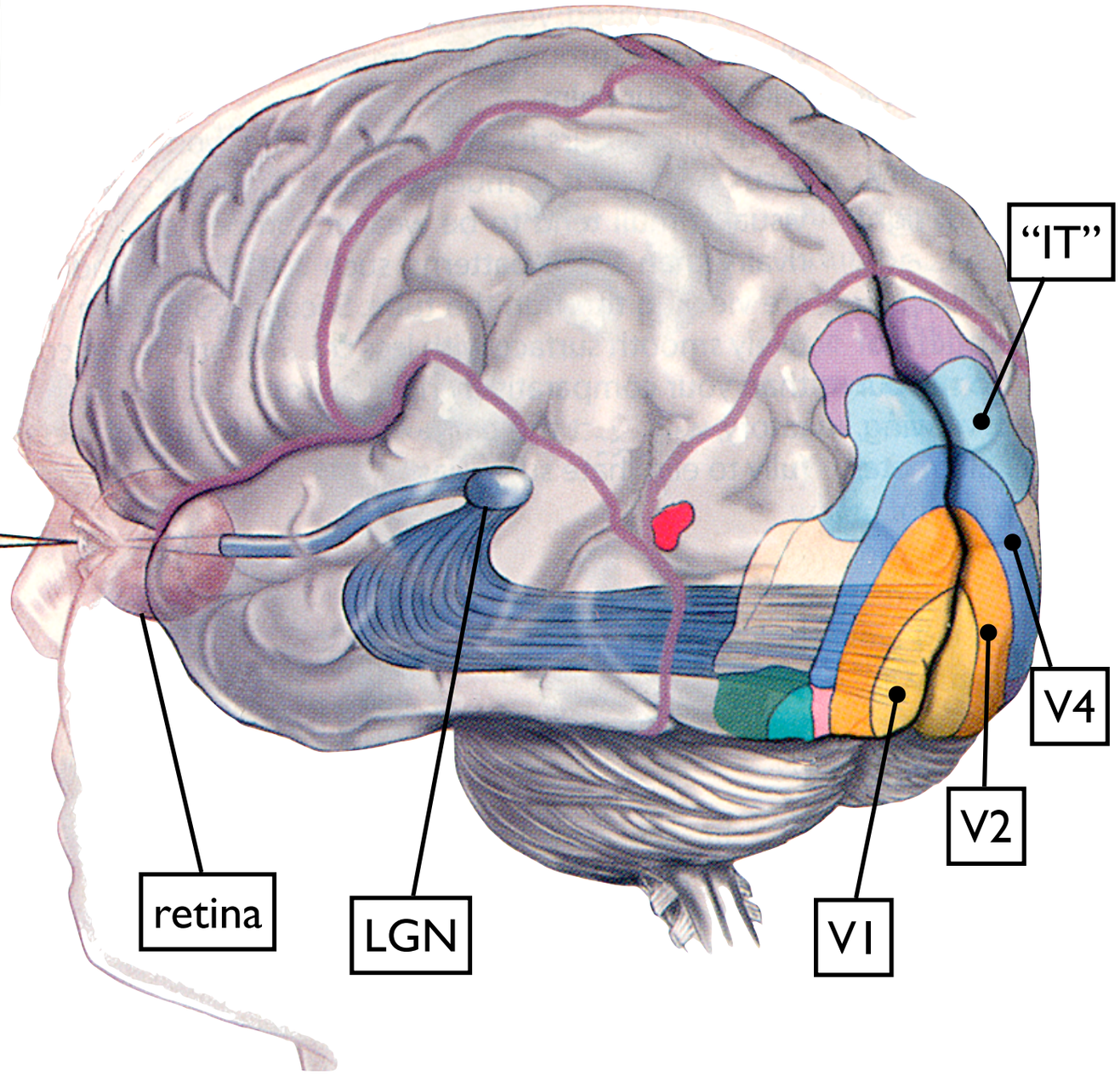
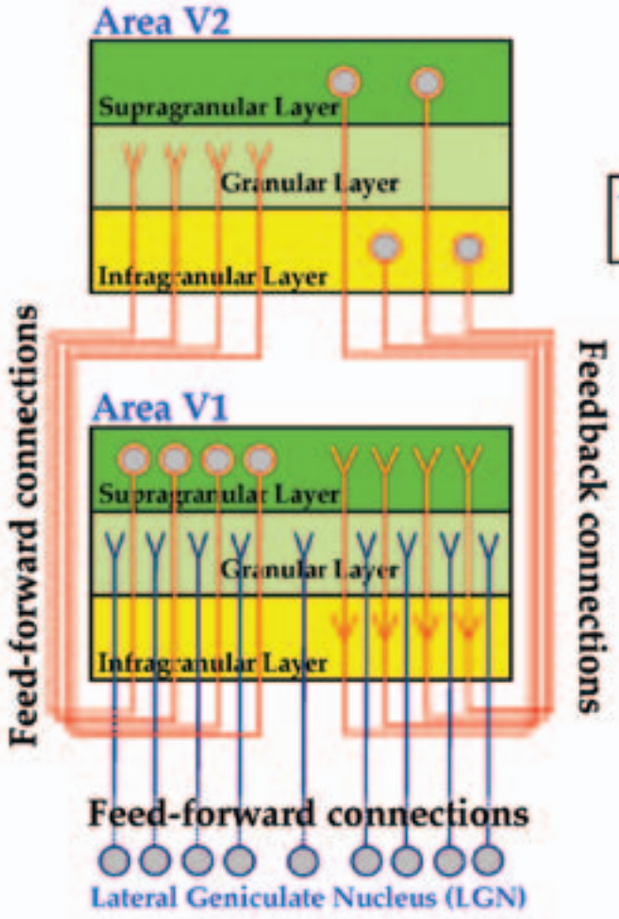
Sensory Coding & Hierarchical Representations

Michael S. Lewicki

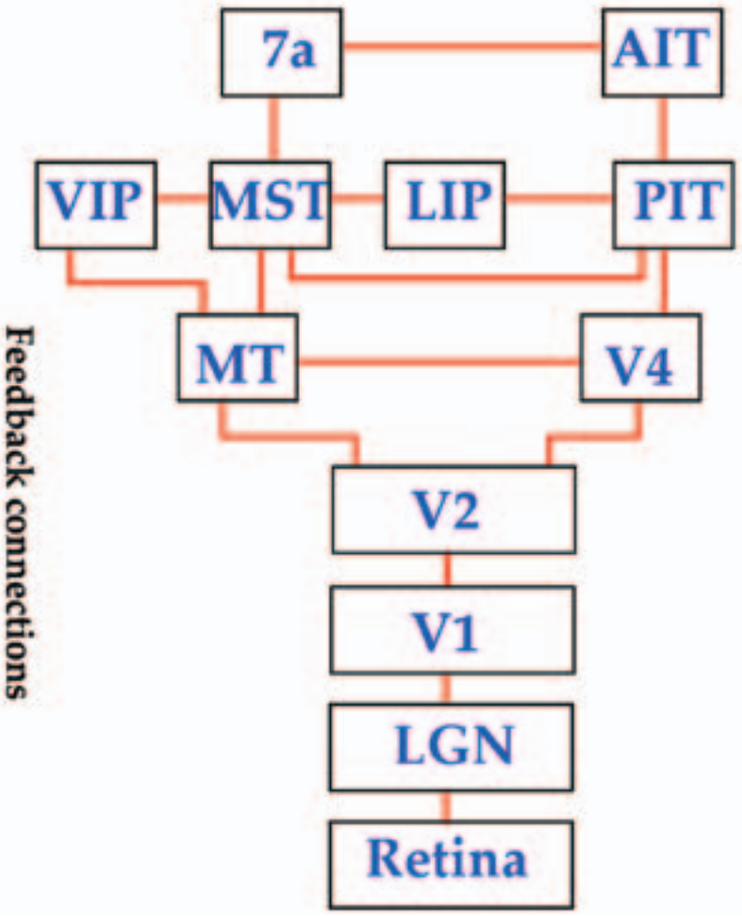
Computer Science Department &
Center for the Neural Basis of Cognition
Carnegie Mellon University


"IT"
V4
V2
V1
LGN
retina

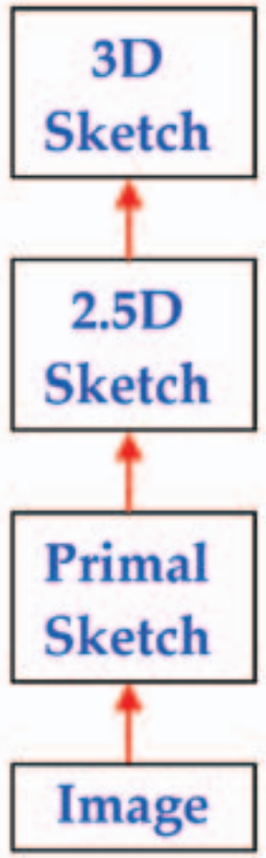
A Laminar patterns of interconnections



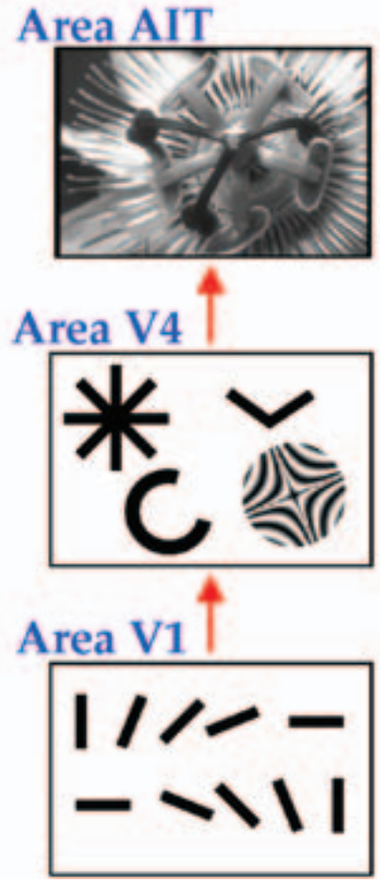
B Visual anatomical hierarchy

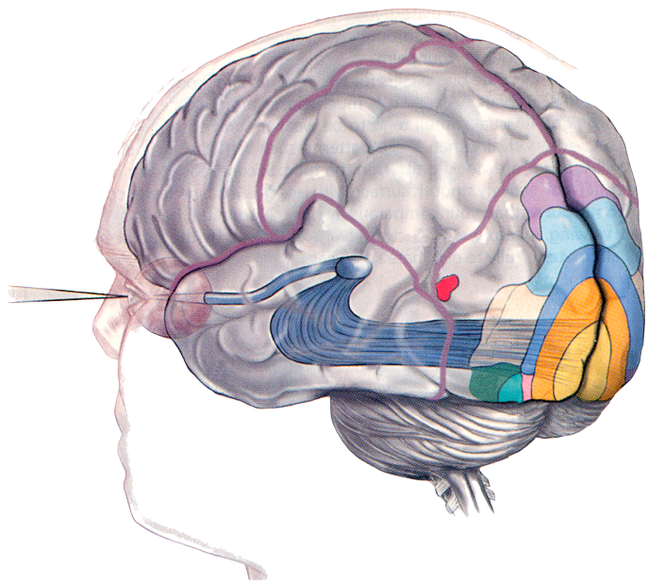


C Marr model of hierarchical visual processing

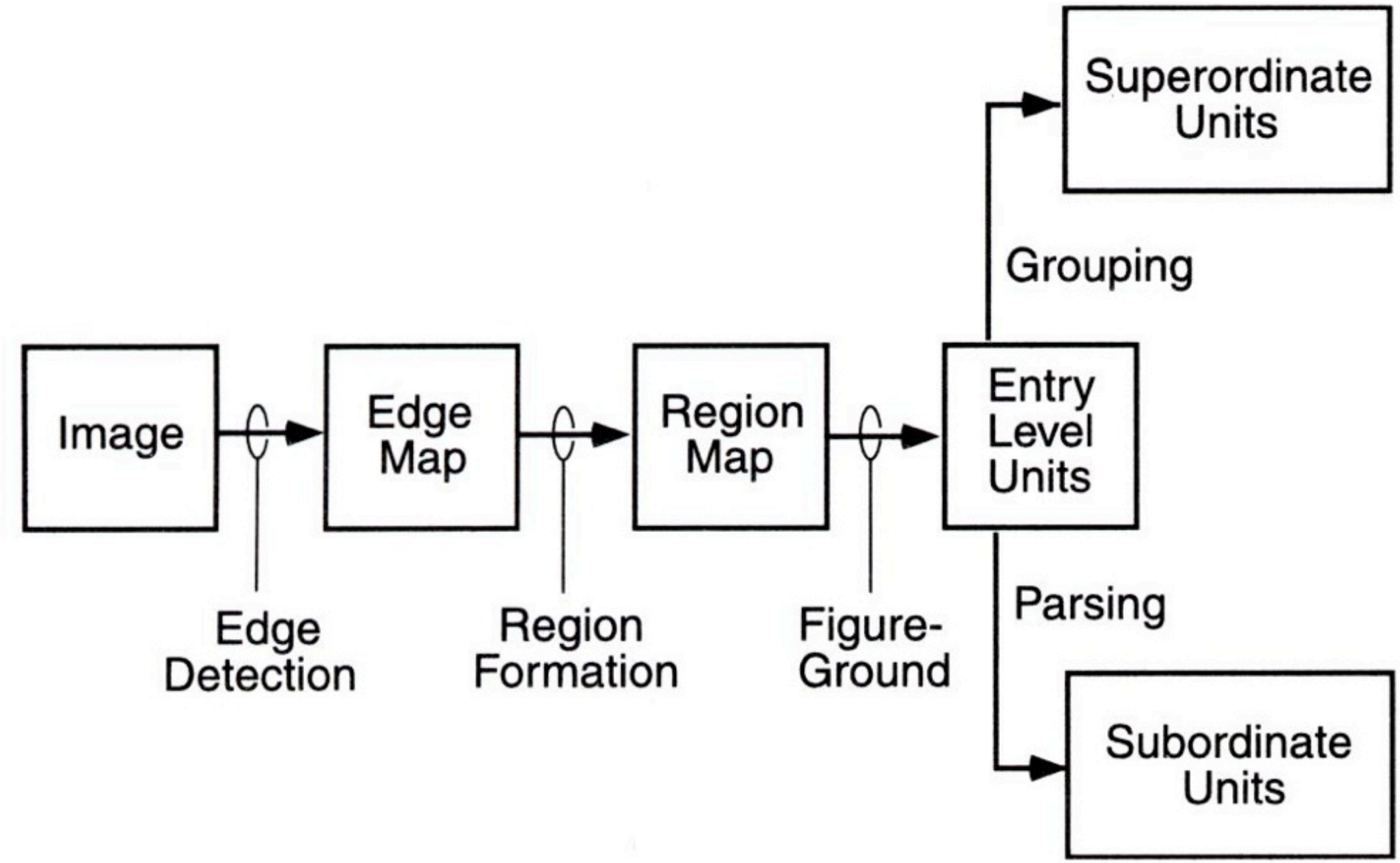


D Anatomical hierarchy as a substrate for the functional hierarchy



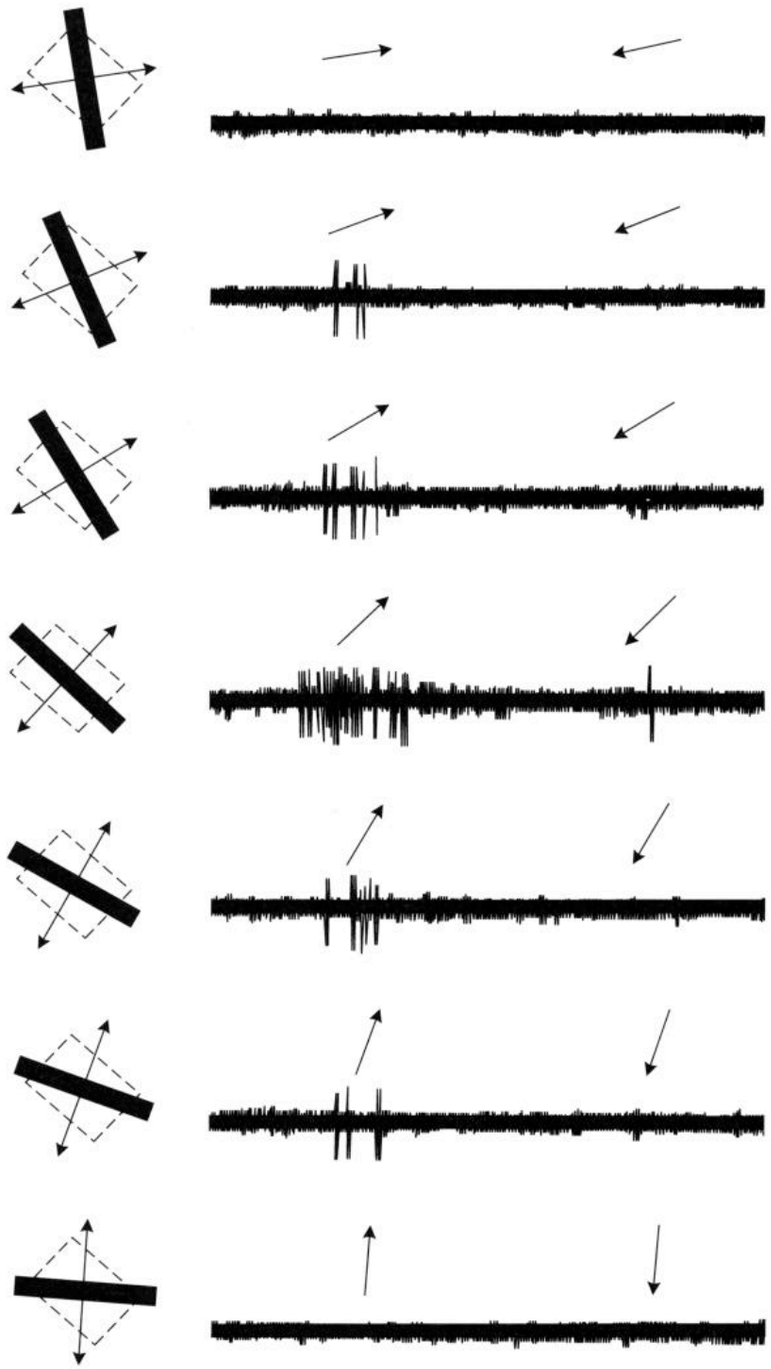


from Hedgé and Felleman, 2007

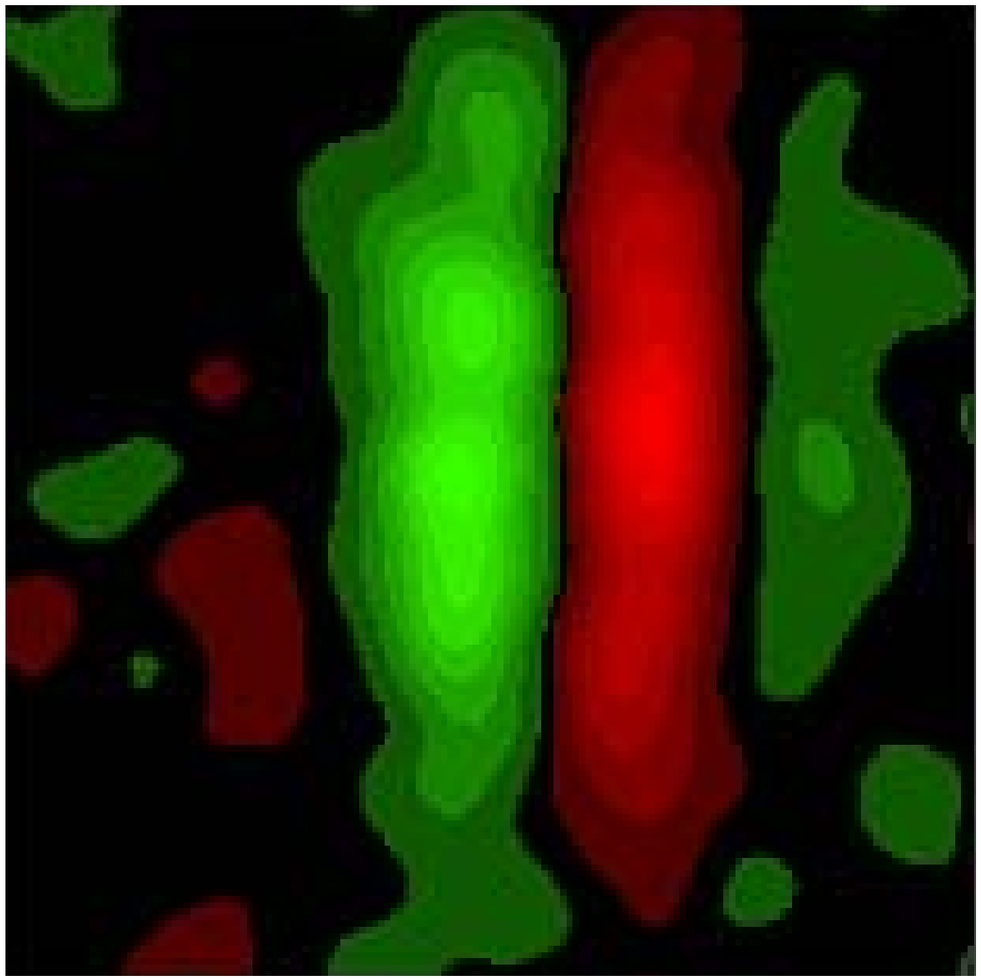


Palmer and Rock, 1994

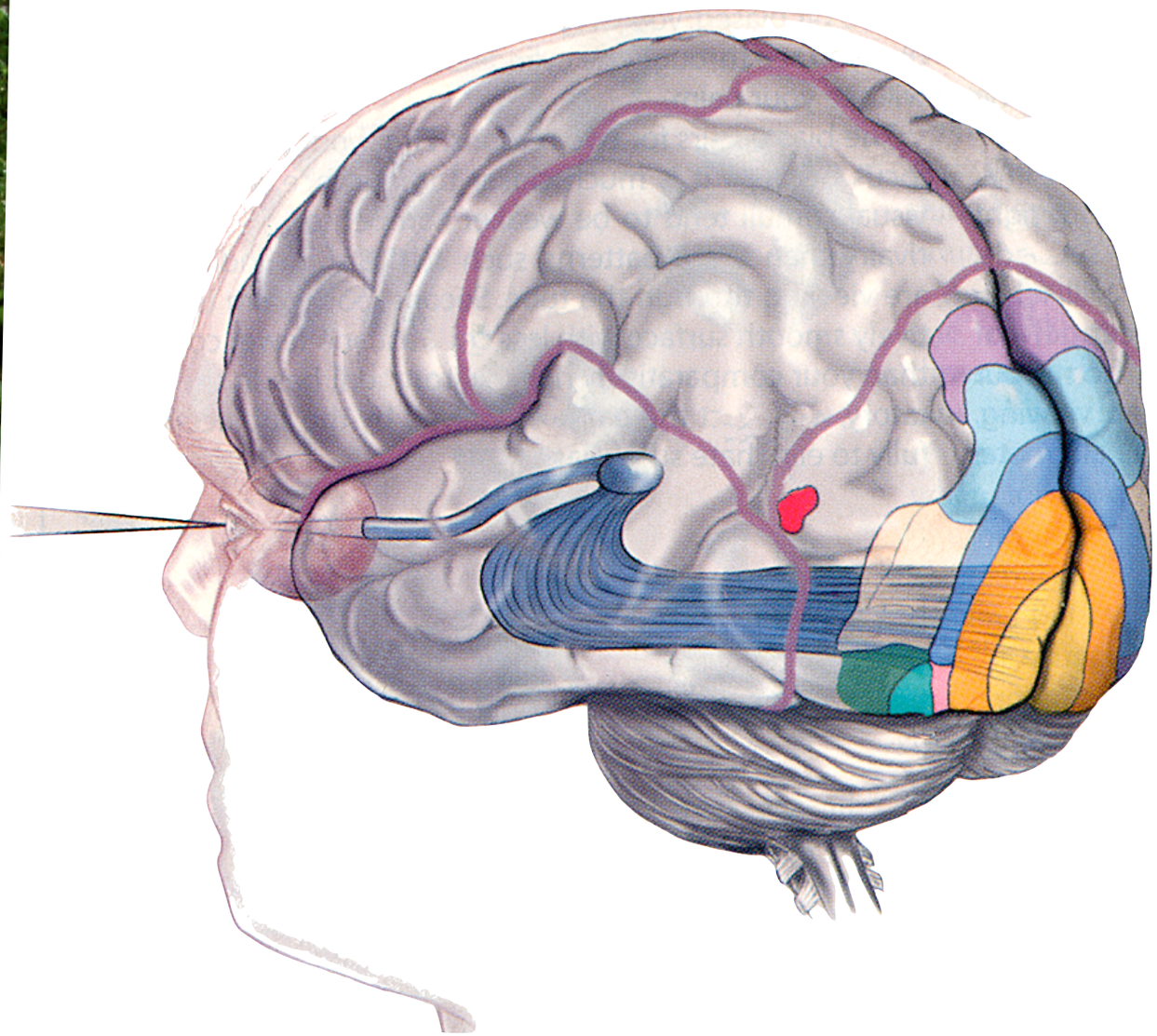
V1 simple cells

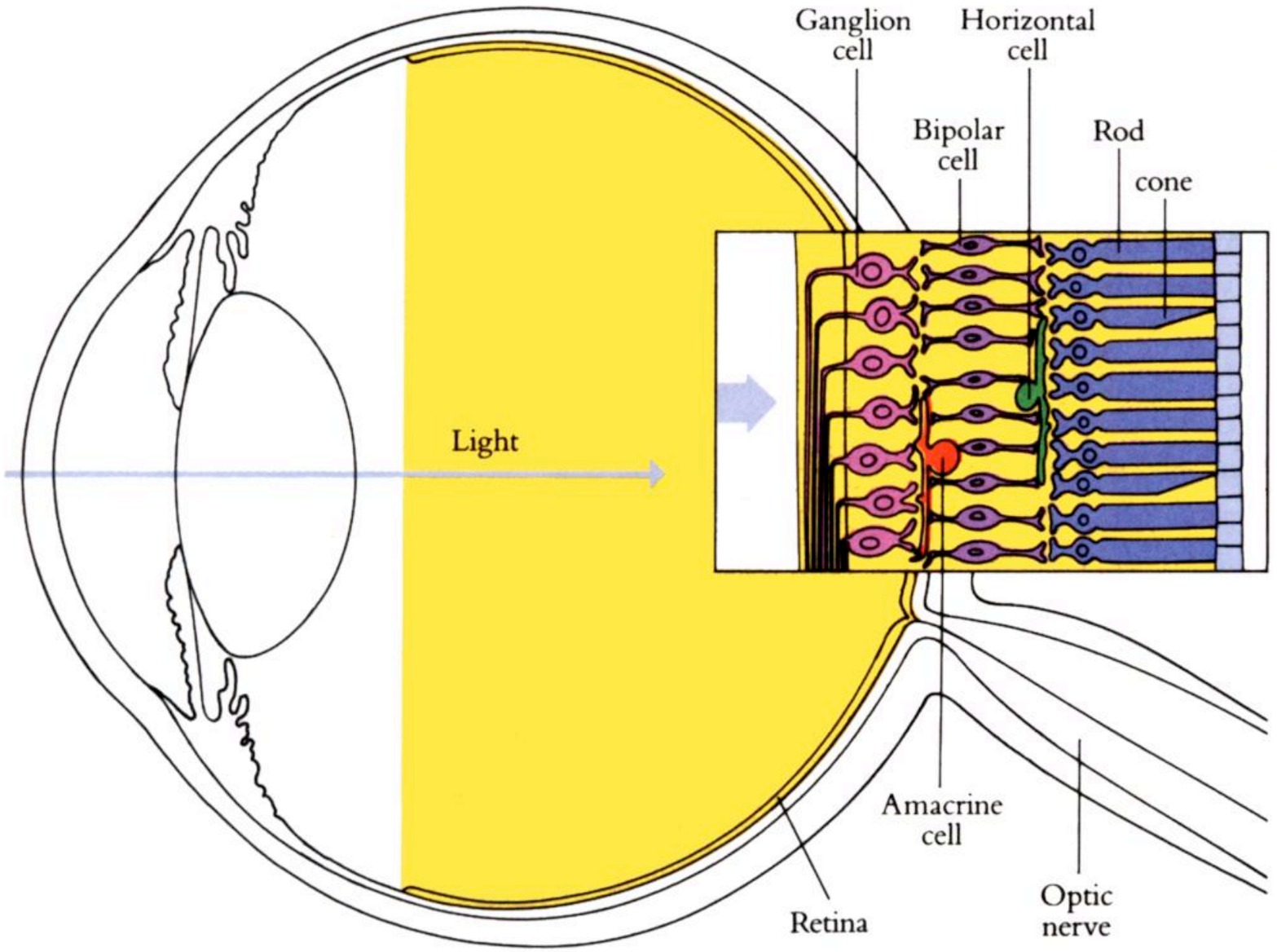


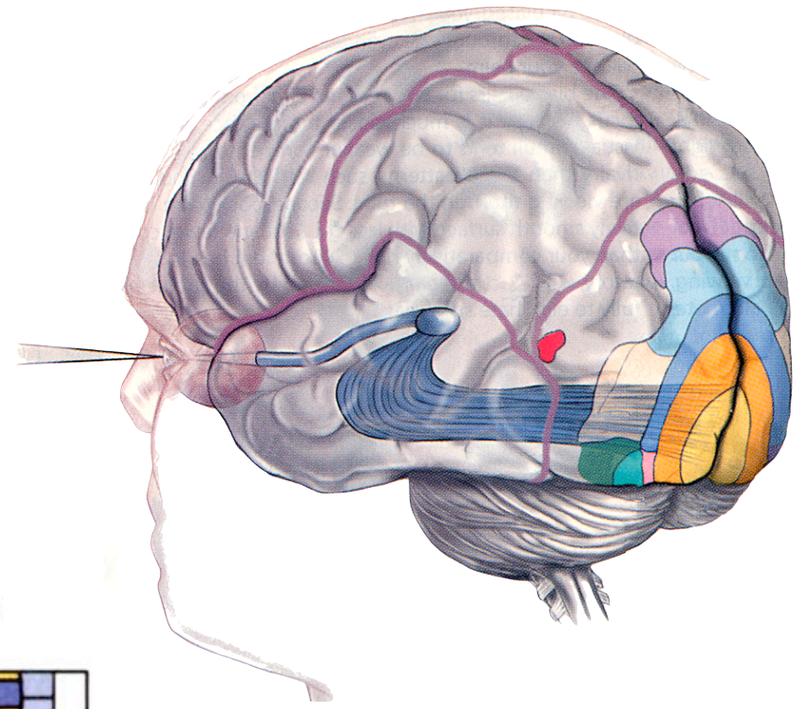
Hubel and Wiesel, 1959



DeAngelis, et al, 1995

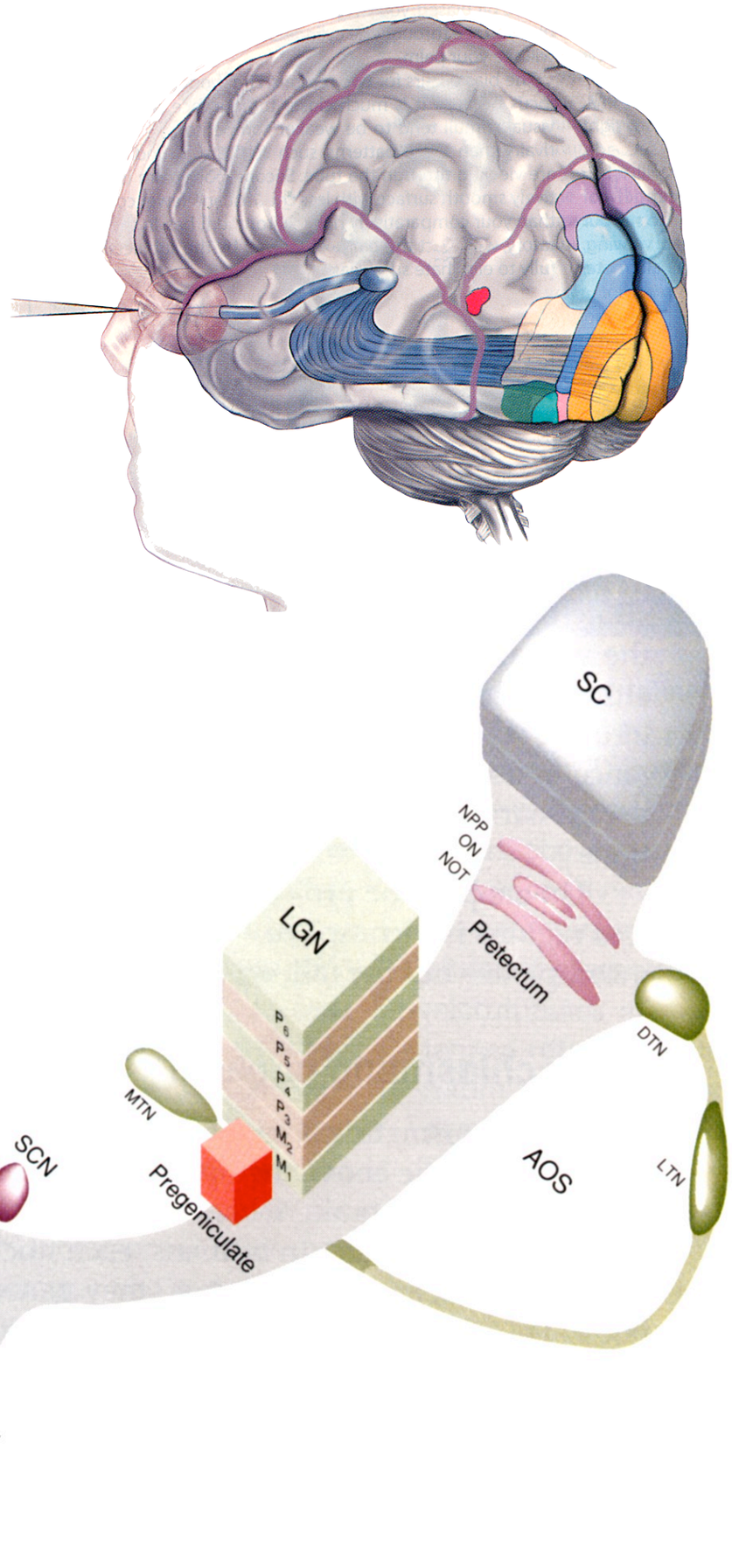



Ganglion cell
Horizontal cell
Bipolar cell
Rod
cone
Light
Amacrine cell
Retina
Optic nerve

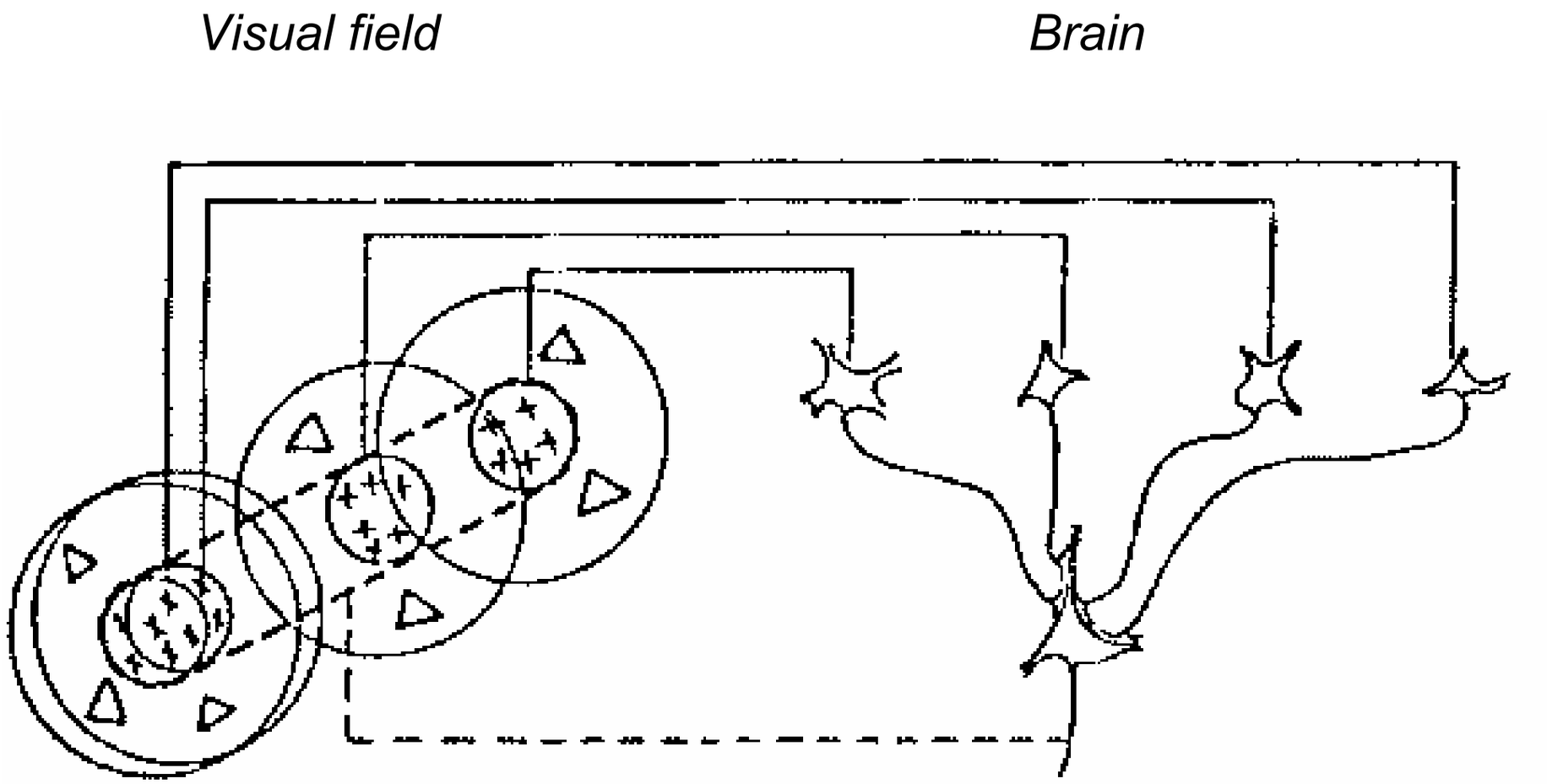


Out of the retina

- > 23 distinct neural pathways, no simple function division
- *Suprachiasmatic nucleus*: circadian rhythm
- *Accessory optic system*: stabilize retinal image
- *Superior colliculus*: integrate visual and auditory information with head movements, direct eyes
- *Pretectum*: plays adjusting pupil size, track large moving objects
- *Pregeniculate*: cells responsive to ambient light
- *Lateral geniculate (LGN)*: main "relay" to visual cortex; contains 6 distinct layers, each with 2 sublayers. Organization very complex

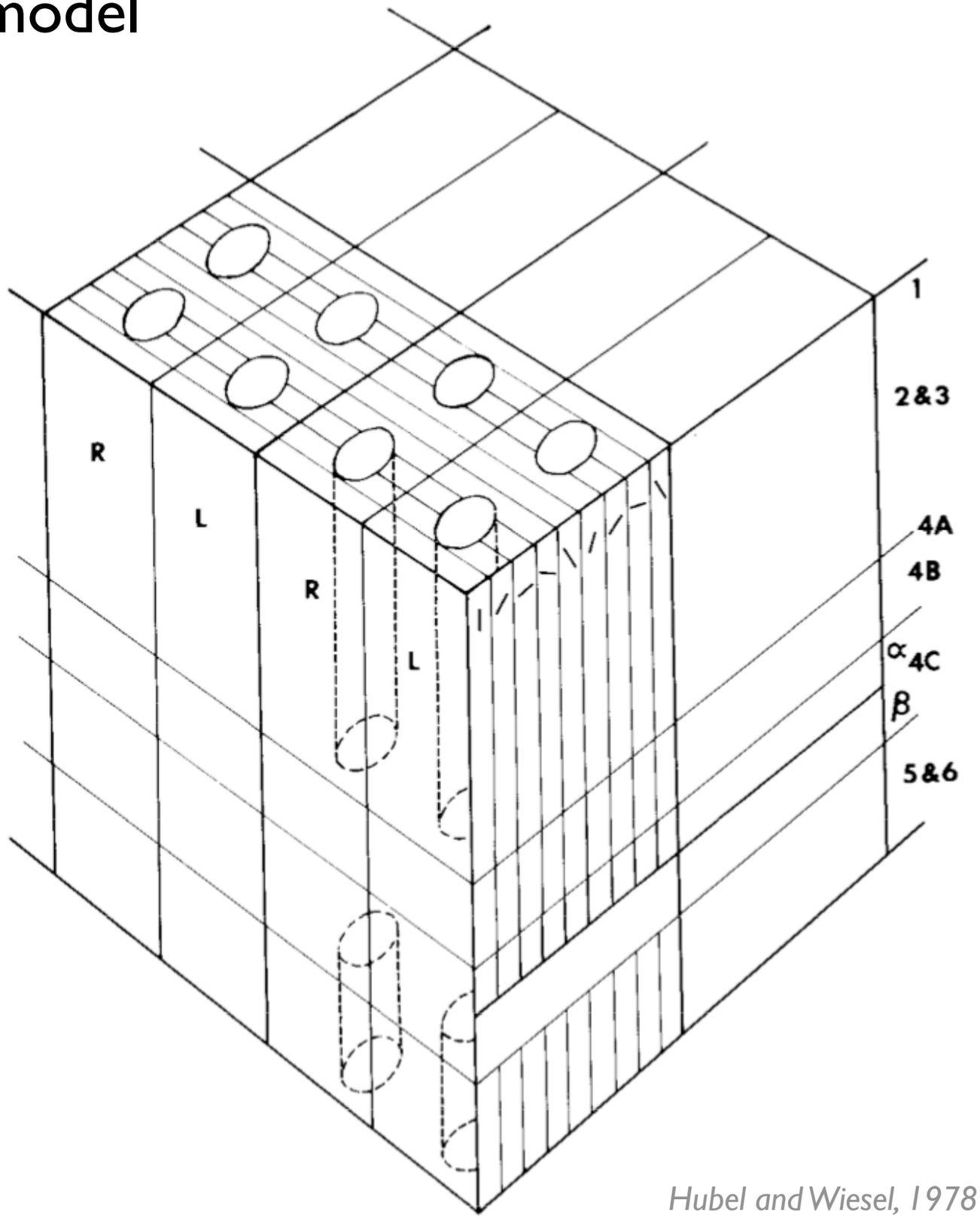


V1 simple cell integration of LGN cells



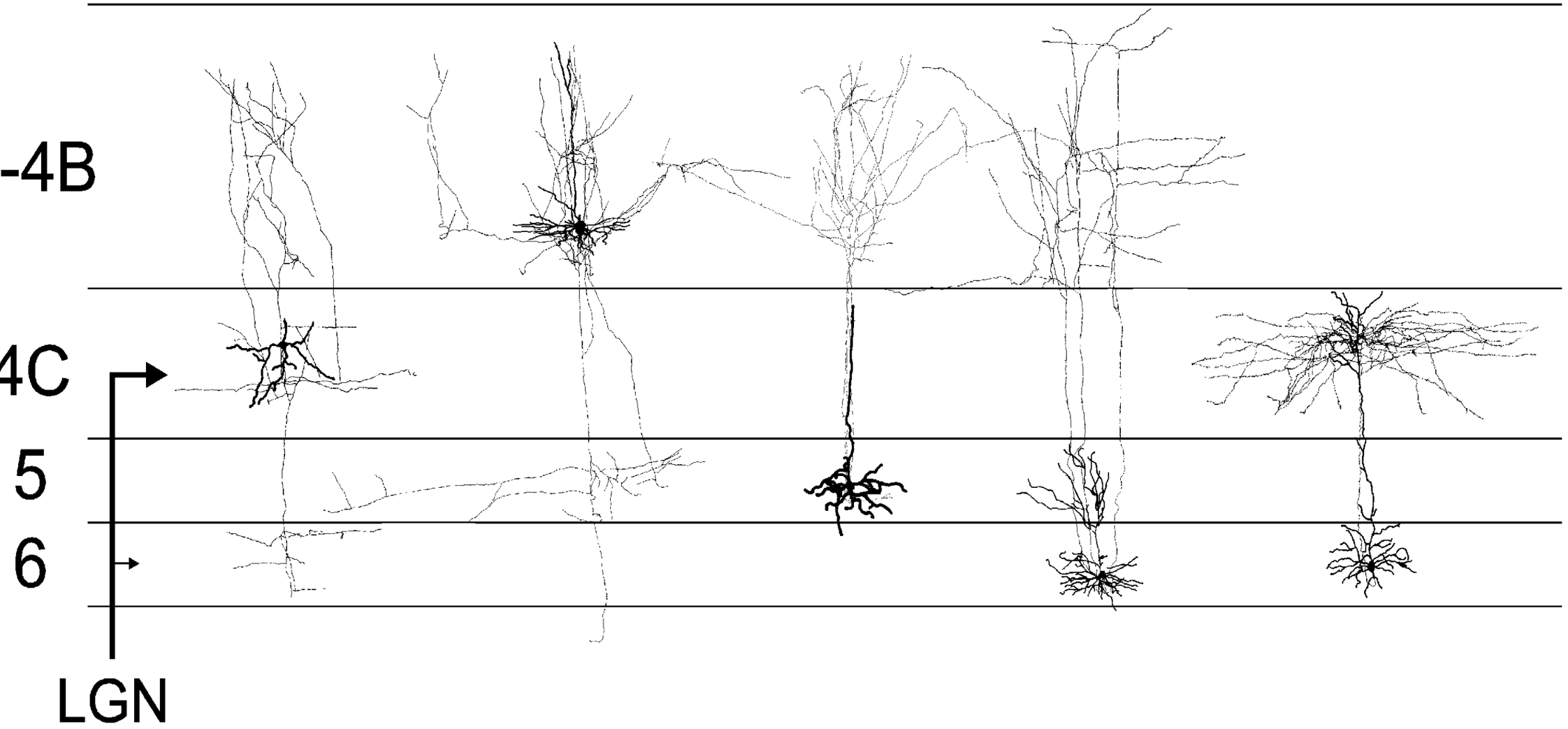
Hubel and Wiesel, 1963

Hyper-column model



Hubel and Wiesel, 1978

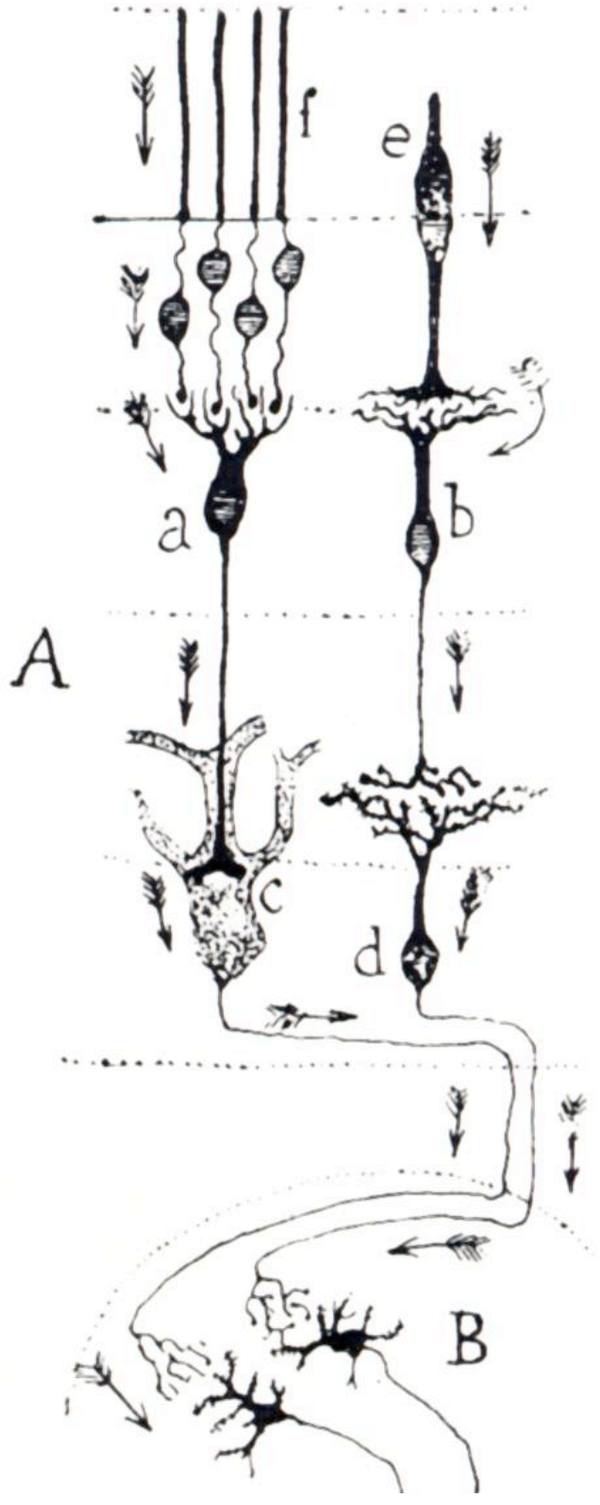
Anatomical circuitry in V1



from Callaway, 1998

Where is this headed?

Ramon y Cajal



A wing would be a most mystifying structure if one did not know that birds flew.

Horace Barlow, 1961

An algorithm is likely to be understood more readily by understanding the nature of the problem being solved than by examining the mechanism in which it is embodied.

David Marr, 1982

A theoretical approach

- Look at the system from a functional perspective:
  What problems does it need to solve?

- Abstract from the details:
  Make predictions from theoretical principles.

- Models are bottom-up; theories are top-down.

What are the relevant computational principles?

Representing structure in natural images

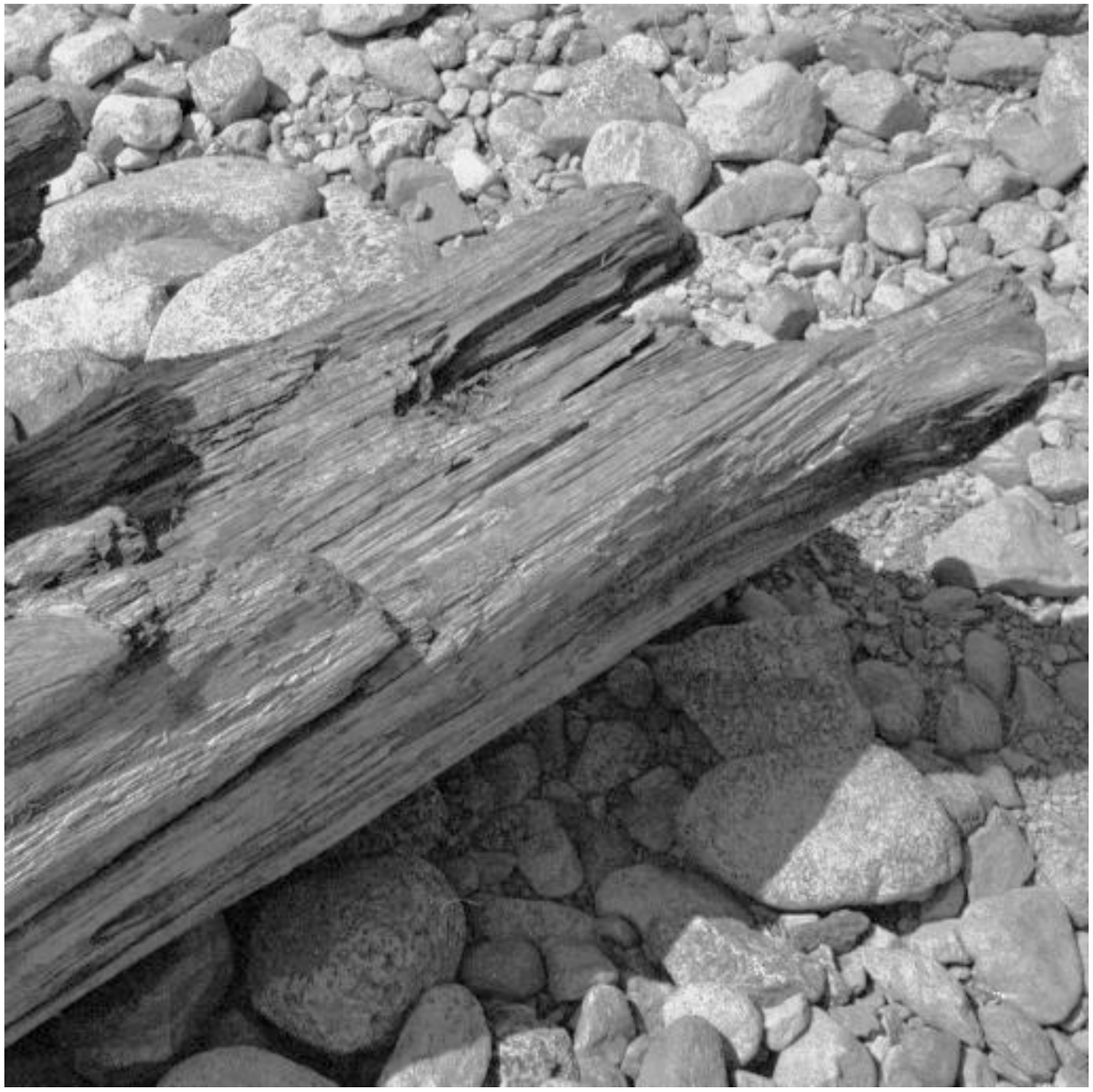


image from Field (1994)

Representing structure in natural images

image from Field (1994)

Representing structure in natural images

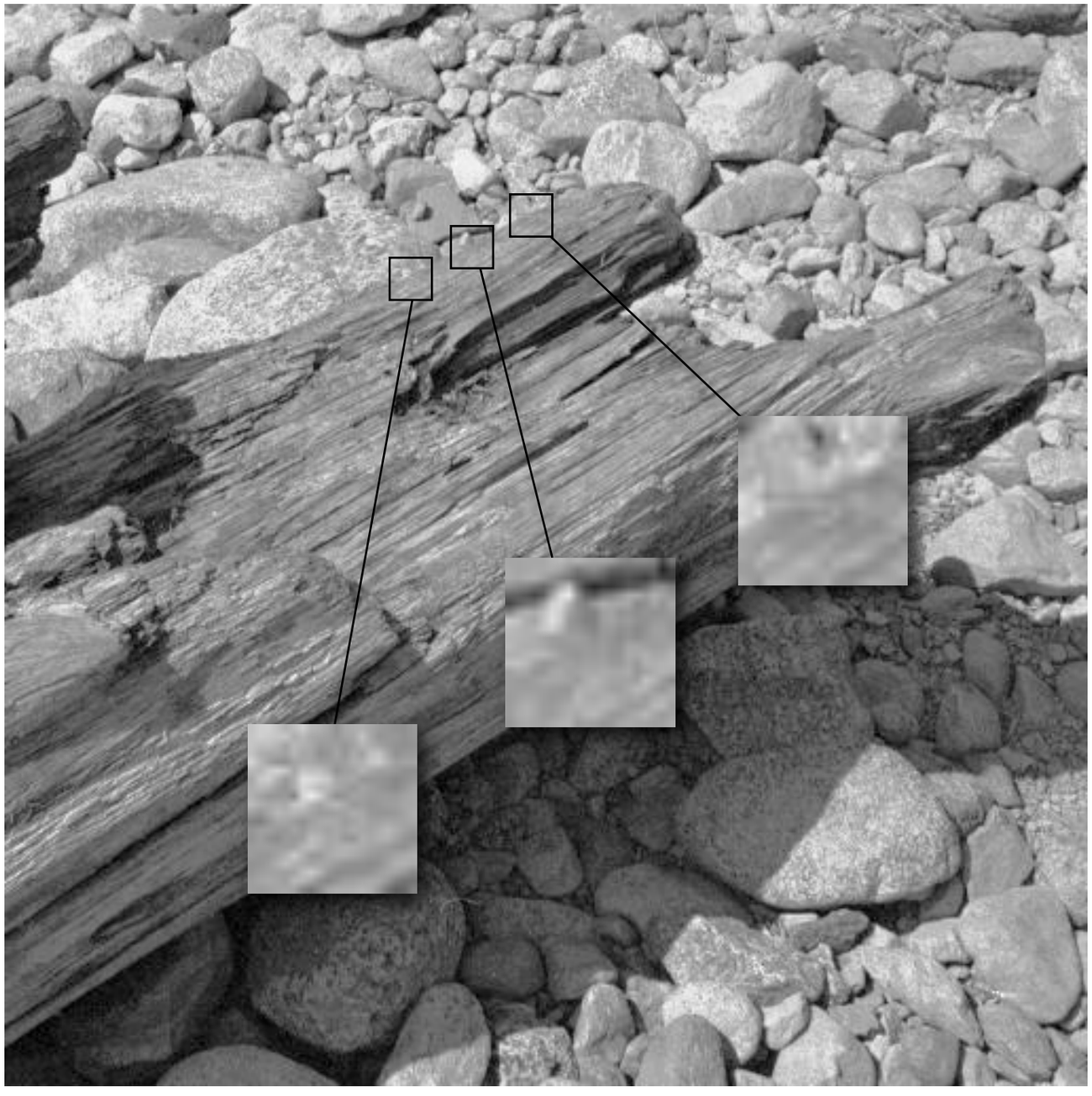
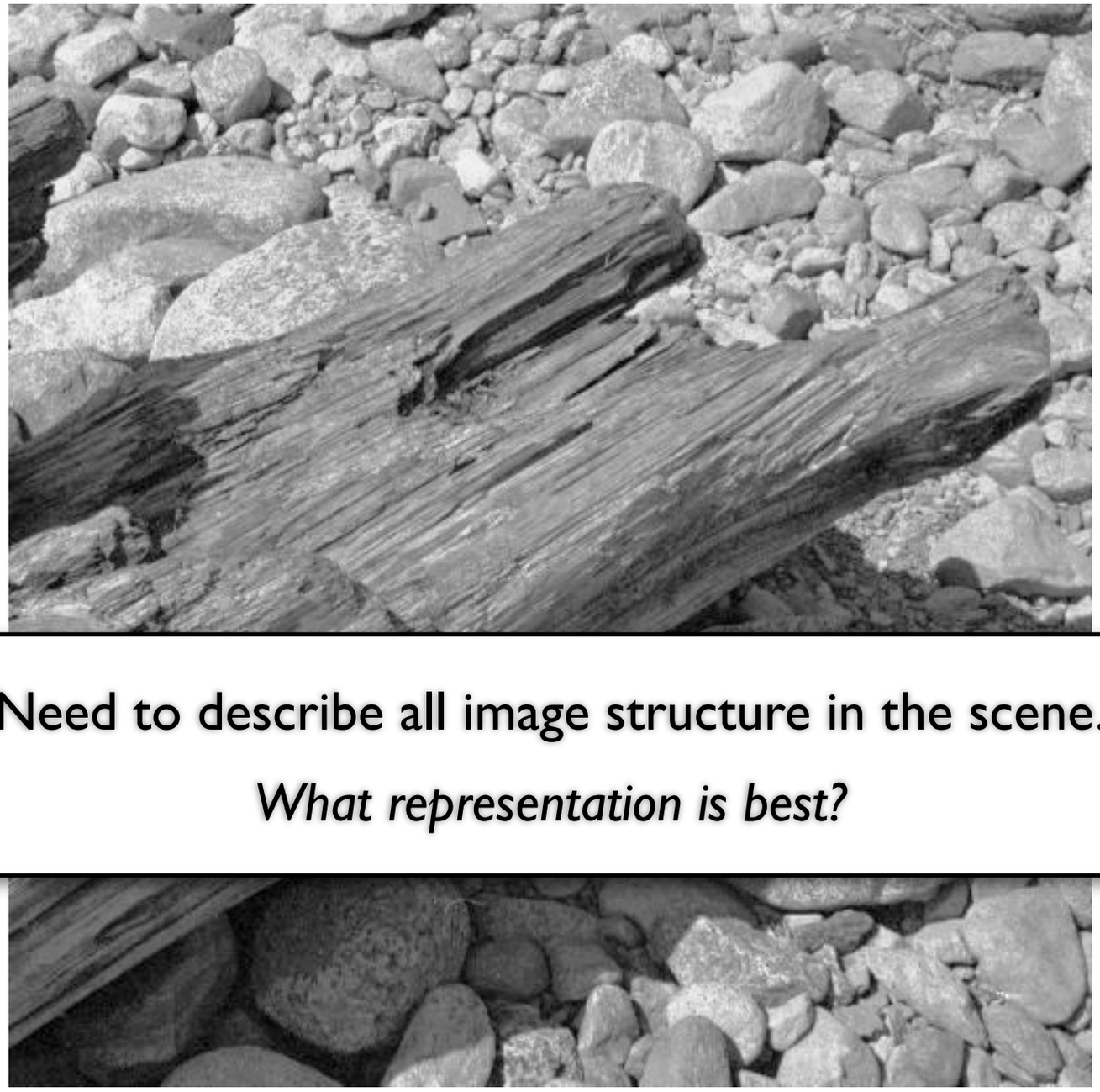


image from Field (1994)

Representing structure in natural images

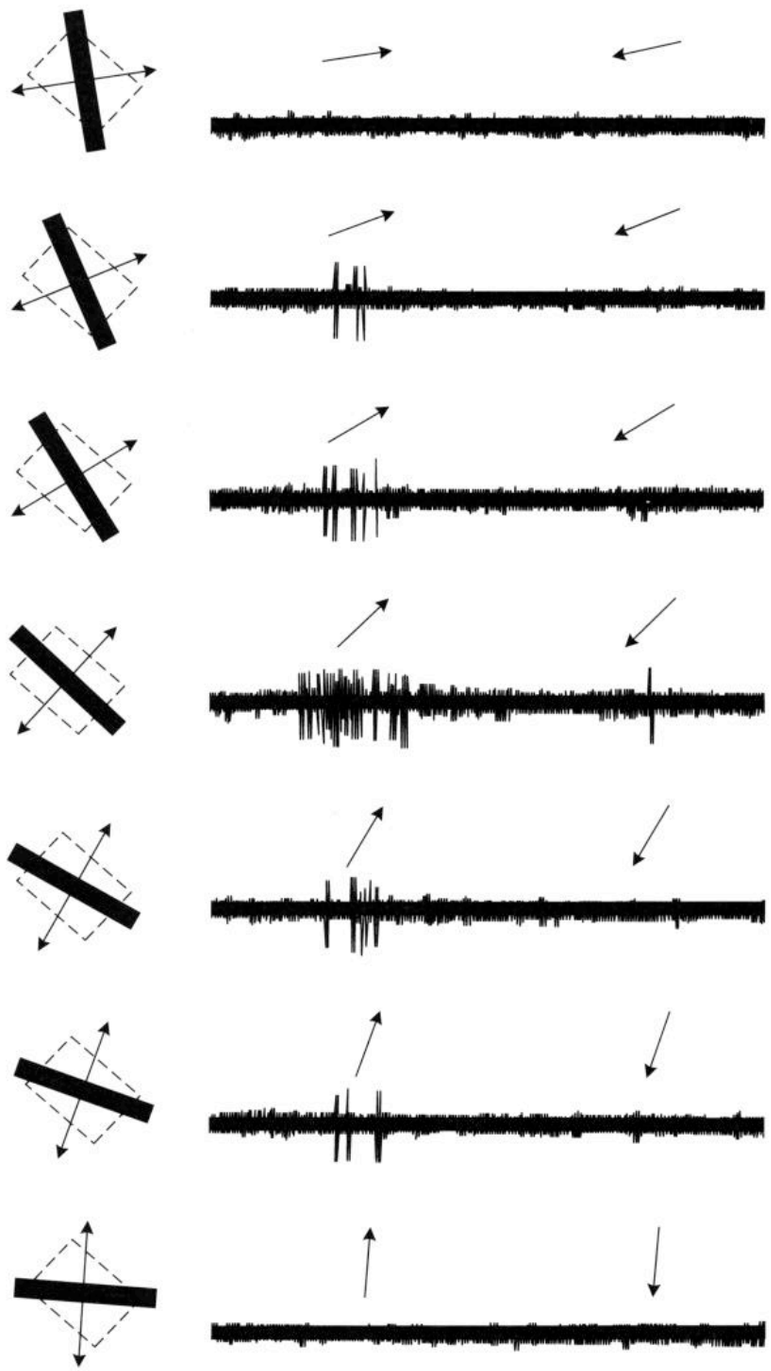


Need to describe all image structure in the scene.

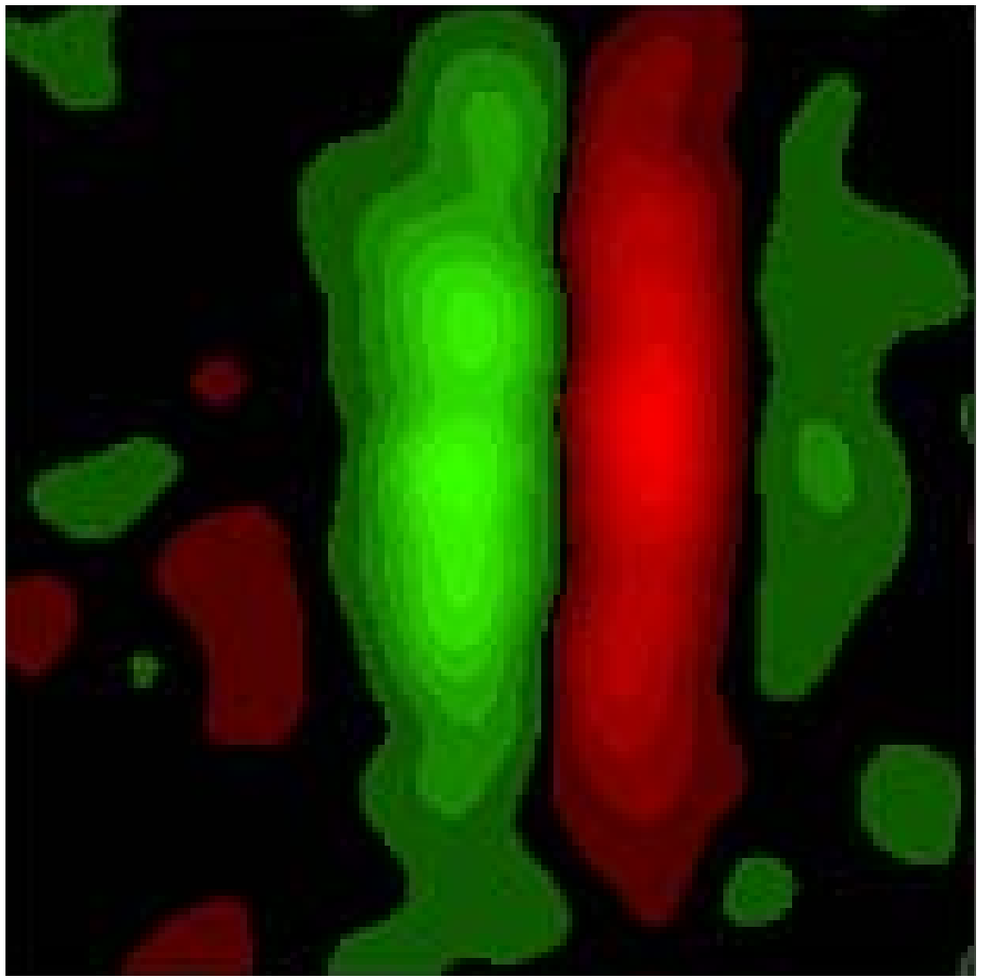
What representation is best?

image from Field (1994)

V1 simple cells



Hubel and Wiesel, 1959



DeAngelis, et al, 1995

Oriented Gabor models of individual simple cells

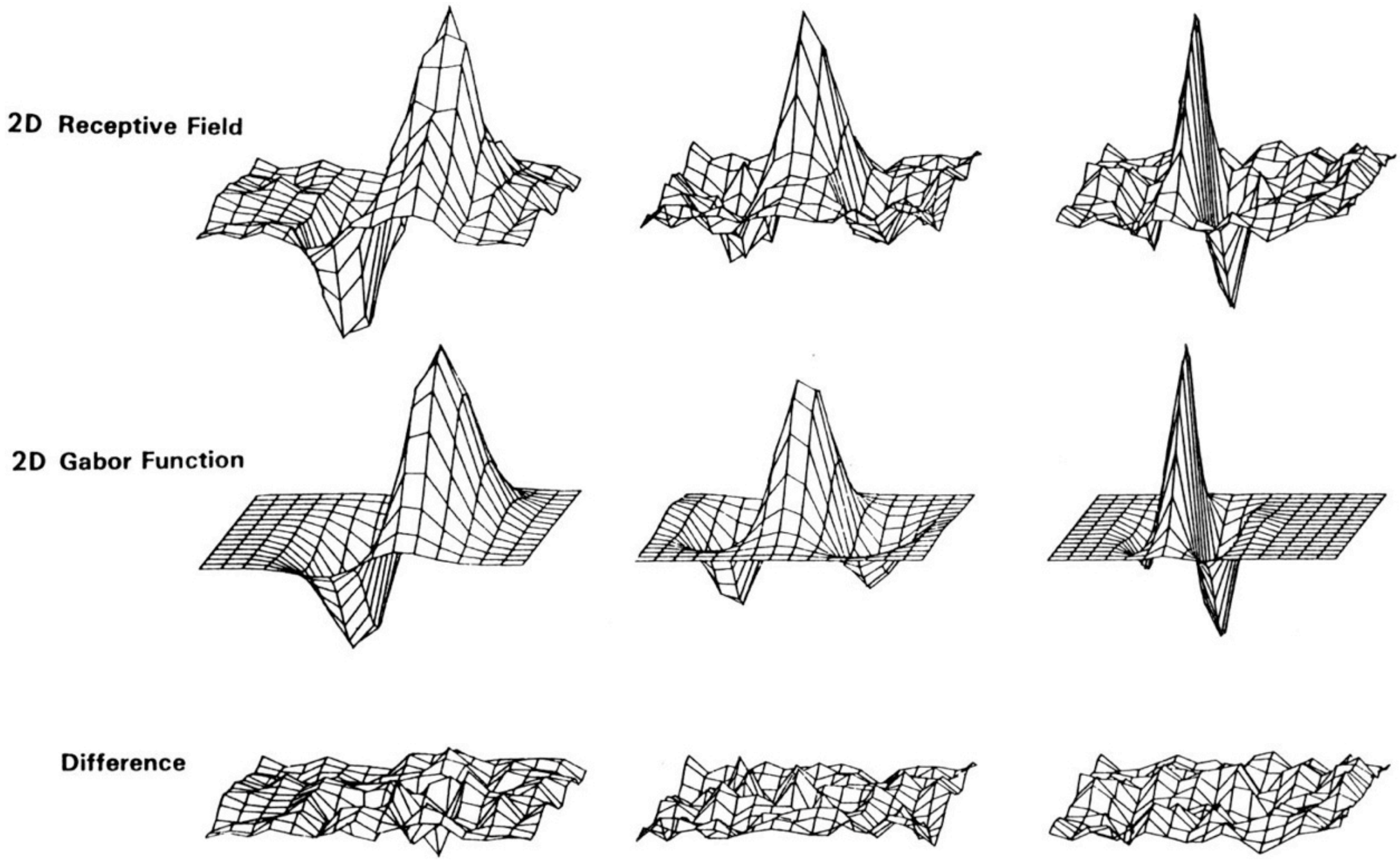
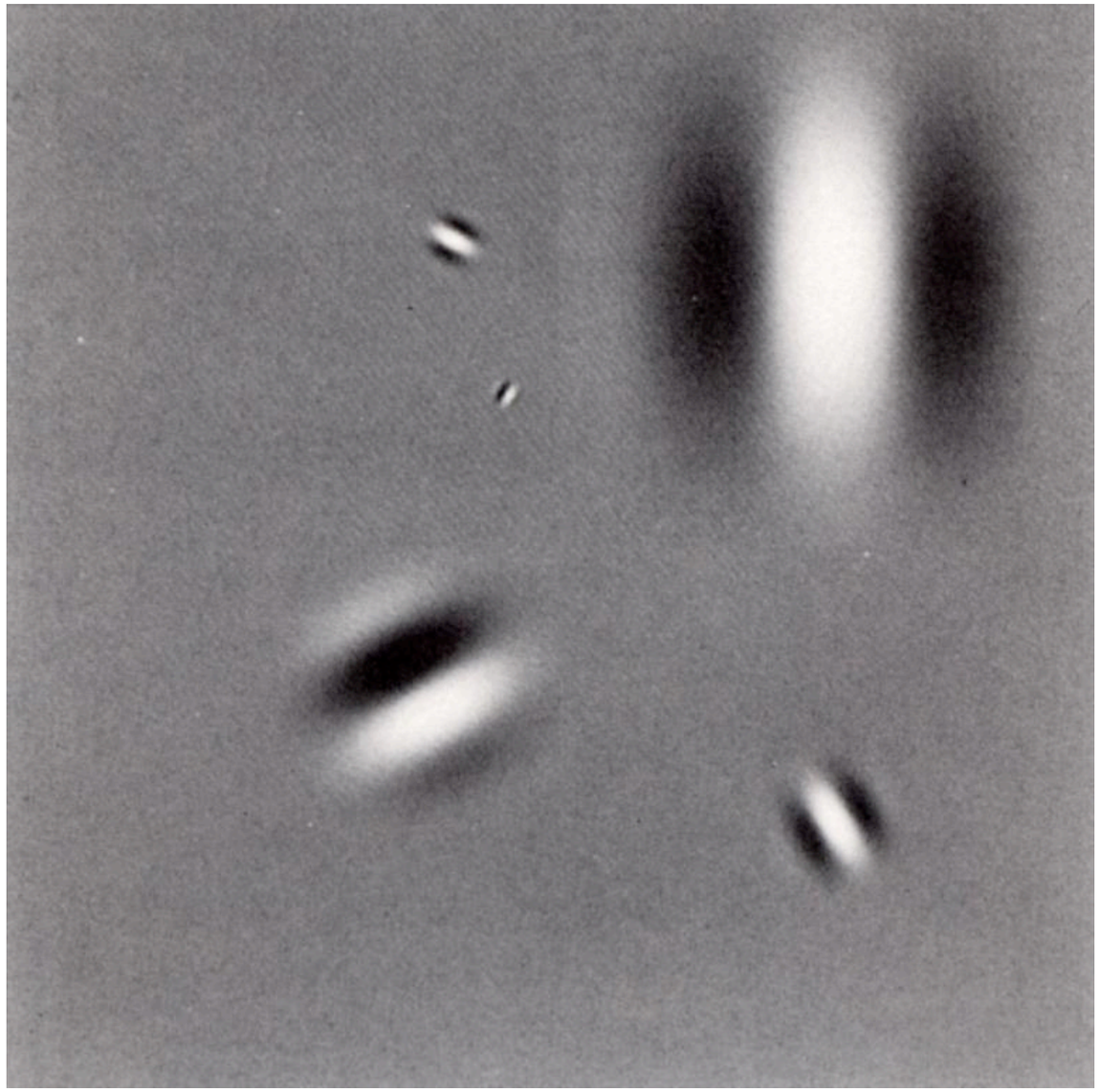


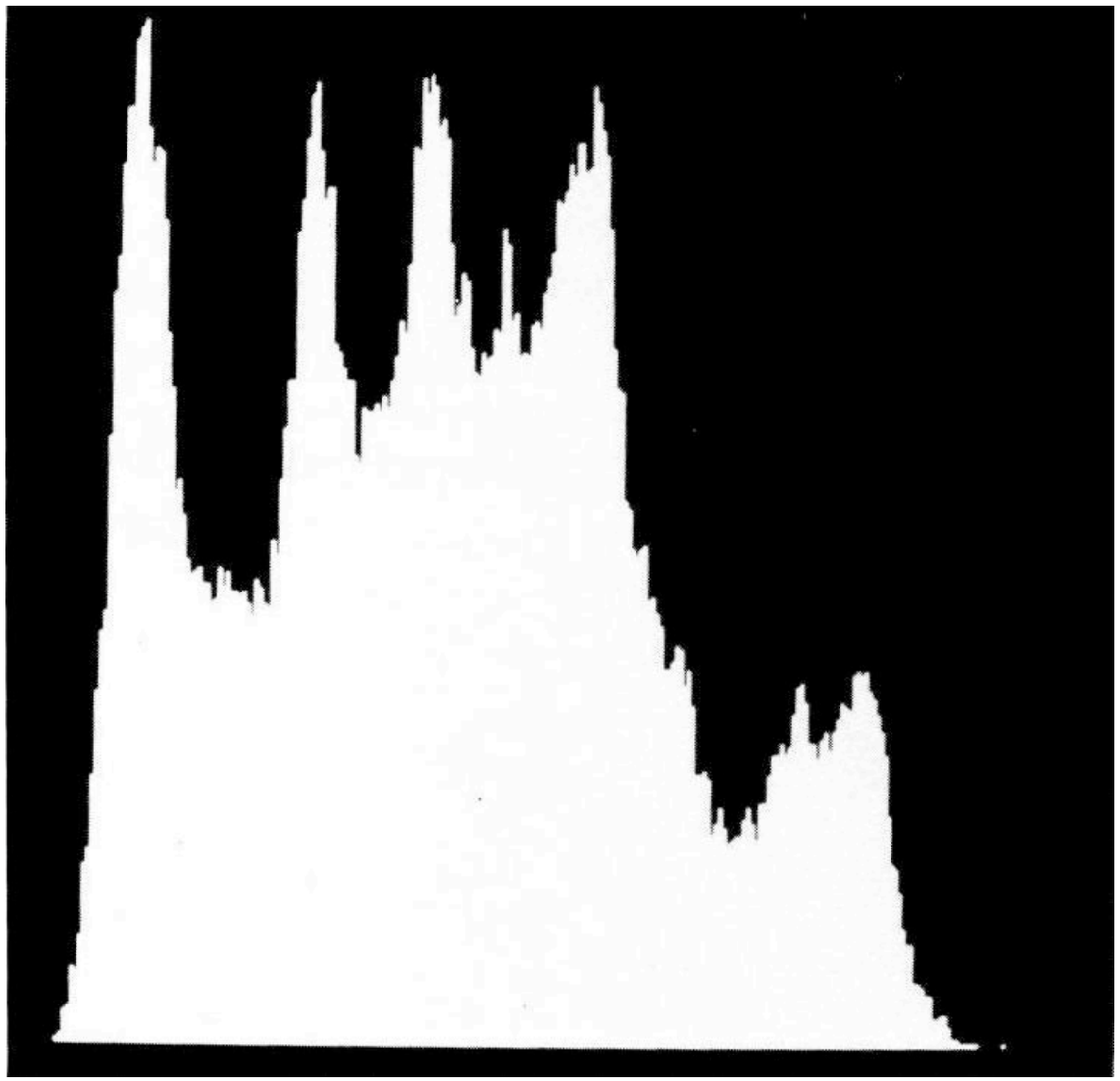
figure from Daugman, 1990; data from Jones and Palmer, 1987

2D Gabor wavelet captures spatial structure

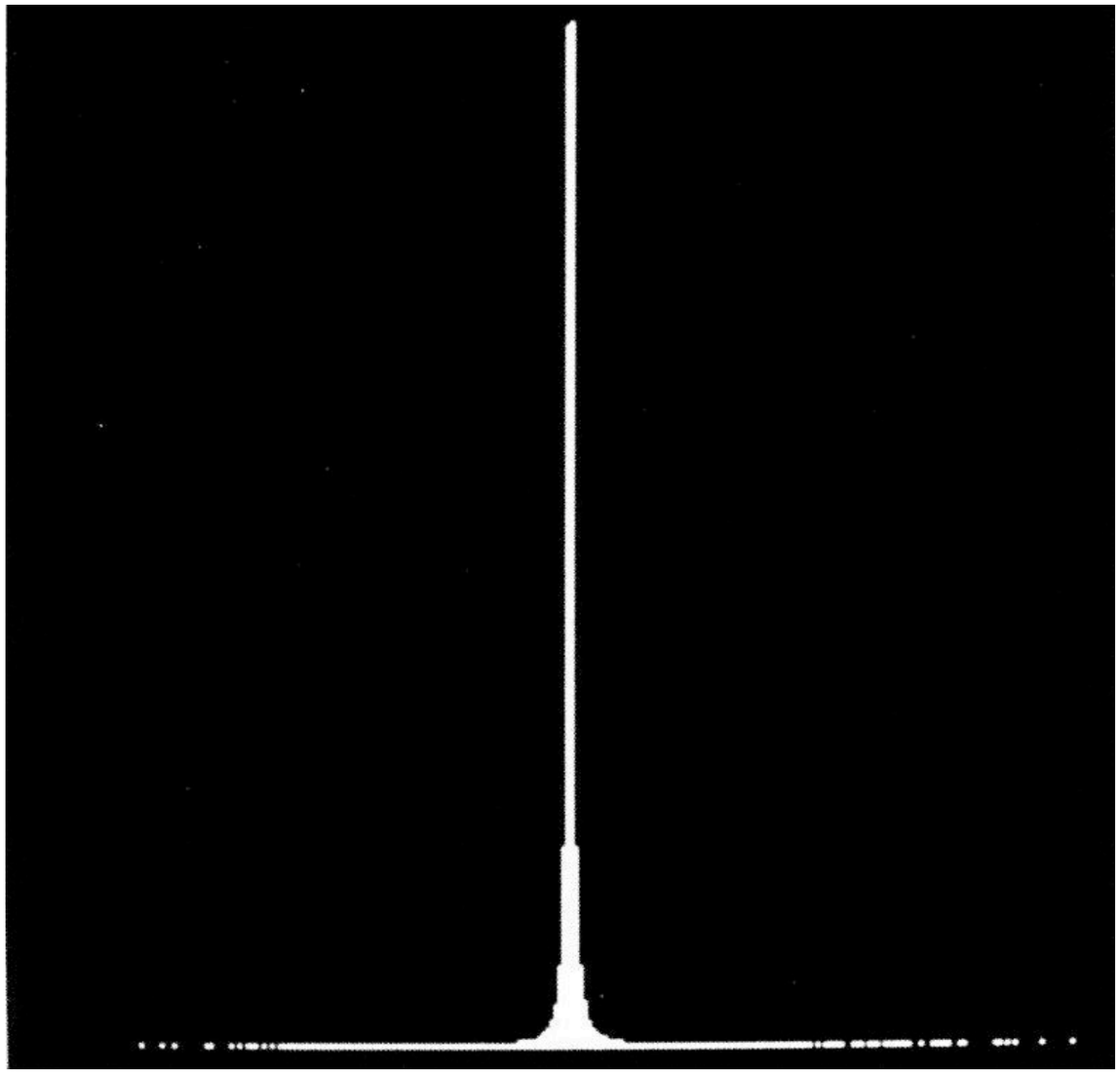


- 2D Gabor functions
- Wavelet basis generated by dilations, translations, and rotations of a single basis function
- Can also control phase and aspect ratio
- (drifting) Gabor functions are what the eye “sees best”

Recoding with Gabor functions

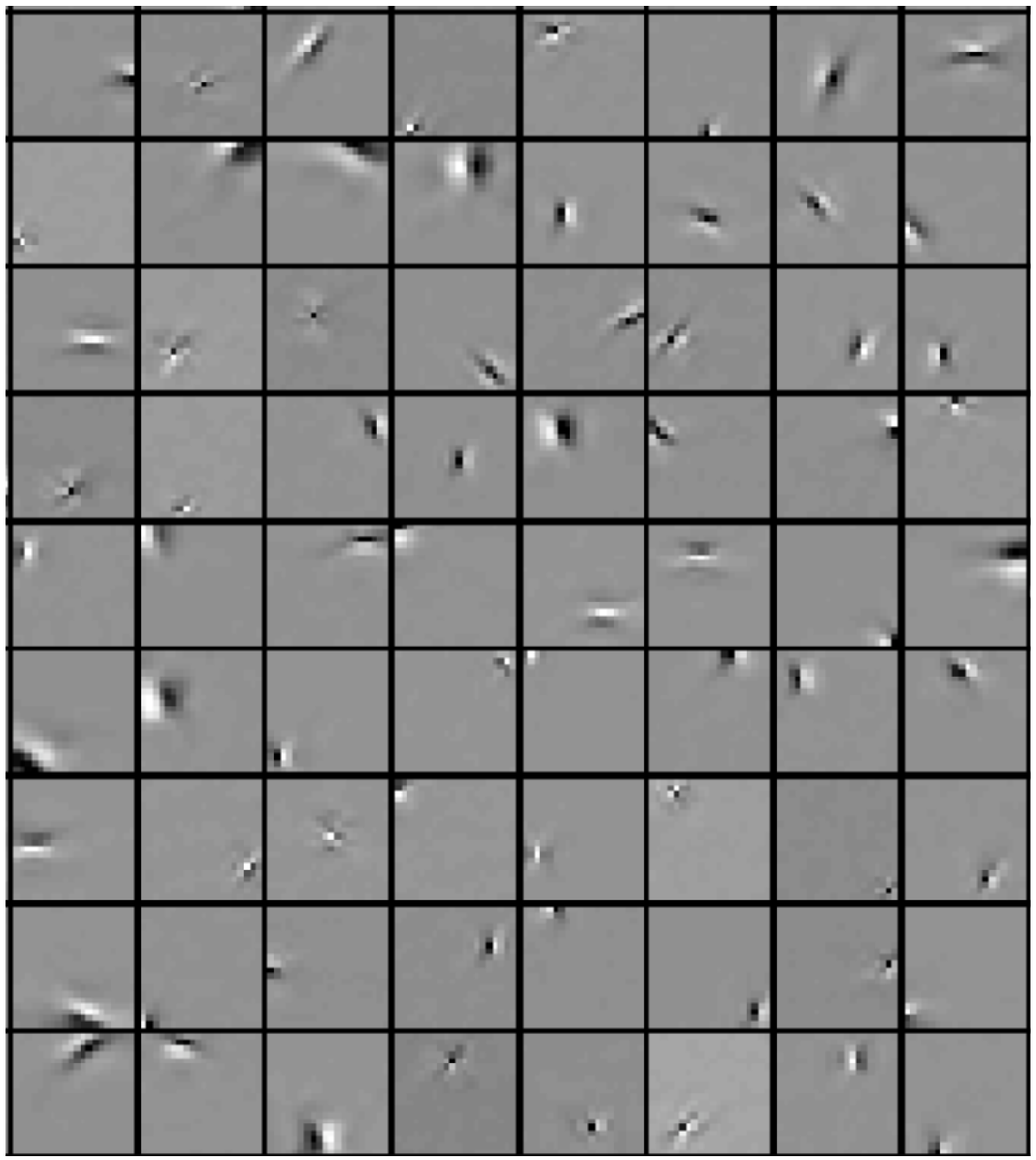


Pixel entropy = 7.57 bits

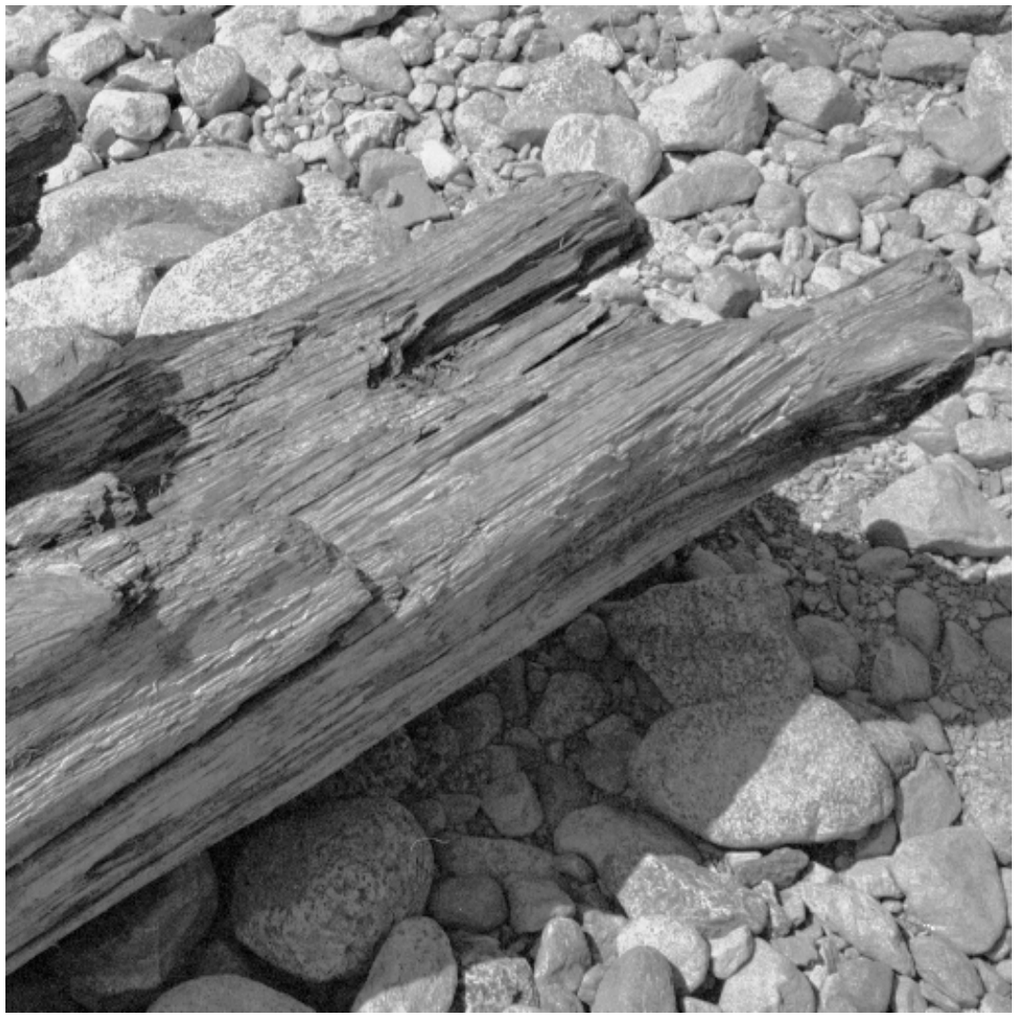


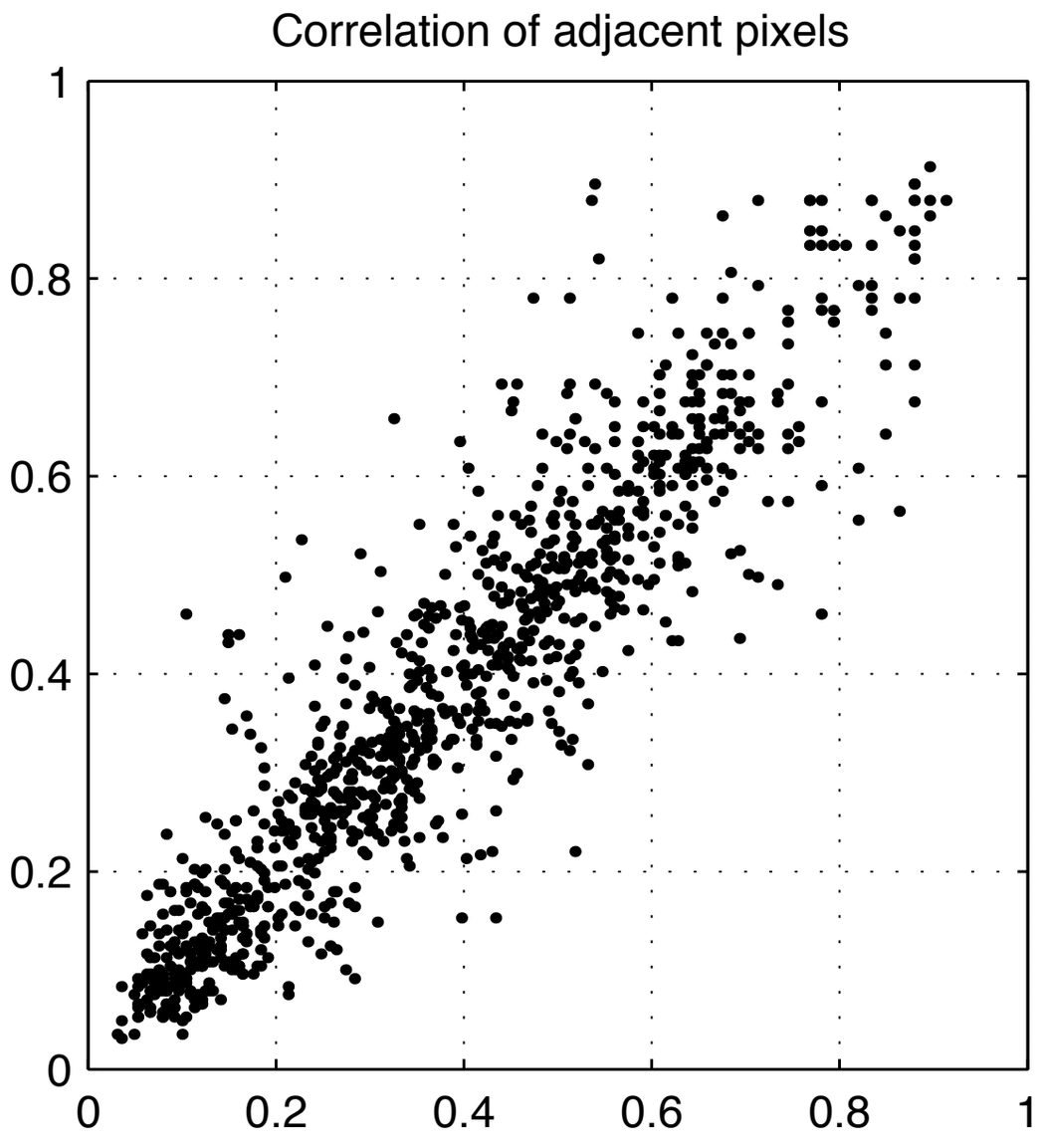
Recoding with 2D Gabor functions
Coefficient entropy = 2.55 bits

How is the V1 population organized?



A general approach to coding: redundancy reduction





Redundancy reduction is equivalent to efficient coding.

Describing signals with a simple statistical model

Principle

Good codes capture the statistical distribution of sensory patterns.

How do we describe the distribution?

- Goal is to encode the data to desired precision

$$\begin{aligned} \mathbf{x} &= \vec{a}_1 s_1 + \vec{a}_2 s_2 + \cdots + \vec{a}_L s_L + \vec{\epsilon} \\ &= \mathbf{A}\mathbf{s} + \boldsymbol{\epsilon} \end{aligned}$$

- Can solve for the coefficients in the no noise case

$$\hat{\mathbf{s}} = \mathbf{A}^{-1}\mathbf{x}$$

An algorithm for deriving efficient linear codes: ICA

Learning objective:

maximize coding efficiency

⇒ maximize P(**x**|**A**) over $\mathbf{A}$.

Probability of the pattern ensemble is:

$$P(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N|\mathbf{A}) = \prod_k P(\mathbf{x}_k|\mathbf{A})$$

To obtain P(**x**|**A**) marginalize over s:

$$\begin{aligned} P(\mathbf{x}|\mathbf{A}) &= \int d\mathbf{s}\, P(\mathbf{x}|\mathbf{A}, \mathbf{s}) P(\mathbf{s}) \\ &= \frac{P(\mathbf{s})}{|\det \mathbf{A}|} \end{aligned}$$

Using independent component analysis (ICA) to optimize **A**:

$$\begin{aligned} \Delta\mathbf{A} &\propto \mathbf{A}\mathbf{A}^T \frac{\partial}{\partial \mathbf{A}} \log P(\mathbf{x}|\mathbf{A}) \\ &= -\mathbf{A}(\mathbf{z}\mathbf{s}^T - \mathbf{I}) \end{aligned}$$

where z = (log P(s))'.

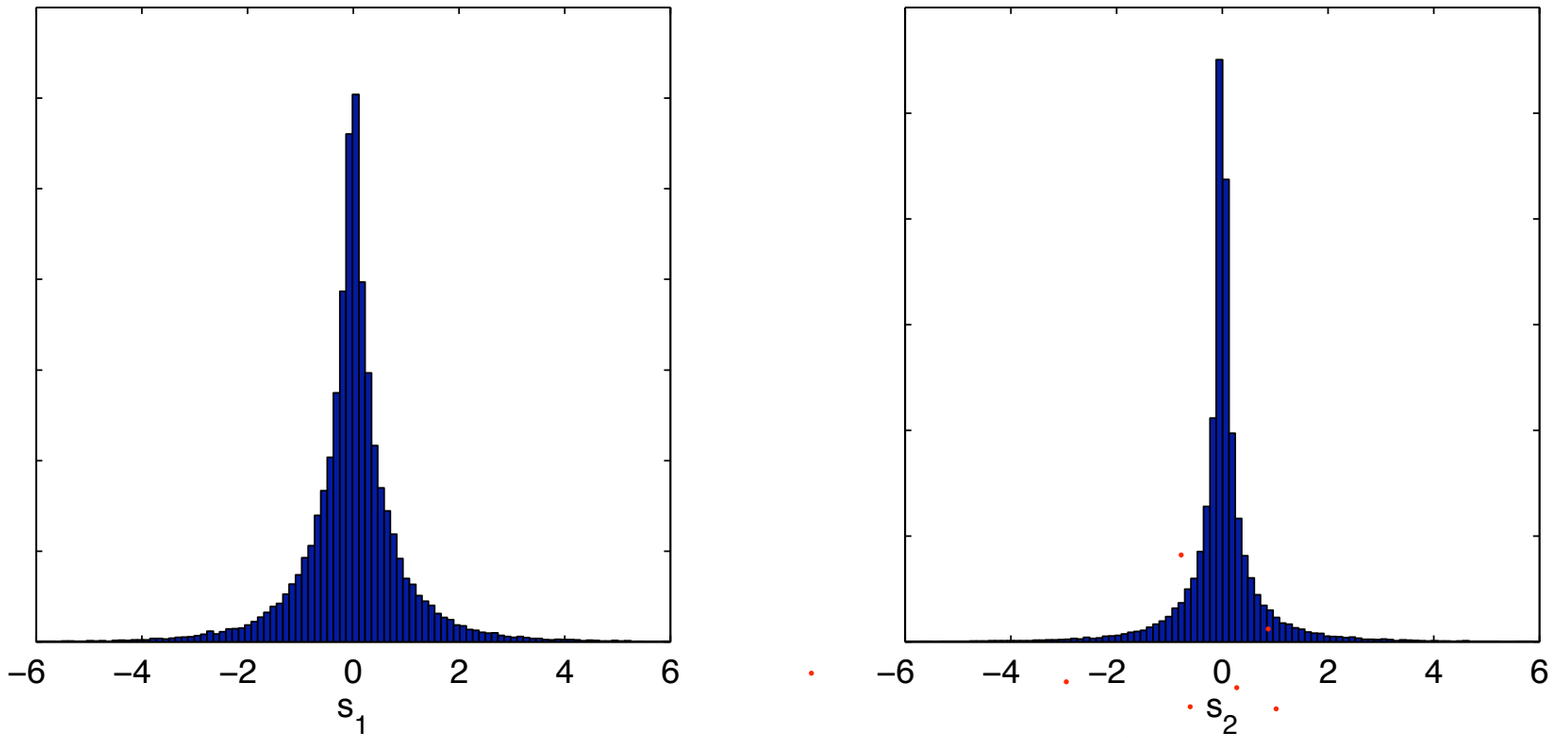
This learning rule:

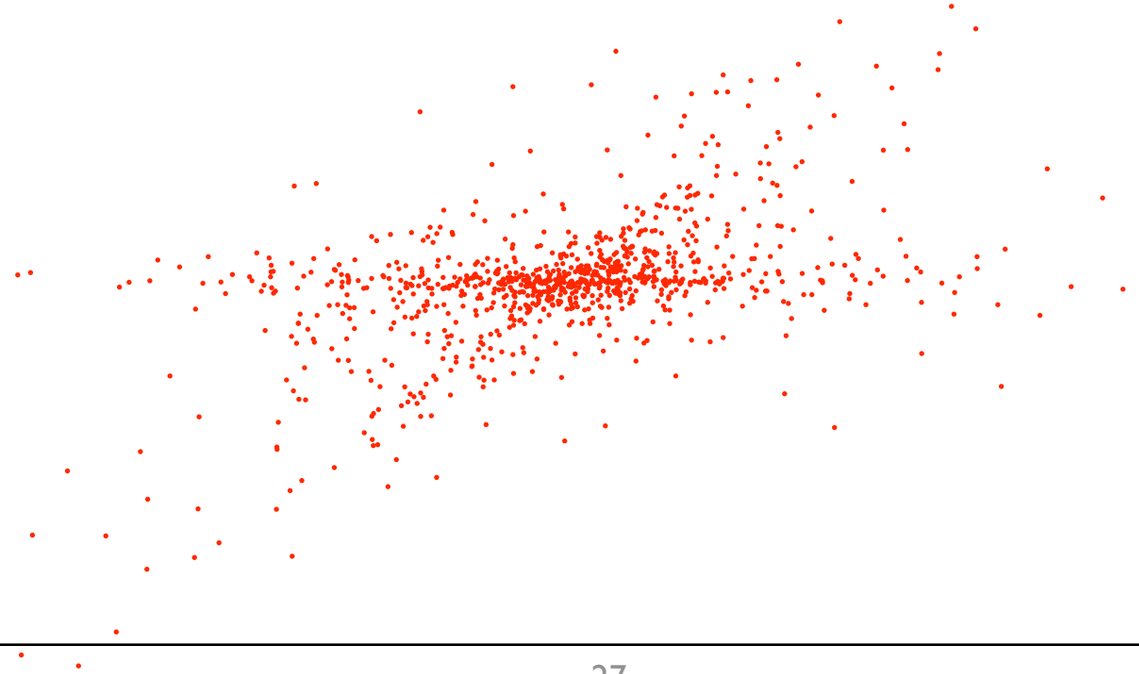
- learns the features that capture the most structure
- optimizes the efficiency of the code

What should we use for P(**s**)?

Modeling Non-Gaussian distributions

- Typical coeff. distributions of natural signals are *non-Gaussian*.





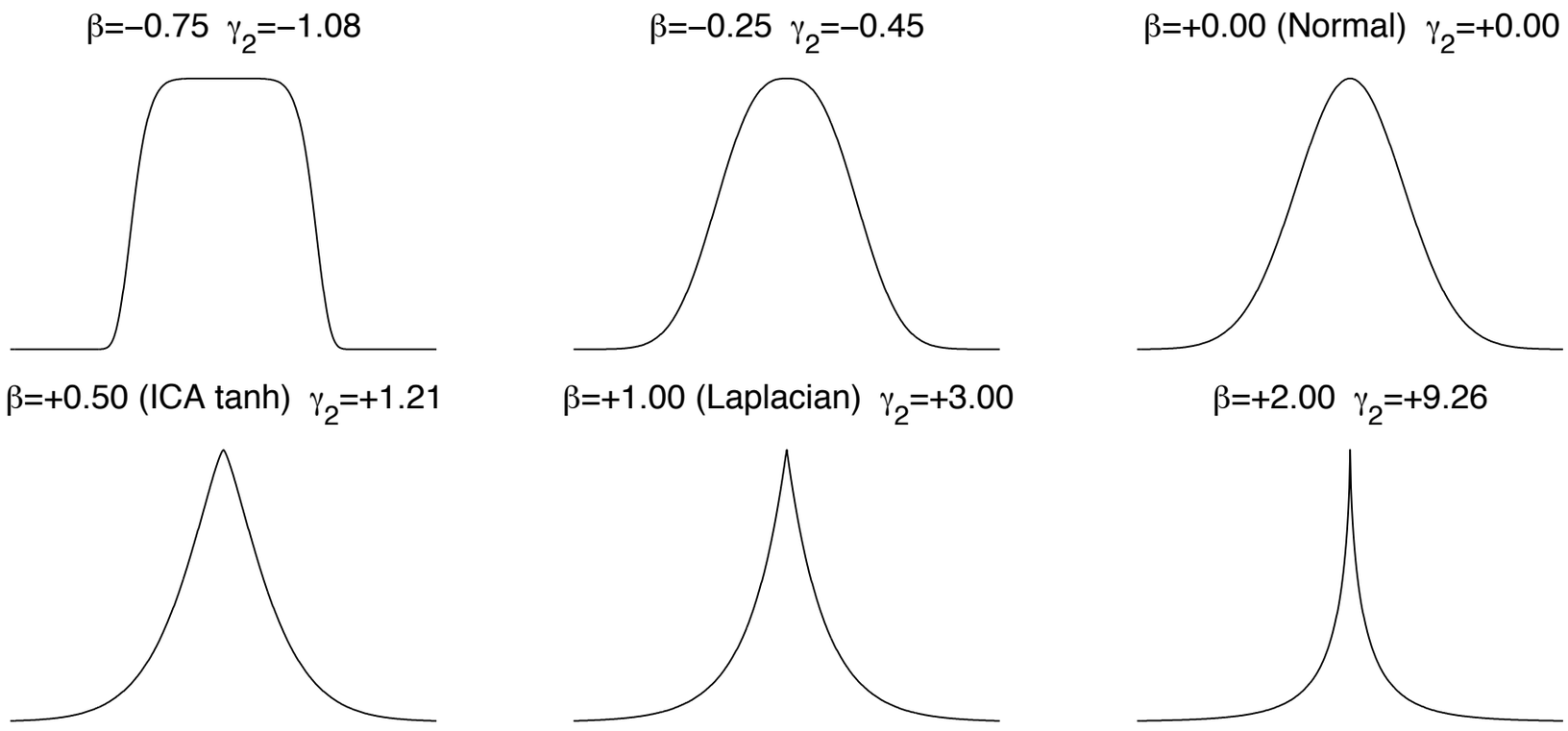
The generalized Gaussian distribution

$$P(x|q) \propto \exp(-\frac{1}{2}|x|^q)$$

- Or equivalently, and exponential power distribution (Box and Tiao, 1973):

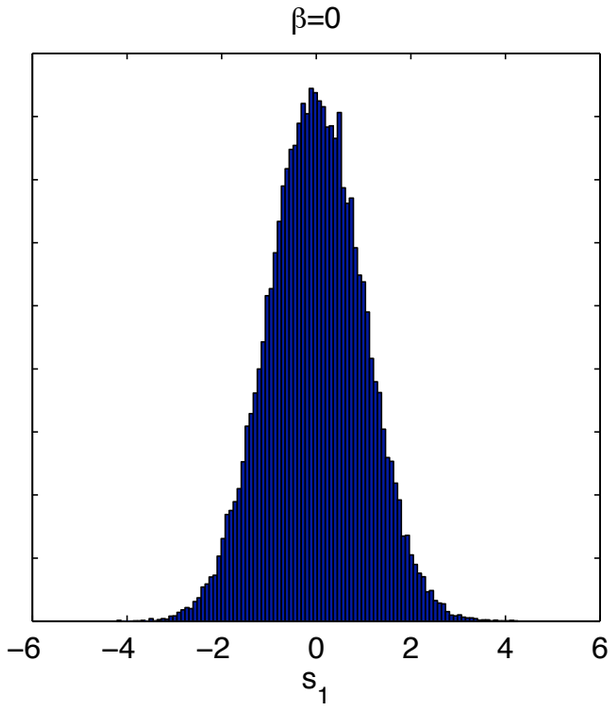
$$P(x|\mu,\sigma,\beta) = \frac{\omega(\beta)}{\sigma} \exp\left[-c(\beta)\left|\frac{x-\mu}{\sigma}\right|^{2/(1+\beta)}\right]$$

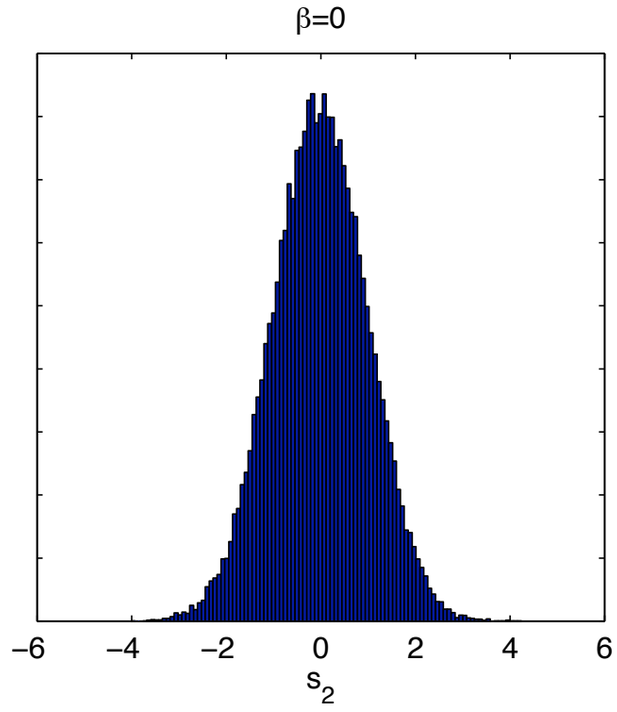
- β varies monotonically with the kurtosis, γ_2:

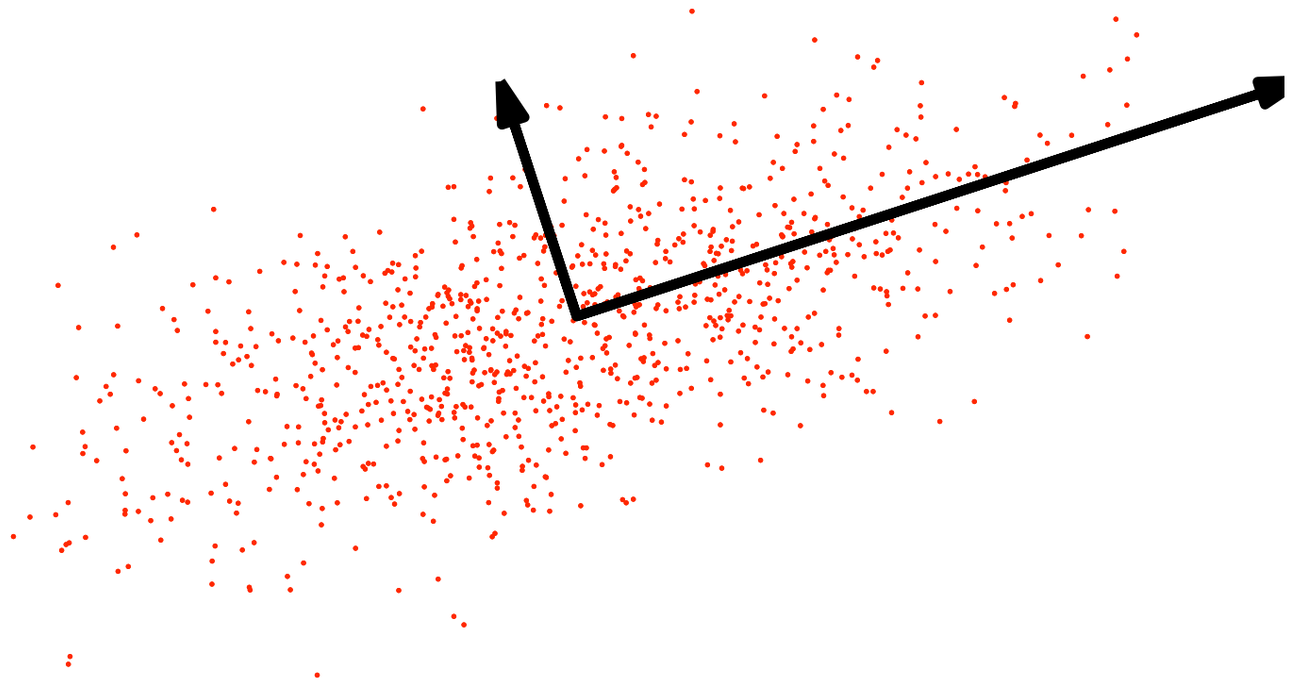


Modeling Gaussian distributions with PCA

- Principal component analysis (PCA) describes the principal axes of variation in the data distribution.
- This is equivalent to fitting the data with a multivariate Gaussian.

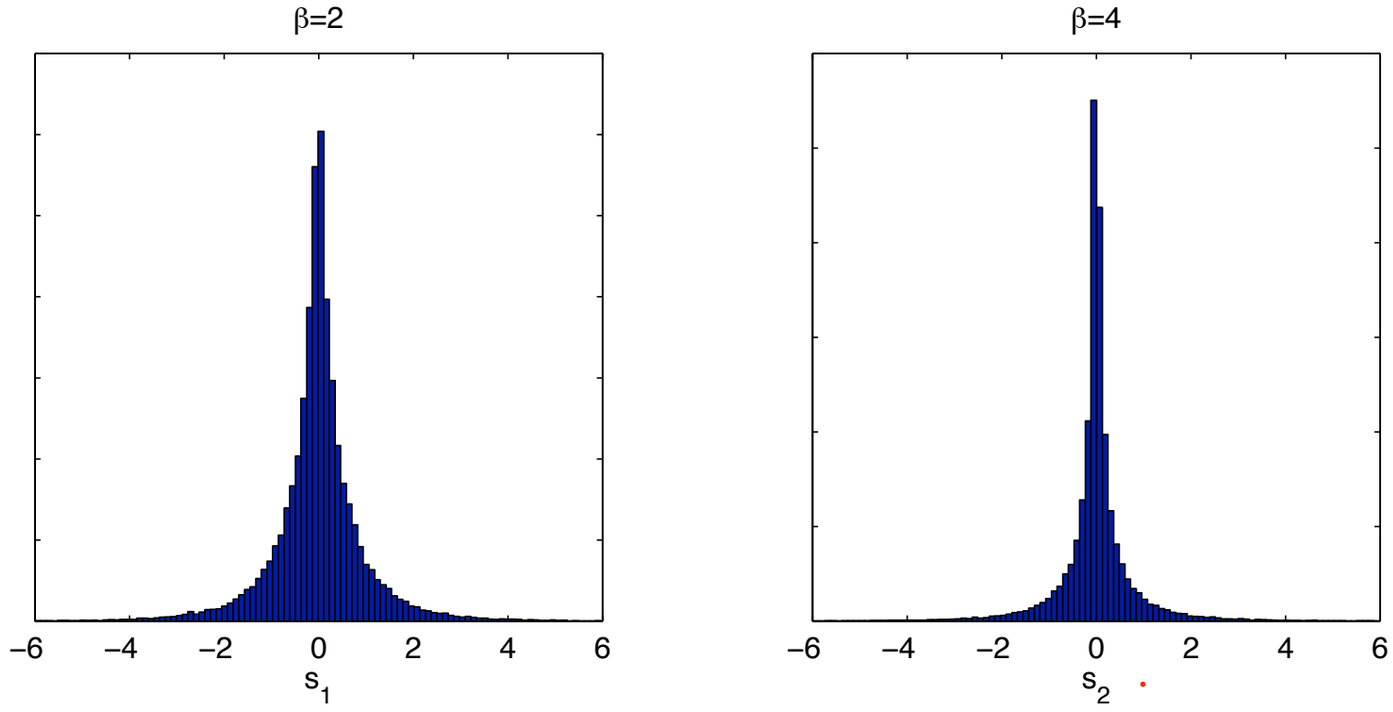




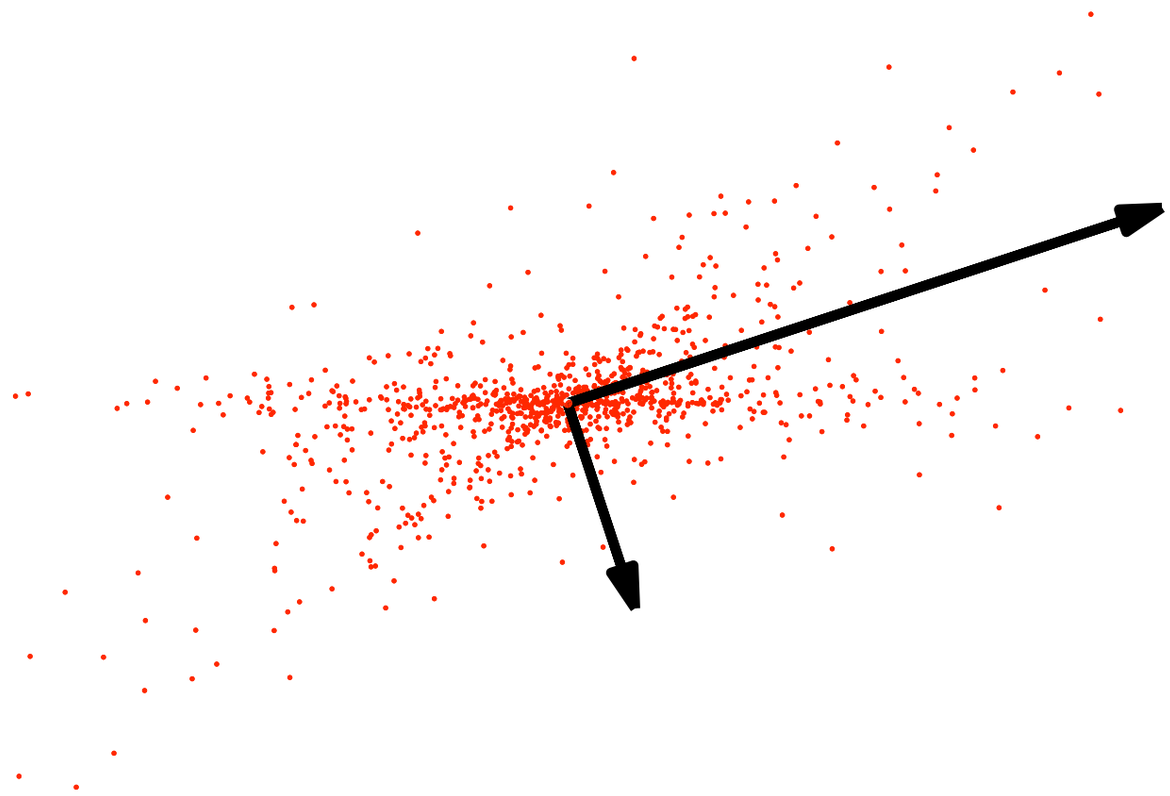


Modeling non-Gaussian distributions

- What about non-Gaussian marginals?

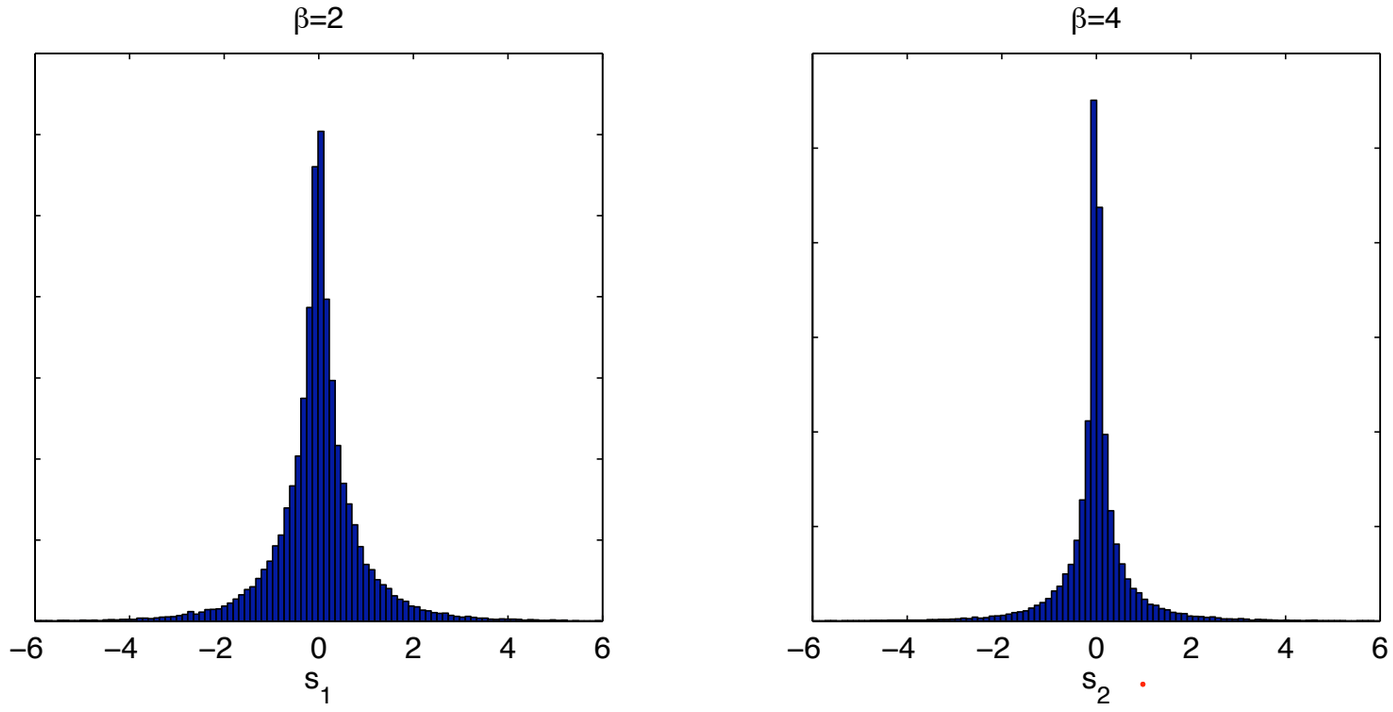


- How would this distribution be modeled by PCA?



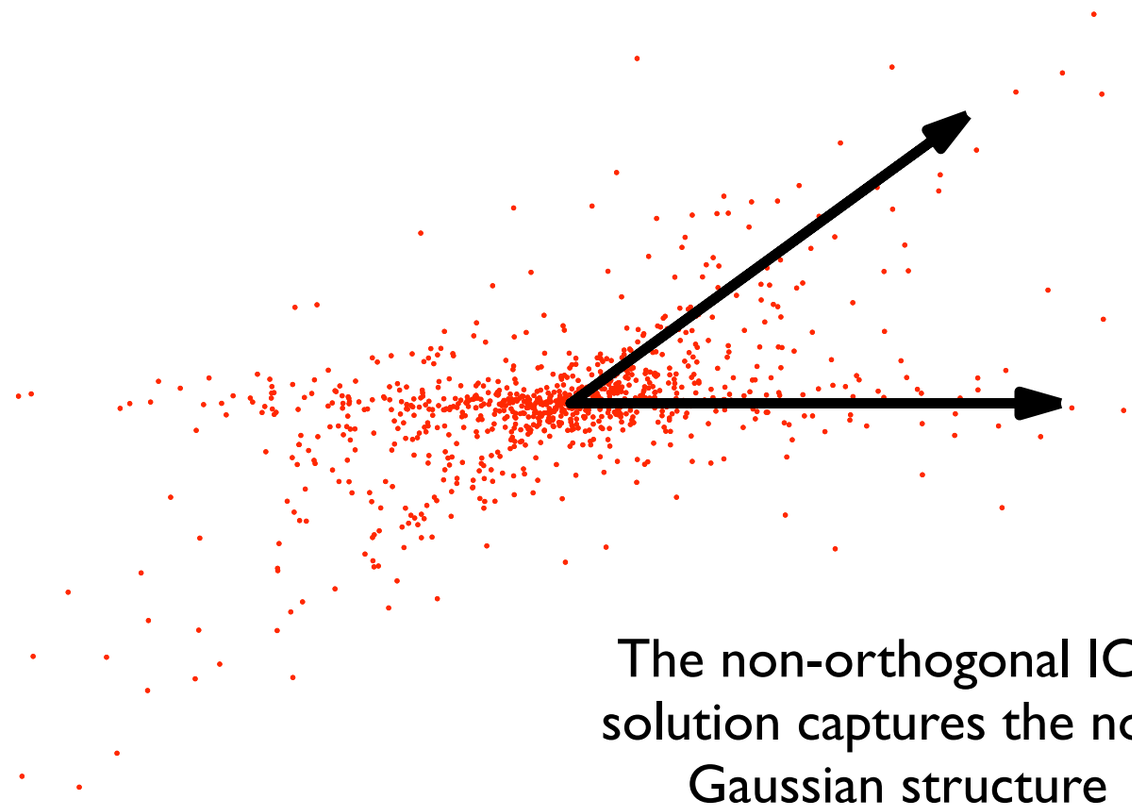
Modeling non-Gaussian distributions

- What about non-Gaussian marginals?



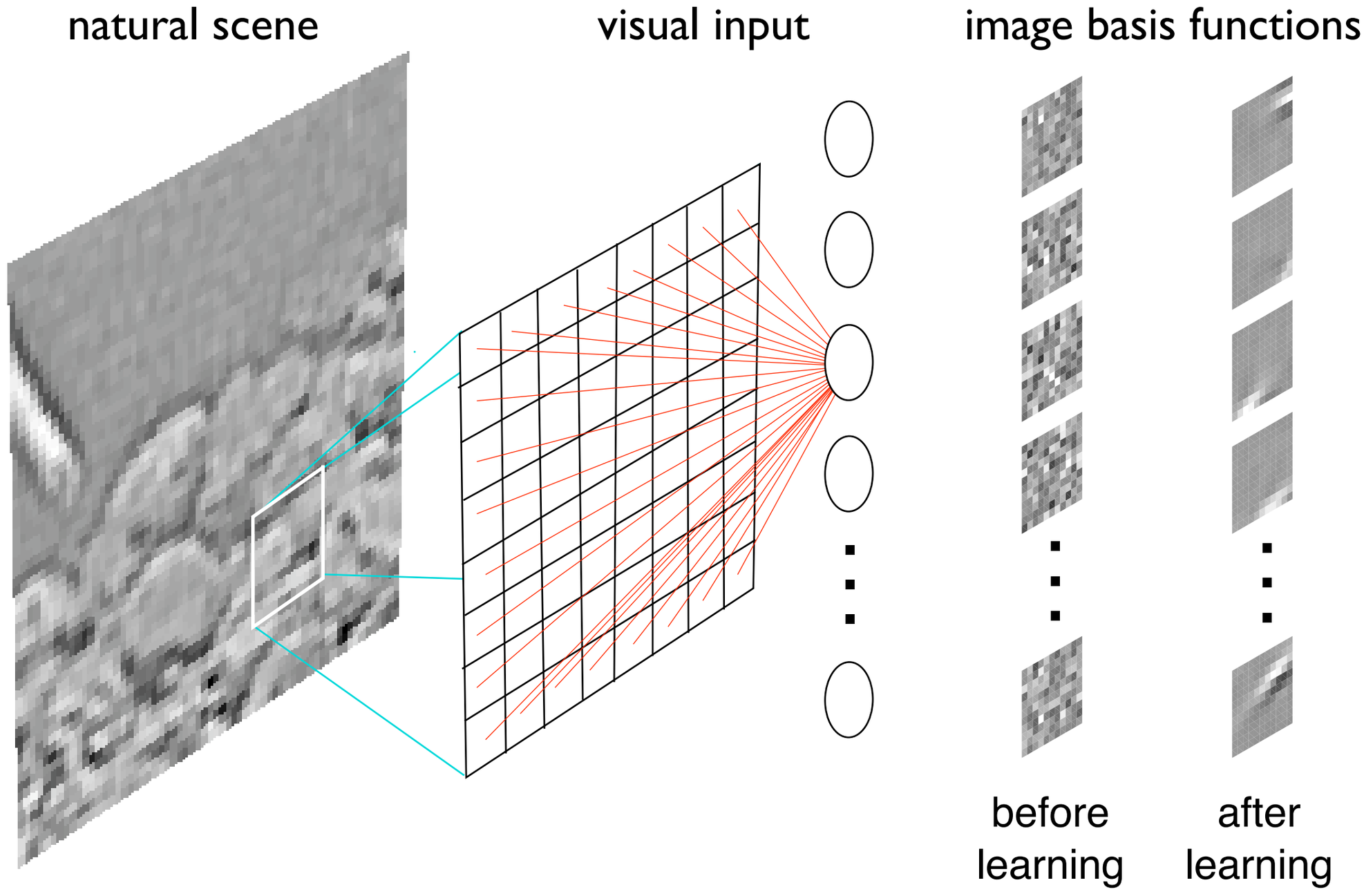
- How would this distribution be modeled by PCA?

- How should the distribution be described?



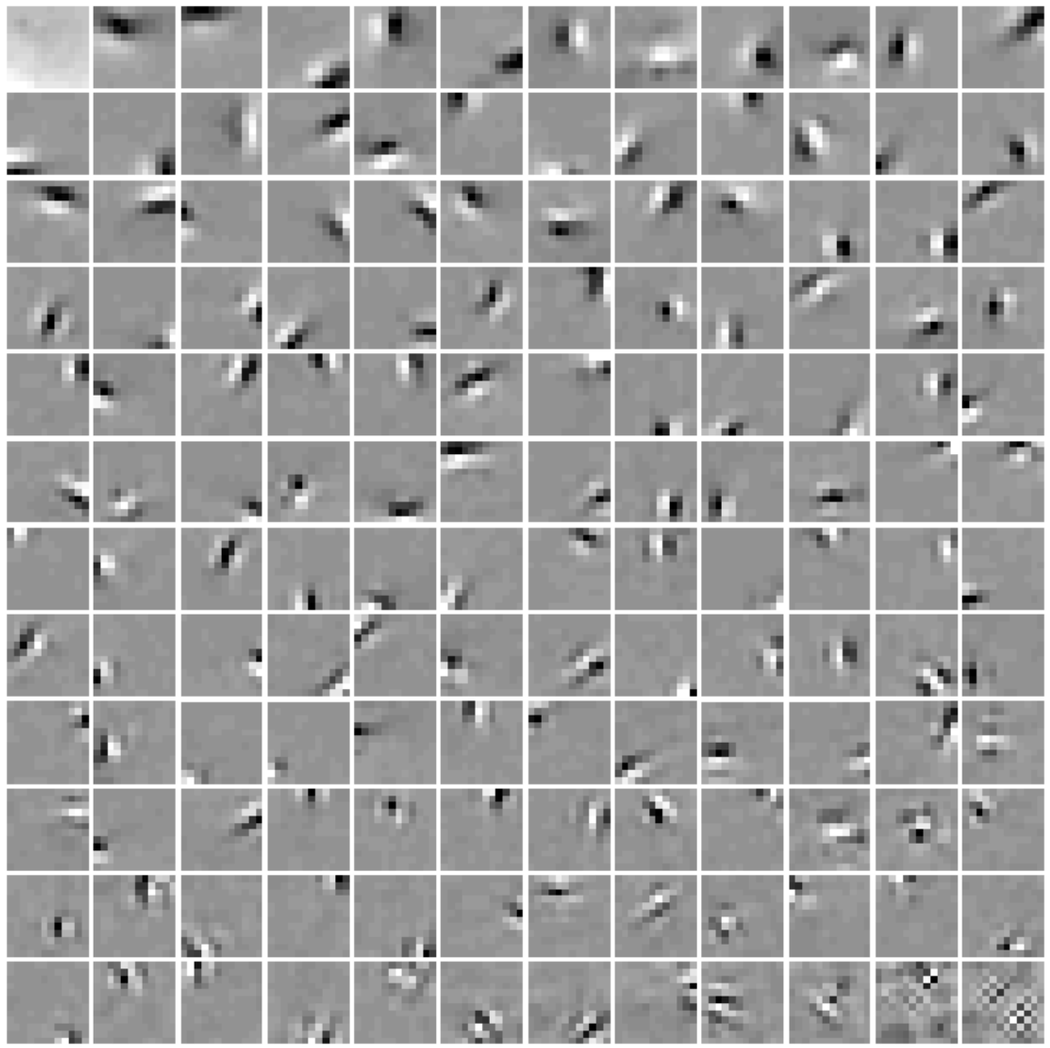
The non-orthogonal ICA solution captures the non-Gaussian structure

Efficient coding of natural images: Olshausen and Field, 1996

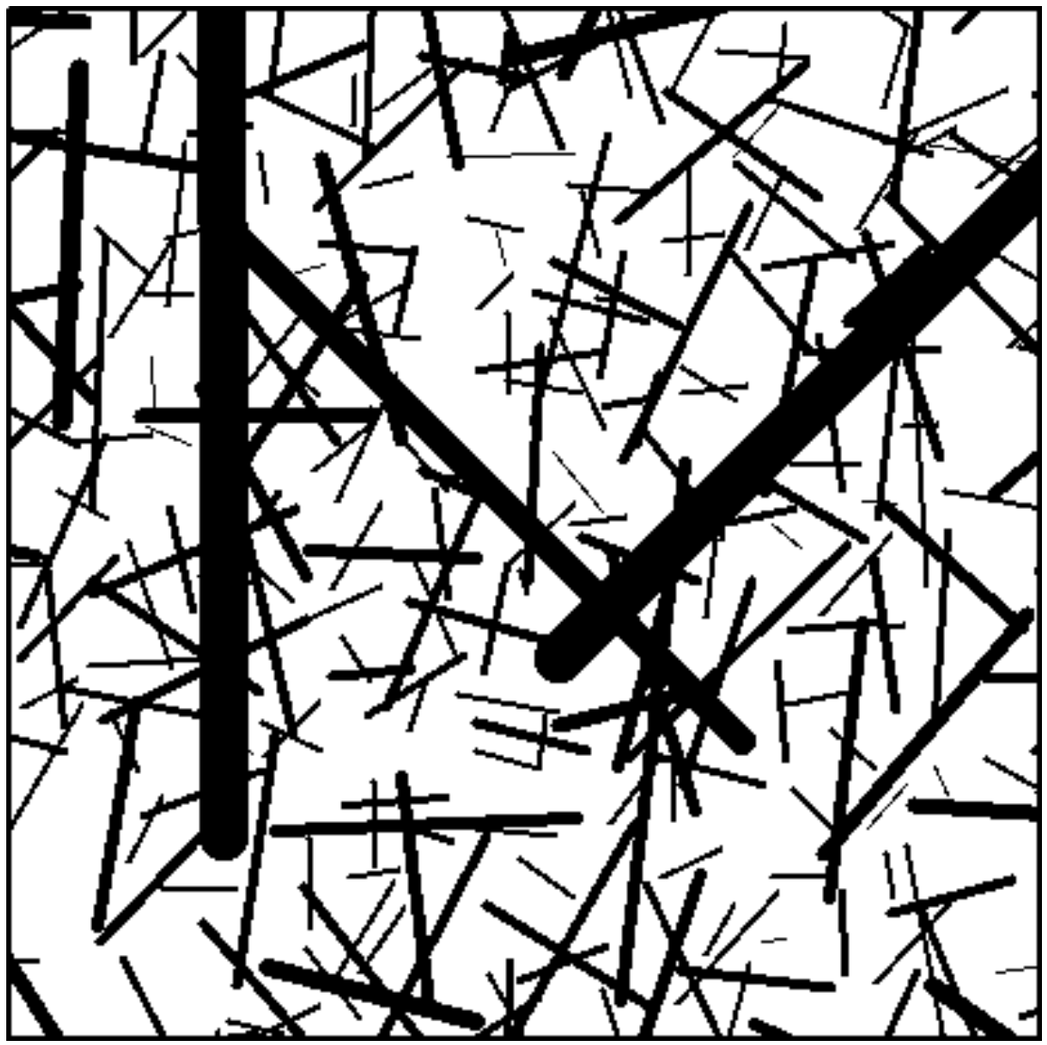


Network weights are adapted to maximize coding efficiency:
minimizes redundancy and maximizes the independence of the outputs

Model predicts local and global receptive field properties



Learned basis for natural images

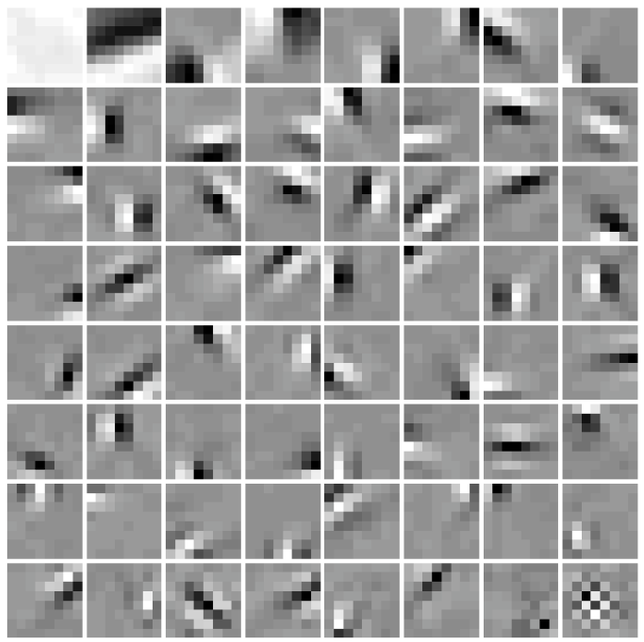


Overlaid basis function properties

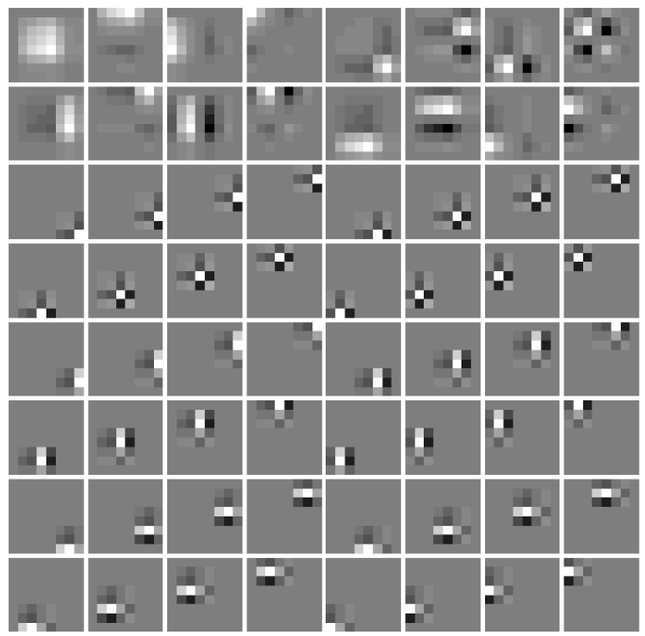
from Lewicki and Olshausen, 1999

Algorithm selects best of many possible sensory codes

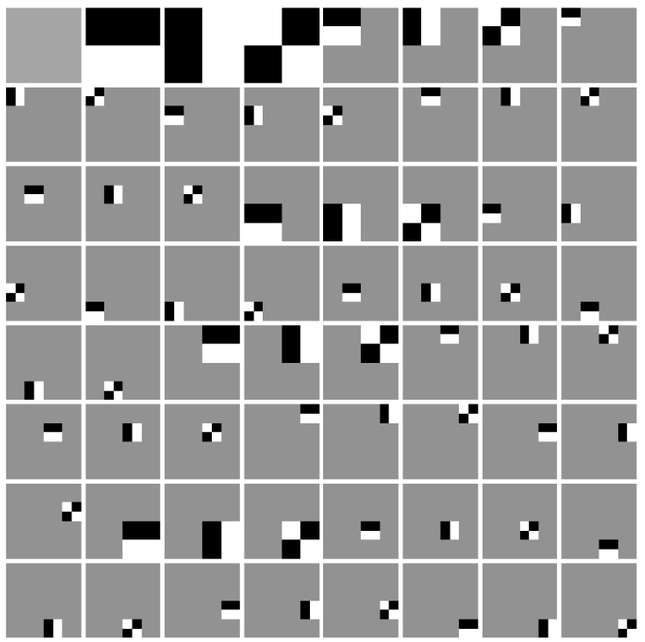
Learned



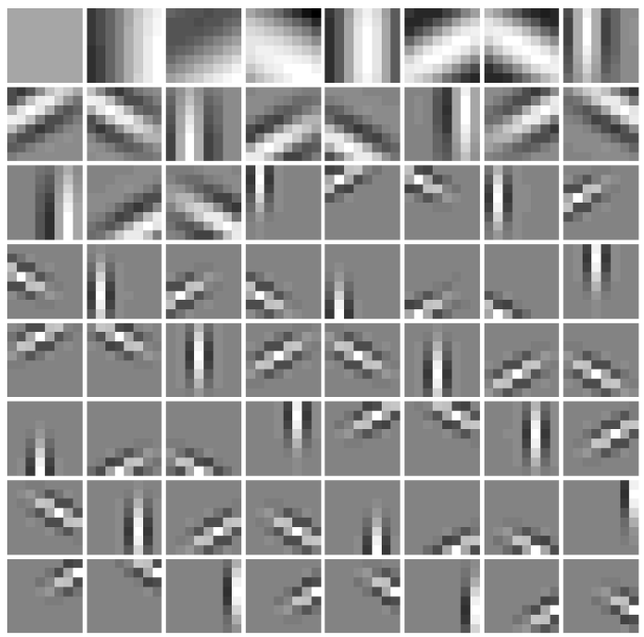
Wavelet



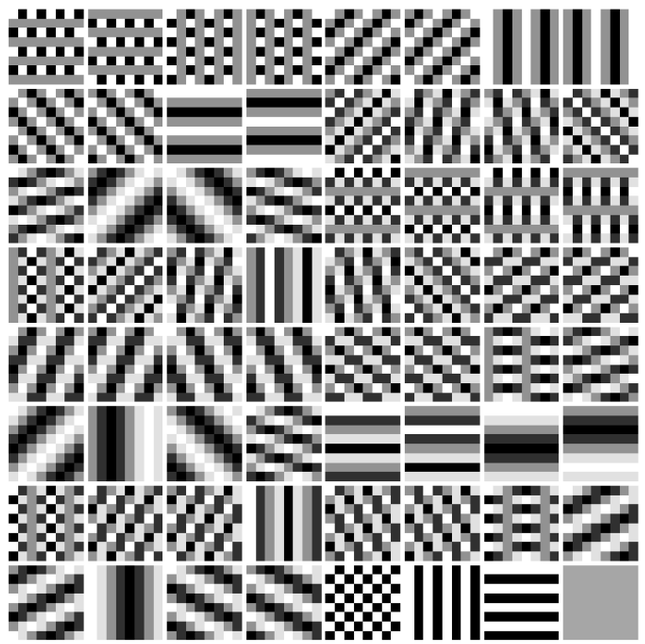
Haar



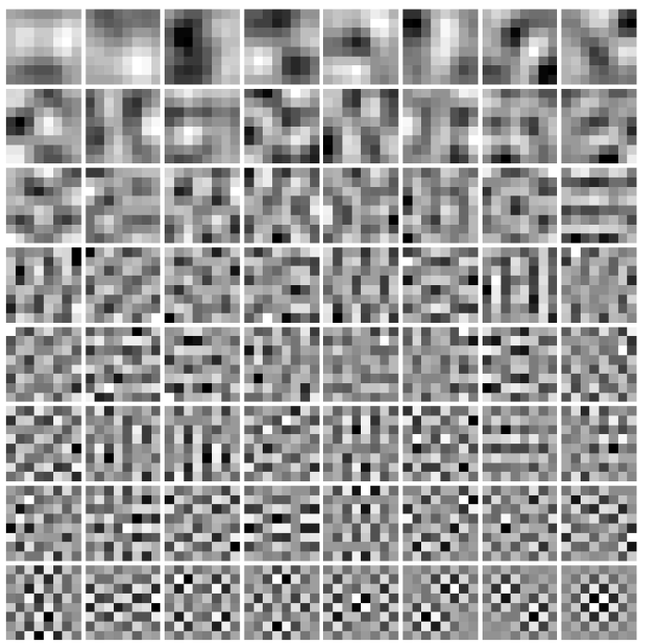
Gabor



Fourier



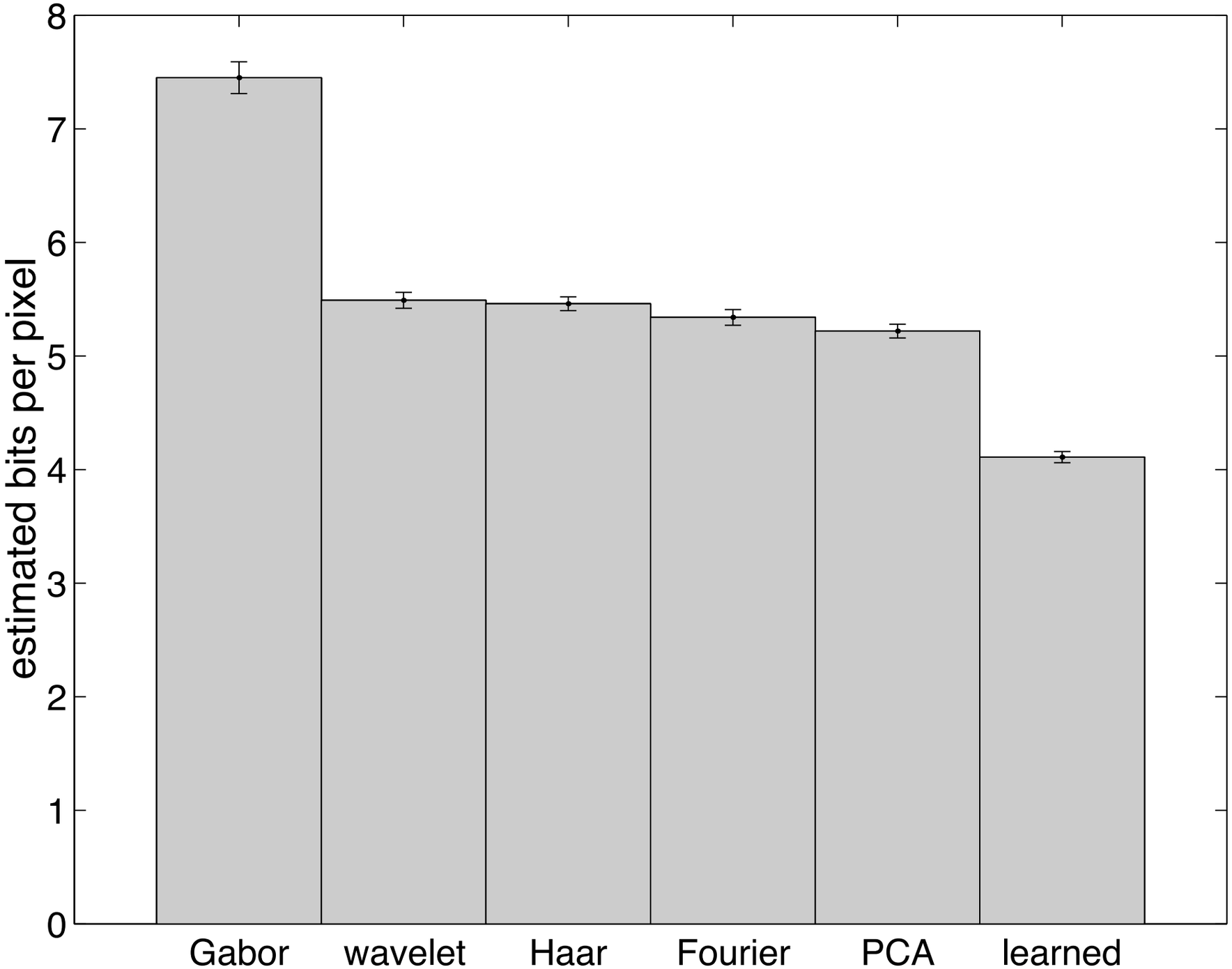
PCA



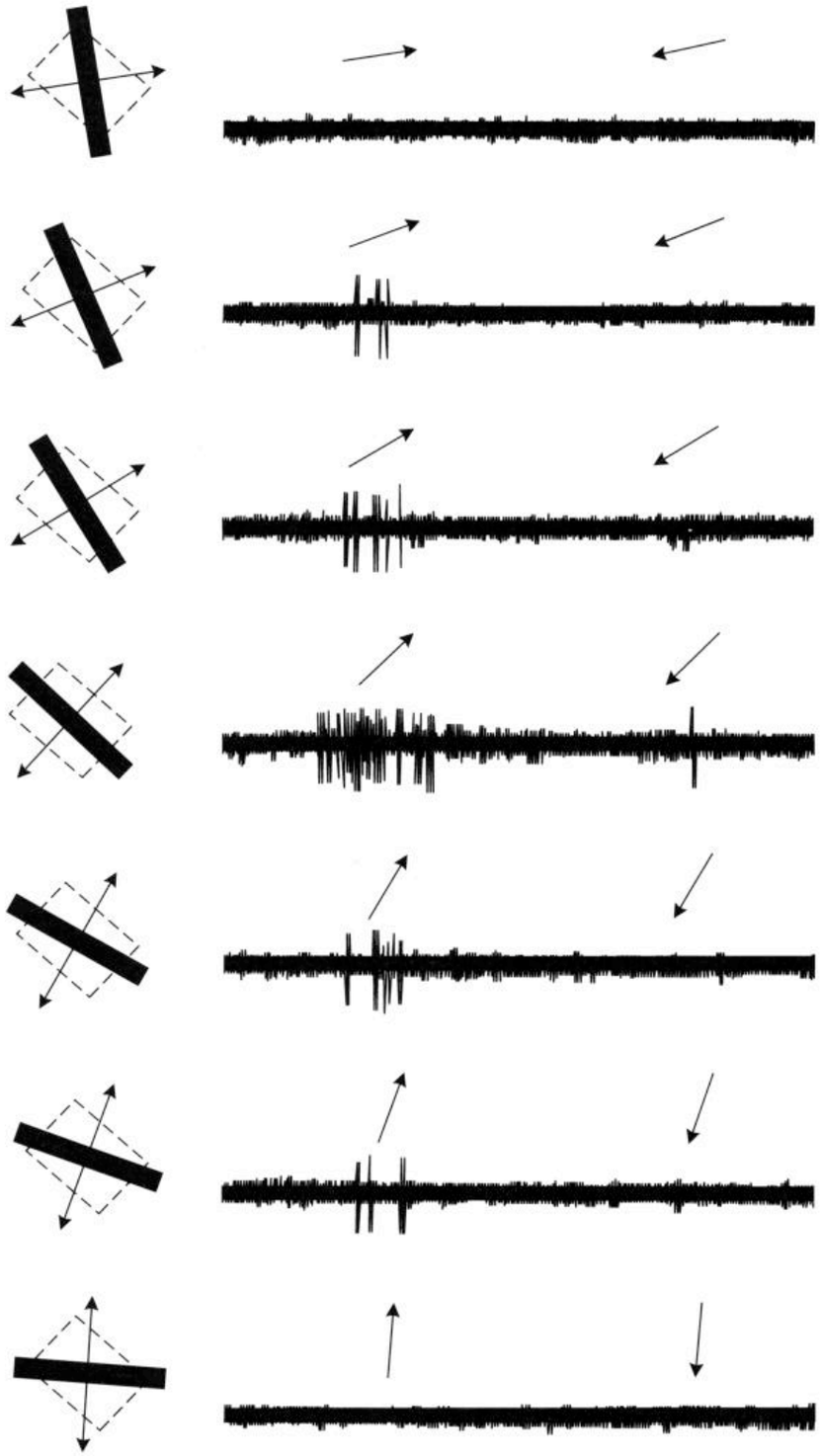
Theoretical perspective: Not edge "detectors."
An efficient code for natural images.

from Lewicki and Olshausen, 1999

Comparing coding efficiency on natural images

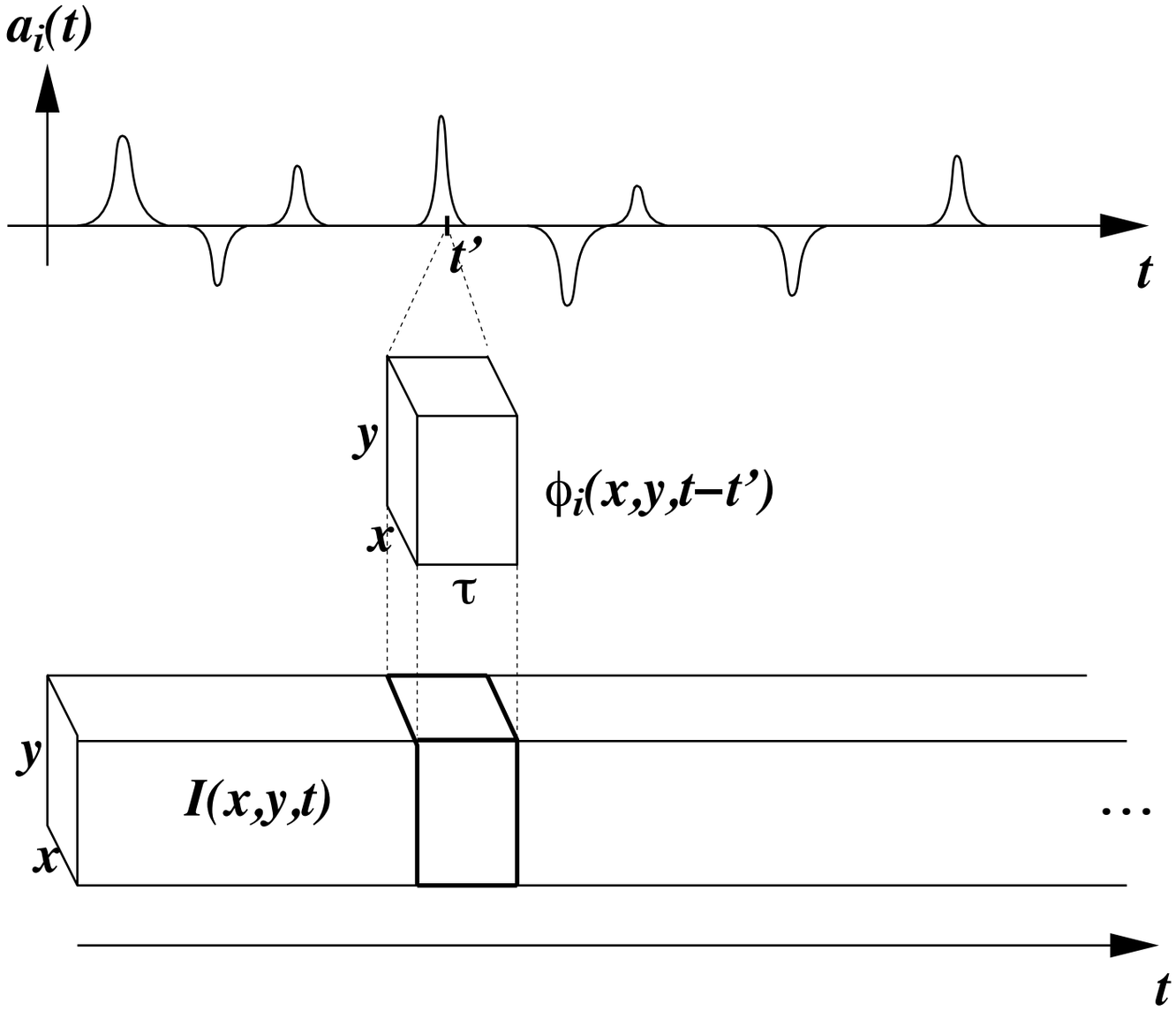


Responses in primary visual cortex to visual motion



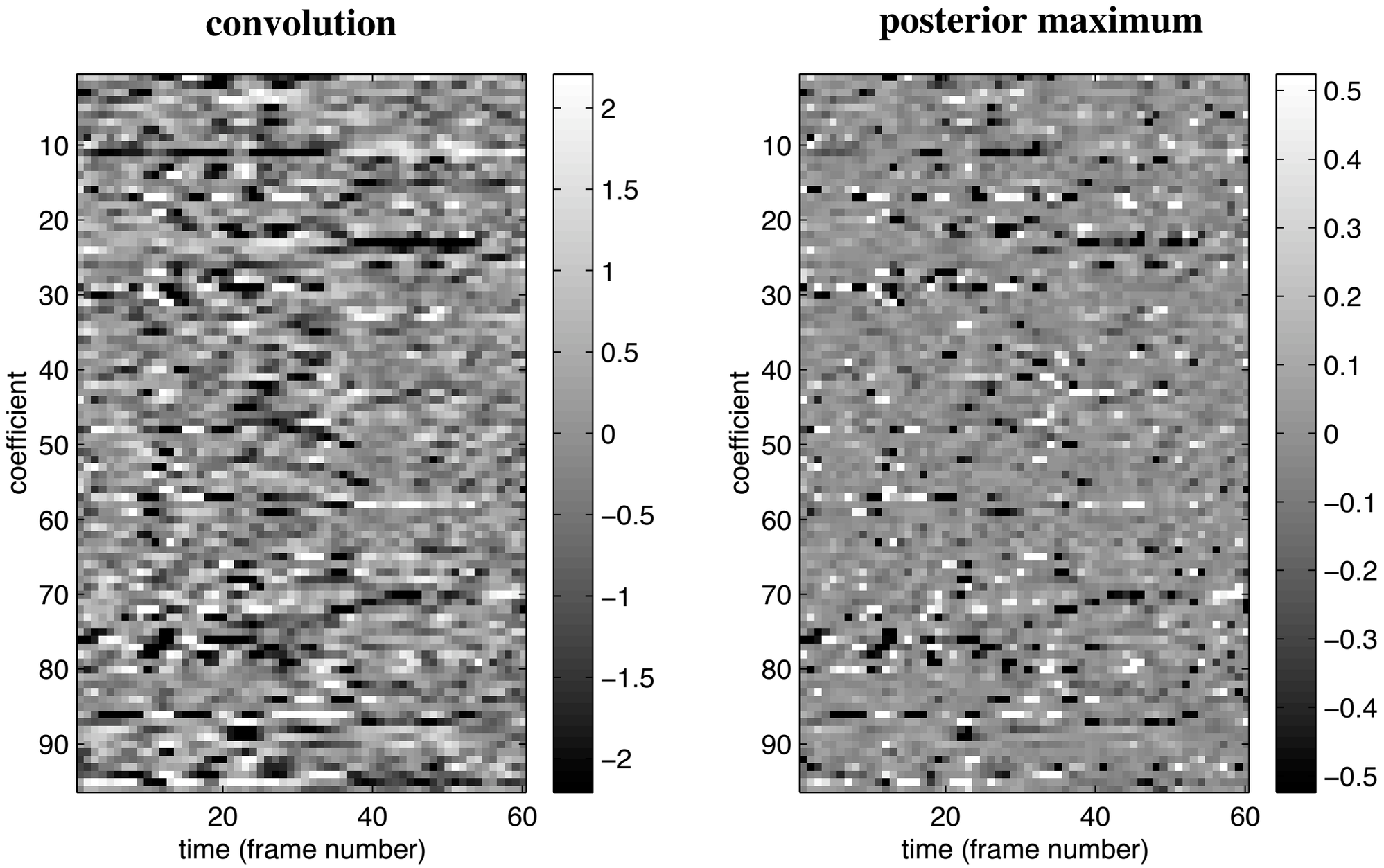
from Wandell, 1995

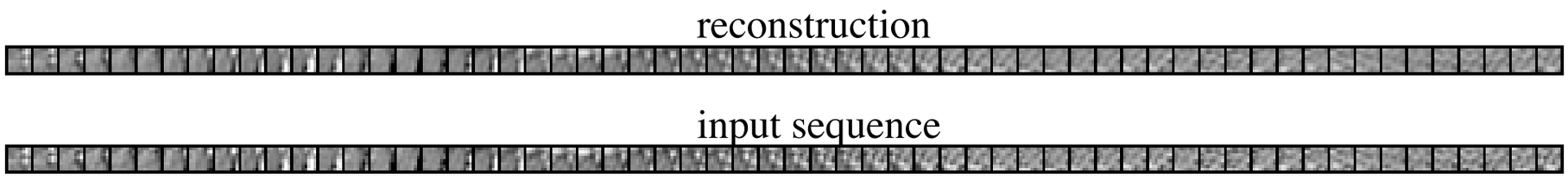
Sparse coding of time-varying images (Olshausen, 2002)



$$
\begin{aligned}
I(x,y,t) &= \sum_i \sum_{t'} a_i(t')\phi_i(x,y,t-t') + \epsilon(x,y,t) \\
&= \sum_i a_i(t) * \phi_i(x,y,t) + \epsilon(x,y,t)
\end{aligned}
$$

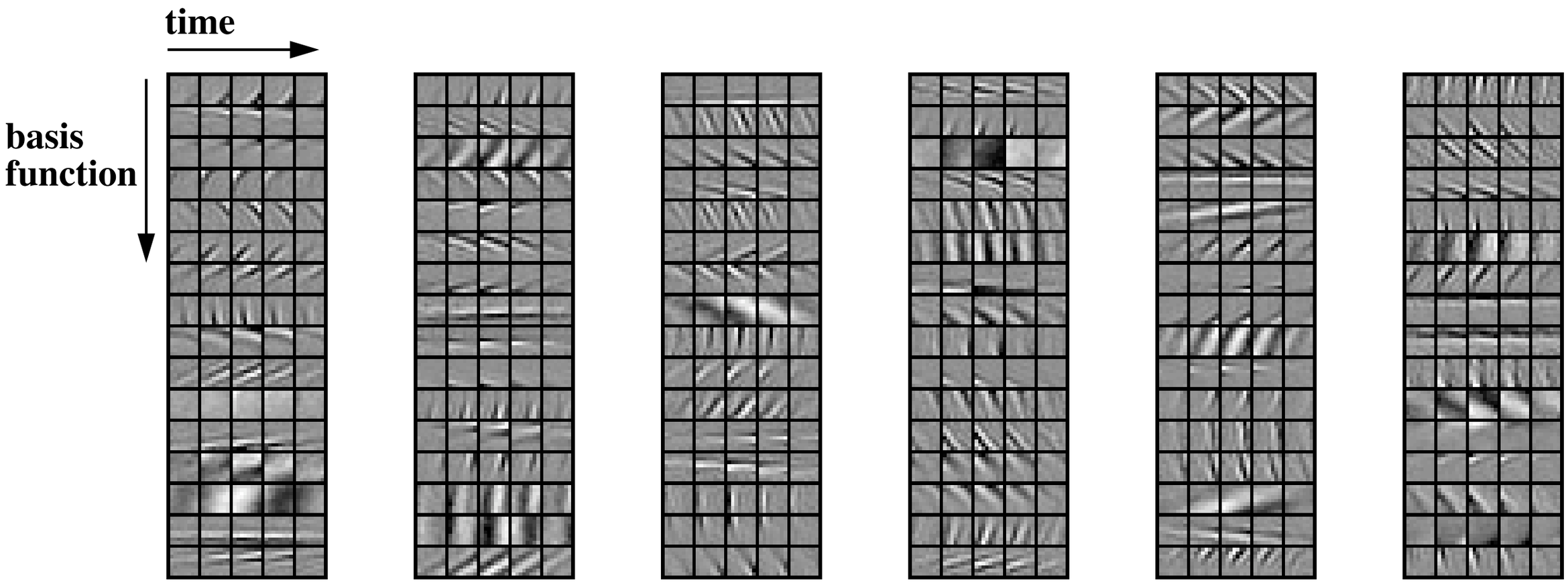
Sparse decomposition of image sequences





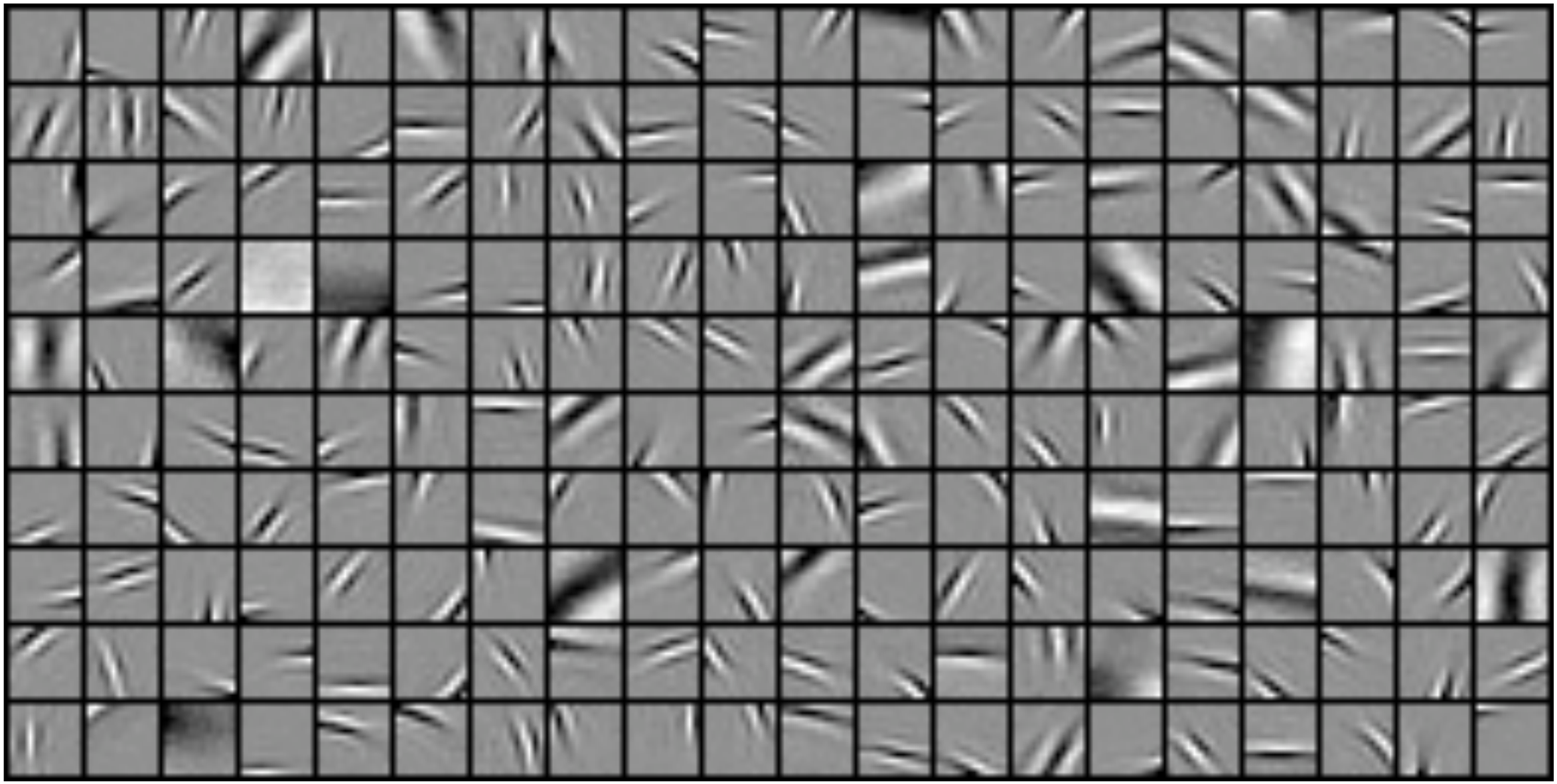
from Olshausen, 2002

Learned spatio-temporal basis functions



from Olshausen, 2002

Animated spatial-temporal basis functions



from Olshausen, 2002

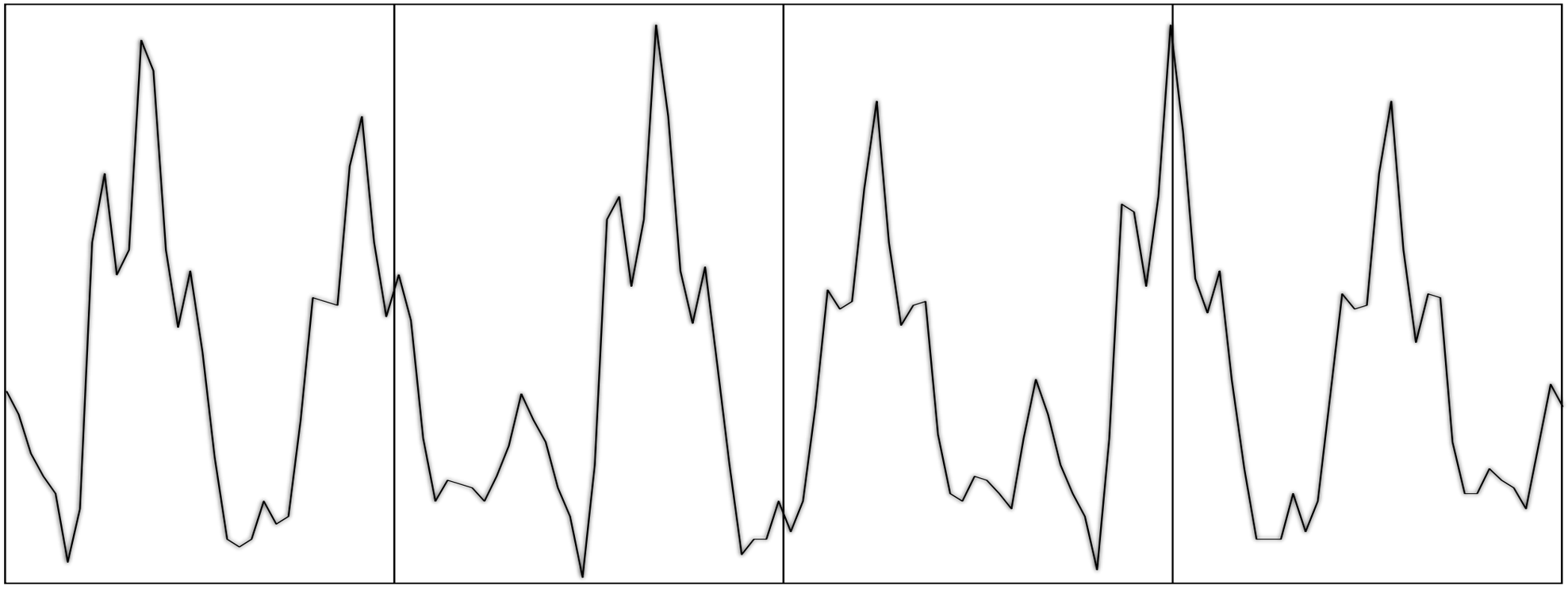
Theory	• explains data from principles • requires idealization and abstraction
Principle	• code signals accurately and efficiently • adapted to natural sensory environment
Idealization	• cell response is linear
Methodology	• information theory, natural images
Prediction	• explains individual receptive fields • explains population organization

How general is the efficient coding principle?

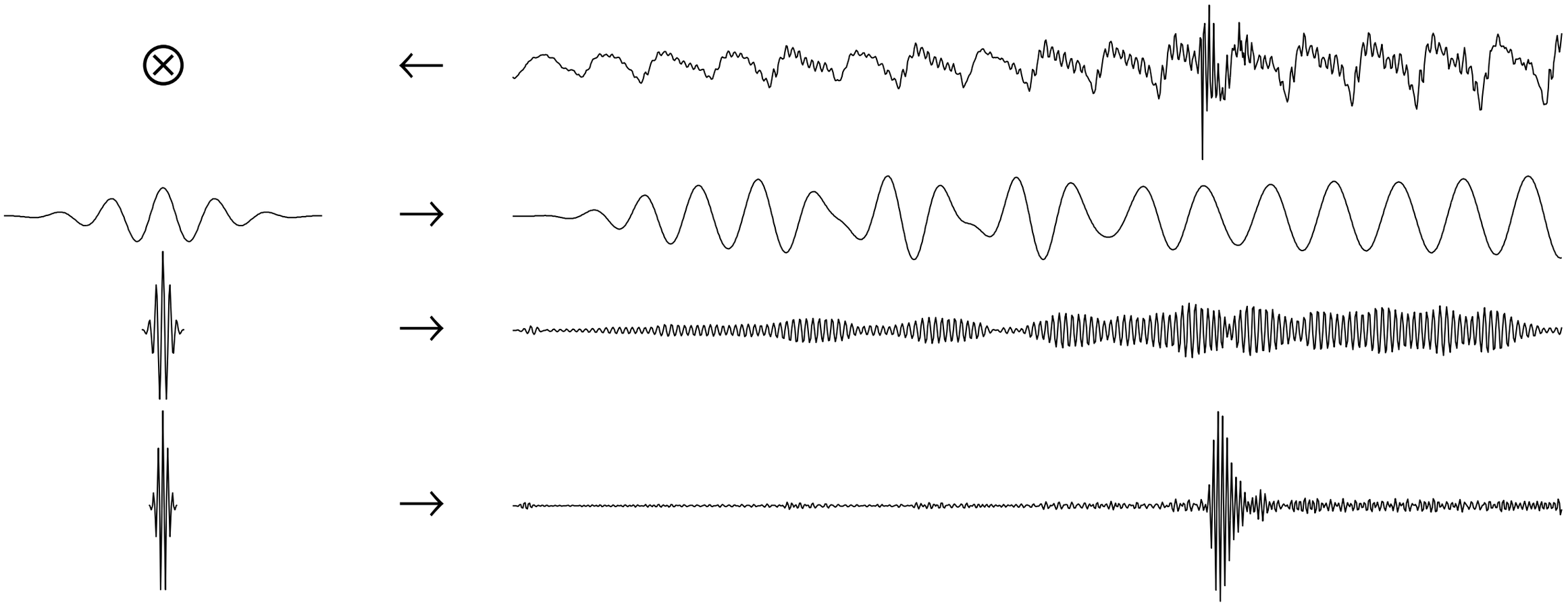
Can it explain auditory coding?

Limitations of the linear model

- linear
- only optimal for block, not whole signal
- no phase-locking
- representation depends on block alignment



A continuous filterbank does not form an efficient code

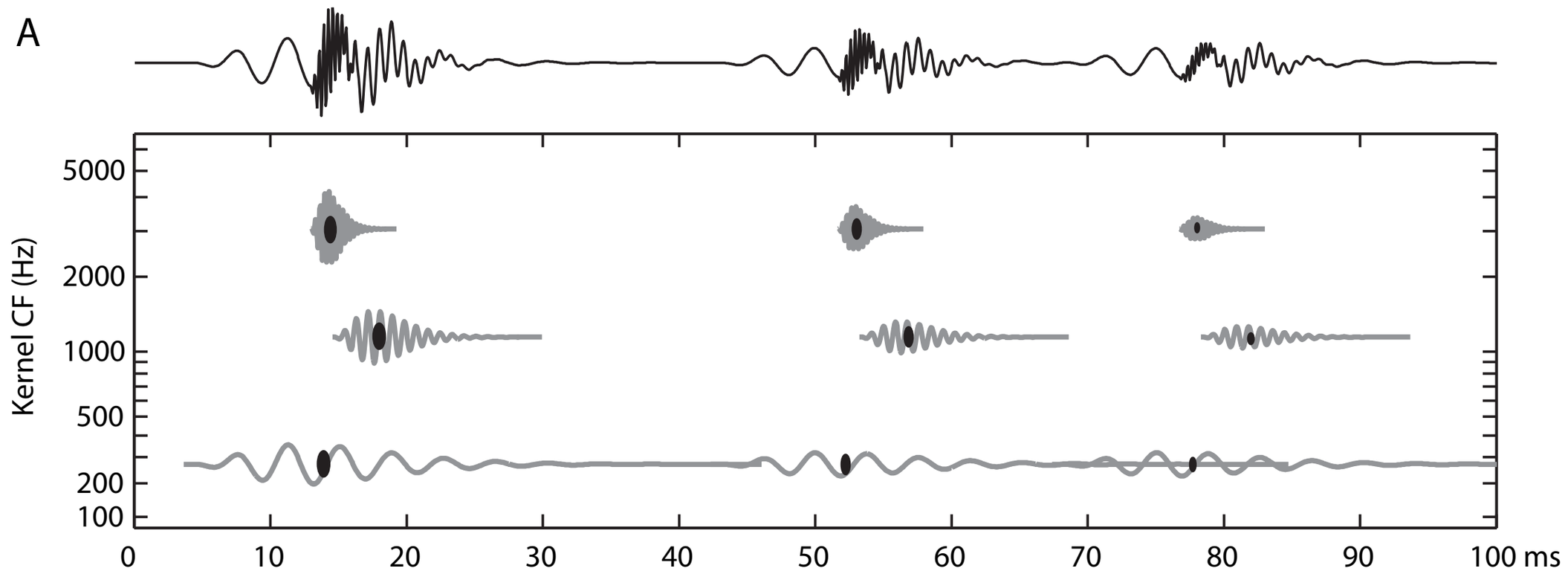


Goal:

find a representation that is both *time-relative* and *efficient*

Efficient signal representation using time-shiftable kernels (spikes)

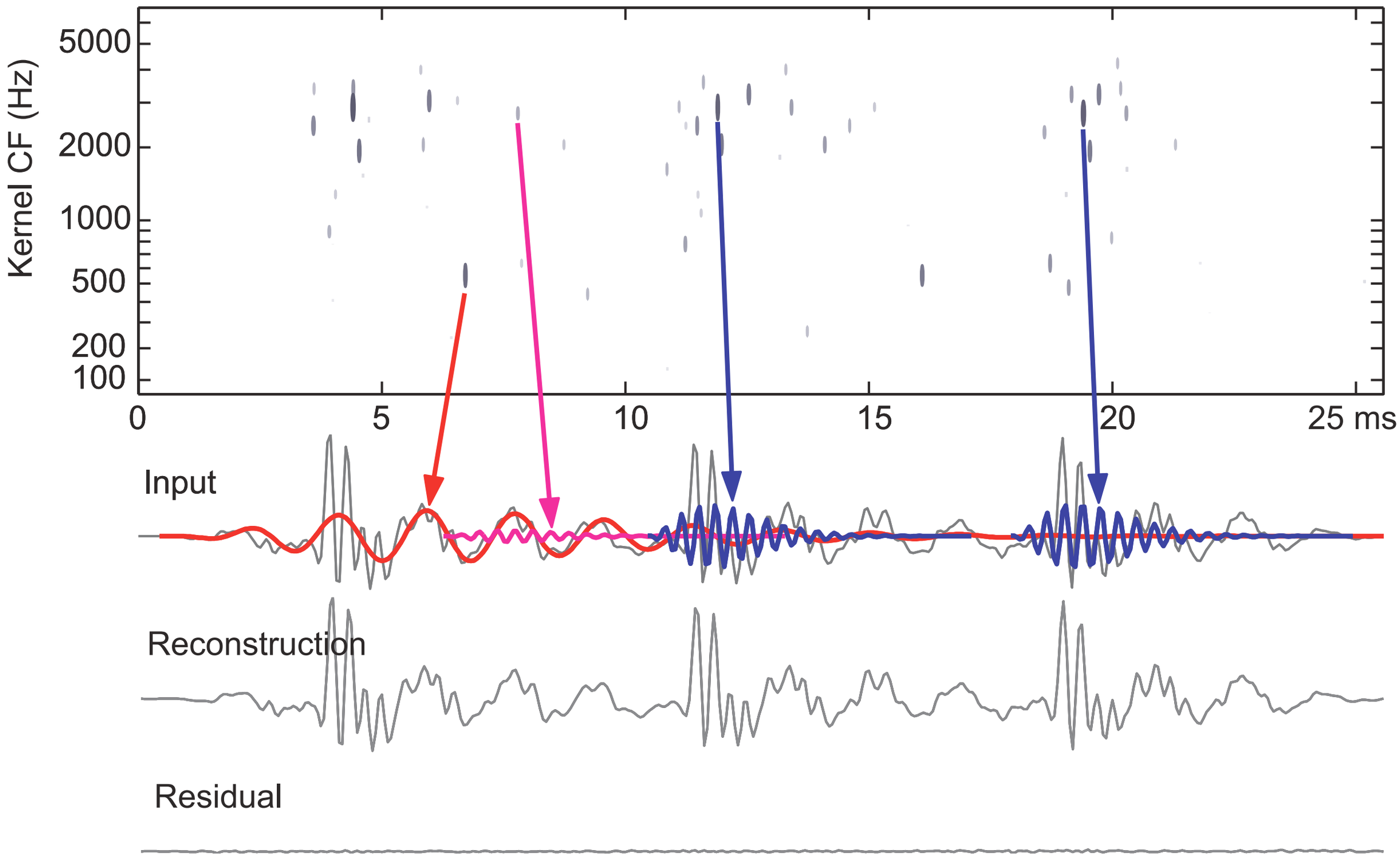
Smith and Lewicki (2005) *Neural Comp.* 17:19-45



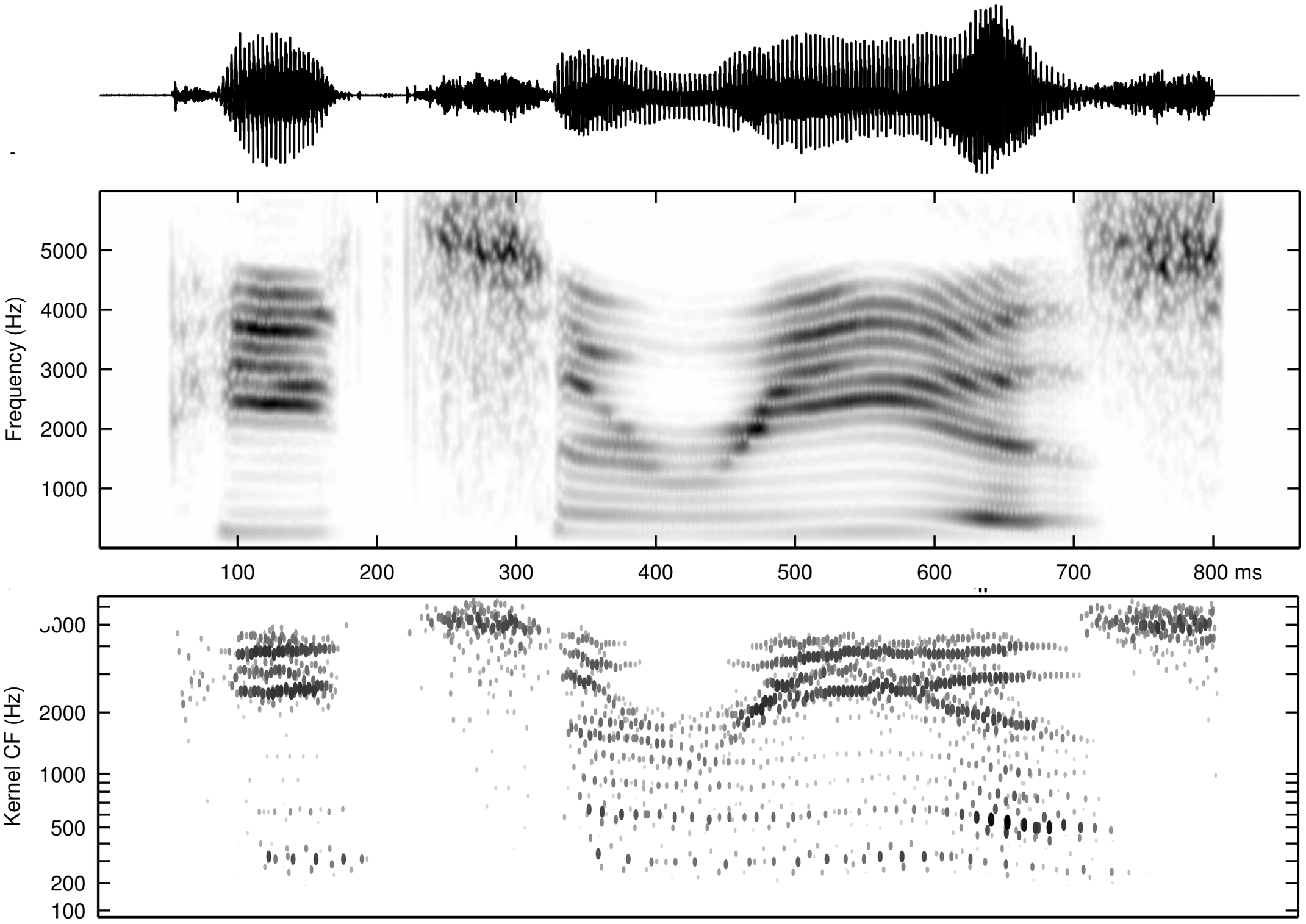
$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_{m,i}\, \phi_m(t - \tau_{m,i}) + \epsilon(t)$$

- Each spike encodes the precise time and magnitude of an acoustic feature
- Two important theoretical abstractions for "spikes"
  - not binary
  - not probabilistic
- Can convert to a population of stochastic, binary spikes

Coding audio signals with spikes

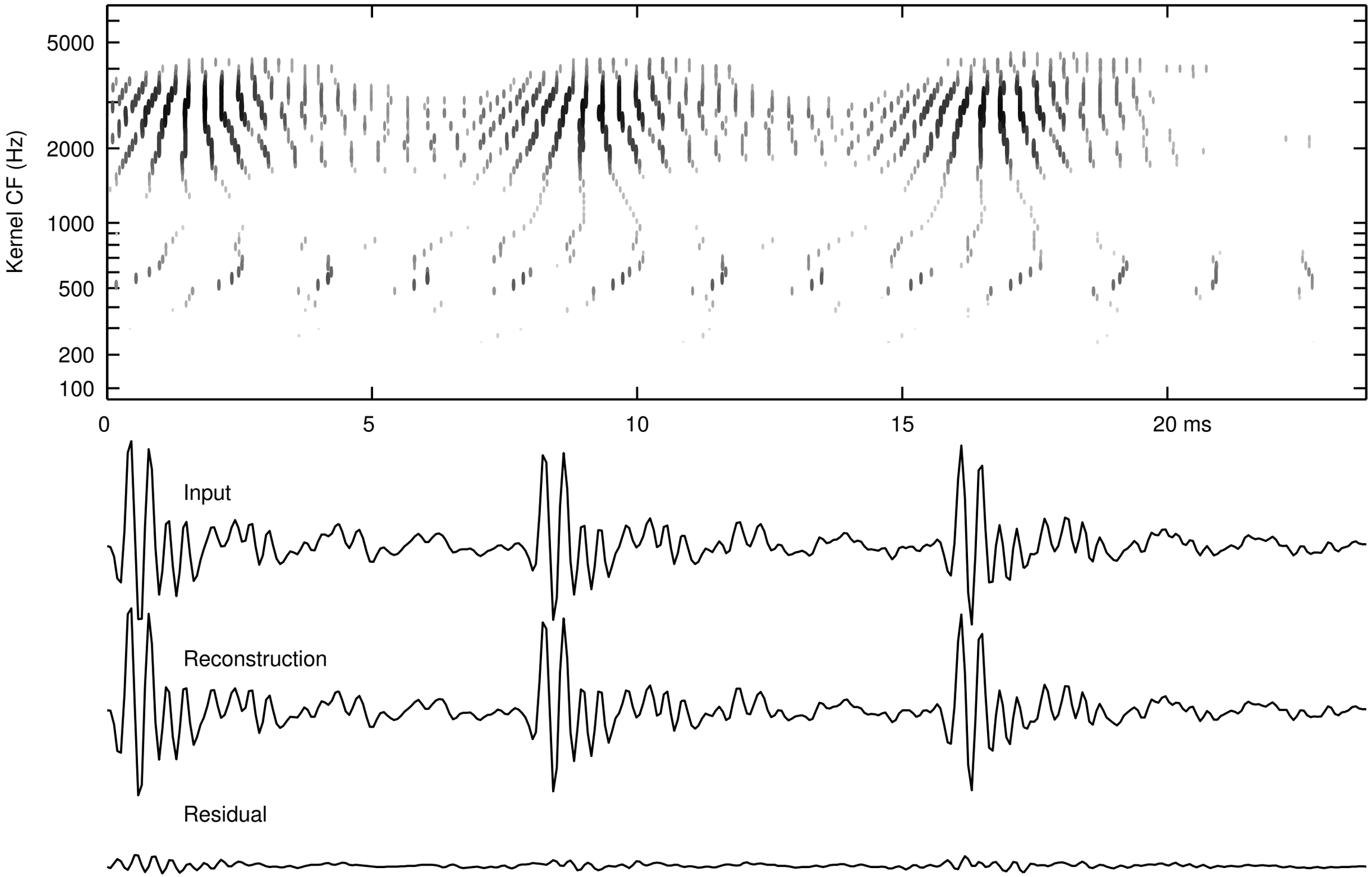


Comparing a spike code to a spectrogram



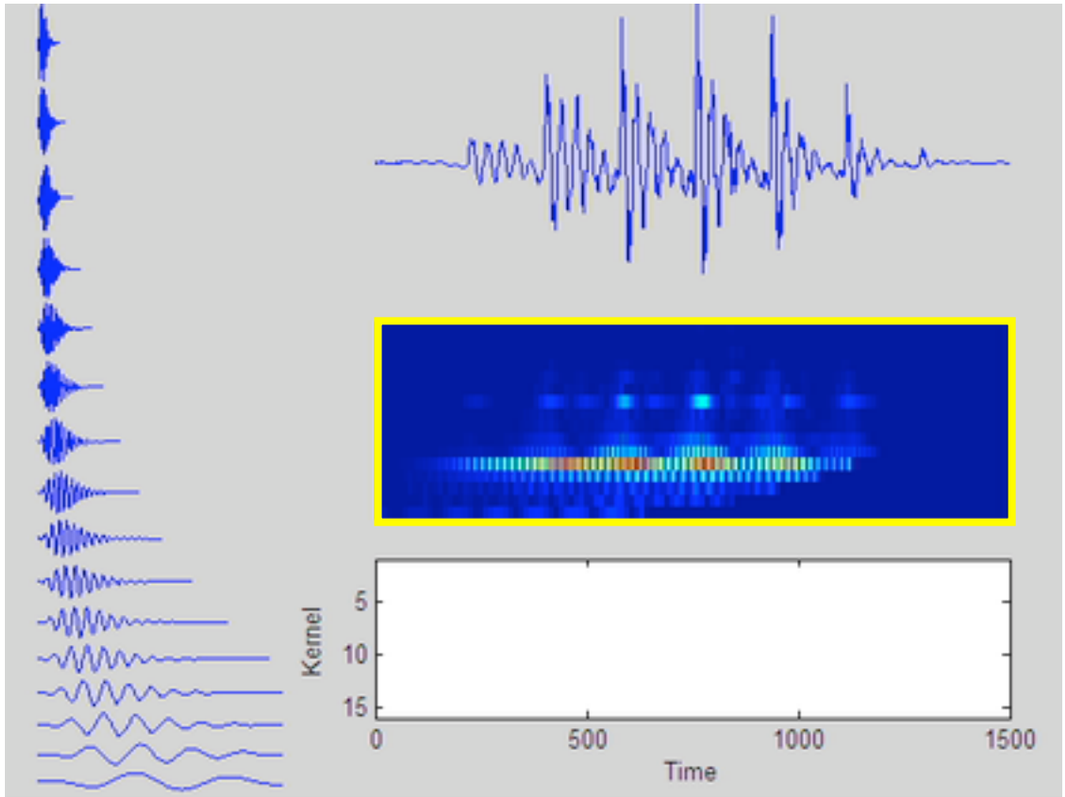
How do we compute the spikes?

“can” with filter-threshold



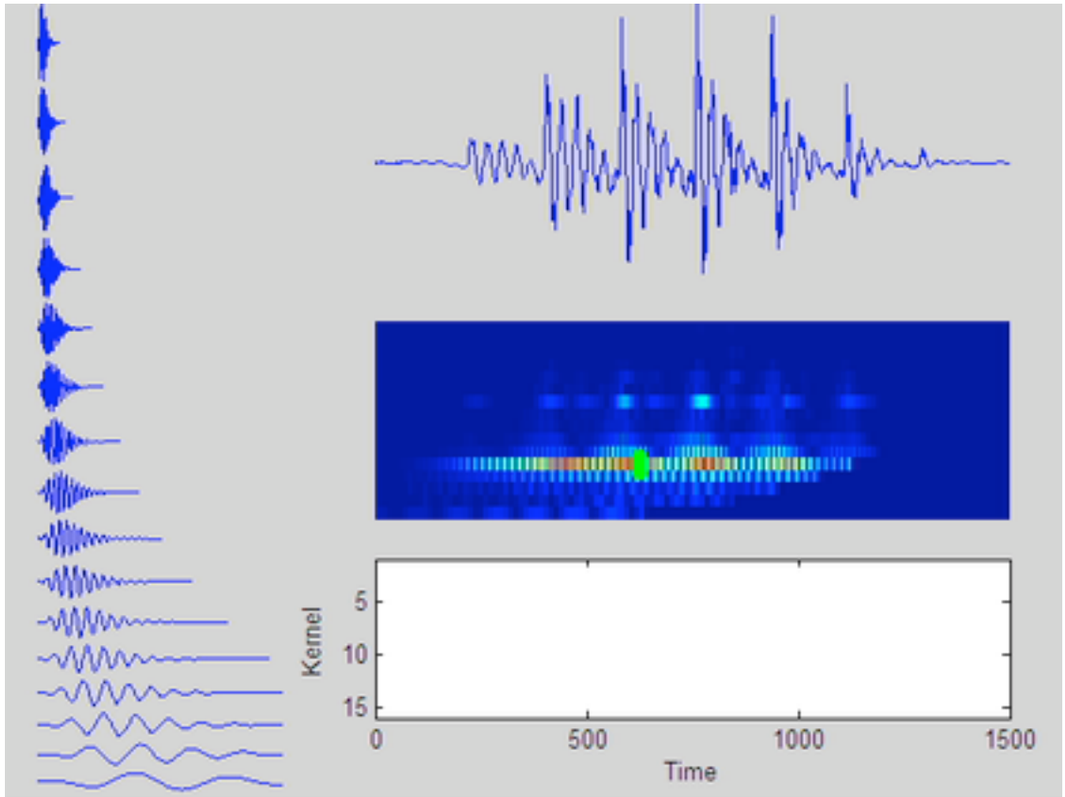
Spike Coding with Matching Pursuit

1. convolve signal with kernels



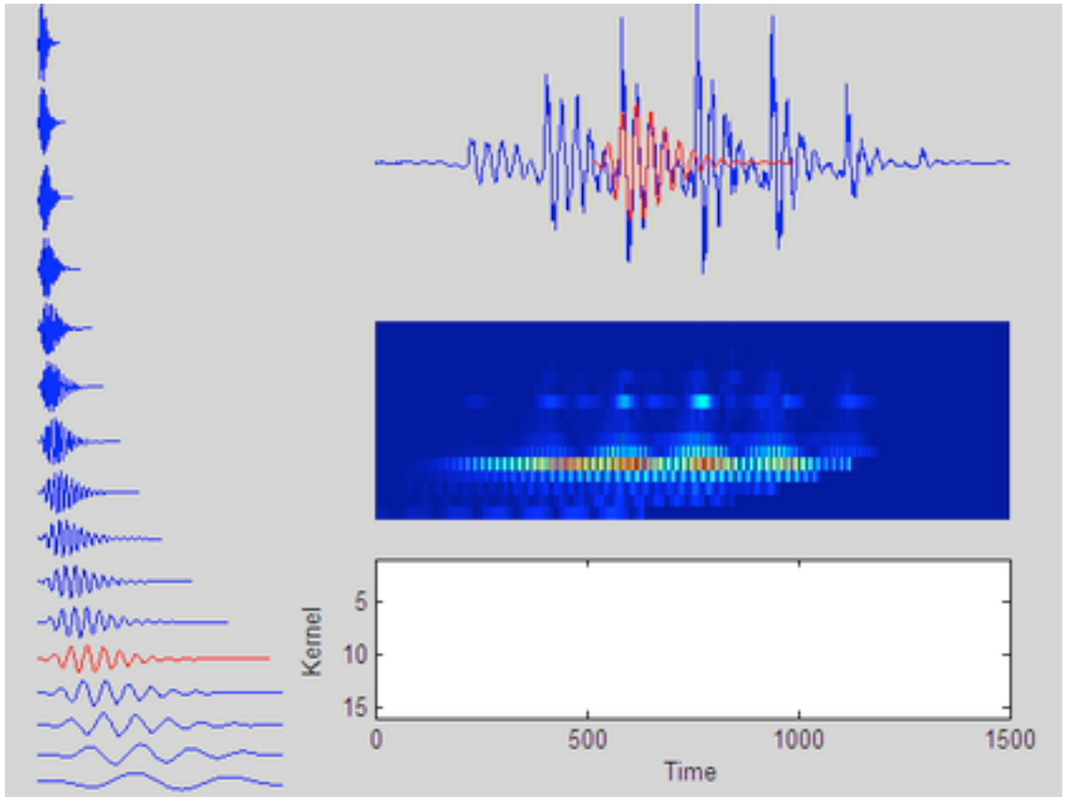
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set



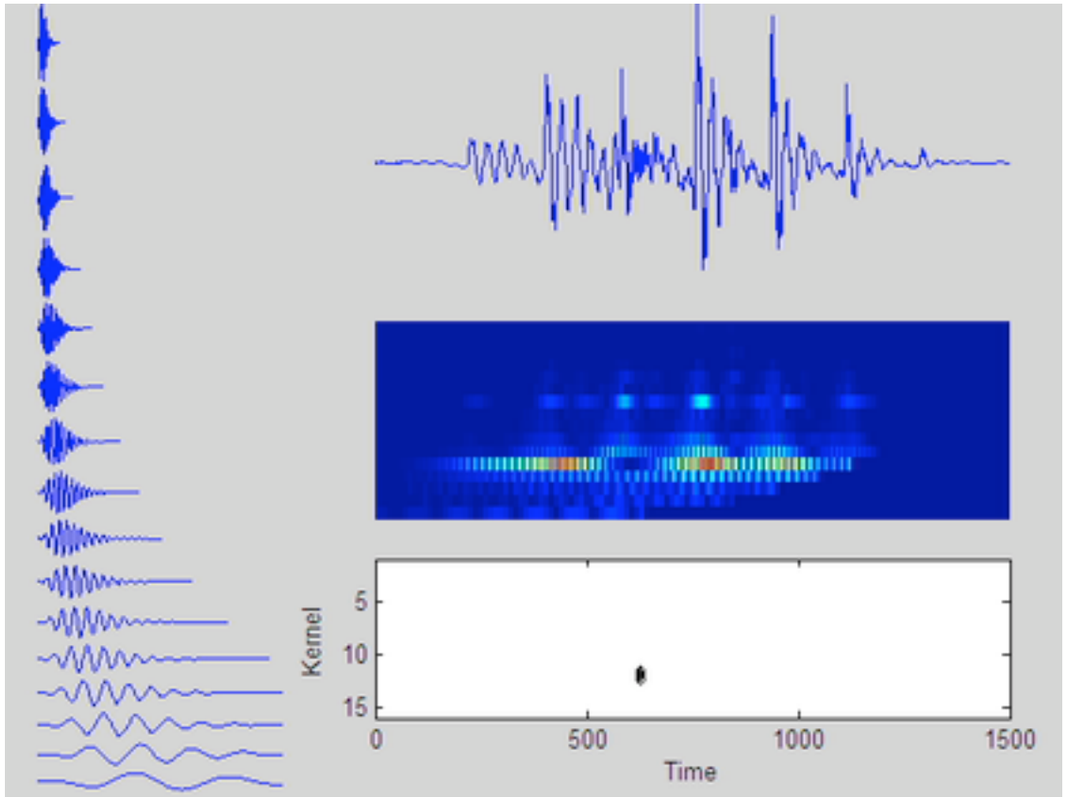
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel



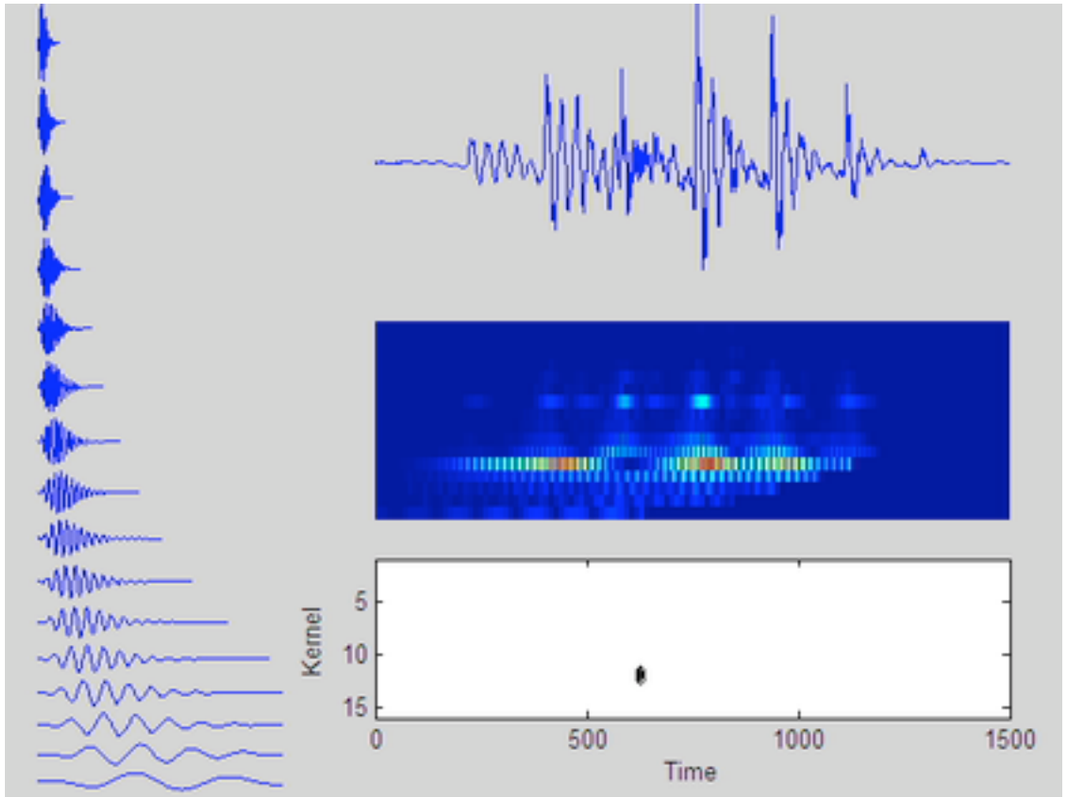
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel
4. subtract kernel from signal, record spike, and adjust convolutions



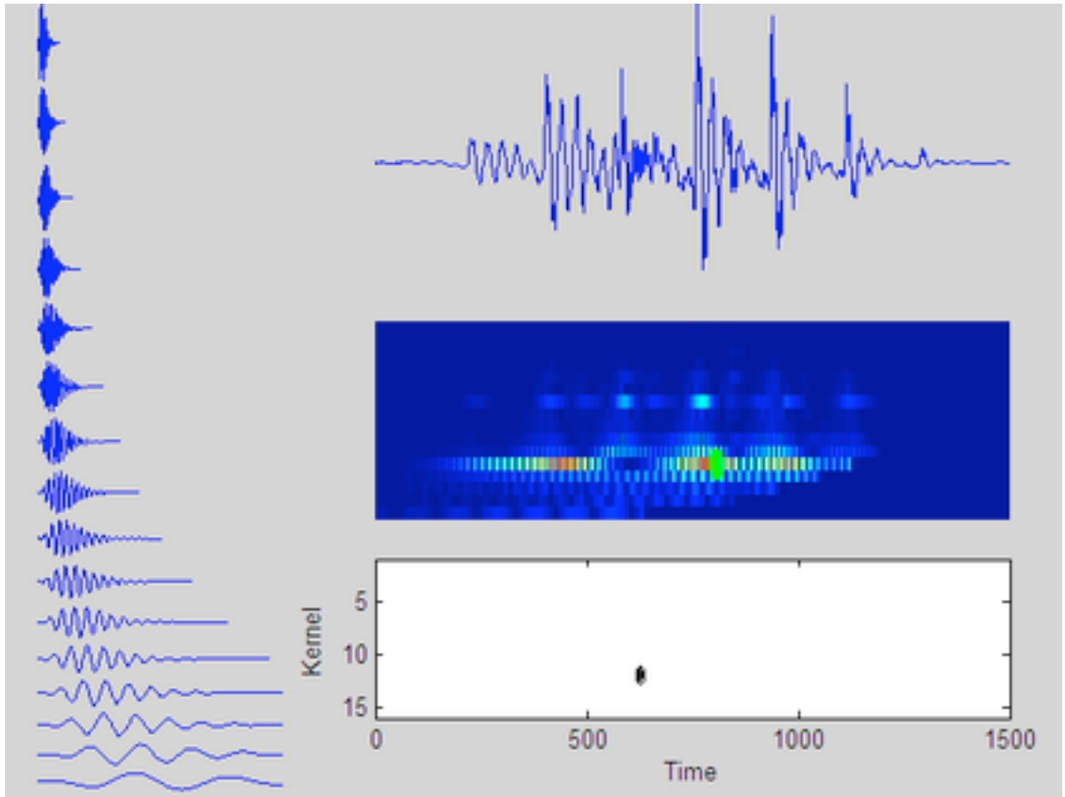
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel
4. subtract kernel from signal, record spike, and adjust convolutions
5. repeat



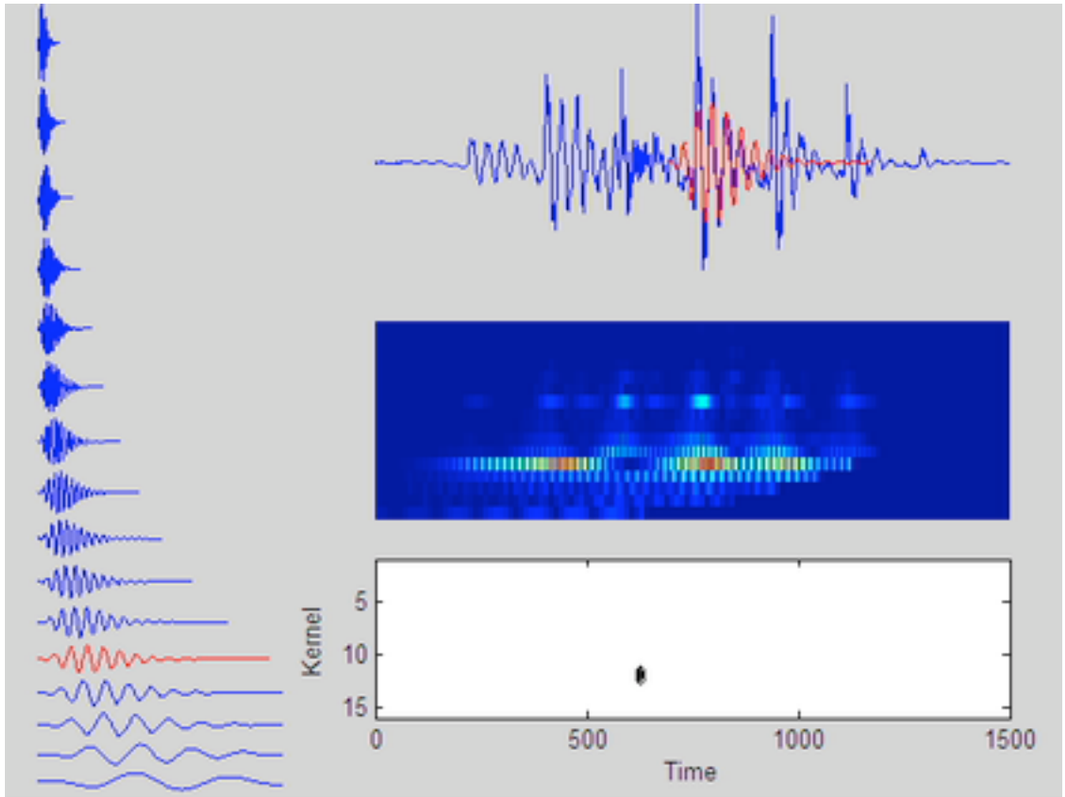
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. **find largest peak over convolution set**
3. fit signal with kernel
4. subtract kernel from signal, record spike, and adjust convolutions
5. repeat



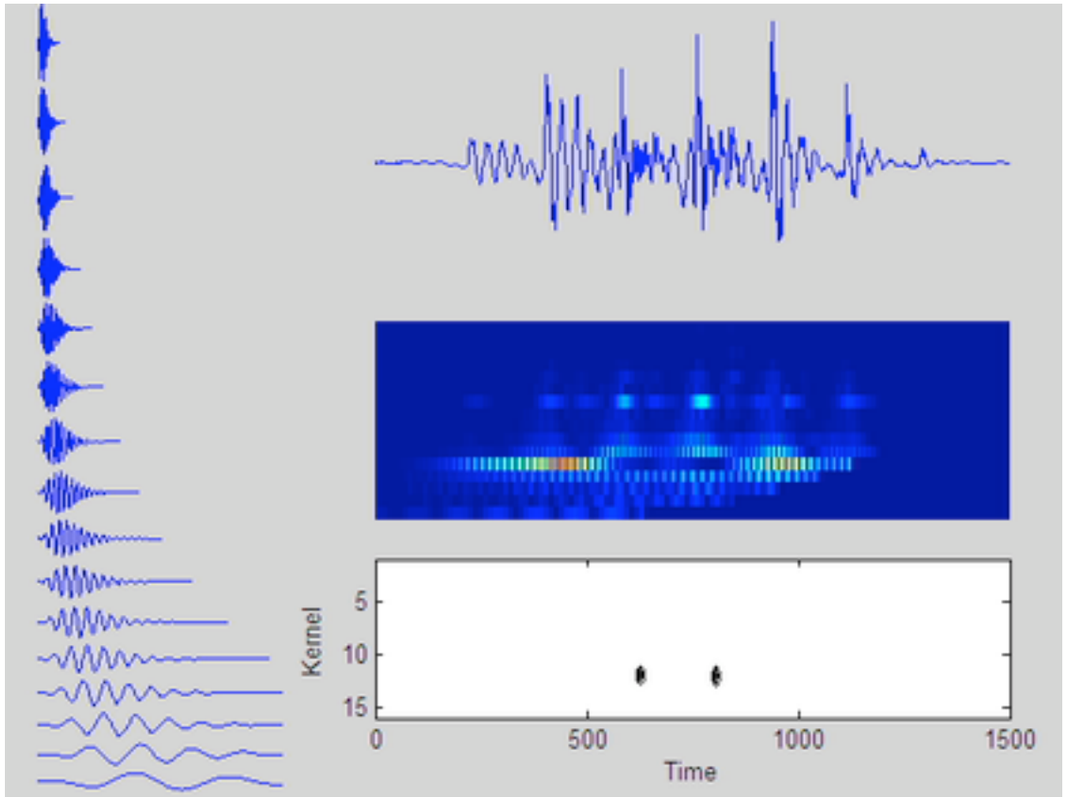
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. **fit signal with kernel**
4. subtract kernel from signal, record spike, and adjust convolutions
5. repeat



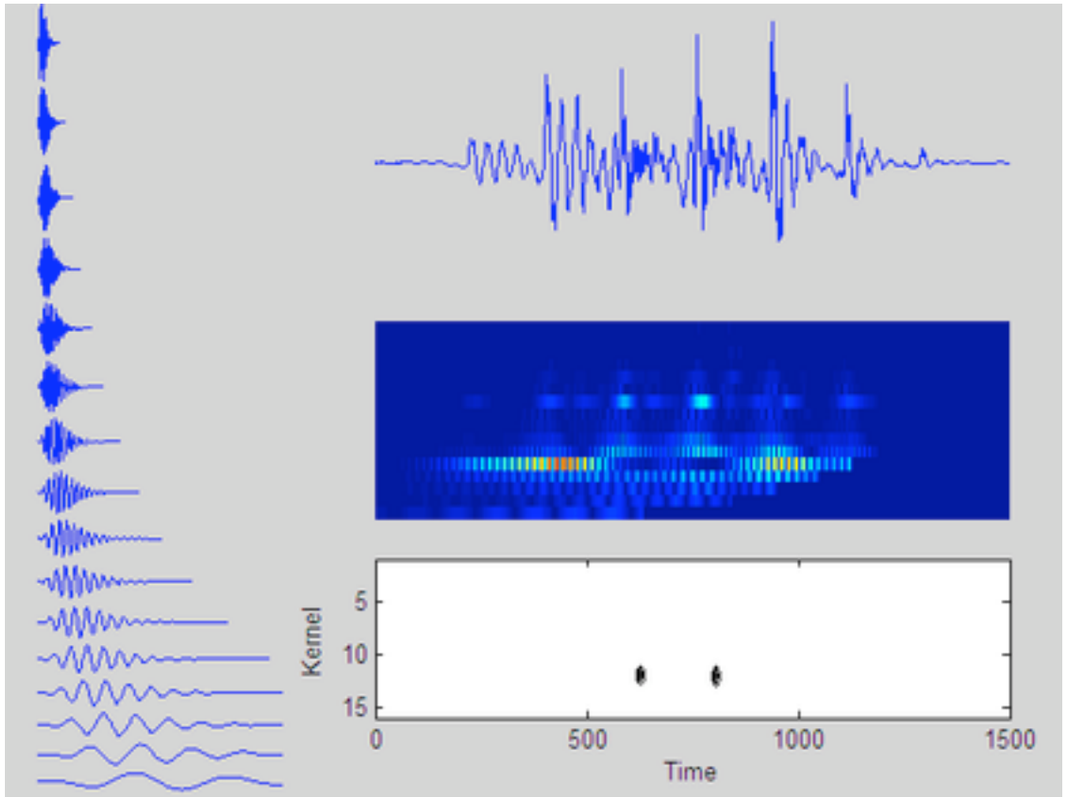
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel
4. **subtract kernel from signal, record spike, and adjust convolutions**
5. repeat



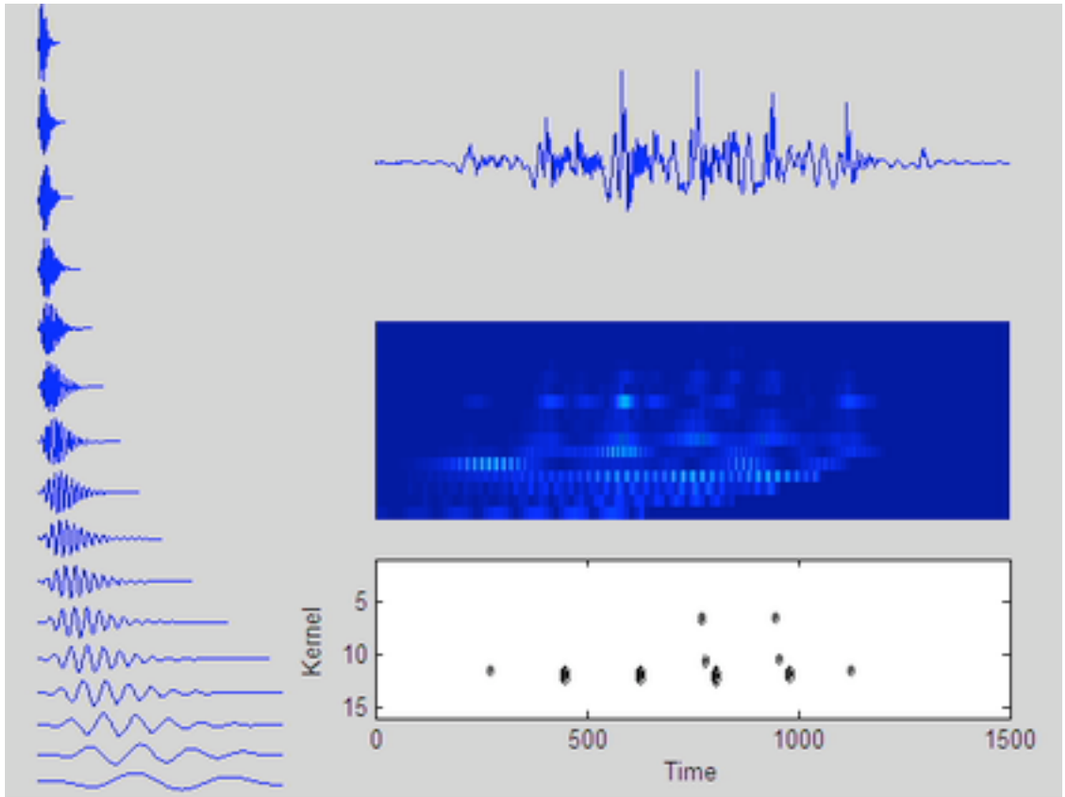
Spike Coding with Matching Pursuit

1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel
4. subtract kernel from signal, record spike, and adjust convolutions
5. **repeat . . .**

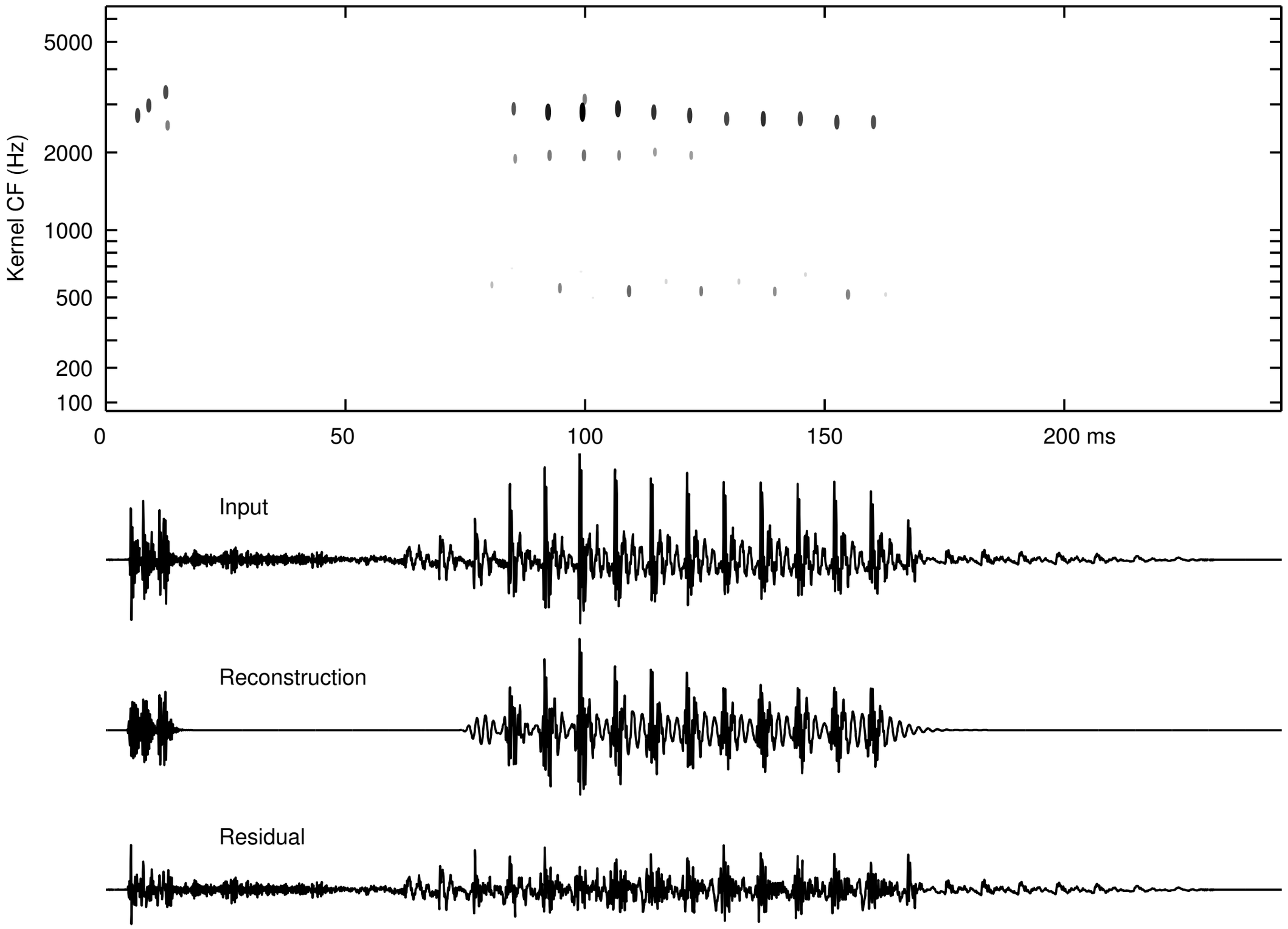


Spike Coding with Matching Pursuit

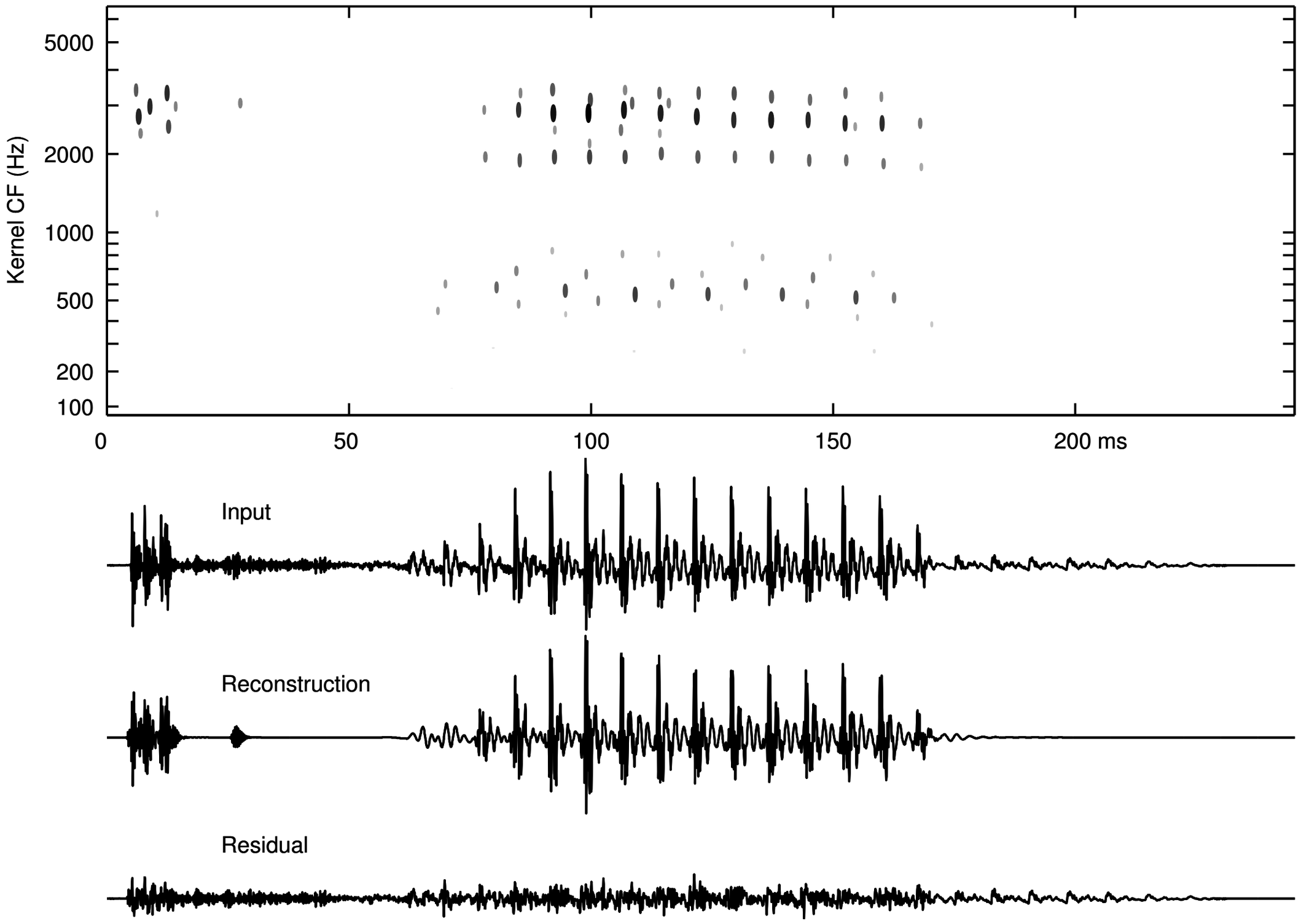
1. convolve signal with kernels
2. find largest peak over convolution set
3. fit signal with kernel
4. subtract kernel from signal, record spike, and adjust convolutions
5. repeat . . .
6. halt when desired fidelity is reached



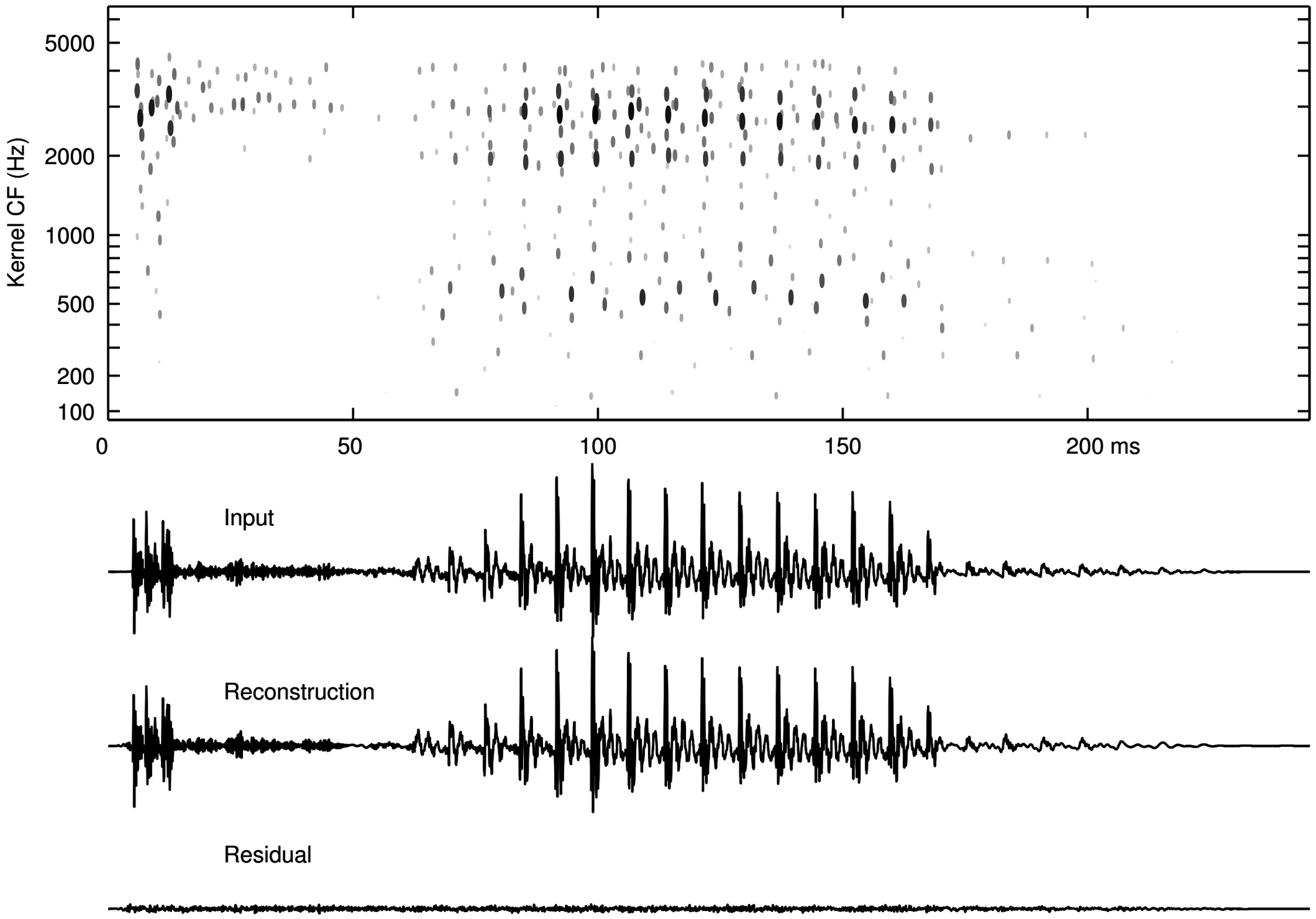
“can” 5 dB SNR, 36 spikes, 145 sp/sec



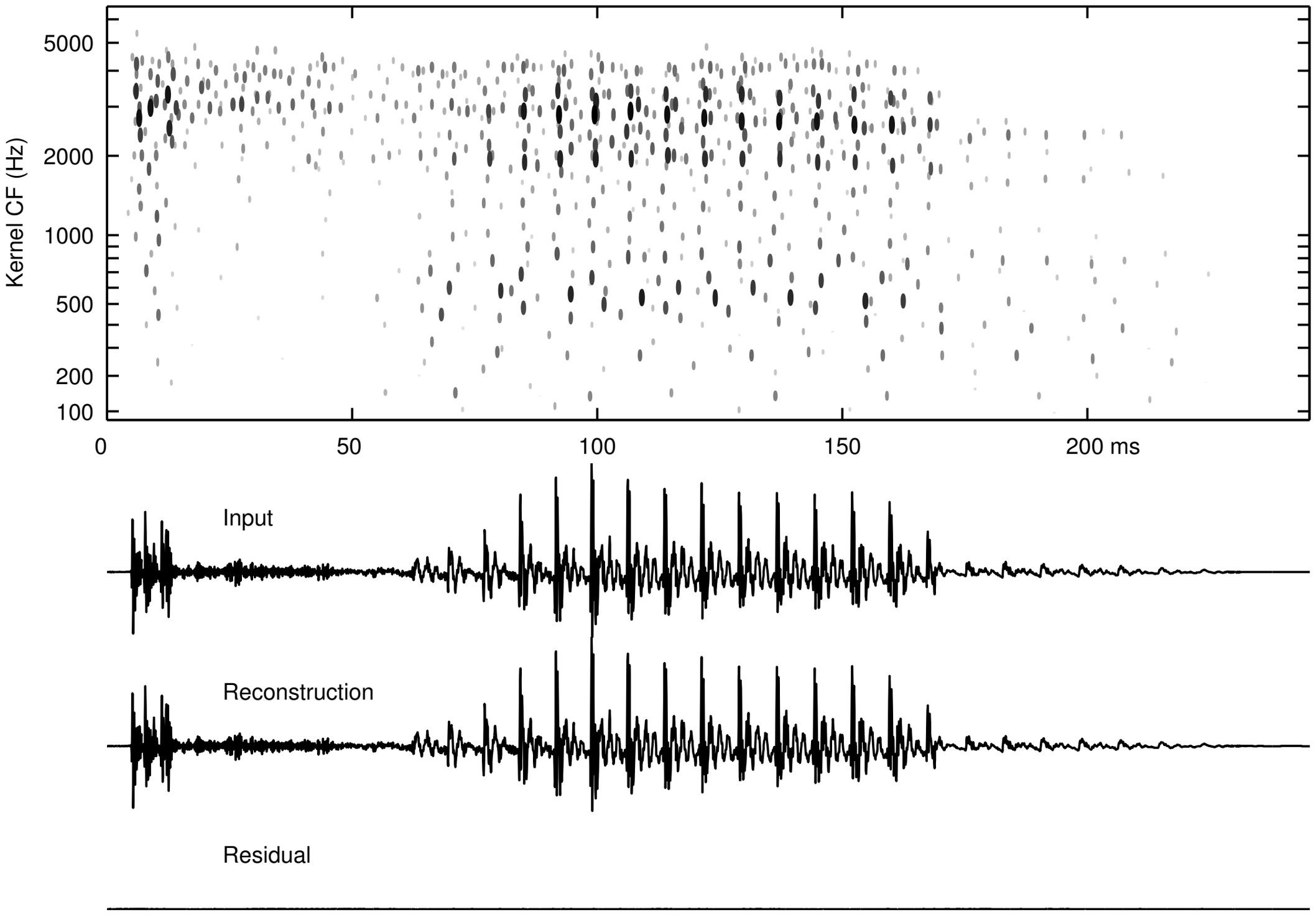
“can” 10 dB SNR, 93 spikes, 379 sp/sec



“can” 20 dB SNR, 391 spikes, 1700 sp/sec



“can” 40 dB SNR, 1285 spikes, 5238 sp/sec

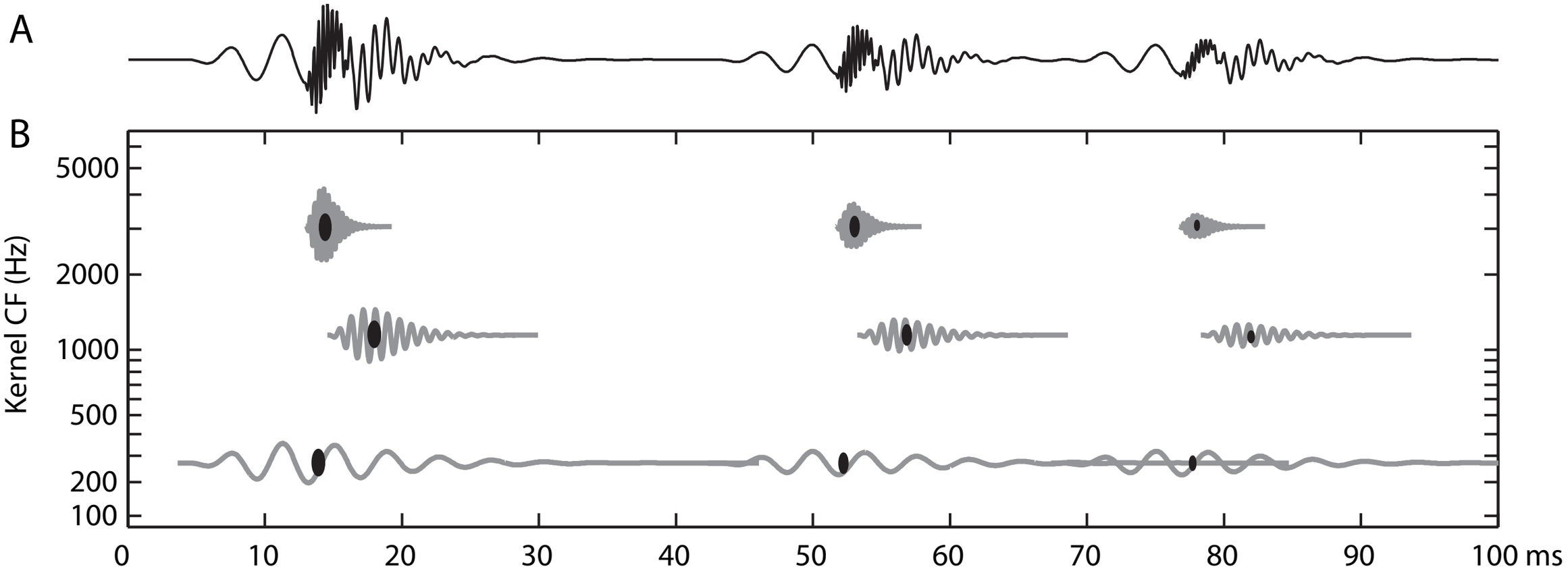


Efficient auditory coding with optimized kernel shapes

Smith and Lewicki (2006) *Nature* 439:978-982

What are the optimal kernel shapes?

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_{m,i}\, \phi_m(t - \tau_{m,i}) + \epsilon(t)$$



Adapt algorithm of Olshausen (2002)

Optimizing the probabilistic model

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_i^m \, \phi_m(t - \tau_i^m) + \epsilon(t),$$

$$\begin{aligned} p(x|\Phi) &= \int p(x|\Phi, s, \tau) p(s) p(\tau) ds d\tau \\ &\approx p(x|\Phi, \hat{s}, \hat{\tau}) p(\hat{s}) p(\hat{\tau}) \end{aligned}$$

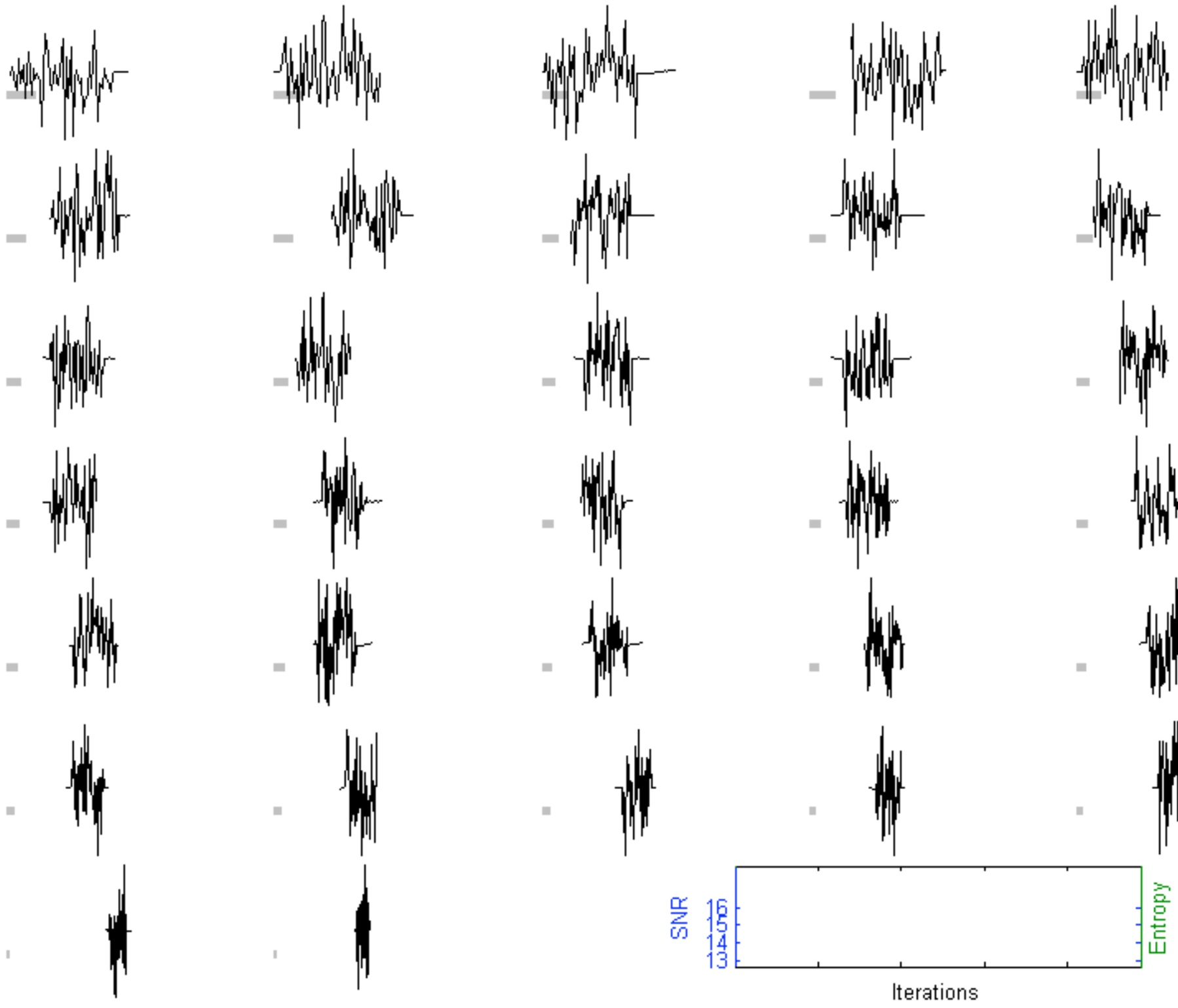
$$\epsilon(t) \sim \mathcal{N}(0, \sigma_\epsilon)$$

Learning (Olshausen, 2002):

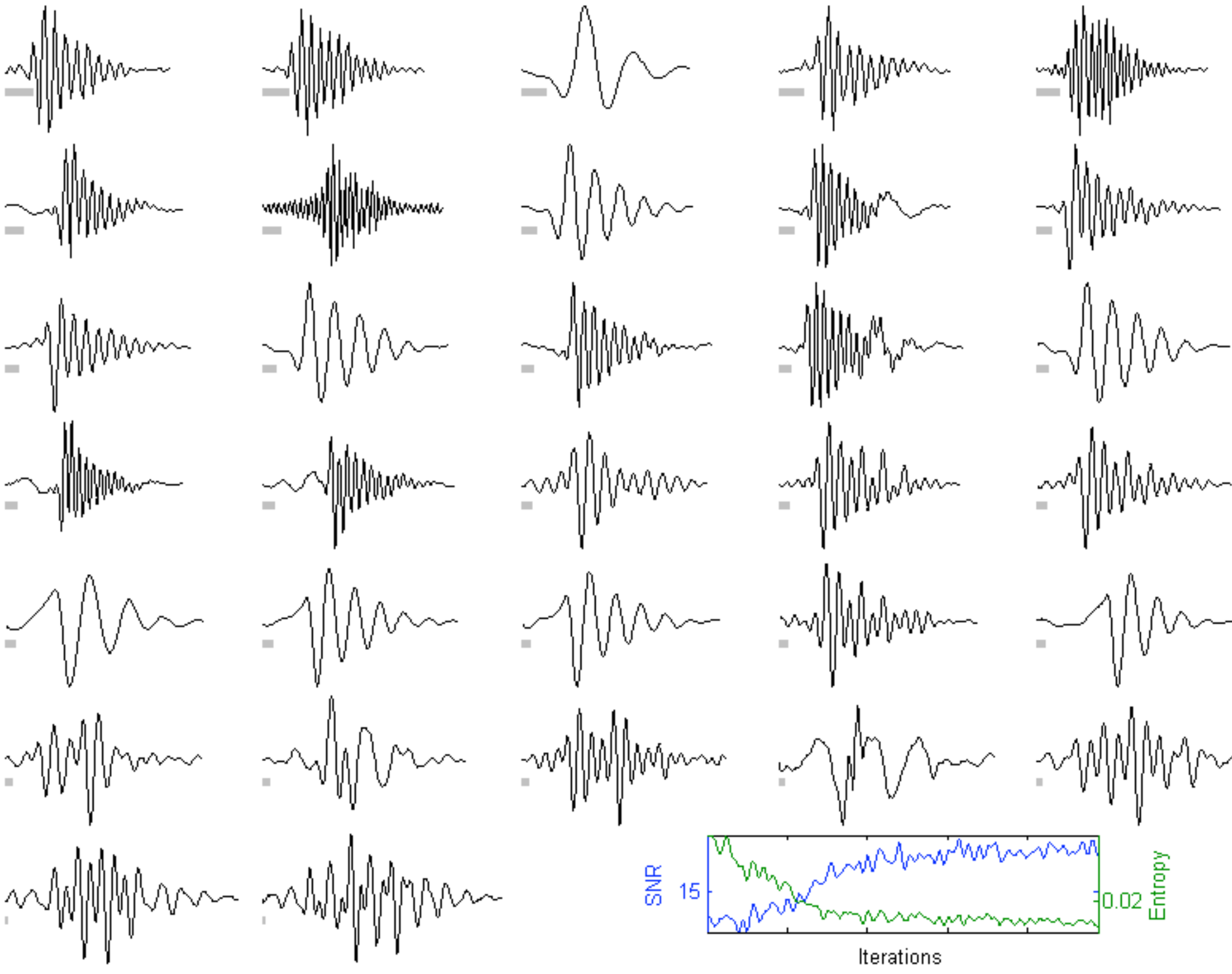
$$\begin{aligned} \frac{\partial}{\partial \phi_m} \log p(x|\Phi) &= \frac{\partial}{\partial \phi_m} \log p(x|\Phi, \hat{s}, \hat{\tau}) + \log p(\hat{s}) p(\hat{\tau}) \\ &= \frac{1}{2\sigma_\varepsilon} \frac{\partial}{\partial \phi_m} [x - \sum_{m=1}^{M} \sum_{i=1}^{n_m} \hat{s}_i^m \, \phi_m(t - \tau_i^m)]^2 \\ &= \frac{1}{\sigma_\varepsilon} [x - \hat{x}] \sum_i \hat{s}_i^m \end{aligned}$$

Also adapt kernel lengths

Adapting the optimal kernel shapes



Kernel functions optimized for coding speech



Quantifying coding efficiency

1. *fit* signal
2. *quantize* time and amplitude values
3. *prune* zero values
4. *measure coding efficiency* using the entropy of quantized values
5. *reconstruct signal* using quantized values
6. *measure fidelity* using signal-to-noise ratio (SNR) of residual error

- identical procedure for other codes (e.g. Fourier and wavelet)

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_{m,i}\, \phi_m(t - \tau_{m,i}) + \epsilon(t)$$

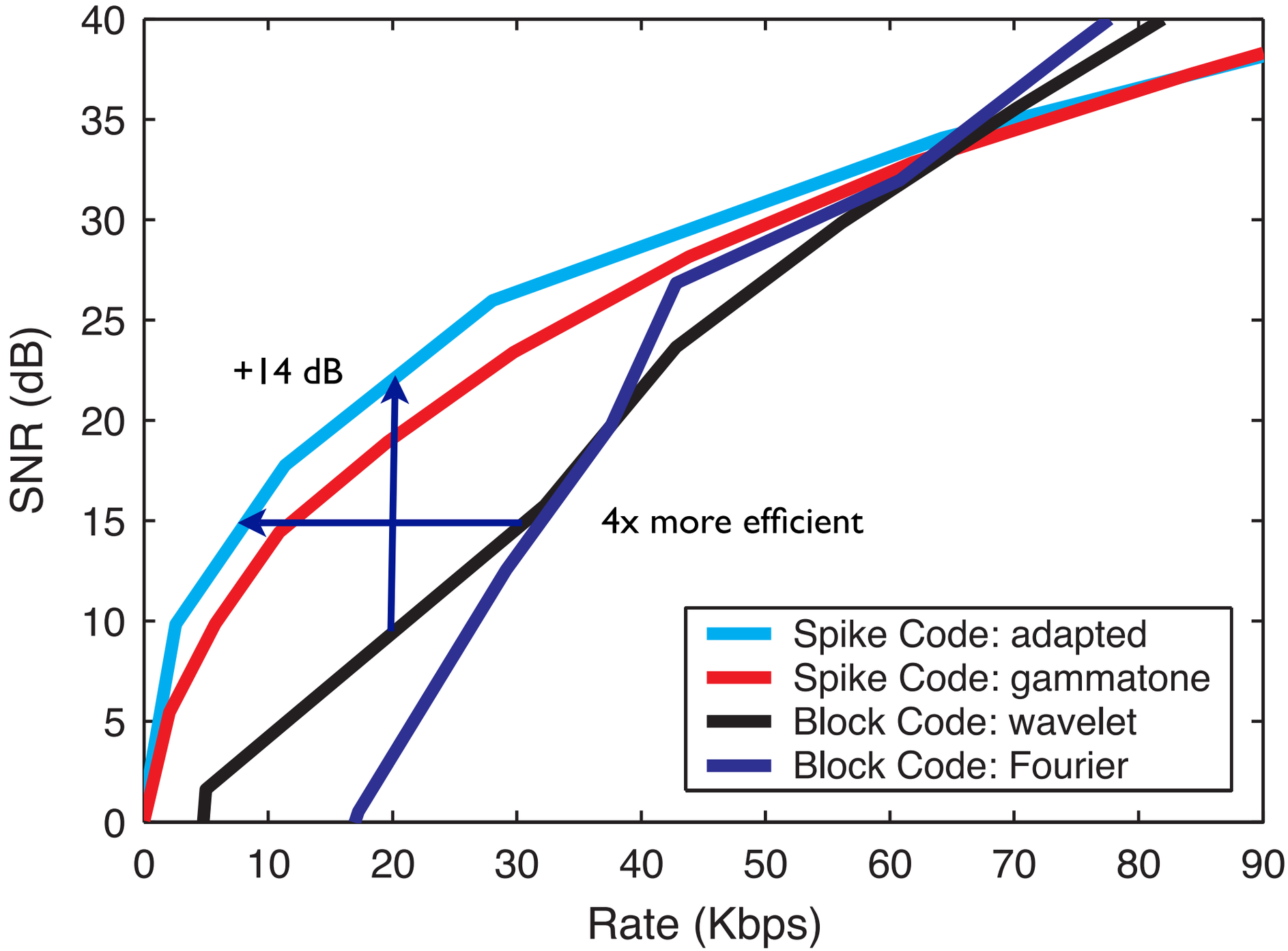
spike code

original

reconstruction

residual error

Coding efficiency curves



Using efficient coding theory to make theoretical predictions

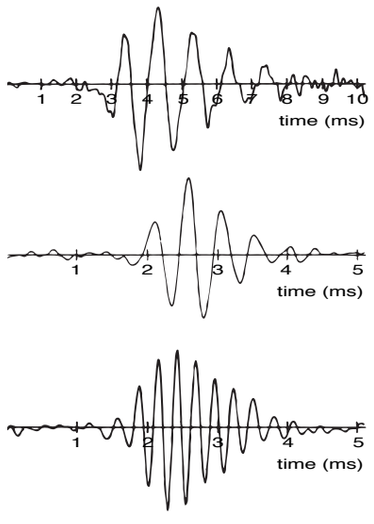
Natural Sound Environment

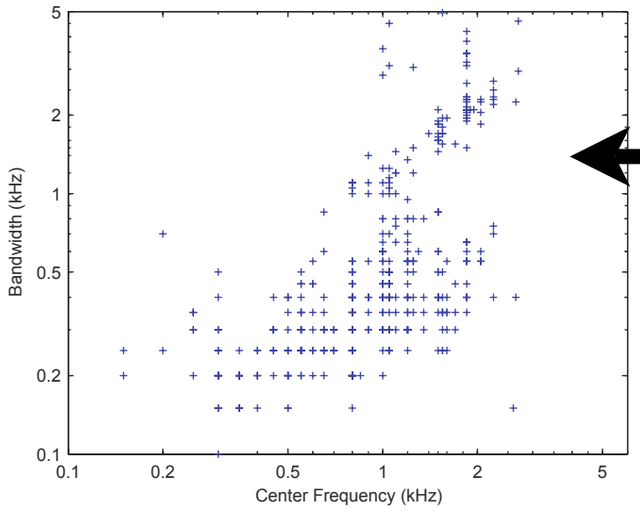
evolution

theory

physiological data:
- auditory nerve filter shapes
- population trends

optimal kernels:
- properties
- coding efficiency



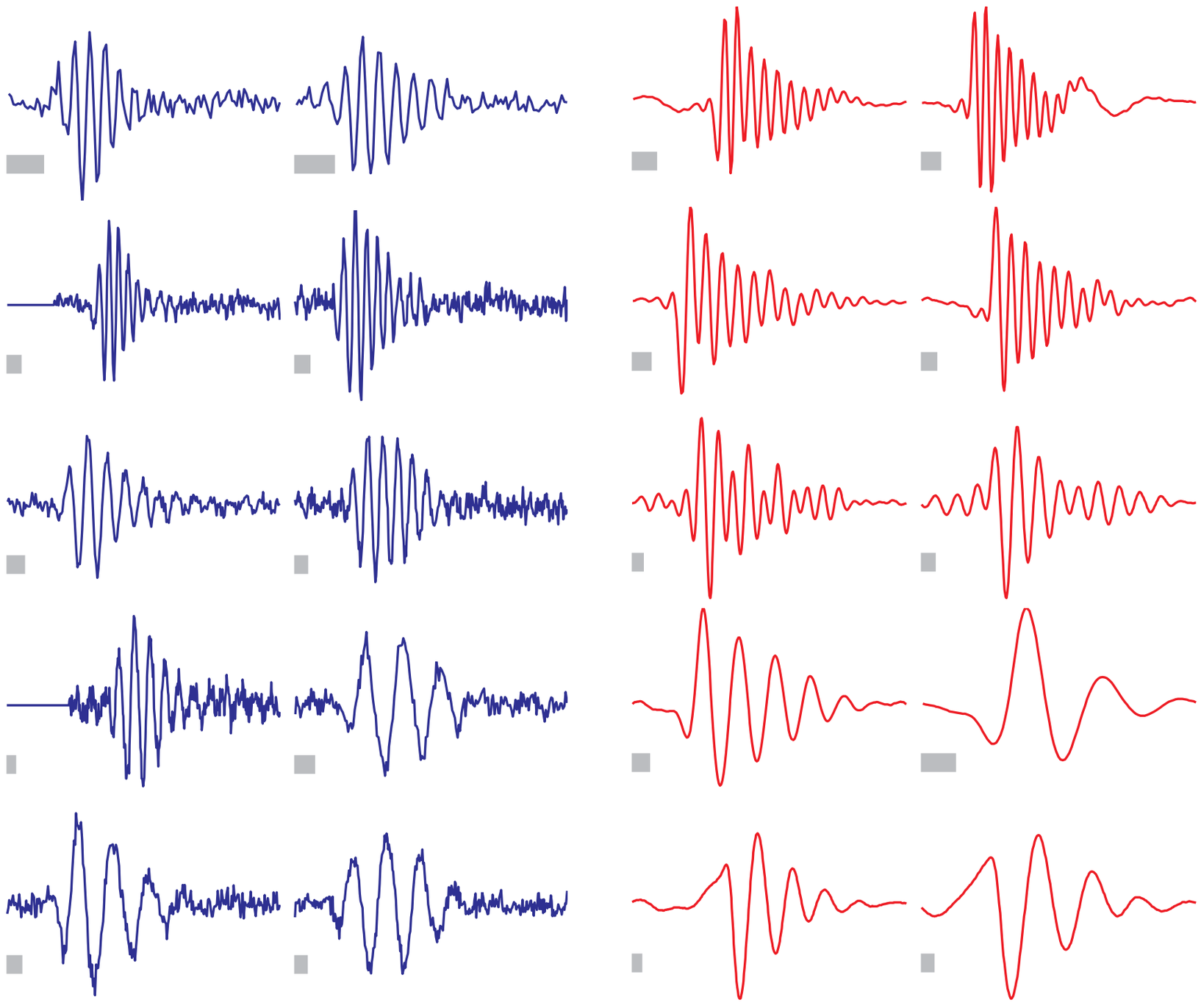


?

only compare to the data *after* optimizing
we do not fit the data

Prediction depends on sound ensemble.

Learned kernels share features of auditory nerve filters



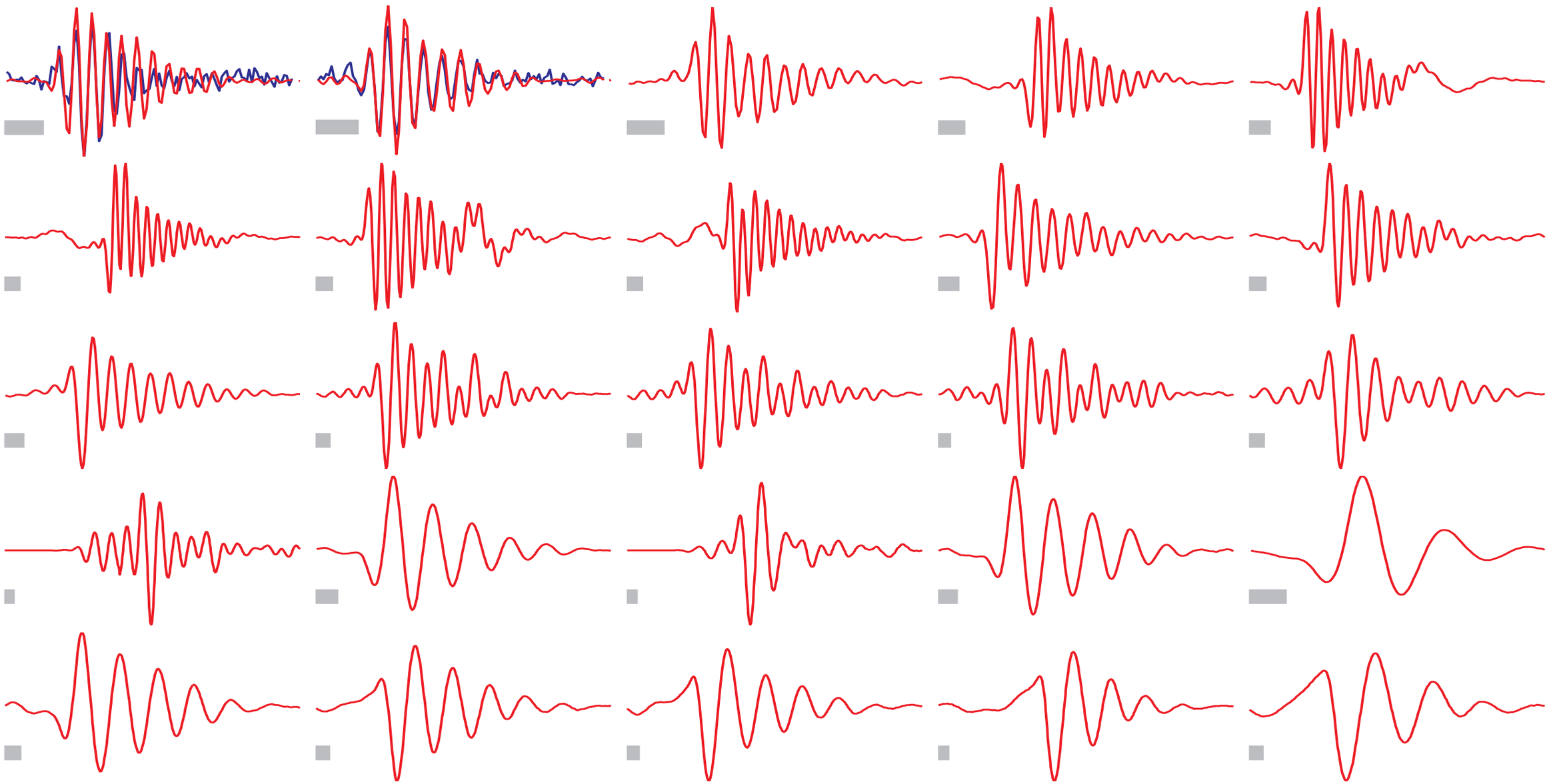
Auditory nerve filters

from Carney, McDuffy, and Shekhter, 1999

Optimized kernels

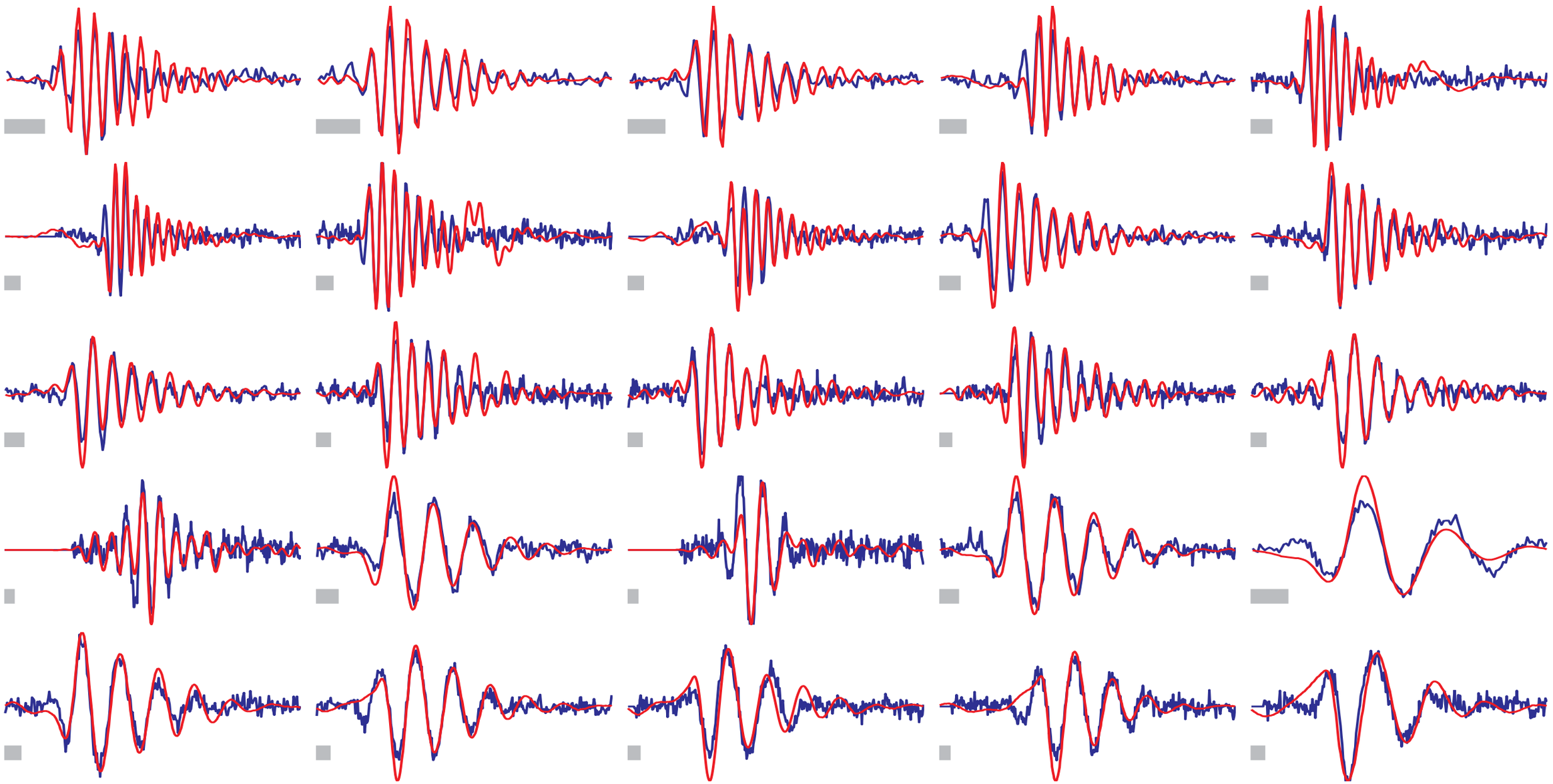
scale bar = 1 msec

Learned kernels closely match individual auditory nerve filters



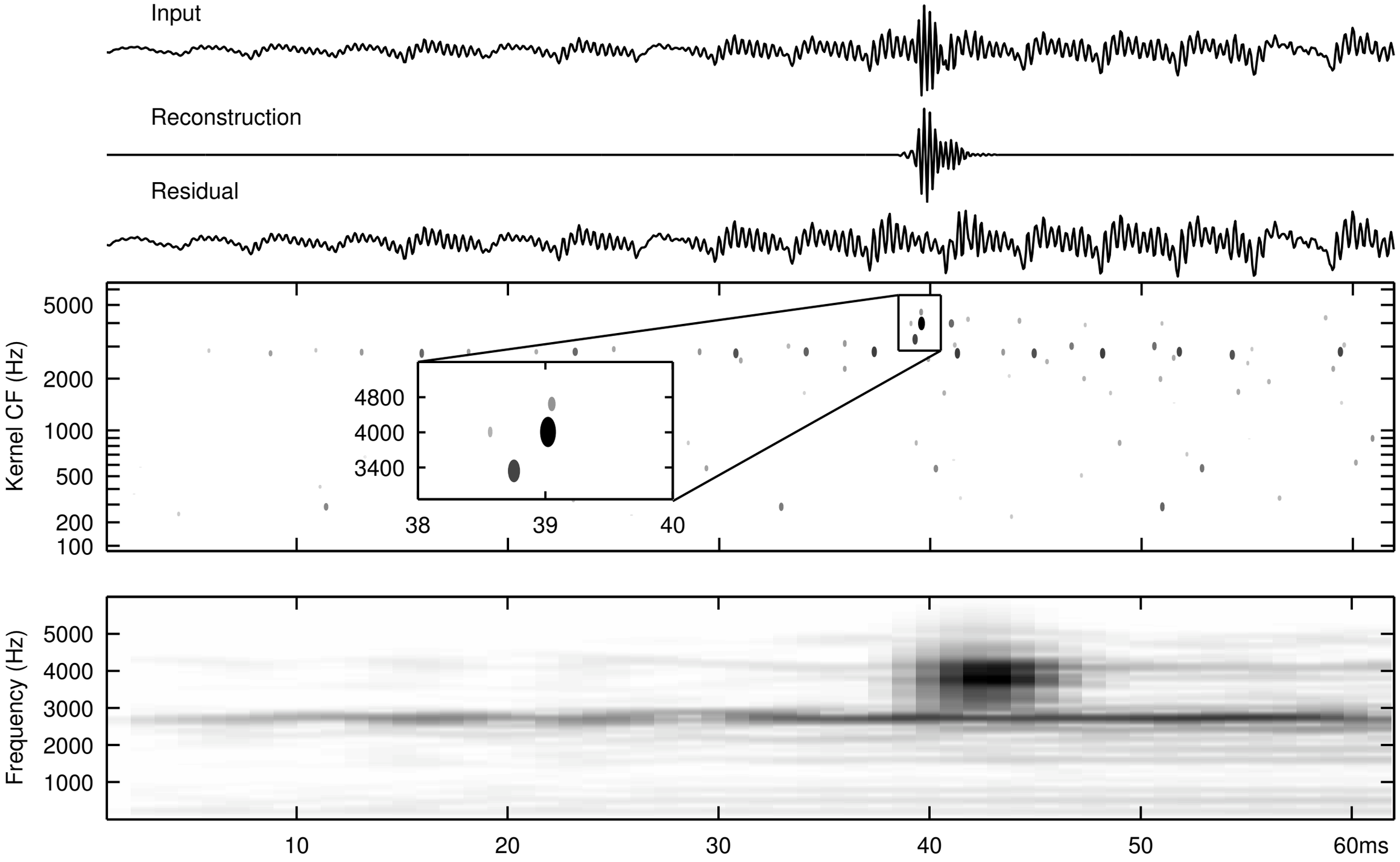
for each kernel find closet matching auditory nerve filter
in Laurel Carney's database of ~100 filters.

Learned kernels overlaid on selected auditory nerve filters

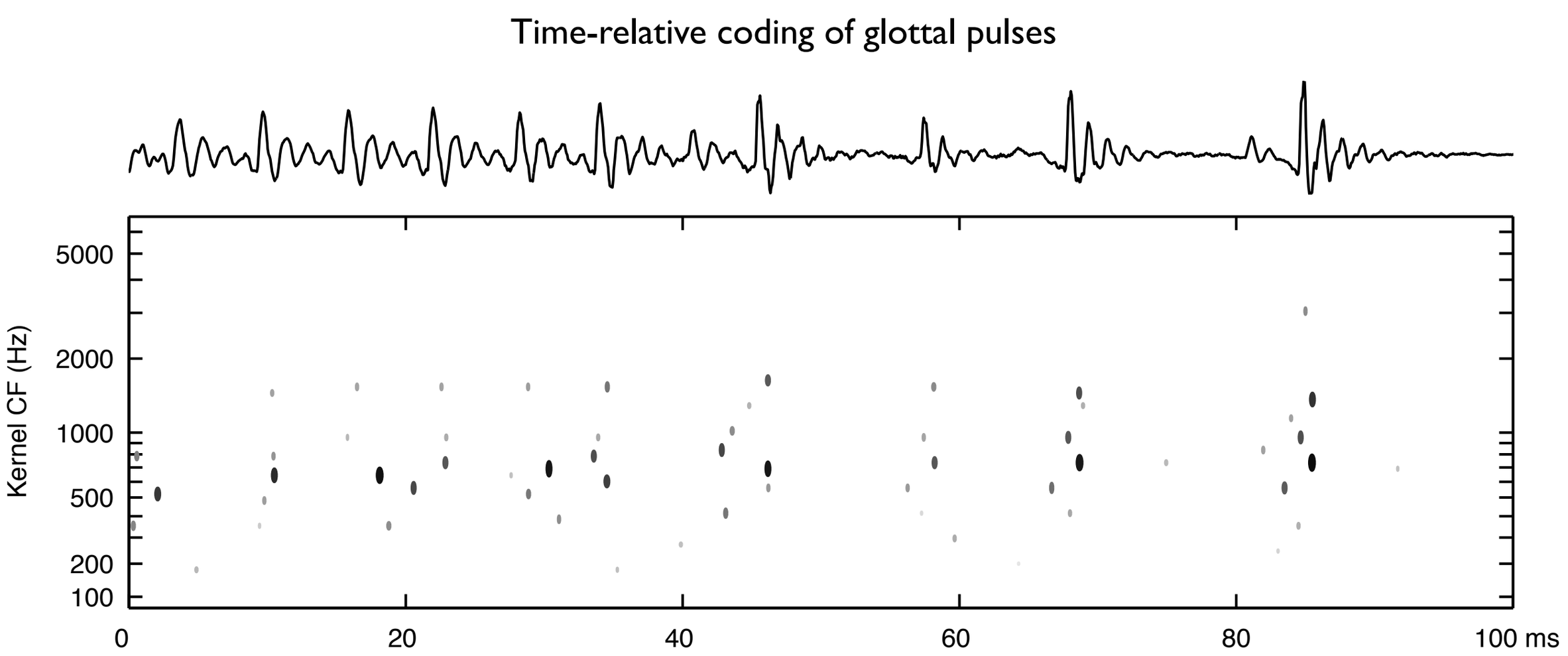


For almost all learned kernels there is a closely matching auditory nerve filter.

Coding of a speech consonant

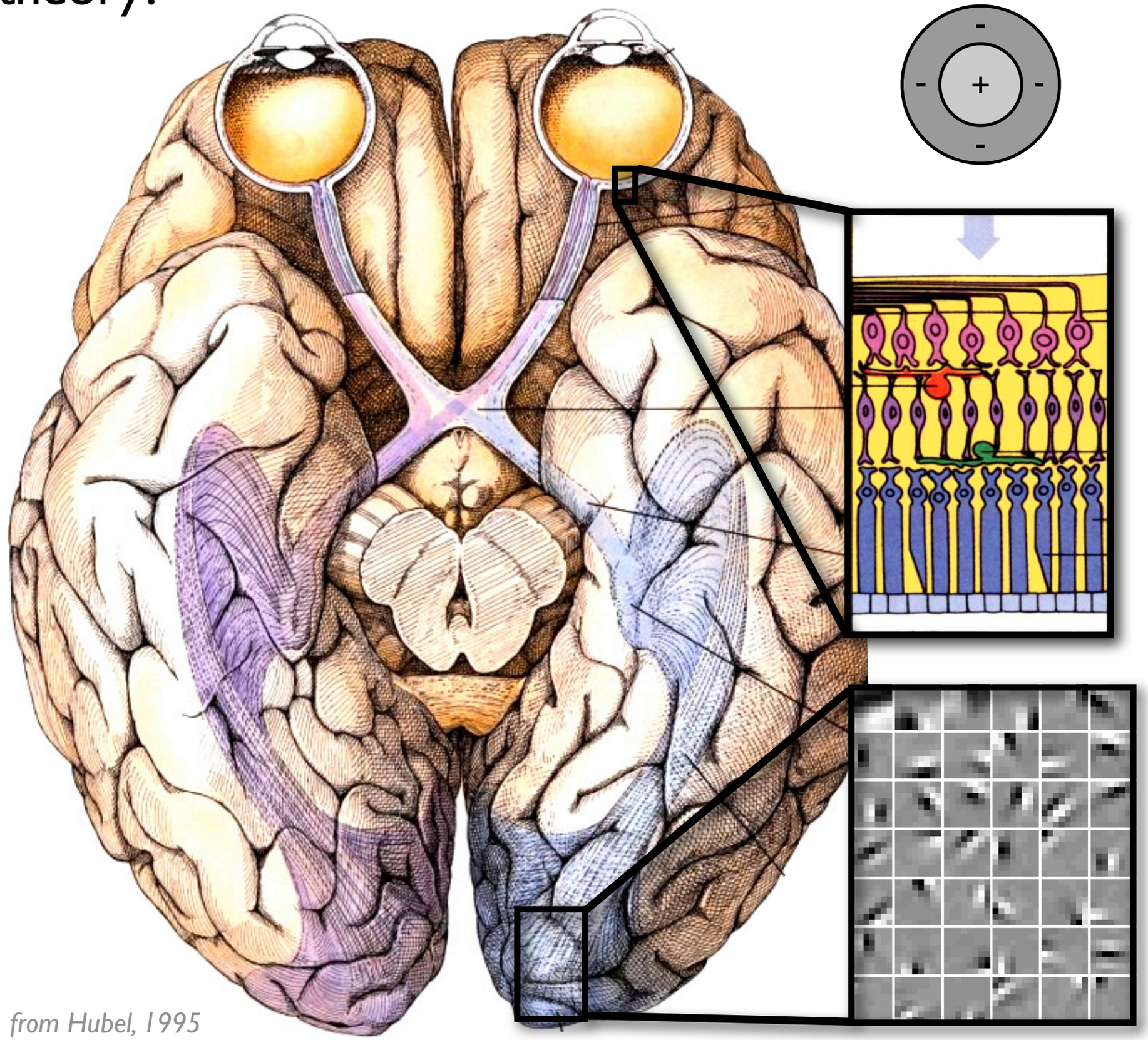


How is this achieving an efficient, time-relative code?

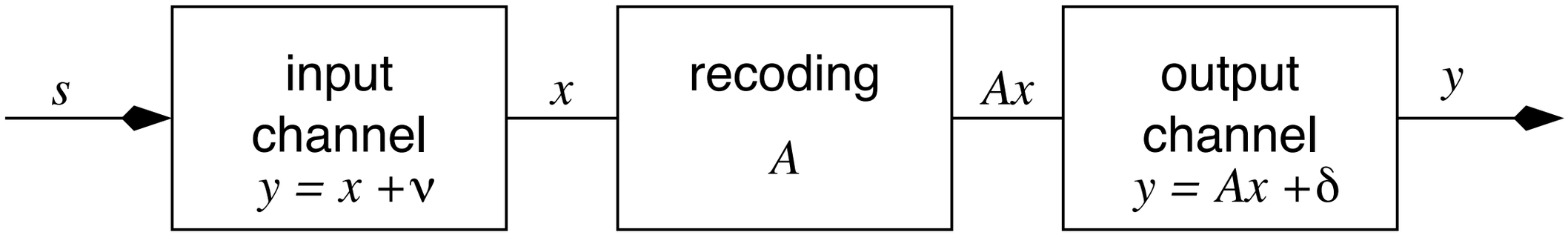


Theory	• explains data from principles • requires idealization and abstraction
Principle	• code signals accurately and efficiently • adapted to natural sensory environment
Idealization	• analog spikes
Methodology	• information theory, natural sounds • optimization
Prediction	• explains individual receptive fields • explains population organization

A gap in the theory?



Redundancy reduction for noisy channels (Atick, 1992)

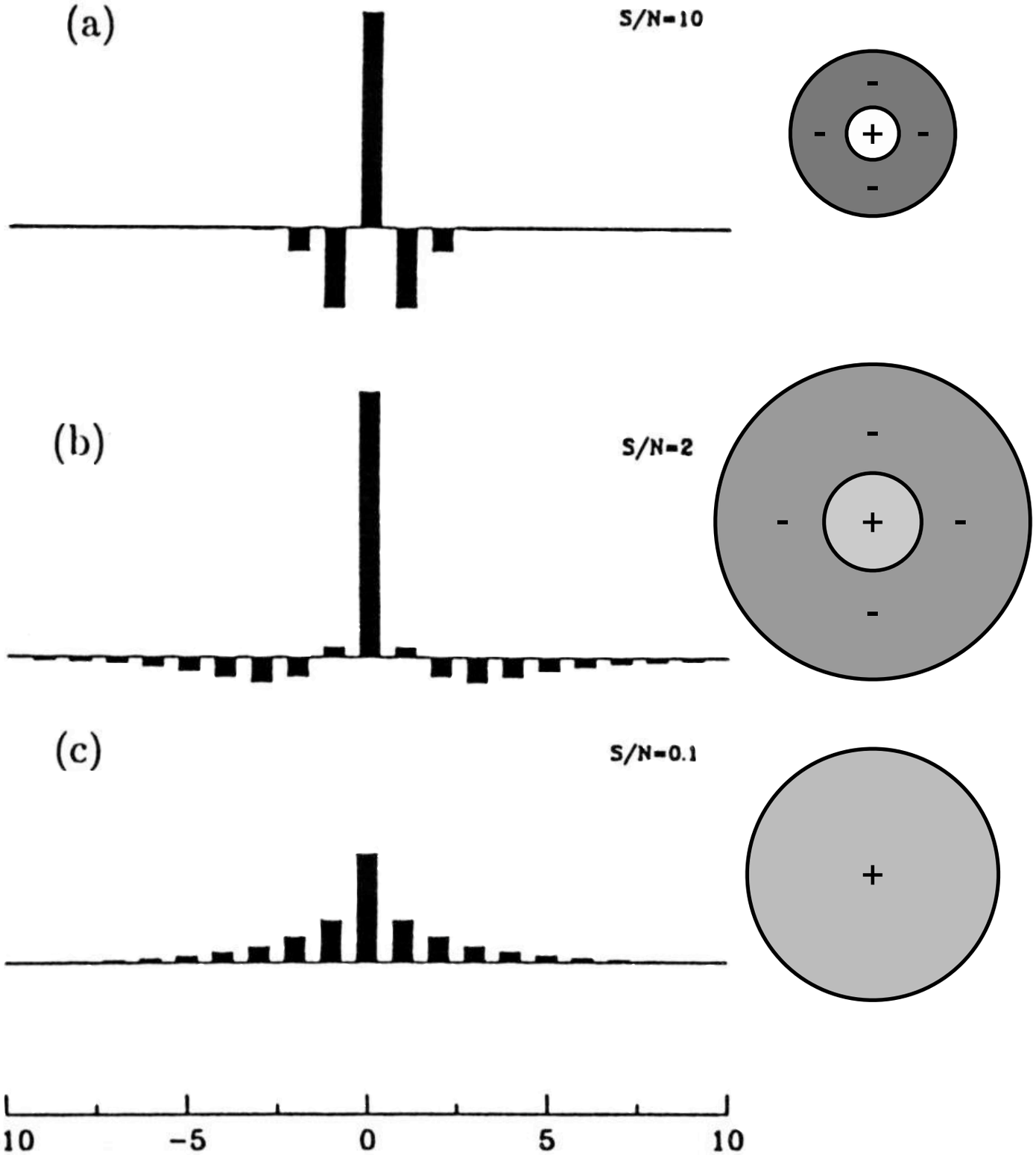


Mutual information

$$I(x,s) = \sum_{s,x} P(x,s) \log_2 \left[\frac{P(x,s)}{P(s)P(x)} \right]$$

$I(x,s) = 0$ iff $P(x,s) = P(x)P(s)$, i.e. x and s are independent.

Profiles of optimal filters

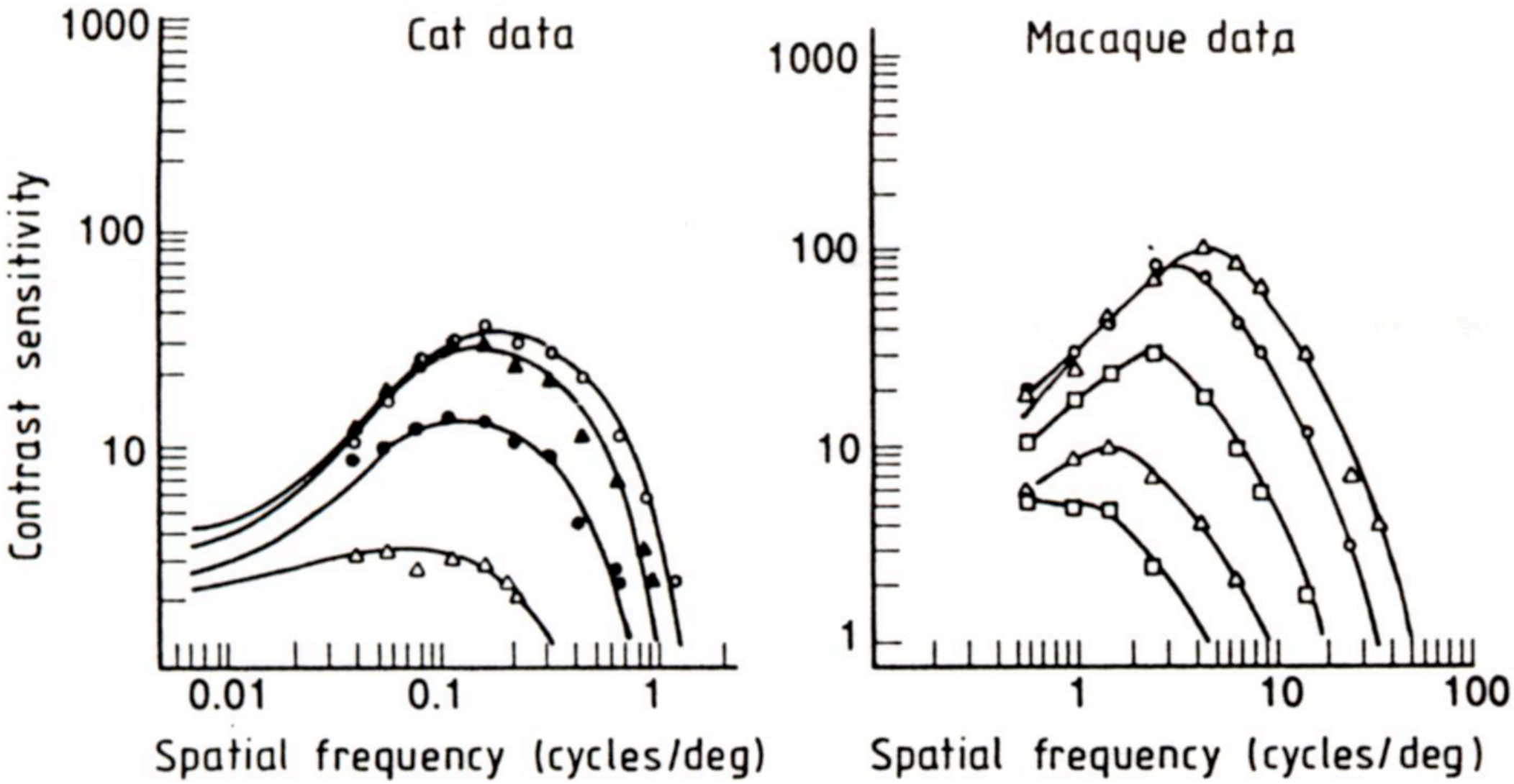


- high SNR
  - reduce redundancy
  - center-surround

- low SNR
  - average
  - low-pass filter

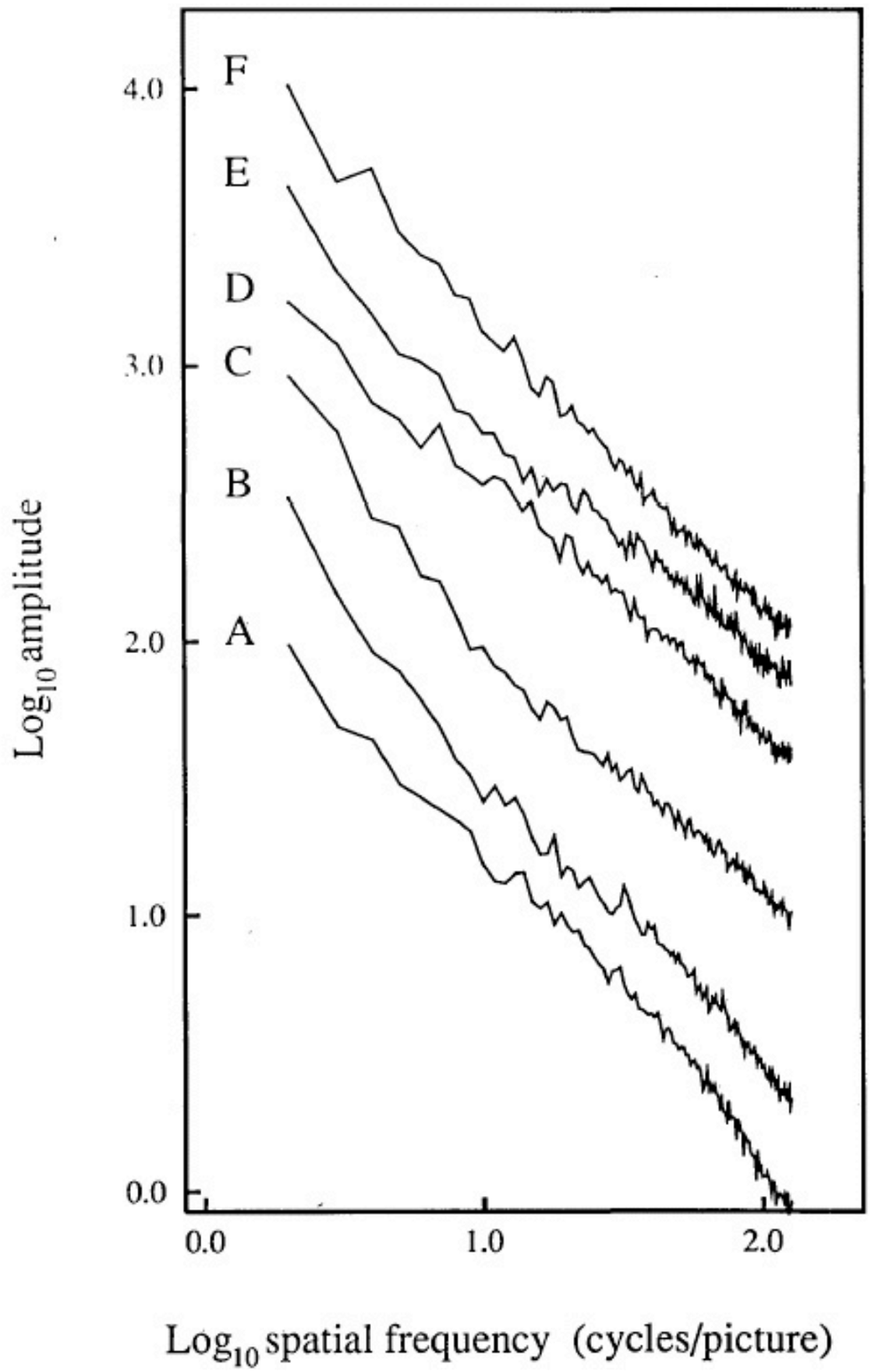
- matches behavior of retinal ganglion cells

An observation: Contrast sensitivity of ganglion cells



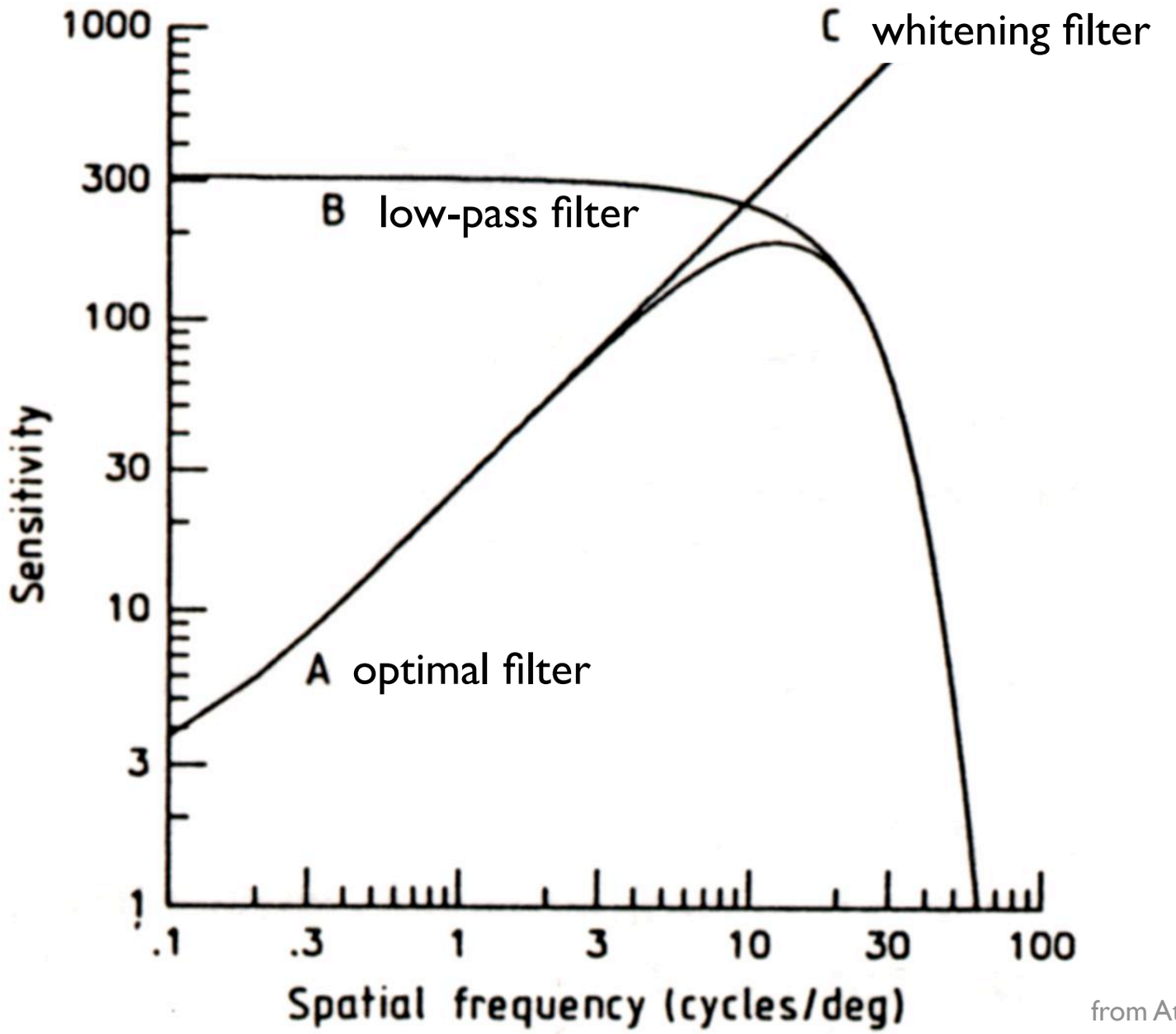
Luminance level decreases one log unit each time we go to lower curve.

Natural images have a 1/f amplitude spectrum



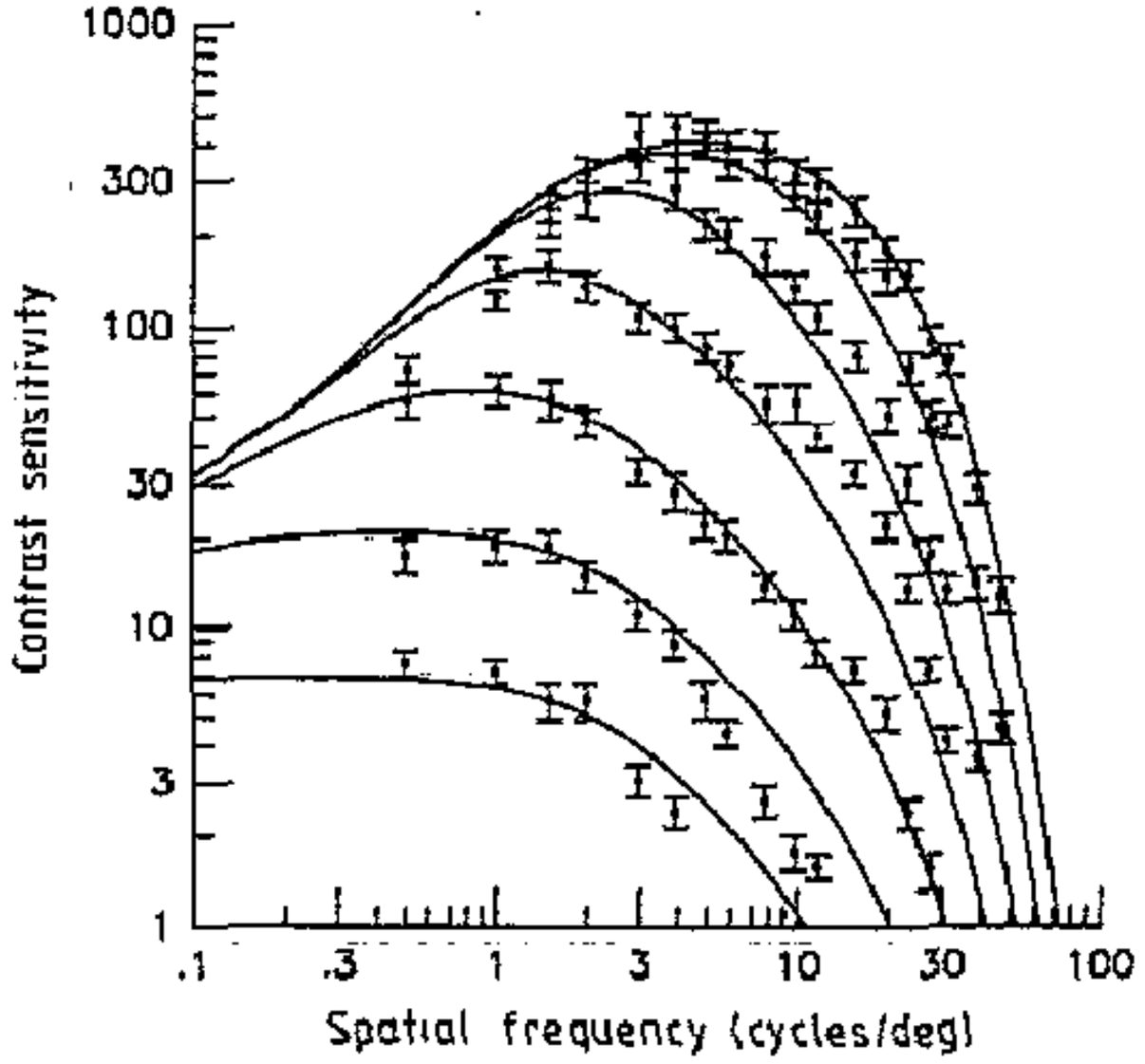
Field, 1987

Components of predicted filters



from Atick, 1992

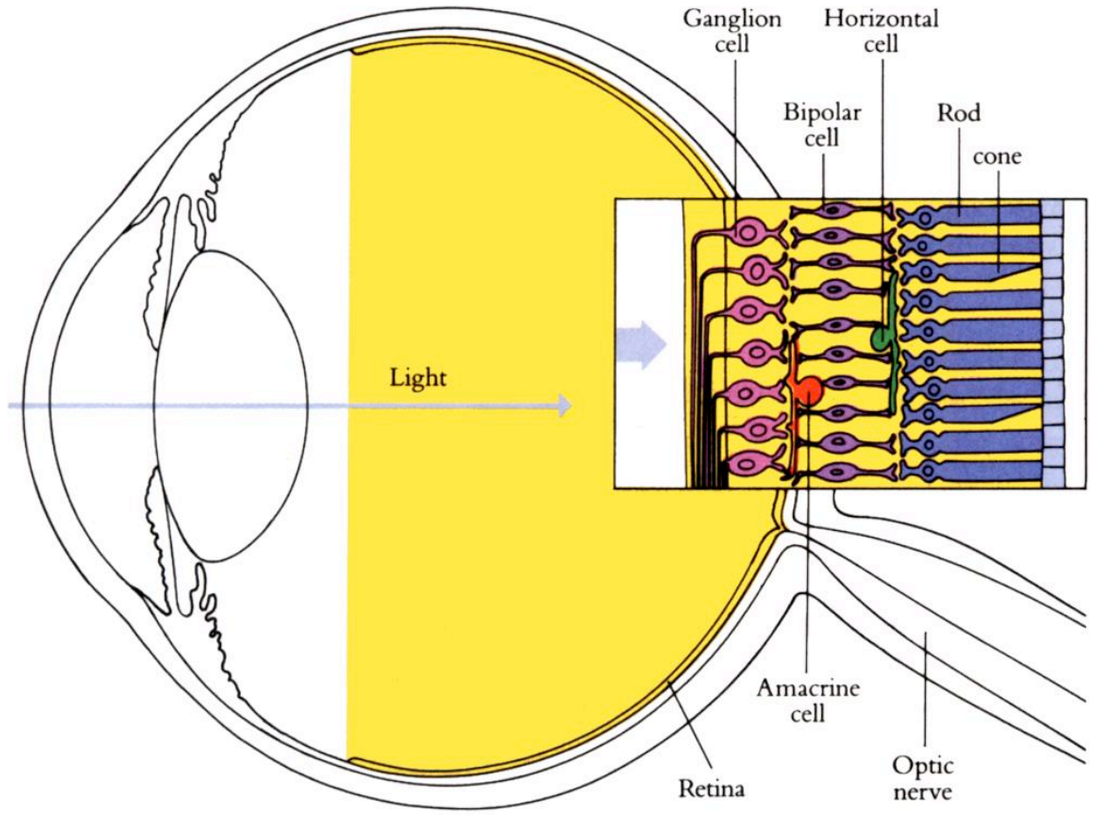
Predicted contrast sensitivity functions match neural data



from Atick, 1992

Robust coding of natural images

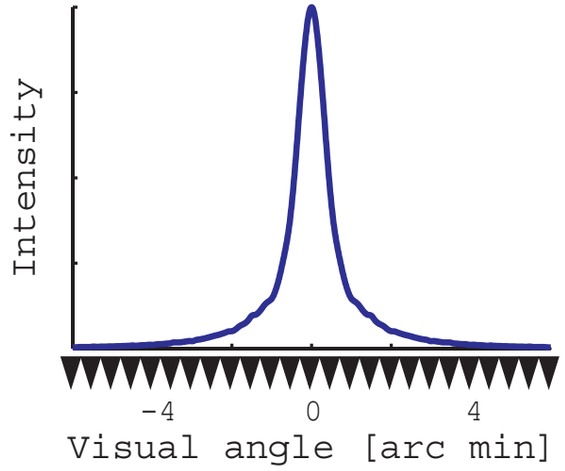
- Theory refined:
  - image is noisy and blurred
  - neural population size changes
  - neurons are noisy



from Hubel, 1995

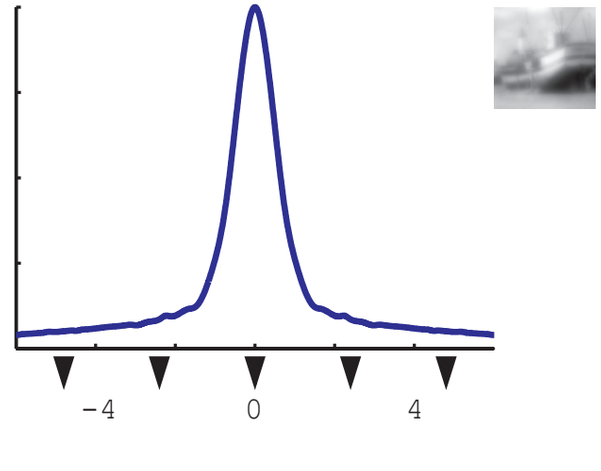
(a) Undistorted image

(b) Fovea



retinal image

(c) 40 degrees eccentricity



from Doi and Lewicki, 2006

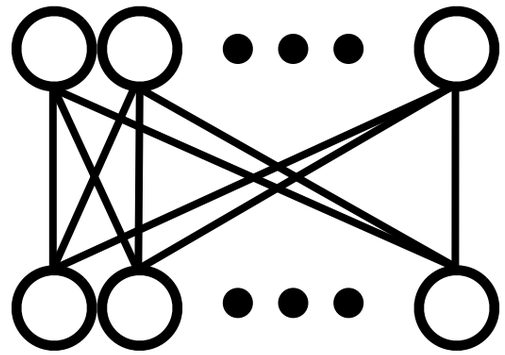
Problem 1: Real neurons are "noisy"

Estimates of neural information capacity

system (area)	**stimulus**	**bits / sec**	**bits / spike**
fly visual (H1)	motion	64	~1
monkey visual (MT)	motion	5.5 - 12	0.6 - 1.5
frog auditory (auditory nerve)	noise & call	46 & 133	1.4 & 7.8
Salamander visual (ganglinon cells)	rand. spots	3.2	1.6
cricket cercal (sensory afferent)	mech. motion	294	3.2
cricket cercal (sensory afferent)	wind noise	75 - 220	0.6 - 3.1
cricket cercal (10-2 and 10-3)	wind noise	8 - 80	avg. = 1
Electric fish (P-afferent)	amp. modulation	0 - 200	0 - 1.2

Limited capacity ⇒ neural codes need to be *robust.*

Traditional codes are *not* robust

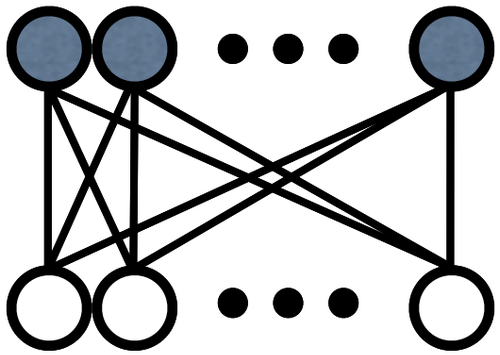


Original

Traditional codes are *not* robust

encoding neurons

sensory input



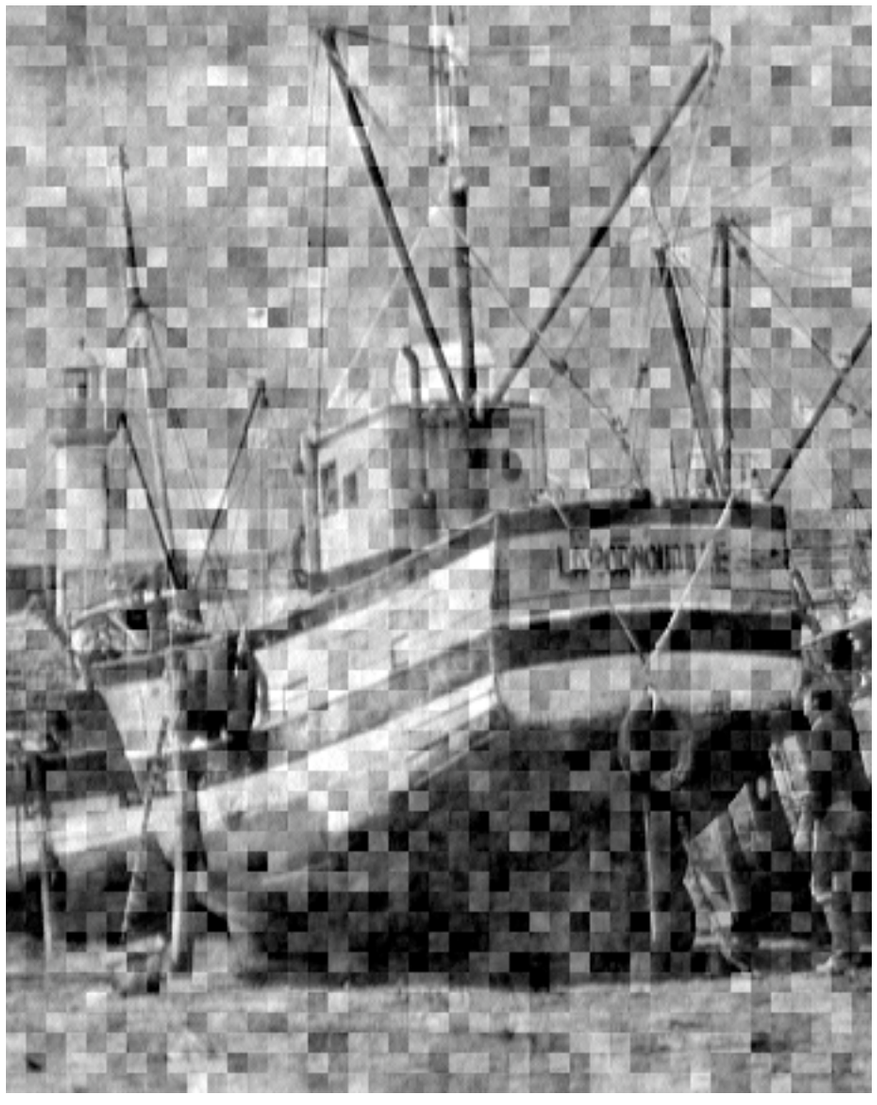
Add
noise equivalent
to 1 bit precision

Original

1x efficient coding

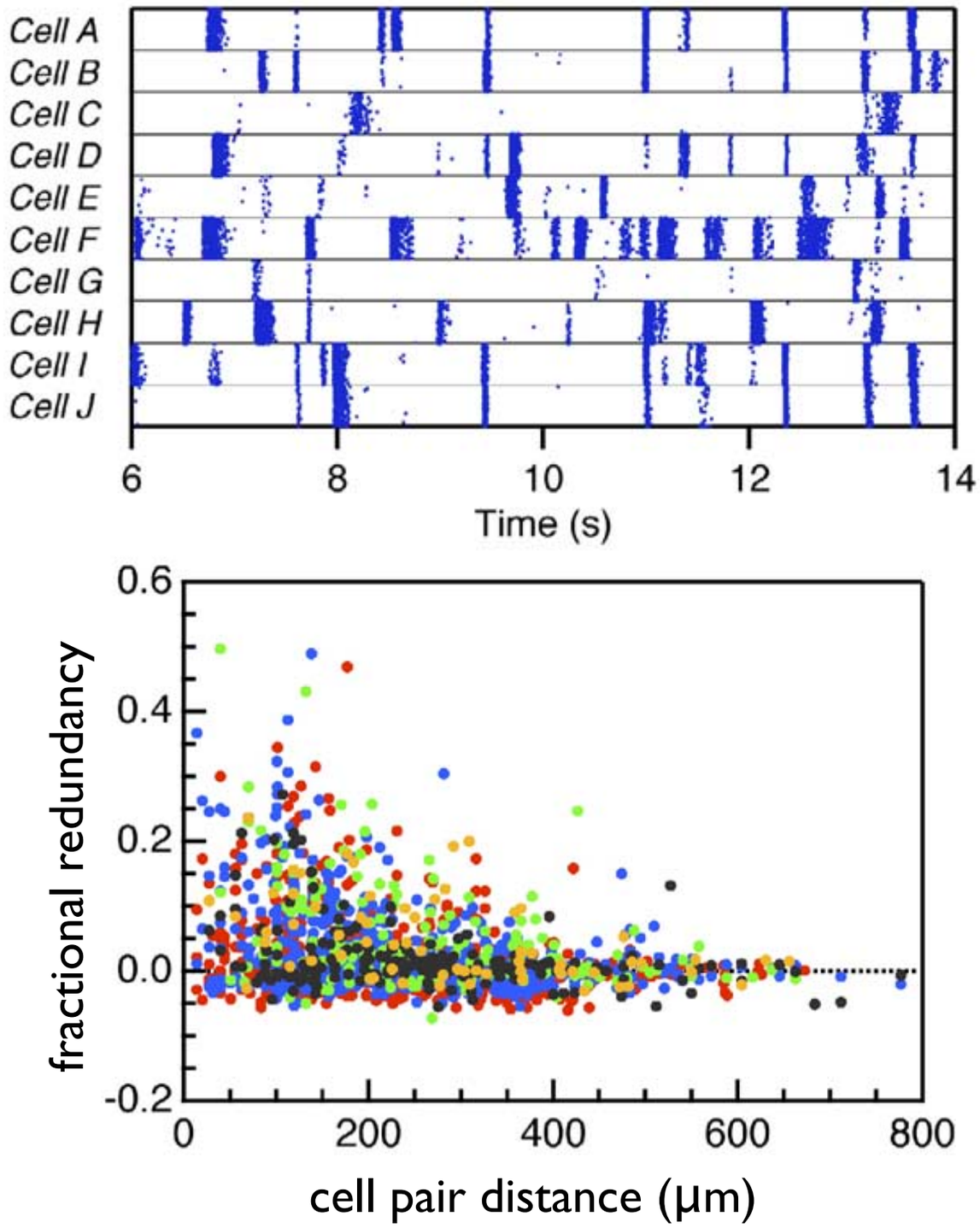
1 bit precision

reconstruction (34% error)



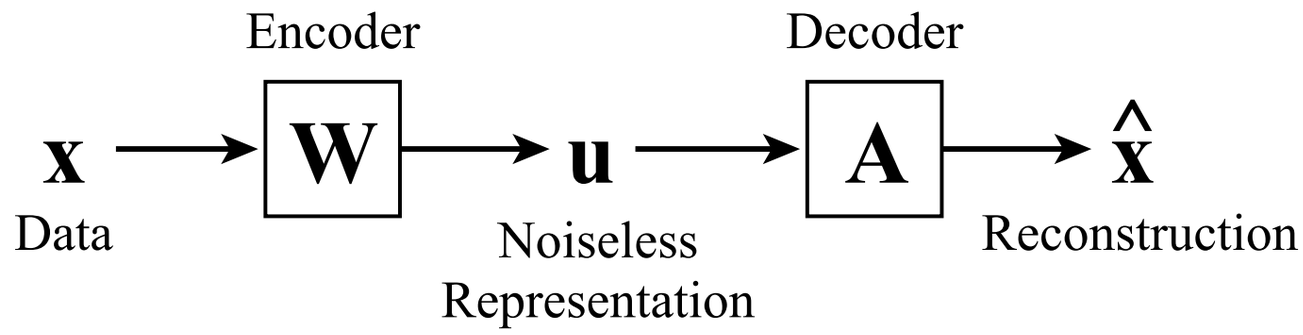
Redundancy plays an important role in neural coding

Response of salamander retinal ganglion cells to natural movies

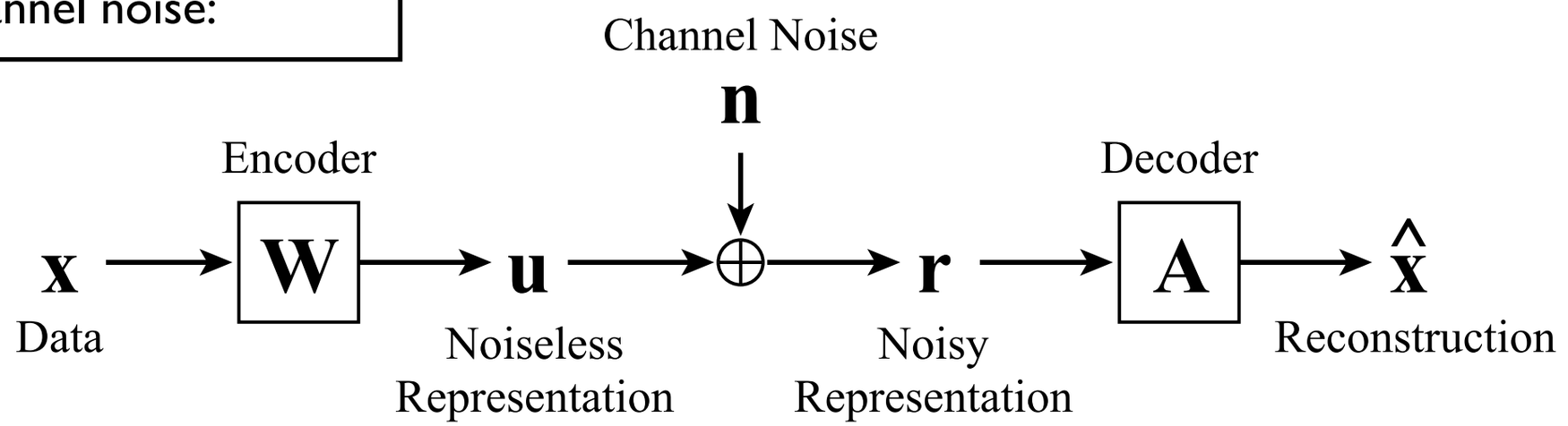


From Puchalla et al, 2005

Robust coding (Doi and Lewicki, 2005,2006)



Model limited precision using additive channel noise:

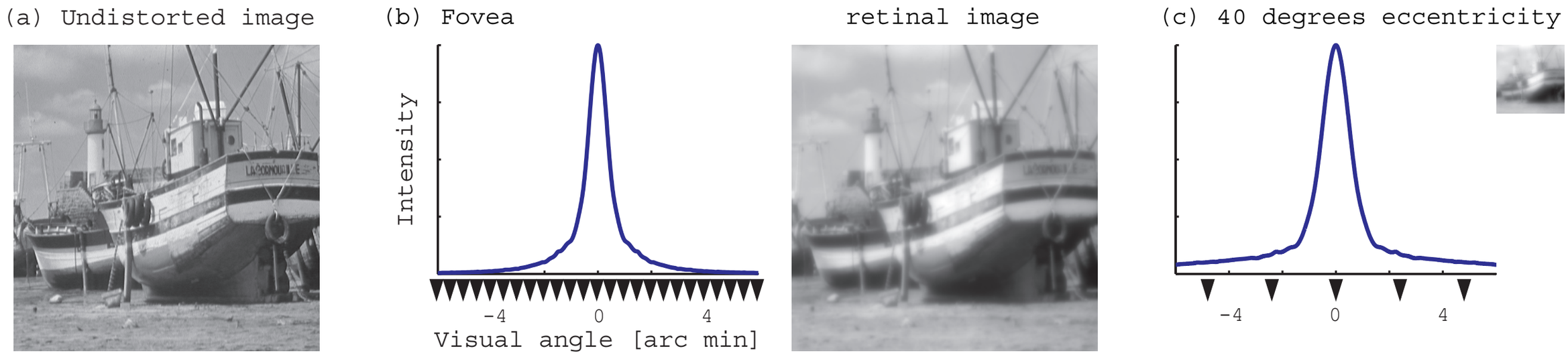


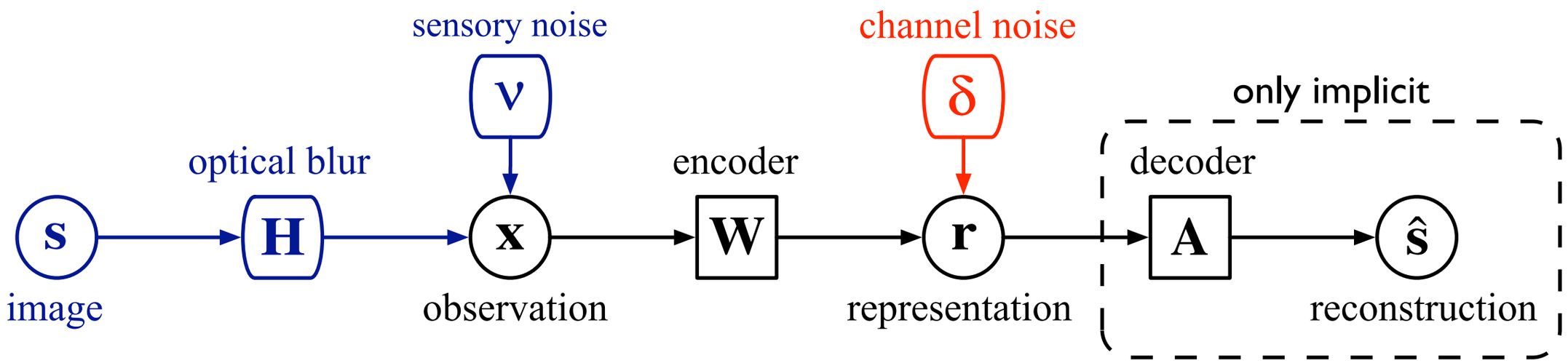
$$\mathbf{r} = \mathbf{W}\mathbf{x} + \mathbf{n} = \mathbf{u} + \mathbf{n}.$$

$$\hat{\mathbf{x}} = \mathbf{A}\mathbf{r} = \mathbf{A}\mathbf{W}\mathbf{x} + \mathbf{A}\mathbf{n}.$$

Caveat: linear, but can evaluate for different noise levels.

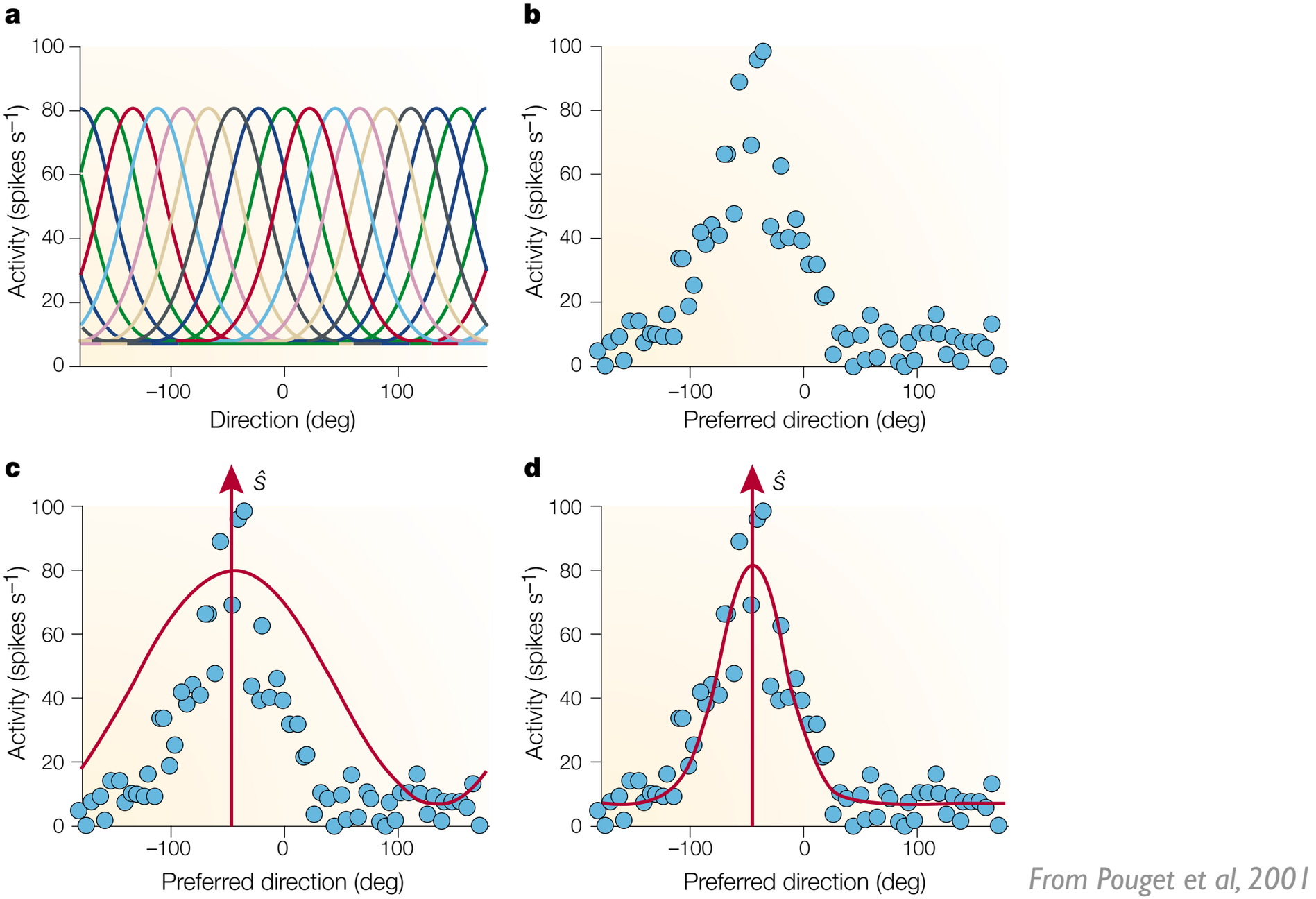
Generalizing the model: sensory noise and optical blur





Can also add sparseness and resource constraints

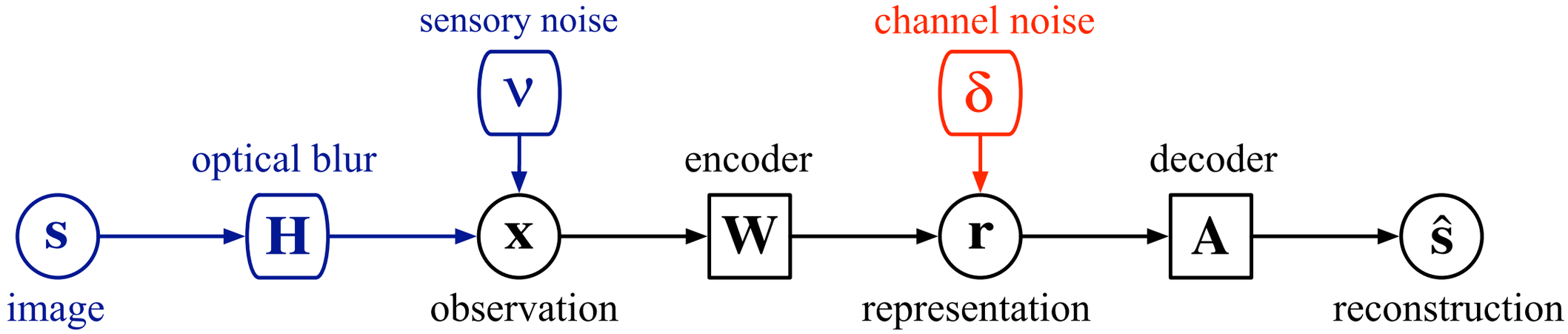
Robust coding is distinct from "reading" noisy populations



From Pouget et al, 2001

Here, we want to *learn* an optimal image code using noisy neurons.

How do we learn robust codes?



Objective:

Find **W** and **A** that minimize reconstruction error.

- Channel capacity of the i[th] neuron:

$$C_i = \frac{1}{2}\ln(\mathrm{SNR}_i + 1)$$

- To limit capacity, fix the coefficient signal to noise ratio:

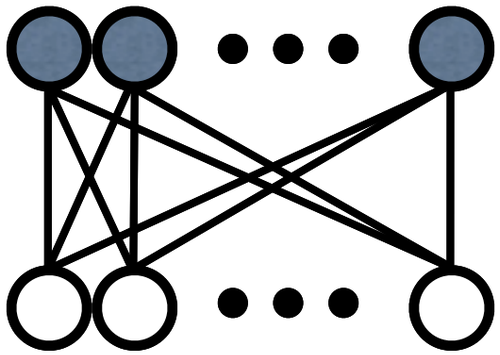
$$\mathrm{SNR}_i = \frac{\langle u_i^2 \rangle}{\sigma_n^2}$$

Now robust coding is formulated as a *constrained optimization problem*.

Robust coding of natural images

encoding neurons

sensory input



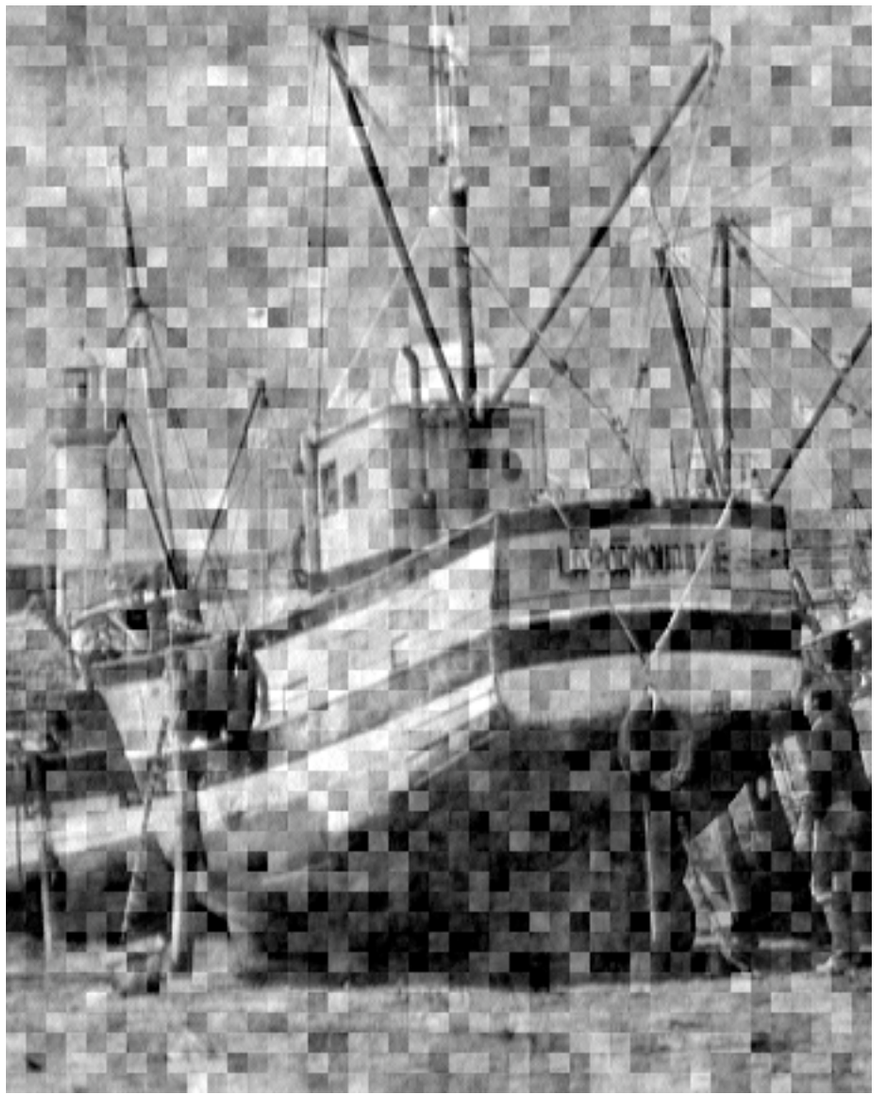
Add
noise equivalent
to 1 bit precision

Original

1x efficient coding

1 bit precision

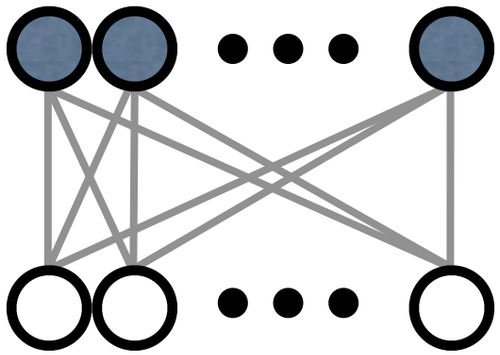
reconstruction (34% error)



Robust coding of natural images

encoding neurons

sensory input



Weights adapted for optimal robustness

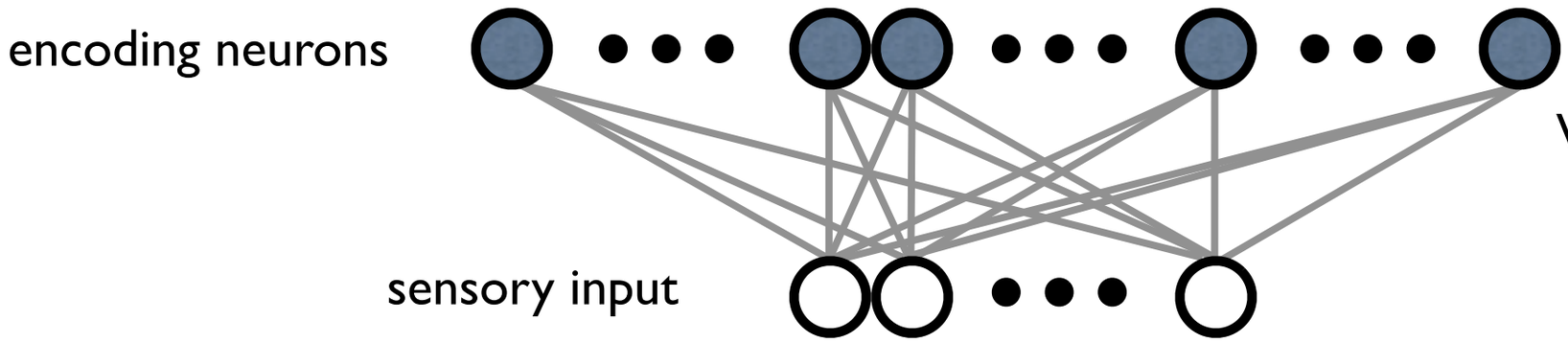
Original

1x robust coding

1 bit precision

reconstruction (3.8% error)

Reconstruction improves by adding neurons



Weights adapted for optimal robustness

Original

8x robust coding

1 bit precision

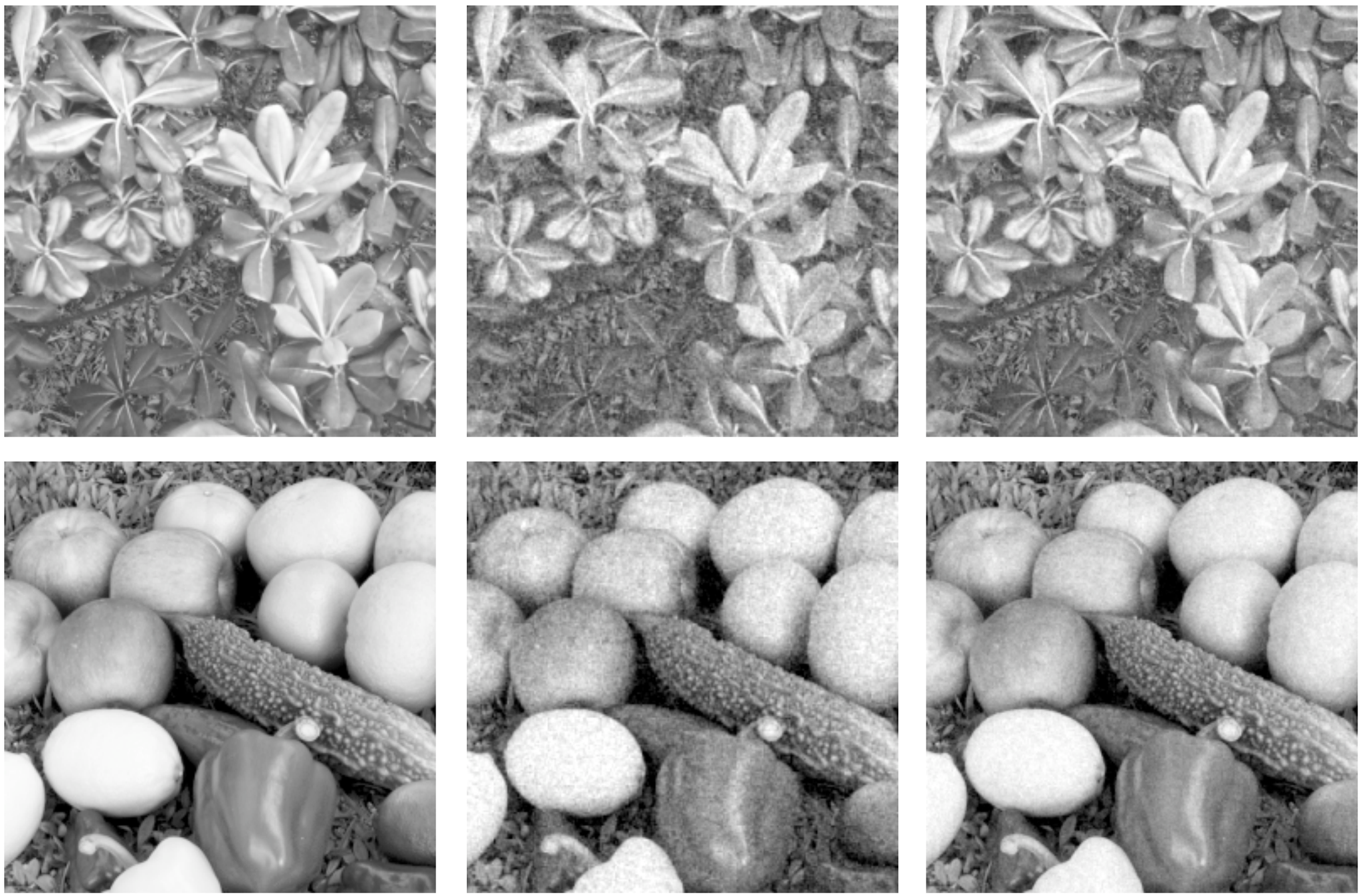
reconstruction error: 0.6%

Adding precision to 1x robust coding

original images

1 bit: 12.5% error

2 bit: 3.1% error



Can derive minimum theoretical average error bound

$$\mathcal{E} = \frac{1}{\frac{M}{N} \cdot \mathrm{SNR} + 1} \; \frac{1}{N} \left[\sum_{i=1}^{N} \sqrt{\lambda_i} \right]^2 \qquad \text{if } \mathrm{SNR} \geq \mathrm{SNR}_c$$

λ_i - ith eigenvalue of the data covariance

N - input dimensionality

M - # of coding units (neurons)

Algorithm achieves theoretical lower bound

	Results	Bound
0.5x	19.9%	20.3%
1x	12.4%	12.5%
8x	2.0%	2.0%

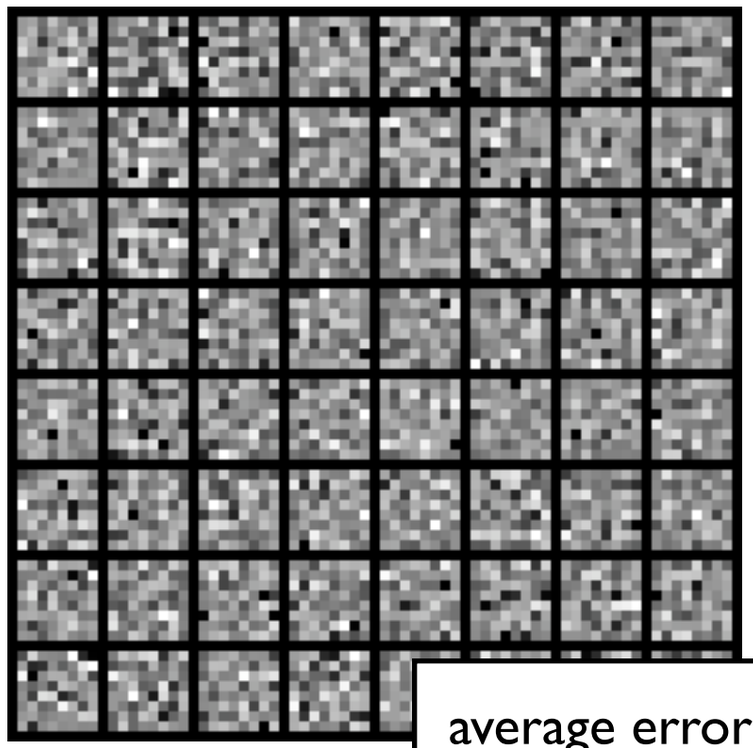
Balcan, Doi, and Lewicki, 2007; Balcan and Lewicki, 2007

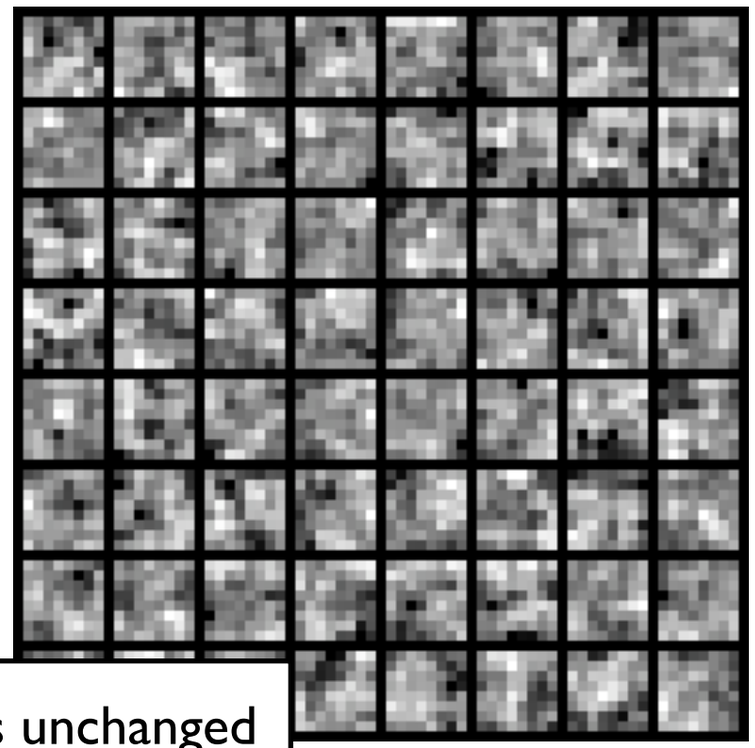
Sparseness localizes the vectors and increases coding efficiency

encoding vectors ($\mathbf{W}$)

decoding vectors ($\mathbf{A}$)

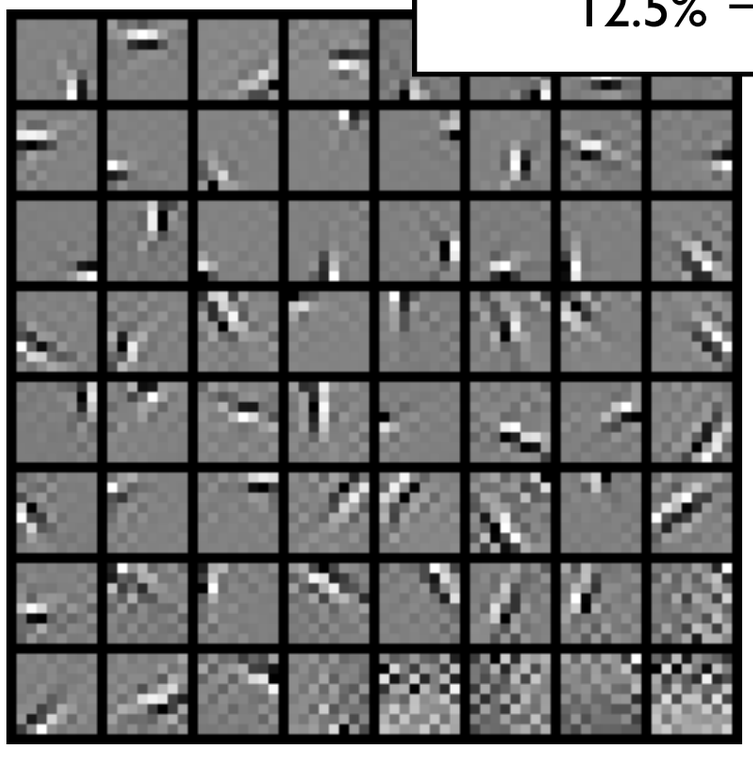
robust coding

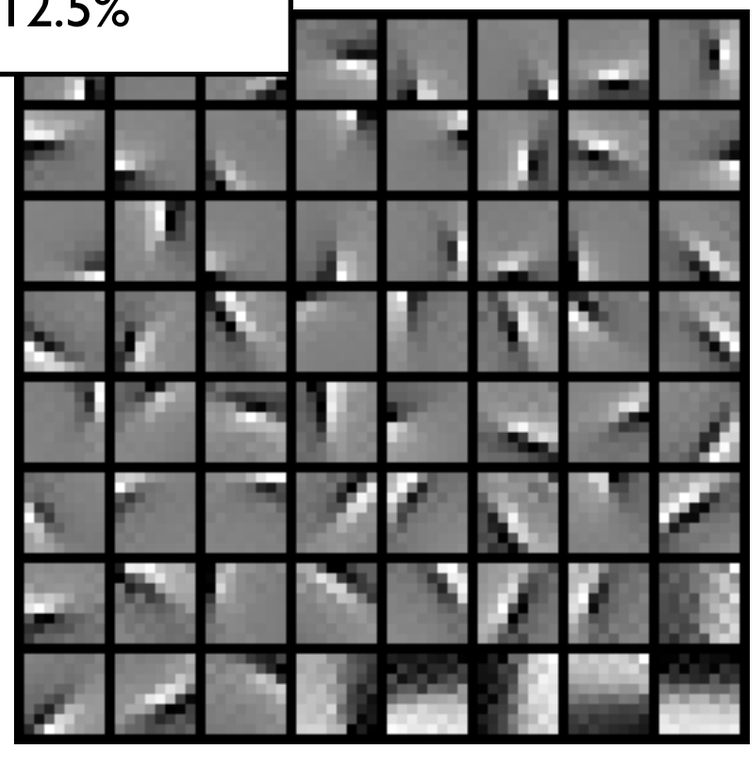




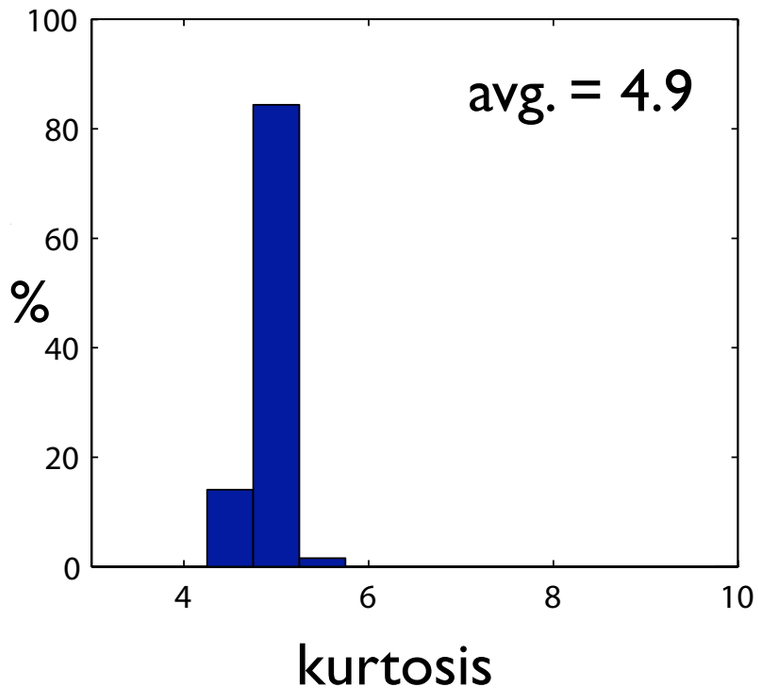
average error is unchanged
12.5% → 12.5%

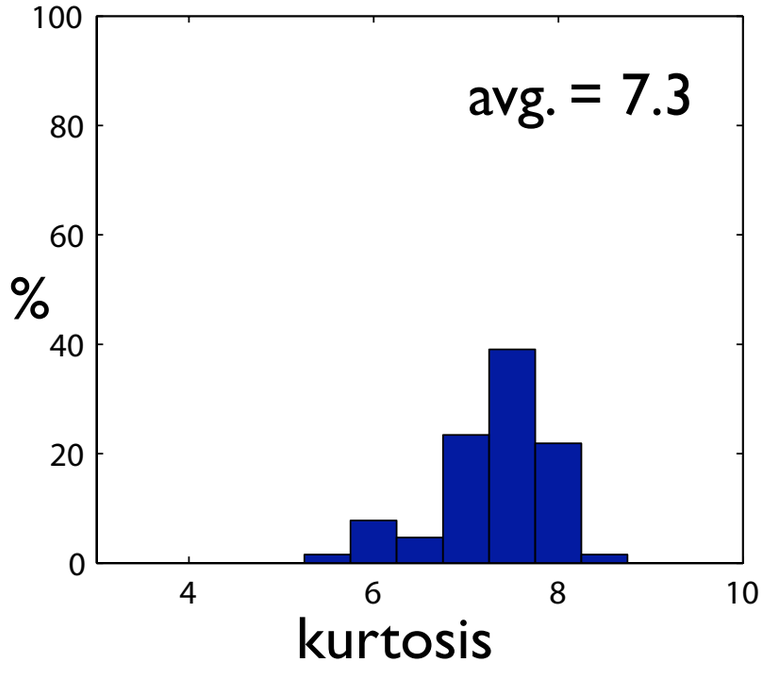
robust sparse coding





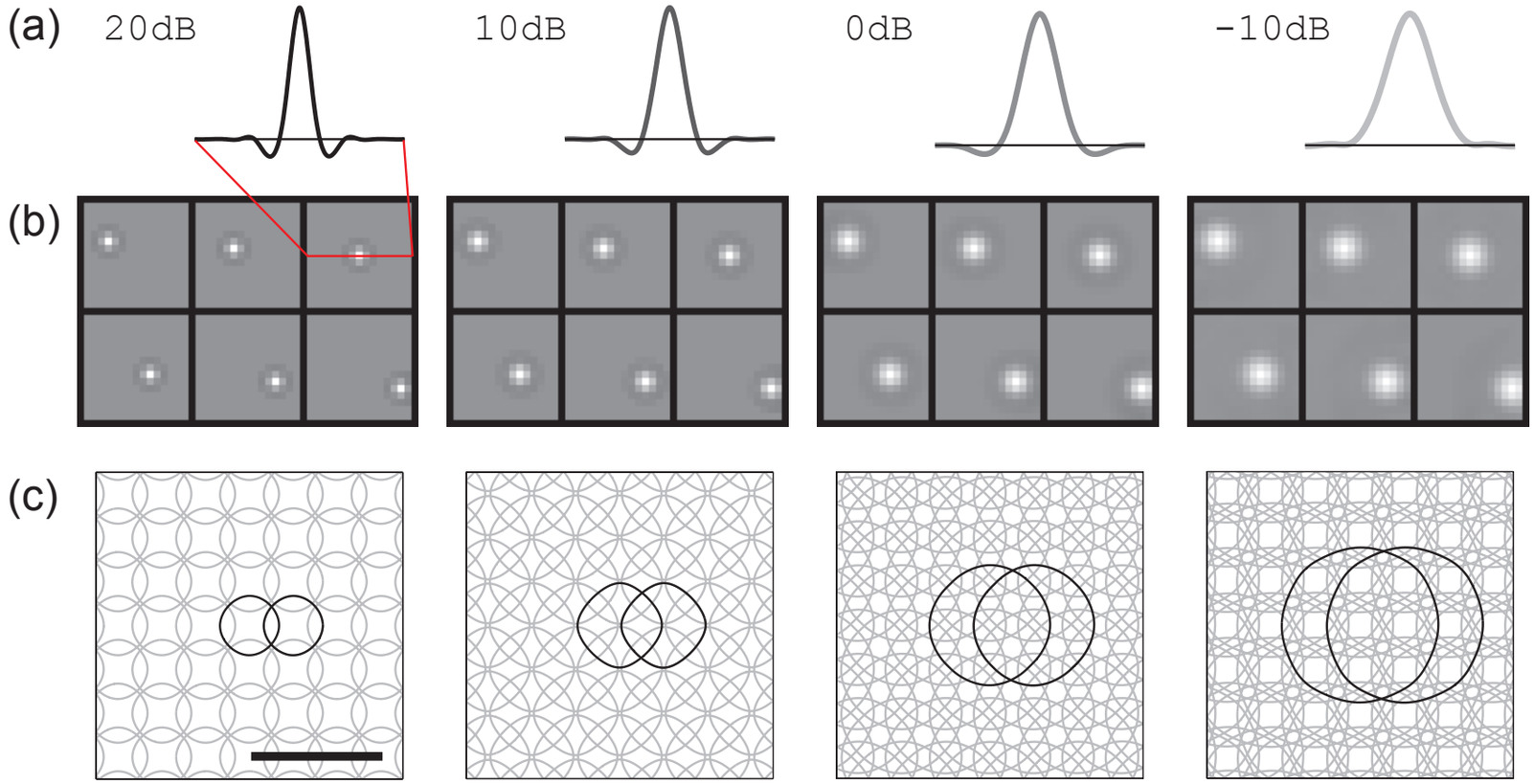
normalized histogram
of coeff. kurtosis





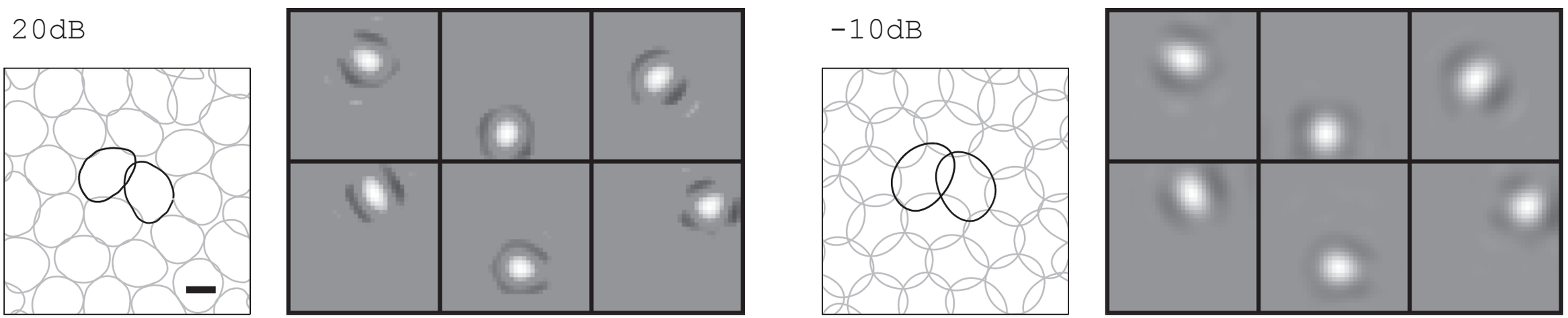
Non-zero resource constraints localize weights

fovea



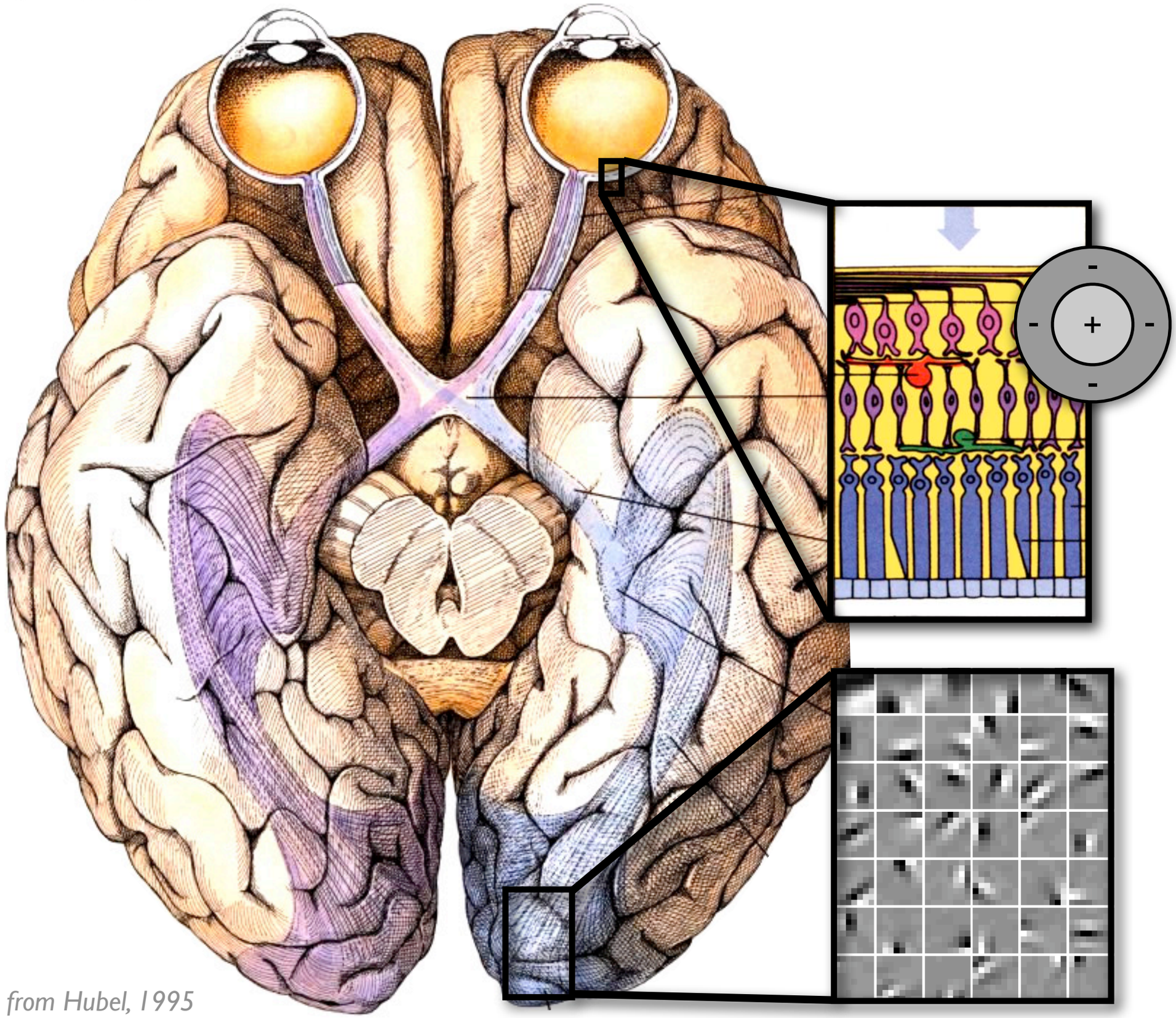
average error is also unchanged

40 degrees



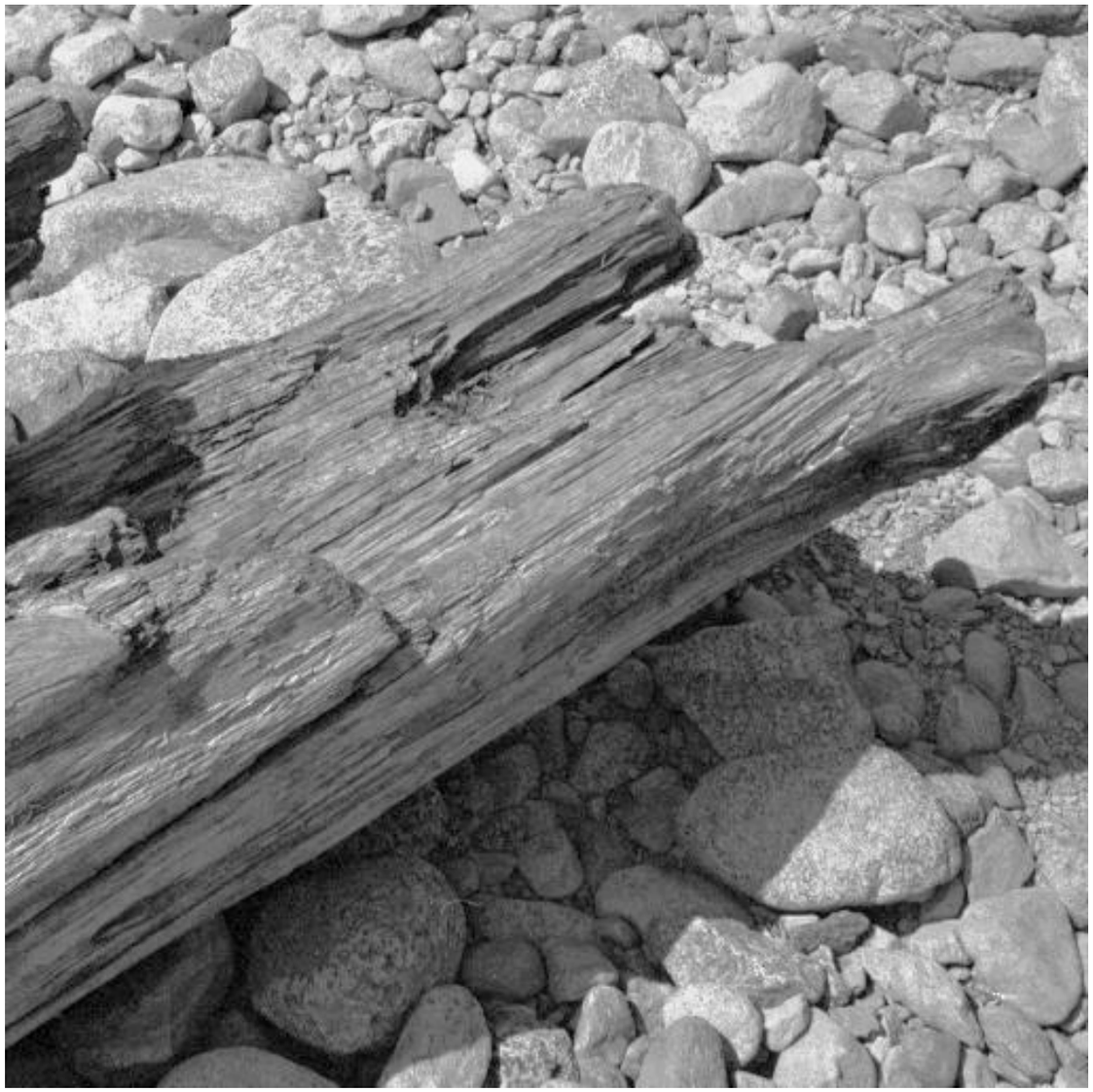
Theory	• explains data from principles • requires idealization and abstraction
Principle	• code signals accurately, efficiently, robustly • adapted to natural sensory environment
Idealization	• cell response is linear, noise is additive • simple, additive resource constraints
Methodology	• information theory, natural images • constrained optimization
Prediction	• explains individual receptive fields • explains non-linear response • explains population organization

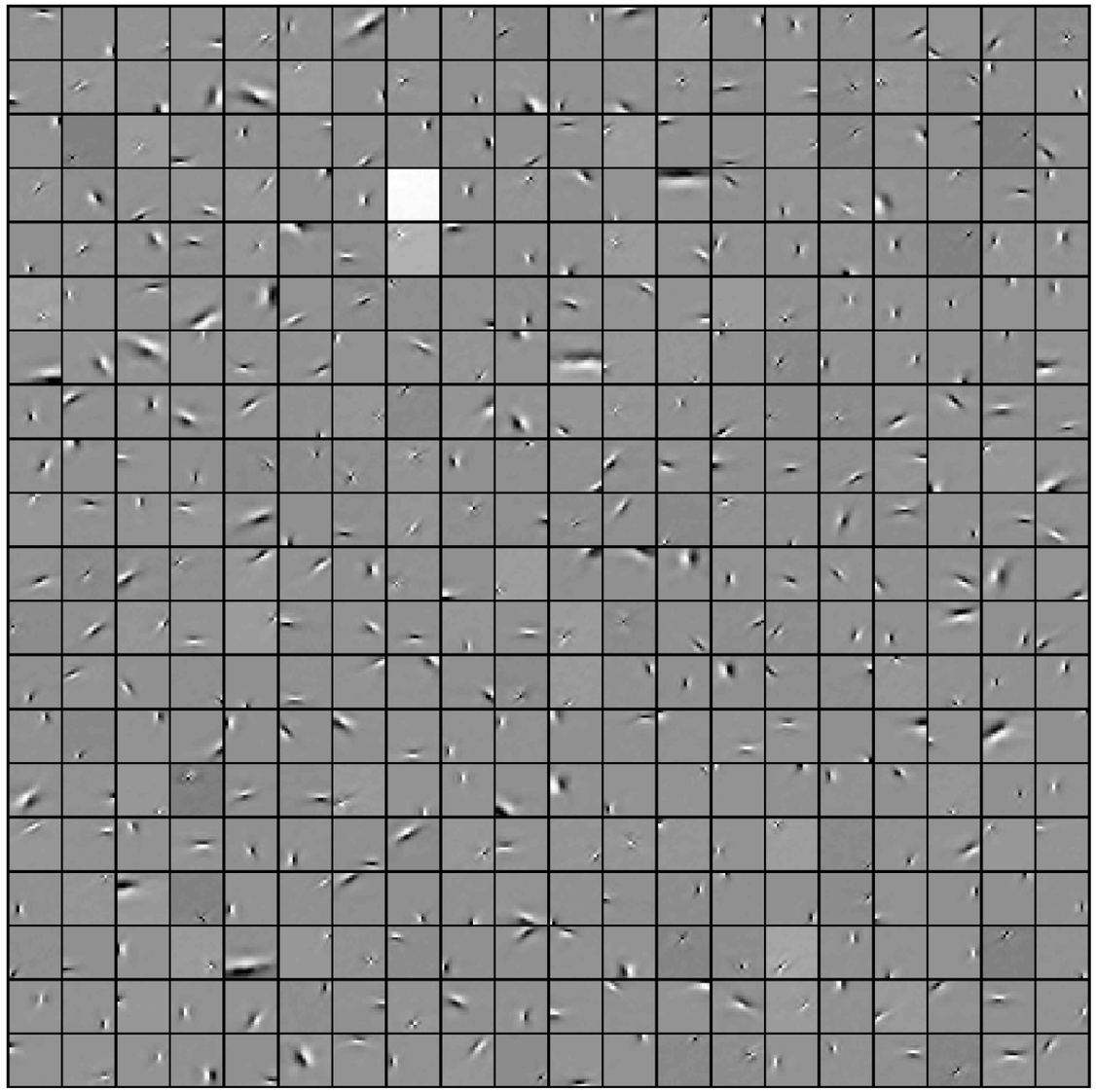
What happens next?



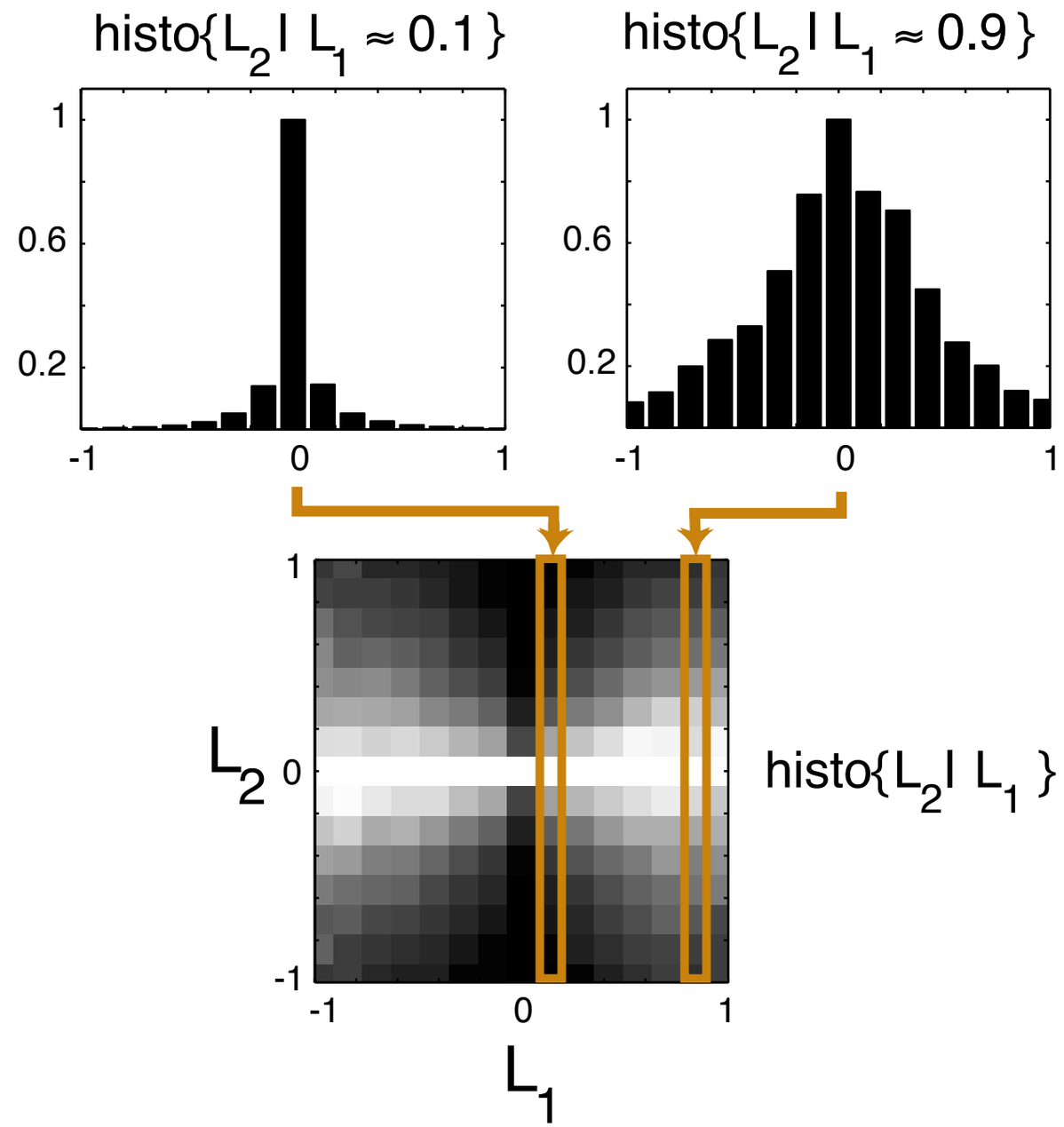
from Hubel, 1995

What happens next?





Joint statistics of filter outputs show magnitude dependence

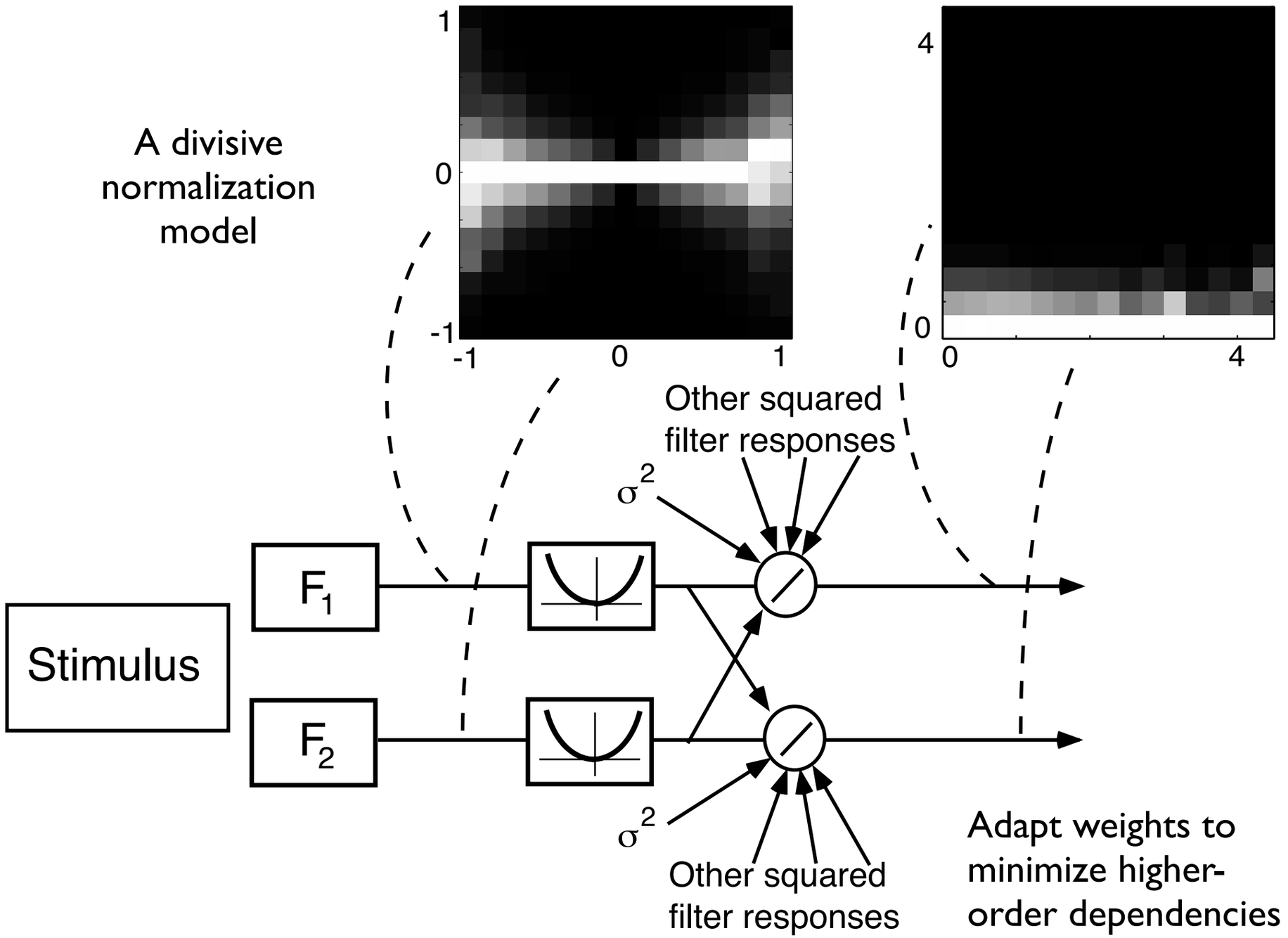


from Schwartz and Simoncelli (2001)

Using sensory gain control to reduce redundancy

Schwartz and Simoncelli (2001)

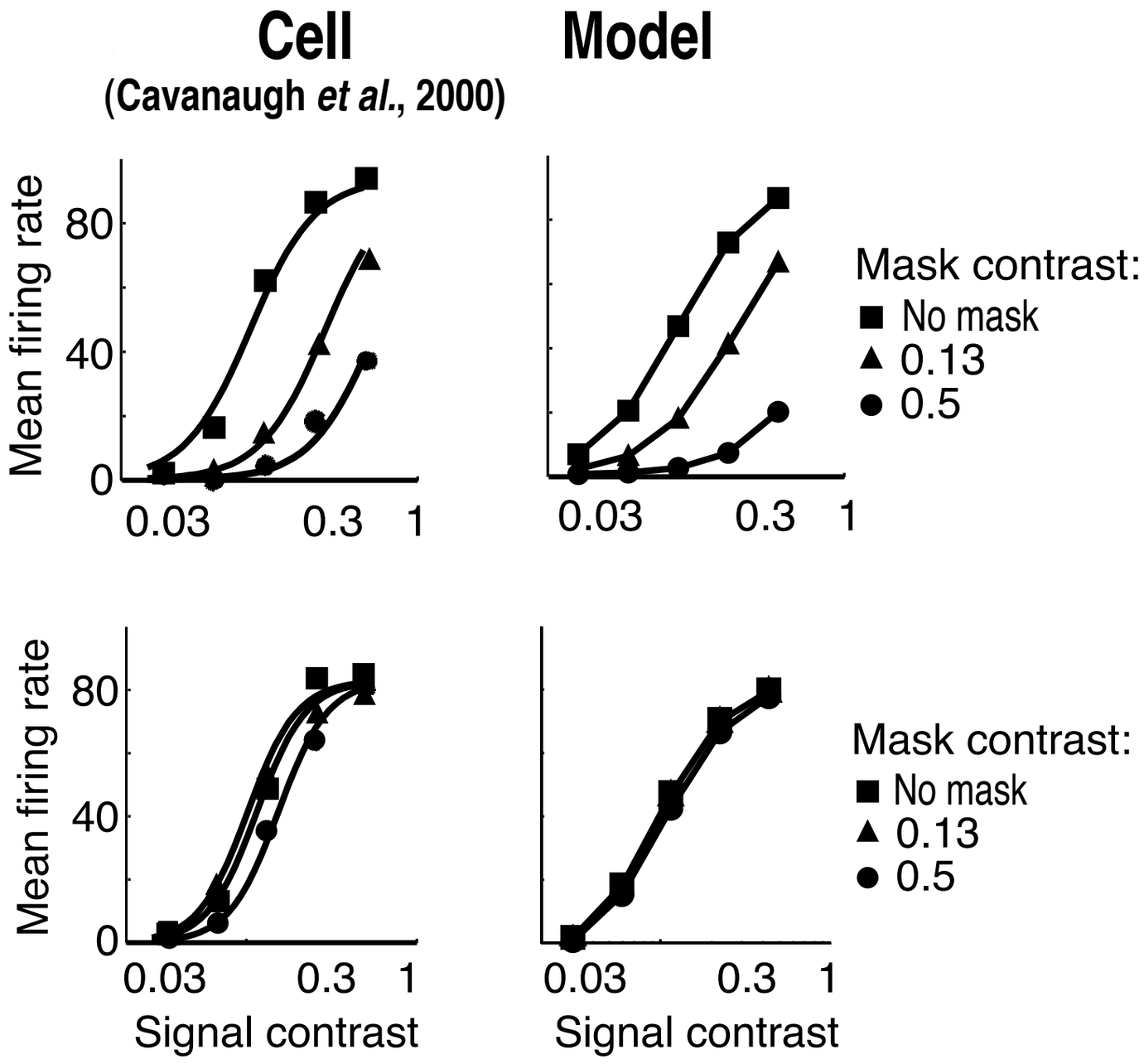
A divisive normalization model

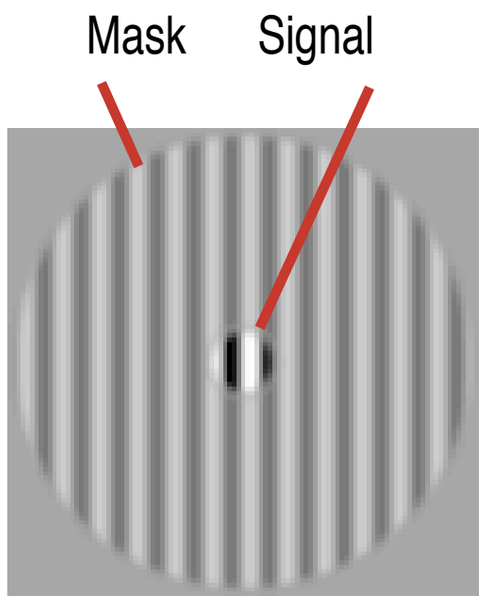


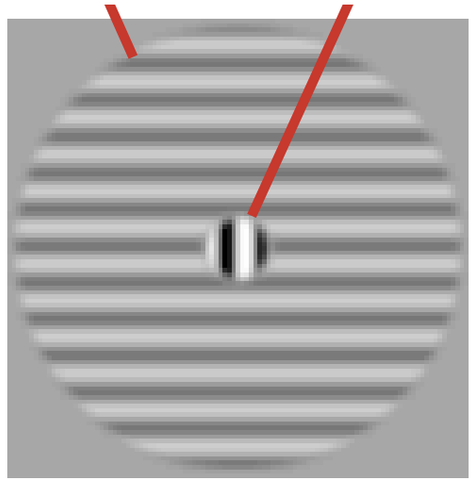
Adapt weights to minimize higher-order dependencies from natural images

Model accounts for several non-linear response properties

Schwartz and Simoncelli (2001)







What about image structure?

- Bottom-up approaches focus on the non-linearity.
- Our aim here is to focus on the *computational* problem:

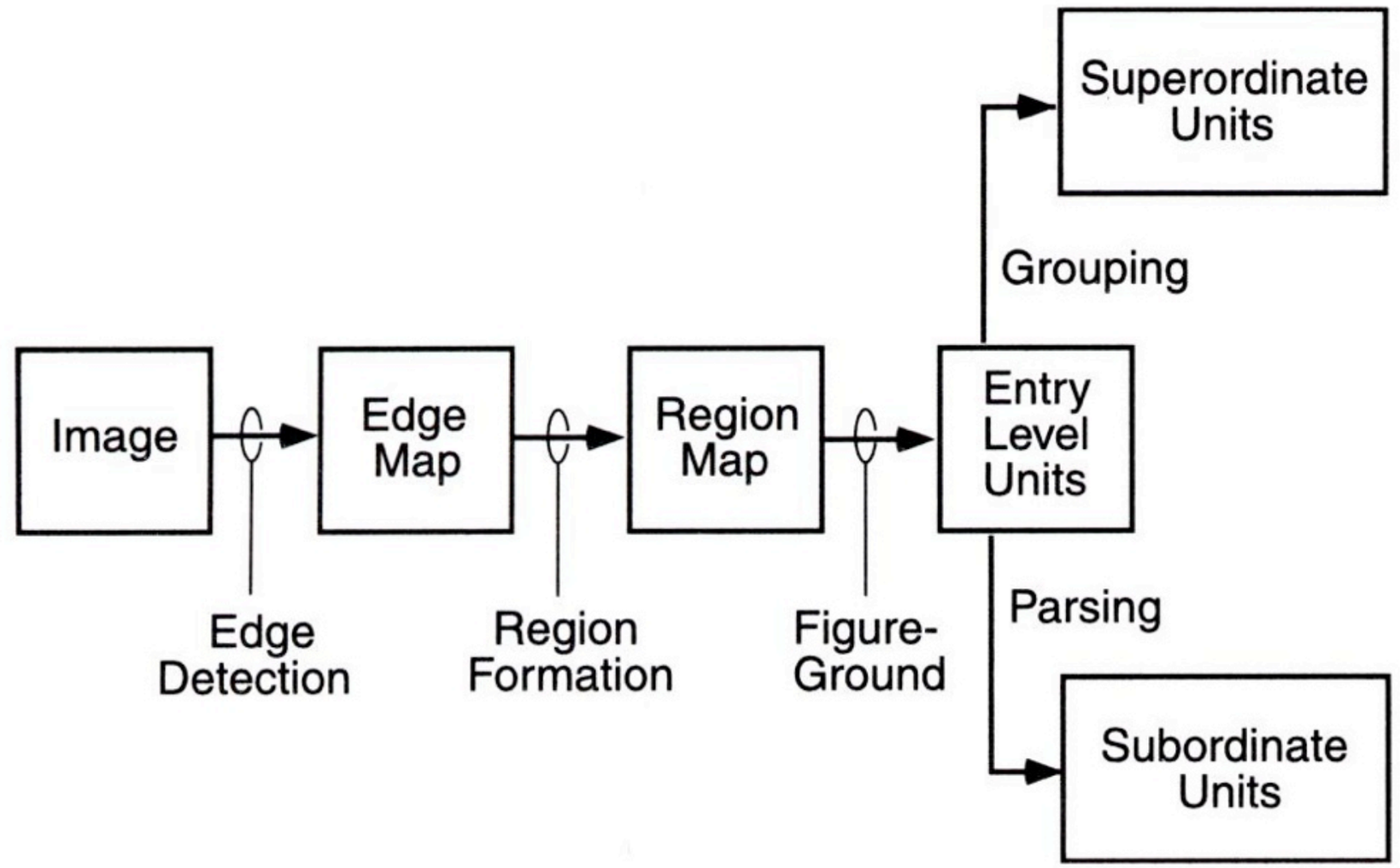
  How do we learn the intrinsic dimensions of natural image structure?

- Idea: characterize how the local image *distribution* changes.

Perceptual organization in natural scenes

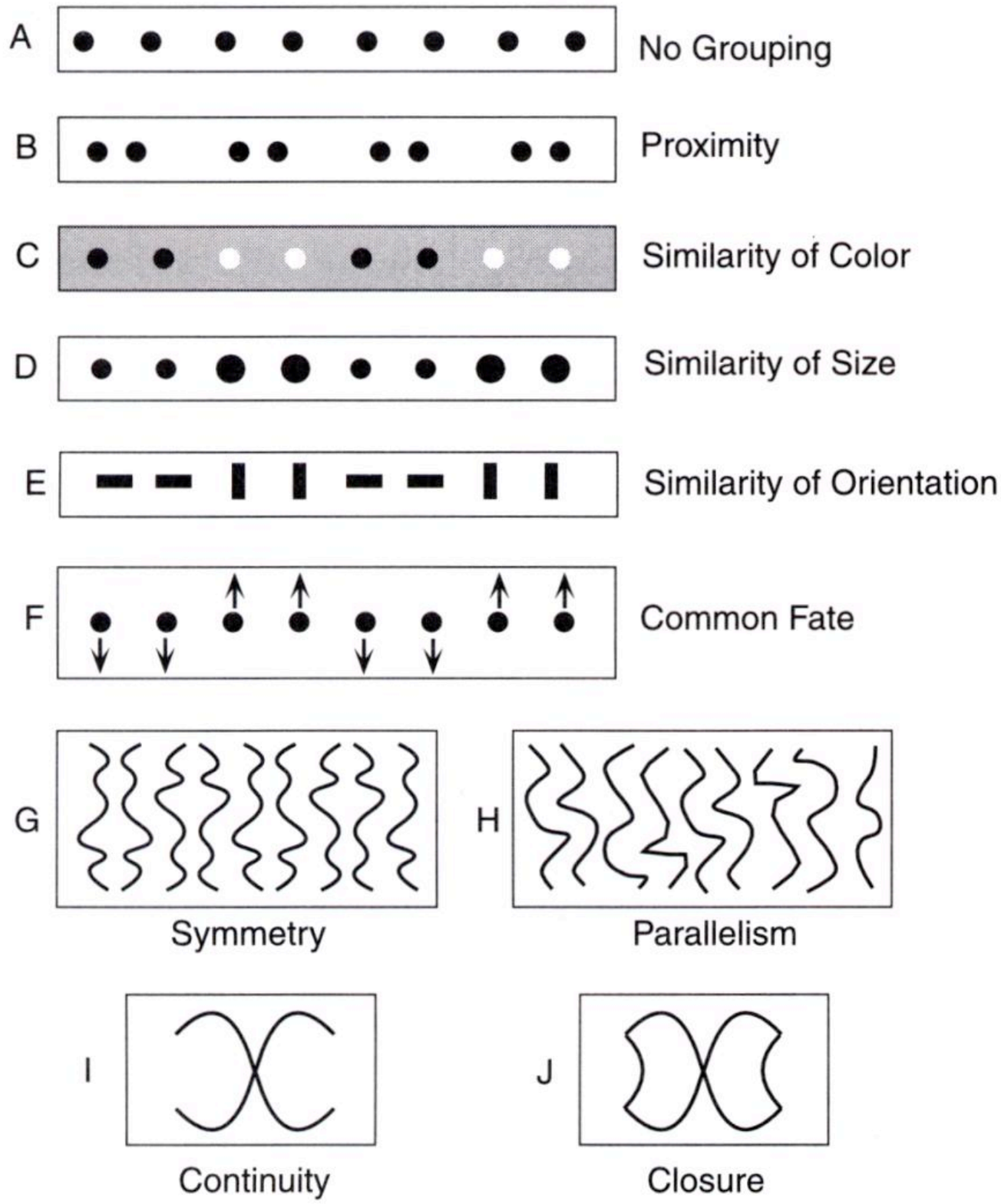
image of Kyoto, Japan from E. Doi

Palmer and Rock's theory of perceptual organization



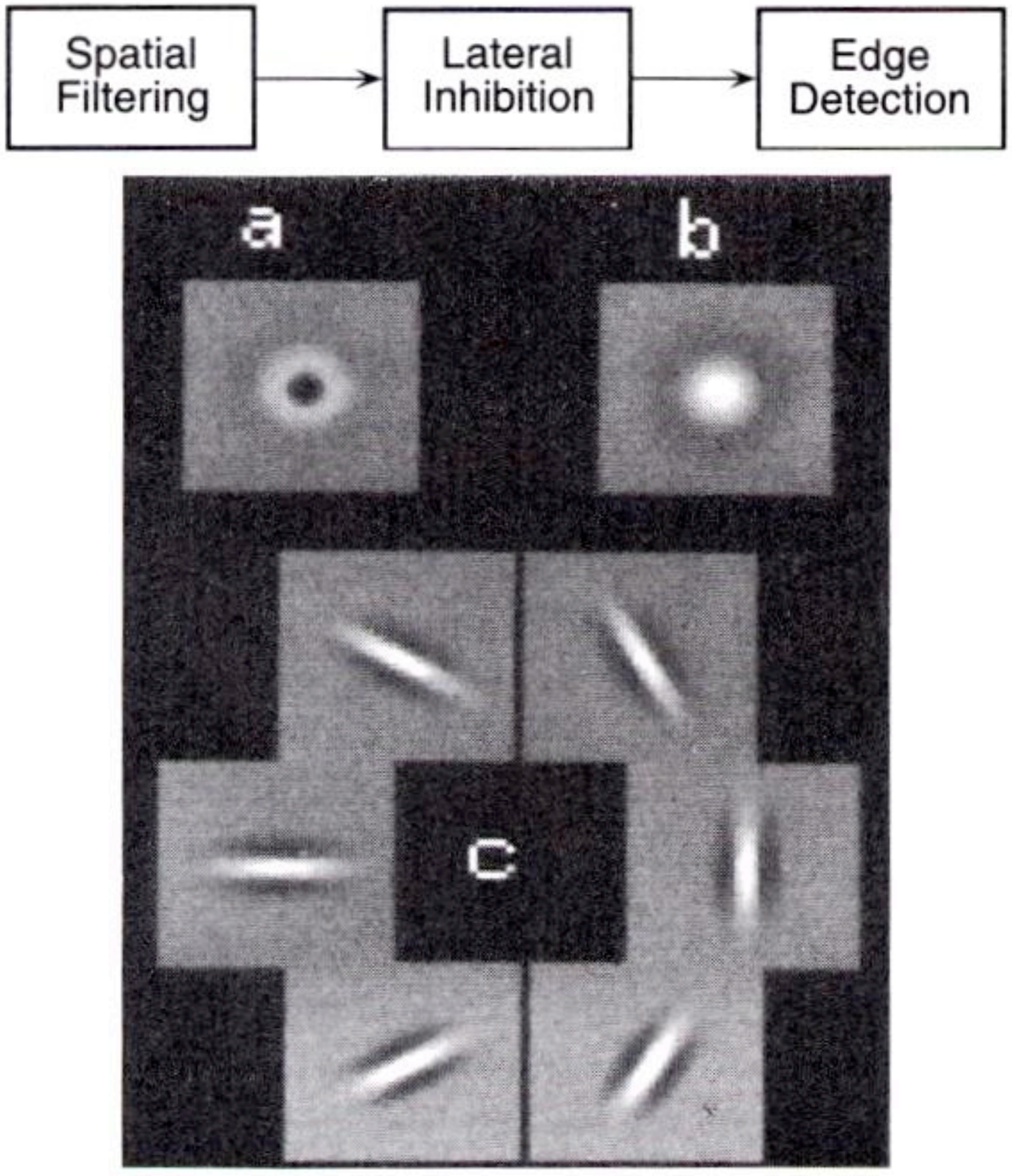
Palmer and Rock, 1994

Gestalt grouping principles



from Palmer, 1999

Malik and Perona's (1991) model of texture segregation



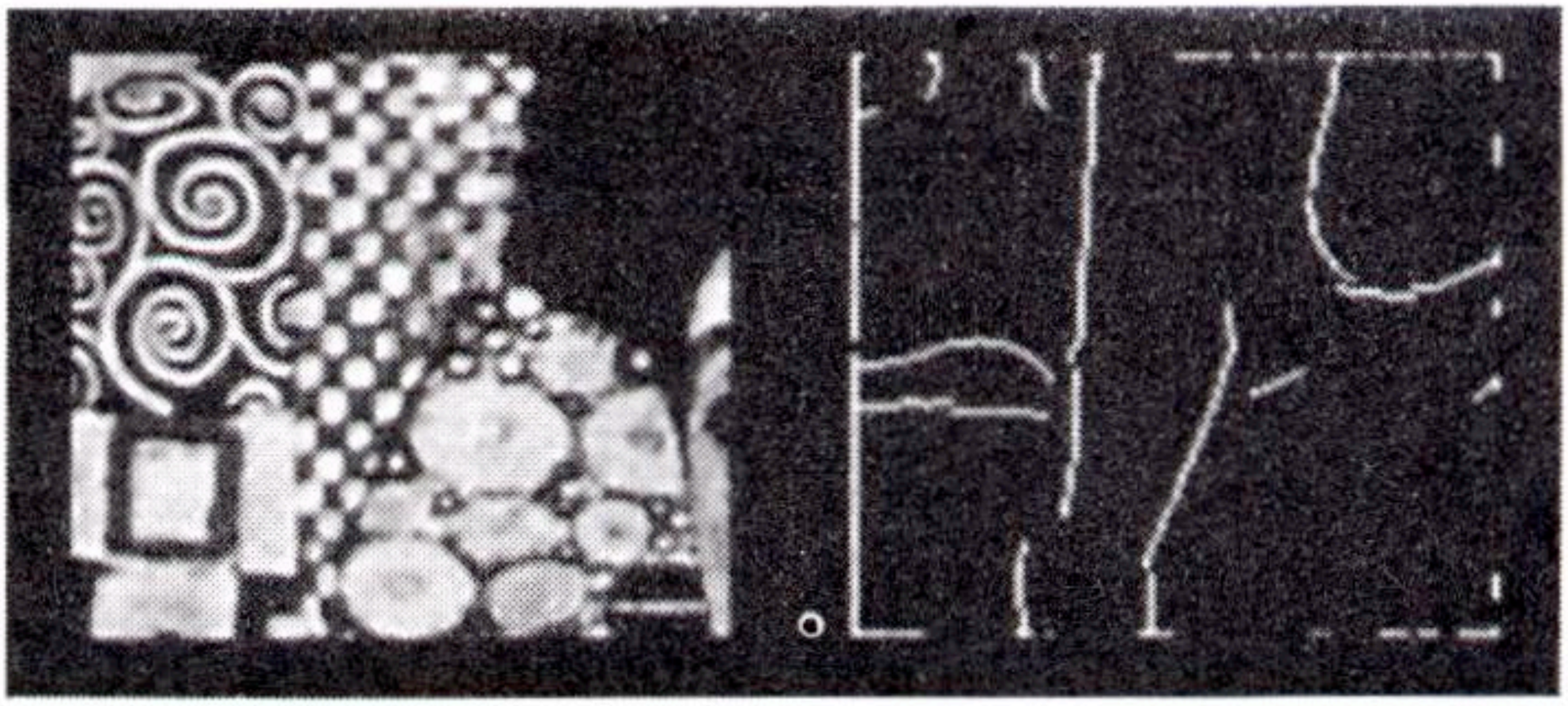
input image

output of dark bar filters

output of light bar filters

Visual Features

Malik and Perona algorithm can find texture boundaries



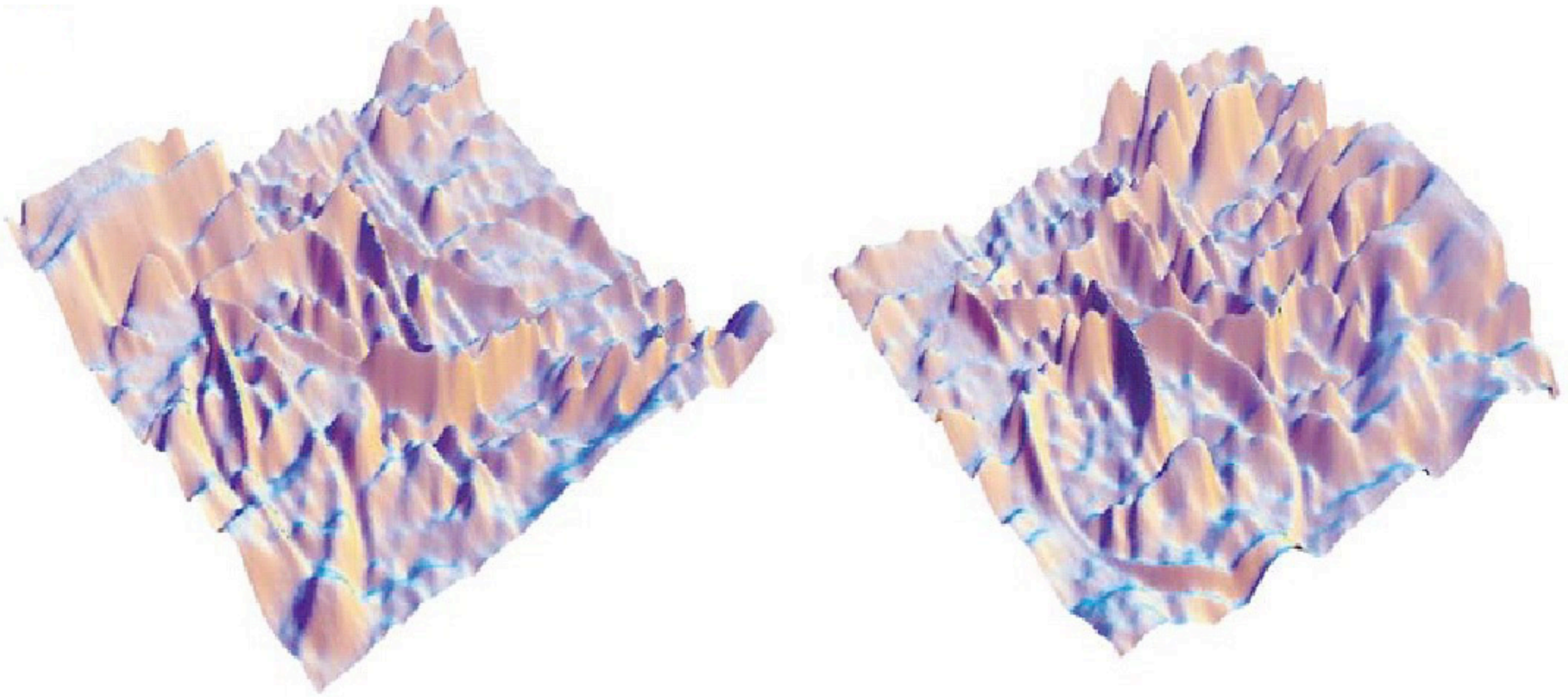
Painting by Gustav Klimt

texture boundaries from Malik and Perona algorithm

Can we apply these to natural scenes?

natural scenes are more complex, and the structures more subtle

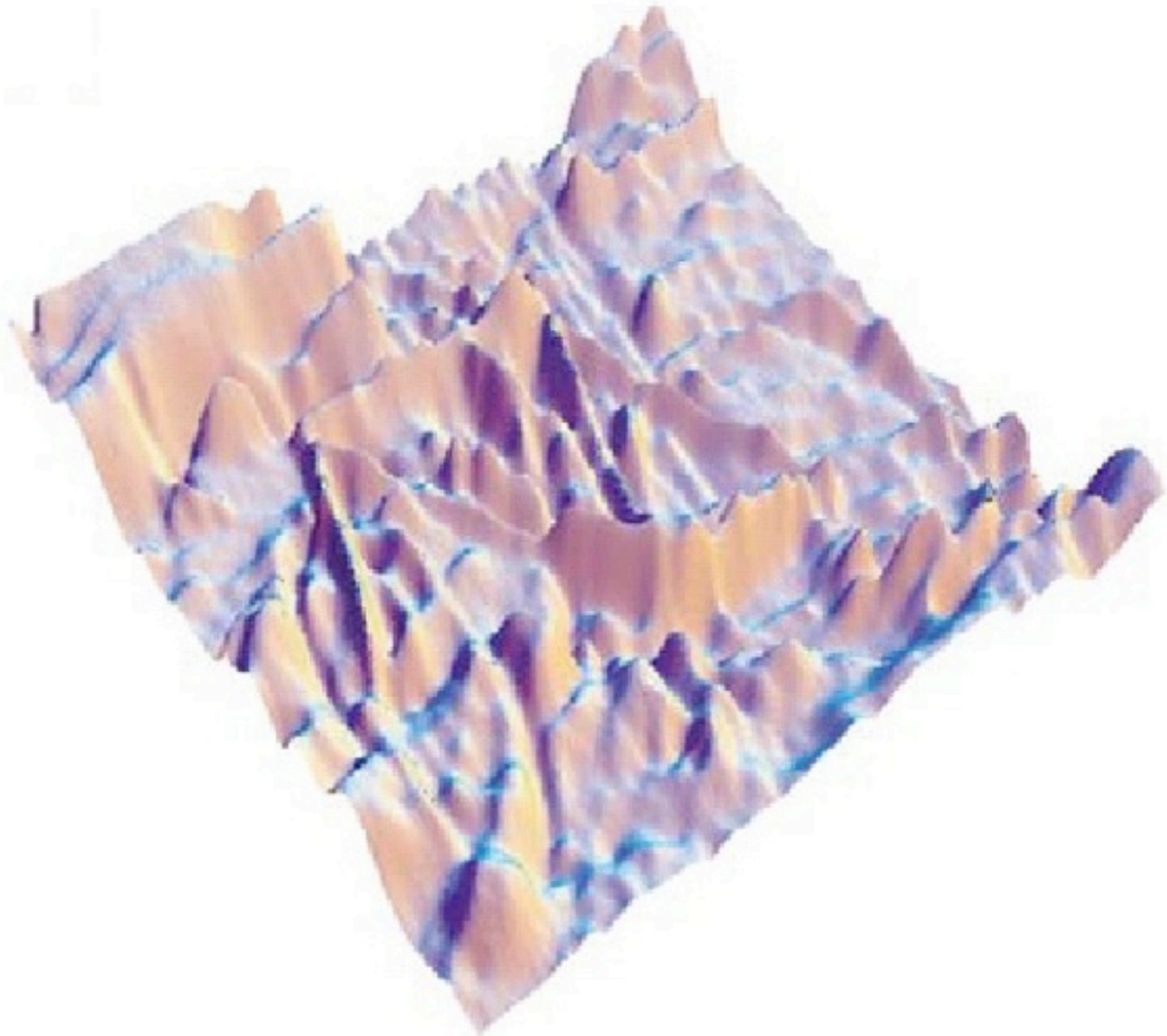
A different representation of a natural scene (Kersten and Yuille, 2003)

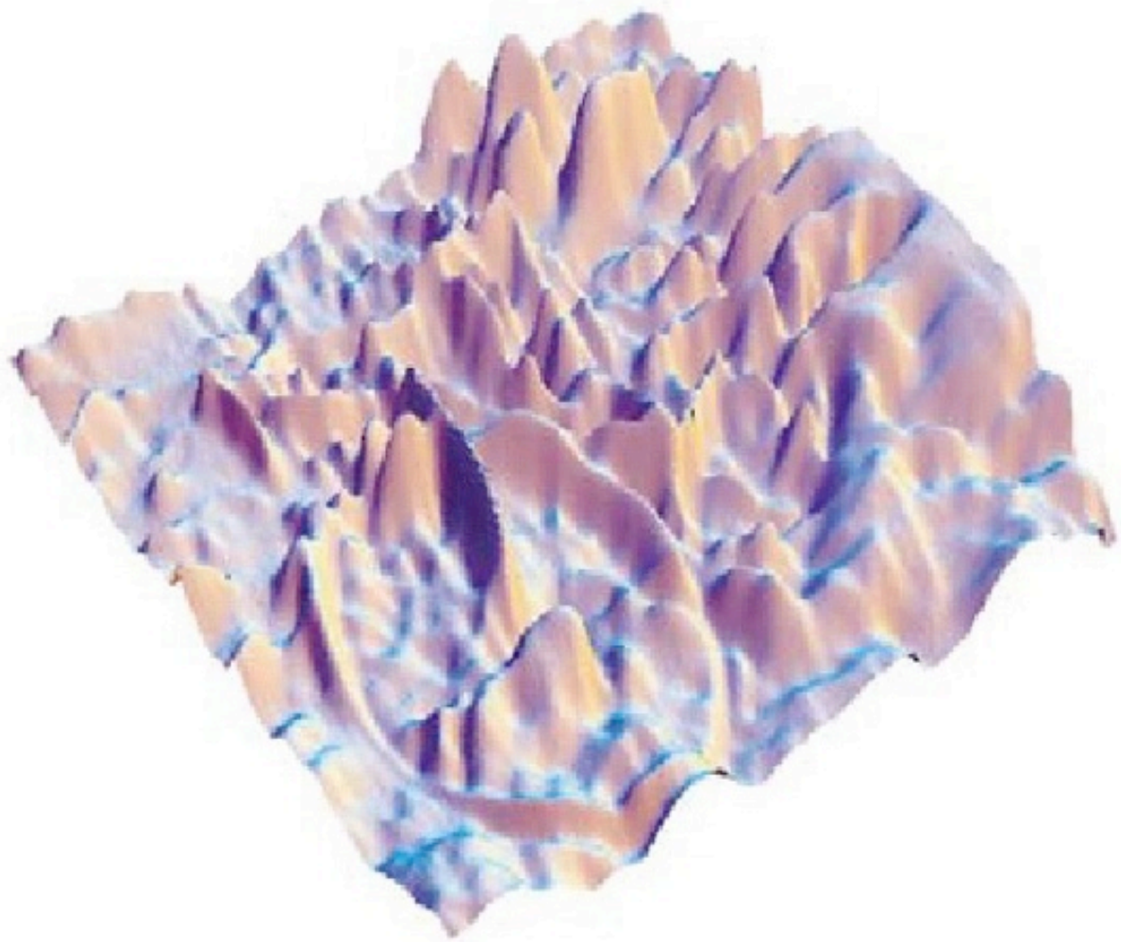


A representation we're more familiar with

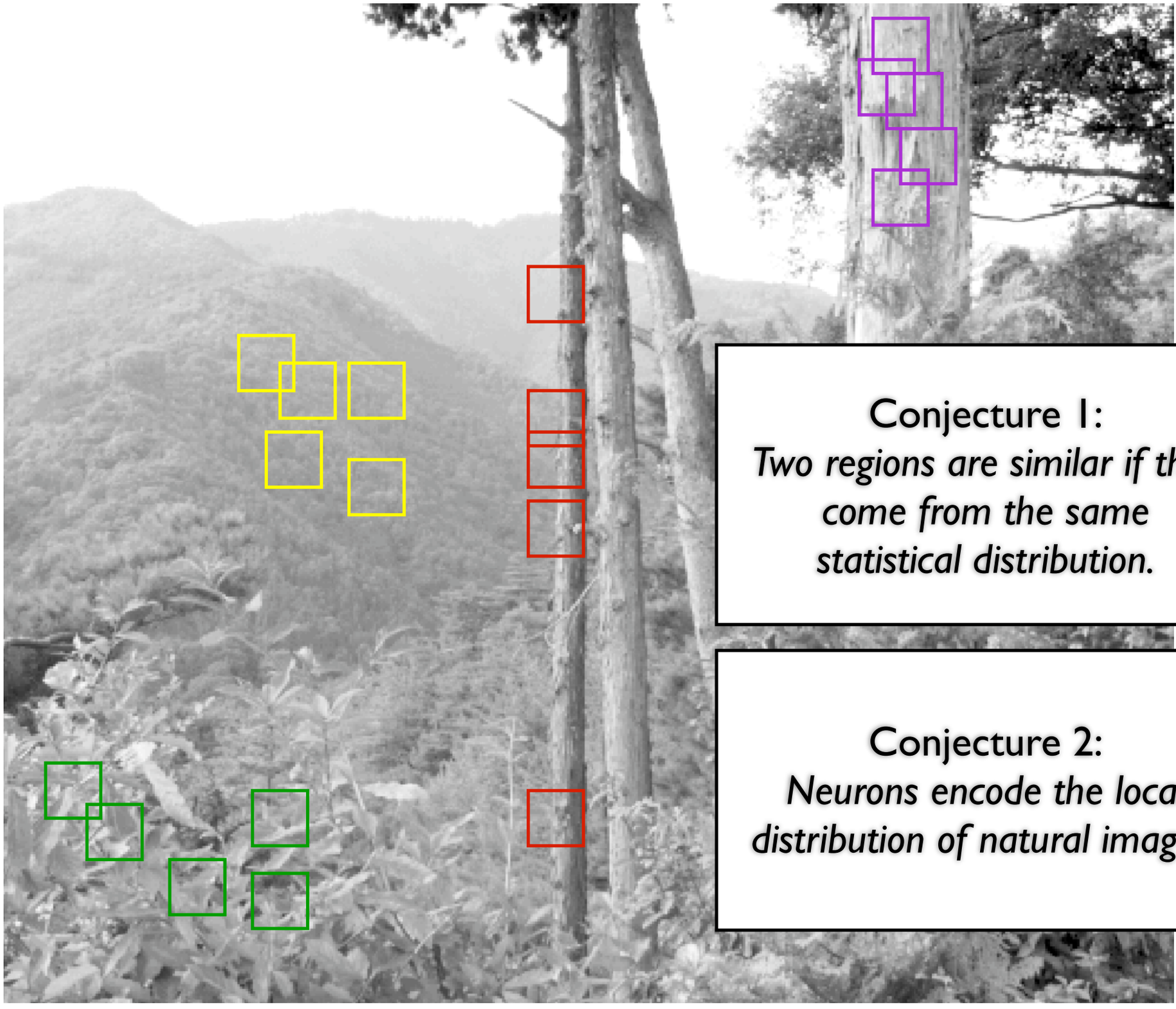
A representation we're more familiar with

This is what our brain does





How do neurons in the visual cortex generalize?



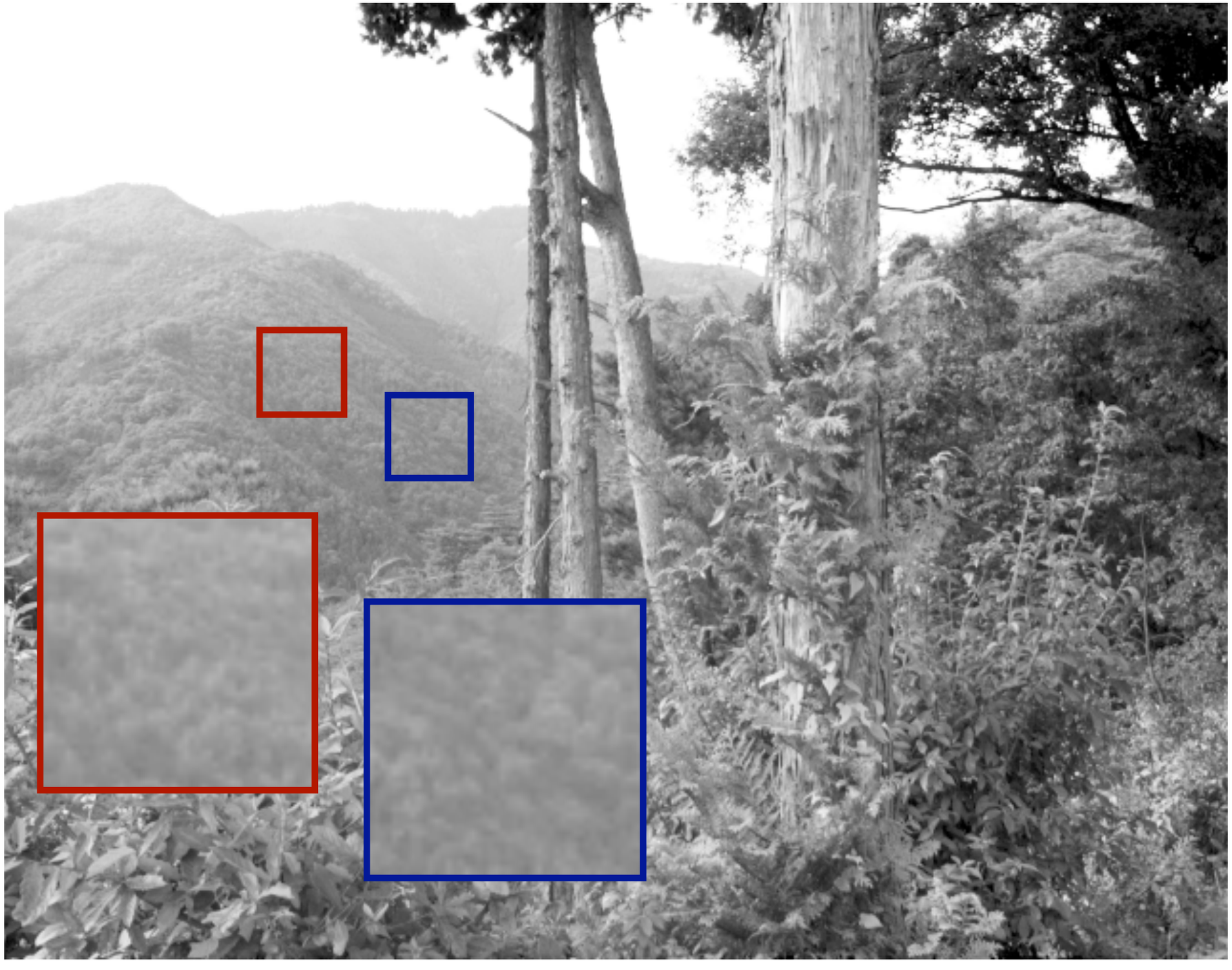
Conjecture 1:
Two regions are similar if they come from the same statistical distribution.

Conjecture 2:
Neurons encode the local distribution of natural images.

(image data from Doi et al, 2003)

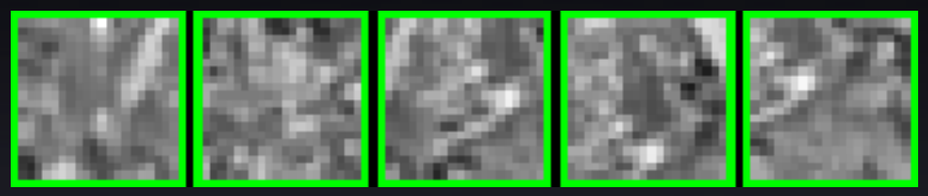
Perceptual generalization in natural scenes

Perceptual generalization in natural scenes



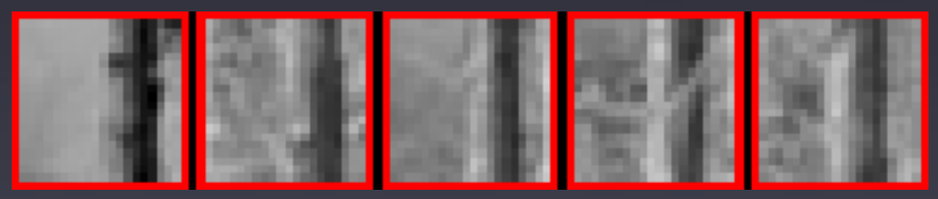
Linear representations do not separate the image classes

- bushes

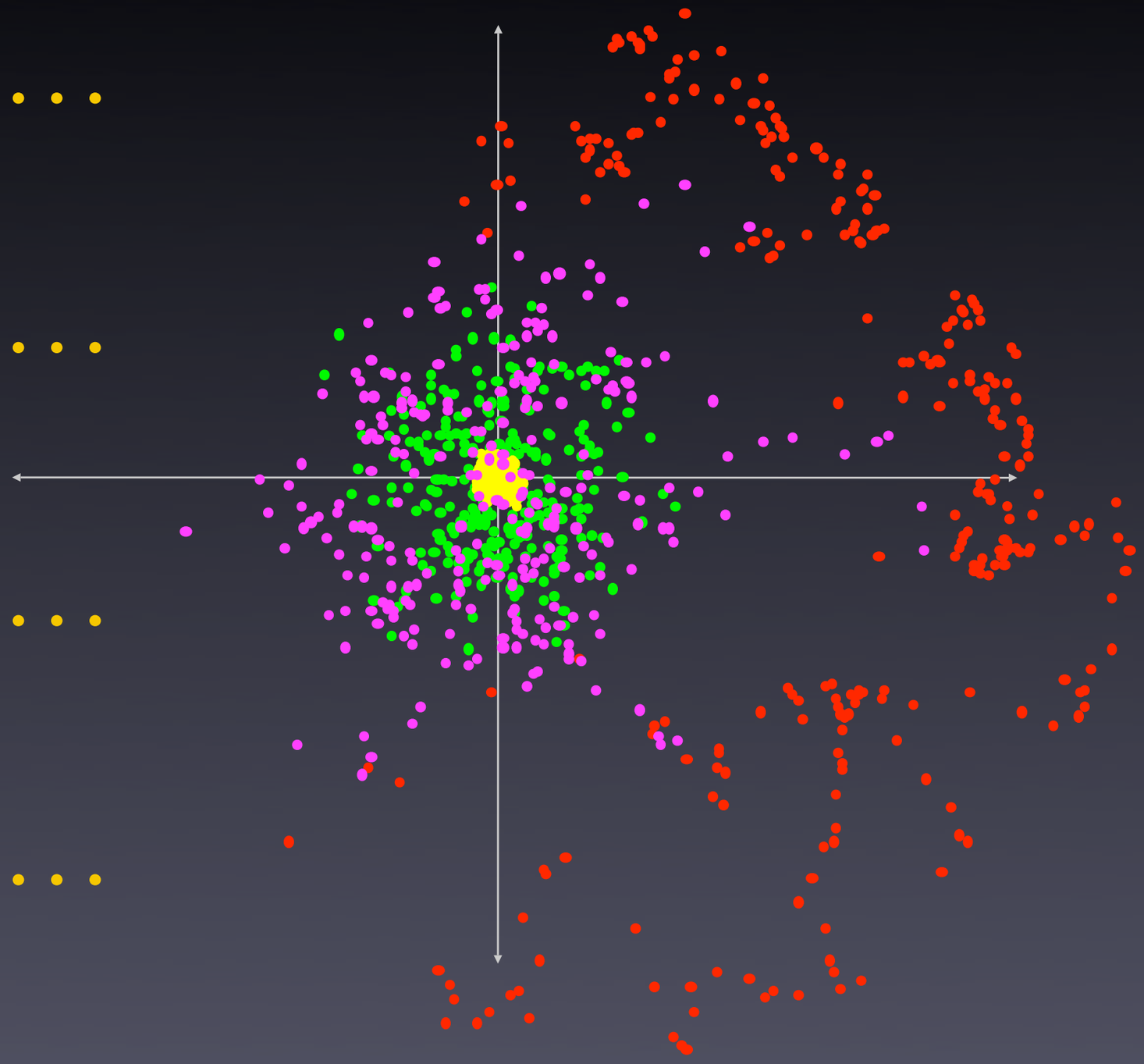


- hillside

- tree edge



- tree bark

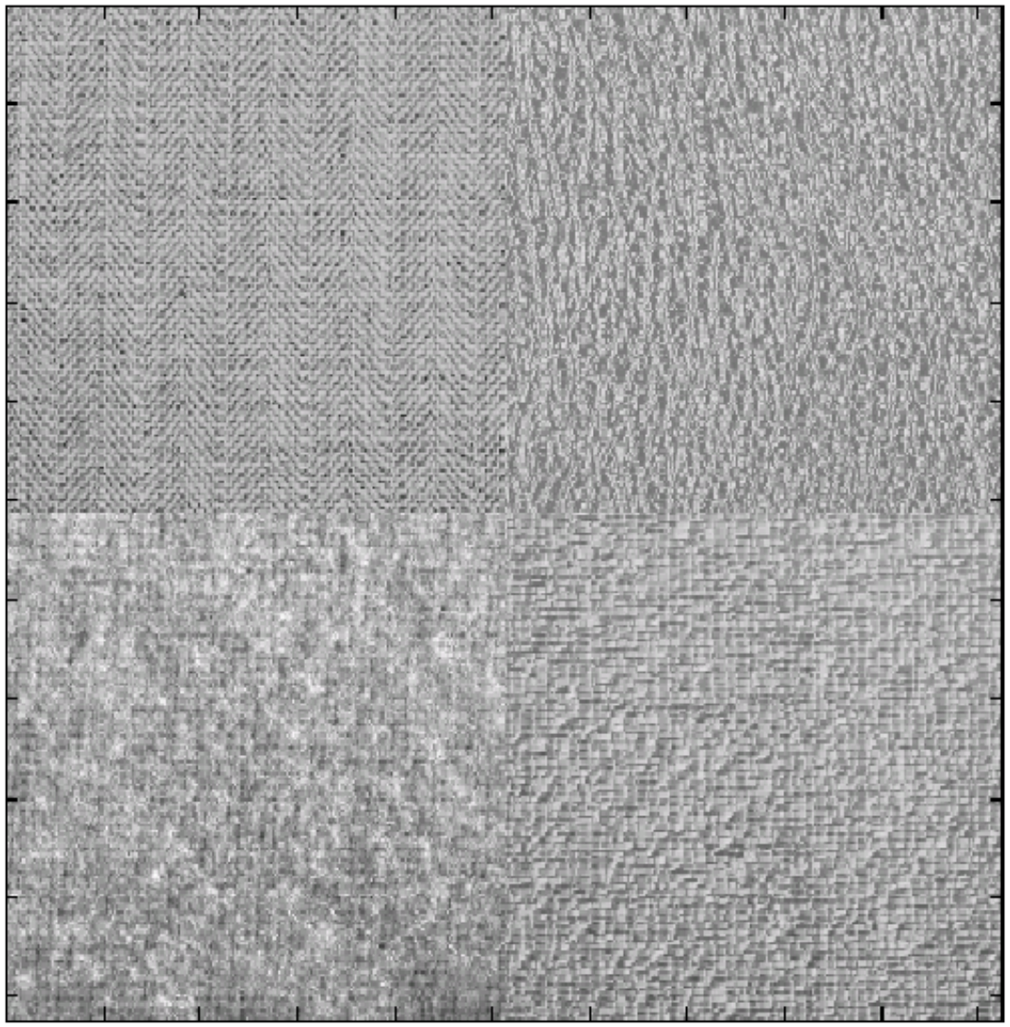


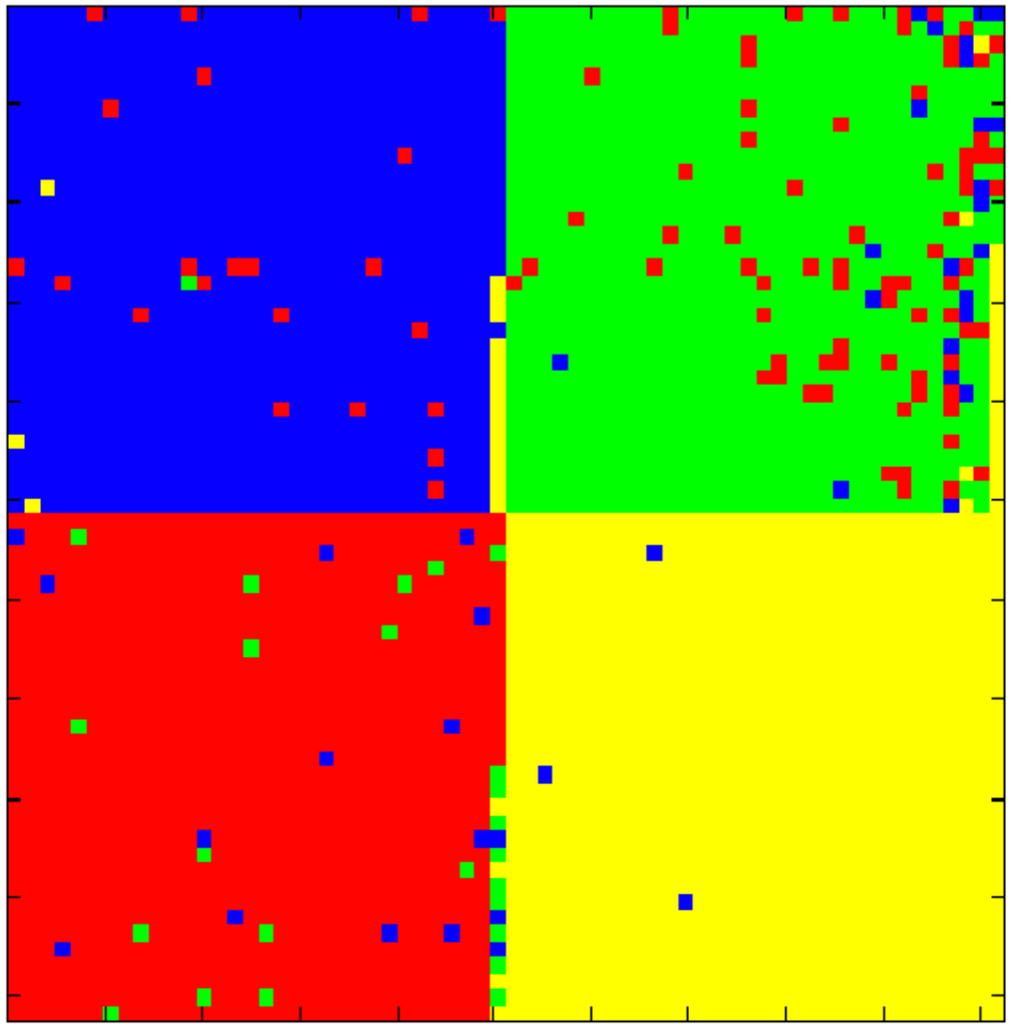
projection onto the first 2 principal components of the data

Modeling local natural scene statistics

- need to model *local* scene structure, not average scene statistics
- need to model *all* structure
  - want a "complete" code
  - a universal "texture" model
- code should be *distributed* and statistically *efficient*

ICA mixtures for similar images



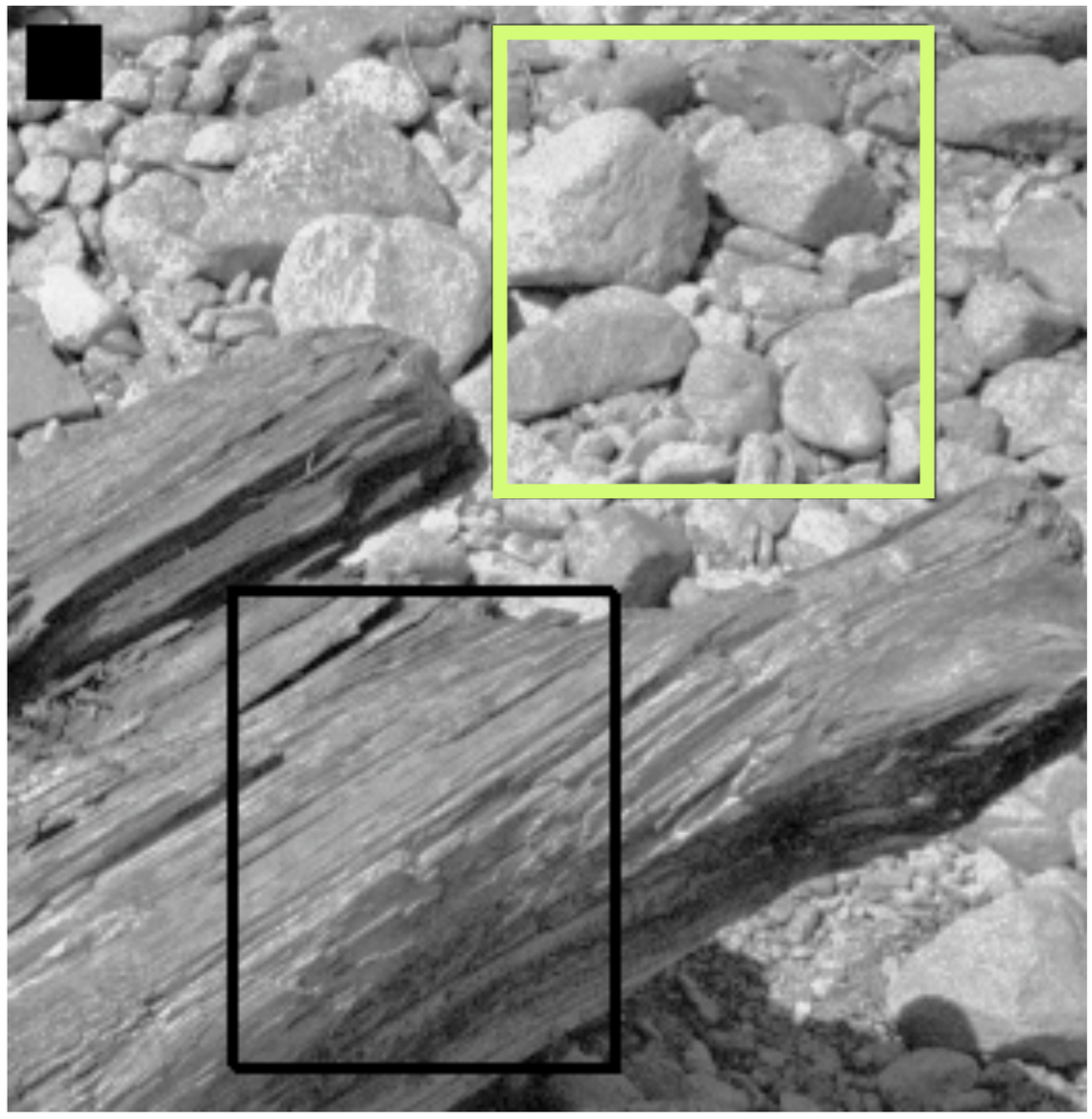


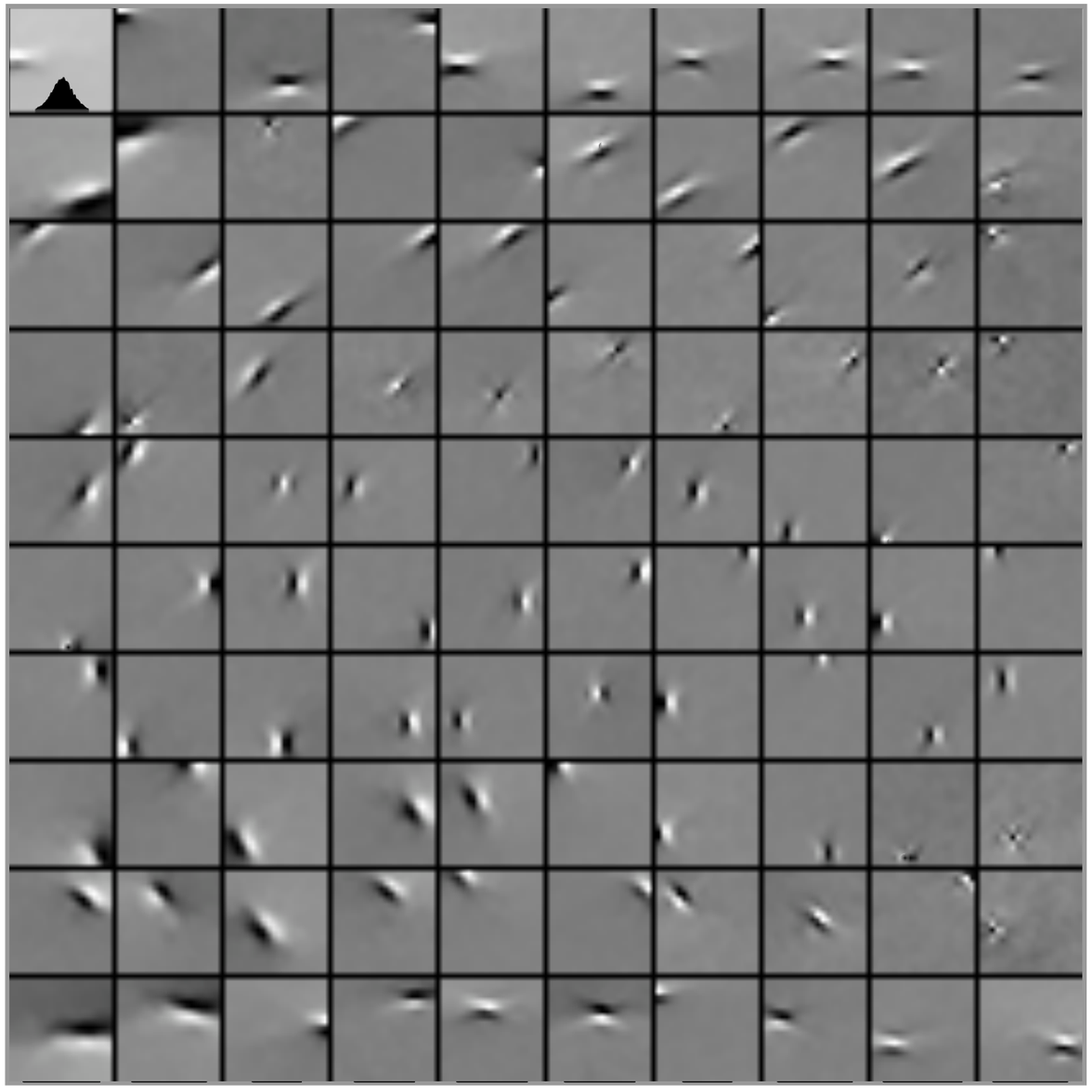
from Lee, Lewicki, and Sejnowski 2000

Limitations:

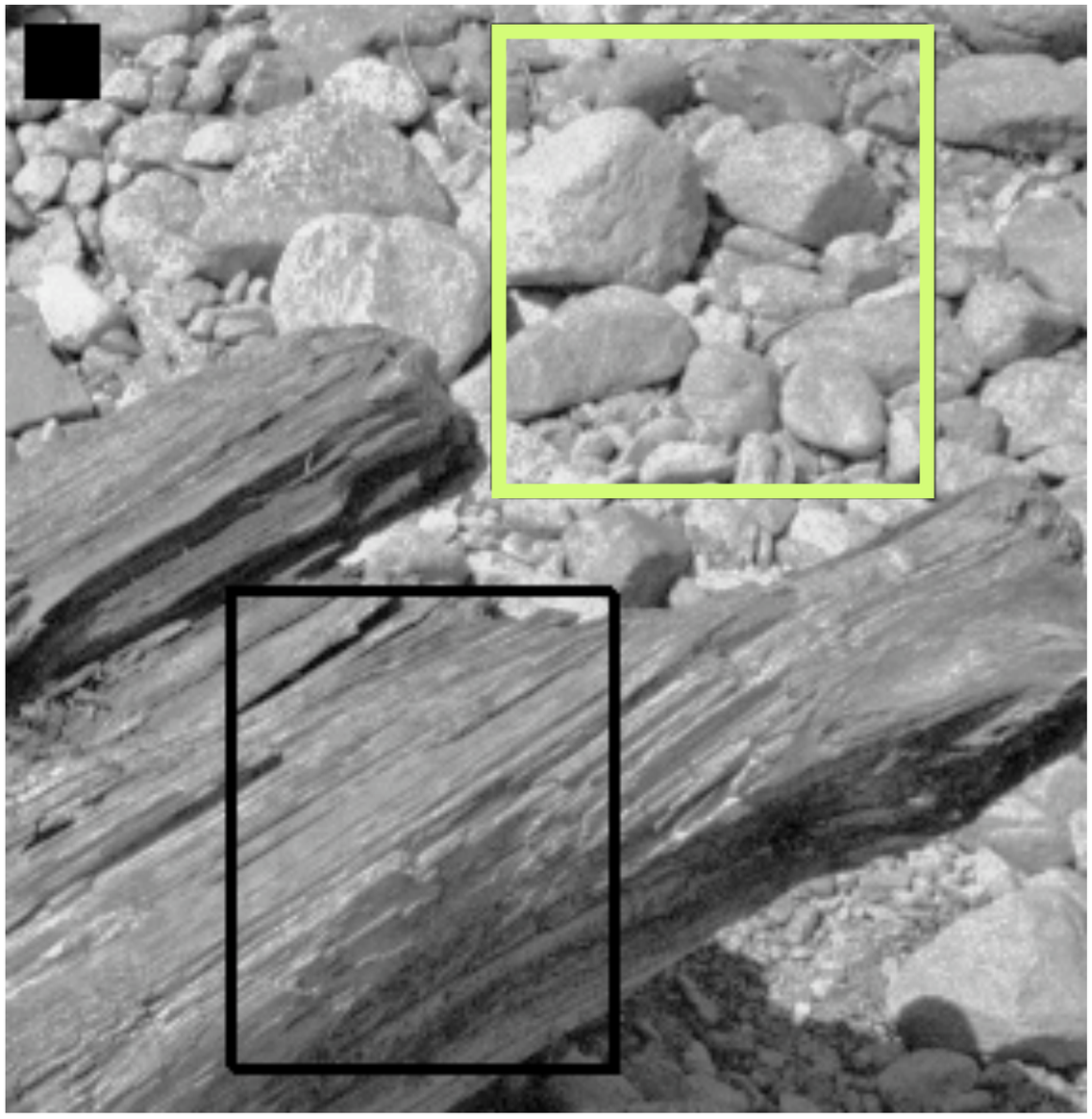
- can only have a small number of classes
- representations are not shared
- cannot learn *intrinsic* dimensions

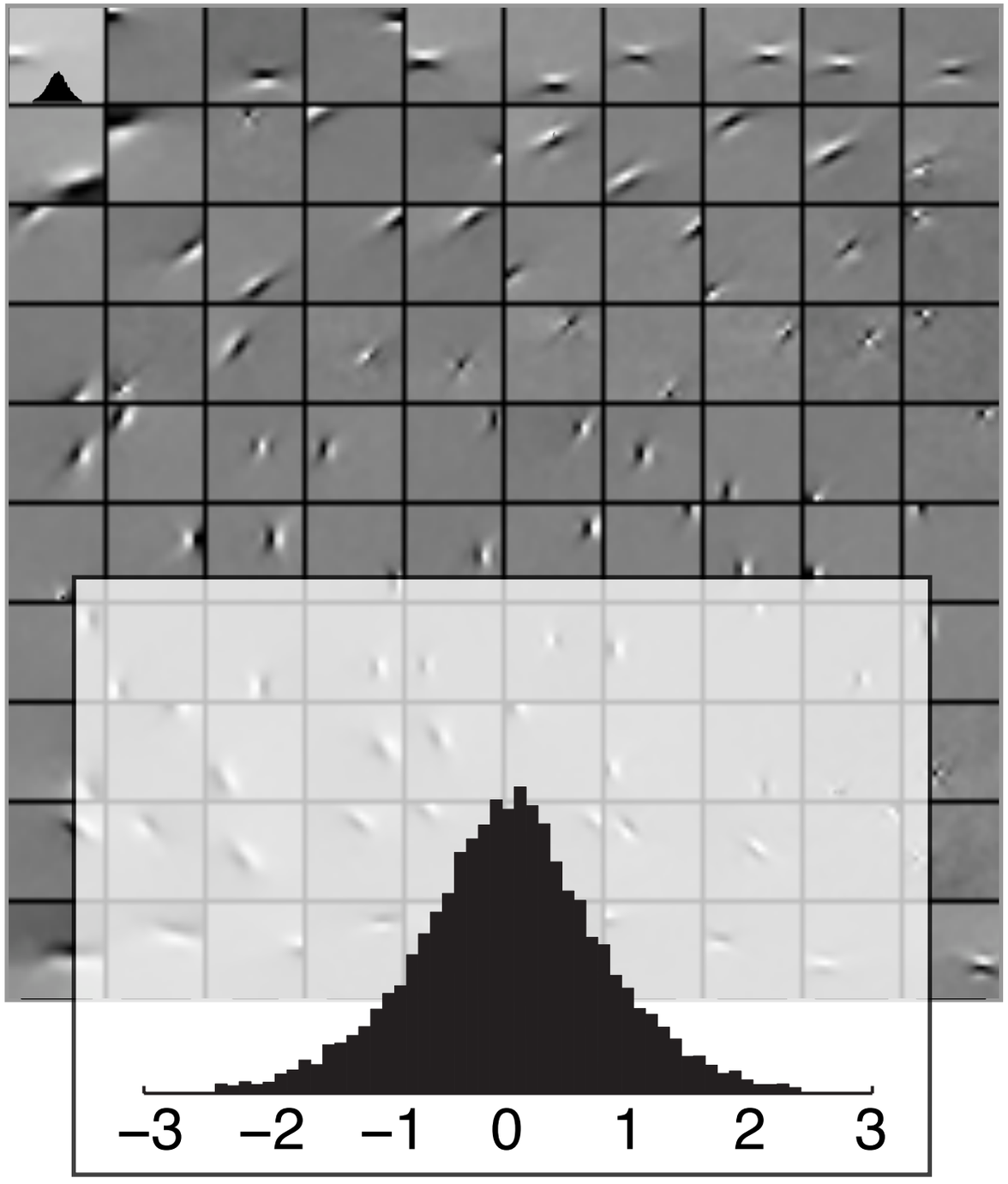
Characterizing different natural image densities



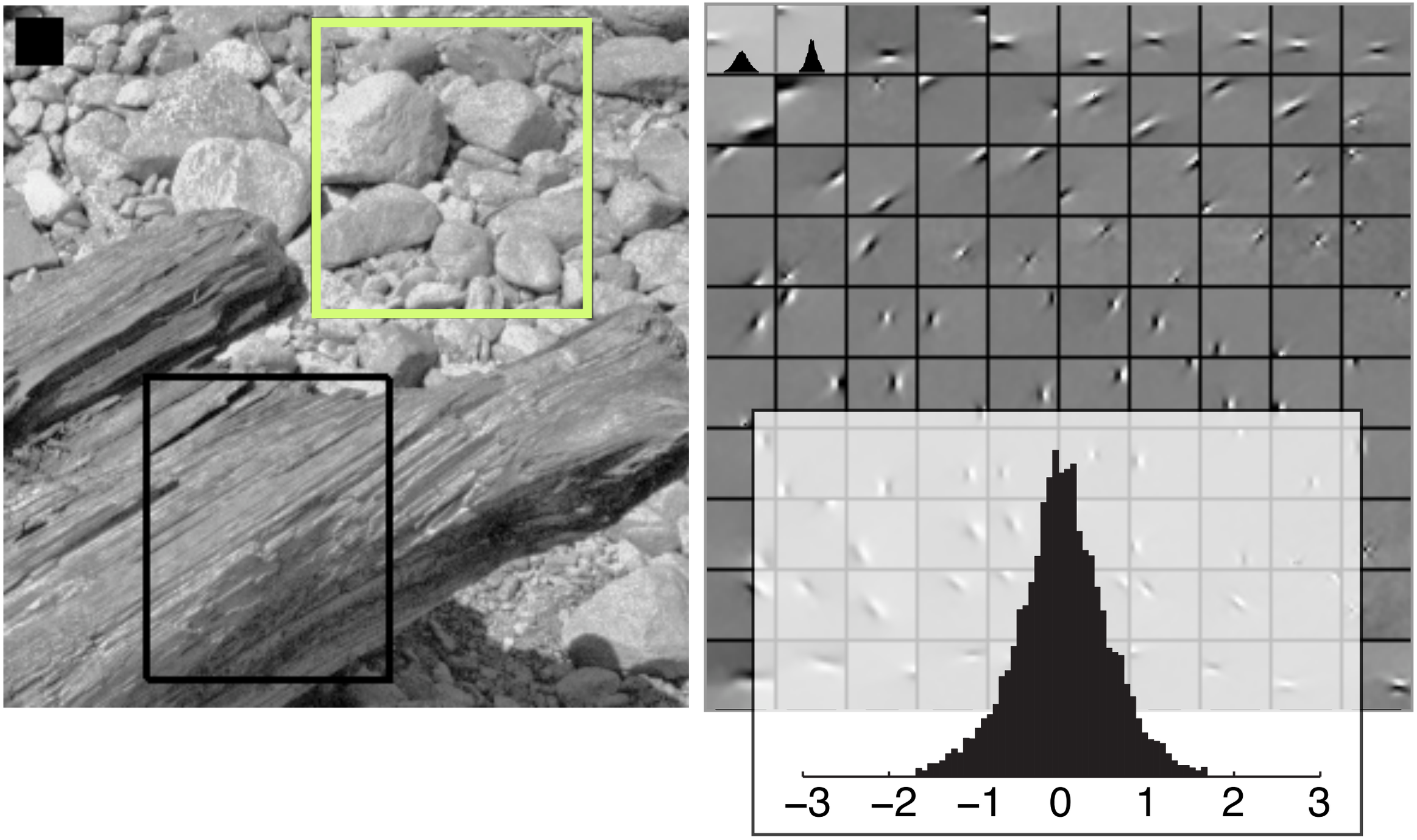


Characterizing different natural image densities

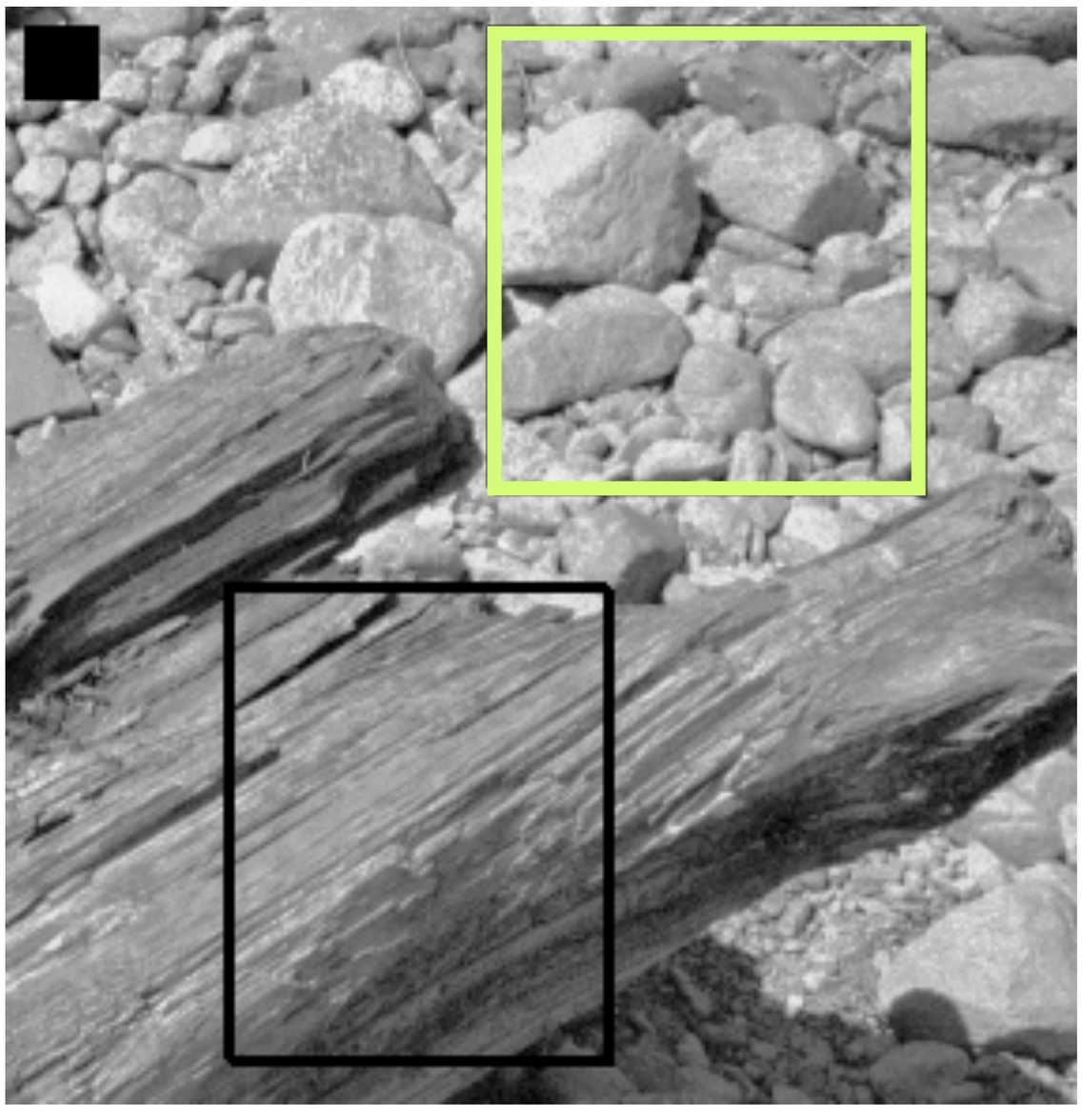


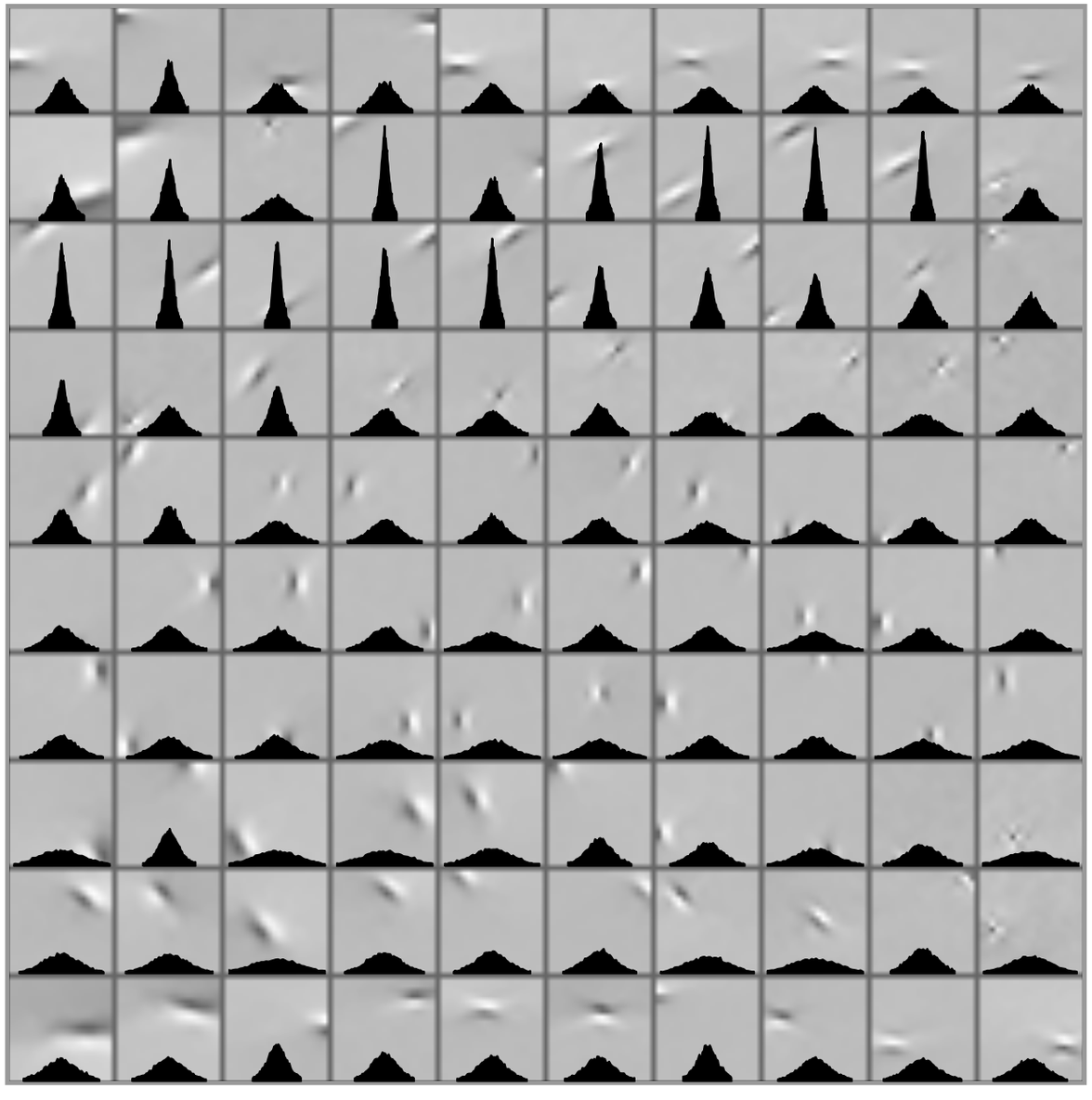


Characterizing different natural image densities

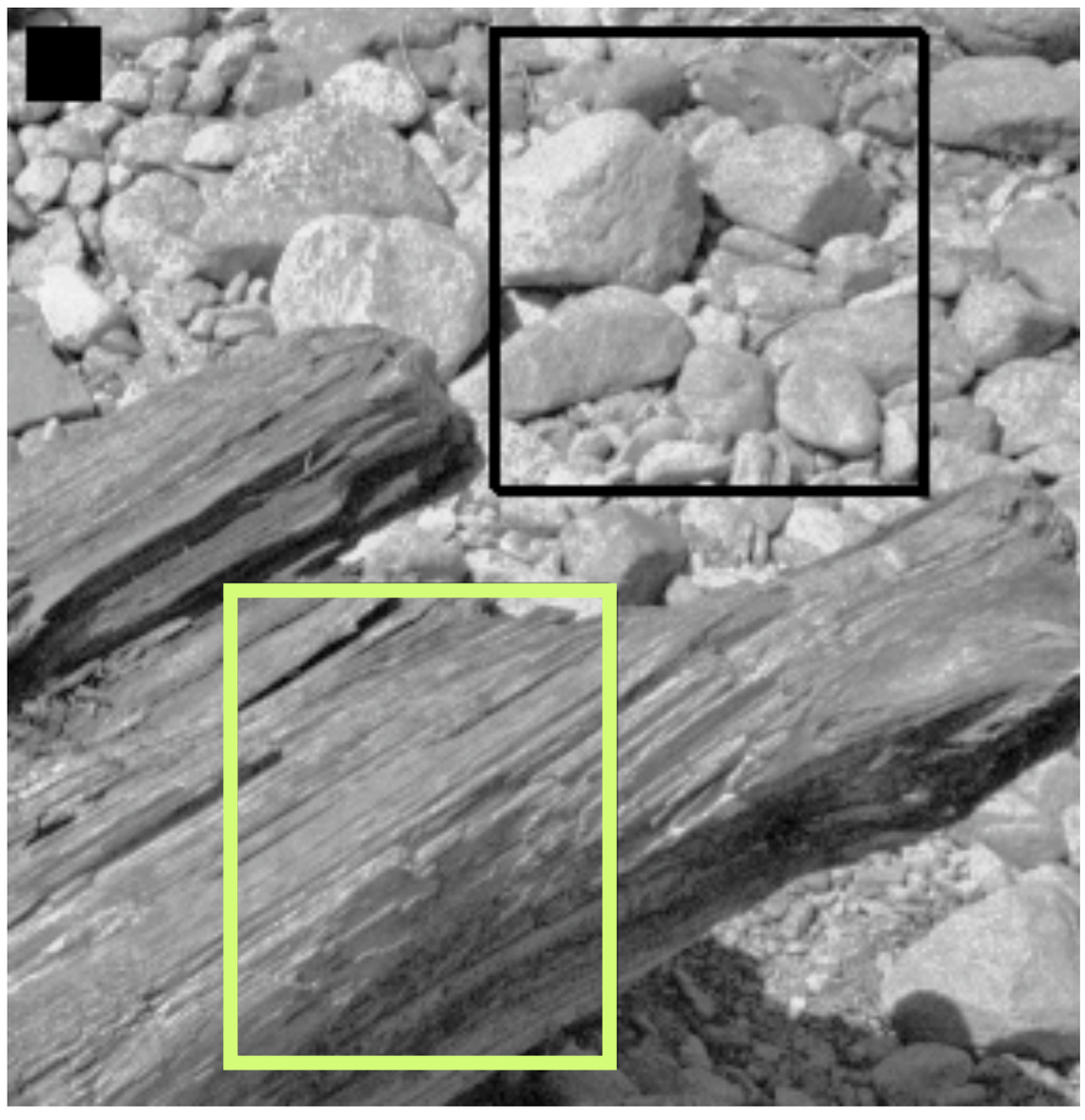


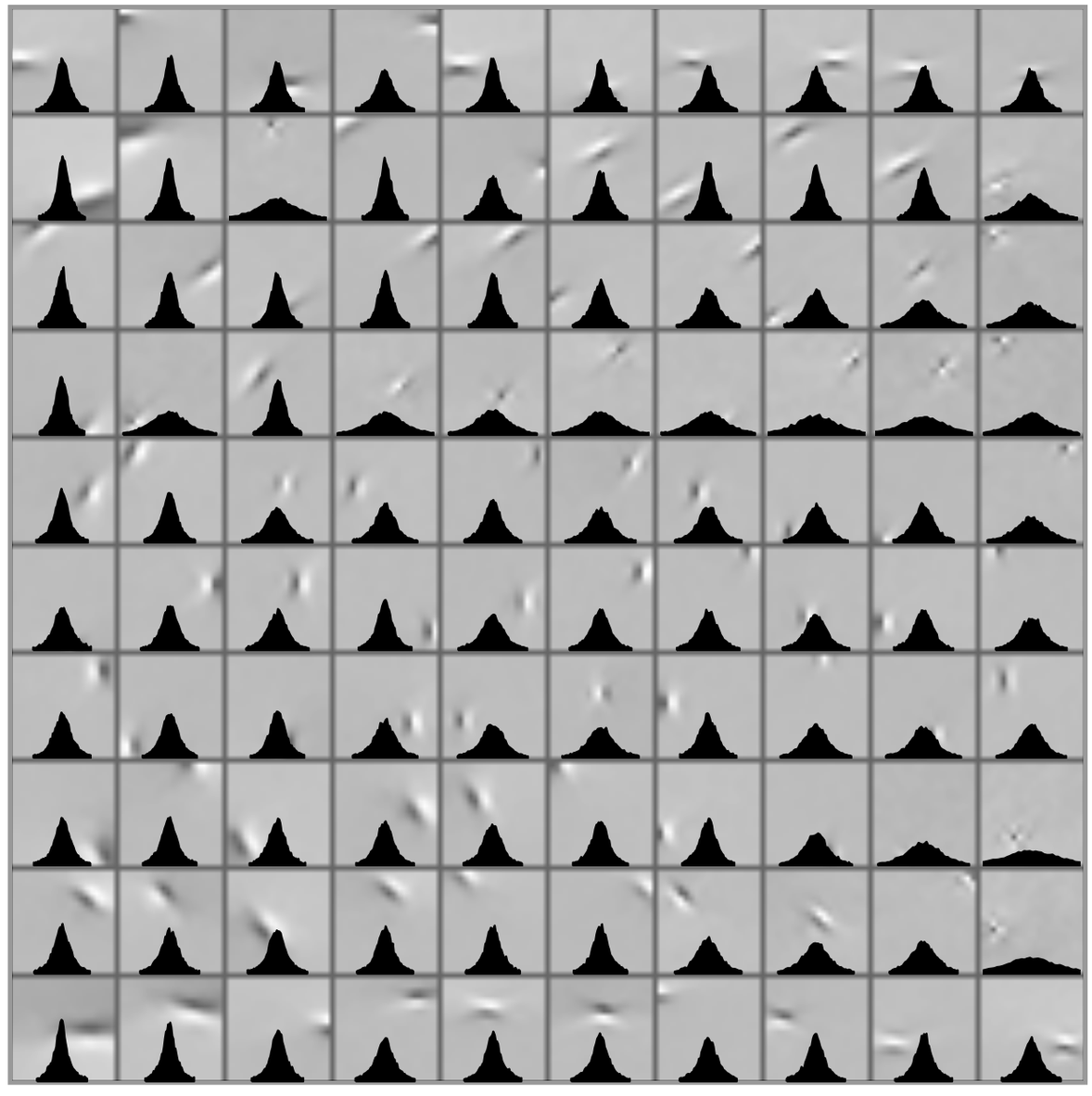
Characterizing different natural image densities





Characterizing different natural image *distributions*



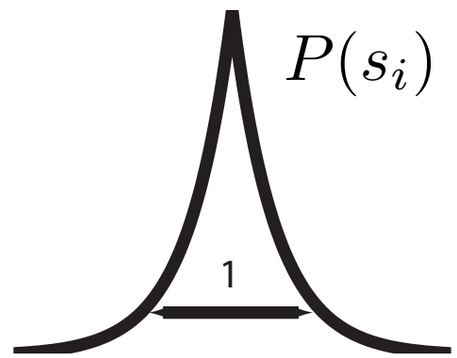


Generalizing the standard ICA model

Karklin and Lewicki (2003; 2005)

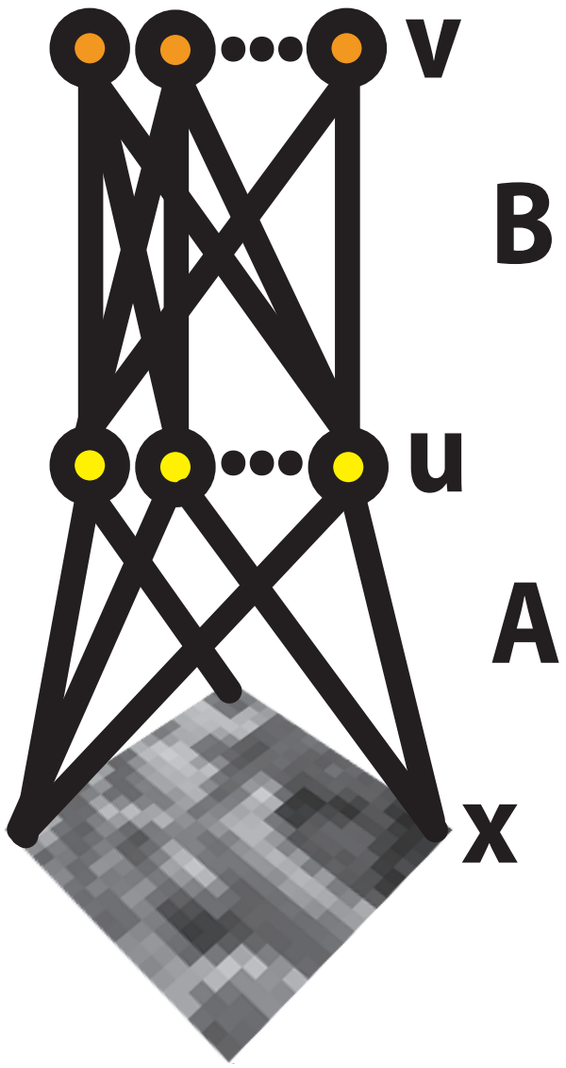
$$P(\mathbf{s}) = \prod_i P(s_i)$$

$$P(s_i) \propto \exp\left[-\left|\frac{s_i}{\lambda_i}\right|^{q_i}\right]$$



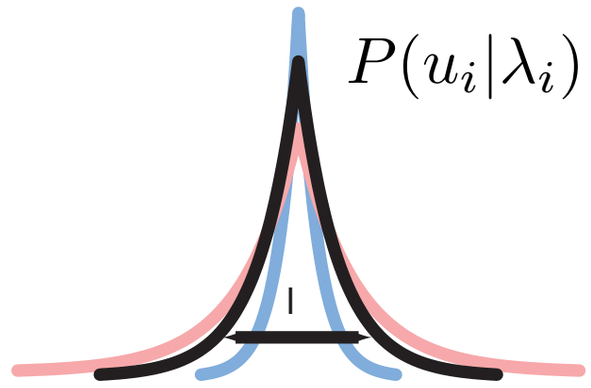
Generalizing the standard ICA model

Karklin and Lewicki (2003; 2005)

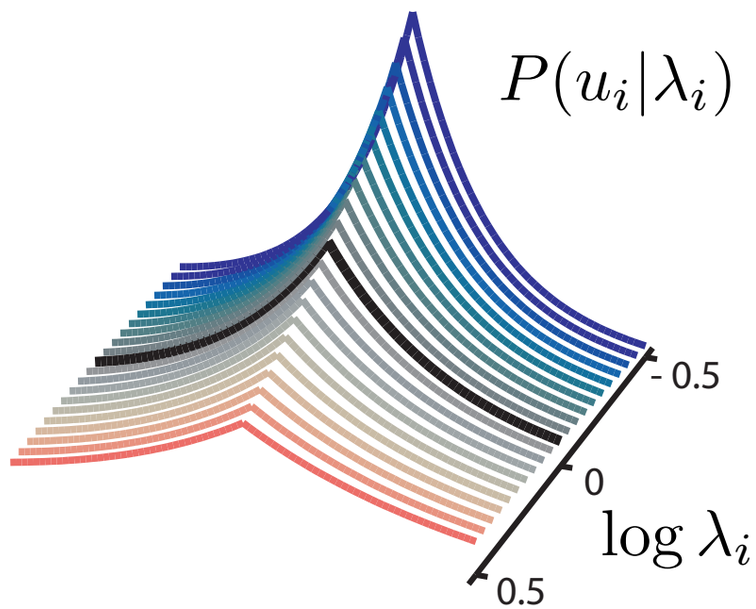


$$-\log p(\boldsymbol{u}|\mathbf{B},\boldsymbol{v}) \propto \sum_i \left| \frac{u_i}{\exp([\mathbf{B}\boldsymbol{v}]_i)} \right|^{q_i}$$

$$P(u_i|\lambda_i) \propto \exp\left[-\left| \frac{u_i}{\lambda_i} \right|^{q_i} \right]$$



$$\log \lambda_i = [\mathbf{B}\boldsymbol{v}]_i$$



Independent density components

$$-\log p(\boldsymbol{u}|\mathbf{B},\boldsymbol{v}) \propto \sum_i \left| \frac{u_i}{\exp([\mathbf{B}\boldsymbol{v}]_i)} \right|^{q_i}$$

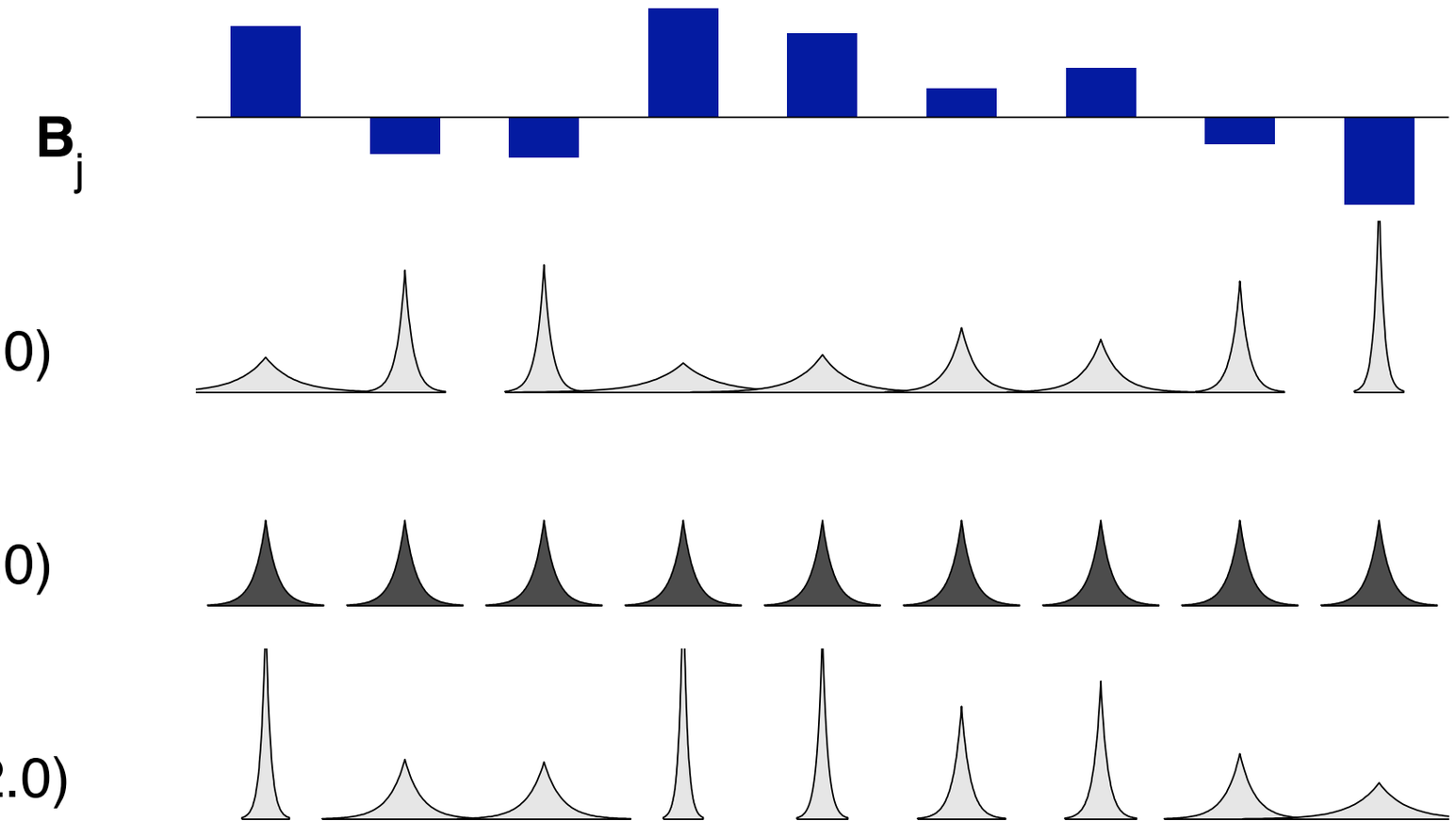
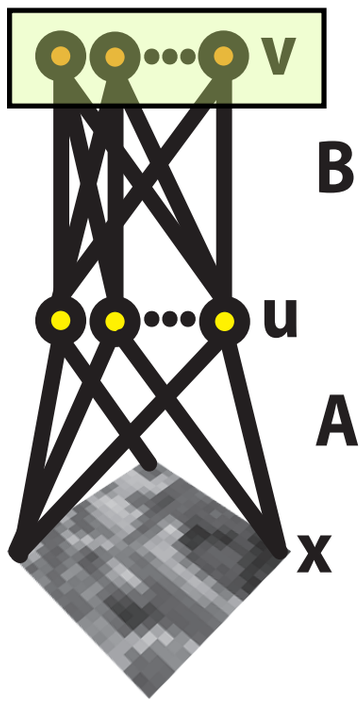


Illustration of inference in the model using synthesized data



$$P(v_i) \propto \exp(-|v_i|^{q_i})$$

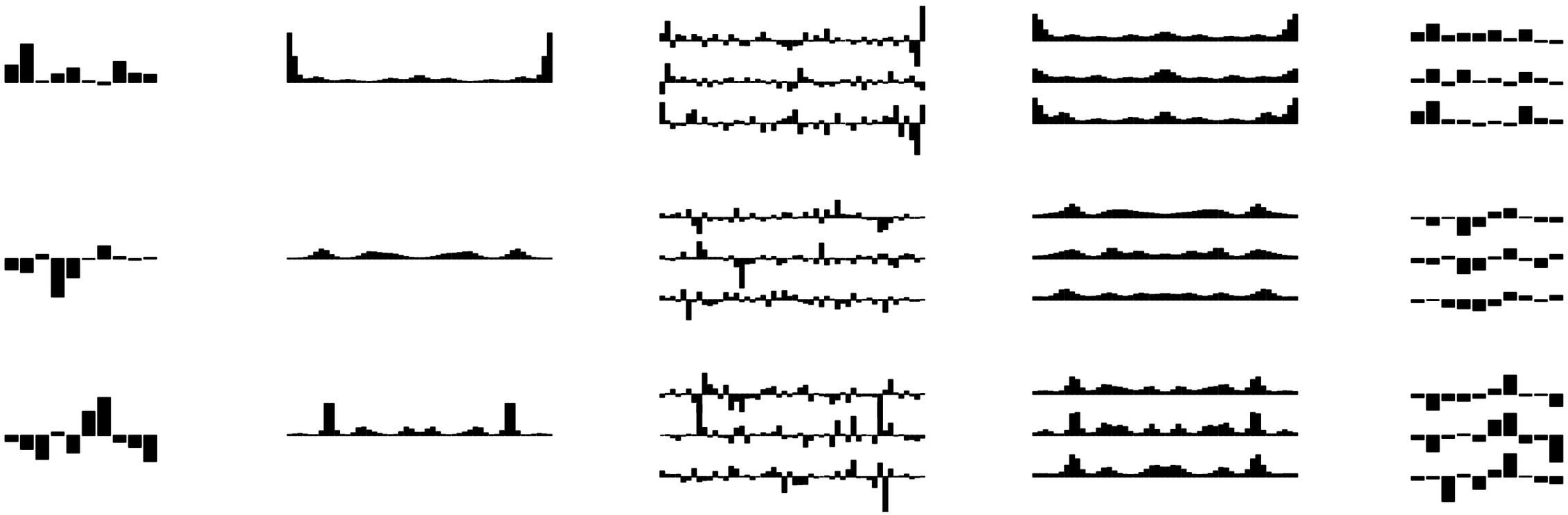
$$\log \lambda_i = [\mathbf{B}\boldsymbol{v}]_i$$

$$P(u_i) \propto \exp\left[-\left|\frac{u_i}{\lambda_i}\right|^{q_i}\right]$$

$$\hat{\boldsymbol{v}} = \arg\max_{v} P(\boldsymbol{v}|\mathbf{B}, \boldsymbol{u})$$

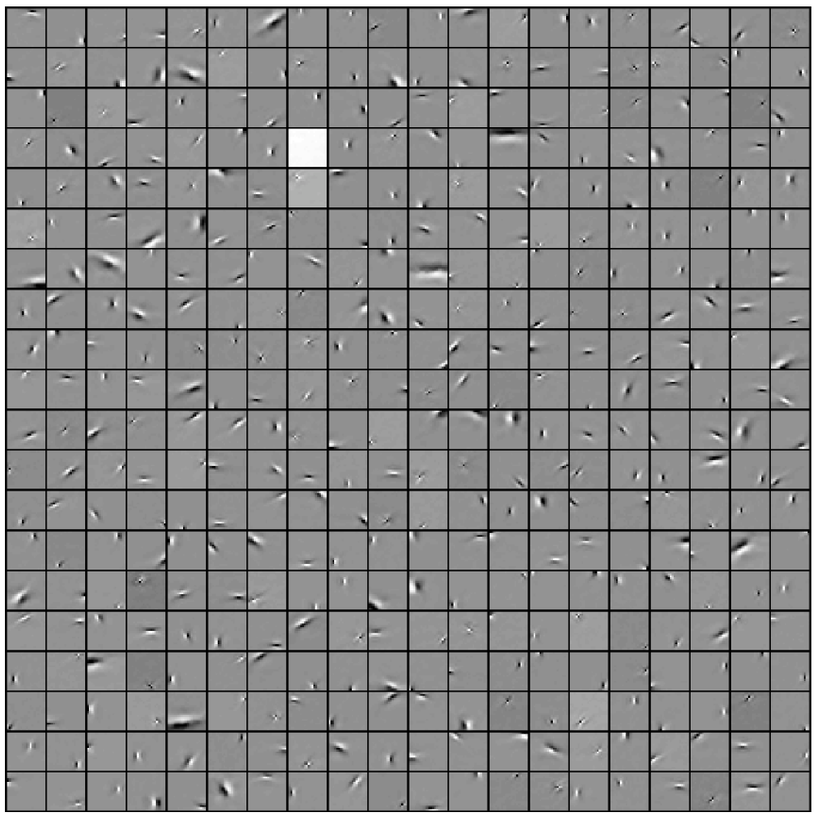
Synthesis → Inference

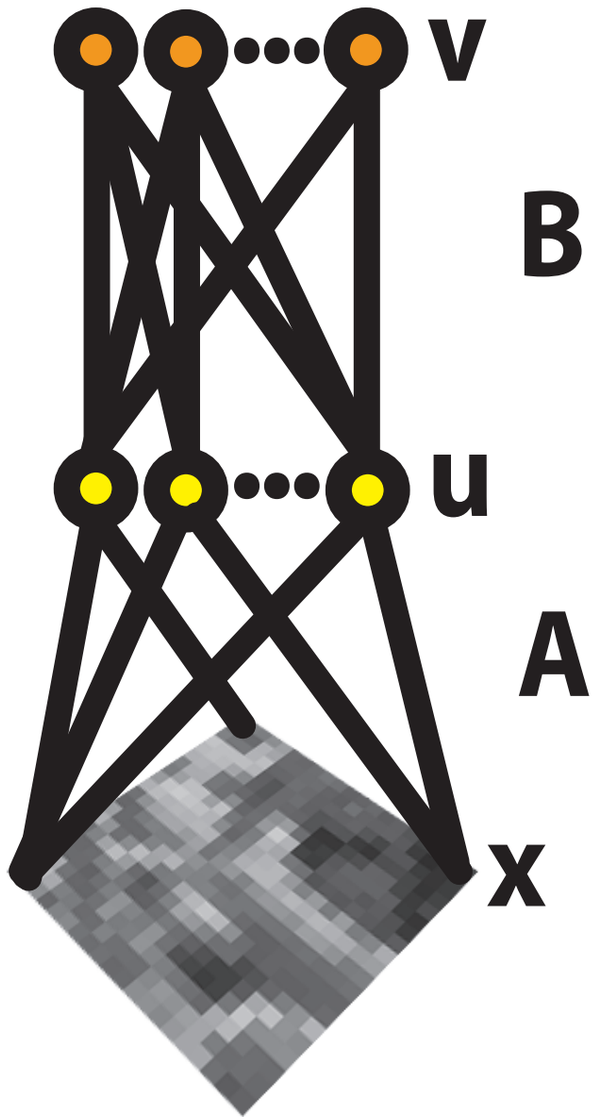
$\mathbf{v} \sim p(\mathbf{v})$ → $\boldsymbol{\lambda} = c\exp(\mathbf{B}\mathbf{v})$ → $\mathbf{u} \sim p(\mathbf{u}|\boldsymbol{\lambda})$ → $\hat{\boldsymbol{\lambda}}$ → $\hat{\mathbf{v}}$



Learning higher-order structure of natural images

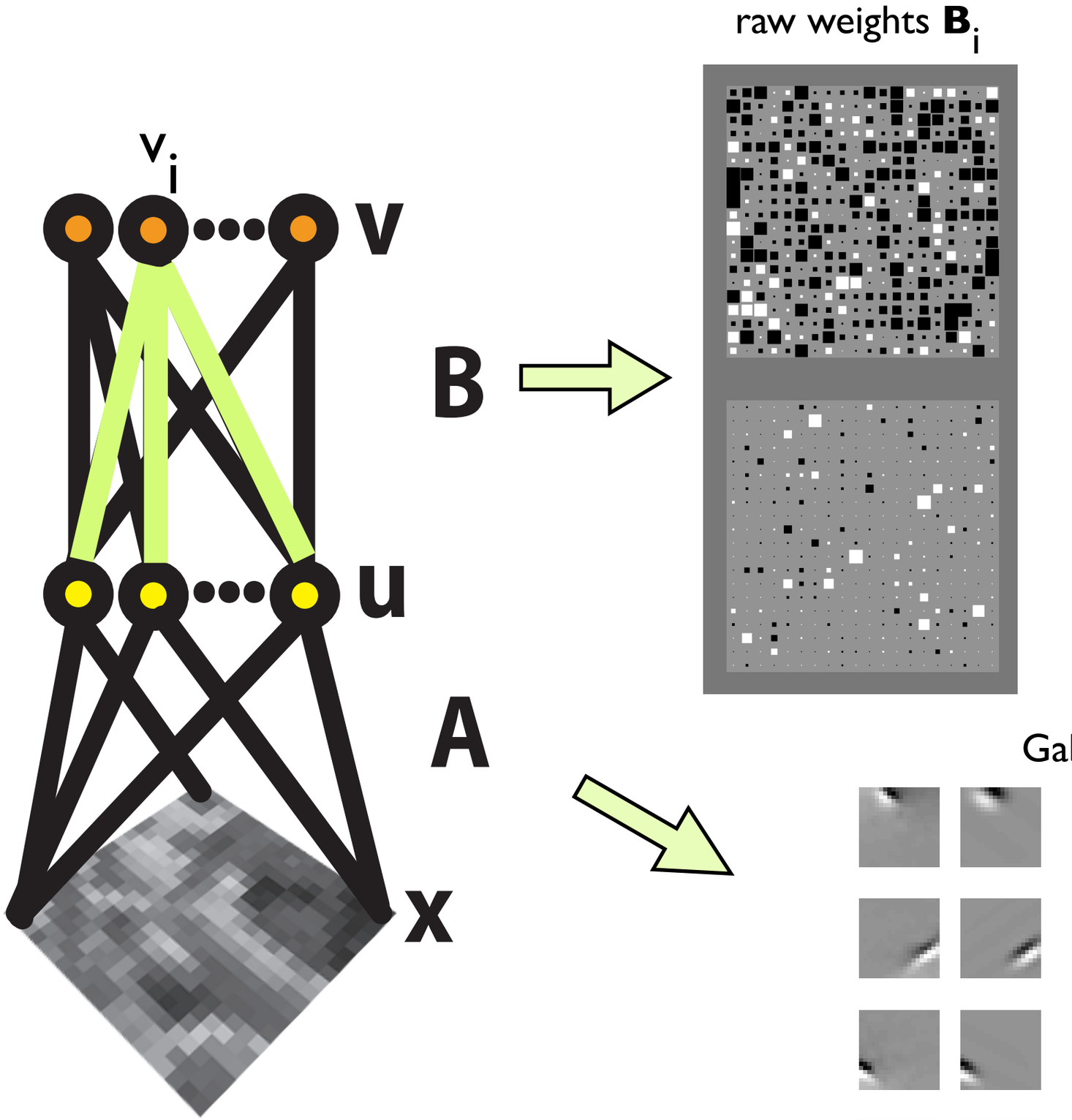
- **A** is learned with ICA
- **B** is learned by maximizing the posterior distribution





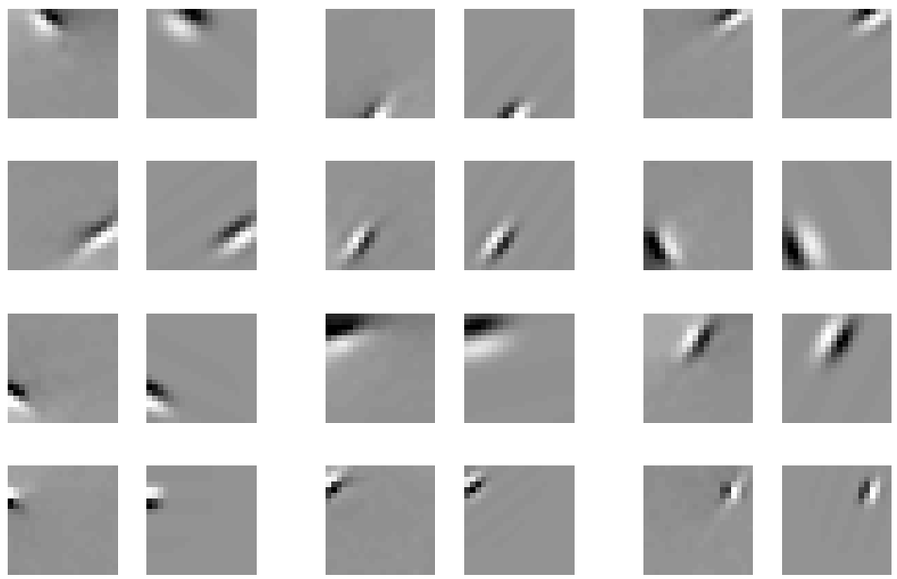
Train on natural images

Interpreting higher-order density components

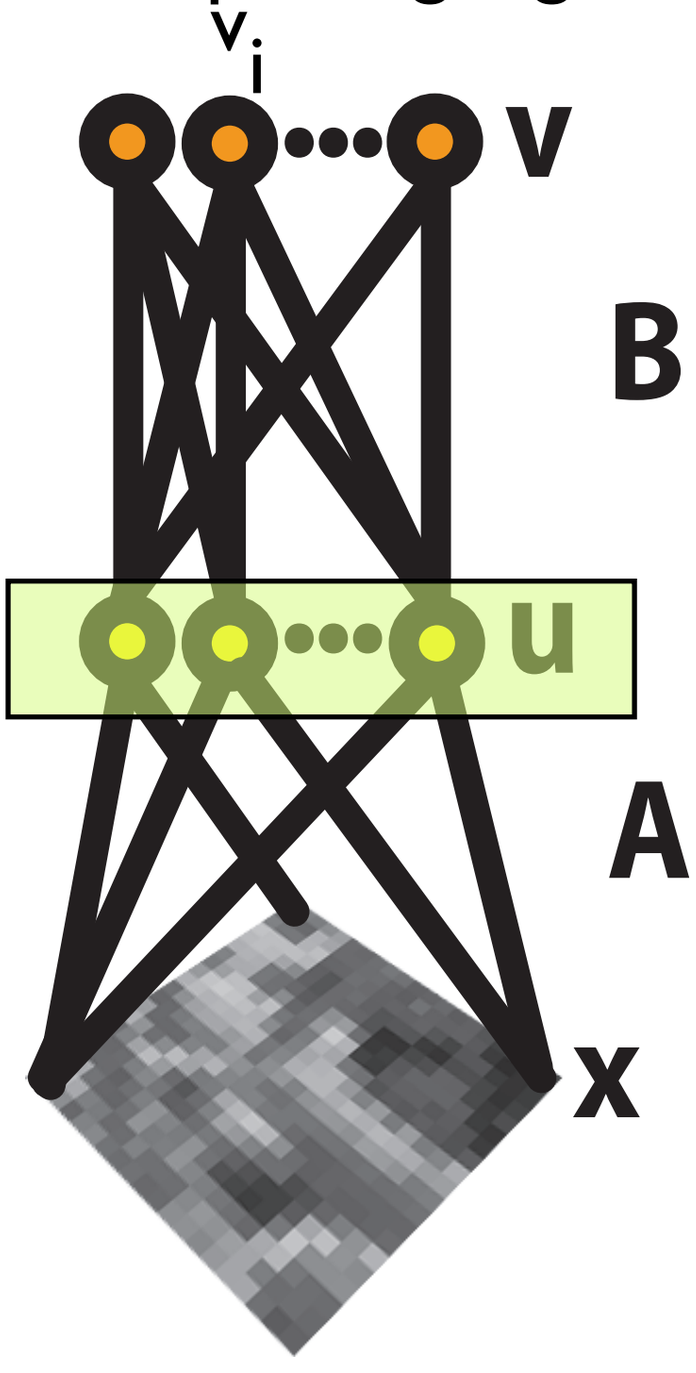


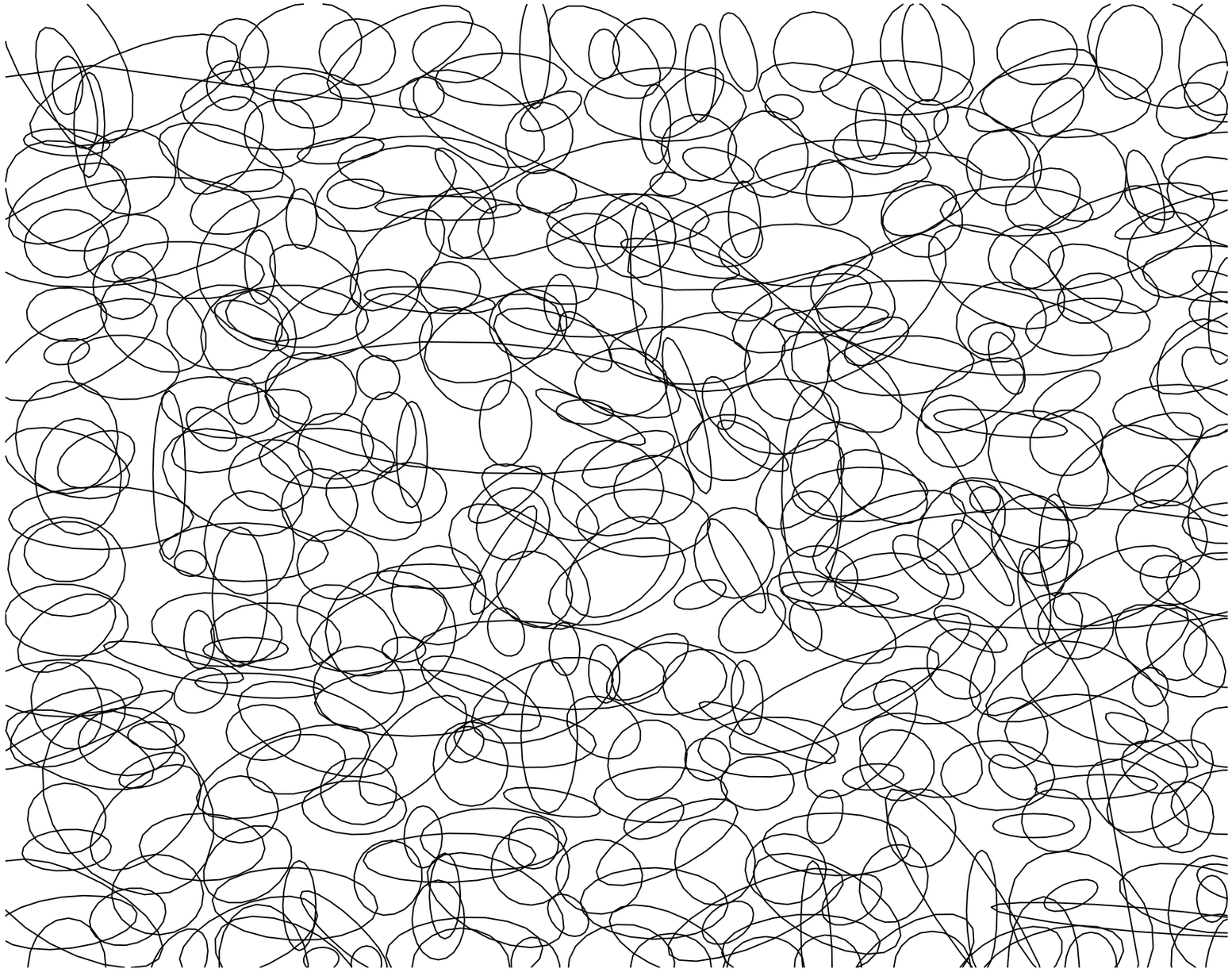
raw weights $\mathbf{B}_i$

Gabor function fits

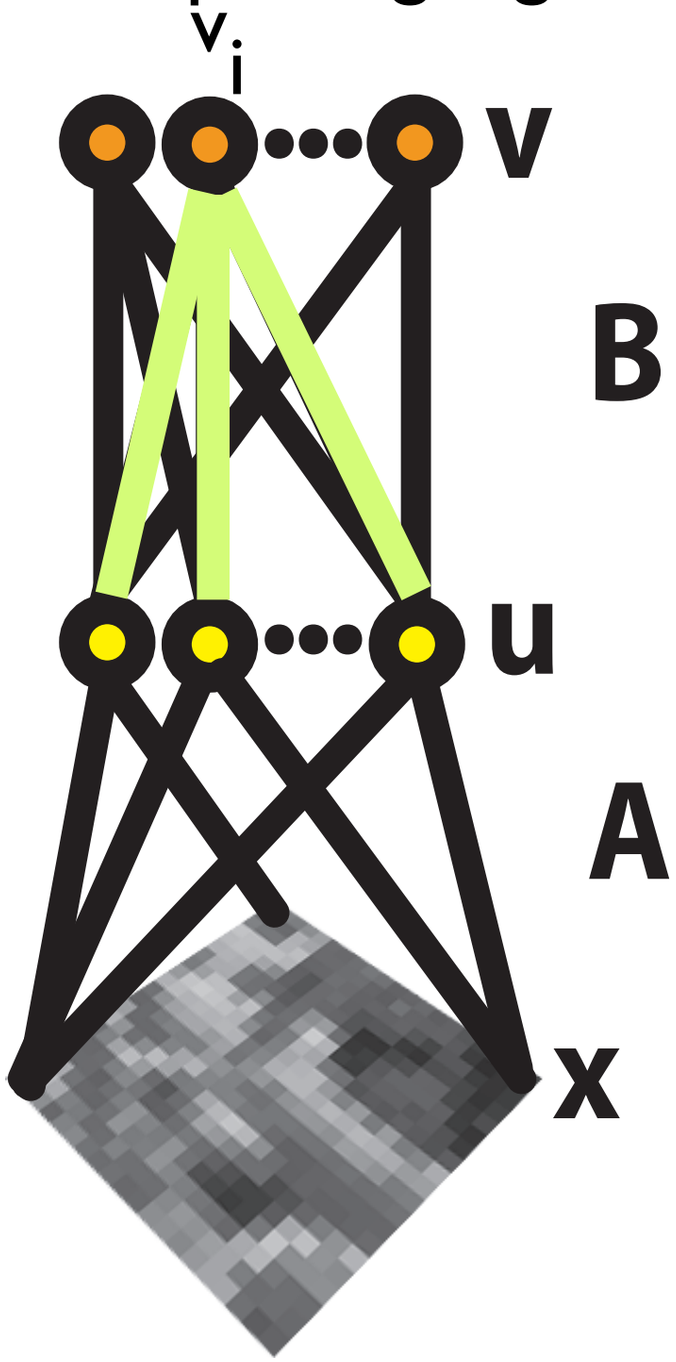


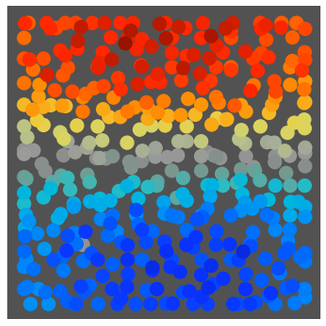
Interpreting higher-order density components



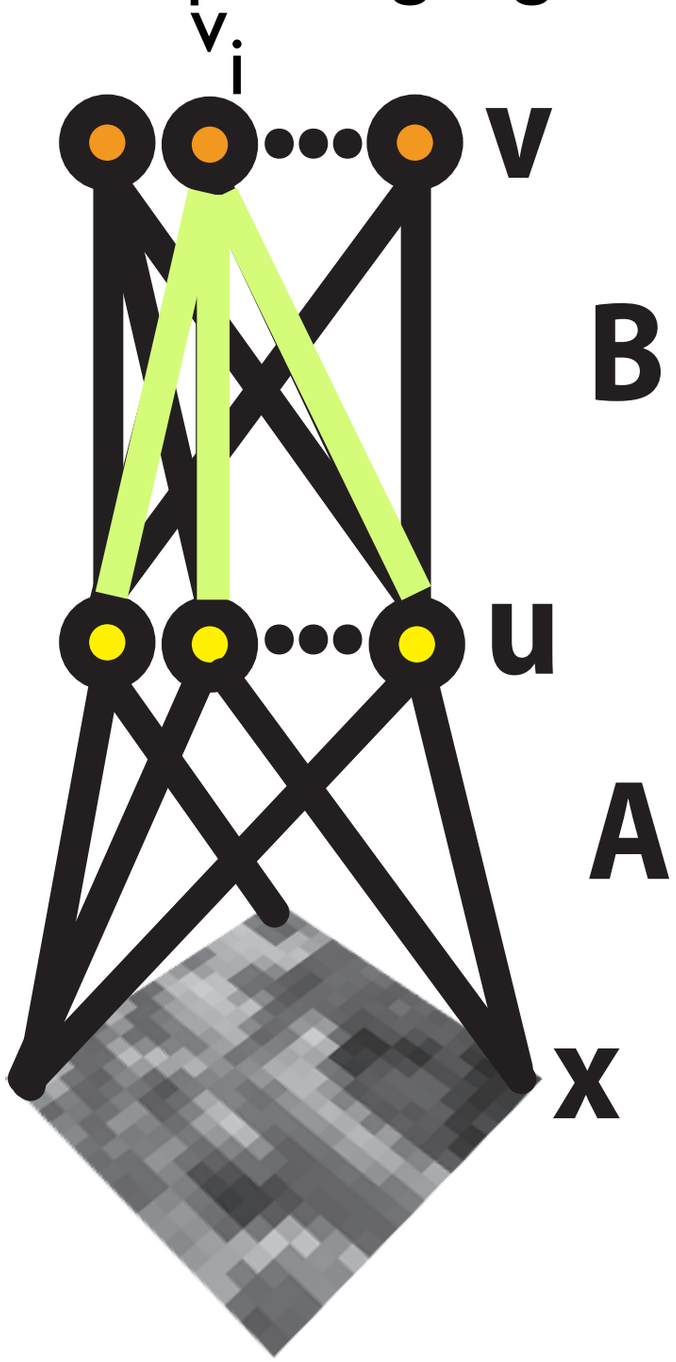


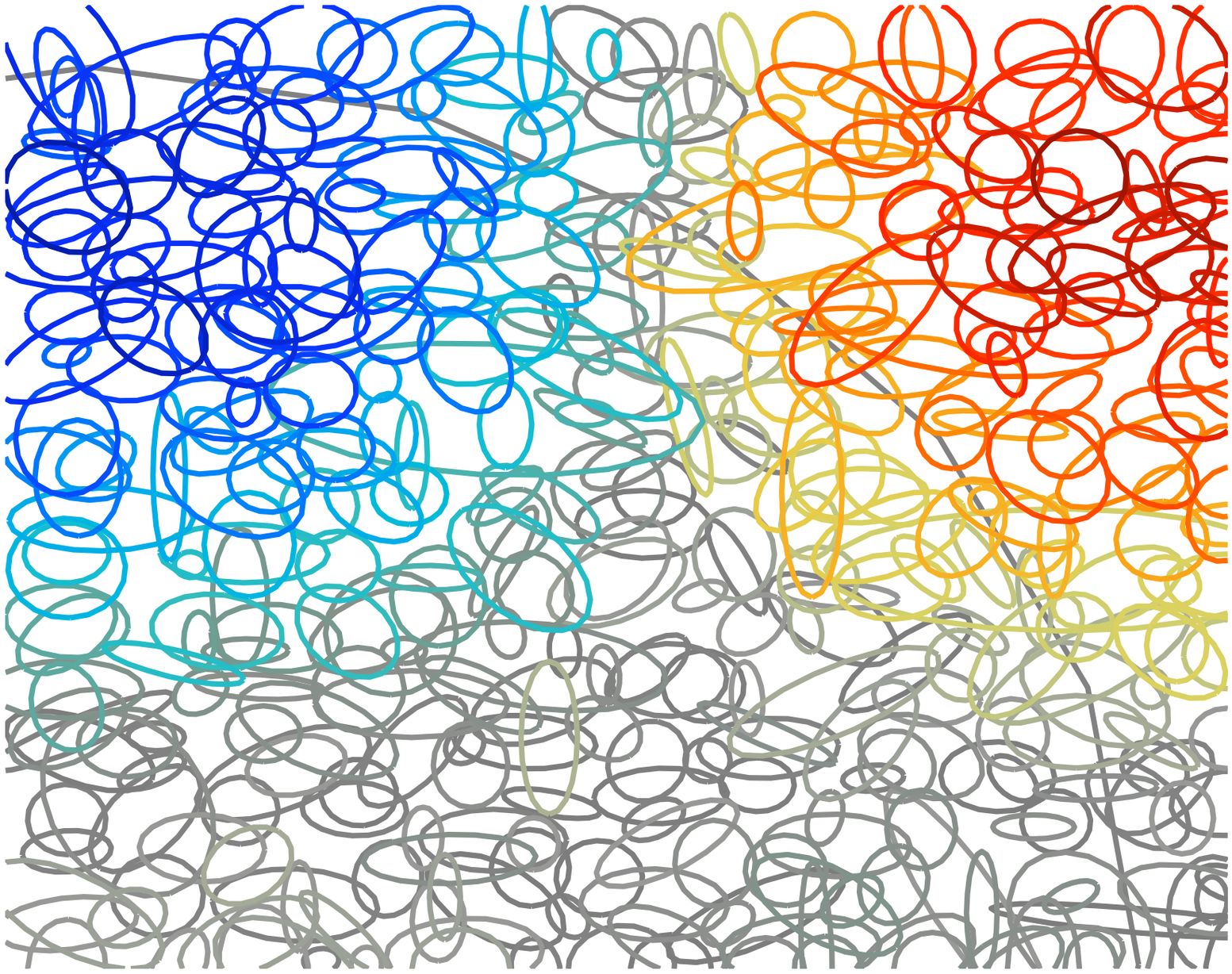
Interpreting higher-order density components

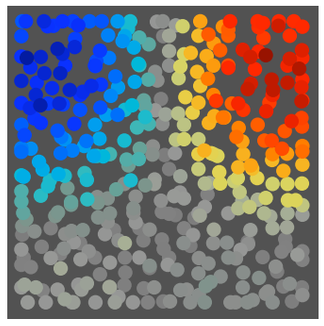




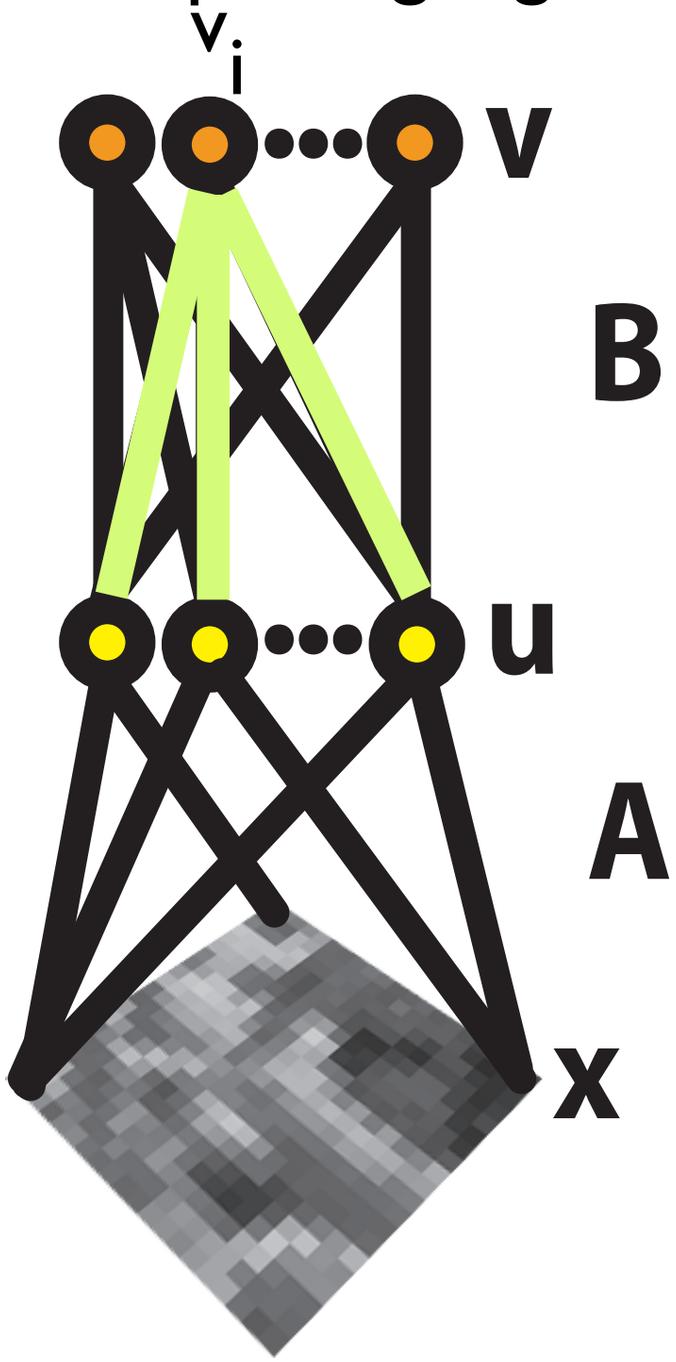
Interpreting higher-order density components

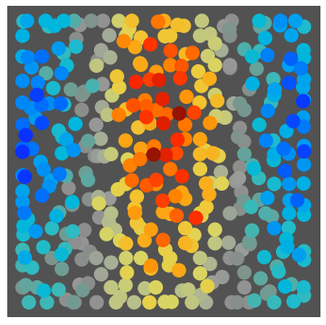




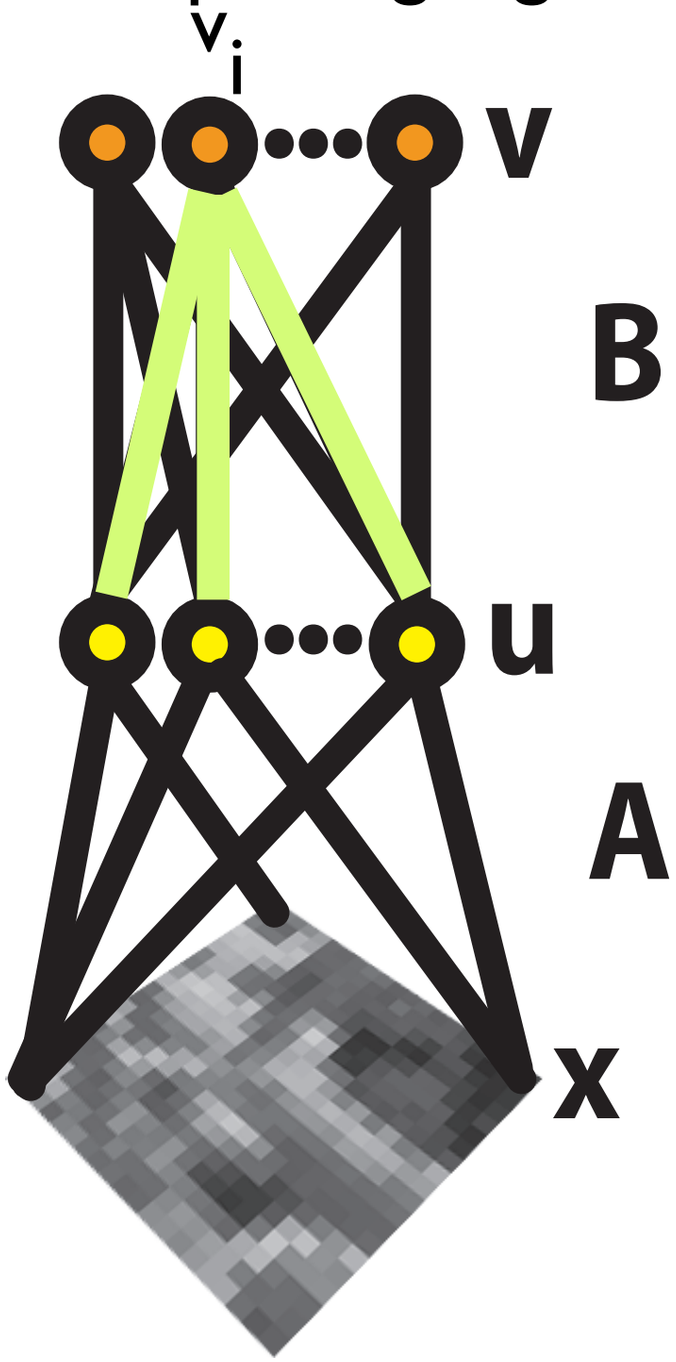


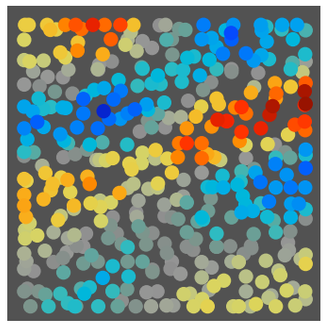
Interpreting higher-order density components



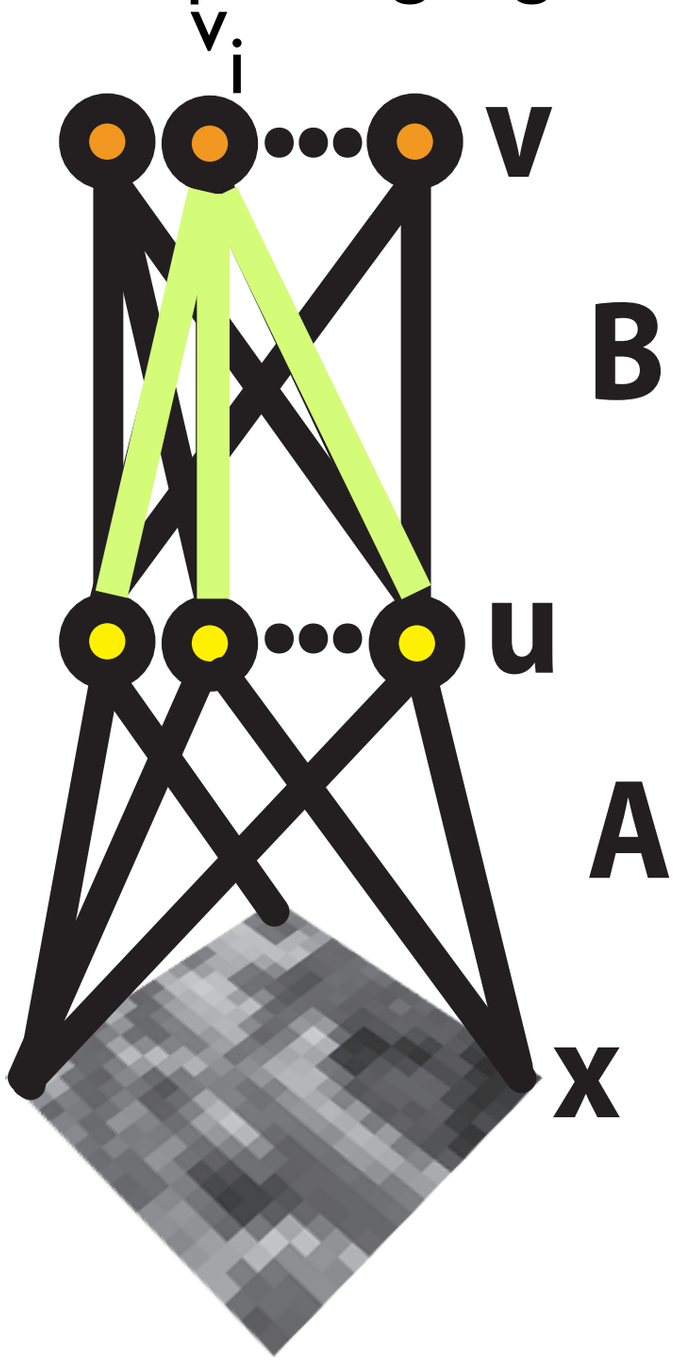


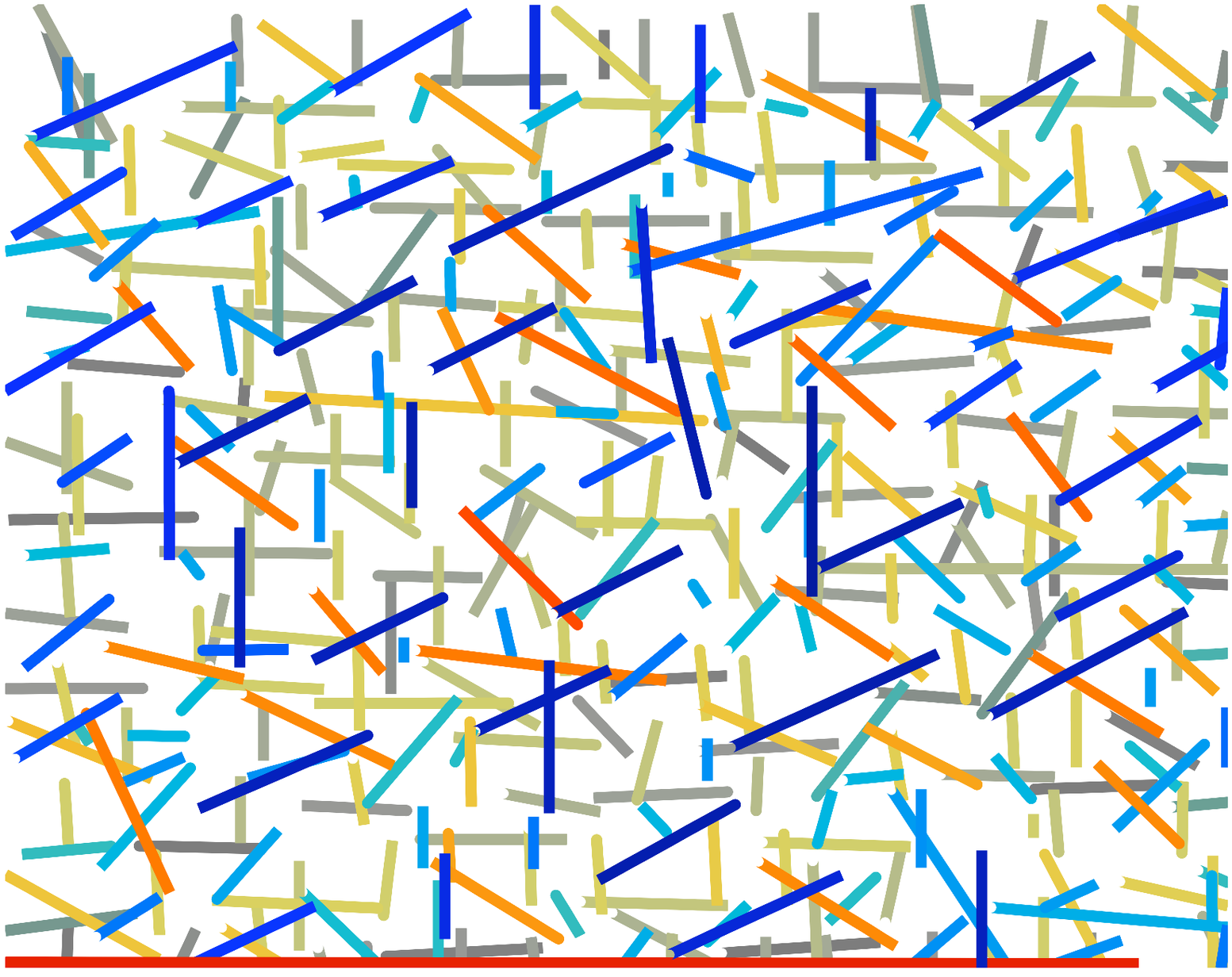
Interpreting higher-order density components

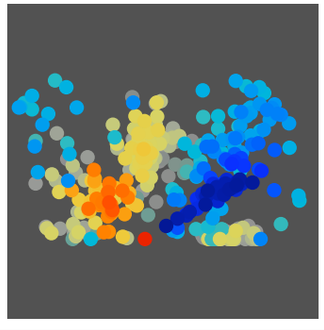




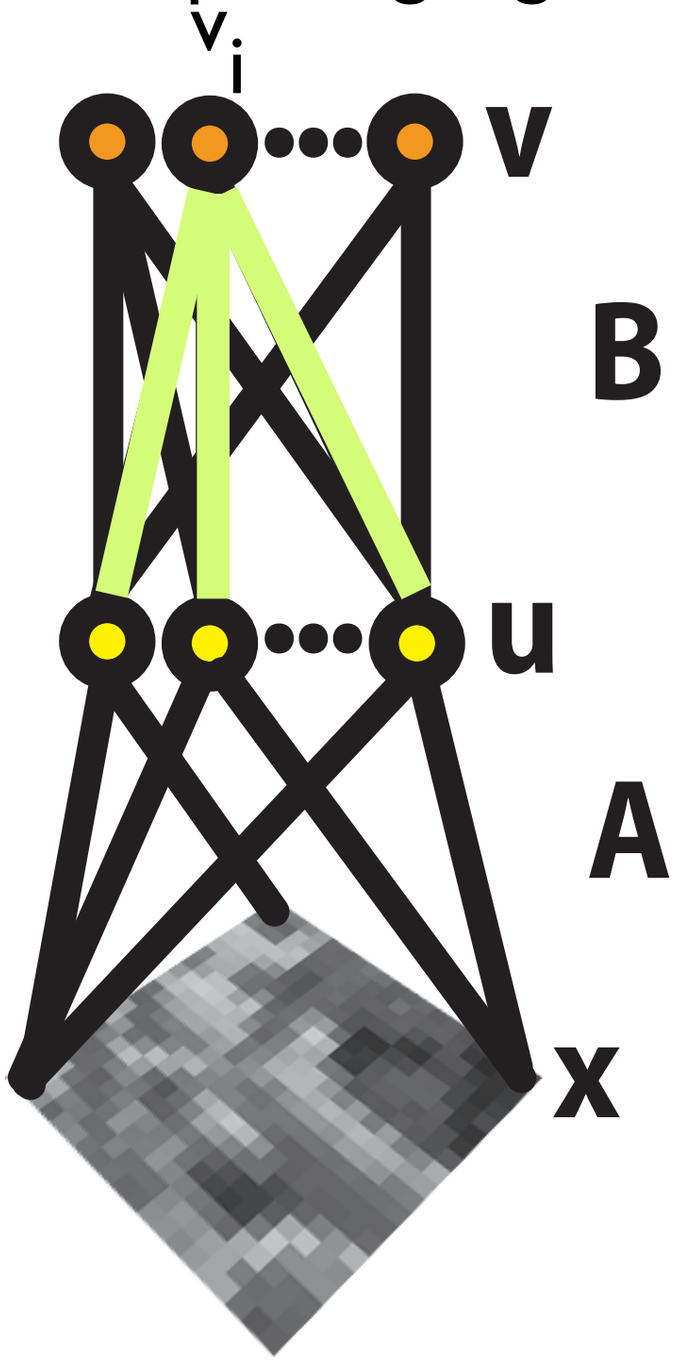
Interpreting higher-order density components

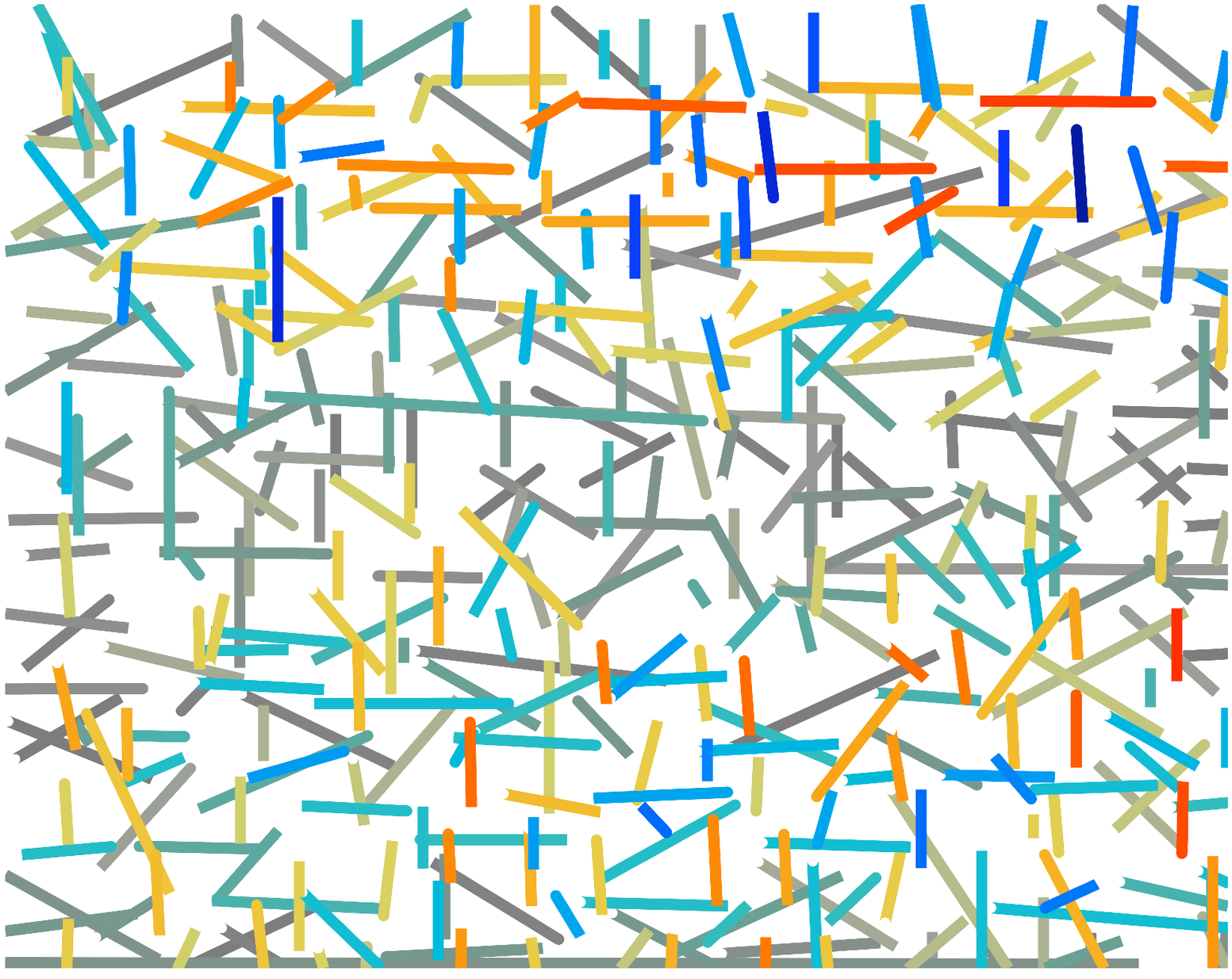


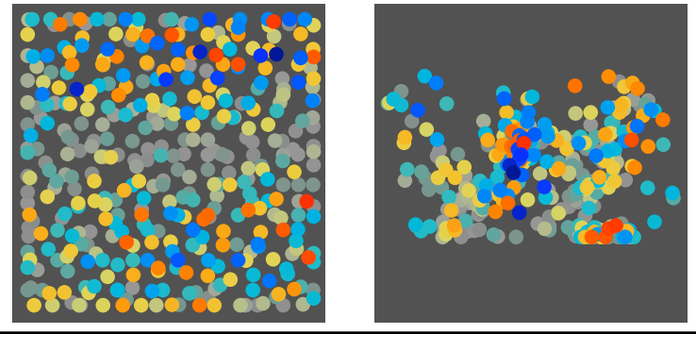




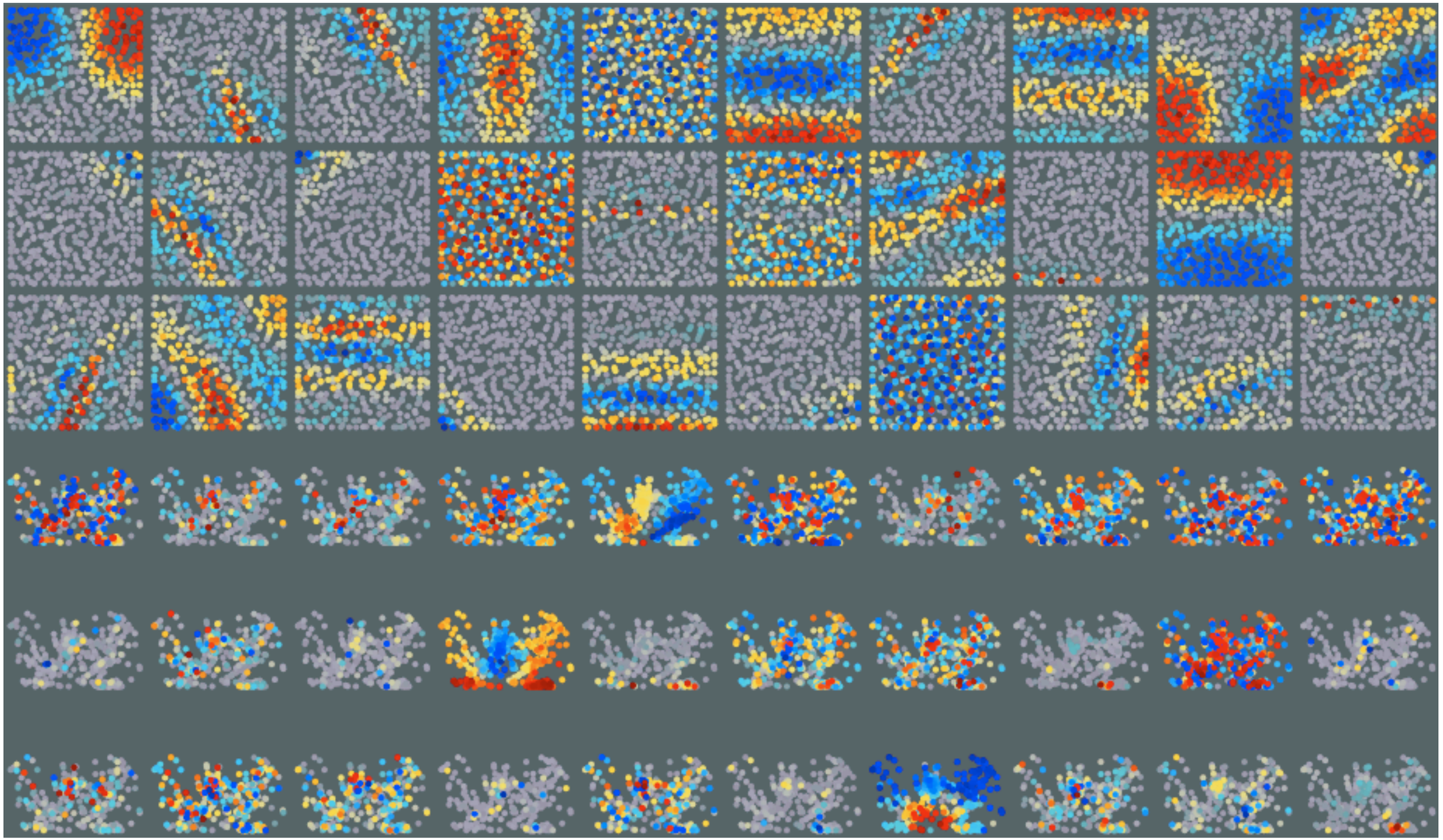
Interpreting higher-order density components





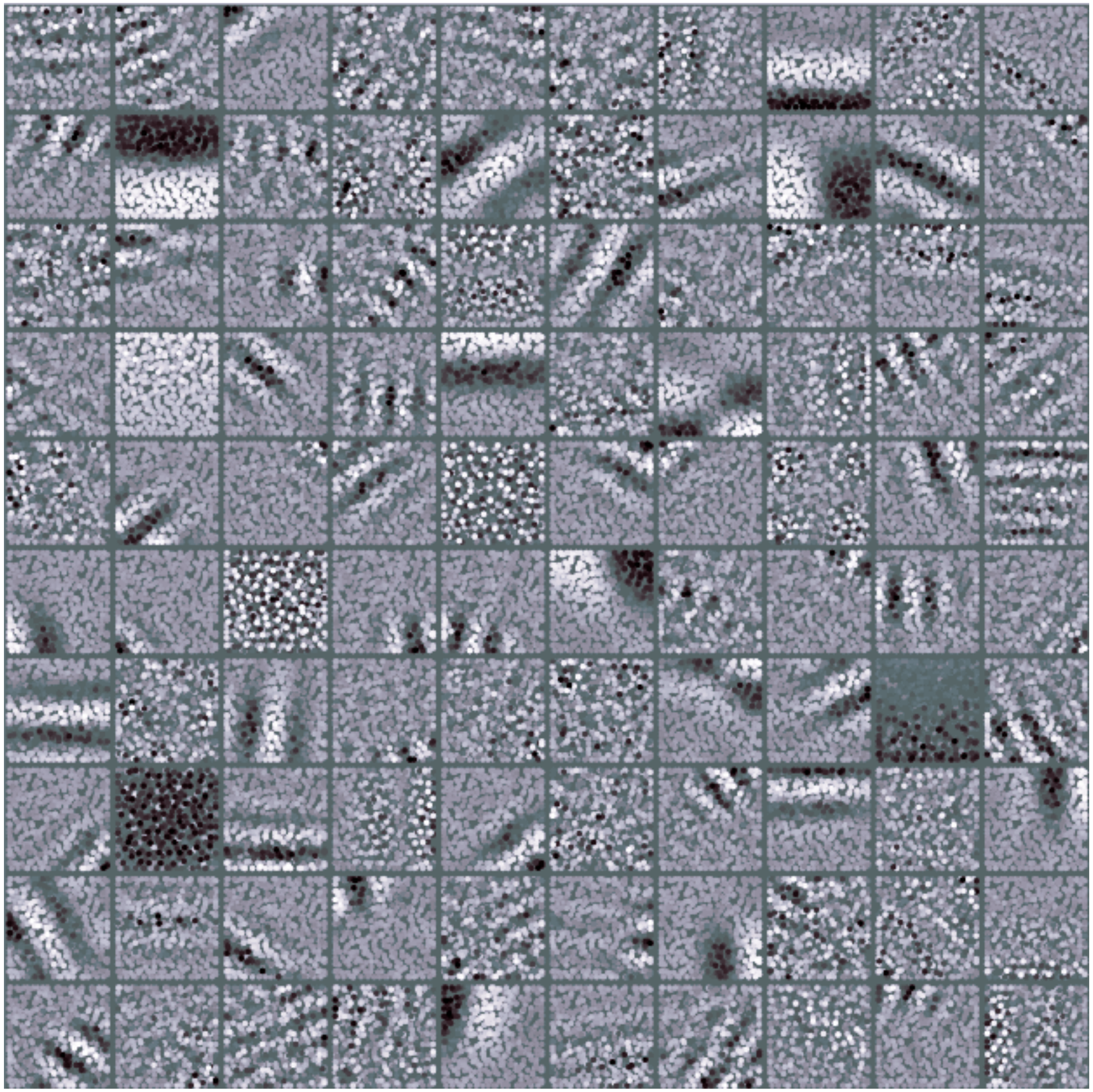


Learned density components of natural images



(30 out of 100 shown)

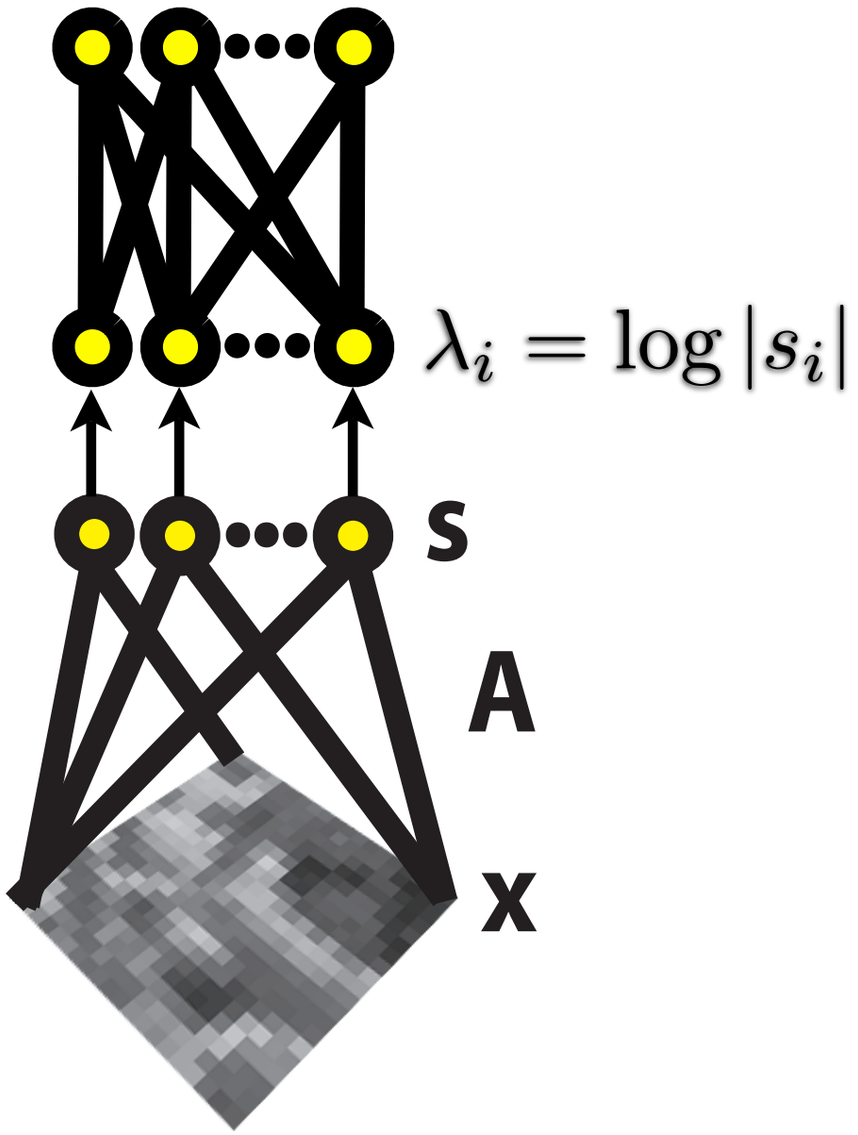
Full set of natural image density components



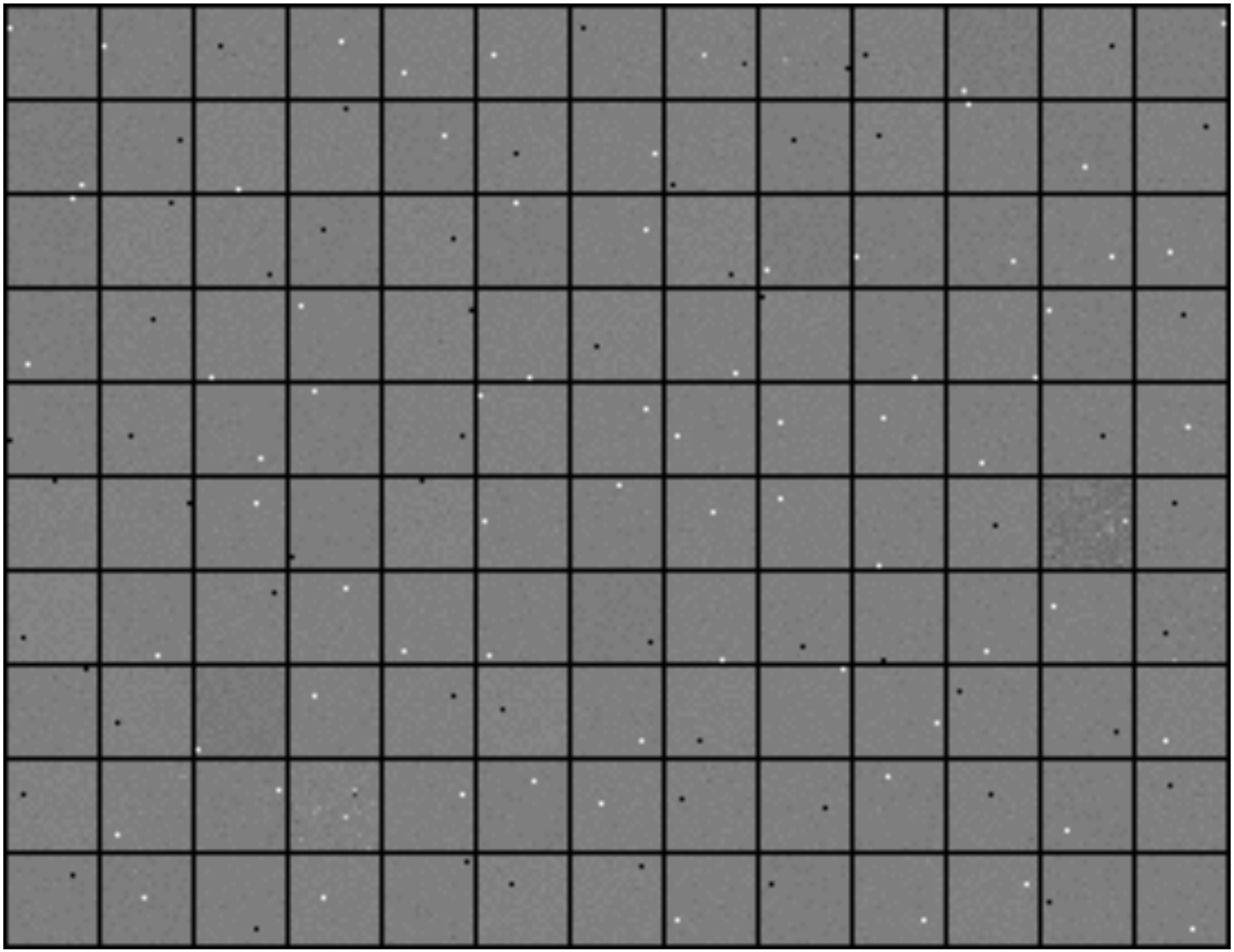
What about a feed-forward non-linearity?

Isn't this the same?

3. Do ICA again on output:
2. Add non-linearity:
1. Take standard ICA model:

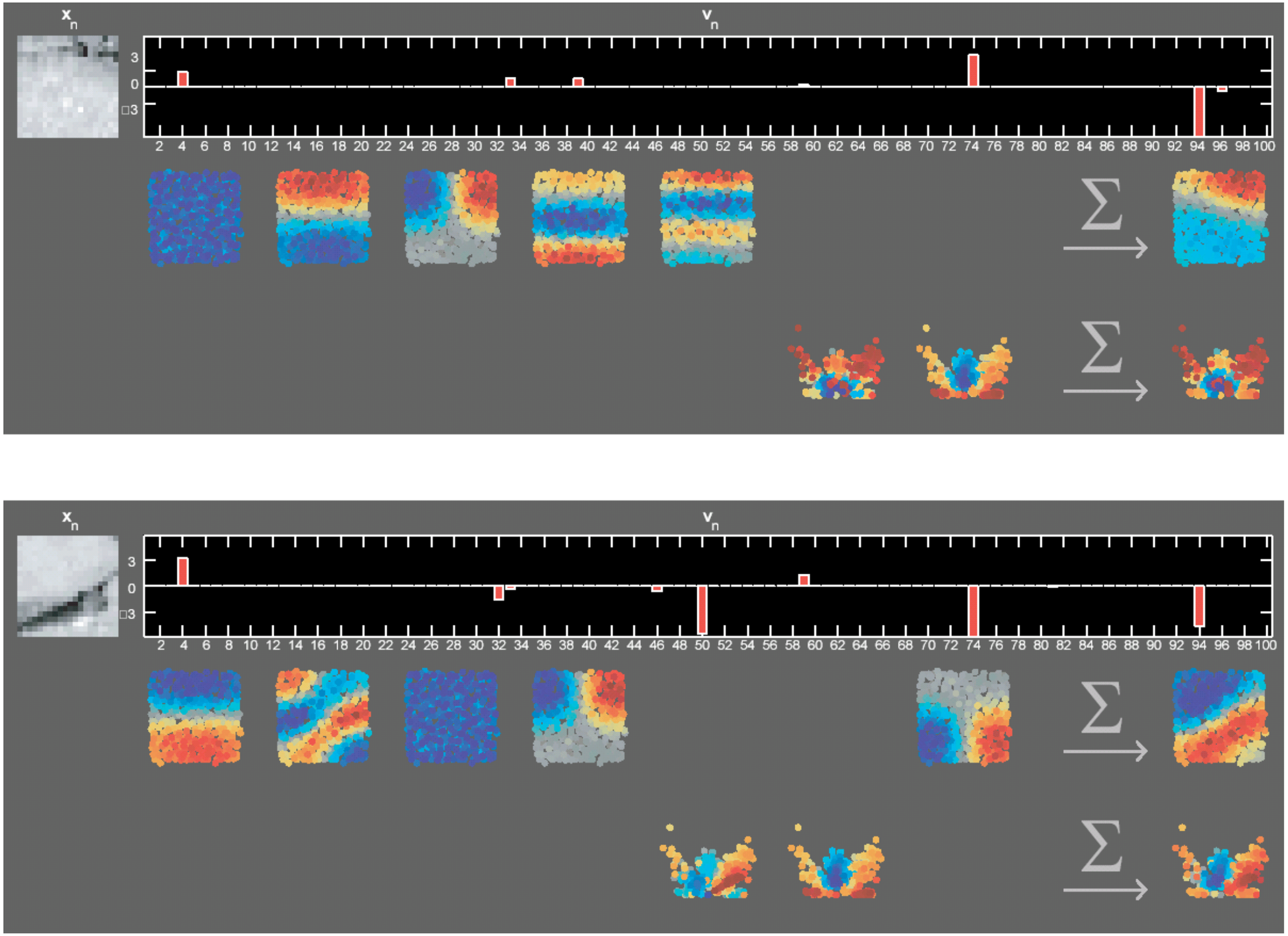


ICA on non-linearity reveals no structure

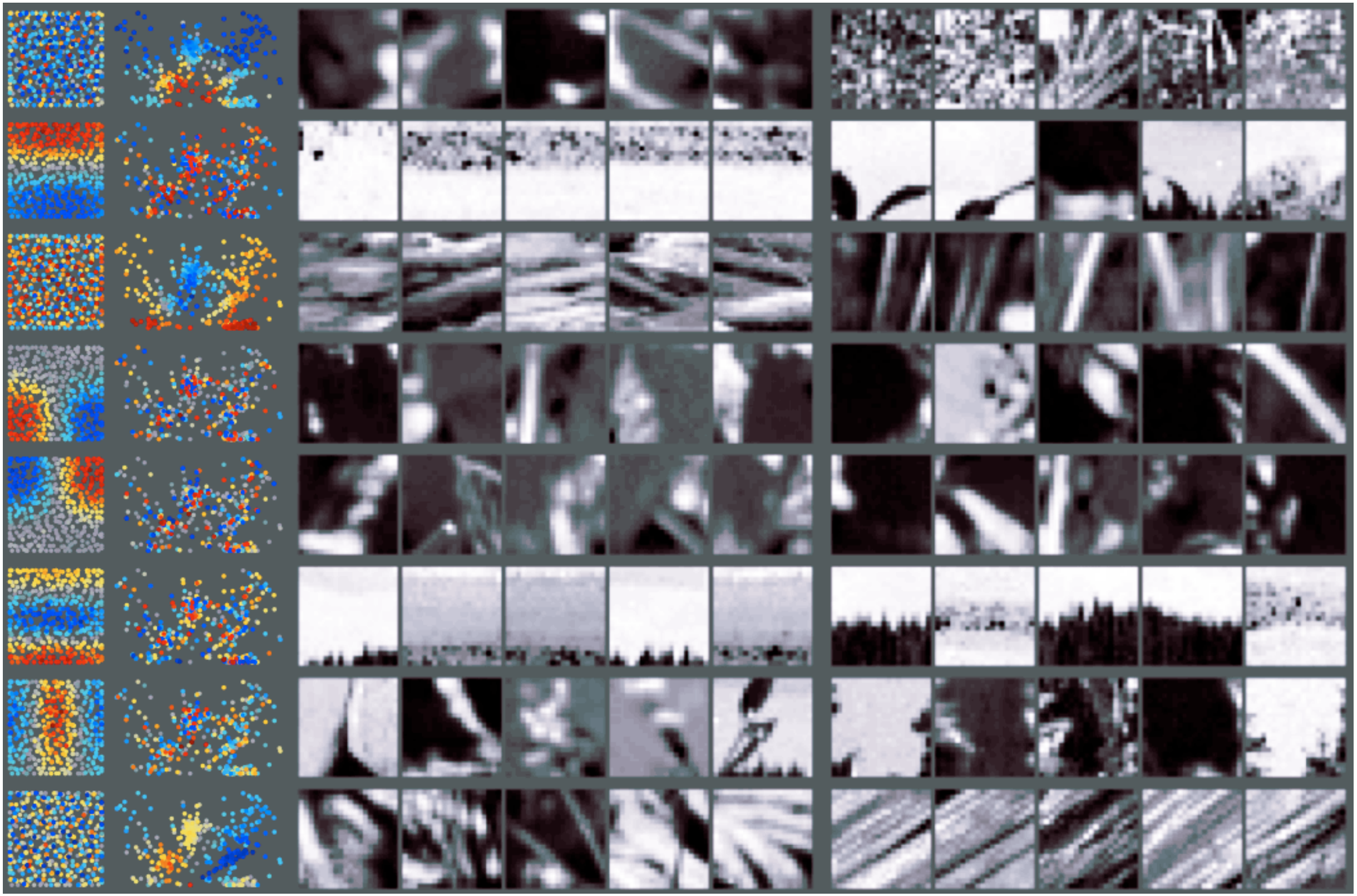


subset of ICA basis functions on log |s|

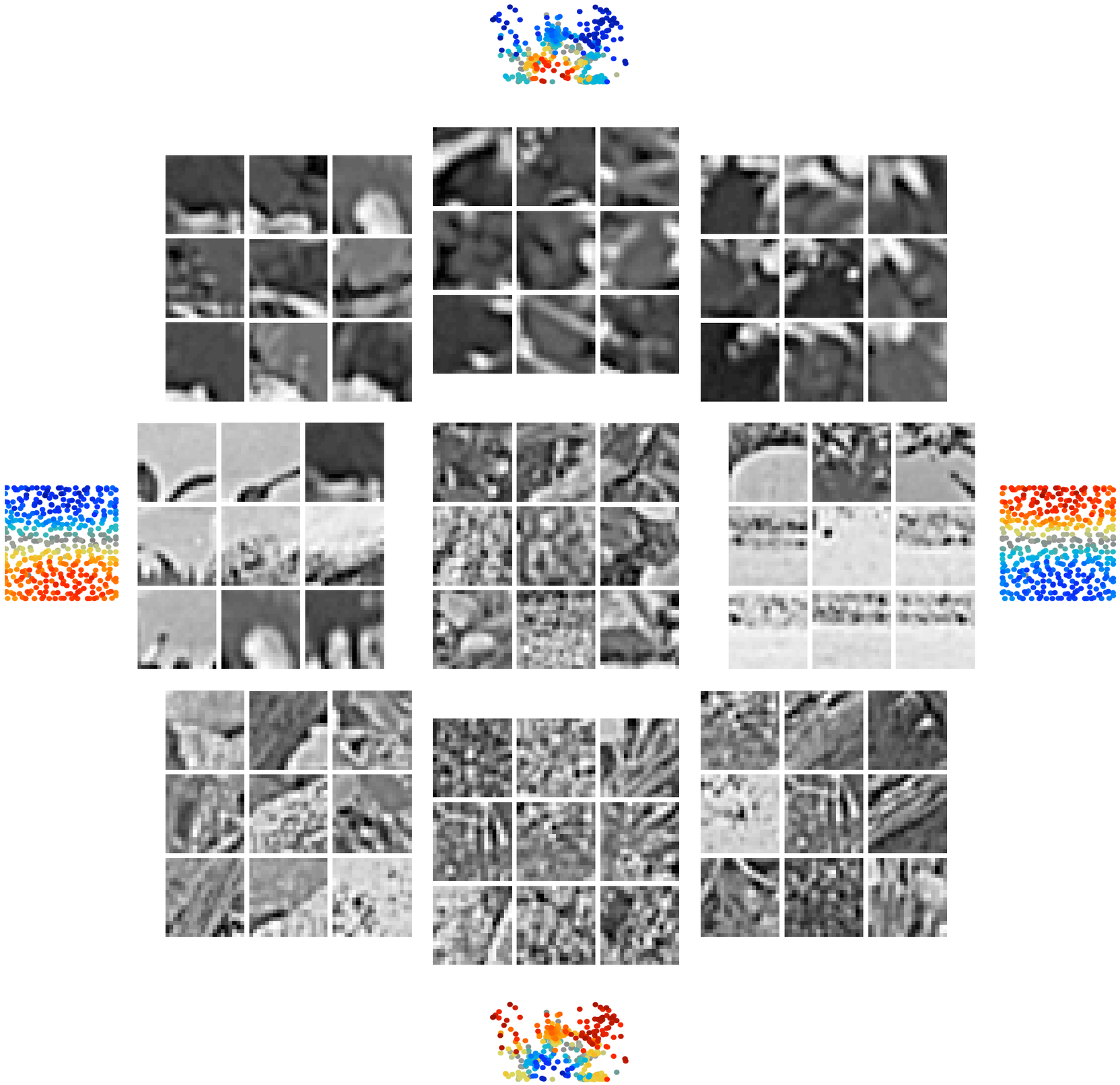
Inferred v forms a sparse distributed code



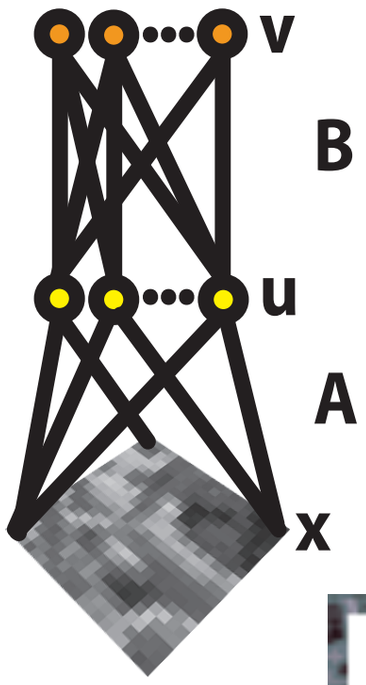
Most typical images for selected density components

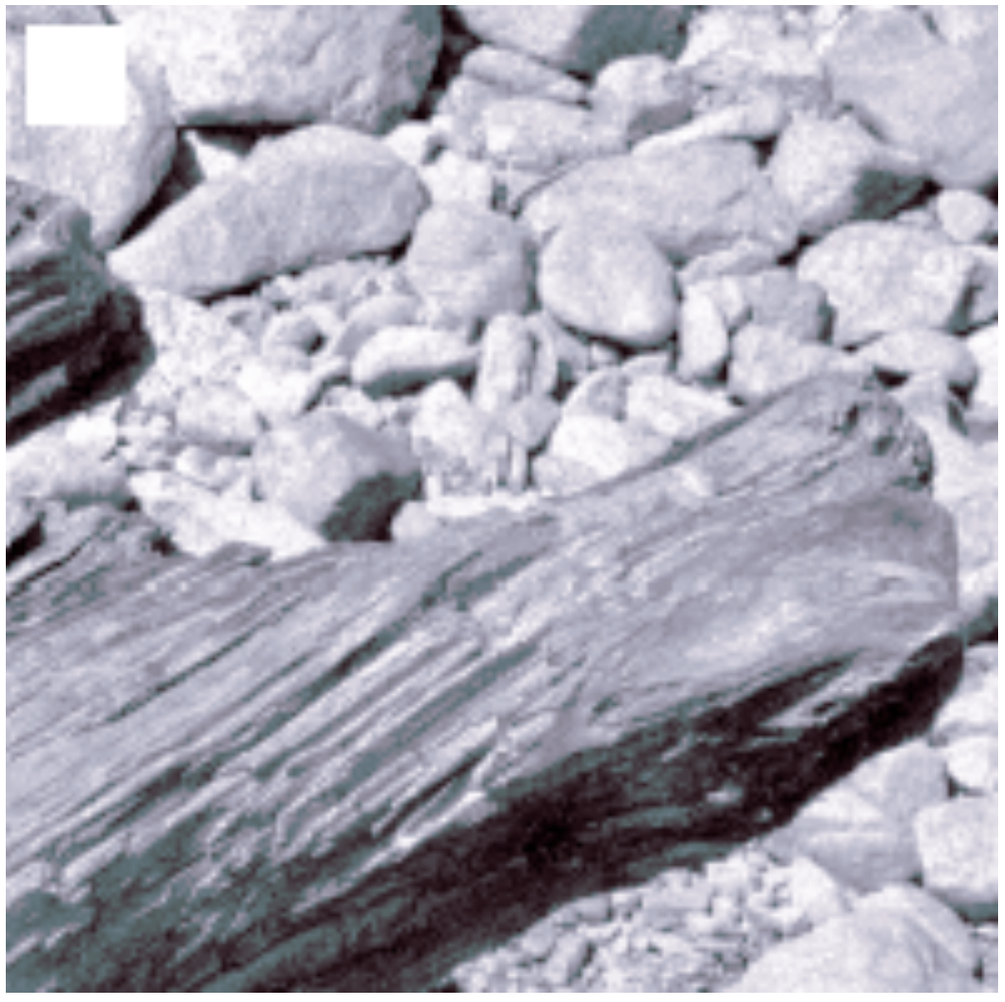


Distributed representation of natural image densities

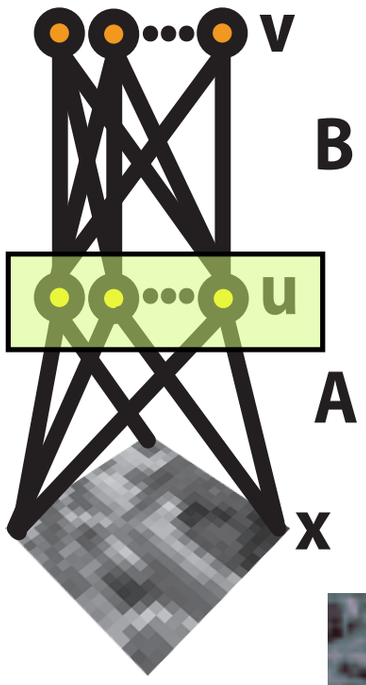


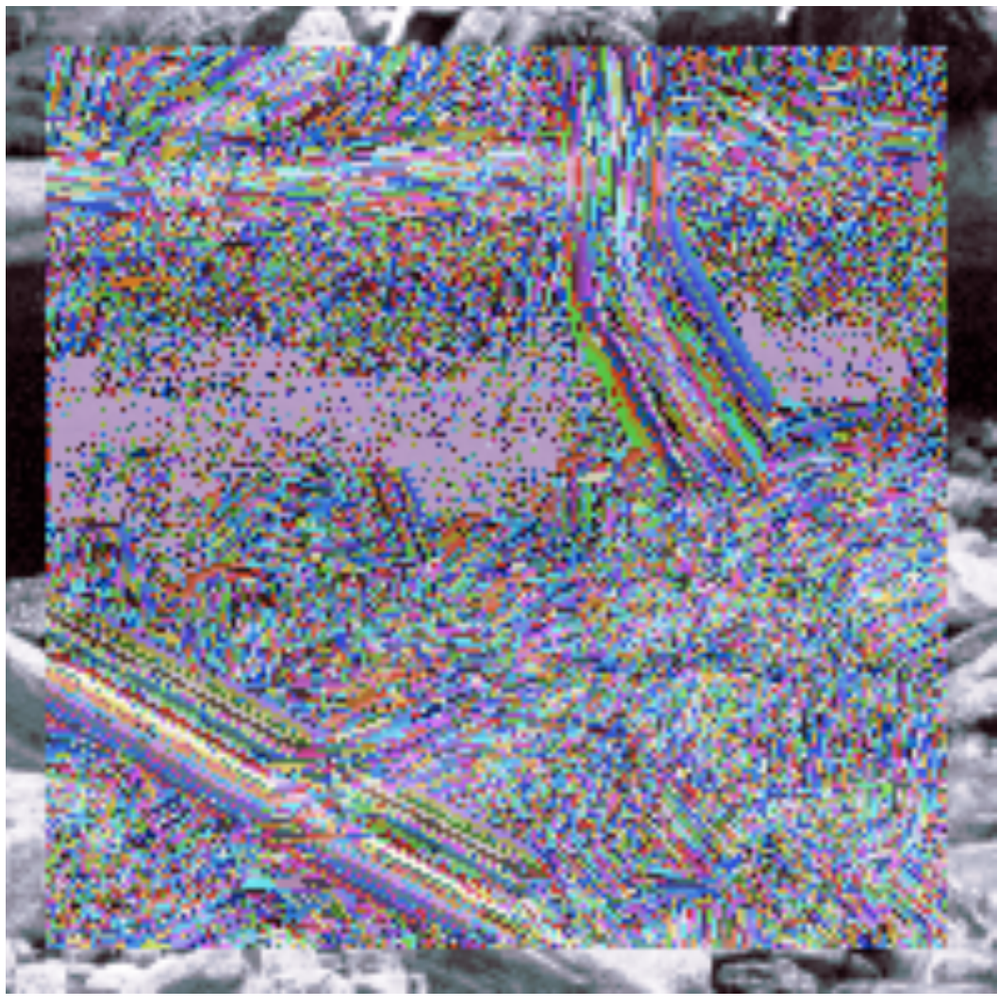
Comparing the degree of abstraction

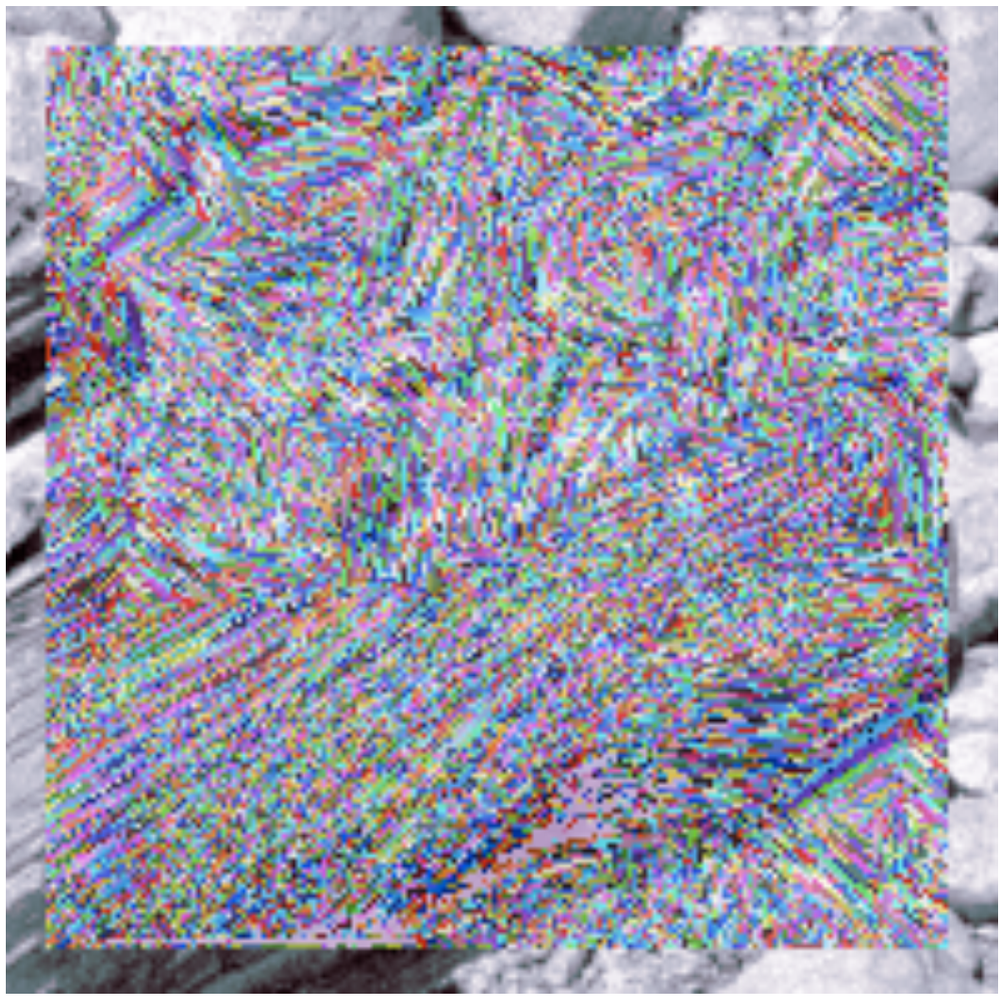




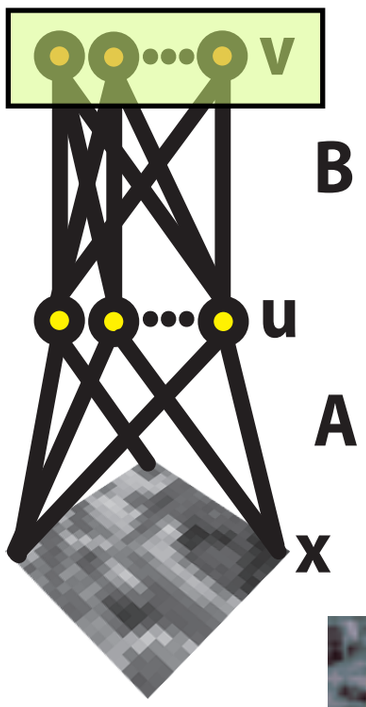
Winner maps: Maximum $|u_i|$ for each pixel

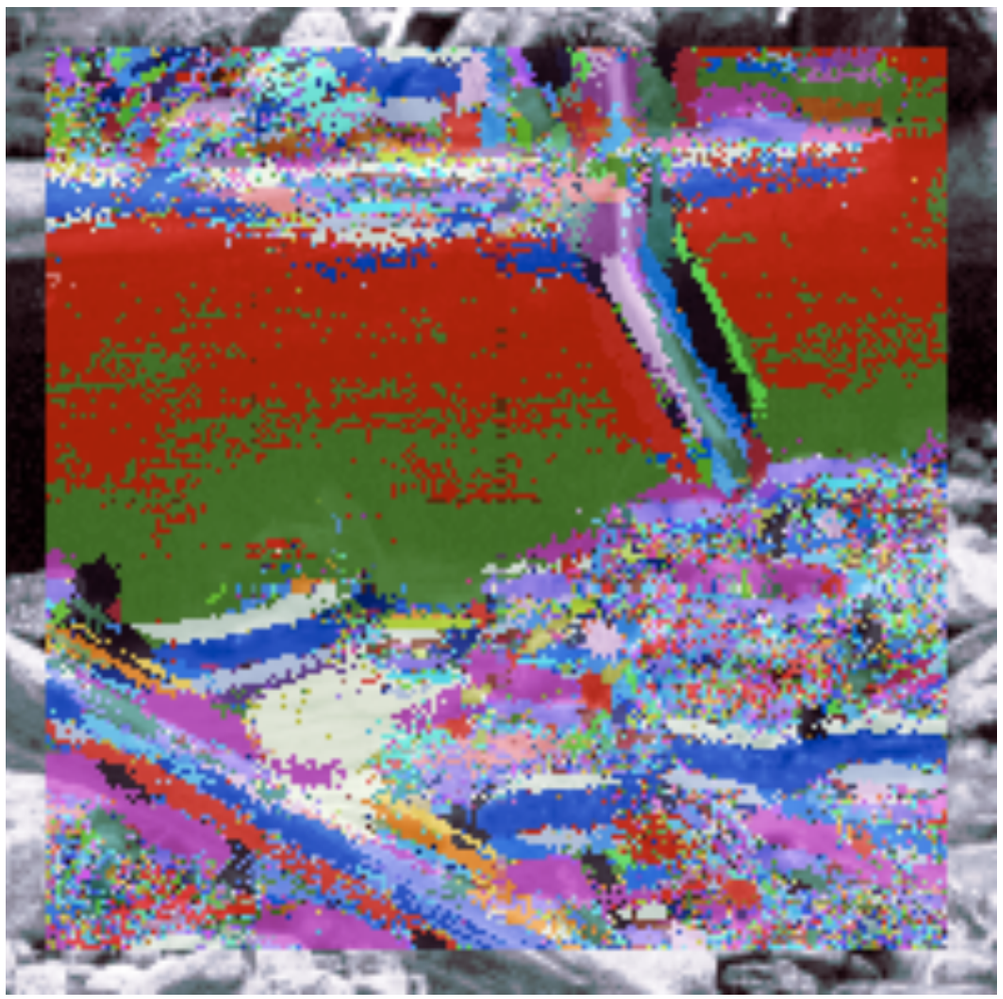


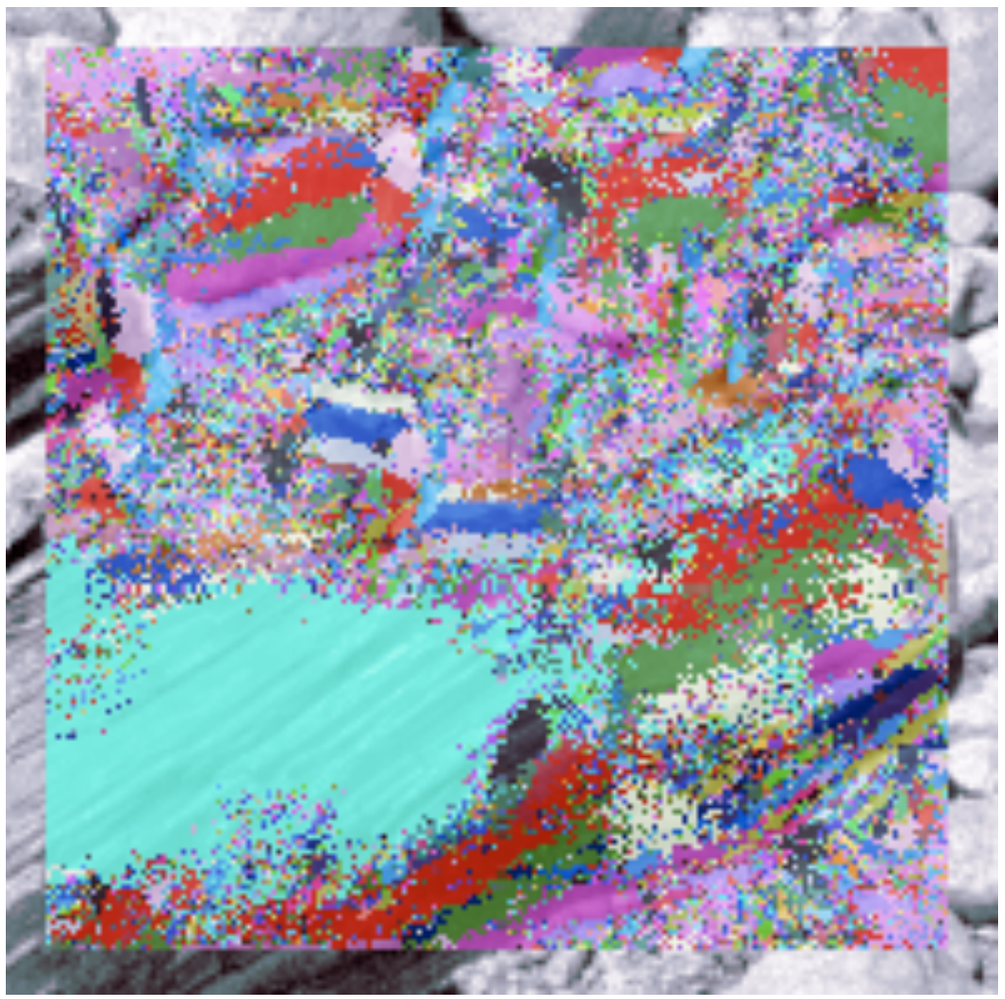




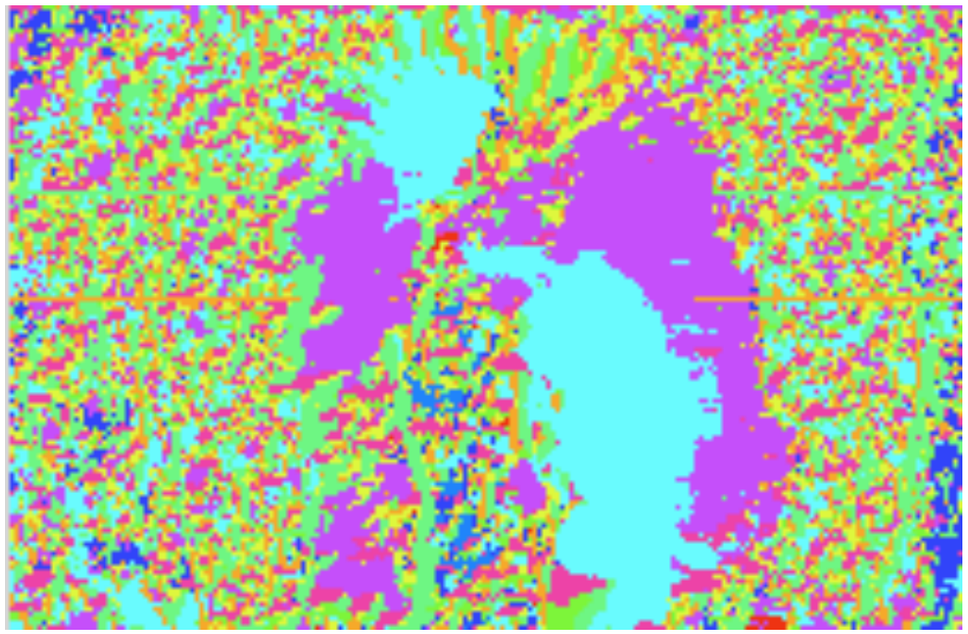
Winner maps: Maximum $|v_i|$ for each pixel

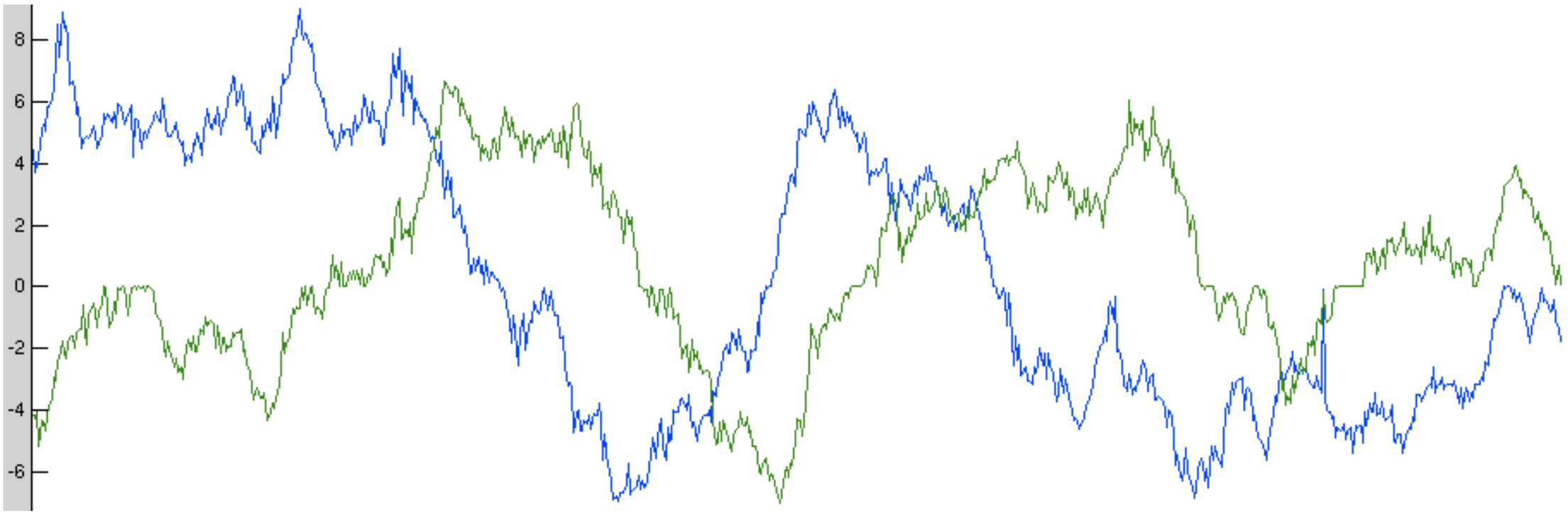




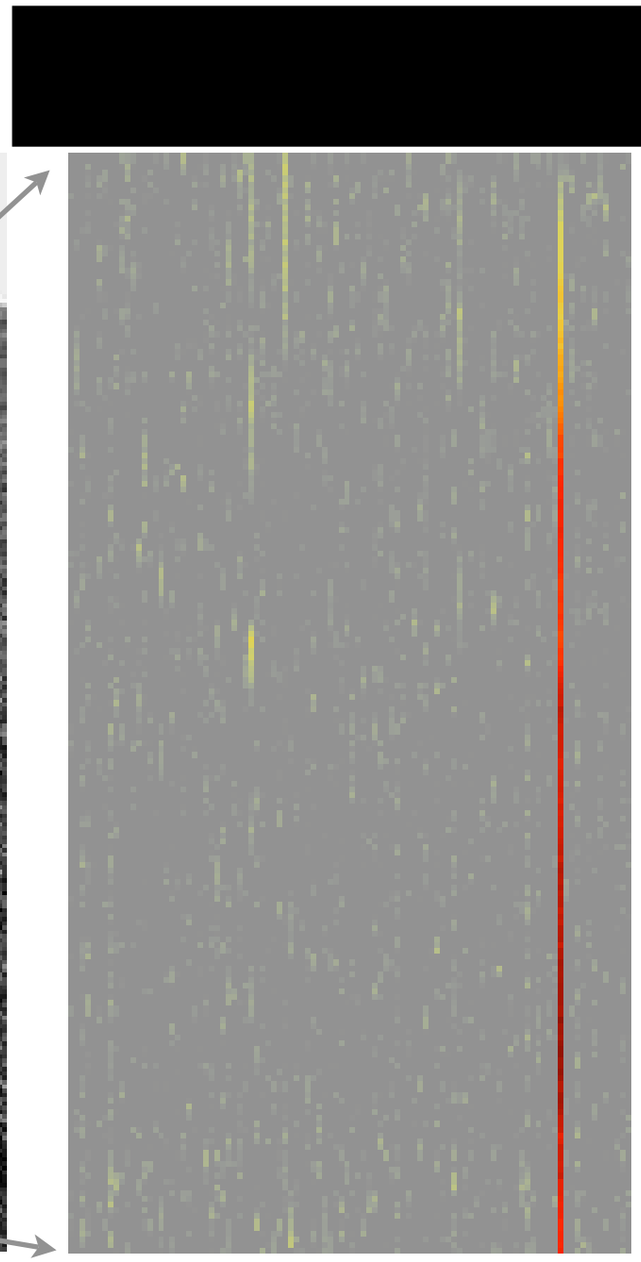


A distributed code for visual surfaces

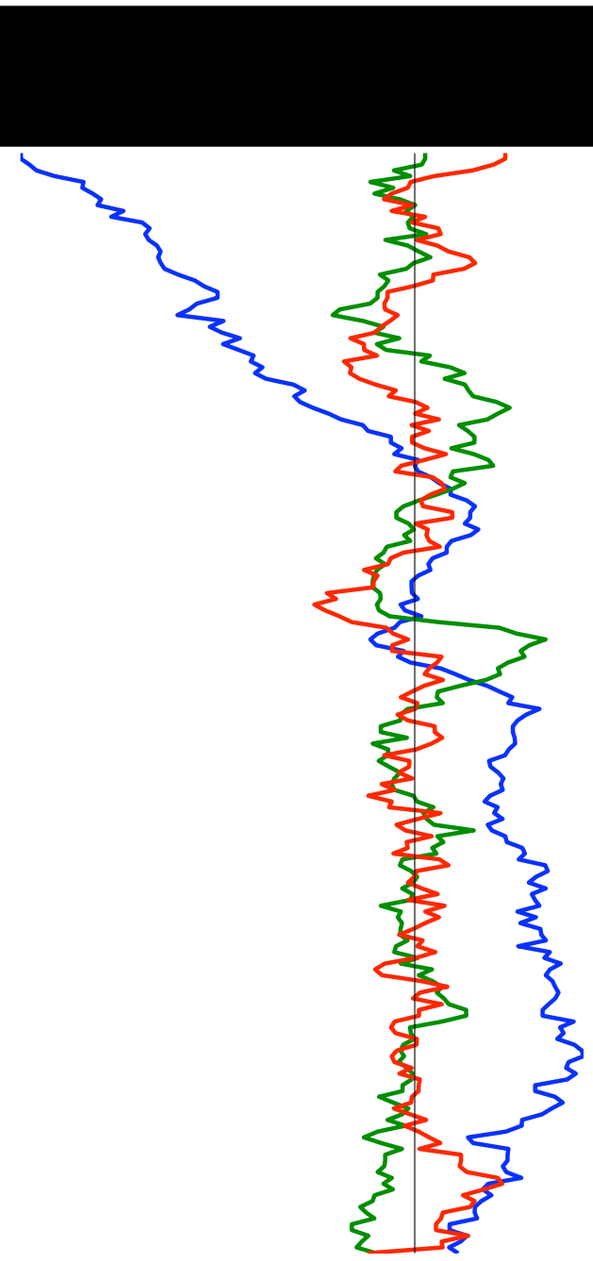




Smooth changes in representation for texture gradients

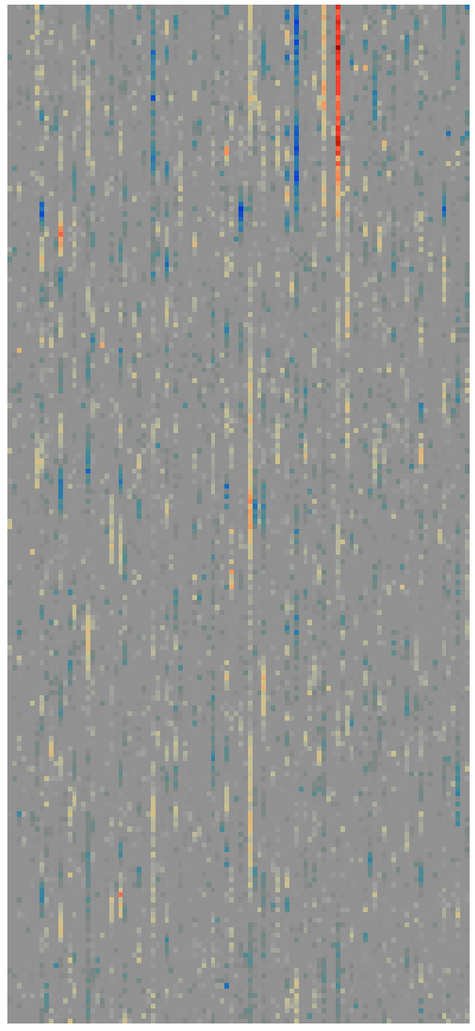


higher-level output

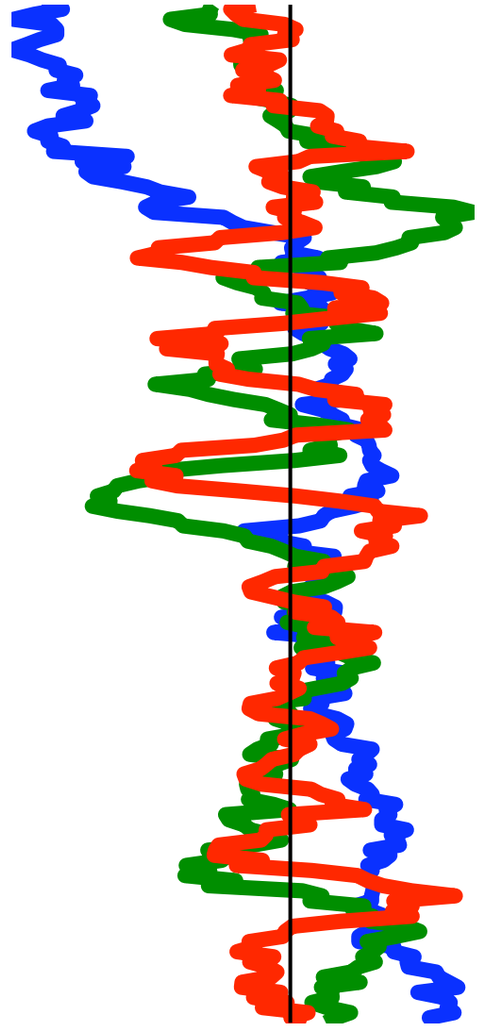


First 3 PCs

Smooth changes in representation for texture gradients



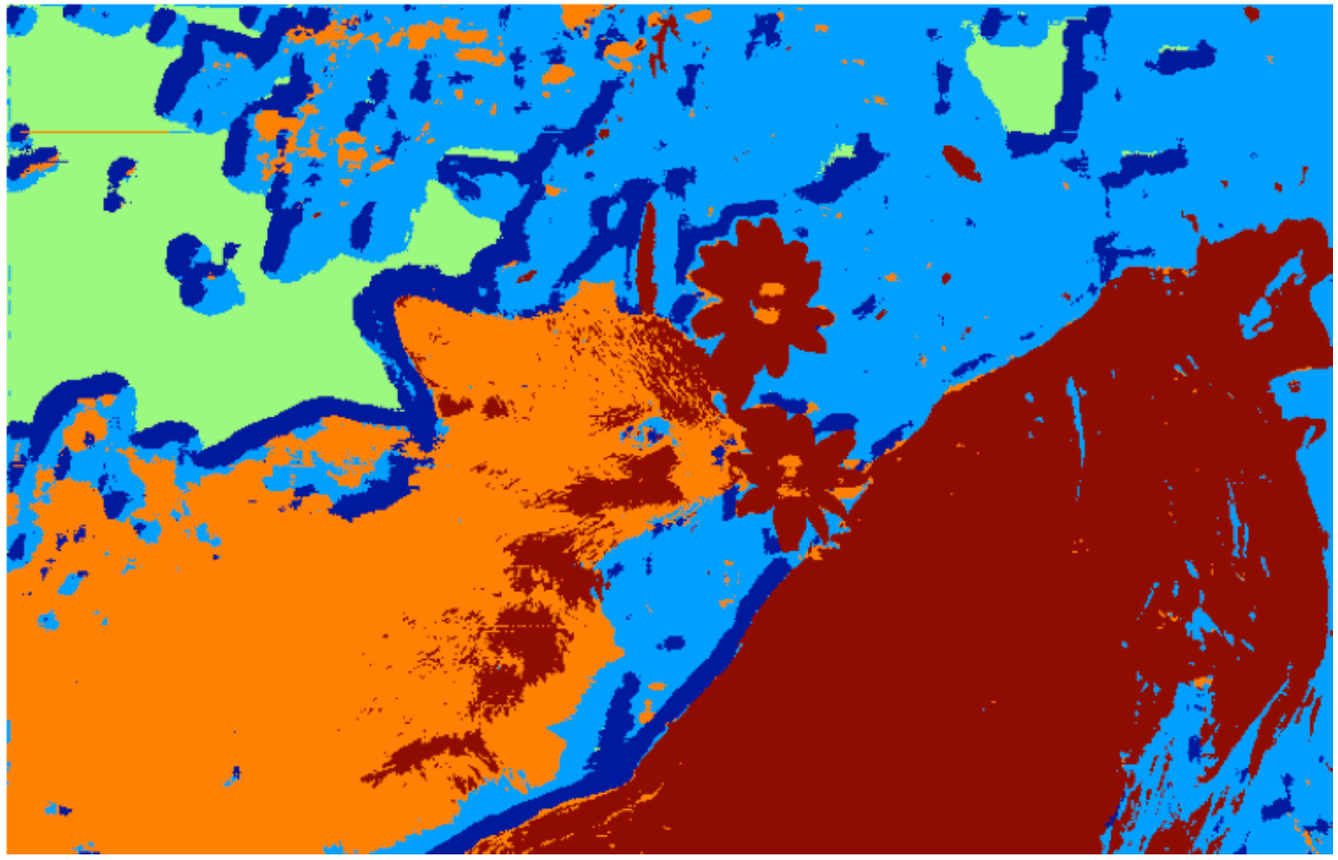
higher-level output



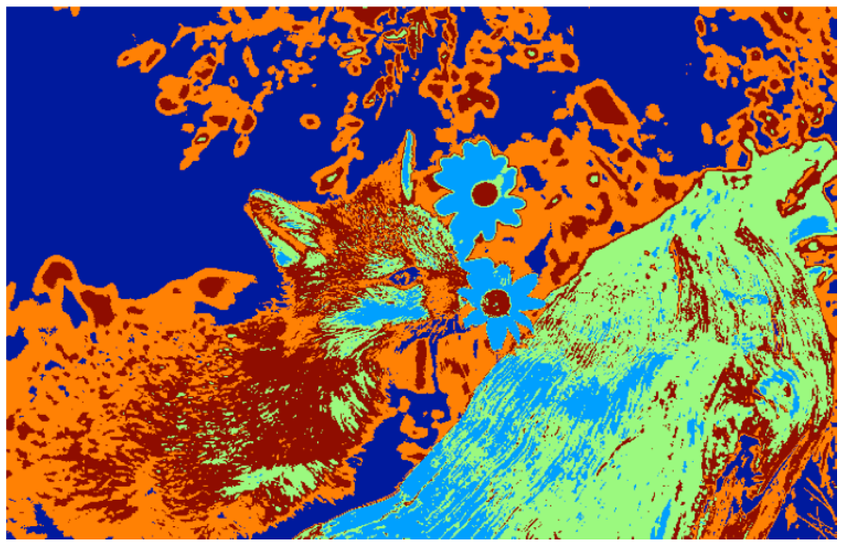
First 3 PCs

Clustering the higher-order representation yields segmentation

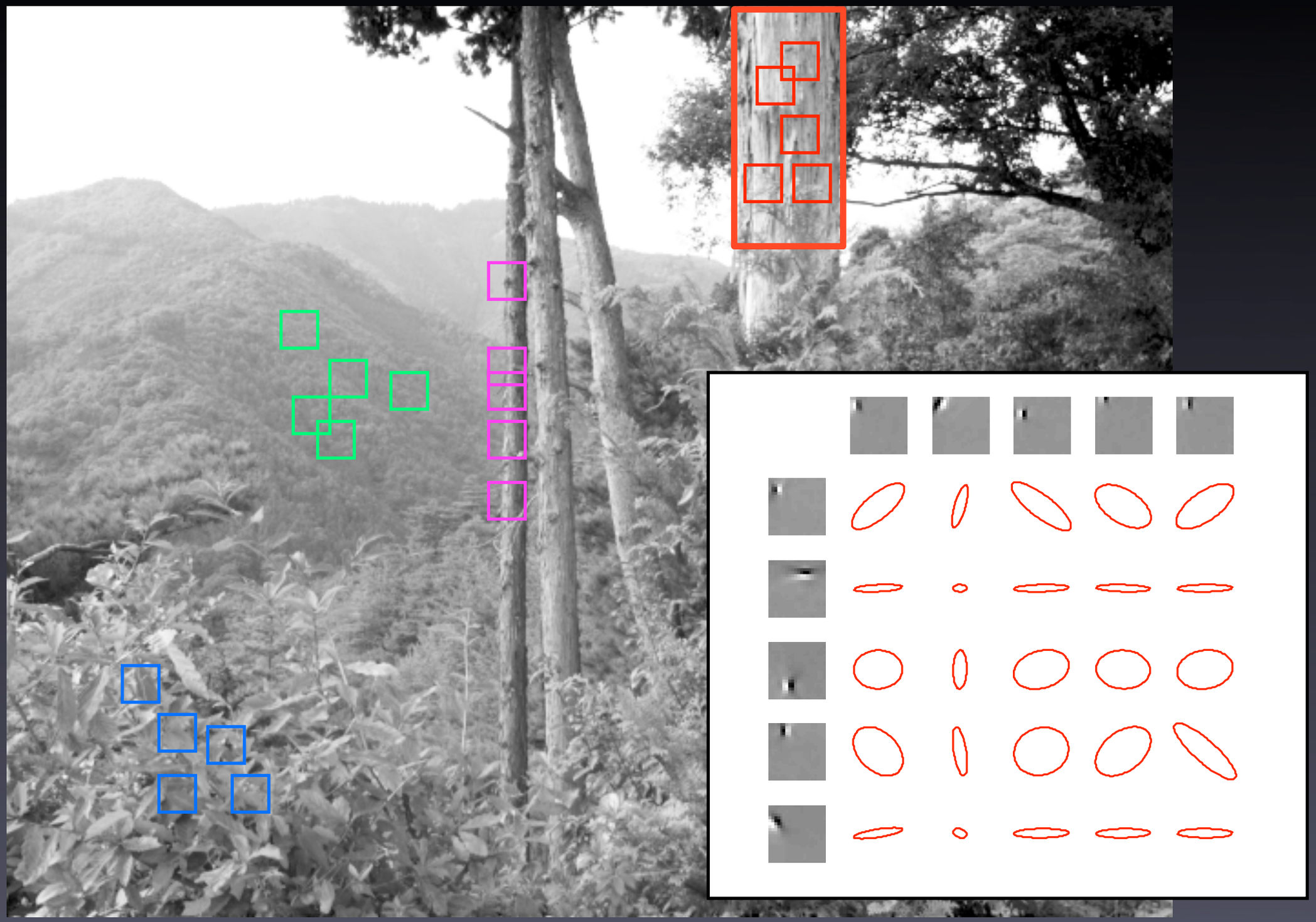
clustering v's



clustering color

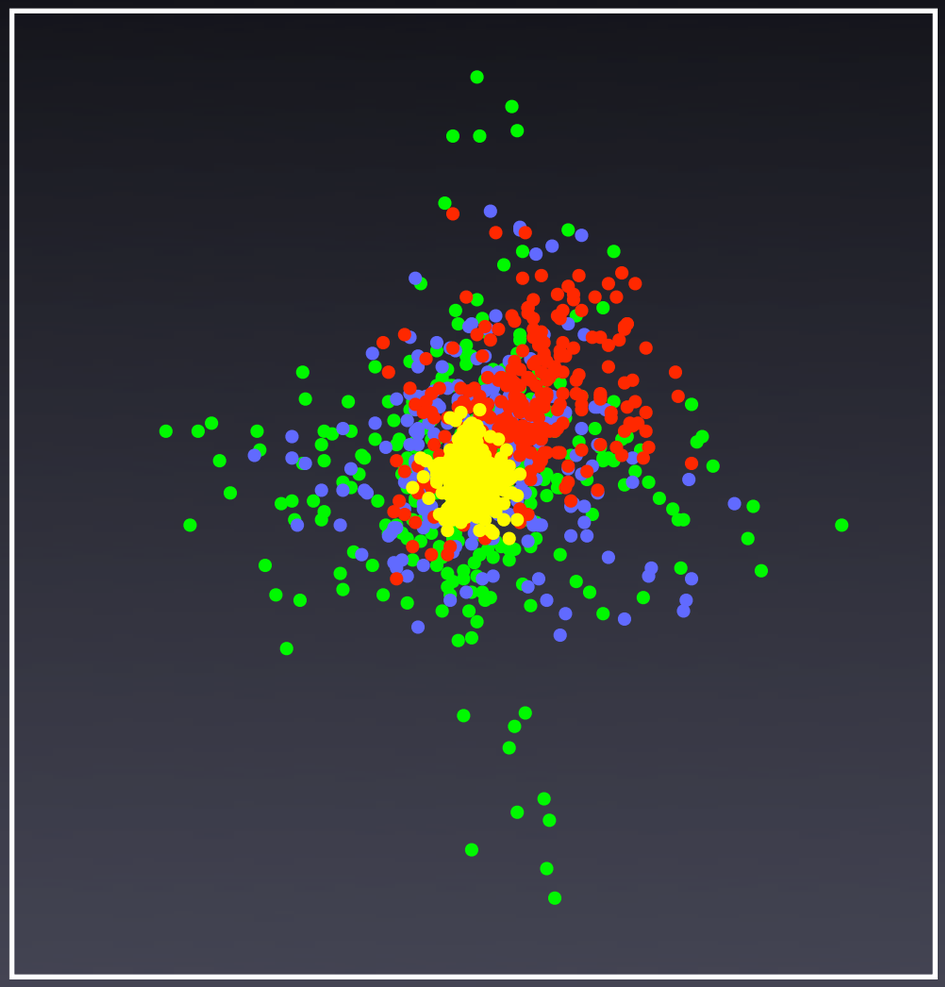


Model can be extended to model local covariance structure



Linear code of image regions

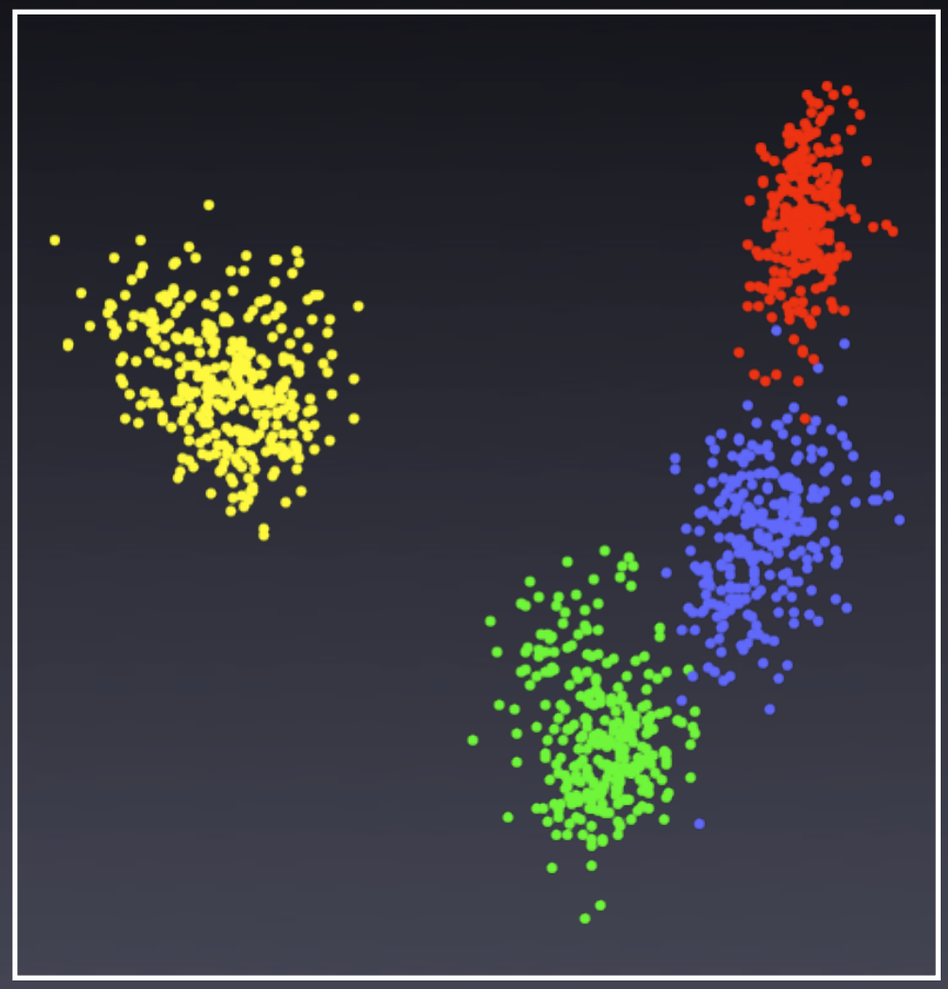
distributions in simple cell space (V1)



2D projection of 400D space

Distributed covariance model of image regions

distributions in higher-order space



2D projection of 150D space

For more, see workshop talk

Theory	• explains data from principles • requires idealization and abstraction
Principle	• code signals accurately and efficiently • adapted to natural sensory environment
Idealization	• neurons encode local image distributions
Methodology	• information theory, natural images • hierarchical, probabilistic models
Prediction	• good generalization in natural images • functional explanation of non-simple cells?

Thank you!