Informative projections

Kaggle results so far!

#	Team Name	Notebook	Team Members	Score	Entries	Last
1	**Devanshu Desai**			0.99280	3	38m
2	**Arman Ashrafian**			0.98880	3	10h
3	**Kennen**			0.97760	4	12h
4	**H2OTONY**			0.09420	1	4d

Please fill out CAPE evals!
Should be in your inbox.
Due Sun Dec 8

Dimensionality reduction

Why reduce the number of features in a data set?

1. It reduces storage and computation time.
2. High-dimensional data often has a lot of redundancy.
3. Remove noisy or irrelevant features.

Dimensionality reduction

Why reduce the number of features in a data set?

1. It reduces storage and computation time.
2. High-dimensional data often has a lot of redundancy.
3. Remove noisy or irrelevant features.

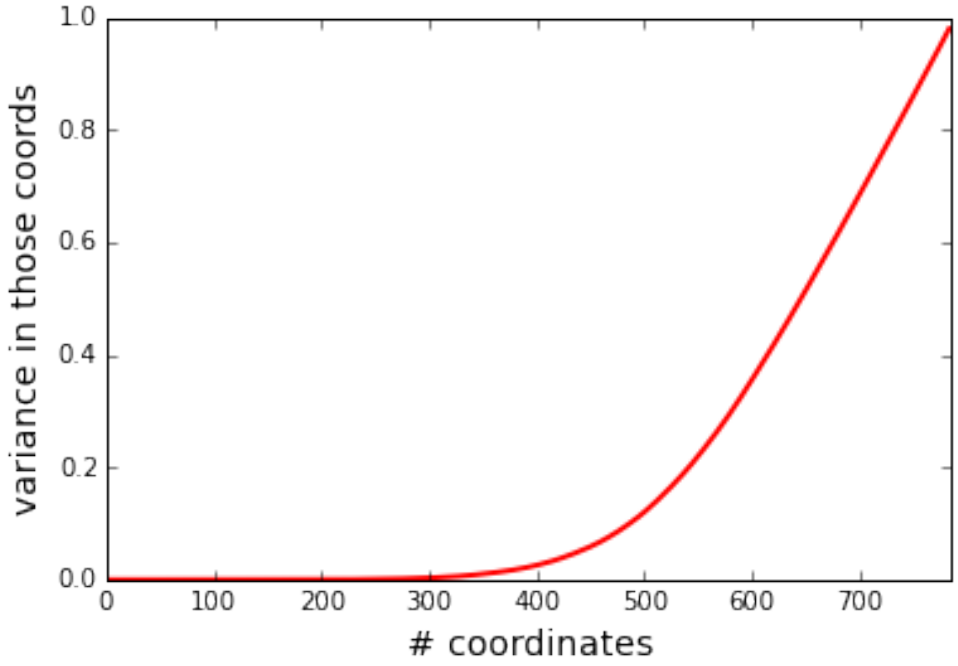
Example: are all the pixels in an image equally informative?



If we were to choose a few pixels to discard, which would be the prime candidates?

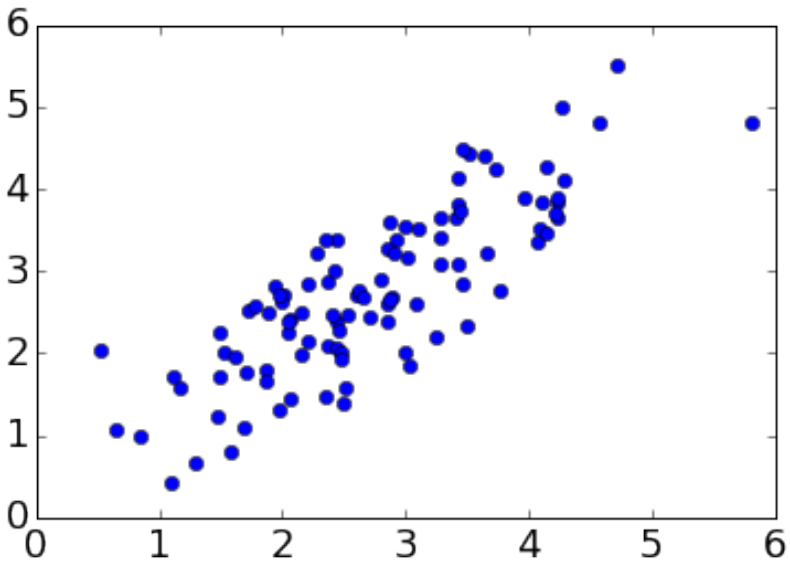
Eliminating low variance coordinates

MNIST: what fraction of the total variance lies in the 100 (or 200, or 300) coordinates with lowest variance?



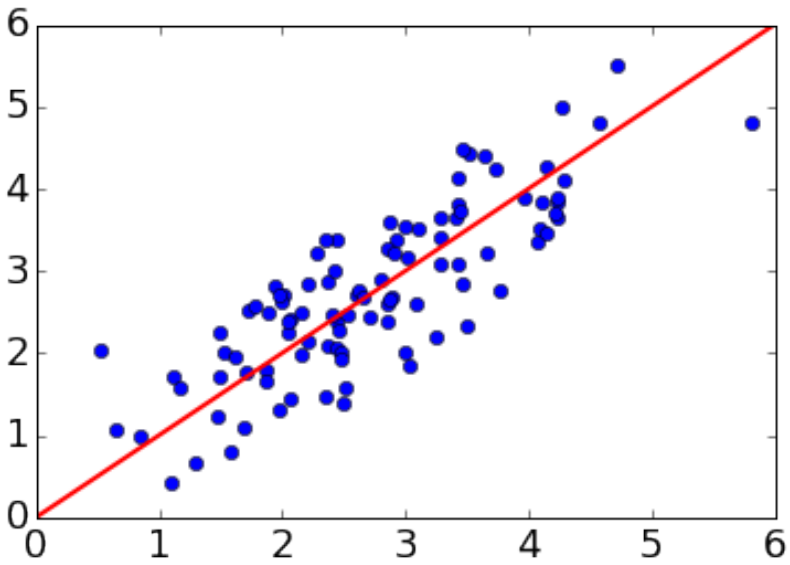
The effect of correlation

Suppose we wanted just one feature for the following data.



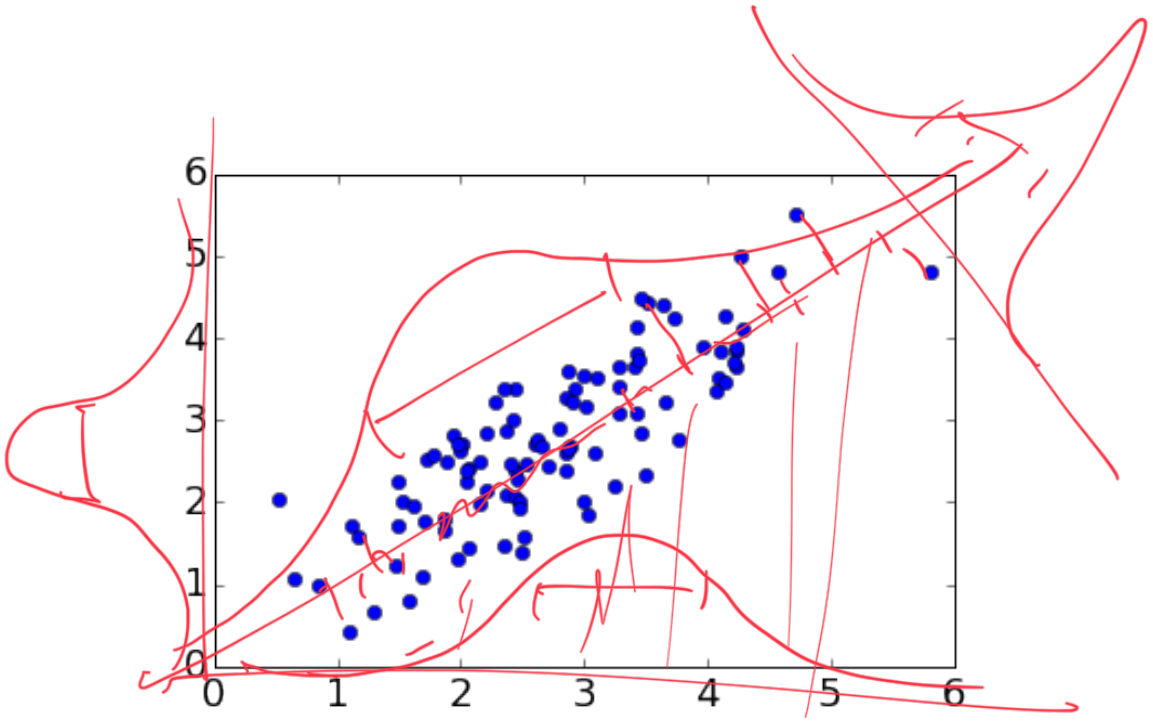
The effect of correlation

Suppose we wanted just one feature for the following data.



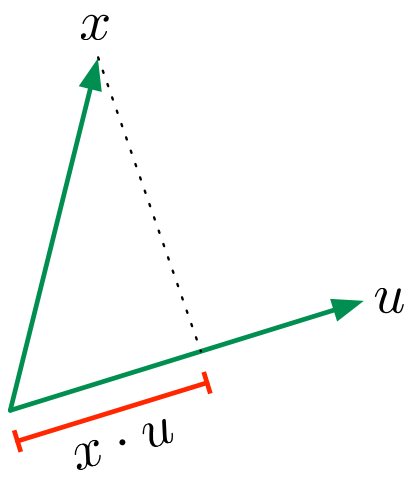
This is the **direction of maximum variance**.

Comparing projections



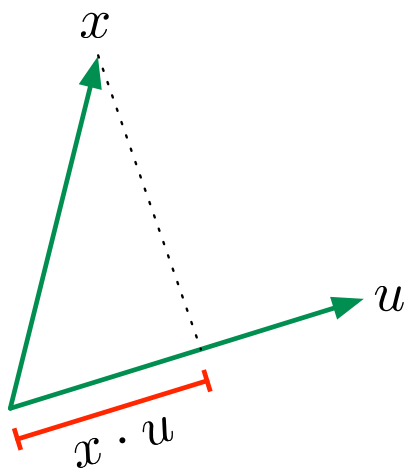
Projection: formally

What is the projection of $x \in \mathbb{R}^d$ in the **direction** $u \in \mathbb{R}^d$?
Assume u is a unit vector (i.e. $\|u\| = 1$).



Projection: formally

What is the projection of $x \in \mathbb{R}^d$ in the **direction** $u \in \mathbb{R}^d$?
Assume u is a unit vector (i.e. $\|u\| = 1$).



Projection is

$$x \cdot u = u \cdot x = u^T x = \sum_{i=1}^{d} u_i x_i.$$

Examples

What is the projection of $x = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$ along the following directions?

1. The x_1-axis?
2. The direction of $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$?

The best direction

Suppose we need to map our data $x \in \mathbb{R}^d$ into just **one** dimension:

$$x \mapsto u \cdot x \quad \text{for some unit direction } u \in \mathbb{R}^d$$

What is the direction u of maximum variance?

The best direction

Suppose we need to map our data $x \in \mathbb{R}^d$ into just **one** dimension:

$$x \mapsto u \cdot x \quad \text{for some unit direction } u \in \mathbb{R}^d$$

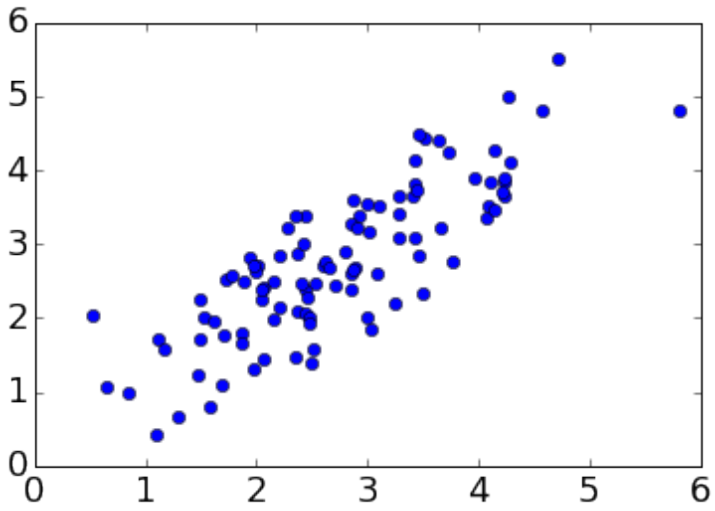
What is the direction u of maximum variance?

Useful fact 1:

- Let Σ be the $d \times d$ covariance matrix of X.
- The variance of X in direction u (the variance of $X \cdot u$) is:

$$u^T \Sigma u.$$

Best direction: example



Here covariance matrix $\Sigma = \begin{pmatrix} 1 & 0.85 \\ 0.85 & 1 \end{pmatrix}$

The best direction

Suppose we need to map our data $x \in \mathbb{R}^d$ into just **one** dimension:

$$x \mapsto u \cdot x \quad \text{for some unit direction } u \in \mathbb{R}^d$$

What is the direction u of maximum variance?

Useful fact 1:

- Let Σ be the $d \times d$ covariance matrix of X.
- The variance of X in direction u is given by $u^T \Sigma u$.

The best direction

Suppose we need to map our data $x \in \mathbb{R}^d$ into just **one** dimension:

$$x \mapsto u \cdot x \quad \text{for some unit direction } u \in \mathbb{R}^d$$

What is the direction u of maximum variance?

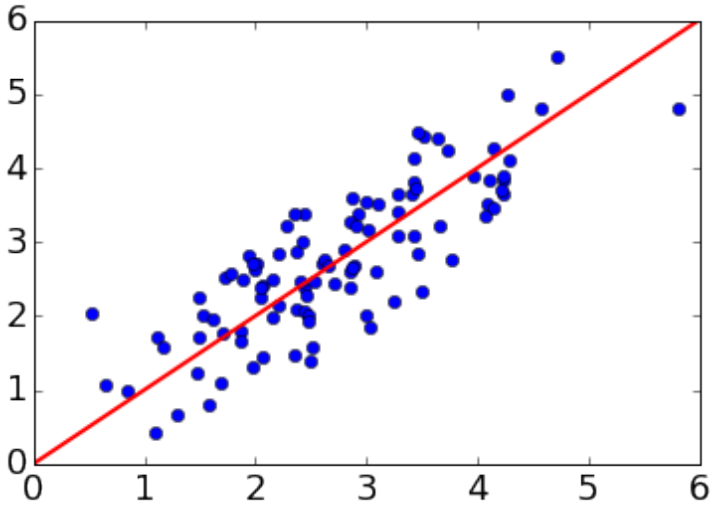
Useful fact 1:

- Let Σ be the $d \times d$ covariance matrix of X.
- The variance of X in direction u is given by $u^T \Sigma u$.

Useful fact 2:

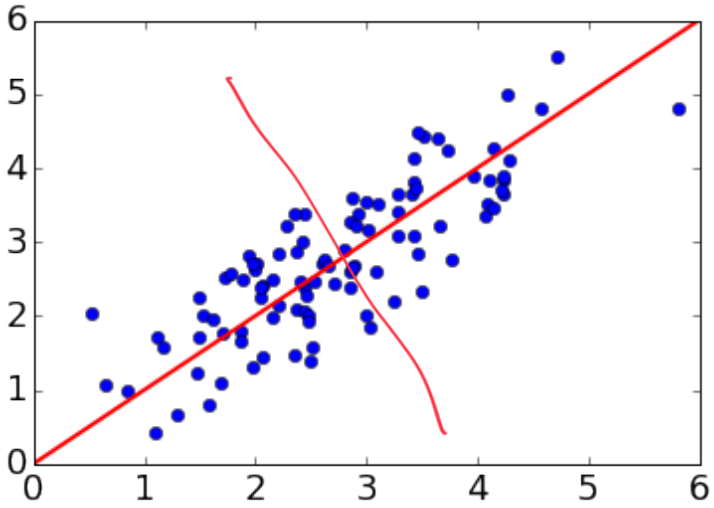
- $u^T \Sigma u$ is maximized by setting u to the first **eigenvector** of Σ.
- The maximum value is the corresponding **eigenvalue**.

Best direction: example

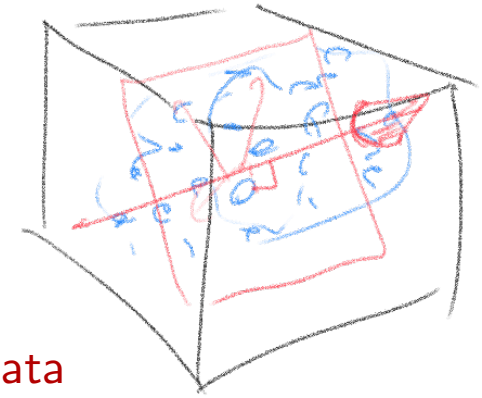


Direction:
first eigenvector of the 2×2 covariance matrix of the data.

Best direction: example



Direction:
first eigenvector of the 2×2 covariance matrix of the data.



Projection onto this direction:
the top **principal component** of the data

Projection onto multiple directions

Projecting $x \in \mathbb{R}^d$ into the k-dimensional subspace defined by vectors $u_1, \ldots, u_k \in \mathbb{R}^d$.

Projection onto multiple directions

Projecting $x \in \mathbb{R}^d$ into the k-dimensional subspace defined by vectors $u_1, \ldots, u_k \in \mathbb{R}^d$.

This is easiest when the u_i's are **orthonormal**:

- They have length one.
- They are at right angles to each other: $u_i \cdot u_j = 0$ when $i \neq j$

Projection onto multiple directions

Projecting $x \in \mathbb{R}^d$ into the k-dimensional subspace defined by vectors $u_1, \ldots, u_k \in \mathbb{R}^d$.

This is easiest when the u_i's are **orthonormal**:

- They have length one.
- They are at right angles to each other: $u_i \cdot u_j = 0$ when $i \neq j$

The projection is a k-dimensional vector:

$$(x \cdot u_1, x \cdot u_2, \ldots, x \cdot u_k) = \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_k & \longrightarrow \end{pmatrix}}_{\text{call this } U^T} \begin{pmatrix} \uparrow \\ x \\ \downarrow \end{pmatrix}$$

U is the $d \times k$ matrix with columns $u_1, \ldots, u_k$.

Projection onto multiple directions: example

E.g. project data in $\mathbb{R}^4$ onto the first two coordinates.

Projection onto multiple directions: example

E.g. project data in $\mathbb{R}^4$ onto the first two coordinates.

Take vectors $u_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $u_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ (notice: orthonormal)

Write $U^T = \begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

Projection onto multiple directions: example

E.g. project data in $\mathbb{R}^4$ onto the first two coordinates.

Take vectors $u_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $u_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ (notice: orthonormal)

Write $U^T = \begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

The projection of $x \in \mathbb{R}^4$ is $U^T x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

The best k-dimensional projection

Let Σ be the $d \times d$ covariance matrix of X.
In $O(d^3)$ time, we can compute its **eigendecomposition**, consisting of

- real **eigenvalues** $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
- corresponding **eigenvectors** $u_1, \ldots, u_d \in \mathbb{R}^d$ that are orthonormal (unit length and at right angles to each other)

The best k-dimensional projection

Let Σ be the $d \times d$ covariance matrix of X.
In $O(d^3)$ time, we can compute its **eigendecomposition**, consisting of

- real **eigenvalues** $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
- corresponding **eigenvectors** $u_1, \ldots, u_d \in \mathbb{R}^d$ that are orthonormal (unit length and at right angles to each other)

Fact: Suppose we want to map data $X \in \mathbb{R}^d$ to just k dimensions, while capturing as much of the variance of X as possible. The best choice of projection is:

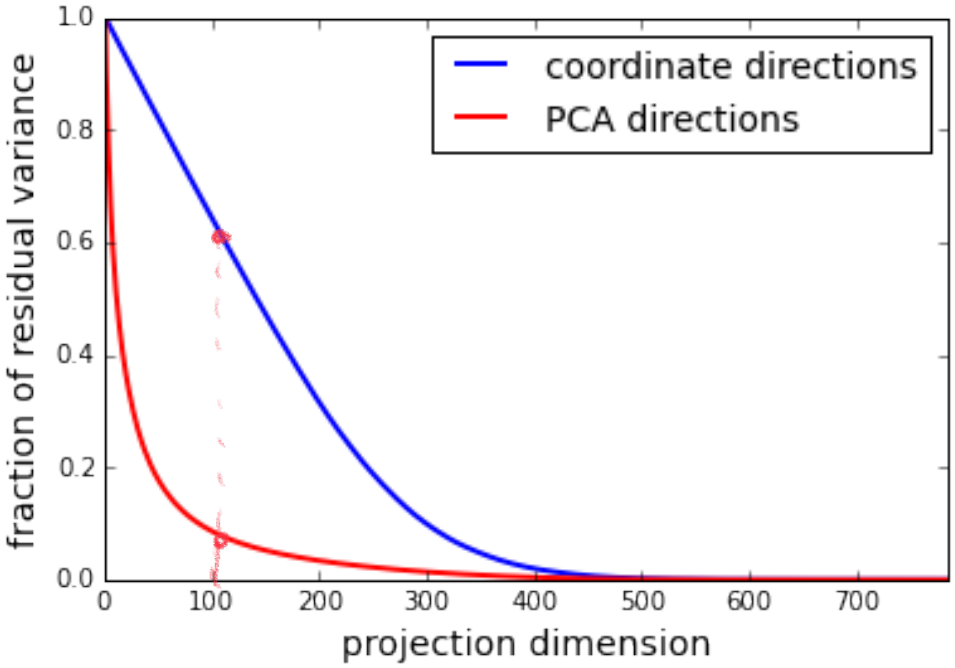
$$x \mapsto (u_1 \cdot x, u_2 \cdot x, \ldots, u_k \cdot x),$$

where u_i are the eigenvectors described above.

This projection is called **principal component analysis** (PCA).

Example: MNIST

Contrast coordinate projections with PCA:



Applying PCA to MNIST: examples



Reconstruct this original image from its PCA projection to k dimensions.

$k = 200$



$k = 150$



$k = 100$



$k = 50$



Applying PCA to MNIST: examples



Reconstruct this original image from its PCA projection to k dimensions.

$k = 200$



$k = 150$



$k = 100$

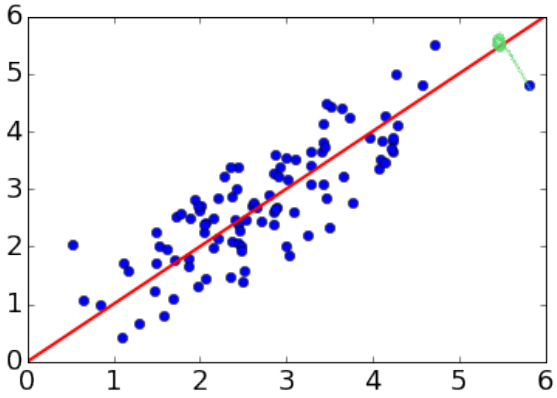


$k = 50$

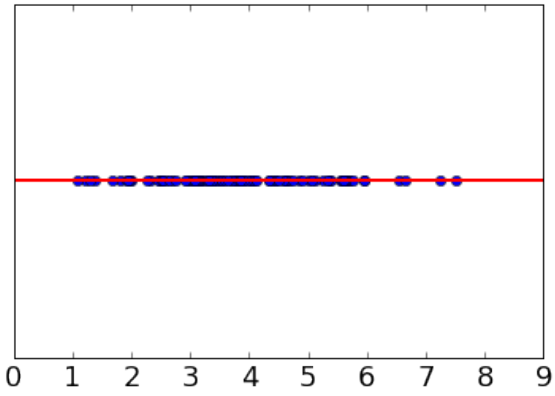


How do we get these **reconstructions**?

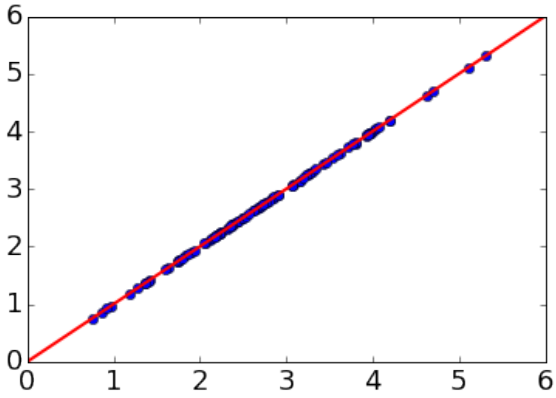
Reconstruction from a 1-d projection



Projection onto $\mathbb{R}$:



Reconstruction in $\mathbb{R}^2$:



Reconstruction from multiple projections

Projecting into the k-dimensional subspace defined by **orthonormal** $u_1, \ldots, u_k \in \mathbb{R}^d$.

The projection of x is a k-dimensional vector:

$$(x \cdot u_1, x \cdot u_2, \ldots, x \cdot u_k) = \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_k & \longrightarrow \end{pmatrix}}_{\text{call this } U^T} \begin{pmatrix} \uparrow \\ x \\ \downarrow \end{pmatrix}$$

Reconstruction from multiple projections

Projecting into the k-dimensional subspace defined by **orthonormal** $u_1, \ldots, u_k \in \mathbb{R}^d$.

The projection of x is a k-dimensional vector:

$$(x \cdot u_1, x \cdot u_2, \ldots, x \cdot u_k) = \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_k & \longrightarrow \end{pmatrix}}_{\text{call this } U^T} \begin{pmatrix} \uparrow \\ x \\ \downarrow \end{pmatrix}$$

The reconstruction from this projection is:

$$(x \cdot u_1)u_1 + (x \cdot u_2)u_2 + \cdots + (x \cdot u_k)u_k = UU^T x.$$

MNIST: image reconstruction



Reconstruct this original image x from its PCA projection to k dimensions.

$k = 200$



$k = 150$



$k = 100$



$k = 50$

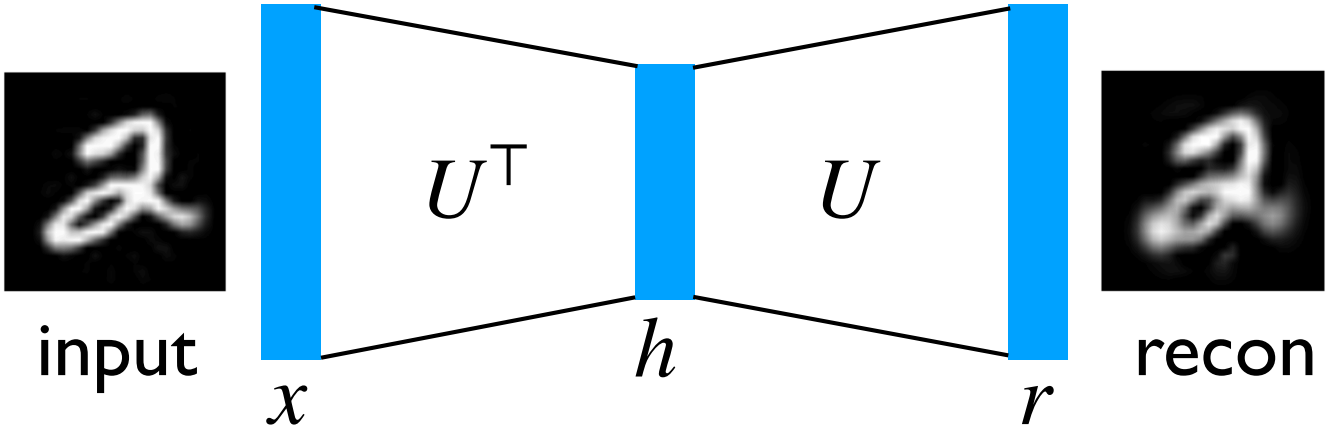


Reconstruction $UU^T x$, where U's columns are top k eigenvectors of Σ.

Autoencoders

Another view of PCA as fully-connected neural net with no activation function:

$$r(x) = UU^\top x$$



Reconstruction error as loss function?

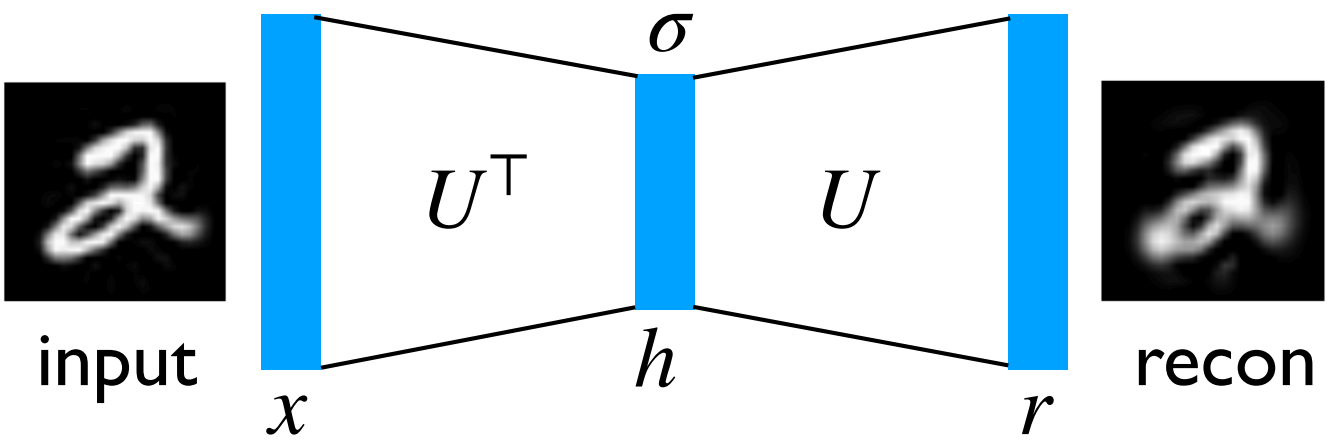
$$L(U) = \|x - r(x)\|_2^2 = \|x - UU^\top x\|_2^2$$

Regression where linear output layer (no softmax) is prediction!
And x is both your input and your ground-truth label!

Autoencoders

What if we add a non-linearity to hidden layer?

$$r(x) = U\sigma(U^\top x)$$

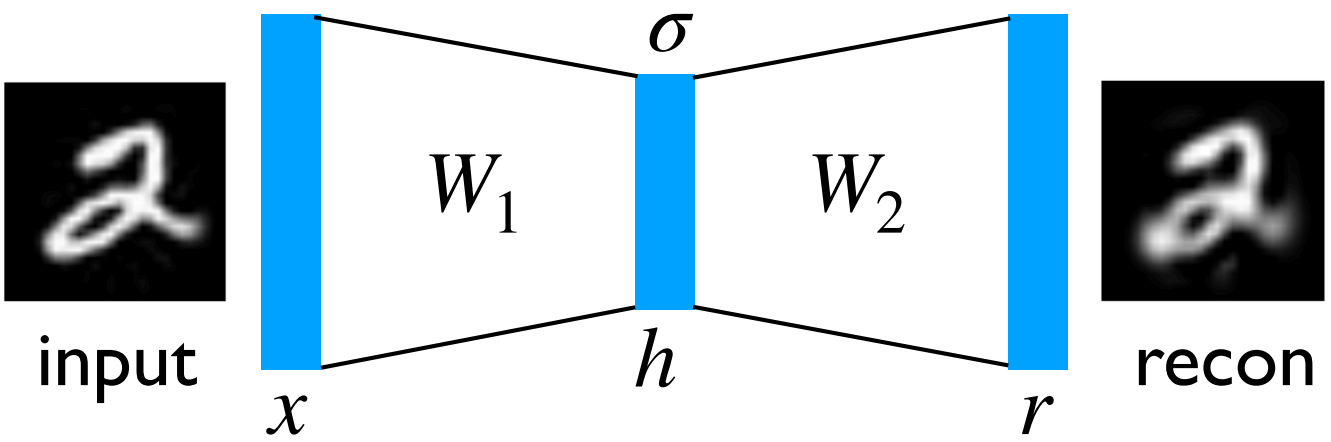


$$L(U) = \|x - r(x)\|_2^2 = \|x - U\sigma(U^\top x)\|_2^2$$

Autoencoders

What if we let the weights for each layer be different?

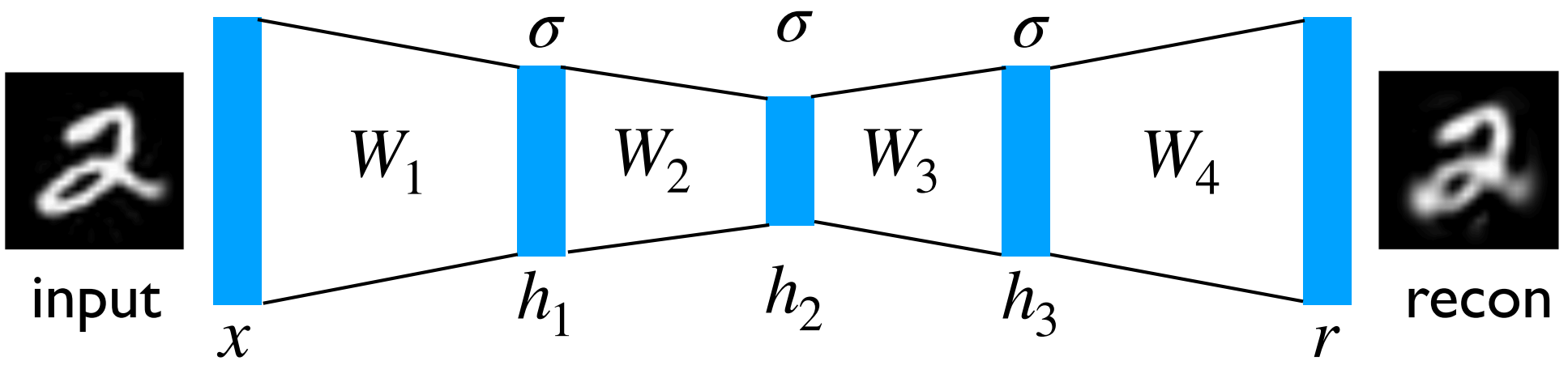
$$r(x) = W_2\sigma(W_1 x)$$



$$L(W_1, W_2) = \|x - r(x)\|_2^2 = \|x - W_2\sigma(W_1 x)\|_2^2$$

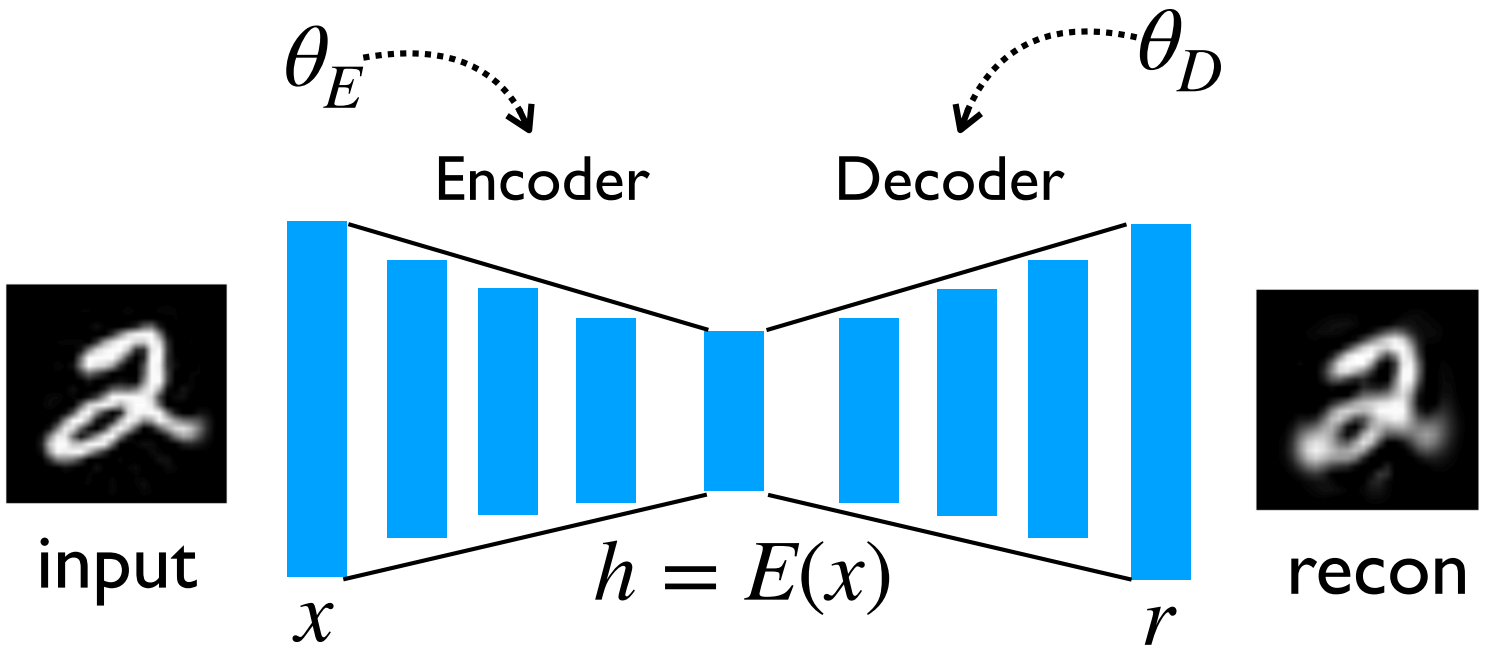
Autoencoders

More layers?



Autoencoders

More generally…



Middle layer is the
thing we care about!

$$L(\theta_E, \theta_D) = \|x - r(x)\|_2^2 = \|x - D_{\theta_D}(E_{\theta_E}(x))\|_2^2$$

Autoencoders

Can work with…

Different architecture for encoder and decoder
Convolutional encoders!
Different loss functions (e.g. binary log. regr. for each pixel)
Very deep architectures
Many many options…

What do you do with it?

Use it when you only have x in large part of train set (i.e. *unlabeled*)
Learn a better data representation to use as input to classifier
Think of it like pre-processing your data…
…can simply replace x with h when training supervised later on

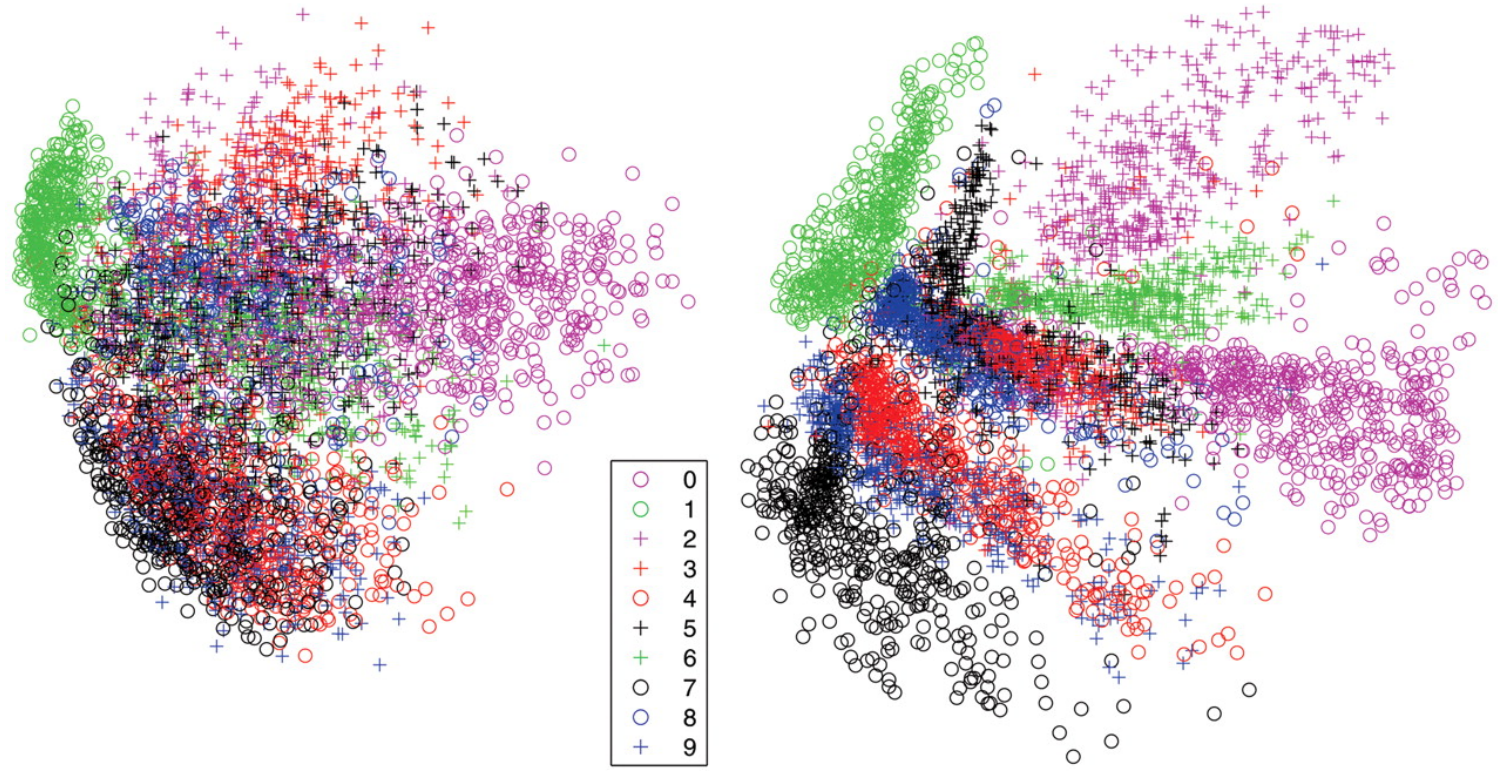
AE vs PCA on MNIST

Try it on your MNIST data!

Which representation looks easier to classify?

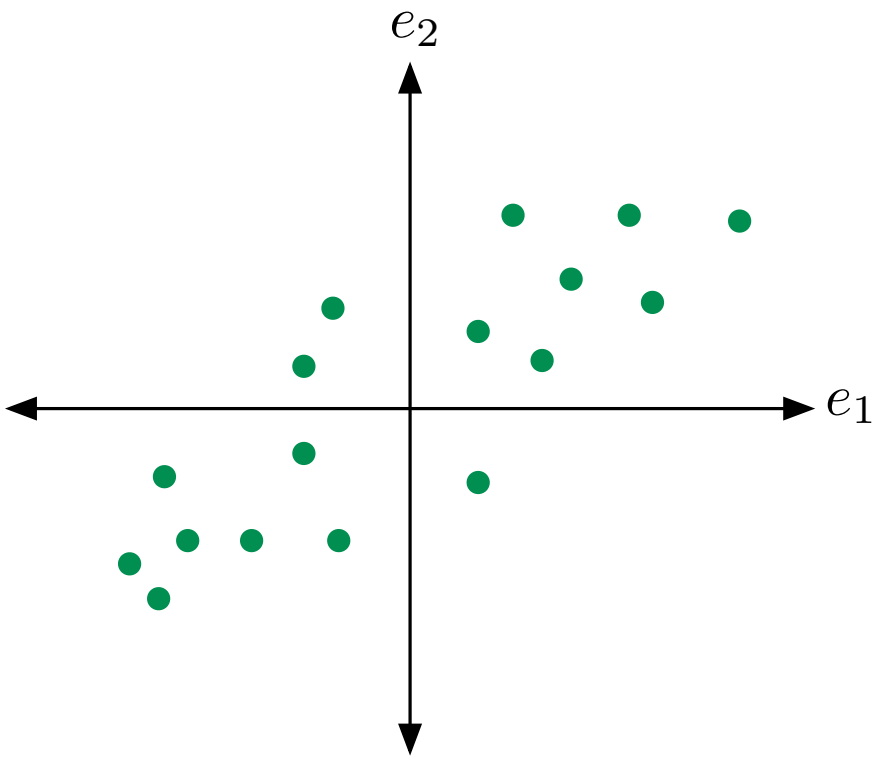
First 2 components PCA

Autoencoder 2D hidden layer

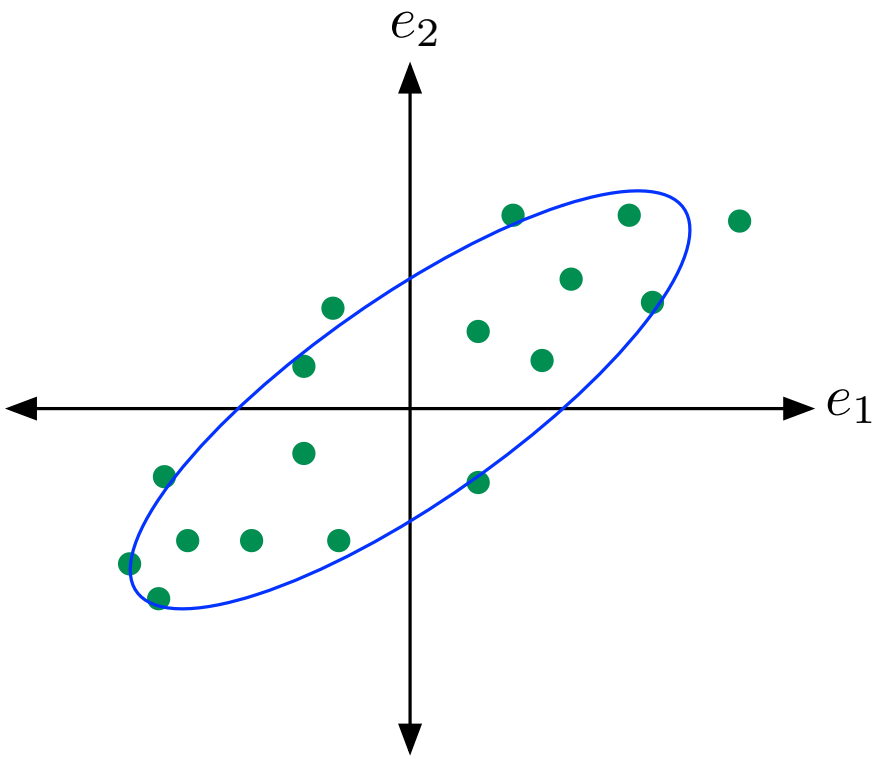


Hinton and Salakhutdinov, *Science 2006*

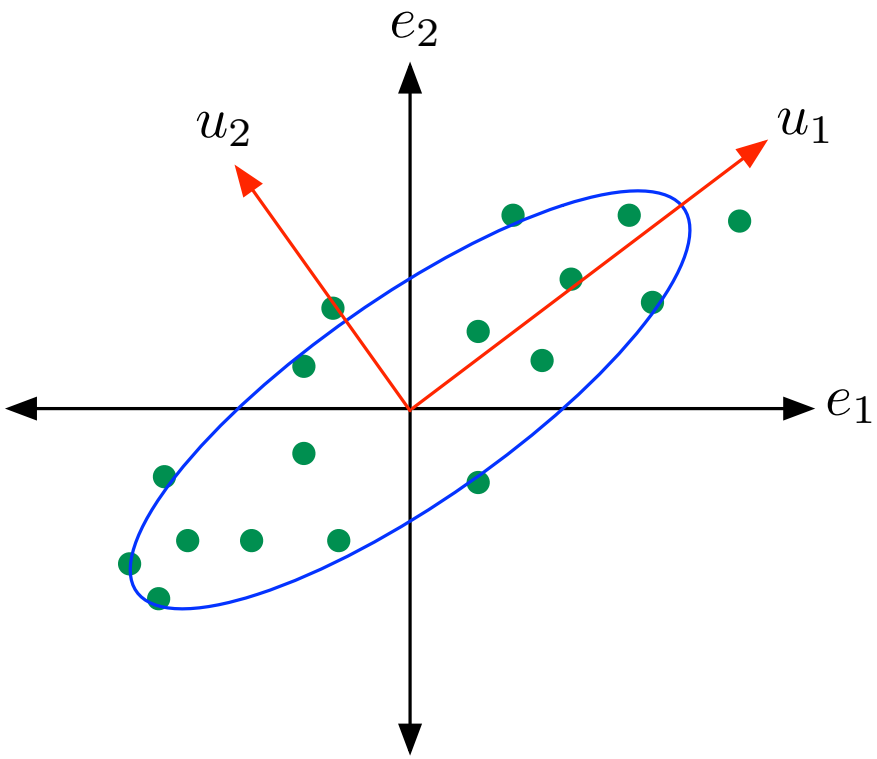
Linear algebra: eigenvalues and eigenvectors



Linear algebra: eigenvalues and eigenvectors



Linear algebra: eigenvalues and eigenvectors



The linear function defined by a matrix

- Any matrix M defines a linear function, $x \mapsto Mx$.
  If M is a $d \times d$ matrix, this maps $\mathbb{R}^d$ to $\mathbb{R}^d$.

The linear function defined by a matrix

- Any matrix M defines a linear function, $x \mapsto Mx$.
  If M is a $d \times d$ matrix, this maps $\mathbb{R}^d$ to $\mathbb{R}^d$.

- This function is easy to understand when M is **diagonal**:

$$\underbrace{\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 10 \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 2x_1 \\ -x_2 \\ 10x_3 \end{pmatrix}}_{Mx}$$

  In this case, M simply scales each coordinate separately.

The linear function defined by a matrix

- Any matrix M defines a linear function, $x \mapsto Mx$.
  If M is a $d \times d$ matrix, this maps $\mathbb{R}^d$ to $\mathbb{R}^d$.

- This function is easy to understand when M is **diagonal**:

$$\underbrace{\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 10 \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 2x_1 \\ -x_2 \\ 10x_3 \end{pmatrix}}_{Mx}$$

  In this case, M simply scales each coordinate separately.

- General symmetric matrices also just scale coordinates separately... but in a **different coordinate system!**

Eigenvector and eigenvalue: definition

Let M be a $d \times d$ matrix. We say $u \in \mathbb{R}^d$ is an **eigenvector** of M if

$$Mu = \lambda u$$

for some scaling constant λ. This λ is the **eigenvalue** associated with u.

Key point: M **maps eigenvector** u **onto the same direction**.

Question: What are the eigenvectors and eigenvalues of:

$$M = \begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 10 \end{pmatrix} \ ?$$

Eigenvectors of a real symmetric matrix

Fact: Let M be any real symmetric $d \times d$ matrix. Then M has

- d eigenvalues $\lambda_1, \ldots, \lambda_d$
- corresponding eigenvectors $u_1, \ldots, u_d \in \mathbb{R}^d$ that are orthonormal

Can think of $u_1, \ldots, u_d$ as the axes of the natural coordinate system for M.

Example

$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix} \text{ has eigenvectors } u_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \; u_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

1. Are these orthonormal?
2. What are the corresponding eigenvalues?

Spectral decomposition

Fact: Let M be any real symmetric $d \times d$ matrix. Then M has orthonormal eigenvectors $u_1, \ldots, u_d \in \mathbb{R}^d$ and corresponding eigenvalues $\lambda_1, \ldots, \lambda_d$.

Spectral decomposition

Fact: Let M be any real symmetric $d \times d$ matrix. Then M has orthonormal eigenvectors $u_1, \ldots, u_d \in \mathbb{R}^d$ and corresponding eigenvalues $\lambda_1, \ldots, \lambda_d$.

Spectral decomposition: Another way to write M:

$$M = \underbrace{\begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ u_1 & u_2 & \cdots & u_d \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}}_{U\text{: columns are eigenvectors}} \underbrace{\begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_d \end{pmatrix}}_{\Lambda\text{: eigenvalues on diagonal}} \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_d & \longrightarrow \end{pmatrix}}_{U^T}$$

Spectral decomposition

Fact: Let M be any real symmetric $d \times d$ matrix. Then M has orthonormal eigenvectors $u_1, \ldots, u_d \in \mathbb{R}^d$ and corresponding eigenvalues $\lambda_1, \ldots, \lambda_d$.

Spectral decomposition: Another way to write M:

$$M = \underbrace{\begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ u_1 & u_2 & \cdots & u_d \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}}_{U\text{: columns are eigenvectors}} \underbrace{\begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_d \end{pmatrix}}_{\Lambda\text{: eigenvalues on diagonal}} \underbrace{\begin{pmatrix} \leftarrow & u_1 & \rightarrow \\ \leftarrow & u_2 & \rightarrow \\ & \vdots & \\ \leftarrow & u_d & \rightarrow \end{pmatrix}}_{U^T}$$

Thus $Mx = U\Lambda U^T x$:

- U^T rewrites x in the $\{u_i\}$ coordinate system
- Λ is a simple coordinate scaling in that basis
- U sends the scaled vector back into the usual coordinate basis

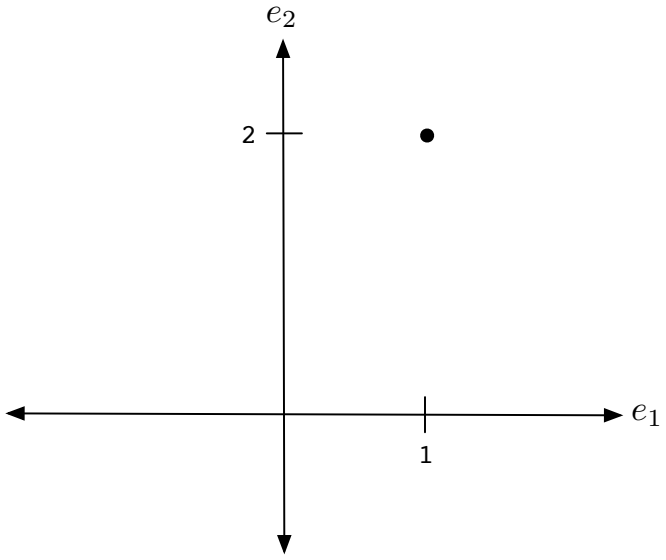
Apply spectral decomposition to the matrix we saw earlier:

$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix}$$

- Eigenvectors $u_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $u_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$
- Eigenvalues $\lambda_1 = -1,\ \lambda_2 = 3.$

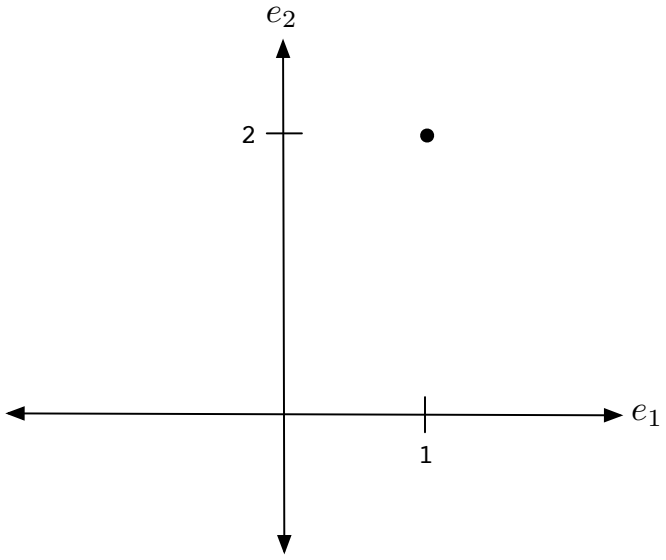
$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix} = \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} -1 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}}_{U^T}$$

$$M \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$



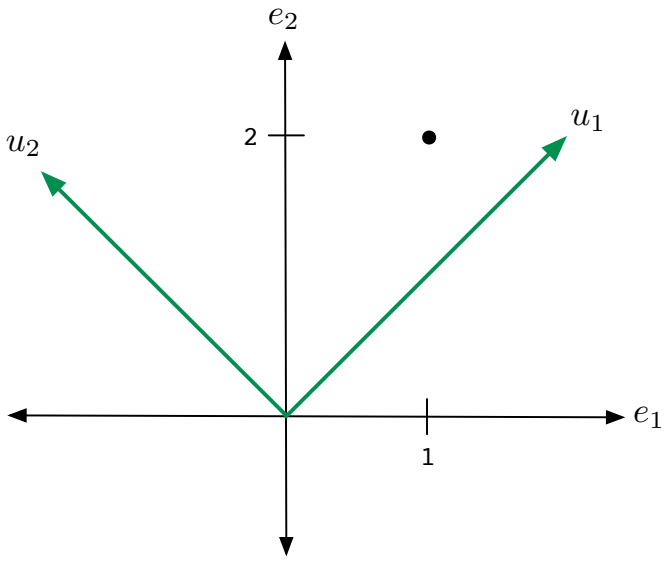
$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix} = \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} -1 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}}_{U^T}$$

$$M \begin{pmatrix} 1 \\ 2 \end{pmatrix} = U \Lambda U^T \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$



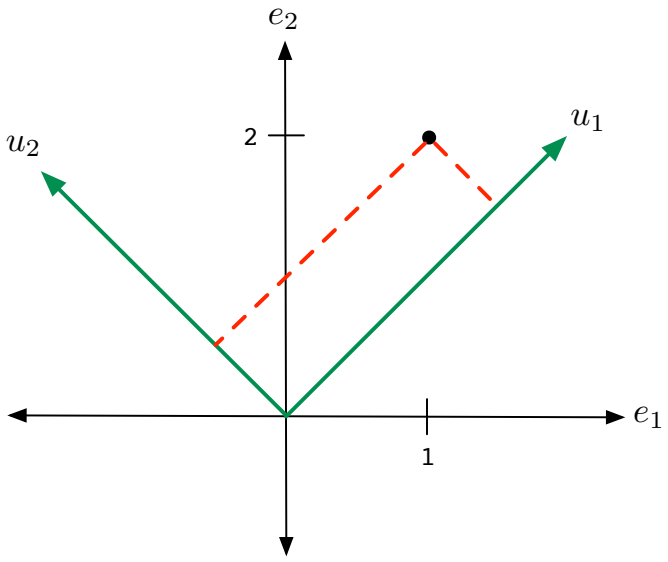
$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix} = \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} -1 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}}_{U^T}$$

$$M \begin{pmatrix} 1 \\ 2 \end{pmatrix} = U \Lambda U^T \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

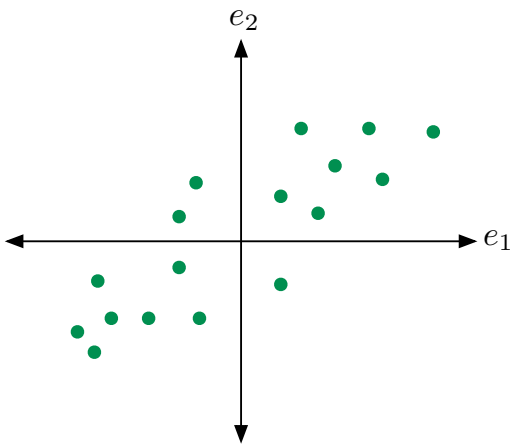


$$M = \begin{pmatrix} 1 & -2 \\ -2 & 1 \end{pmatrix} = \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} -1 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}}_{U^T}$$

$$M \begin{pmatrix} 1 \\ 2 \end{pmatrix} = U \Lambda U^T \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

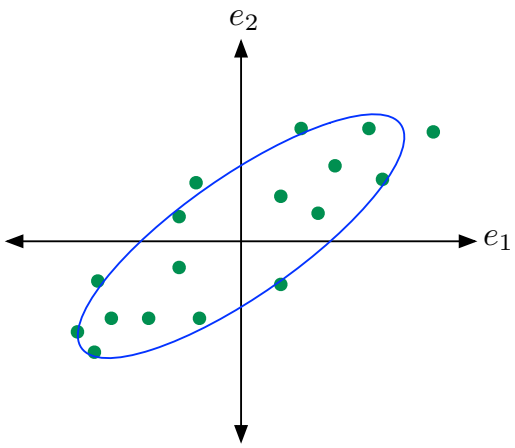


Principal component analysis revisited



Data vectors $X \in \mathbb{R}^d$

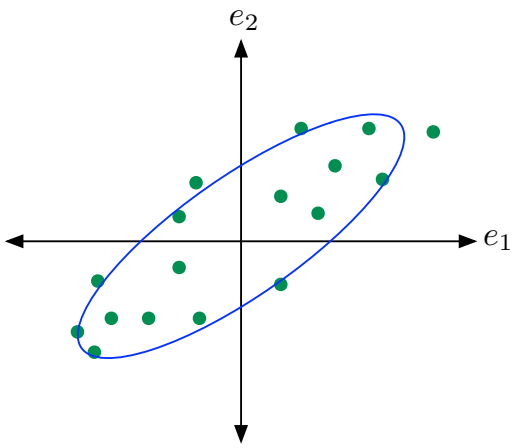
Principal component analysis revisited



Data vectors $X \in \mathbb{R}^d$

- $d \times d$ covariance matrix Σ is symmetric.

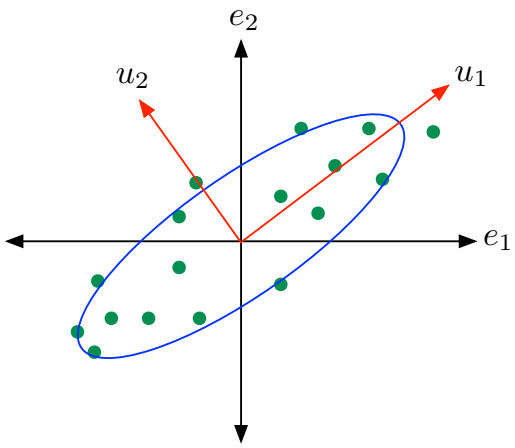
Principal component analysis revisited



Data vectors $X \in \mathbb{R}^d$

- $d \times d$ covariance matrix Σ is symmetric.
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
  Eigenvectors $u_1, \ldots, u_d$.

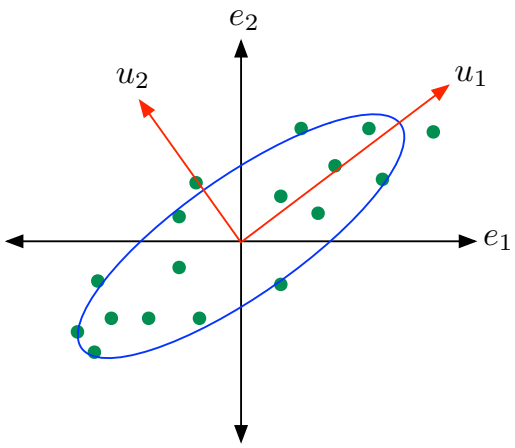
Principal component analysis revisited



Data vectors $X \in \mathbb{R}^d$

- $d \times d$ covariance matrix Σ is symmetric.
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
  Eigenvectors $u_1, \ldots, u_d$.
- $u_1, \ldots, u_d$: another basis for data.
- Variance of X in direction u_i is λ_i.

Principal component analysis revisited



Data vectors $X \in \mathbb{R}^d$

- $d \times d$ covariance matrix Σ is symmetric.
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$
  Eigenvectors $u_1, \ldots, u_d$.
- $u_1, \ldots, u_d$: another basis for data.
- Variance of X in direction u_i is λ_i.
- Projection to k dimensions:
  $x \mapsto (x \cdot u_1, \ldots, x \cdot u_k)$.

What is the covariance of the projected data?