BIOSYSTEMS II: NEUROSCIENCES

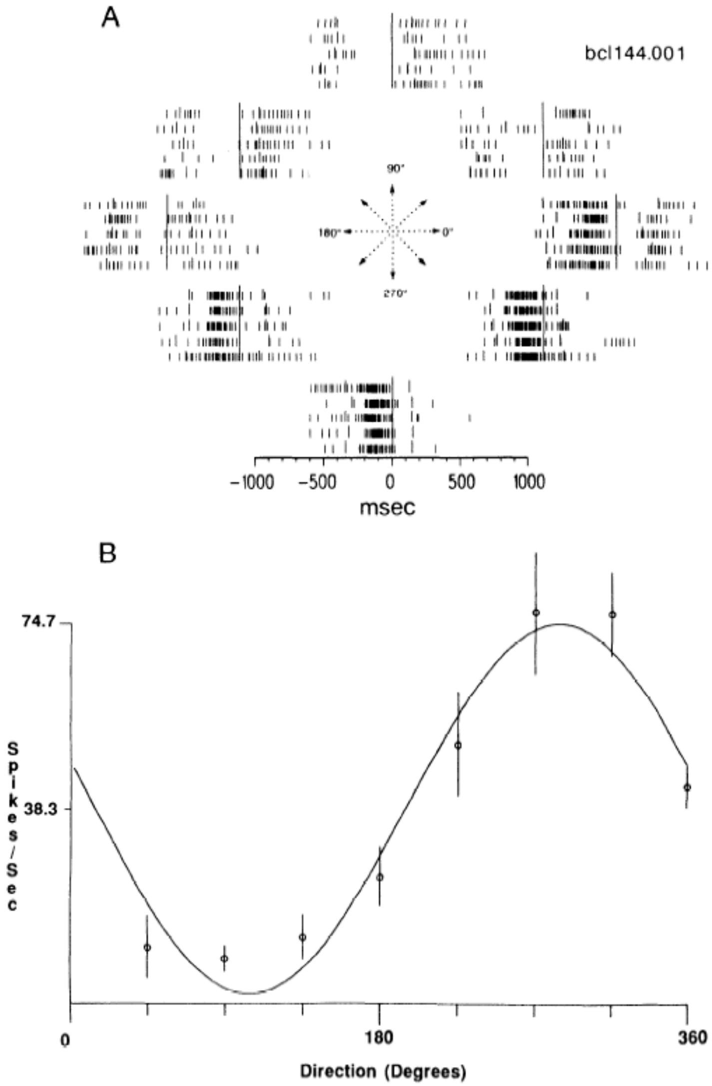
2015 Spring Semester

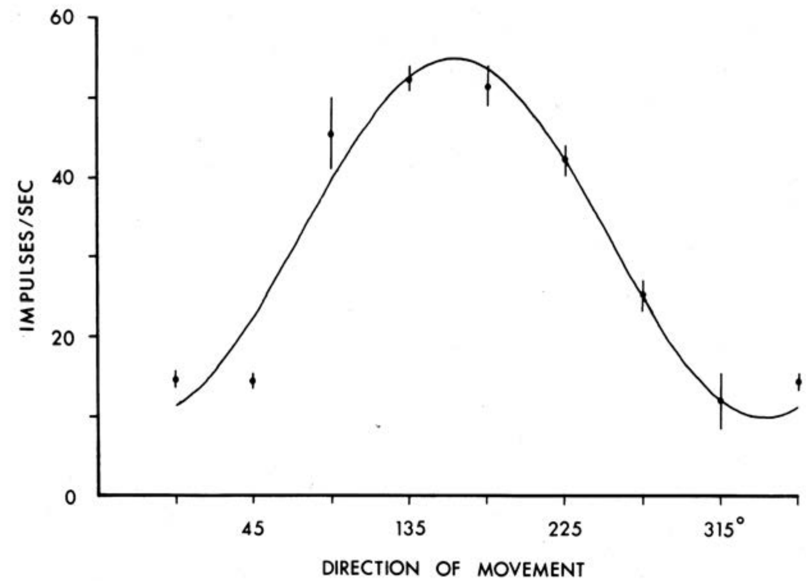
Lecture 35

Kechen Zhang

4/22/2015

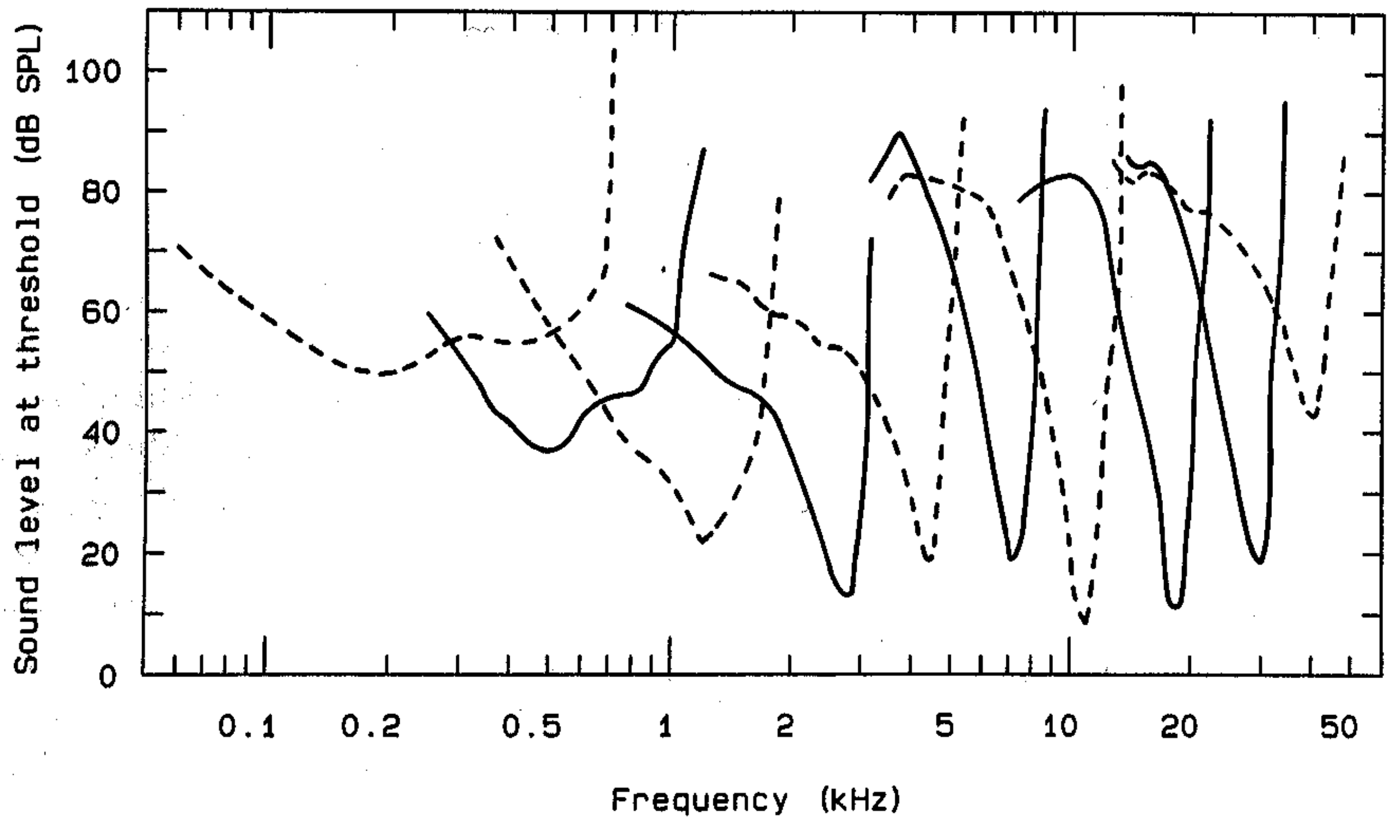
Example of Broad Tuning Curve: Motor Cortical Neuron in Reaching Task



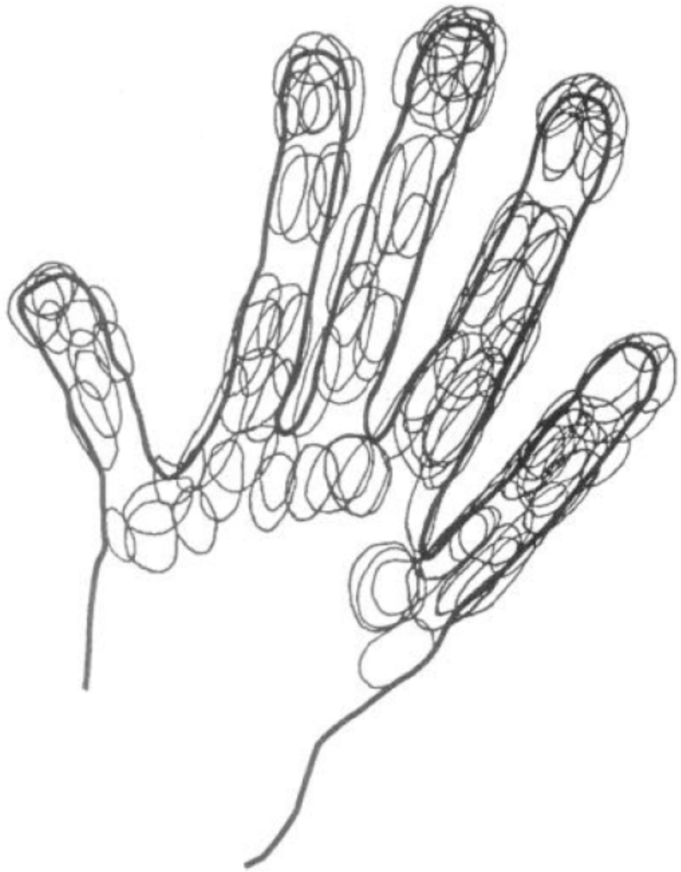


Broad tuning curve and noisy nature of neuronal spikes imply that a neuron does not carry very precise information about an encoded variable (direction of movement)

Example of population coding:
Auditory nerve tuning curves (threshold-frequency curves)

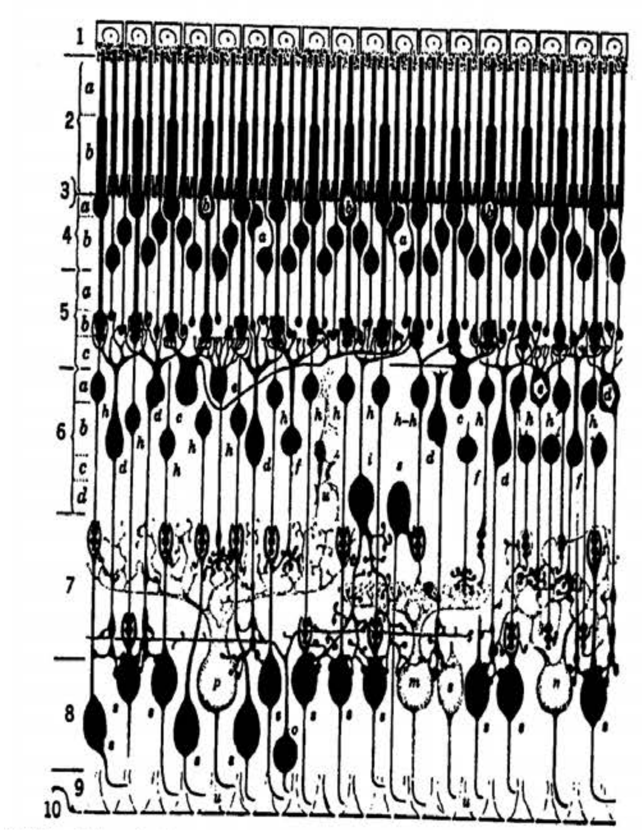


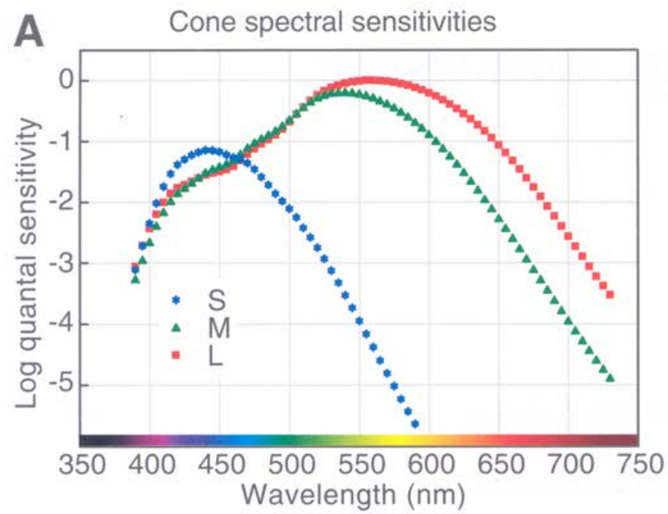
Example of population coding: Receptive fields for somatosensory cortical neurons

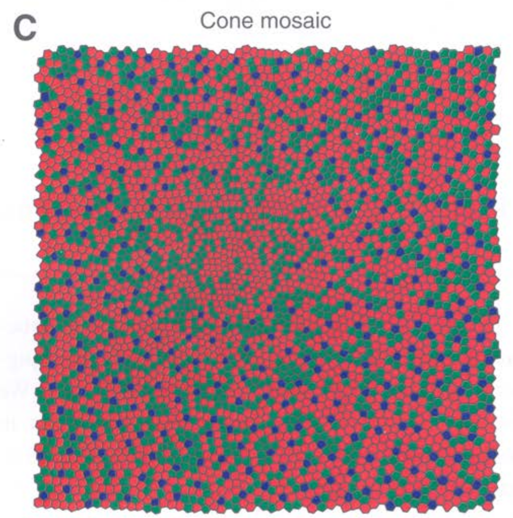


Overlapping receptive fields on the hand of a monkey.

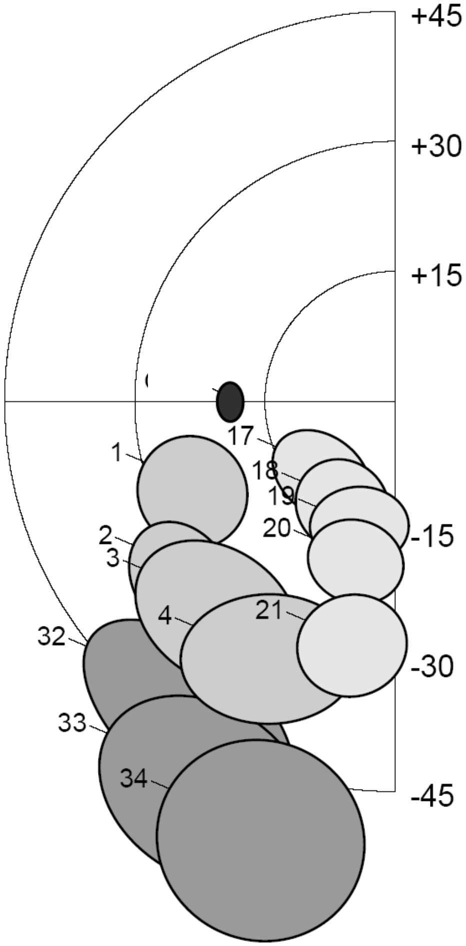
Example of population coding:
Tuning curves of the cones in the retina

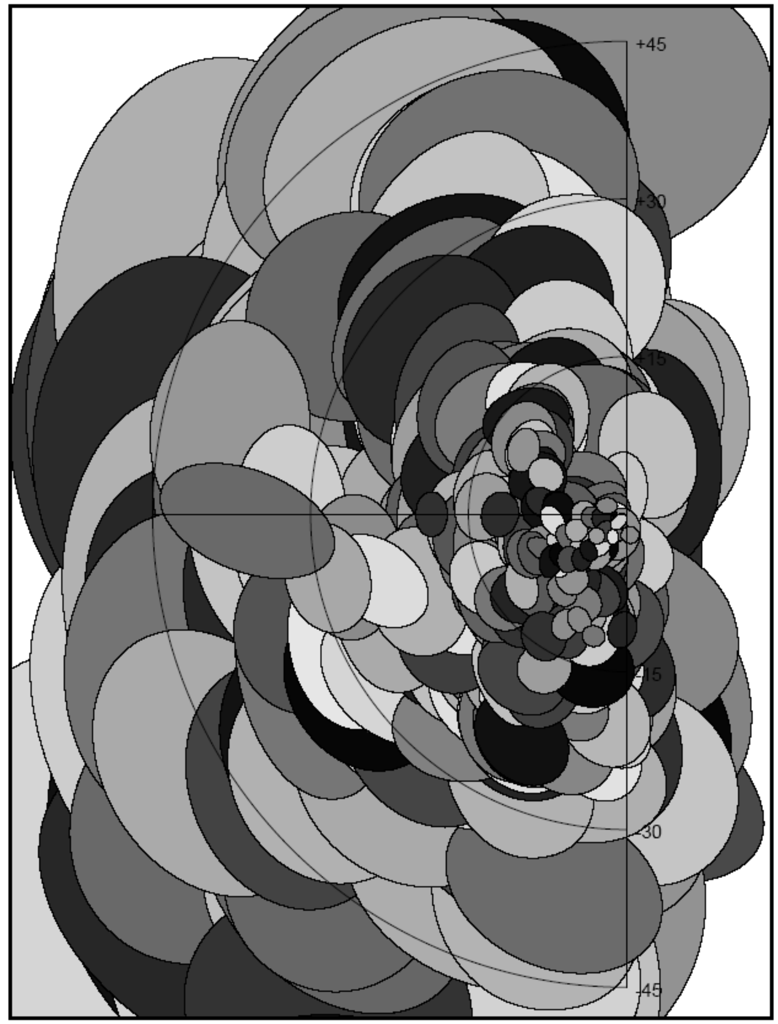






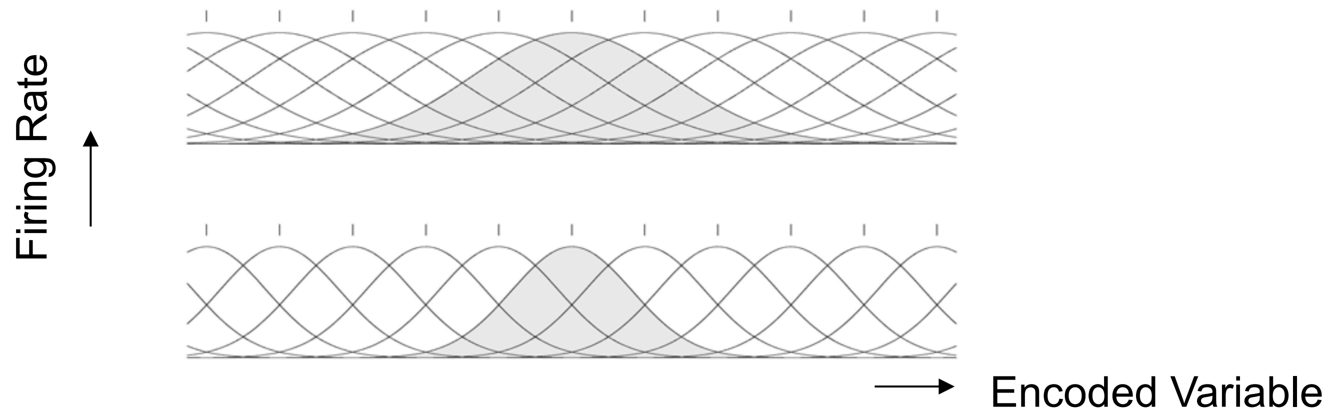
Example of population coding:
Receptive fields of visual cortical neurons





(Sereno)

Population Coding



Two competing factors:

- Broader tuning function always decreases the information carried by a single neuron.
- Broader tuning function also implies that more neurons are activated at the same time so that the total information carried by the whole population increases.

How to read out information from a neuronal population
Example: population vector method

Cosine Tuning Curve

For reaching direction given by vector $\mathbf{r}$ or angle θ , the firing rate above baseline for neuron i is

$$a_i = \mathbf{p}_i \cdot \mathbf{r} = |\mathbf{p}_i||\mathbf{r}|\cos(\theta - \theta_i)$$

where vector $\mathbf{p}_i$ and angle θ_i describe its preferred direction.

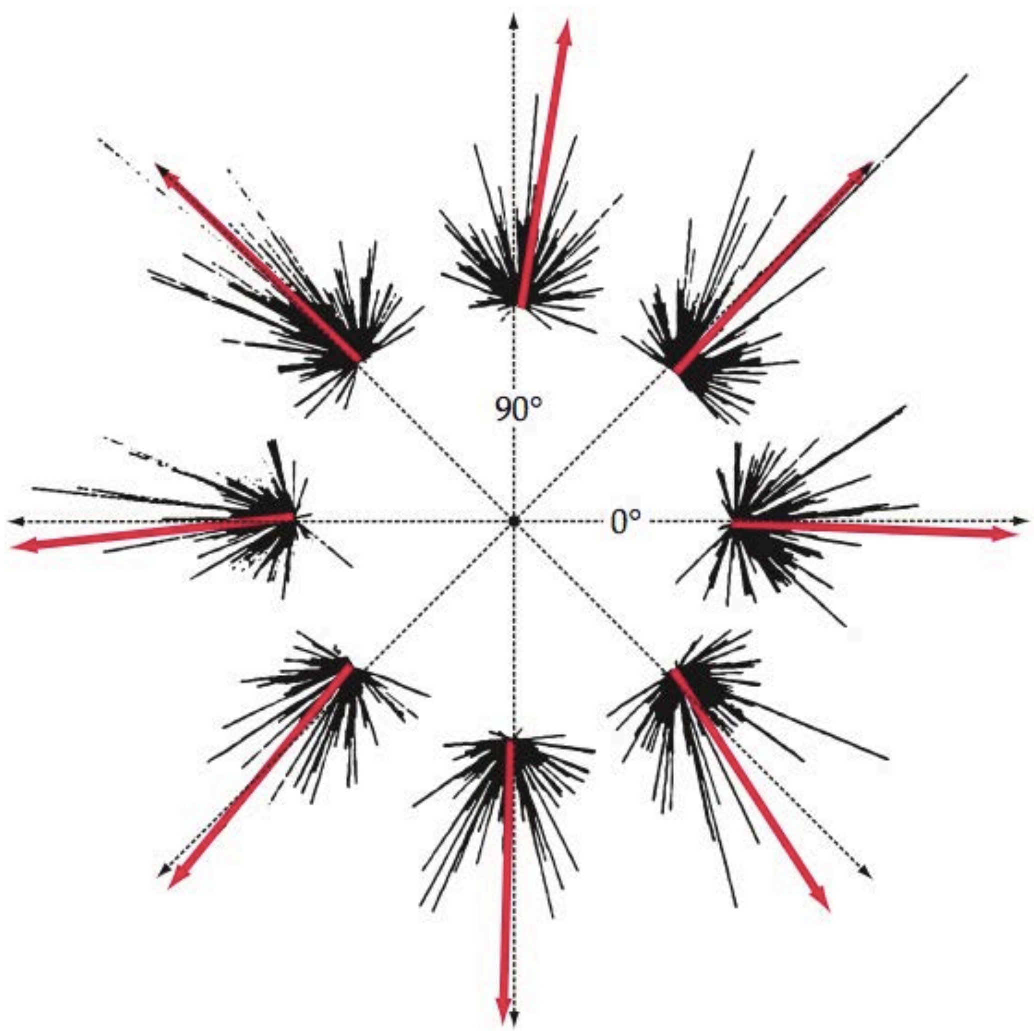
Population Vector

Population vector is defined by

$$\mathbf{v} = \sum_i a_i \mathbf{p}_i$$

It is a linear combination of the preferred directions weighted by the activity of each neuron.

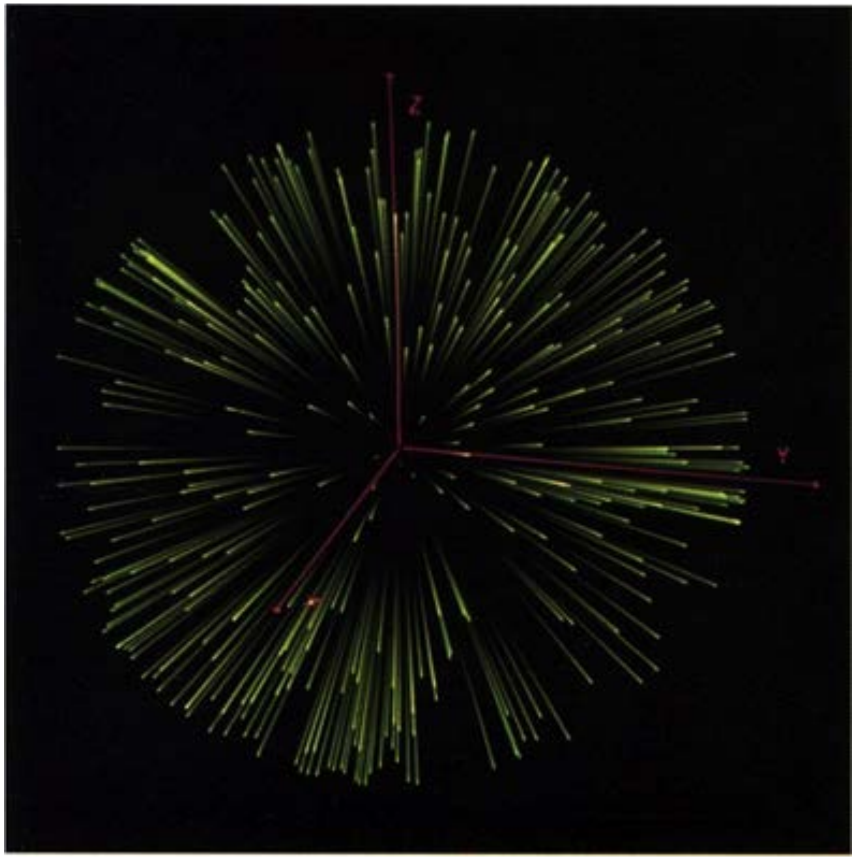
Population Vector Recovers True Reaching Direction



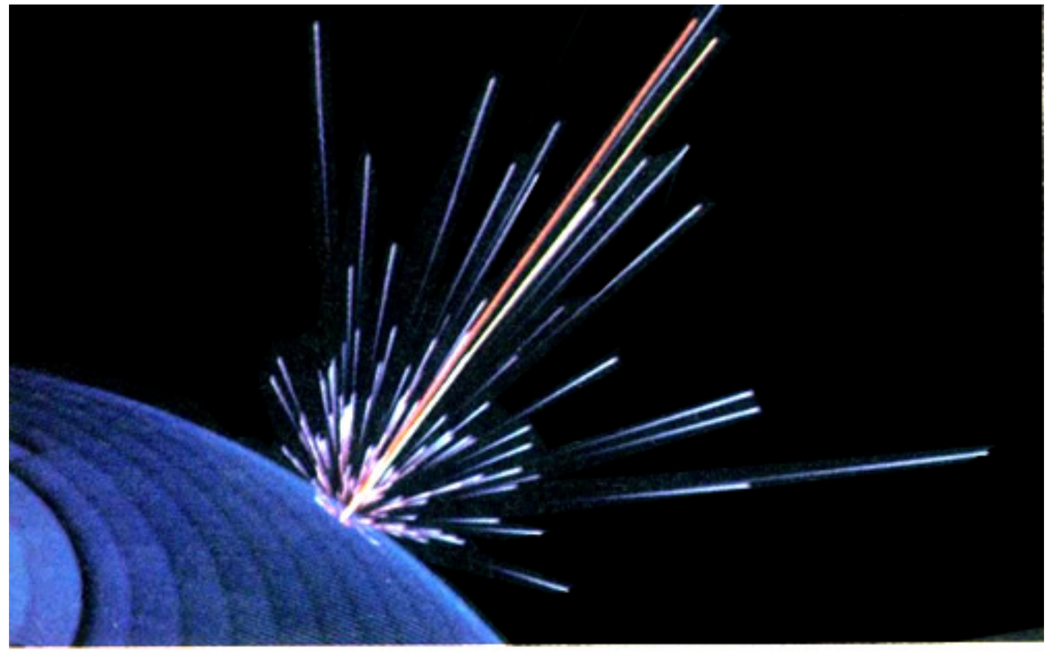
Red arrow: population vector
Black arrow: reaching direction

Each line segment indicates the preferred direction of a neuron weighted by activity above baseline

Population Vector Works in Three Dimensions



The preferred directions of motor cortical neurons are distributed across all directions in 3D



Population vector recovers the true reaching direction in 3D

Schwartz et al

Principle of Population Vector

Population vector always recovers the true reaching direction:

$$\mathbf{v} = \mathbf{r}$$

if and only if

$$\sum_i \mathbf{p}_i \mathbf{p}_i^T = \mathbf{I}$$

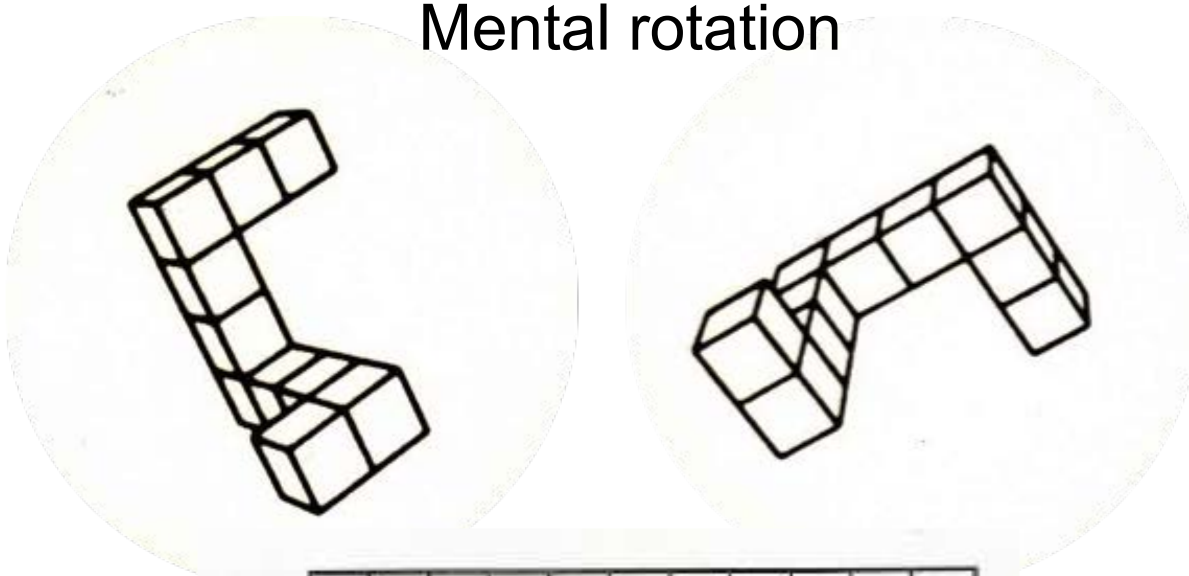
where $\mathbf{I}$ is the identity matrix, and the preferred direction $\mathbf{p}_i$ is taken as a column vector with T indicating transpose.

This is because $\mathbf{v} = \sum_i \mathbf{p}_i a_i = \sum_i \mathbf{p}_i \left(\mathbf{p}_i^T \mathbf{r}\right) = \left(\sum_i \mathbf{p}_i \mathbf{p}_i^T\right)\mathbf{r}$

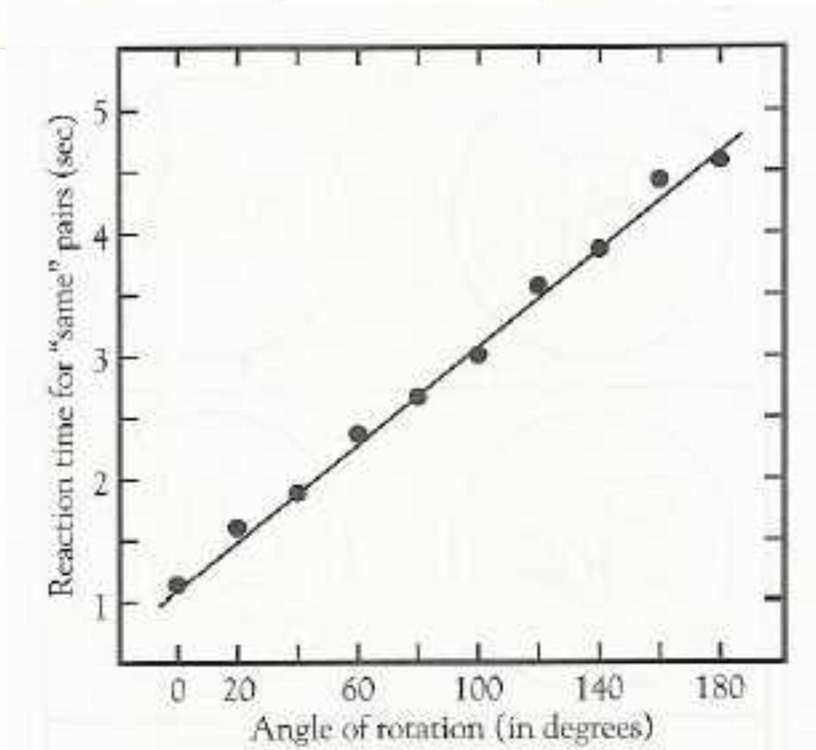
where the 1st step is a definition and the 2nd step is by cosine tuning.

A special case that satisfies the identity matrix condition is a population of uniformly distributed preferred directions, which is roughly true in the motor cortex.

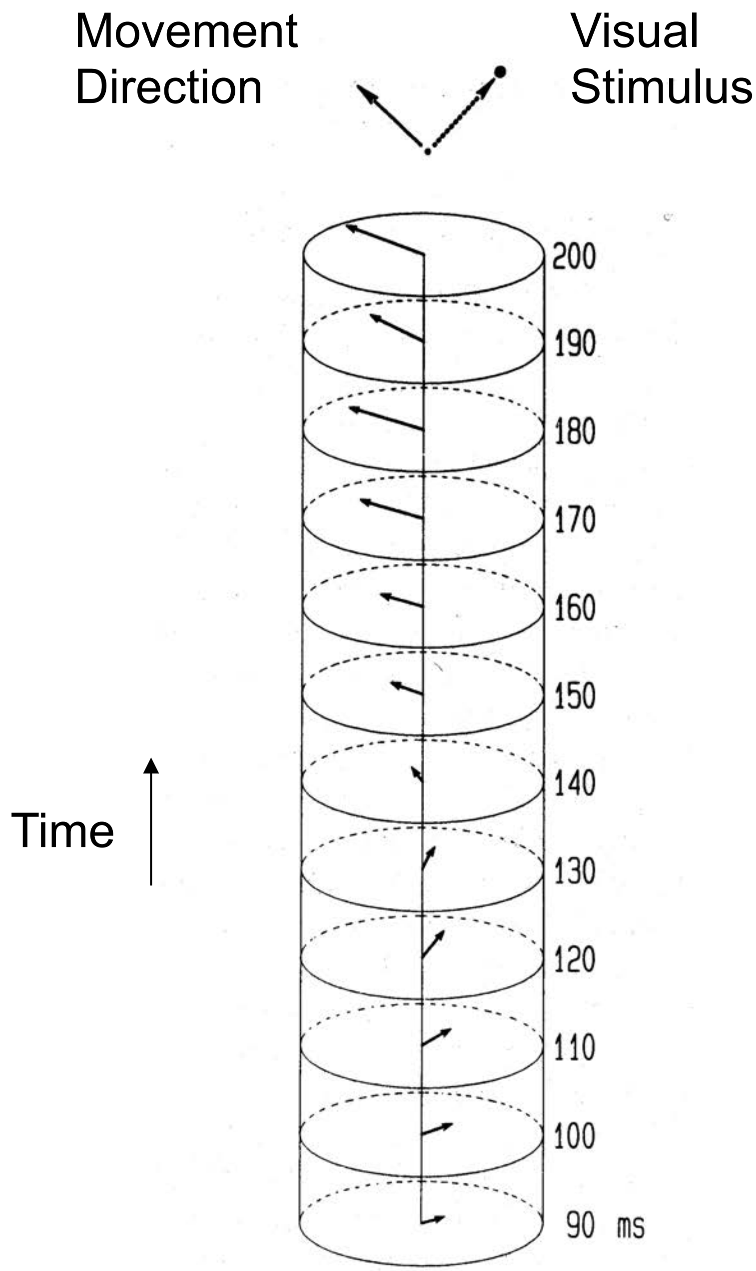
Mental rotation



Same or different?



Reaction time increases linearly with rotation angle



Population vector
in a “mental rotation” task

Video
Hand-position reconstruction by population vector
(A. Schwartz)

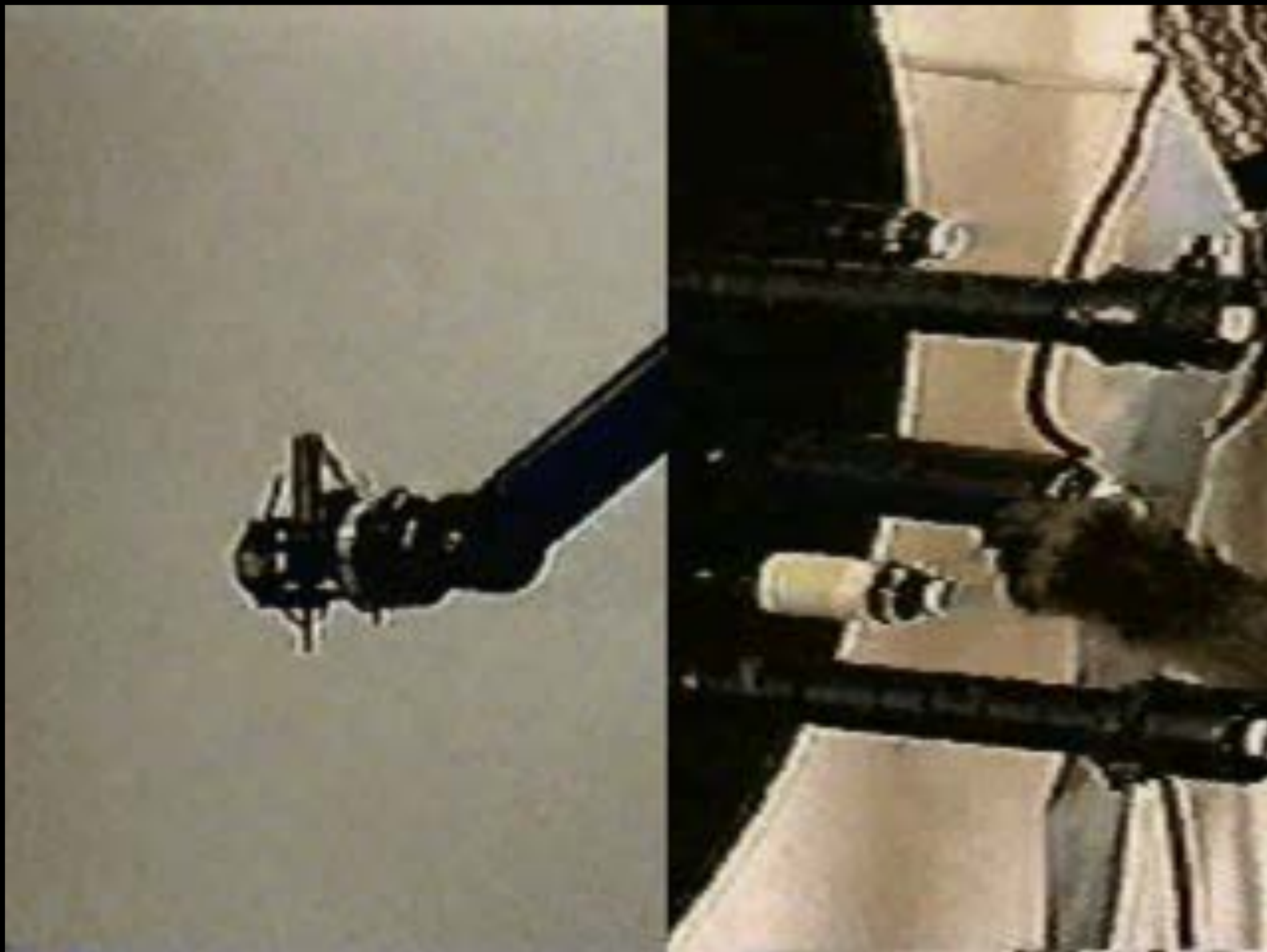
Spikes from a single neuron in a behaving monkey

(Schwartz)

Drawing in circle:
Head-position reconstruction by population vector

Video
Robotic arm controlled by
population vector based on motor
cortical activities
(A. Schwartz)

Robotic Arm | Monkey Arm



(Schwartz)

Tuning functions other than cosine

Population vector is equivalent to finding the peak in the linear combination of cosine functions weighted by activity a_i of each neuron:

$$\sum_i a_i \cos(\theta - \theta_i)$$

More generally, if the tuning curve (or basis function) is a function Φ_i other than the cosine, we write

$$\sum_i a_i \Phi_i(\theta)$$

The counterpart of the population vector algorithm for recovering the true variable is to find the peak of the above function.

Maximum likelihood method for parameter estimation

Maximum likelihood finds parameter that maximizes the probability of the data:

P(Data | Parameter)

Example: least square method for estimating linear model $y = a\,x + b$

Output data $\{y_1, \ldots y_n\}$ contain independent Gaussian noise

$$P(\{y_1,\cdots,y_n\} \mid \{a,b\}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - (ax_i + b))^2}{2\sigma^2} \right)$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma} \right)^n \exp\left(-\frac{\sum_{i=1}^{n} (y_i - (ax_i + b))^2}{2\sigma^2} \right)$$

Maximizing the probability of data (likelihood) is equivalent to minimizing the square error:

$$E = \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

Probabilistic framework: Maximum-likelihood

A more general method is the maximum-likelihood method based on probability theory.

The response of neuron i is stochastic when the same stimulus is repeated.

Consider a Gaussian noise model:

$$p(a_i \mid \theta) = A \exp\left(-\left| a_i - \Phi_i(\theta) \right|^2 \Big/ 2\sigma^2 \right)$$

Assuming the activities of different neurons are statistically independent, we have

$$p(a_1, a_2, \cdots, a_N \mid \theta) = \prod_{i=1}^{N} p(a_i \mid \theta) = \prod_{i=1}^{N} A \exp\left(-\left| a_i - \Phi_i(\theta) \right|^2 \Big/ 2\sigma^2 \right)$$

The most likely direction θ is the one that maximizes this probability (maximum-likelihood).

This is equivalent to maximizing $-\sum_{i=1}^{N} \left| a_i - \Phi_i(\theta) \right|^2 = 2\sum_{i=1}^{N} a_i \Phi_i(\theta) - \sum_{i=1}^{N} a_i^2 - \sum_{i=1}^{N} \Phi_i(\theta)^2$

which in turn is equivalent to maximizing $\sum_{i=1}^{N} a_i \Phi_i(\theta)$ when the last term is a constant

independent of θ (approximately true for uniform distributions of tuning curves).

Bayesian method vs maximum likelihood

Consider joint probability of neural response r and stimulus s (or any variable):

$$P(r,s) = P(r \mid s)P(s) = P(s \mid r)P(r)$$

Bayes rule for inferring stimulus from the response:

$$P(s \mid r) = \frac{P(r \mid s)P(s)}{P(r)}$$

where $P(s)$ is prior distribution of stimulus, $P(s \mid r)$ is the posterior distribution of stimulus given the stimulus, and $P(r \mid s)$ describes the neural response distribution.

How to "decode" the stimulus from neural response r?

Most likely stimulus = $\arg\max_s \; P(s \mid r) = \arg\max_s \; P(r \mid s)P(s)$

Comparison with maximum-likelihood:

Most likely stimulus = $\arg\max_s \; P(r \mid s)$

This is equivalent to the Bayesian method if $P(s) = \text{const}$ (all stimuli are equally likely)

Videos

Position decoding from place cell population