image } x \end{aligned}$$

Handwritten Digits (3s) – Top 2 components

$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

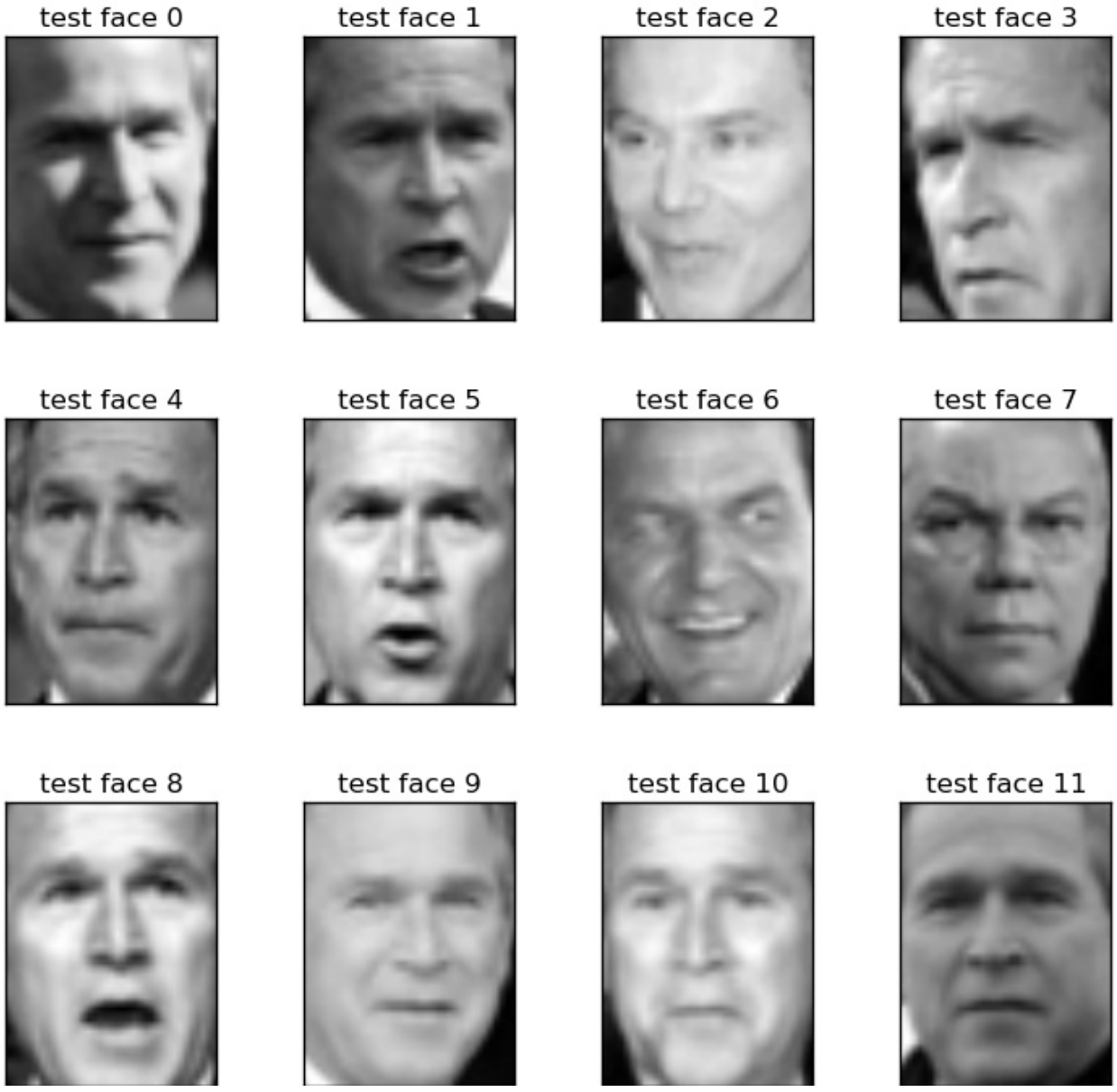
$$= \bar{x} + \lambda_1 \cdot v_1 + \lambda_2 \cdot v_2 .$$



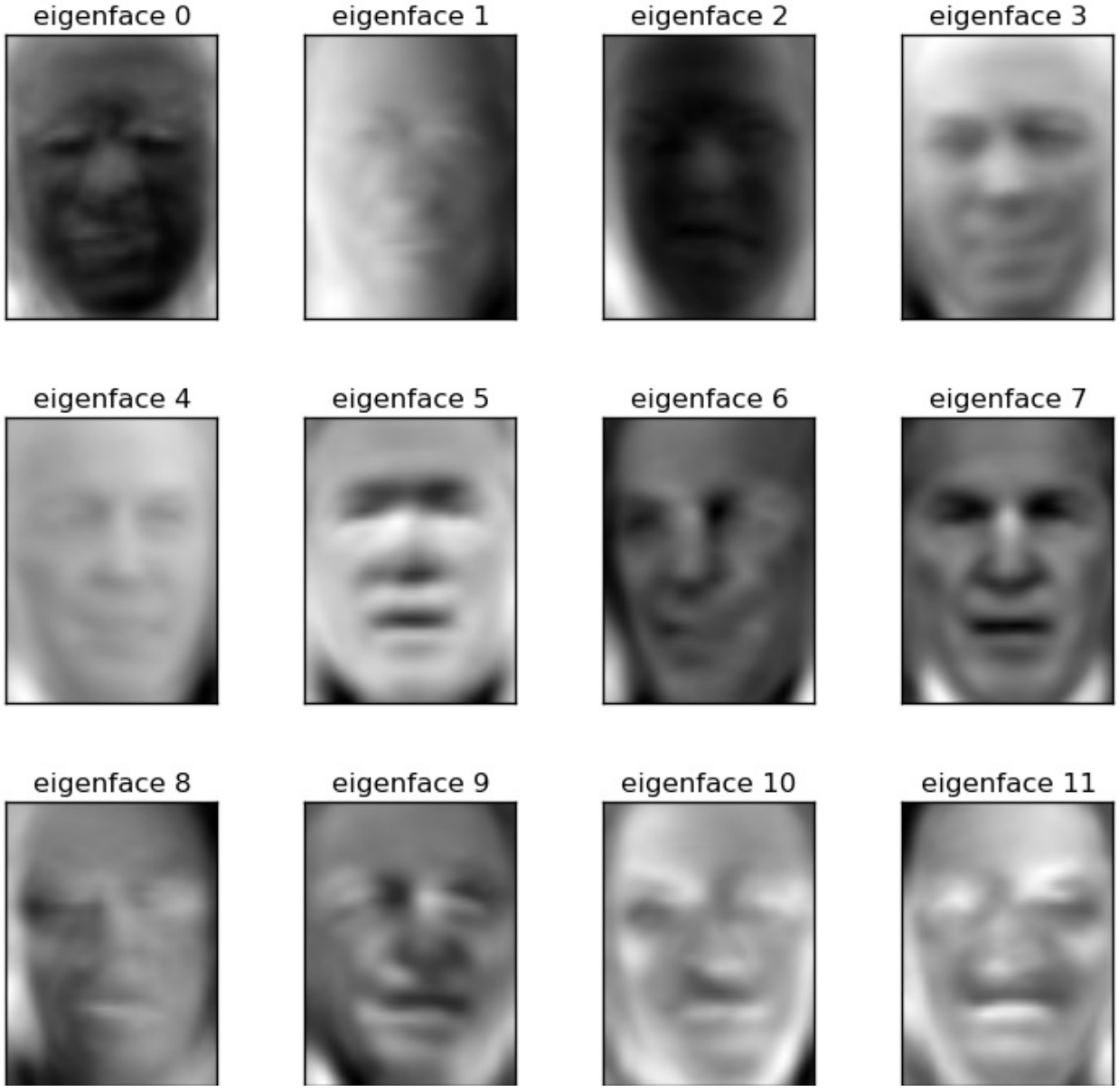


Handwritten Digits (3s) – PCA reconstruction

Faces


test face 0
test face 1
test face 2
test face 3
test face 4
test face 5
test face 6
test face 7
test face 8
test face 9
test face 10
test face 11

Eigenfaces (principal vectors)



Societal bias in image databases

Original Research | Open Access | Published: 27 October 2021

Bias, awareness, and ignorance in deep-learning-based face recognition

Samuel Wehrli, Corinna Hertweck, Mohammadreza Amirian, Stefan Glüge & Thilo Stadelmann

AI and Ethics **2**, 509–522 (2022) | Cite this article

2187 Accesses | **3** Altmetric | Metrics

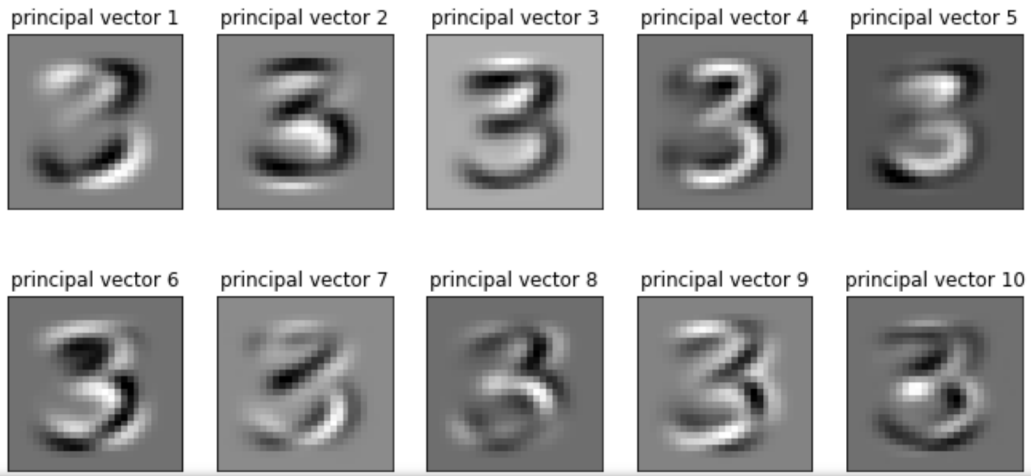
Abstract

Face Recognition (FR) is increasingly influencing our lives: we use it to unlock our phones; police uses it to identify suspects. Two main concerns are associated with this increase in facial recognition: (1) the fact that these systems are typically less accurate for marginalized groups, which can be described as “bias”, and (2) the increased surveillance through these systems. Our paper is concerned with the first issue. Specifically, we explore an intuitive technique for reducing this bias, namely “blinding” models to sensitive features, such as gender or race, and show why this cannot be equated with reducing bias. Even when not designed for this task, facial recognition models can deduce sensitive features, such as gender or race, from pictures

Let’s go to the notebook

```
pca = PCA(num_components).fit(cimages)
principal_vectors = pca.components_
principal_vectors = principal_vectors.reshape((num_components, height, width))
pcs = pca.fit_transform(cimages)
capprox = pca.inverse_transform(pcs)
labels = ['principal vector %d' % (i+1) for i in np.arange(num_components)]
plot_images(principal_vectors, labels, height, width, int(num_components/5.), 5)
ratio = pca.explained_variance_ratio_.sum()
print('Variance explained by first %d principal vectors: %.2f%%' % (num_components, ratio*100))
```

```
Variance explained by first 25 principal vectors: 72.46%
```



Published in final edited form as:
Nature. 2008 Nov 6; 456(7218): 98–101.
Published online 2008 Aug 31. doi: 10.1038/nature07331
NIHMSID: NIHMS132060

Genes mirror geography within Europe

John Novembre,[1,2] Toby Johnson,[4,5,6] Katarzyna Bryc,[7] Zoltán Kutalik,[4,6] Adam R. Boyko,[7] Adam Auton,[7] Amit Indap,[7] Karen S. King,[8] Sven Bergmann,[4,6] Matthew R. Nelson,[8] Matthew Stephens,[2,3] and Carlos D. Bustamante[7]

Author information ▶ Copyright and License information ▶

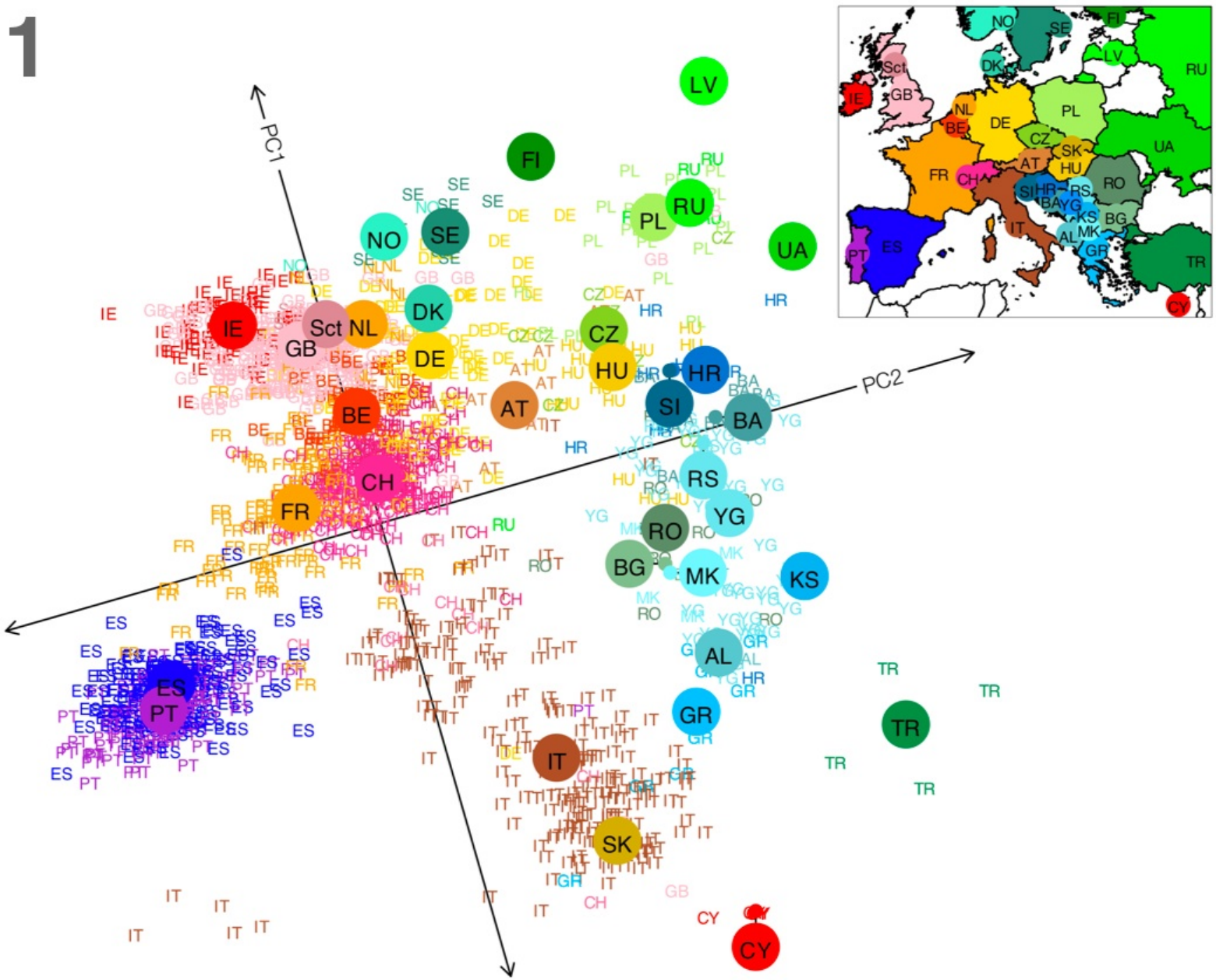
Abstract

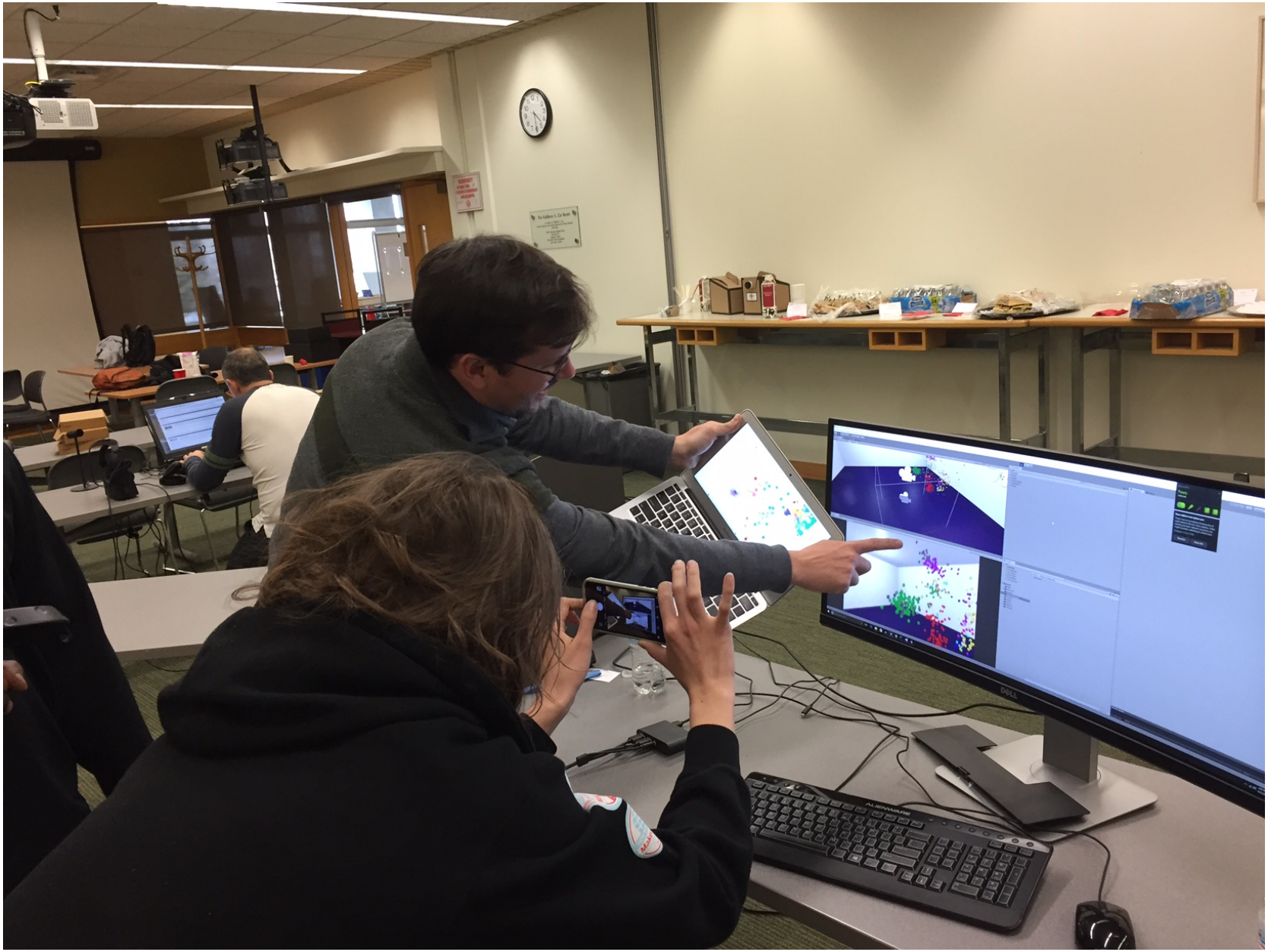
Go to:

Understanding the genetic structure of human populations is of fundamental interest to medical, forensic and anthropological sciences. Advances in high-throughput genotyping technology have markedly improved our understanding of global patterns of human genetic variation and suggest the potential to use large samples to uncover variation among closely spaced populations[1–5]. Here we characterize genetic variation in a sample of 3,000 European individuals genotyped at over half a million variable DNA sites in the human genome. Despite low average levels of genetic differentiation among Europeans, we find a close correspondence between genetic and geographic distances; indeed, a geographical map of Europe arises naturally as an efficient two-dimensional summary of genetic variation in Europeans. The results emphasize that when mapping the genetic basis of a disease phenotype, spurious associations can arise if genetic structure is not properly accounted for. In addition, the results are relevant to the prospects of genetic ancestry testing[6]; an individual's DNA can be used to infer their geographic origin with surprising accuracy—often to within a few hundred kilometres.

https://ghr.nlm.nih.gov/primer/genomicresearch/snp

1


PC1
PC2
LV
FI
RU
PL
NO
SE
UA
DK
IE
Sct
NL
GB
DE
CZ
HU
HR
SI
AT
BE
BA
CH
RS
FR
YG
RO
BG
MK
KS
AL
ES
PT
GR
TR
IT
SK
CY


DELL
ALIENWARE

Examples

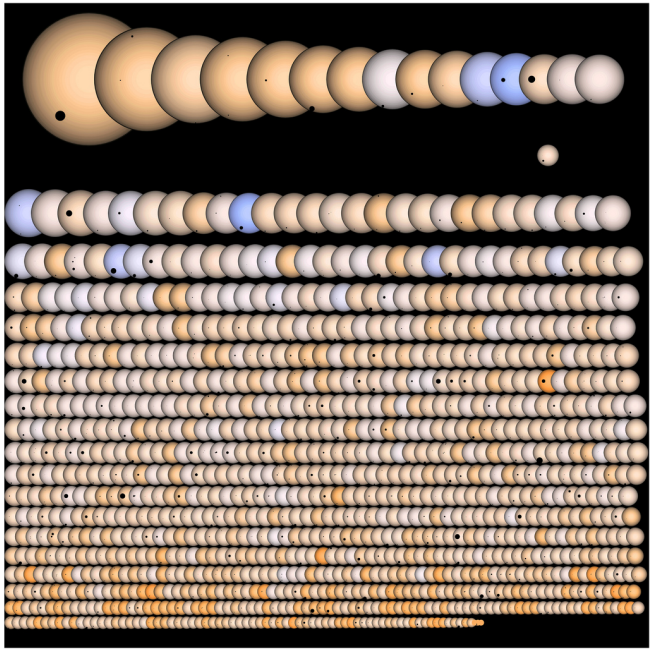
A couple of other interesting examples of PCA follow (without going into any detail)

Robust PCA

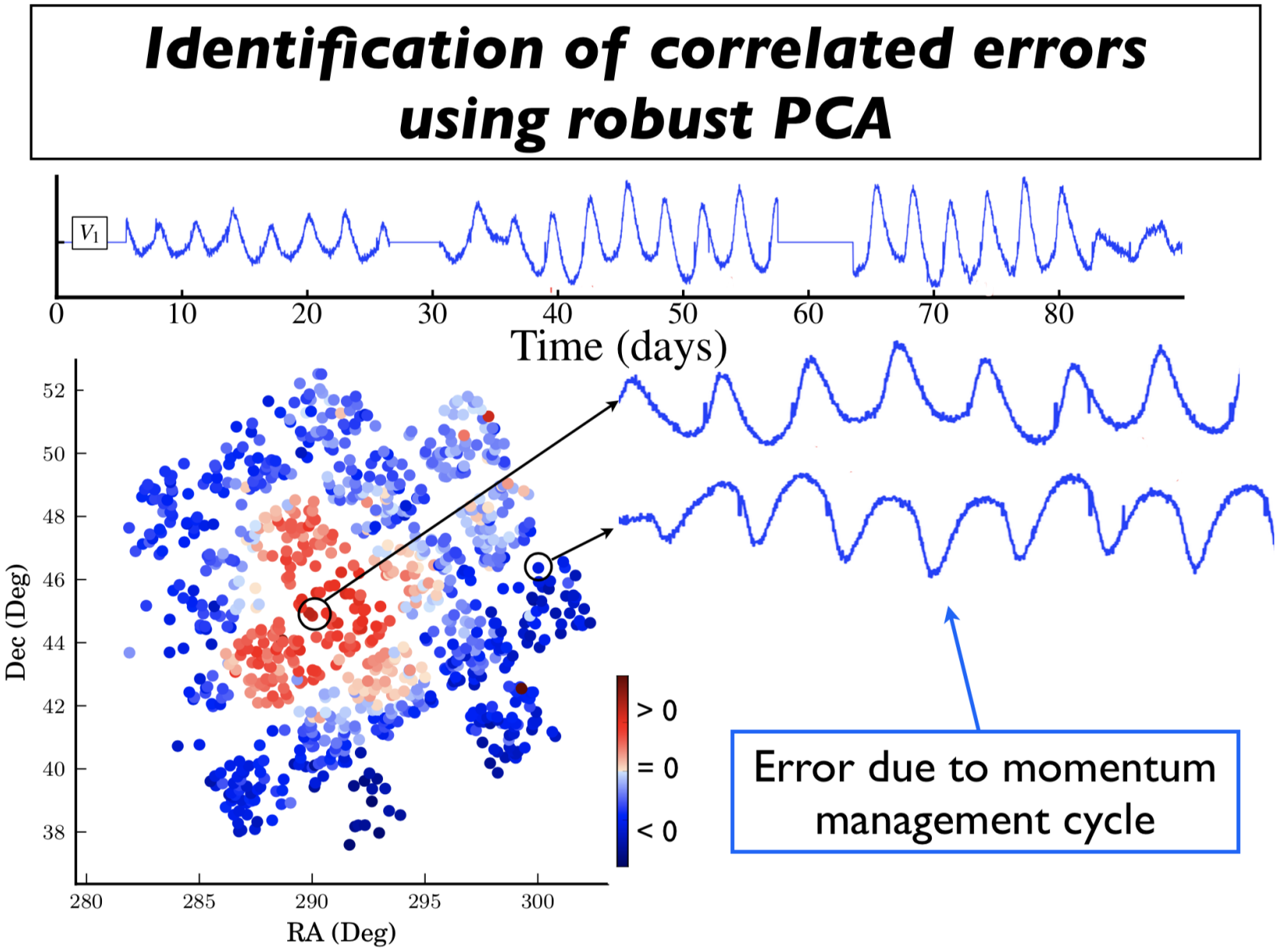
Robust PCA (low rank plus sparse) can be used for background subtraction in video.

https://www.youtube.com/watch?v=BTrbow8u4Cw

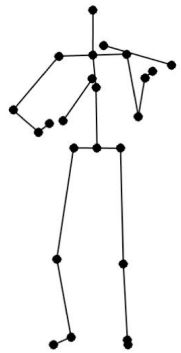
Kepler telescope



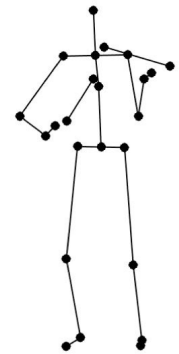
Kepler telescope



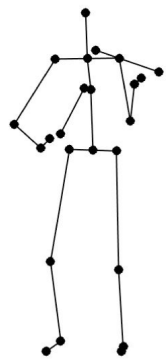
RESULTS: PC VISUALISATION, TECHNICAL VS. NON-TECHNICAL MOTION



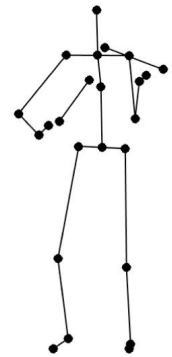
Original



PC1 (45% of variance accounted)
Side-to-side swaying (non-technical)



PC2 (25% of variance accounted)
Right arm string crossing (technical)



PC3 (15% of variance accounted)
Left-right rotation (non-technical)

Aditya Chander

PCA: Algorithm

1. Center the data: $x_i \mapsto x_i - \overline{x}$
2. Compute the $d \times d$ sample covariance $S = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$
3. Find the first k eigenvectors of S
4. Project the data onto those k vectors

PCA: Algorithm

1. Center the data: $x_i \mapsto x_i - \frac{1}{n}\sum_{j=1}^{n} x_j = x_i - \overline{x}$

2. Compute the $d \times d$ sample covariance $S = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$. Note that

$$\frac{1}{n}\sum_{i} \left(x_{ij} - \overline{x}\right)^2$$

   is the sample variance of jth coordinate of data.

3. Find the first k eigenvectors of S,

$$\phi_1, \ldots, \phi_k \in \mathbb{R}^d, \qquad S\phi_j = \lambda_j \phi_j$$

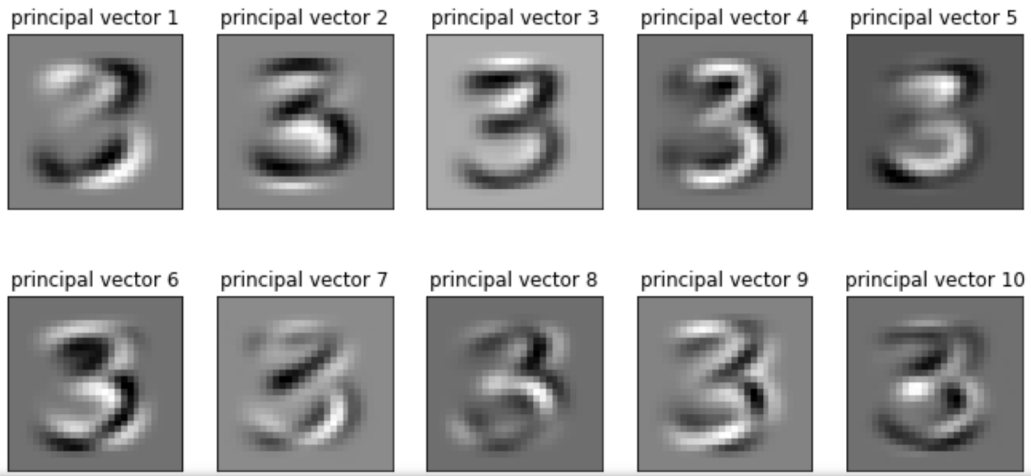
4. Project the data onto those k vectors:

$$x_i \mapsto \overline{x} + (\phi_1^T x_i)\phi_1 + \ldots + (\phi_k^T x_i)\phi_k$$

Let’s revisit the notebook

```
pca = PCA(num_components).fit(cimages)
principal_vectors = pca.components_
principal_vectors = principal_vectors.reshape((num_components, height, width))
pcs = pca.fit_transform(cimages)
capprox = pca.inverse_transform(pcs)
labels = ['principal vector %d' % (i+1) for i in np.arange(num_components)]
plot_images(principal_vectors, labels, height, width, int(num_components/5.), 5)
ratio = pca.explained_variance_ratio_.sum()
print('Variance explained by first %d principal vectors: %.2f%%' % (num_components, ratio*100))
```

```
Variance explained by first 25 principal vectors: 72.46%
```



Dimension reduction—another demo

Flower Power: PCA and classification (30 points)

In this problem you will carry out principal components analysis and classification on the iris data. The task will be to reduce the dimension from four to two using PCA, and then to train logistic regression models on the projected data.

+ Code + Text

```
[ ]  import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split
     from sklearn.decomposition import PCA
     from sklearn.linear_model import LogisticRegression

     import matplotlib.pyplot as plt
     %matplotlib inline
```

PCA: Summary

- PCA is an unsupervised method
- Finds directions of greatest variation in the data
- The directions are called the *principal vectors*; the weightings on the vectors are called the *principal components*
- The first few vectors may be interpretable
- Orthogonality makes interpretation difficult for the higher components
- Can be used for visualization or dimensionality reduction