

Visual feature learning

Winter 2013

Roland Memisevic

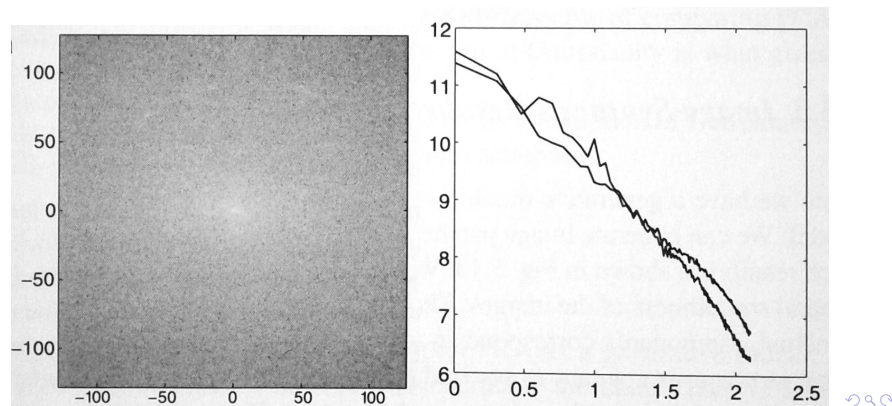
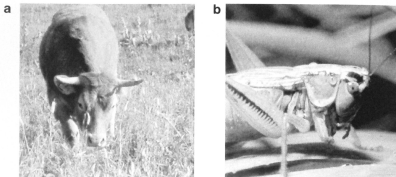
Lecture 5, February 5, 2013



Roland Memisevic

Visual feature learning

More amplitude spectrum examples



Roland Memisevic

Visual feature learning

Last time

- ▶ 1d Fourier Transform
- ▶ 2d waves and 2d Fourier Transform
- ▶ Multiplication – Convolution duality
- ▶ Convolution – Multiplication duality
- ▶ Ringing and filter design
- ▶ Aliasing



Roland Memisevic

Visual feature learning

Amplitude spectra of natural images

- ▶ Amplitude spectra of natural images show strong regularity.
- ▶ They typically take the form

$$\frac{1}{f}$$

where f is the absolute value of frequency.

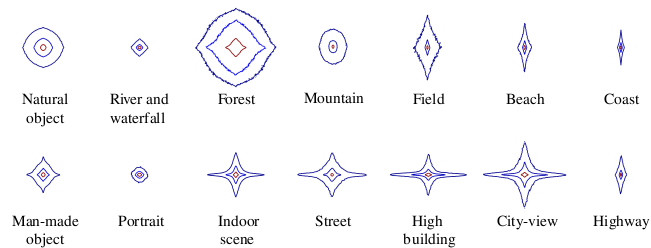
- ▶ Equivalently, the power spectrum $|S(f)|^2$ takes the form $\frac{1}{f^2}$
- ▶ This is known as *power law*.
- ▶ Power law relationships occur frequently in nature and they can be related to scale invariance.
- ▶ They can also be related to the covariance structure of natural images.



Roland Memisevic

Visual feature learning

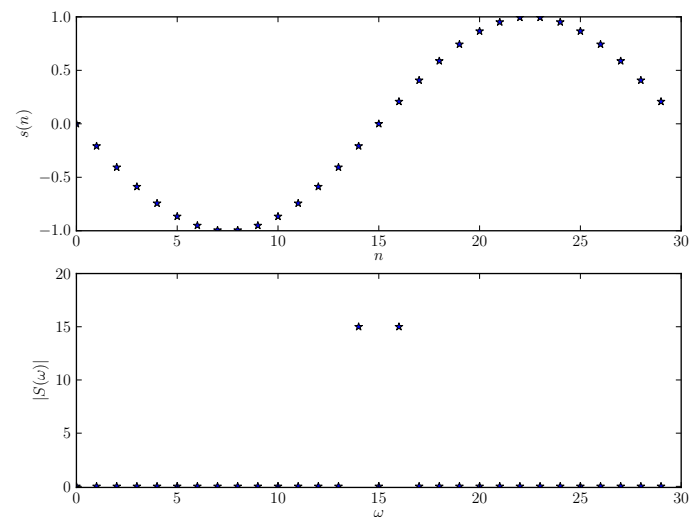
Statistics on image spectra



► Torralba, Oliva; 2003



Leakage example

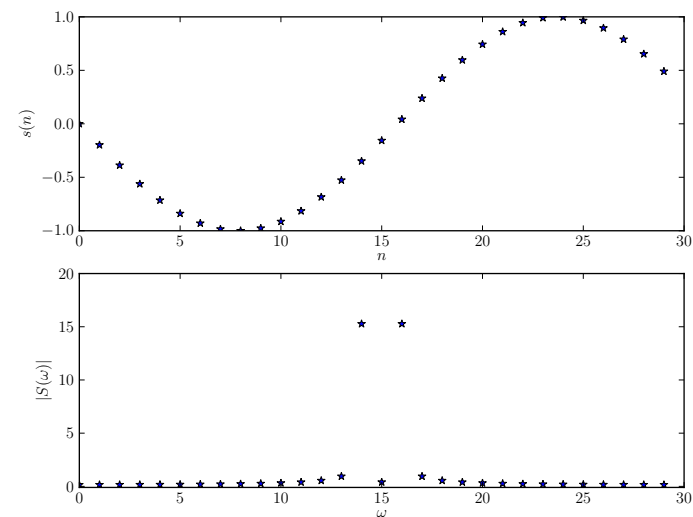


DFT leakage

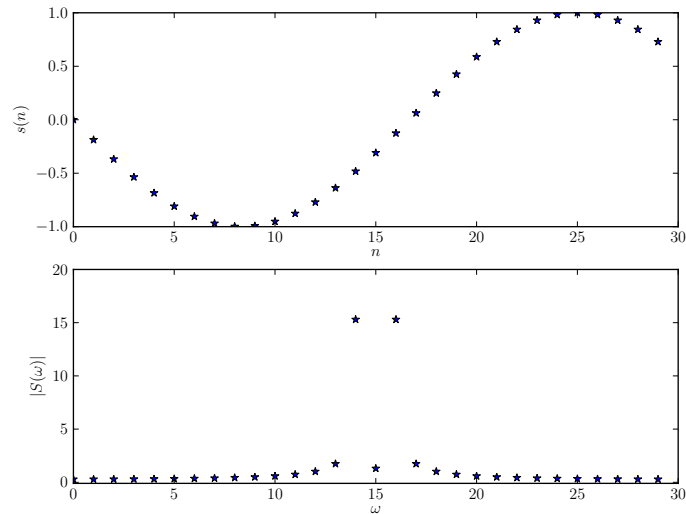
- We can think of the DFT of a finite signal as the DFT on a periodic signal that has been multiplied by a rectangular window.
- The DFT spectrum is therefore the spectrum of the periodic signal, convolved with a sinc-function.
- Because of the zero-crossings of the sinc-function the convolution will have no effect on signals whose frequencies are integer multiples of the window length.
- For any other signal the convolution will generate additional components in the spectrum.
- This effect is known as **leakage**.



Leakage example



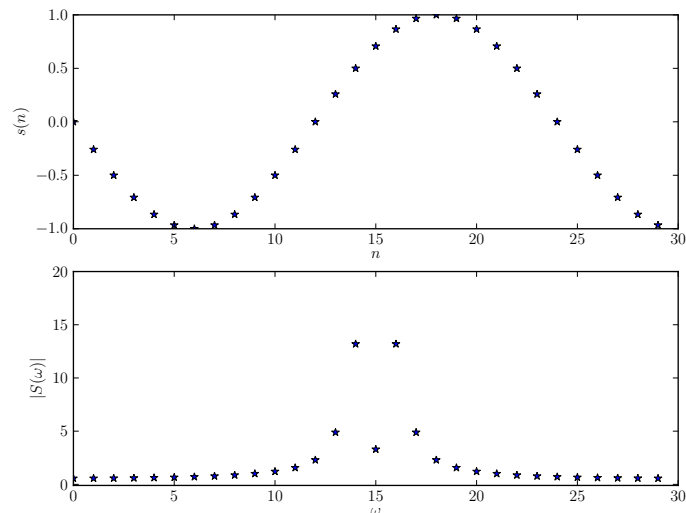
Leakage example



Roland Memisevic

Visual feature learning

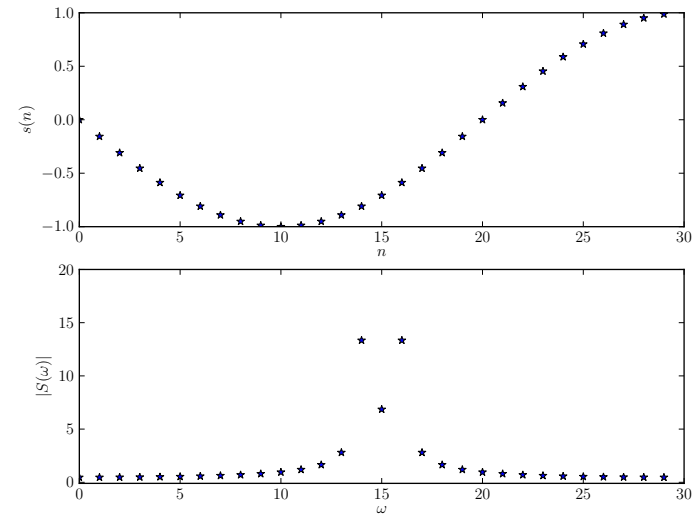
Leakage example



Roland Memisevic

Visual feature learning

Leakage example



Roland Memisevic

Visual feature learning

Windowing

- ▶ The main reason that leakage can be a problem is that other components or noise can leak *into* the frequency bins that one may want to detect.
- ▶ Leakage cannot be avoided, but different types of window will have different leakage-properties.
- ▶ By choosing an appropriate window function, one can re-distribute the leakage effect across bins such that it causes the least harm for the application at hand.

Roland Memisevic

Visual feature learning

Windowing and Short Time Fourier Transform

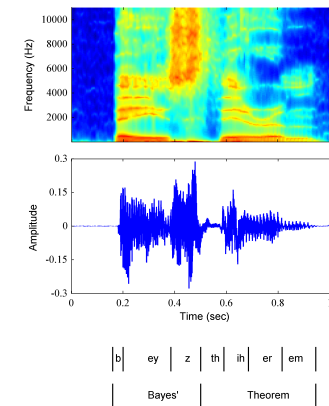
- ▶ The Short-Time Fourier Transform (STFT) is a common application of windowing:
- ▶ Fourier transform a signal *locally*, then view the resulting set of spectra as a function of time/position.
- ▶ In 1d the result is usually called *spectrogram*.
- ▶ An STFT using a Gaussian window is also called *Gabor transform*.



PCA and whitening



A spectrogram (top) of an utterance



- ▶ (Bishop 2006)
- ▶ The image analog of a spectrogram is 3-dimensional.



The “vision equation”

- ▶ The purpose of vision: Infer world properties (or hidden “causes”), \mathbf{s} , from an image, I .
- ▶ We can express this with an *analysis, encoder, inference, or backward* equation:

$$\mathbf{s} = g(I)$$

- ▶ Learning amounts to estimating the parameters of g from data.



Latent variables and generative models

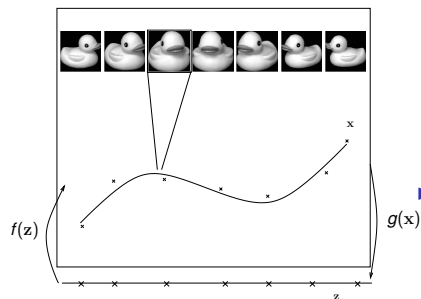
- ▶ In practice, it can be easier to write down how images get formed, *given* the causes.
- ▶ This leads to the *synthesis*, or *decoder*, or *forward* equation:

$$I = f(\mathbf{s})$$

- ▶ It describes how images depend on the state of the world.
- ▶ \mathbf{s} is called “latent variable” or “hidden variable”, because unlike the image, I , we do not observe it.



Manifold learning



- ▶ When the dimensionality of the latent variables is smaller than the dimensionality of the data, then we can think of the data as being distributed along some lower-dimensional *manifold* in the dataspace.
- ▶ Learning the manifold is known as *dimensionality reduction*.
- ▶ We shall use \mathbf{x} in the following to denote data and \mathbf{z} to denote the latent variables.



Latent variables and generative models

- ▶ To incorporate ambiguities and uncertainties, we can re-phrase this equation as a *conditional probability*:

$$I \sim p(I|\mathbf{s})$$

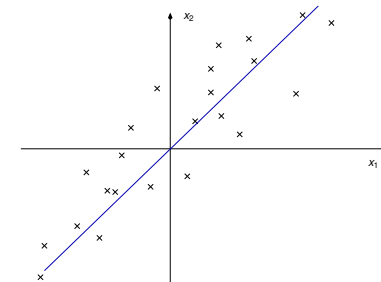
- ▶ Analysis then follows from Bayes' rule:

$$p(\mathbf{s}|I) = \frac{p(I|\mathbf{s})p(\mathbf{s})}{p(I)}$$

- ▶ Analysis then requires a *prior*, $p(\mathbf{s})$, over the latent variables.



Principal Components Analysis (PCA)



- ▶ If we assume the manifold to be *linear*, learning is simple and it can be done in closed form.
- ▶ Because the manifold is just a lower-dimensional *subspace*.
- ▶ Learning amounts to finding the optimal subspace. Inference amounts to projecting data into the subspace.

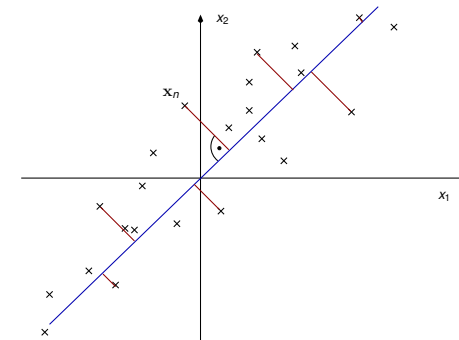


Principal Components Analysis

- ▶ Learning the linear manifold is known as Principal Components Analysis (PCA).
- ▶ There are a lot of equivalent learning criteria leading to PCA.
- ▶ Two of the most well-known are
 1. find the subspace in which the projection of the training data has *maximal variance*
 2. maximize the *average distance* between the projections and the original points.



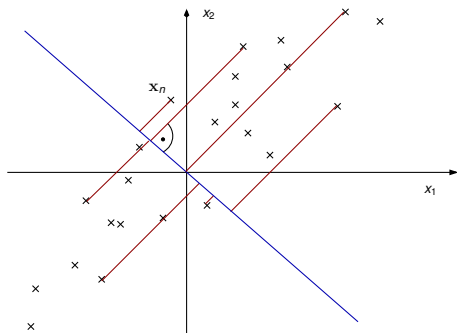
Principal Components Analysis



- ▶ The variance along the manifold is **large**.
- ▶ The average projection error is **small**.



Principal Components Analysis



- ▶ The variance along the manifold is **small**.
- ▶ The average projection error is **large**.



Principal Components Analysis

- ▶ To learn a *subspace* means we need to work under the assumption that the data is *mean-centered*:

$$\frac{1}{N} \sum_{n=1}^N \mathbf{x}_n = \mathbf{0}$$

- ▶ To derive PCA, we define an *orthonormal basis* for the lower-dimensional subspace, consisting of vectors

$$\mathbf{u}_1, \dots, \mathbf{u}_M$$

where M is smaller than the dimensionality of the data.

- ▶ PCA amounts to *learning* this basis.



Principal Components Analysis

- ▶ It is convenient to stack the basis-vectors column-wise in matrix \mathbf{U} .
- ▶ Assuming we had already learned the optimal basis, we can write the forward and backward mappings as:

Projecting data (backward mapping)

- ▶ The optimal coefficients that approximate \mathbf{x} within the subspace are given by

$$\mathbf{z} = \mathbf{U}^T \mathbf{x}$$

Reconstructing data (forward mapping)

- ▶ The approximation $\tilde{\mathbf{x}}$ of \mathbf{x} is given by

$$\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} = \mathbf{U}\mathbf{U}^T \mathbf{x}$$

Principal Components Analysis

Optimizing quadratic forms

- ▶ The maximizer of

$$\text{Tr}(\mathbf{U}^T \mathbf{A} \mathbf{U})$$

subject to

$$\mathbf{U}^T \mathbf{U} = \mathbf{I}$$

(where \mathbf{U} is $D \times M$) is given by the matrix whose columns are the eigenvectors of \mathbf{A} corresponding to the M largest eigenvalues.

- ▶ So to find principal components perform an **eigen-decomposition of the data covariance matrix**.

Principal Components Analysis

- ▶ One way to learn the subspace: minimize reconstruction error

$$E(\mathbf{U}) = \sum_n \|\mathbf{x}_n - \mathbf{U}\mathbf{U}^T \mathbf{x}_n\|^2$$

under the constraint $\mathbf{U}^T \mathbf{U} = \mathbf{I}$

- ▶ To solve the problem, we stack the data row-wise in matrix \mathbf{X} and rewrite the objective function as a *quadratic form* in \mathbf{U} :

$$\begin{aligned} E(\mathbf{U}) &= \|\mathbf{X}^T - \mathbf{U}\mathbf{U}^T \mathbf{X}^T\|_F^2 \\ &= \text{Tr}((\mathbf{X}^T - \mathbf{U}\mathbf{U}^T \mathbf{X}^T)^T (\mathbf{X}^T - \mathbf{U}\mathbf{U}^T \mathbf{X}^T)) \\ &= \text{Tr}(\mathbf{X}\mathbf{X}^T) - \text{Tr}(\mathbf{U}^T \mathbf{X}^T \mathbf{X} \mathbf{U}) \\ &= -\text{Tr}(\mathbf{U}^T \mathbf{X}^T \mathbf{X} \mathbf{U}) + \text{const} \end{aligned}$$

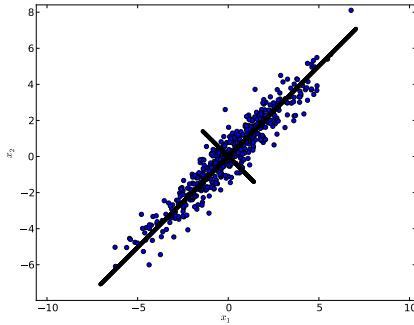
- ▶ Optimizing a quadratic form under an orthonormality constraint is a common exercise in linear algebra:

Principal Components Analysis

Summary: Computing principal components

1. Mean-center the data.
2. Compute the covariance matrix $\mathbf{C} = \frac{1}{N} \mathbf{X}^T \mathbf{X}$.
3. Perform an eigen-decomposition of \mathbf{C} .
4. Sort the eigen-vectors according to the size of their eigenvalues.
5. Stack the leading M eigen-vectors in matrix \mathbf{U} .

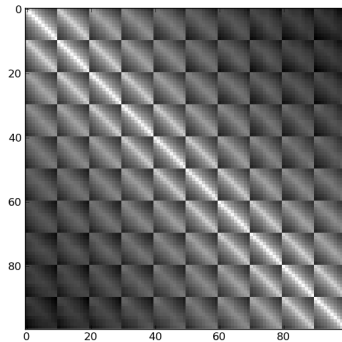
Principal Components Analysis



- ▶ A two-dimensional dataset and the two principal components.
- ▶ Projections onto the leading eigenvectors preserve most of the variability in the data. So PCA performs *lossy compression*.



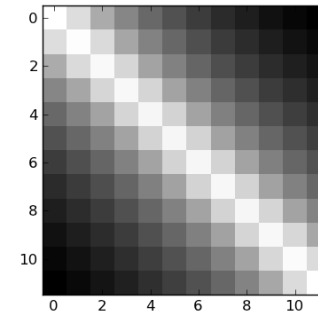
Translation invariance (2d)



- ▶ The covariance between natural image pixels does not depend much on their absolute position.



Translation invariance (1d)



- ▶ The covariance between natural image pixels does not depend much on their absolute position.



PCA and Fourier analysis (1d)

- ▶ A (covariance) matrix whose entries are translation-invariant has phasors as eigenvectors:

$$\begin{aligned}
 & \sum_{t'} \text{cov}(t, t') e^{i\omega t'} \\
 &= \sum_{t'} c(t - t') e^{i\omega t'} \\
 &= \sum_z c(z) e^{i\omega t} e^{-i\omega z} \\
 &= \left[\sum_z c(z) e^{-i\omega z} \right] e^{i\omega t}
 \end{aligned}$$

- ▶ In fact, the covariance defines a convolution.



PCA and Fourier analysis (1d)

- ▶ Covariance matrices are symmetric ($c(z) = c(T - z)$)
- ▶ So the eigenvalues are real:

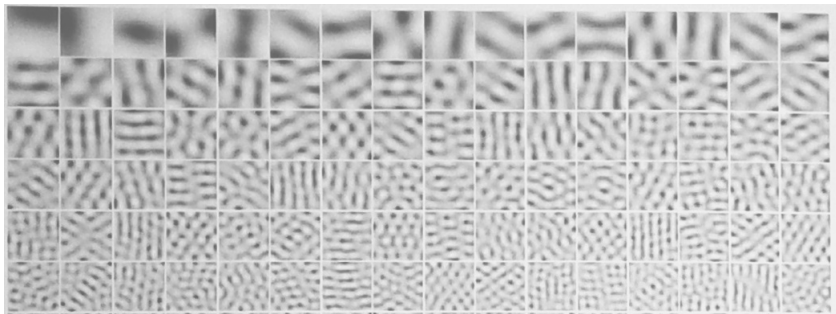
$$\begin{aligned} & \left[\sum_z c(z) e^{i\omega z} \right] e^{i\omega t} \\ &= \left[c(0) + \sum_{z=1}^{\frac{T-1}{2}} c(z) (e^{i\omega z} + e^{-i\omega z}) \right] e^{i\omega t} \\ &= \left[c(0) + 2 \sum_{z=1}^{\frac{T-1}{2}} c(z) \cos(\omega z) \right] e^{i\omega t} \end{aligned}$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Roland Memisevic

Visual feature learning

PCA example (first 96 EVs)



◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Roland Memisevic

Visual feature learning

PCA and Fourier analysis 2d

- ▶ An image covariance matrix whose entries are translation-invariant has 2d waves as eigenvectors:

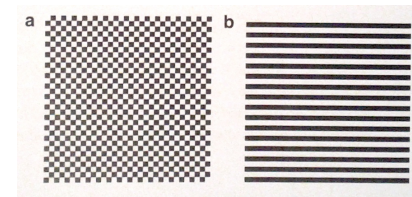
$$\begin{aligned} & \sum_{x',y'} \text{cov}((x, y), (x', y')) e^{i(\omega_1 x' + \omega_2 y')} \\ &= \sum_{x',y'} c((x - x')^2 + (y - y')^2) e^{i(\omega_1 x' + \omega_2 y')} \\ &= \sum_{\xi, \eta} c(\xi, \eta) e^{i(\omega_1 x - \omega_1 \xi + \omega_2 y - \omega_2 \eta)} \\ &= \left[\sum_{\xi, \eta} c(\xi, \eta) e^{-i(\omega_1 \xi + \omega_2 \eta)} \right] e^{i(\omega_1 x + \omega_2 y)} \end{aligned}$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Roland Memisevic

Visual feature learning

Dimensionality reduction and anti-aliasing



- ▶ Thus, PCA on natural images amounts to approximately performing a Fourier decomposition, and dimensionality reduction amounts to low-pass filtering.
- ▶ Low-pass filtering can be a good idea because:
 1. In rectangular images, oblique frequencies are overrepresented as compared to vertical or horizontal frequencies.
 2. Phase becomes meaningless at the highest representable frequencies.

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↻

Roland Memisevic

Visual feature learning

PCA and Whitening

- ▶ The components of the features, \mathbf{Z} , are uncorrelated (that is, \mathbf{Z} has a diagonal covariance matrix):

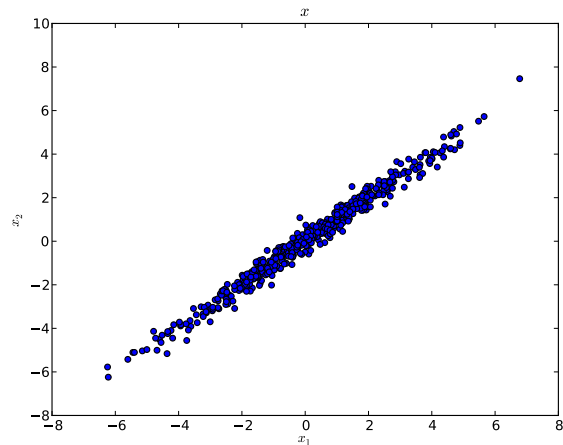
$$\begin{aligned}\frac{1}{N} \sum_n \mathbf{z}_n \mathbf{z}_n^T &= \frac{1}{N} \sum_n \mathbf{U}^T \mathbf{x}_n \mathbf{x}_n^T \mathbf{U} \\ &= \mathbf{U}^T \left(\frac{1}{N} \sum_n \mathbf{x}_n \mathbf{x}_n^T \right) \mathbf{U} \\ &= \mathbf{U}^T \mathbf{C} \mathbf{U} \\ &= \mathbf{L}\end{aligned}$$

where the diagonal matrix \mathbf{L} contains the eigenvalues of \mathbf{C} on its diagonal.

- ▶ (The last step follows from the eigenvalue definition: $\mathbf{C} \mathbf{u}_j = \lambda_j \mathbf{u}_j$)



Whitening example



PCA and Whitening

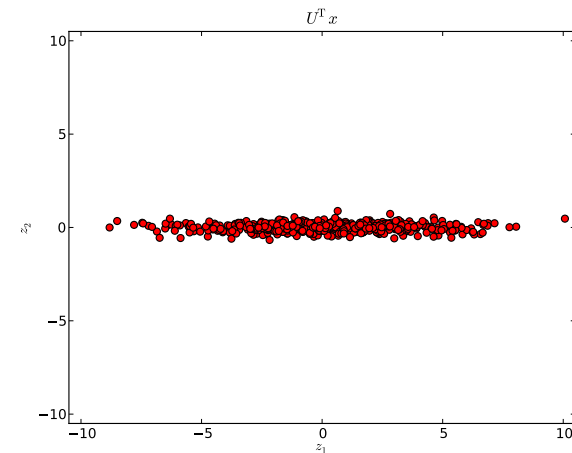
- ▶ We can obtain the *identity* as the covariance matrix for \mathbf{Z} , if we define the forward mapping as

$$\mathbf{V} = \mathbf{L}^{-\frac{1}{2}} \mathbf{U}^T$$

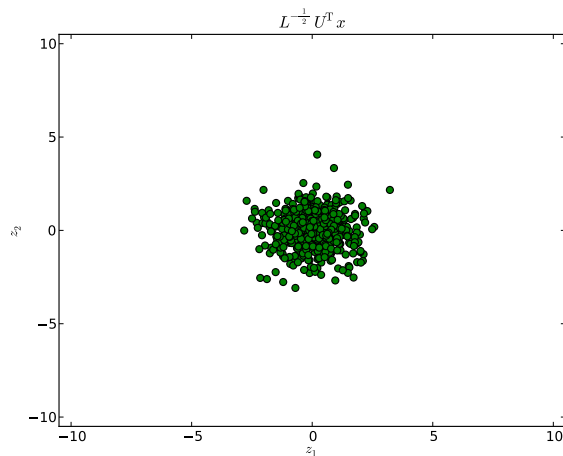
- ▶ Data with identity covariance matrix is known as **white**; multiplying data by \mathbf{V} as **whitening**.
- ▶ Whitening may be performed without reducing the dimensionality.
- ▶ This amounts to just rotating the coordinate system of the data, followed by independently “stretching” or “squeezing” the dimensions to obtain unit variance in each.



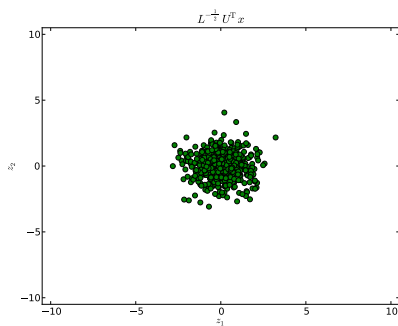
Whitening example



Whitening example



PCA whitened data



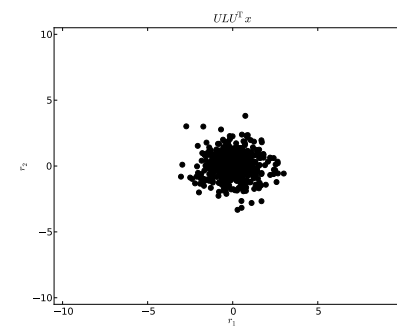
ZCA Whitening

- ▶ Multiplying whitened data with an orthonormal matrix leaves the data white. (exercise)
- ▶ Thus, the whitening matrix $V = L^{-\frac{1}{2}} U^T$ is not the only whitening matrix. Any matrix AV with orthonormal A is, too.
- ▶ One way to define a canonical whitening matrix is to choose the *symmetric* one:

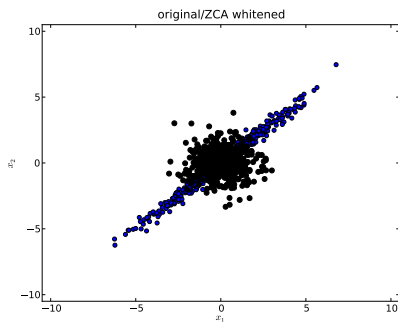
$$W := UV = UL^{-\frac{1}{2}} U^T$$

- ▶ Transforming data with this matrix is known as ZCA (zero-phase components analysis).

ZCA whitened data



ZCA whitened data



De-whitening

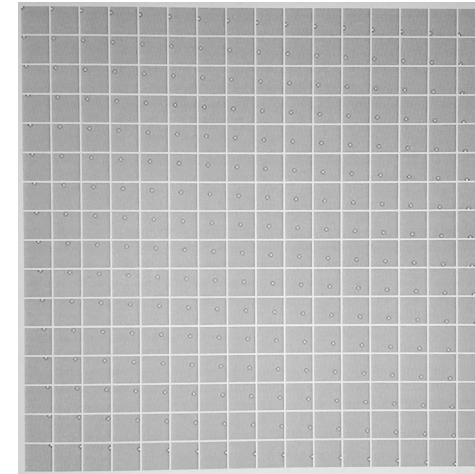
- ▶ To go back from whitened images to images, we need to invert the transform:

$$\mathbf{x} = \mathbf{U}\mathbf{\Lambda}\mathbf{z} = \mathbf{V}^{-1}\mathbf{z} \quad (\text{inverse PCA whitening})$$

$$\mathbf{x} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T\mathbf{z} = \mathbf{W}^{-1}\mathbf{z} \quad (\text{inverse ZCA whitening})$$

where $\mathbf{\Lambda} = \mathbf{L}^{\frac{1}{2}}$, so $\Lambda_{ii} = \sqrt{\lambda_i}$

ZCA example (all columns of \mathbf{W})



Common standardization before whitening

1. DC centering

$$I(x, y) \leftarrow I(x, y) - \frac{1}{MN} \sum_{x', y'} I(x', y')$$

2. Contrast normalization

$$I(x, y) \leftarrow \frac{I(x, y)}{\sqrt{\sum_{x, y} I(x, y)^2 + \epsilon}}$$

Common standardization before whitening

- ▶ This is not the same as:
 1. Centering each pixel.
 2. Setting the standard deviation of each pixel to 1.
- ▶ But one may do this in addition.