Topics

- Introduction
  - Spike trains and sensory coding
  - Example data: Multi-electrode recordings from the retina
  - Basic spike train analysis
- Bayesian stimulus reconstruction
  - Idea, algorithm, example results
- Metric-based clustering
  - Idea, algorithm, example results
Coding

- Transmission of a message by a sequence of defined symbols

E.g. Morse code:
- Message: Sequences of letters
- 3 symbols: dot (·), dash (—), space ( )

L    I    E    B     E
· — · ·   · ·   ·  — · · ·  ·

LOVE

L    I    E    B     E
· — · ·   · ·   ·  — · · ·  ·

LOVE
Coding

- Transmission of a message by a sequence of defined symbols

- E.g. Morse code:
  - Message: Sequences of letters
  - 3 symbols: dot (·), dash (—), space ( )

LOVE

THIEVES
Neurons communicate with sequences of action potentials ("spike trains")

\[ S(t) = \begin{cases} 
1 & \text{if spike detected,} \\
0 & \text{otherwise} 
\end{cases} \]

Spikes of one neuron look very similar, therefore only the spike times are considered to be relevant.
Retinal Coding

Stimulus:

Responses

dim flash

bright flash

time

time
Retinal Coding

- Neuronal responses depend on stimulus properties
- Considerable variability between stimulus presentations
- What are the symbols?
  - Spike count?
  - Response latency?
  - Response patterns?

Goal: Estimate stimulus properties based on responses
Extracellular multi-electrode recordings (isolated turtle retina)

- Moving dot pattern:
  - 500 ms constant velocity, then transition
  - 9 velocities in 2 directions
  - 648 presentations
  - recording of ~100 RGCs
Retinal Coding
- Experiments -

Extracellular multi-electrode recordings (isolated turtle retina)

Moving dot pattern: 500 ms constant velocity, then transition to 9 velocities in 2 directions, 648 presentations recording of ~100 RGCs

<table>
<thead>
<tr>
<th>Velocity (mm/s)</th>
<th>Spike rate (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.25</td>
<td>0</td>
</tr>
<tr>
<td>2.5</td>
<td>0</td>
</tr>
<tr>
<td>-1.25</td>
<td>0</td>
</tr>
<tr>
<td>-2.5</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \theta \] stimulus

\[ R_{\text{pop}}(t) \] population response

\[ S_i(t) \] cell responses
Basic Spike Train Analysis

- Sampling Rate -

○ The membrane potential changes continuously

○ However, it is measured only at discrete points in time, e.g. every 0.1 ms

○ The sampling rate has to be high enough to see the phenomena under study (e.g. the spike shape for spike sorting)

○ For binary representation of spike trains down-sampling may be useful
Basic Spike Train Analysis

- Spike Rate -

Spike rate of single cell
For representation of neuronal activity on a coarser time scale, average responses $S_i(t)$ of neuron $i$ in time window $T$:

$$R_i(t, T) = \frac{1}{T} \sum_{\tau=t}^{t+T} S_i(\tau) \text{ [spikes/s]}$$

Population spike rate
Population activity is calculated as the average of the spike rates of $C$ neurons in a population:

$$R_{\text{pop}}(t, T) = \frac{1}{C} \sum_{i=1}^{C} R_i(t, T) = \frac{1}{T} \cdot \frac{1}{C} \sum_{\tau=t}^{t+T} \sum_{i=1}^{C} S_i(\tau) \text{ [spikes/s]}$$
Retinal Coding
- Experiments -

\[ \theta \text{ stimulus} \]

\[ R_{\text{pop}}(t) \]
population response

\[ S_i(t) \]
cell responses
Basic Spike Train Analysis

- Tuning -

Idea of stimulus tuning of single cell responses

- Represent the dependency of responses $r_i(\theta_l, T)$ on a stimulus property $\theta_l$
- Divide the response sequence into sub-sequences with constant stimulus property
- For each stimulus $N(\theta_l)$ response sequences are used
- Use time-windows of length $T$ after stimulus onset
- Calculate mean and variance of responses to identical stimulation
Basic Spike Train Analysis

- Tuning -

Block-wise analysis of single cell responses

○ Use response sequences with constant stimulation
○ Use time window $T$ after stimulus onset

$$r_i(\theta_1, T) = \frac{(3+4+4)}{3T} = \frac{3.67}{T}$$
$$r_i(\theta_2, T) = \frac{(2+2+3)}{3T} = \frac{2.33}{T}$$
$$r_i(\theta_3, T) = \frac{(1+2+0)}{3T} = \frac{1}{T}$$
Basic Spike Train Analysis

- Tuning -

Average responses to stimulus $\theta_l$

Spike rates of $N(\theta_l)$ responses of cell $i$ to stimulus $\theta_l$ in intervals of length $T$ after stimulus onsets $t_{l,n}$

$$r_i(\theta_l, T) = \frac{1}{N(\theta_l)} \sum_{n=1}^{N(\theta_l)} R_i(t_{l,n}, T) = \frac{1}{N(\theta_l) \cdot T} \sum_{n=1}^{N(\theta_l)} \sum_{\tau=t_{l,n}}^{t_{l,n}+T} S_i(\tau) \text{ [spikes/s]}$$

Tuning function

The tuning function assigns to each value $\theta_l$ of the stimulus set $\Theta$ the expected response $E(r_i | \theta_l)$

$$E(r_i | \theta_l) = r_i(\theta_l, T)$$
Basic Spike Train Analysis

- Tuning -

Examples of RGC velocity tuning

Dependency of spike rate on stimulus velocity:

Symmetrically tuned cell  Direction-sensitive cell

![Graph a](image1)

![Graph b](image2)
Bayesian Stimulus Reconstruction
Bayesian Stimulus Reconstruction

- Idea -

Use the observed neuronal **response** \( r \) to calculate the probability that it was elicited by **stimulus** \( \theta \):

\[
p(\theta|r) = \frac{p(r|\theta)p(\theta)}{p(r)} \propto p(r|\theta)p(\theta)
\]

\[
= \prod_{i=1}^{C} \frac{p(r_i|\theta)p(\theta)}{p(\theta|r_i)}
\]

under the assumption that responses \( r_i \) of all \( C \) cells are conditionally independent (naive Bayes approach).
Bayesian Stimulus Reconstruction

- General Algorithm -

1. Prior function

2. Response functions

3. Split data set

4. Tuning functions

5. Posterior functions

6. Combined posterior

7. Stimulus estimation

8. Estimation performance

9. Repeat
Prior function

**Algorithm:** The prior function $p(\theta)$ describes the probability of each stimulus to occur. Usually it is given by the experimental design.

**Application:** All stimulus velocities $\theta_1-\theta_9$ were presented equally often. The prior function is flat.
Bayesian Stimulus Reconstruction
- Algorithm and Application-

### Response function

**Algorithm:** Calculate responses $r_{i,n}$ for each cell $i$ and stimulus presentation $n$ in time intervals $T$

$$r_{i,n} = R(t_n, T) = \frac{1}{T} \sum_{\tau=t_n}^{t_n+T} S_i(\tau) \quad [\text{spikes/s}]$$

**Application:** Intervals start at each presentation of a new stimulus velocity ($t_n$). $T$ is varied between 10 and 500 ms.
Bayesian Stimulus Reconstruction
- Algorithm and Application-

1. Prior function
2. Response functions
3. Split data set
4. Tuning functions
5. Posterior functions
6. Combined posterior
7. Stimulus estimation
8. Estimation performance
9. Repeat

**Split data set**

**Algorithm:** The data set is divided into two sub-sets. $T_{\text{train}}$ used to determine the response statistics, $T_{\text{test}}$ for stimulus reconstruction.

**Application:** 7/8 of the data set (567 presentations of each stimulus $\theta_l$) is used for training, 1/8 (81 presentations) for testing.
Bayesian Stimulus Reconstruction
- Algorithm and Application-

1. Prior function
2. Response functions
3. Split data set
4. Tuning functions
5. Posterior functions
6. Combined posterior
7. Stimulus estimation
8. Estimation performance
9. Repeat

### Tuning functions

**Algorithm:** Histogram how often each spike rate $r_i$ occurs for $N(\theta_l)$ presentations of stimulus $\theta_l$ in training data set $T_{\text{train}}$

$$p(r_i | \theta_l) = \frac{1}{N(\theta_l)} \sum_{t_l,n \in T_{\text{train}}} \begin{cases} 1 & \text{if } r_i = R_i(t_l,n,T), \\ 0 & \text{otherwise}. \end{cases}$$

**Application:** Individual tuning functions for all 107 cells
Bayesian Stimulus Reconstruction
- Algorithm and Application-

1. Prior function
2. Response functions
3. Split data set
4. Tuning functions
5. Posterior functions
6. Combined posterior
7. Stimulus estimation
8. Estimation performance
9. Repeat

**Posterior functions**

**Algorithm:** Calculate the posterior function for each cell \( i \) and time each time interval \([t_{l,n}, t_{l,n} + T] \in T_{\text{test}}\), using tuning and stimulus prior

\[
p(\theta | r_i(t_{l,n}, T)) = p(r_i | \theta) p(\theta)
\]

**Application:** Calculate posterior functions for all 107 cells and all 729 (81 presentations of 9 stimuli) time intervals \([t_{l,n}, t_{l,n} + T]\)
Bayesian Stimulus Reconstruction
- Algorithm and Application-

**Algorithm:** Calculate the final posterior function by multiplying the posterior functions of all cells. This step assumes the independence of cell responses.

\[ p(\theta|r(t_l,n,T)) = \prod_{i=1}^{C} p(\theta|r_i(t_l,n,T)) \]

**Application:** All 107 cell responses are combined for each time interval.
Bayesian Stimulus Reconstruction
- Algorithm and Application-

**Stimulus estimation**

**Algorithm:** Determine the mode of the posterior distribution as stimulus estimation \( \hat{s}(t_{l,n}) \) for the time interval \([t_{l,n}, t_{l,n} + T] \in T_{\text{test}}\)

**Application:** We use the maximum posterior probability for the entire population as stimulus estimation for each time interval

\[
\hat{s}(t_{l,n}) = \arg \max_{\theta_l \in \Theta} p(\theta_l | r(t_{l,n}, T))
\]
Bayesian Stimulus Reconstruction
- Algorithm and Application-

1. Prior function
2. Response functions
3. Split data set
4. Tuning functions
5. Posterior functions
6. Combined posterior
7. Stimulus estimation
8. Estimation performance

**Estimation performance**

**Algorithm:** Compare the estimated and the actual stimulus sequence with a performance measure $E(\hat{s}(T_{test}), s(T_{test}))$.

**Application:** As simple performance measure we use the percentage of correct estimations

$$E(\hat{s}(T_{test}), s(T_{test})) = \frac{100}{|T_{test}|} \sum_{t \in T_{test}} \left\{ \begin{array}{ll} 1 & \text{if } \hat{s}(t) = s(t) \\ 0 & \text{otherwise} \end{array} \right.$$  

with $|T_{test}|$ the number of elements in $T_{test}$.
Bayesian Stimulus Reconstruction
- Algorithm and Application -

1. Prior function
2. Response functions
3. Split data set
4. Tuning functions
5. Posterior functions
6. Combined posterior
7. Stimulus estimation
8. Estimation performance

**Algorithm:** Steps 3-8 are repeated with different selections of $T_{test}$ and $T_{train}$ to determine mean and standard deviation of estimation performance.

**Application:** Each of the 8 identical stimulus sequences is selected once for testing.
Stimulus velocity (combining direction and speed of motion) is reconstructed well, actual and estimated velocity are very similar.

Occasionally, estimation errors occur.
Bayesian Stimulus Reconstruction
- Results for Retinal Coding -

Estimation errors

- Estimation errors can be due to incorrect estimation of stimulus speed or direction.
- Velocity 0 and high velocities can be reconstructed more reliably than low velocities.

Winzenborg et al., 2009
Bayesian Stimulus Reconstruction
- Results for Retinal Coding -

Estimation depends on cell number

- Estimation performance improves with cell number.
- Small populations encode better than chance level.
- Steep performance increase up to ~30 cells.
Bayesian Stimulus Reconstruction
- Results for Retinal Coding -

Estimation depends on integration time $T$

- Estimation performance improves with longer integration time.
- Velocity is encoded by spike counts on a coarse time scale.
Bayesian Stimulus Reconstruction

Results for Retinal Coding

Comparison of coding hypotheses

- By using different types of tuning functions, different coding hypotheses can be compared.
Bayesian Stimulus Reconstruction

- Summary -

Strengths of Bayesian stimulus estimation
- Based on individual tuning in large populations, explicit coding assumption
- Fast and easy to compute (at least in the simple version presented here)
- Parameters to analyze coding:
  - Integration time scale (vary T)
  - Number of cells (vary C)
  - Type of tuning (vary calculation of $r_i$)

Weakness: Assumes independence of cells
Bayesian Stimulus Reconstruction

- References -

○ Textbook:
  ○ Hastie, Tibshiani & Friedman. The Elements of Statistical Learning. Springer 2001

○ Original publications:
  ○ Winzenborg et al. (under revision)

http://www.sinnesphysiologie.uni-oldenburg.de/en/41986.html
Break...
Metric-based Clustering
Metric-Based Clustering

- Idea -

- Idea: Spike trains are more similar if they were elicited by identical stimuli.

- Known: Several responses to different stimuli.

- Task: Assign a new spike train to the most similar response group.
Metric-Based Clustering

- Idea -

- Spike train $S_a$ is transformed into $S_b$ by a sequence of steps
  - insert a spike $c = 1$
  - delete a spike $c = 1$
  - shift a spike in time $c = q \cdot |s_a-s_b|$

- Each transformation step produces costs
  - Parameter $q$ determines the time scale:
    - If $q \cdot |s_a-s_b| > 2$ it is cheaper to delete the spike at $s_a$ and insert a new one at $s_b$
Metric-Based Clustering

- General Algorithm -

\[ S_a \]

\[ X_1 \]

\[ X_2 \]

\[ X_3 \]

\[ S_b = X_4 \]

\[ \Delta t \]

Deletion cost 1

Time shift cost \( q \cdot \Delta t \)

Time shift cost \( q \cdot \Delta t \)

Insertion cost 1
Metric-Based Clustering
- General Algorithm -

1. Distance $D_{\text{spike}}$ between two spike trains

2. Distances between all pairs of spike trains

3. Cluster distances into response classes

4. Compare response and stimulus classes

5. Estimation performance

6. Vary parameter $q$
1. Distance between two spike trains

Algorithm: The metric $D^{\text{spike}}[q]$ defines the distance between spike trains $S_a$ and $S_b$ as minimal transformation costs for $n$ elementary steps $X_j$

$$D^{\text{spike}}[q](S_a, S_b) = \min\left\{ \sum_{j=0}^{n-1} c(X_j, X_{j+1}) \right\}$$

- insert a spike: $c=1$
- delete a spike: $c=1$
- shift a spike in time: $c=q\cdot\Delta t$

2. $D^{\text{spike}}$ for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter q
Metric-Based Clustering
- Algorithm and Application -

1. $D_{\text{spike}}$

2. $D_{\text{spike}}$ for all pairs

3. Cluster distances

4. Compare classes

5. Estimation performance

6. Vary parameter $q$

Distance between two spike trains

Algorithm: Rules to find the minimal costs to transform $S_a = \{a_1, a_2, ..., a_m\}$ into $S_b = \{b_1, b_2, ..., b_n\}$:
- The last spike is inserted, deleted, or shifted
- Paths of two spikes cannot cross
- No spikes needs to be shifted more than once

Step by step spike times $a_i$ are transformed into $b_j$
Distance between two spike trains

Algorithm: The costs of all possible paths from $S_a$ to $S_b$ are given by a $(m+1 \times n+1)$-matrix $G^{\text{spike}}$

$$G^{\text{spike}}_{i+1,j+1}(S_a, S_b) = \min\{G^{\text{spike}}_{i,j+1} + 1, G^{\text{spike}}_{i+1,j} + 1, G^{\text{spike}}_{i,j} + q \cdot |a_i - b_j|\}$$

$G^{\text{spike}}_{i+1,j+1}$ is the distance between the first $i$ spikes of $S_a$ and the first $j$ spikes of $S_b$

The distance between $S_a$ and $S_b$ is

$$D^{\text{spike}}[q](S_a, S_b) = G^{\text{spike}}_{m+1,n+1}(S_a, S_b)$$
Distance between two spike trains

**Application:** We selected clearly separable single units with stable activity for our analysis.

Responses are divided into sequences to constant motion stimuli of 500 ms length.
Metric-Based Clustering
- Algorithm and Application -

1. \(D_{\text{spike}}\)

2. \(D_{\text{spike}}\) for all pairs

3. Cluster distances

4. Compare classes

5. Estimation performance

6. Vary parameter \(q\)

Distance between all pairs

**Algorithm:** Calculate \(D_{\text{spike}}[q](S_a,S_b)\) for all pairs of spike trains.

For \(N\) presentations of \(K\) stimuli and \(L=N\cdot K\) this results in a \((L\times L)\)-matrix \(M\) with
- \(M_{i,i} = 0\)
- \(M_{i,j} = M_{j,i}\)

**Application:** \(K=9,\ N=648,\ L=5832\) require \(>17,000,000\ G_{\text{spike}}\) matrices
Metric-Based Clustering
- Algorithm and Application-

1. $D_{\text{spike}}$
2. $D_{\text{spike}}$ for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter $q$

Distance between all pairs

responses are sorted in stimulus classes
Metric-Based Clustering
- Algorithm and Application -

1. $D^{\text{spike}}$
2. $D^{\text{spike}}$ for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter $q$

Distance between all pairs

(reduced data set with $N=5$)
Metric-Based Clustering
- Algorithm and Application-

1. $D^{\text{spike}}$

2. $D^{\text{spike}}$ for all pairs

3. Cluster distances

4. Compare classes

5. Estimation performance

6. Vary parameter $q$

Cluster distances into response classes

**Algorithm:** Compare each spike train $S$ to the stimulus class $s_l$ (responses to $\theta_l$) with distance between $S$ and $s_l$:

$$d(S, s_l) = \left( \langle D[q](S, S') \mid S' \in s_l \rangle \right)^{1/z}$$

$S$ is assigned to response class $r_l$ (estimated stimulus $\theta_l$) with minimal $d(S, s_l)$.

**Application:** $z=1$: mean distance
Metric-Based Clustering
- Algorithm and Application -

1. \( D_{\text{spike}} \)
2. \( D_{\text{spike}} \) for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter \( q \)

**Algorithm:** Generate a confusion matrix \( C(s_l, r_m) \):

- Initialize with 0
- For each \( S \) increase \( C_{l,m} \) by 1 with actual stimulus \( \theta_l \) and response class \( r_m \)

**Application:** A (9x9)-Matrix shows how often responses to velocity \( \theta_l \) were estimated as responses to \( \theta_m \)

**Compare resp. & stim. classes**
Metric-Based Clustering
- Algorithm and Application-

1. $D_{\text{spike}}$

2. $D_{\text{spike}}$ for all pairs

3. Cluster distances

4. Compare classes

5. Estimation performance

6. Vary parameter $q$

Compare resp. & stim. classes

Real case example

![Matrix Plot]

- Estimated Velocity [mm/s]
- Actual Velocity [mm/s]

$1/q = \text{Inf}$
Metric-Based Clustering
- Algorithm and Application-

1. $D_{\text{spike}}$

2. $D_{\text{spike}}$ for all pairs

3. Cluster distances

4. Compare classes

5. Estimation performance

6. Vary parameter $q$

**Estimation performance**

**Algorithm:** Originally, transmitted Information was used to measure estimation performance (Victor 1997, 2005).

However, for simplicity and easy comparison with other methods we use percentage of correct classification
Metric-Based Clustering
- Algorithm and Application -

1. $D_{\text{spike}}$
2. $D_{\text{spike}}$ for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter q

Vary parameter q

**Algorithm:** Parameter $q$ determines the studied time scale.
- $q=0$ (no costs for time shifts) spike count code
- $q$ small (low costs for time shifts) spike rate code with large integration window
- $q$ large (high costs for time shifts) temporal code with small integration window
Metric-Based Clustering
- Algorithm and Application-

1. $D_{\text{spike}}$
2. $D_{\text{spike}}$ for all pairs
3. Cluster distances
4. Compare classes
5. Estimation performance
6. Vary parameter $q$

Application: $q$ values to cover the entire range from spike count to temporal coding

<table>
<thead>
<tr>
<th>$q$ [1/s]</th>
<th>time scale (1/q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>spike count</td>
</tr>
<tr>
<td>1</td>
<td>1 s</td>
</tr>
<tr>
<td>2</td>
<td>500 ms</td>
</tr>
<tr>
<td>4</td>
<td>250 ms</td>
</tr>
<tr>
<td>8</td>
<td>125 ms</td>
</tr>
<tr>
<td>16</td>
<td>62.5 ms</td>
</tr>
<tr>
<td>32</td>
<td>31.25 ms</td>
</tr>
<tr>
<td>64</td>
<td>16 ms</td>
</tr>
<tr>
<td>128</td>
<td>8 ms</td>
</tr>
</tbody>
</table>
Metric-Based Clustering

- Results for Retinal Coding -

Confusion matrices shows coding of cell types:

Good speed estimation, bad direction estimation

Speed and direction estimation only good for preferred direction
Metric-Based Clustering

- Results for Retinal Coding -

Variation of q shows type of coding:

- Same type of coding for all types of cells
- Best estimation for spike rates based on long time windows
- Individual cells better than chance
Metric-Based Clustering

- Analysis of Multi-Unit Activity --

Idea

○ Combine spike trains of more than one neuron to a new sequence of labeled spikes, e.g.
\[ S_a = \{ a_{1}^{(w)}, a_{2}^{(v)}, a_{3}^{(v)} \ldots a_{n}^{(w)} \} \]

- Calculate distances between combined spike trains

○ Additional transformation step: Re-label a spike with cost \( k \).
$S_a^v$
$S_a^w$
$S_a = X_0$
$X_1$
$X_2$
$X_3$
$X_4$
$X_5 = S_b$
$S_b^v$
$S_b^w$

- Deletion cost 1
- Re-label cost $k$
- Time shift cost $q \cdot \Delta t$
- Time shift cost $q \cdot \Delta t$
- Insertion cost 1
Metric-Based Clustering

- Analysis of Multi-Unit Activity -

**Algorithm**

- Parameter $k$ determines how important it is to know the neuron of origin for each spike:
  - $k=0$: no costs for re-labeling, population activity
  - $k=2$: re-labeling as expensive as deletion spike at $a_i^{(v)}$ and insertion a new one at $b_j^{(w)}$

- Modification: Only spikes of the same neuron are not allowed to cross - preceding spikes with different labels need to be considered
Metric-Based Clustering

- Analysis of Multi-Unit Activity -

**Algorithm:**

- The algorithm is extremely time-consuming, computation time grows with $U^{C+1}$ with number of Neurons $C$ and typical spike number $U$

**Application:**

- Calculation time on a PC for one combination of $k$ and $q$ in our example: $U=32,000$
  - $C=3$: 3 hours
  - $C=5$: 11 days
Metric-Based Clustering
- Analysis of Multi-Unit Activity -

Results

Confusion matrix shows encoding of entire ensemble
Good estimation of motion direction
Good estimation of motion speed

Combination of 3 cells, \( q=0, k=2 \)
Metric-Based Clustering

- Analysis of Multi-Unit Activity -

Results

- Combined activity improves estimation if neuron of origin is known ($k > 0$)

- Redundancy of responses: combined responses are not as good as linear sum

More details tomorrow:
Poster by L. Juarez Paz
Metric-Based Clustering

- Summary -

Strengths of Metric-Based Clustering

- Based on responses of one or few cells
- No explicit coding assumption, test several assumptions together
- Parameters to analyze coding:
  - Integration time scale (vary q)
  - Number of cells (vary C)
  - Combination of responses (vary k)

Weakness: Computationally expensive
Bayesian Stimulus Reconstruction

- References -

- Review:

- Original publications:

http://www.sinnesphysiologie.uni-oldenburg.de/en/41986.html
## Conclusions

<table>
<thead>
<tr>
<th>Coding Assumption</th>
<th>Bayesian Reconstruction</th>
<th>Metric-based Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Coding</td>
<td>tuning based on defined response feature</td>
<td>rate &amp; temporal coding, time-scales varied</td>
</tr>
<tr>
<td>Population Size</td>
<td>assumes independent tuning</td>
<td>analysis of importance of spike origin in combined responses</td>
</tr>
<tr>
<td>Computing Time</td>
<td>up to very large populations, combination of recordings possible</td>
<td>individual cells or small groups of synchronously recorded responses</td>
</tr>
<tr>
<td></td>
<td>low enough for PC</td>
<td>high! parallel computing for combined responses</td>
</tr>
</tbody>
</table>
Thanks

The Bayes Group:
Insa Winzenborg
Peter Harmand
Martin Greschner
Andreas Thiel

The Metrics Group:
Leon M. Juarez Paz
Imke Reimer

Thank you!