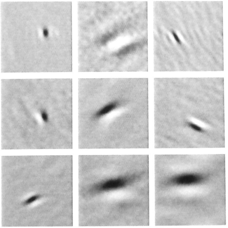
Visual features: From Fourier to Gabor

Roland Memisevic

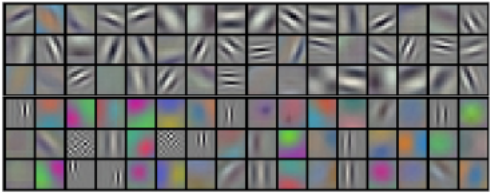
Deep Learning Summer School 2015, Montreal

Hubel and Wiesel, 1959

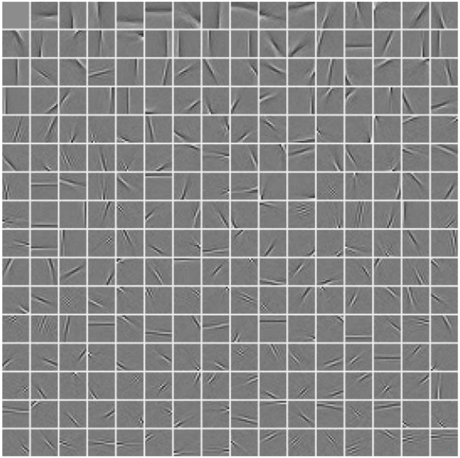


from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

Alexnet

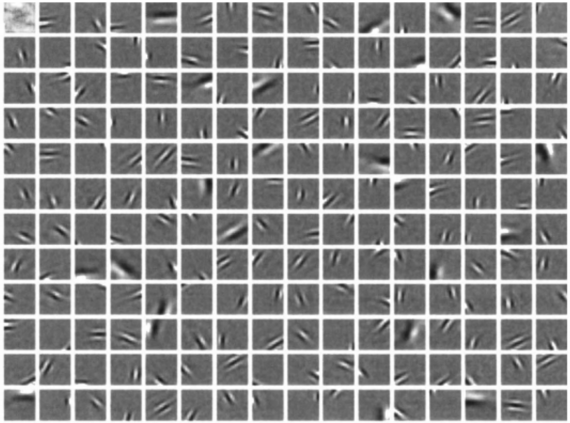


ICA



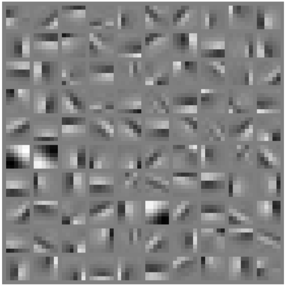
from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

Sparse coding

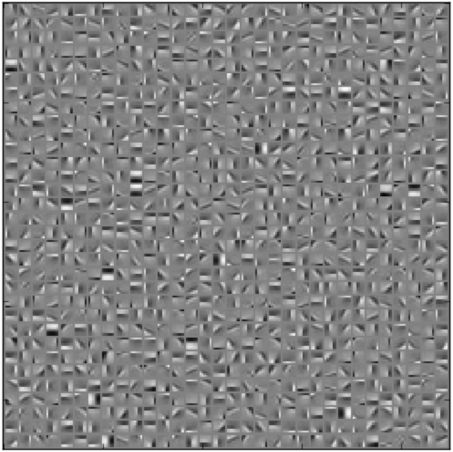


(Olshausen, Field)

Regularized Autoencoder

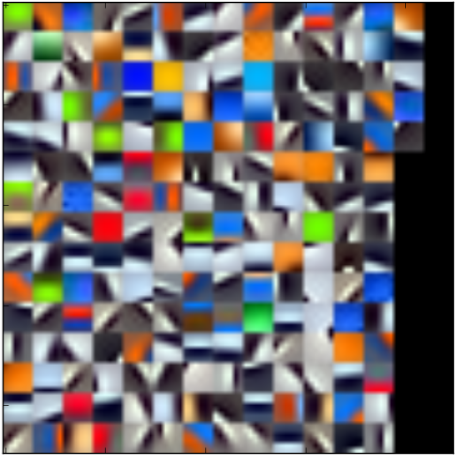


*Un*contractive autoencoder



These are features trained with a contractive autoencoder with *negative* contraction penalty.

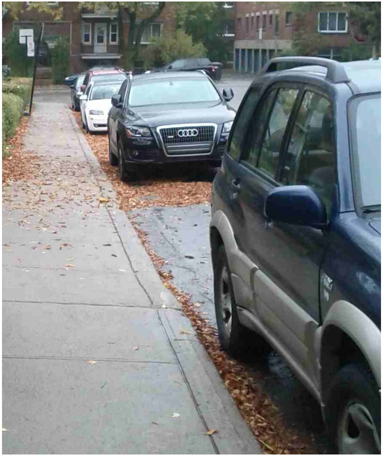
K-means



Fourier (1768-1830)

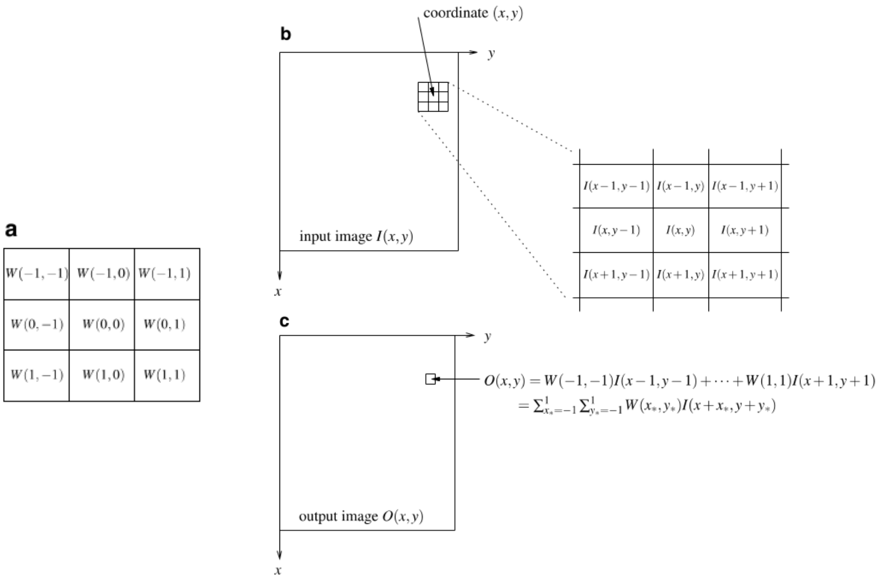
Gabor (1900-1979)

Translation invariance and locality



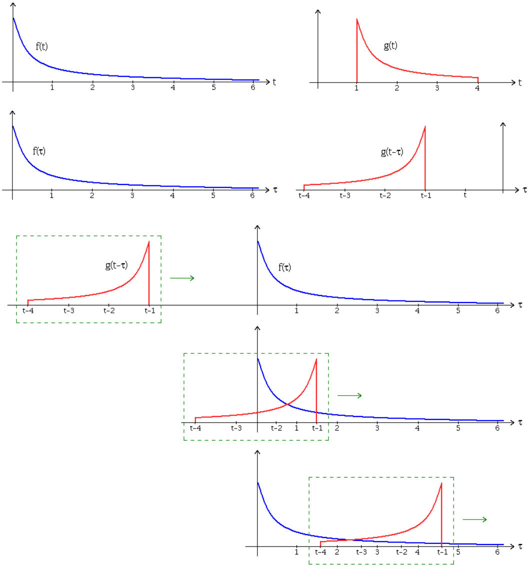
- Almost all structure in natural images is **position-invariant** and **local.** Therefore:
- Almost all low-level vision operations are based on patches.
- The universal mathematical framework for understanding the structure in images is the Fourier transform.

Filtering / Convolution 2-d (aka LSI system)



Figures from Hyvarinen, et al., 2009.

Convolution 1-d (Wikipedia)



Phasors

- The phasor is the complex valued signal

$$p(t) = \exp\left(i\omega t\right) = \cos\omega t + i\sin\omega t, \quad i = \sqrt{-1}$$

It represents sine and cosine in a single signal. (This is useful because all sine waves of a given frequency live in the same, 2-dimensional subspace.)

- Phasors are eigenfunctions of translation:

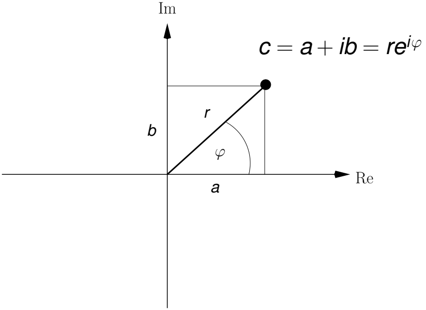
$$p(t-z) = e^{i\omega(t-z)} = e^{i\omega t}e^{-i\omega z} = e^{-i\omega z}p(t)$$

Digression: Complex numbers

- Complex numbers are "2d-vectors" with some special arithmetic, most of which is related to *Euler's formula*:

$$e^{i\varphi} = \cos\varphi + i\sin\varphi$$

- Most applications rely on jumping back-and-forth between cartesian and polar coordinates:



$$a = r\cos(\varphi)$$
$$b = r\sin(\varphi)$$
$$r = |c| = \sqrt{a^2 + b^2} \text{ : "amplitude"}$$
$$\varphi = \arg(c) = \operatorname{atan}\left(\frac{b}{a}\right) \text{ : "phase"}$$

Digression: Complex numbers

- Addition is the same as for 2d vectors.
- Multiplication is standard arithmetic in the polar representation:

$$c_1 \cdot c_2 = r_1 e^{i(\varphi_1)} \cdot r_2 e^{i(\varphi_2)} = r_1 \cdot r_2 \cdot e^{i(\varphi_1 + \varphi_2)}$$

  Thus, multiplication is *stretching* $+$ *rotation*.
- Multiplying a number by a complex number c of length 1.0, ie.

$$c = e^{i\alpha},$$

  amounts to rotating the number by α degrees counter clock-wise around the origin.

Digression: Complex numbers

- Other useful equations:
  - Conjugation is reflection at the real axis:

$$\bar{c} = a - ib = r \exp(-i\varphi)$$

  - It follows that $\bar{c}c = |c|^2$ and $\frac{1}{2}(\bar{c} + c) = \mathrm{real}(c)$
  - The standard inner product uses conjugation:

$$\langle \vec{c}, \vec{d} \rangle = \sum_i \bar{c}_i d_i$$

  - Why? Because now $\langle \vec{c}, \vec{c} \rangle = ||\vec{c}||^2$
  - In practice, use the function $\mathrm{atan2}()$ to compute the atan for polar representations.

— End of digression —

Phasors

- The phasor is the complex valued signal

$$p(t) = \exp\left(i\omega t\right) = \cos\omega t + i\sin\omega t, \quad i = \sqrt{-1}$$

  It represents sine and cosine in a single signal. (This is useful because all sine waves of a given frequency live in the same, 2-dimensional subspace.)
- Phasors are eigenfunctions of translation:

$$p(t-z) = e^{i\omega(t-z)} = e^{i\omega t}e^{-i\omega z} = e^{-i\omega z}p(t)$$

Phasors are eigenfunctions of convolution

$$
\begin{aligned}
(p * h)(t) &= \sum_{z=-\infty}^{\infty} h(z)p(t-z) \\
&= \Big(\sum_{z=-\infty}^{\infty} h(z)e^{-i\omega z} \Big) e^{i\omega t} \\
&=: \big(H(\omega)p\big)(t)
\end{aligned}
$$

- The constant $H(\omega)$ is called *frequency response* of the filter h.
- Its absolute value $|H(\omega)|$ is called *amplitude response*, its phase $\arg H(\omega)$ is called *phase response*.

Discrete Fourier Transform (1d)

- The Fourier transform decomposes a signal into phasors:

Inverse discrete Fourier Transform 1d

$$s(t) = \frac{1}{T} \sum_{k=0}^{T-1} S(k) e^{i \frac{2\pi}{T} tk} \quad t = 0, \ldots, T-1$$

Discrete Fourier Transform (DFT) 1d

$$S(k) = \sum_{t=0}^{T-1} s(t) e^{-i \frac{2\pi}{T} kt} \quad k = 0, \ldots, T-1$$

- $S(\omega)$ is called *spectrum* of the signal.
- $|S(\omega)|$ is called *amplitude spectrum*, $\arg S(\omega)$ is called *phase spectrum*.

2d waves

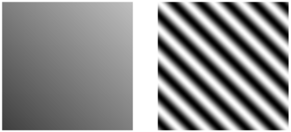
- How to generalize the concept of oscillation to 2d?
- Oscillations are functions of a scalar t. So first assign a scalar to image positions, then pass this scalar to a phasor. For example,

$$S(\boldsymbol{y}) = \exp(i\omega^{\mathrm{T}}\boldsymbol{y})$$

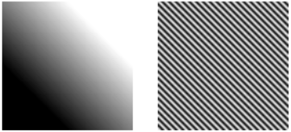
  where ω is called *frequency vector*.
- $\omega^{\mathrm{T}}\boldsymbol{y}$ grows in the direction of ω and is constant in the direction orthogonal to ω.

2d waves

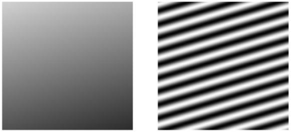
$\vec{\omega} = [0.5; 0.5]$



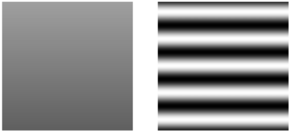
$\vec{\omega} = [2.0; 2.0]$



$\vec{\omega} = [-0.3; 1.0]$



$\vec{\omega} = [0.0; 0.5]$



Separability of complex waves

- Complex valued waves are **separable**:

$$\begin{aligned} S(\boldsymbol{y}) &= \exp(i(\boldsymbol{\omega}^T \boldsymbol{y})) \\ &= \exp(i\omega_1 y_1 + i\omega_2 y_2)) \\ &= \exp(i\omega_1 y_1) \cdot \exp(i\omega_2 y_2) \\ &=: S_1(y_1) \cdot S_2(y_2) \end{aligned}$$

- *The same is not true of real valued waves.*

DFT on images

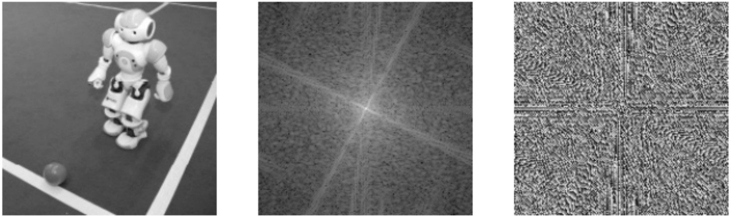
Inverse Discrete Fourier Transform in 2d

$$s(m,n) = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{\ell=0}^{N-1} S(k,\ell)e^{i2\pi\left(\frac{km}{M}+\frac{\ell n}{N}\right)}$$

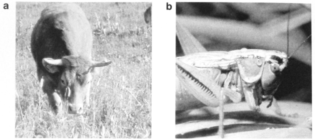
Discrete Fourier Transform (DFT) in 2d

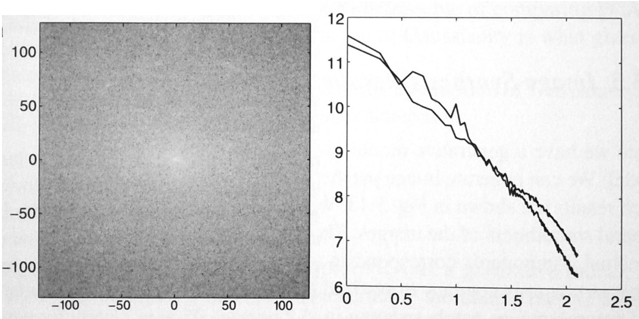
$$S(k,\ell) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} s(m,n)e^{-i2\pi\left(\frac{km}{M}+\frac{\ell n}{N}\right)}$$

Spectrum example



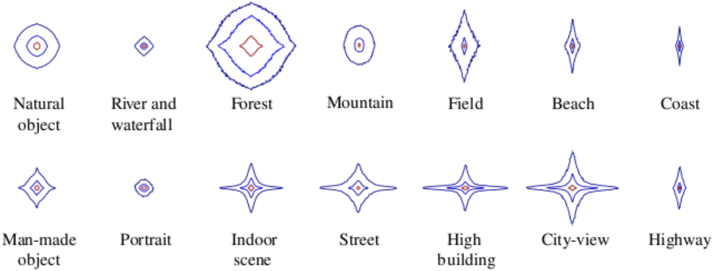
More amplitude spectra (average over cross-sections on the right)





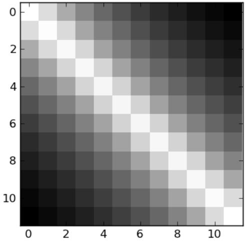
from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

Torralba, Oliva; 2003

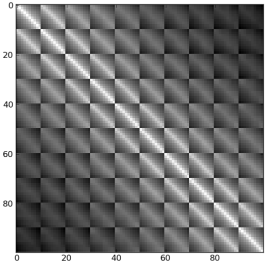


PCA and Fourier transform (1d)

- Due to translation invariance, the covariance matrix of natural images shows very strong structure:



cov 1-d scan lines



cov of images

PCA and Fourier transform (1d)

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

$$
\begin{aligned}
(Cp)(t) &= \sum_{t'} \mathrm{cov}(t, t') e^{i\omega t'} \\
&= \sum_{t'} c(t - t') e^{i\omega t'} \\
&= \sum_{z} c(z) e^{i\omega t} e^{-i\omega z} \\
&= \Big[\sum_{z} c(z) e^{-i\omega z} \Big] e^{i\omega t} =: \lambda_\omega e^{i\omega t}
\end{aligned}
$$

- (In fact, multiplying by the covariance matrix is a convolution.)

PCA and Fourier transform (1d)

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

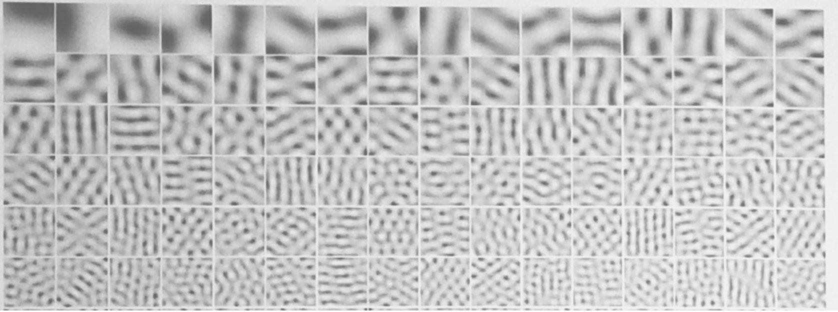
$$
\begin{aligned}
& \sum_{t'} \operatorname{cov}(t, t') e^{i\omega t'} \\
= \; & \Big[\sum_{z} c(z) e^{-i\omega z} \Big] e^{i\omega t} \\
= \; & \Big[c(0) + \sum_{z=1}^{\frac{T-1}{2}} c(z) \big(e^{-i\omega z} + e^{i\omega z} \big) \Big] e^{i\omega t} \\
= \; & \Big[c(0) + 2 \sum_{z=1}^{\frac{T-1}{2}} c(z) \cos(\omega z) \Big] e^{i\omega t}
\end{aligned}
$$

PCA and Fourier transform (2d)

- In 2d:

$$\begin{aligned}
(Cw)(t) &= \sum_{x',y'} \operatorname{cov}\big((x,y),(x',y')\big)\, e^{i(\omega_1 x' + \omega_2 y')} \\
&= \sum_{x',y'} c\big(x-x', y-y'\big)\, e^{i(\omega_1 x' + \omega_2 y')} \\
&= \sum_{\xi,\eta} c(\xi,\eta) e^{i(\omega_1 x - \omega_1 \xi + \omega_2 y - \omega_2 \eta)} \\
&= \Big[\sum_{\xi,\eta} c(\xi,\eta) e^{-i(\omega_1 \xi + \omega_2 \eta)}\Big]\, e^{i(\omega_1 x + \omega_2 y)}
\end{aligned}$$

PCA example (first 96 EVs)



from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

Fourier transform and convolution

Convolution in the time-domain is multiplication in the frequency domain.

- "Proof:" The Fourier transform of the convolved signal, $g(t) = s(t) * h(t) = \sum_k h(k) \cdot s(t-k)$, can be written

$$
\begin{aligned}
G(\omega) &= \sum_t \left[\sum_k h(k) \cdot s(t-k) \right] e^{-i\omega t} \\
&= \sum_t \sum_k h(k) \cdot \; e^{-i\omega k} \cdot s(t-k) \, e^{-i\omega(t-k)} \\
&= \sum_k h(k) \cdot \; e^{-i\omega k} \cdot \sum_t s(t-k) \, e^{-i\omega(t-k)} \\
&= H(\omega) \cdot S(\omega)
\end{aligned}
$$

- This can be used to speed up conv net inference and training using FFT (eg. Mathieu, et al.)

Fourier transform and convolution

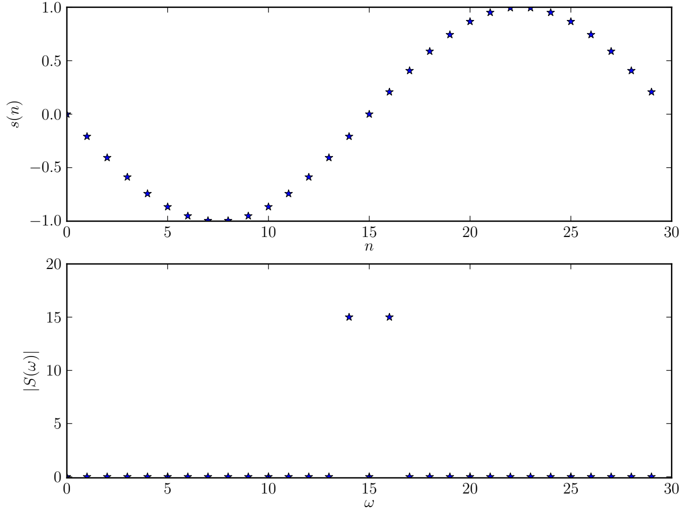
Multiplication in the time-domain is convolution in the frequency domain.

- This is the source of ringing, aliasing and **leakage** effects.

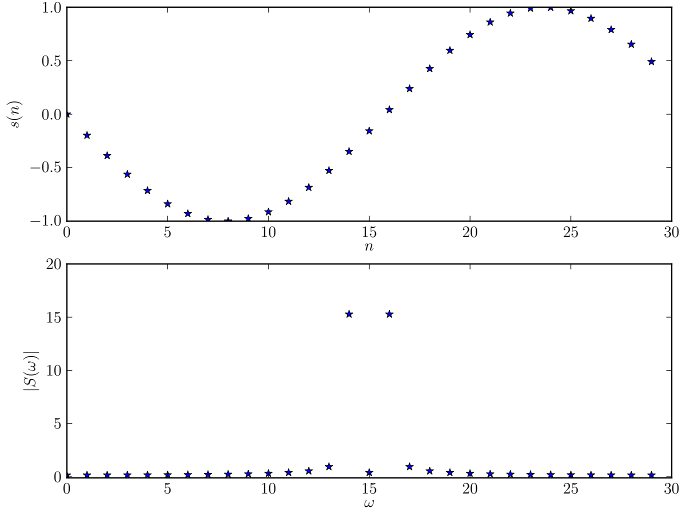
DFT leakage

- We can think of the DFT of a finite signal as the DFT of a periodic signal after multiplying it by a rectangular window.
- The DFT spectrum you get can be thought of as 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 signal components whose frequencies are integer multiples of the window length.
- For any other components, the convolution will generate additional components in the spectrum.
- This effect is known as **leakage**.

Leakage

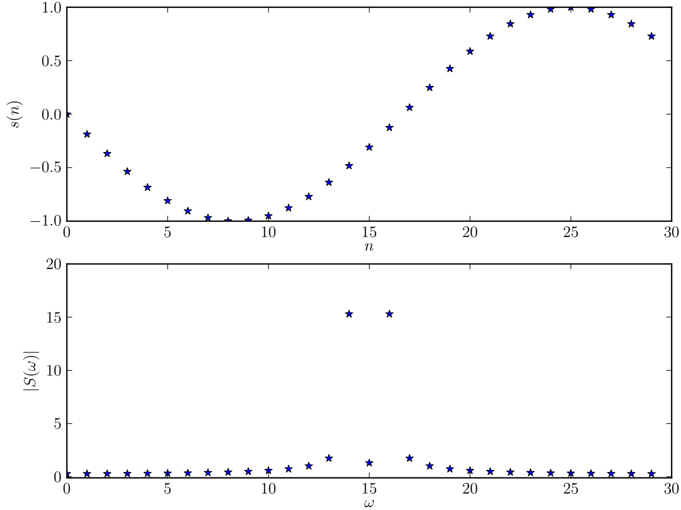


Leakage



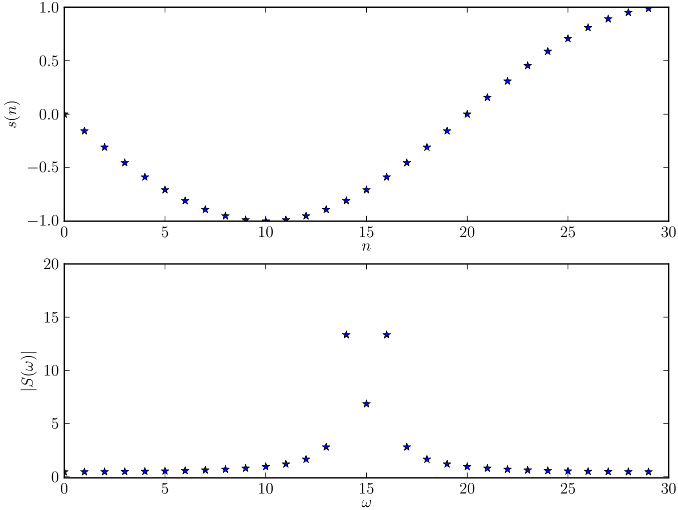
1.0
0.5
0.0
−0.5
−1.0
$s(n)$
0
5
10
15
20
25
30
n
20
15
10
5
0
$|S(\omega)|$
0
5
10
15
20
25
30
ω

Leakage

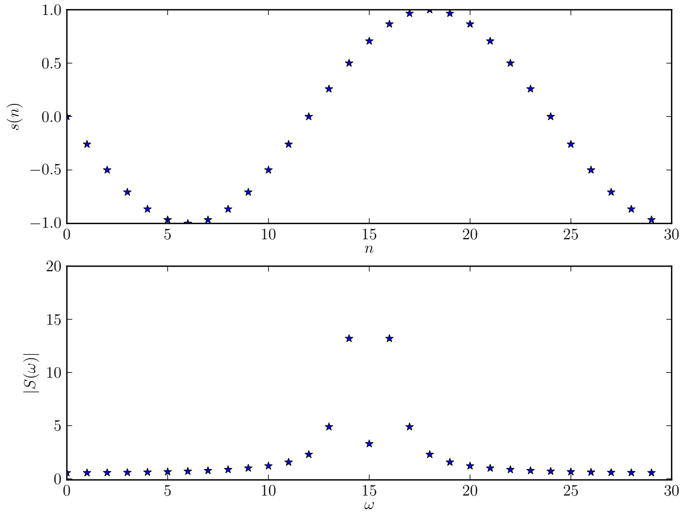


1.0
0.5
0.0
−0.5
−1.0
$s(n)$
0
5
10
15
20
25
30
n
20
15
10
5
0
$|S(\omega)|$
0
5
10
15
20
25
30
ω

Leakage



Leakage

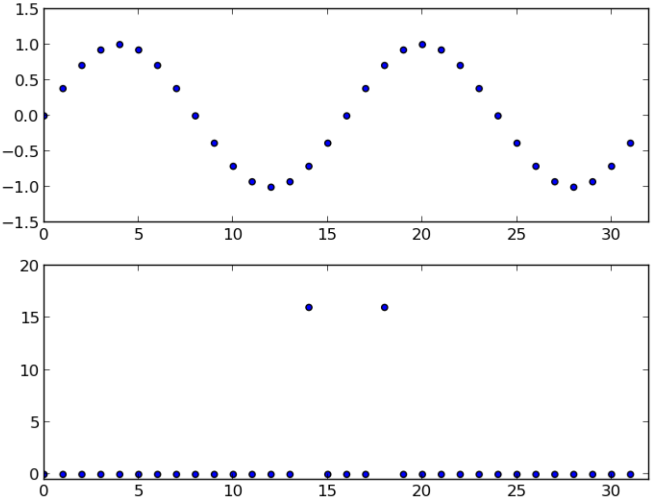


1.0
0.5
0.0
−0.5
−1.0
$s(n)$
0
5
10
15
20
25
30
n
20
15
10
5
0
$|S(\omega)|$
0
5
10
15
20
25
30
ω

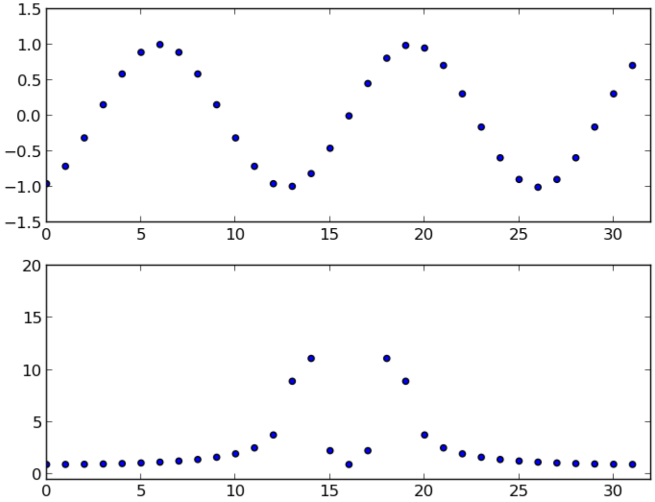
Windowing

- Leakage cannot be avoided.
- But a window other than the box-window may lead to different, possibly less undesirable, leakage properties.

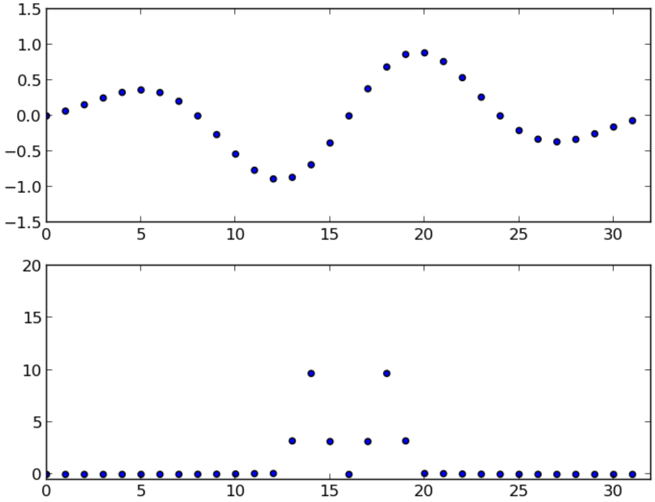
Leakage with box window



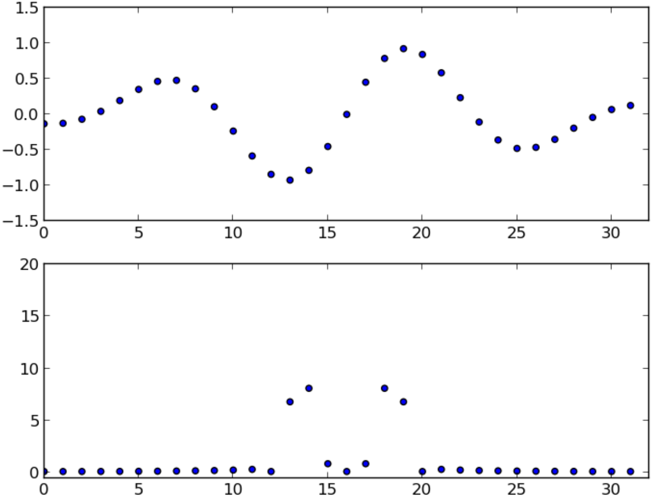
Leakage with box window



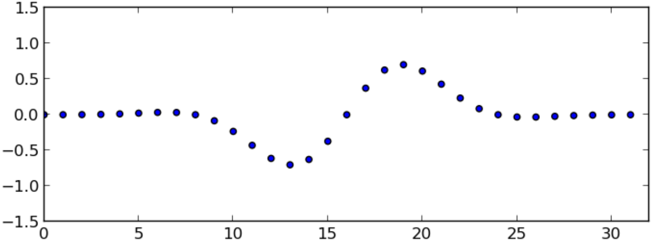
Leakage with Gaussian window

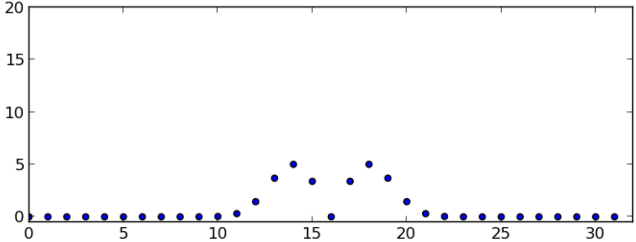


Leakage with Gaussian window

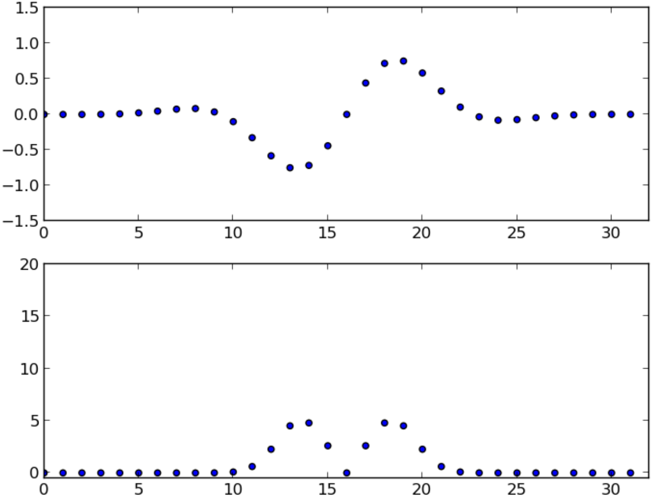


Leakage with small Gaussian window





Leakage with small Gaussian window

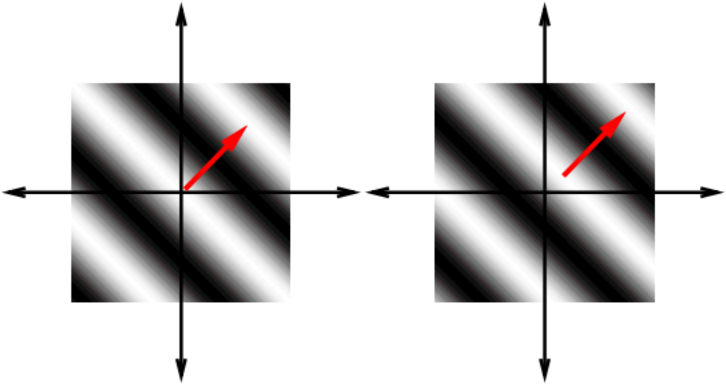


Windowing and Short Time Fourier Transform

- An application of window functions is the **Short-Time Fourier Transform (STFT)**.
- Fourier-transform the signal *locally*, then view the resulting set of spectra as a function of time or space.
- In 1d, the result (sometimes just amplitudes) is called *spectrogram*.
- An STFT using a Gaussian window is also called **Gabor transform**.

Gabor feature

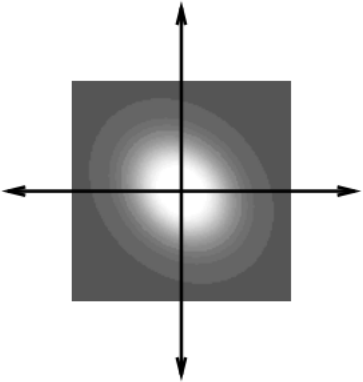
Wave:



figures by Javier Movellan

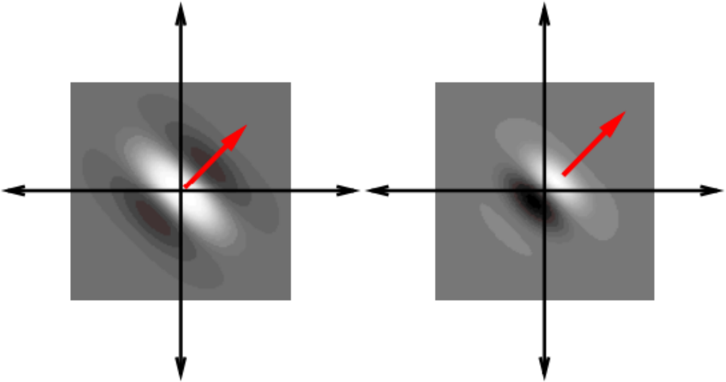
Gabor feature

Window:



figures by Javier Movellan

Gabor feature

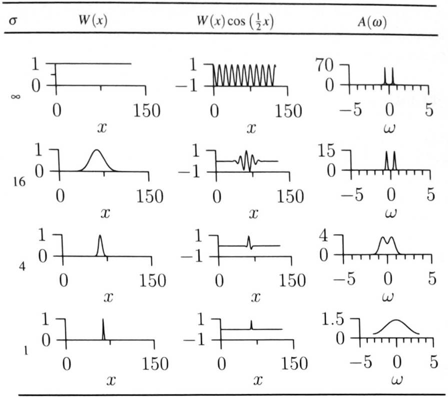


$$\mathrm{gaborfeature}(K, \sigma, x_0, y_0, \gamma, u, v, P) =$$

$$K \exp\left(-\frac{1}{\sigma^2}((x - x_0)^2 + \gamma^2 (y - y_0)^2)\right) \cdot \exp\left(i 2\pi (ux + vy) + P\right)$$

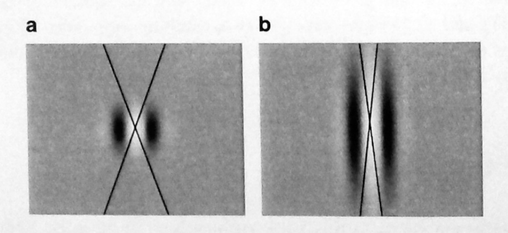
figures by Javier Movellan

The uncertainty principle



from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

In 2d: orientation uncertainty



from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

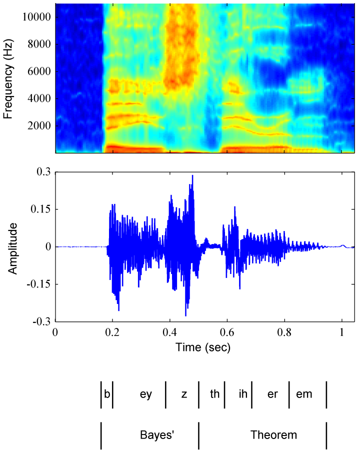
Frequency channels

- In many applications, local Gabor features are used as *filters*, ie. they are scanned across the image.
- This naturally raises the question:
- What is the amplitude response of a Gabor filter?

Frequency channels

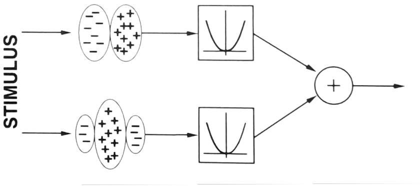
- In many applications, local Gabor features are used as *filters*, ie. they are scanned across the image.
- This naturally raises the question:
- What is the amplitude response of a Gabor filter?
- It is a **localized blob** in the frequency domain, because the Fourier transform of a phasor times a Gaussian will be a delta-peak convolved with a Gaussian.
- So Gabor filters are **oriented bandpass filters**.

A spectrogram (top) of an utterance



- (from Bishop, 2006)
- The visual analog of the spectrogram is the feature map (a 3-dimensional object).

Biological complex cells



- also (Hubel and Wiesel, 1959)
- A Fourier feature pair with 90 deg phase difference is known as **quadrature pair**.
- Conv nets do not typically use these. Instead they pool (after *rectifying*), which has a similar effect.

Why PCA yields Fouriers (part II)

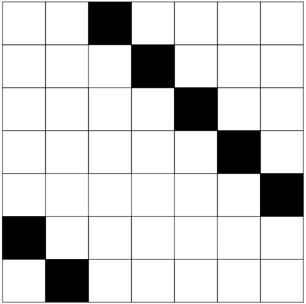
- Assume that the data density is invariant wrt. to orthogonal transformations T, then

$$
\begin{aligned}
& \log p(\boldsymbol{x}) = \log p(T\boldsymbol{x}) \\
\iff \quad & \boldsymbol{x}^{\mathrm{T}}\Sigma^{-1}\boldsymbol{x}^{\mathrm{T}} = \boldsymbol{x}^{\mathrm{T}}T^{\mathrm{T}}\Sigma^{-1}T\boldsymbol{x} \quad \forall \boldsymbol{x} \\
\iff \quad & \Sigma^{-1} = T^{\mathrm{T}}\Sigma^{-1}T \\
\iff \quad & T\Sigma^{-1} = \Sigma^{-1}T
\end{aligned}
$$

- Since Σ^{-1} commutes with T, it has to have the same eigenvectors (which for translations are Fourier components).

Why feature learning yields Fouriers

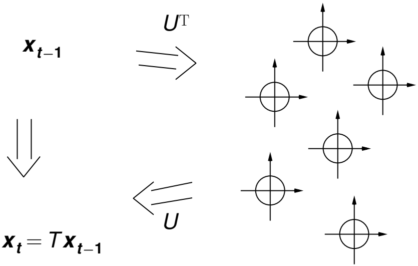
Circulants



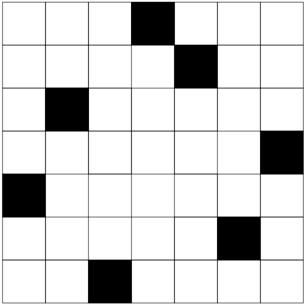
A circulant matrix

Orthogonal transformations

$$U^{\mathrm{T}} T U = \begin{bmatrix} R_1 & & \\ & \ddots & \\ & & R_k \end{bmatrix} \qquad R_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix}$$



Higher layers?



A permutation matrix