Review of last lecture: linear coding methods

Goal is to describe the data to desired precision.
Code signal by linear superposition of basis functions:

\[ x = \vec{a}_1 s_1 + \vec{a}_2 s_2 + \cdots + \vec{a}_L s_L + \vec{\epsilon} \]

\[ = \mathbf{A}s + \mathbf{\epsilon} \]

- \( x(t) \) is represented by a vector \( x \)
- \( \vec{a}_i \) are the basis vectors
- \( \mathbf{A} \) is the basis (could be Fourier, wavelet, etc.)
- \( s_i \) are the coefficients

Can solve for \( \hat{s} \) in the no noise case

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

**Is the peripheral auditory system a linear system?**
Need a functional description of the peripheral auditory system
The outer ear behaves like a series of resonance cavities

![Graph showing pressure gain (dB) vs frequency (Hz) for the total, ear canal & drum, and concha components.](from Yost, 2000)
The middle ear

from Warren, 1999
The bones of the middle ear (ossicles)

Middle ear function and properties:

- ossicles are adapted to (apparently) to minimize non-linearities
- have a balanced center of gravity
- gain of middle ear is frequency dependent (maximum of 30 dB, 700-800 Hz)
- also has muscles that can attenuate loud sounds (by as much as 30 dB for low frequency sounds)
  - trigger is sound (> 80 dB) or chewing
  - can only protect against slow onset, low frequency sounds
  - takes 10 msecs (not fast enough for gunshots)
The inner ear

from Yost, 2000
Traveling waves at different frequencies

- sound waves are roughly converted into frequency by traveling waves
- maximal vibration occurs at different locations depending on frequency (low frequencies travel farther)
- basilar membrane is highly specialized:
  - thick and stiff at basal end
  - flexible at apical end
  - traveling waves slow down by a factor of 100, velocity is an exponential function of distance.

from Yost, 2000
Shape of basilar membrane and its frequency map

- WIDTH AT APEX 0.5 mm
- WIDTH AT BASE 0.04 mm
- AVERAGE WIDTH 0.21 mm BASAL TURN
  - 0.34 mm MIDDLE TURN
  - 0.36 mm APICAL TURN
- LENGTH 32 mm
Mechanical properties of BM determine hearing range

RIGHT WHALE *Eubalaena glacialis* - Type M
2.5 x 1400 μ
apex

basilar membrane

inner osseous lamina

2 mm

25 μ

25 μ

25 x 30 μ

18 mm

5 x 300 μ

26 mm

5 x 380 μ

41 mm

25 x 30 μ

10 mm

75 Hz - 150 kHz
dyn. range: 195 dB

HARBOR PORPOISE
*Phocoena phocoena* - Type I

BOTTLENOSED DOLPHIN
*Tursiops truncatus* - Type II

10 Hz - 22 kHz
(estimated)

1 kHz - 150 kHz
dyn. range: 140 dB
Traveling Waves

Animation from Geisler

from Yost, 2000
Basilar Membrane Impulse Response

from Moore, 1997
The structure of the cochlea

cross-section

from Pickles, 1988
Sensing basilar membrane motion with hair cells

from Yost, 2000
Hair-cell transduction

- basilar membrane motion results in stereocilia vibration
- this results in current flow as an influx of $K^+$ ions
- If influx cross threshold, this results in an action potential at the auditory nerve

from Gold and Morgan, 2000

Stereocilia motion

Ionic channels in stereocilia open when basilar membrane vibrates

Potassium ions enter cell, causing depolarization (voltage increase)

Depolarization opens calcium channels near base of cell

Influx of calcium releases (unknown) neurotransmitter

Neurotransmitter opens channels in afferent nerve fiber

Sodium flows into fiber, depolarizing fiber

Depolarization of fiber causes action potential (spike)
Phase locking of auditory nerve spikes

from Yost, 2000
Spiking probability follows stimulus amplitude

Unit 67-135-7

Tone 1: 798 cps
Tone 2: 1064 cps
R = 3.4

Tone 1: 60 dB SPL
Tone 2: 80 dB SPL
N = 2475

Tone 1: 70 dB SPL
Tone 2: 90 dB SPL
N = 2783

Tone 1: 80 dB SPL
Tone 2: 100 dB SPL
N = 2742

A1, Tone 2: 75 dB
N = 2668

A2, Tone 2: 70 dB
N = 2621

A3, Tone 2: 70 dB
N = 2621

B1, Tone 2: 85 dB
N = 2776

B2, Tone 2: 80 dB
N = 2758

B3, Tone 2: 90 dB
N = 2620

C1, Tone 2: 95 dB
N = 2548
Phase locking breaks down for higher frequencies

Also note adaptation to tone bursts

from Moore, 1997
Auditory nerve fiber responses: frequency tuning

threshold sensitivity curves for different auditory nerves

Sound level at threshold (dB SPL)

Frequency (kHz)

0 100 90 80 70 60 50 40 30 20 10 0

0.1 0.2 0.5 1 2 5 10 20 50
The real system is much more complex than linear model from Warren, 1999

How do we describe what it does?
A linear characterization of auditory nerves

Spike-triggered averaging estimates auditory nerve impulse response functions: “revcor” filters

from deBoer and Kuyper, 1968
Auditory nerve recor filters

deBoer and deJongh, 1978

Carney and Yin, 1988

compare to BM impulse response
Predictions of the linear model

from deBoer and deJongh, 1978
Information theory arose from the problem of speech coding

speech \rightarrow telephone \rightarrow telephone wires \rightarrow telephone \rightarrow ear

sound waveform \rightarrow filterbank \rightarrow static non-linearities \rightarrow stochastic spiking \rightarrow brain
A simple model of auditory coding

Auditory nerve filters can be estimated using reverse correlation (spike-triggered averaging)

sound waveform → filterbank → static non-linearities → stochastic spiking → population spike code

deBoer and deJongh, 1978  Carney and Yin, 1988
Models are data driven

**data:** revcor filter

**model:** gammatone

\[
g(t) = at^{n-1}e^{-bt}\cos(2\pi ft + \phi)
\]

“gammatone” function
Models are data driven

**data**: revcor filter with fitted model

residual error

\[ g(t) = at^{n-1}e^{-bt}\cos(2\pi ft + \phi) \]

“gammatone” function
A theoretical approach

Theoretical questions:
• Why gammatones?
• Why spikes?
• How is sound coded by the spike population?

*How do we develop a theory?*
Efficient coding theory

- Barlow, 1961; Attneave, 1954
  - main goal of sensory coding is to code signals \textit{efficiently}
  - sensory codes are adapted to the sensory environment
  - each code feature should have minimal \textit{redundancy}
  - each “feature” should describe \textit{independent} information

- caveats:
  - applies to \textit{behaviorally relevant} information
  - not all redundancy is bad, e.g. when compensating for noise
Limitations of this theoretical model

- filter bank model is linear
- code is optimal only within a block, not for whole signal
- offers no explanation of phase locking and spikes
- representation depends on the relative alignment of the signal and block
Block coding does not yield time-relative codes

from Smith and Lewicki, 2005
Block coding does not yield time-relative codes

from Smith and Lewicki, 2005
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)

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

- Two important theoretical abstractions for “spikes”:
  - not binary, each has an analog value
  - not probabilistic
- Each spike encodes the precise time and magnitude of a particular kernel
- Population forms a non-redundant signal representation
- Can convert to a population of stochastic, binary spikes

from Smith and Lewicki, 2005
The spikegram

from Smith and Lewicki, 2005
Comparing a spike code to a spectrogram

How do we compute the spikes?

from Smith and Lewicki, 2005
Comparing a spike code to a spectrogram

How do we compute the spikes?

- There are many possible algorithms, varying degrees of biological plausibility
- Here, we use a variation of Matching Pursuit (Mallat and Zhang, 1993)
  - yields near optimal spike representation, but not biologically plausible
  - assume there exists a biol. plausible algorithm that achieves the same end
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
What are the optimal shapes for the kernels?

\[ x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_{m,i} \phi_m(t - \tau_{m,i}) + \epsilon(t) \]
Optimizing the probabilistic model

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

\[ p(x|\Phi) = \int p(x|\Phi, s, \tau)p(s)p(\tau)dsd\tau \]
\[ \approx p(x|\Phi, \hat{s}, \hat{\tau})p(\hat{s})p(\hat{\tau}) \]
\[ \epsilon(t) \sim \mathcal{N}(0, \sigma_\epsilon) \]

Learning (Olshausen, 2002):

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

Also extend algorithm to 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)

\[ x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_{m,i} \phi_m(t - \tau_{m,i}) + \epsilon(t) \]
Coding efficiency curves

- Spike Code: adapted
- Spike Code: gammatone
- Block Code: wavelet
- Block Code: Fourier

3x more efficient

from Smith and Lewicki, 2005
Coding efficiency curves

from Smith and Lewicki, 2005
Coding efficiency curves

from Smith and Lewicki, 2005
Using efficient coding theory to make theoretical predictions

Environment of Natural Sounds
(speech, env. sounds, vocalizations)

optimize kernel
 coding efficiency

optimal kernels:
• properties
• coding efficiency

physiological data:
• auditory nerve filter shapes
• population trends

? only compare to the data after optimizing
we do not fit the data
## Natural sounds

<table>
<thead>
<tr>
<th></th>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fox</td>
<td>walking on leaves</td>
</tr>
<tr>
<td></td>
<td>squirrel</td>
<td>cracking branches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rustling leaves</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stream by waterfall</td>
</tr>
</tbody>
</table>
### Natural sounds

<table>
<thead>
<tr>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>fox</td>
<td>walking on leaves</td>
</tr>
<tr>
<td></td>
<td>rustling leaves</td>
</tr>
<tr>
<td>squirrel</td>
<td>cracking branches</td>
</tr>
<tr>
<td></td>
<td>stream by waterfall</td>
</tr>
</tbody>
</table>
### Natural sounds

<table>
<thead>
<tr>
<th></th>
<th>vocalizations</th>
<th>environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>fox</td>
<td></td>
<td>walking on leaves</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rustling leaves</td>
</tr>
<tr>
<td>squirrel</td>
<td>cracking branches</td>
<td>stream by waterfall</td>
</tr>
</tbody>
</table>
## Natural sounds

<table>
<thead>
<tr>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>fox</td>
<td>rustling leaves</td>
</tr>
<tr>
<td>squirrel</td>
<td>cracking branches</td>
</tr>
</tbody>
</table>

- **Walking on leaves**
- Stream by waterfall
### Natural sounds

<table>
<thead>
<tr>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>fox</td>
<td>transient</td>
</tr>
<tr>
<td>squirrel</td>
<td>walking on leaves</td>
</tr>
<tr>
<td></td>
<td>cracking branches</td>
</tr>
<tr>
<td></td>
<td>rustling leaves</td>
</tr>
<tr>
<td></td>
<td>stream by waterfall</td>
</tr>
</tbody>
</table>
## Natural sounds

<table>
<thead>
<tr>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>fox</em></td>
<td>walking on leaves</td>
</tr>
<tr>
<td><em>squirrel</em></td>
<td>cracking branches</td>
</tr>
<tr>
<td><em>rustling leaves</em></td>
<td>stream by waterfall</td>
</tr>
</tbody>
</table>
## Natural sounds

<table>
<thead>
<tr>
<th>Vocalizations</th>
<th>Environmental sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>fox</strong></td>
<td><strong>walking on leaves</strong></td>
</tr>
<tr>
<td><strong>squirrel</strong></td>
<td><strong>cracking branches</strong></td>
</tr>
<tr>
<td></td>
<td><strong>rustling leaves</strong></td>
</tr>
<tr>
<td></td>
<td><strong>stream by waterfall</strong></td>
</tr>
</tbody>
</table>
Learned kernels share features of auditory nerve filters

Optimized kernels
scale bar = 1 msec

Auditory nerve filters
from Carney, McDuffy, and Shekhter, 1999
Learned kernels closely match individual auditory nerve filters

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

from Smith and Lewicki, 2005
Learned kernels overlaid on selected auditory nerve filters

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

from Smith and Lewicki, 2005
Spike kernels for natural sound mix matches revcor filters
Optimal kernels for environmental sounds are very short
Spike kernels for vocalizations are much longer and symmetric.
Comparing learned kernels to auditory nerve population

![Graph comparing center frequency to bandwidth with different categories: Revcor, Environ, Vocal, Natural]
Population distribution of kernels for natural sounds

![Graph showing population distribution of kernels for different sound categories. The x-axis represents center frequency (kHz), and the y-axis represents bandwidth (kHz). The graph includes data points for Revcor, Environ, Vocal, and Natural sounds, each represented by different markers.](image-url)
Population distribution of kernels for environmental sounds

![Population distribution of kernels for environmental sounds](image)
Population distribution of kernels for animal vocalizations

![Population distribution of kernels for animal vocalizations](image)
Kernel distributions for different sound ensembles

![Diagram showing kernel distributions for different sound ensembles. The x-axis represents Center Frequency (kHz), ranging from 0.1 to 5. The y-axis represents Bandwidth (kHz), ranging from 0 to 5. Different symbols represent different sound ensembles: Revcor (blue stars), Environ (black circles), Vocal (green triangles), and Natural (red squares).]
Population distribution of kernels for natural sounds

![Graph showing the population distribution of kernels for natural sounds. The x-axis represents the center frequency in kHz, ranging from 0.1 to 5.0. The y-axis represents the bandwidth in kHz, ranging from 0.1 to 5.0. The graph includes data points for different categories: Revcor, Environ, Vocal, and Natural.](image-url)
Population distribution of kernels for speech (TIMIT)

![Graph showing population distribution of kernels for speech (TIMIT)](image_url)
Speech matches composition of natural sounds

Best mix for predicting auditory coding: 1.0 : 0.8 : 1.2
How is this achieving an efficient, time-relative code?

Time-relative coding of glottal pulses

from Smith and Lewicki, 2005
Coding of a speech consonant

a

Input
Reconstruction
Residual

b

Kernel CF (Hz)

Kernel CF (Hz)

Kernel CF (Hz)

38
39
40
3400
4000
4800

38
39
40

1000
2000
3000
4000
5000

10
20
30
40
50
60ms

1000
2000
3000
4000
5000
83

from Smith and Lewicki, 2005