Bayesian model selection for statistical analysis of neural data: Lessons from fMRI

Joram Soch, BCCN Berlin

Seminar: Current Topics in Computational Neuroscience
WS 2018/2019: Statistical Analysis of Neural Data
Wed, 13/02/2019, 10.15 a.m.

Outline

Part I: Theory

1. What is (Bayesian) model selection?
2. The (very few) formulas that we'll need
3. A (very brief) introduction to fMRI data analysis
4. Bayesian model selection for fMRI data analysis
5. Theory Q&A

Part II: Practice

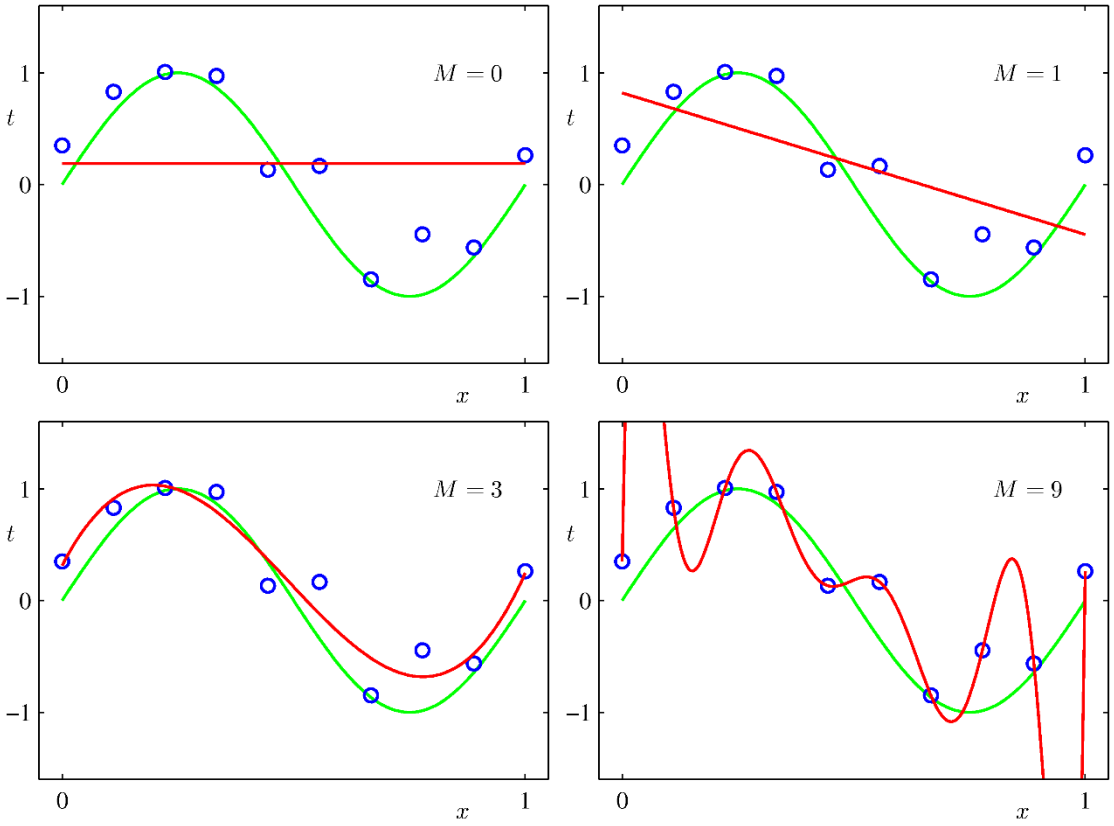
6. Code: PycvBMS – a Python module for model selection
7. Demo: Bayesian model selection for Neuropixel probes
8. What do I need to perform (Bayesian) model selection?
9. Practice Q&A

Part I: Theory

1.

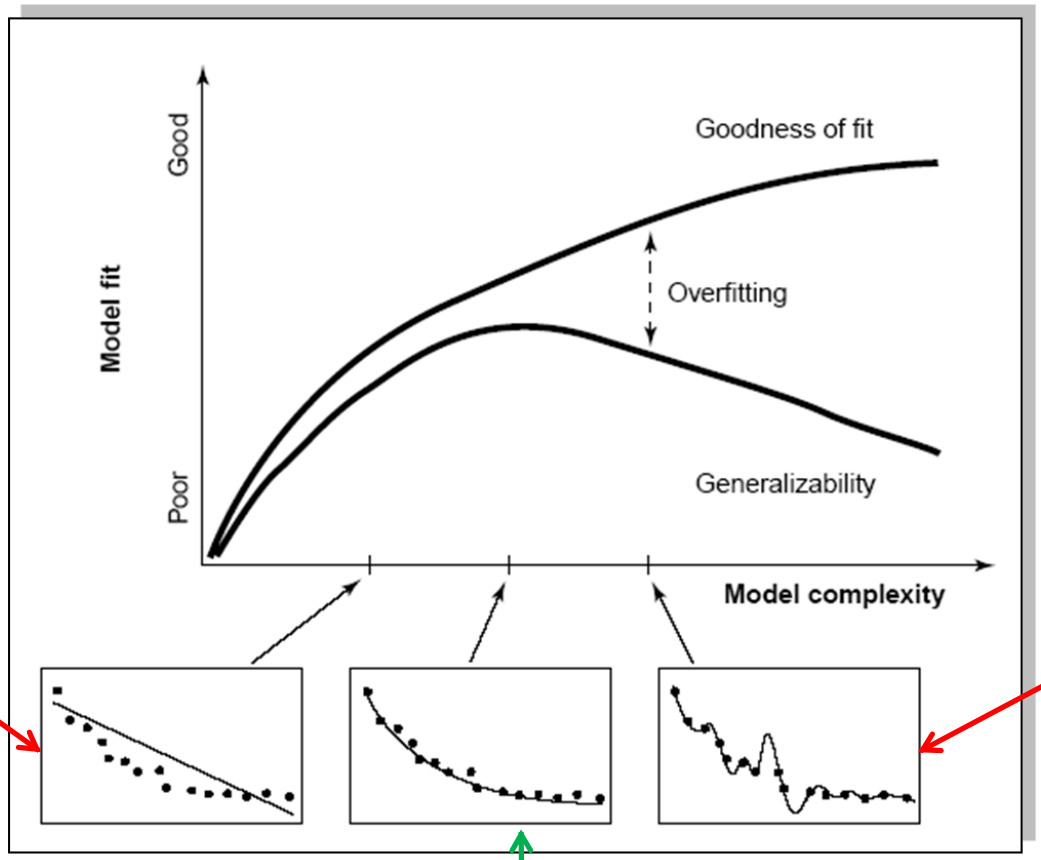
What is (Bayesian) model selection?

Statistical model selection



Bishop, 2006, Fig. 1.4

Underfitting and Overfitting



UNDERFITTING

OVERFITTING

JUST RIGHT

Pitt & Miyung, TICS, 2002

General Principle

$$\text{model quality} = \text{model accuracy} - \text{model complexity}$$

Penny, NeuroImage, 2012

2.

The (very few) formulas that we‘ll need

Classical information criteria

$$p(y|\theta, m)$$

likelihood function

$$\hat{\theta} = \arg\max_{\theta} p(y|\theta, m)$$

maximum likelihood estimation

$$\mathrm{MLL} = \log p(y|\hat{\theta}, m)$$

maximum log-likelihood

$$\mathrm{IC} = -2\,\mathrm{MLL} + f_{\mathrm{IC}}(n, k)$$

classical information criterion

$$f_{\mathrm{AIC}} = 2\,k$$
$$f_{\mathrm{BIC}} = k\,\log n$$

Akaike, IEEE TAC, 1974; Schwarz, Ann Stat, 1978

The Bayesian model evidence

likelihood function prior distribution

posterior distribution $$p(\theta|y,m) = \frac{p(y|\theta,m)\,p(\theta|m)}{p(y|m)}$$

(model) „evidence“

model evidence $$p(y|m) = \int p(y|\theta,m)\,p(\theta|m)\,\mathrm{d}\theta$$

log model evidence $$\mathrm{LME}(m) = \log p(y|m)$$

MacKay, 2003, eq. 28.7; Bishop, 2007, eq. 1.45; Koch, 2007, eq. 3.72; Gelman et al., 2013, eq. 1.3

The cross-validated LME

$$p(y|m) = \int p(y|\theta, m)\, p(\theta|m)\, \mathrm{d}\theta$$

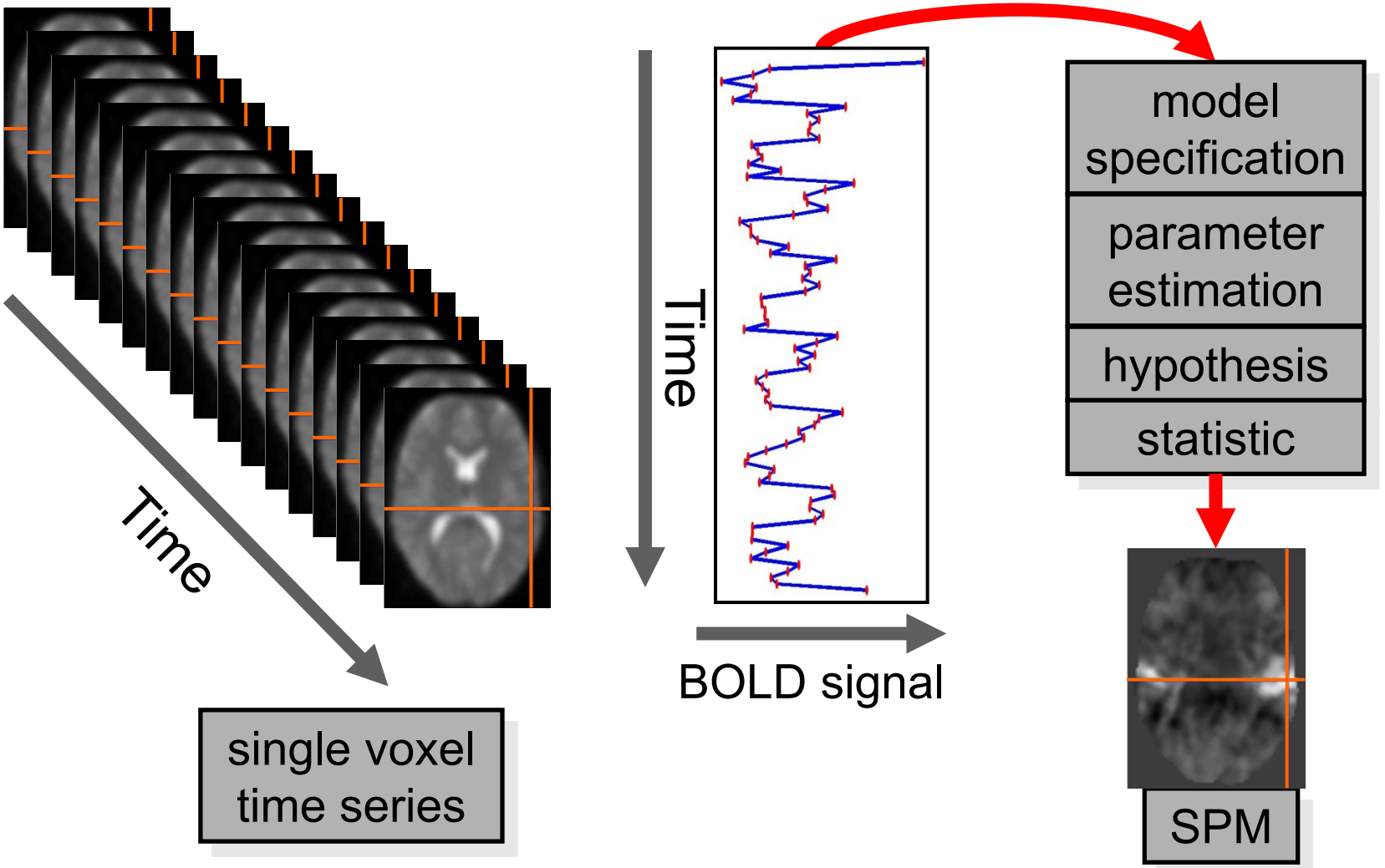
$$\mathrm{cvLME}(m) = \sum_{i=1}^{S} \log \int p(y_i|\theta)\, p(\theta|\, \cup_{j \neq i}\, y_j)\, \mathrm{d}\theta$$

Soch et al., NeuroImage, 2016; Soch et al., NeuroImage, 2017

3.

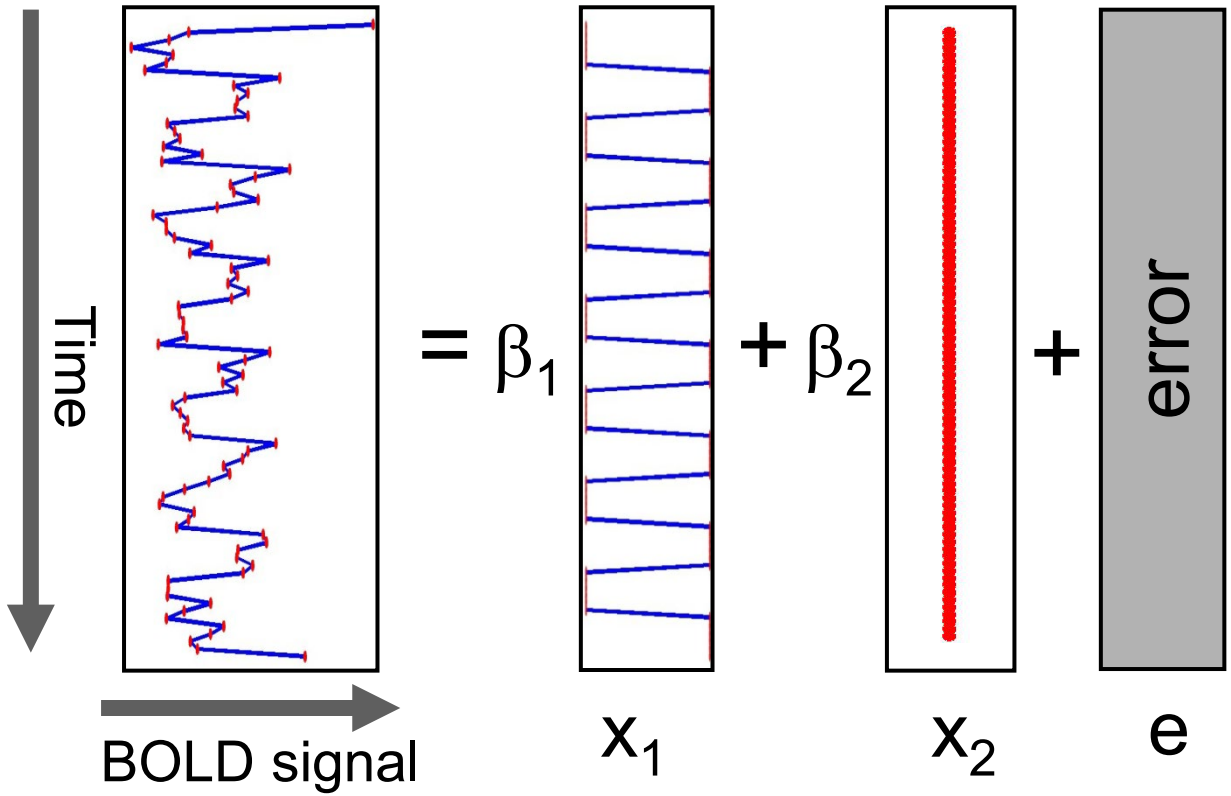
A (very brief) introduction to fMRI data analysis

Voxel-wise time series analysis



Stephan, 2010

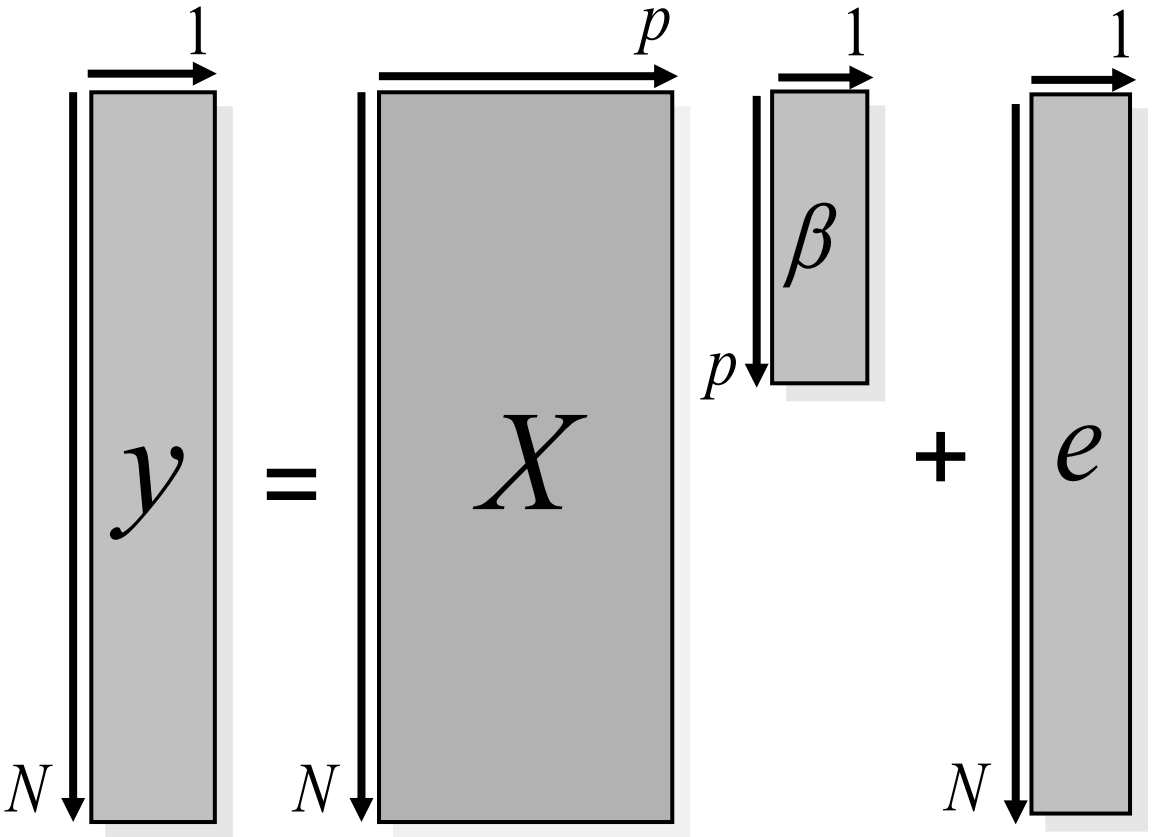
Single-voxel regression model



$$y = x_1\beta_1 + x_2\beta_2 + e$$

Stephan, 2010

Mass-univariate analysis: voxel-wise GLM



$$y = X\beta + e$$

$$e \sim N(0, \sigma^2 I)$$

Model is specified by
1. Design matrix X
2. Assumptions about e

N: number of scans
p: number of regressors

The design matrix embodies all available knowledge about experimentally controlled factors and potential confounds.

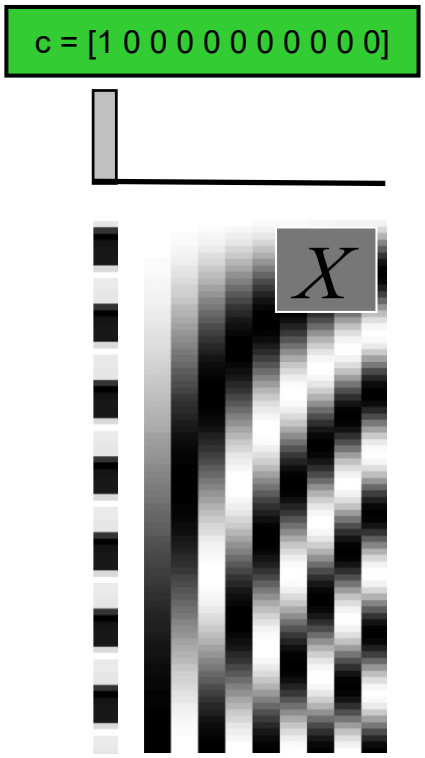
Stephan, 2010

Ordinary least squares: beta estimates

$$
\begin{aligned}
X^T\hat{\varepsilon} &= 0 \\
X^T(y - X\hat{\beta}) &= 0 \\
X^T y - X^T X\hat{\beta} &= 0 \\
X^T X\hat{\beta} &= X^T y \\
\hat{\beta} &= \left(X^T X\right)^{-1} X^T y
\end{aligned}
$$

Stephan, 2010

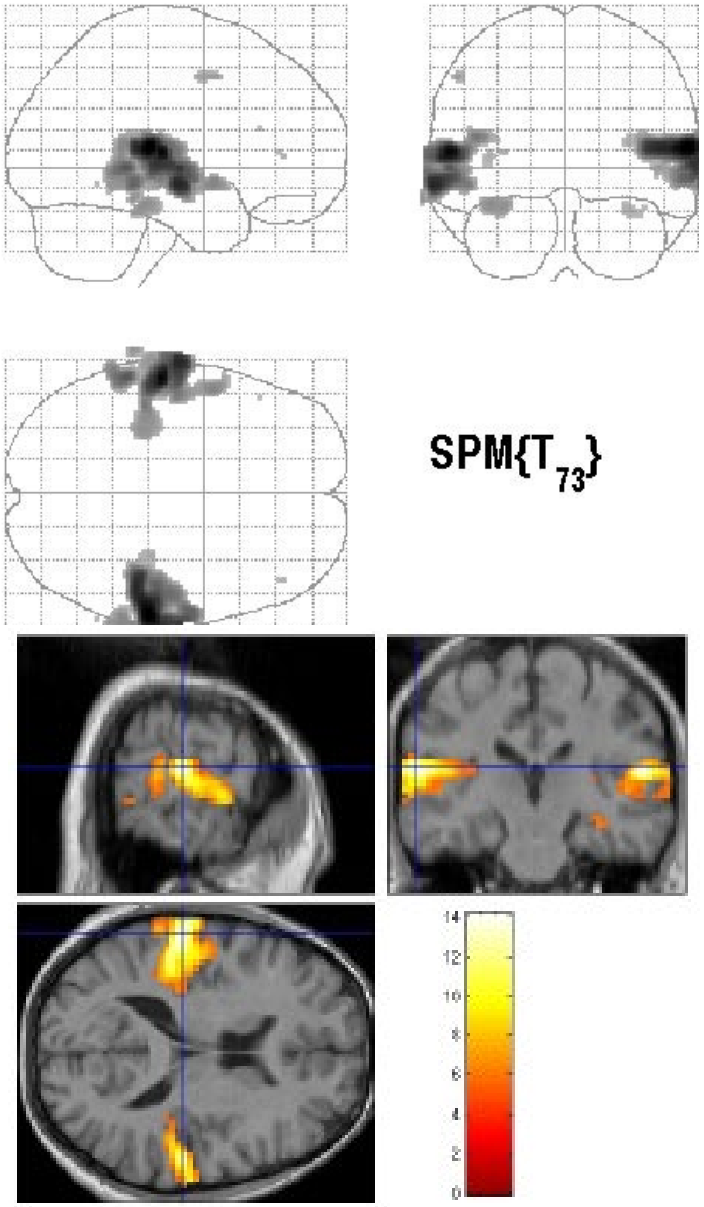
Contrasts & statistical parametric maps



Q: activation during listening ?

Null hypothesis: $\beta_1 = 0$

$$t = \frac{c^T \hat{\beta}}{Std(c^T \hat{\beta})}$$

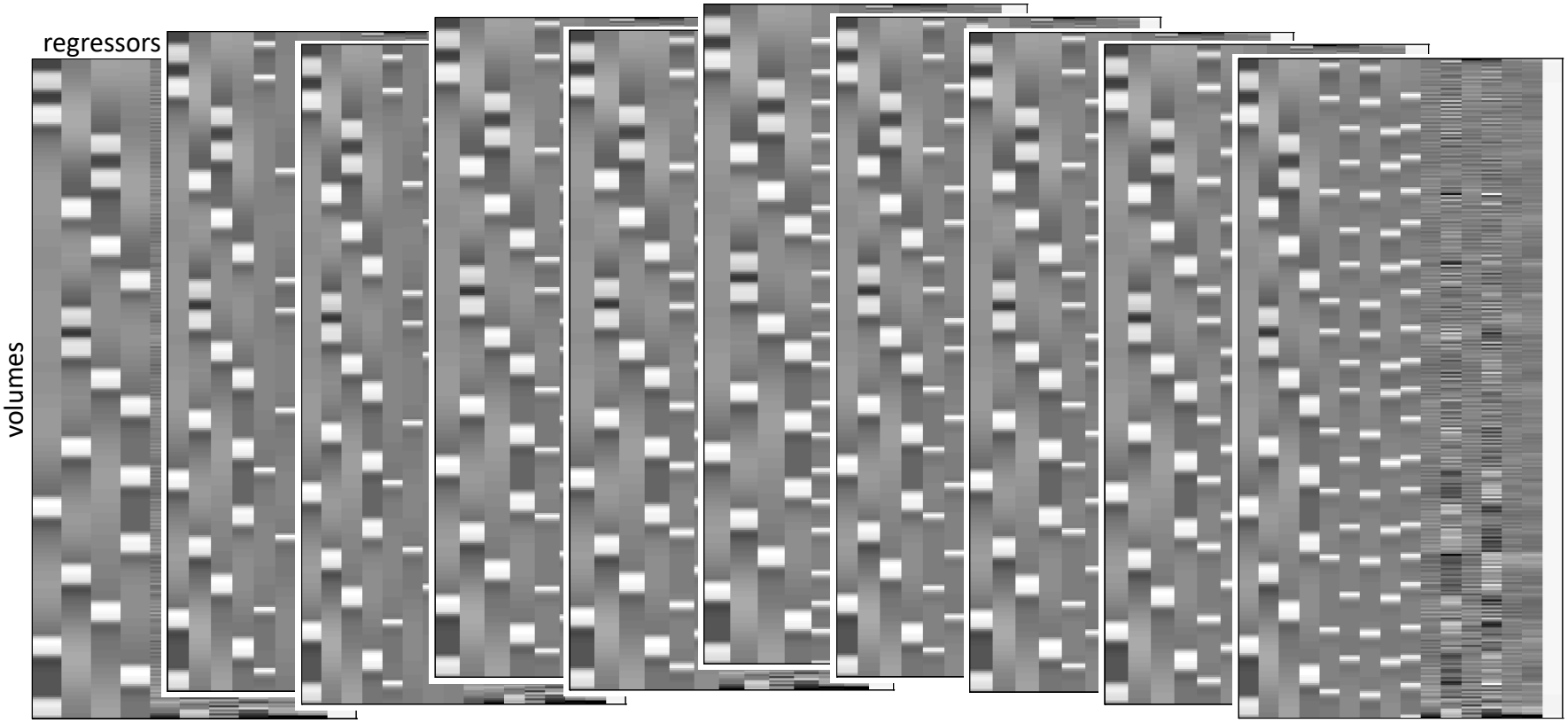


Stephan, 2010

4.

Bayesian model selection for fMRI data analysis

Many plausible GLMs for a given fMRI data set



Soch et al., NeuroImage, 2017

cross-validated Bayesian model selection

First-level analysis:

$$y = X\beta + \varepsilon,\ \varepsilon \sim \mathrm{N}(0, \sigma^2 V)$$

$$p(y|m) = \iint p(y|\beta, \sigma^2, m)\, p(\beta, \sigma^2|m)\, \mathrm{d}\beta\, \mathrm{d}\sigma^2$$

$$\mathrm{cvLME}(m) = \sum_{i=1}^{S} \log \int p(y_i|\theta)\, p(\theta| \cup_{j \neq i} y_j)\, \mathrm{d}\theta$$

Second-level analysis:

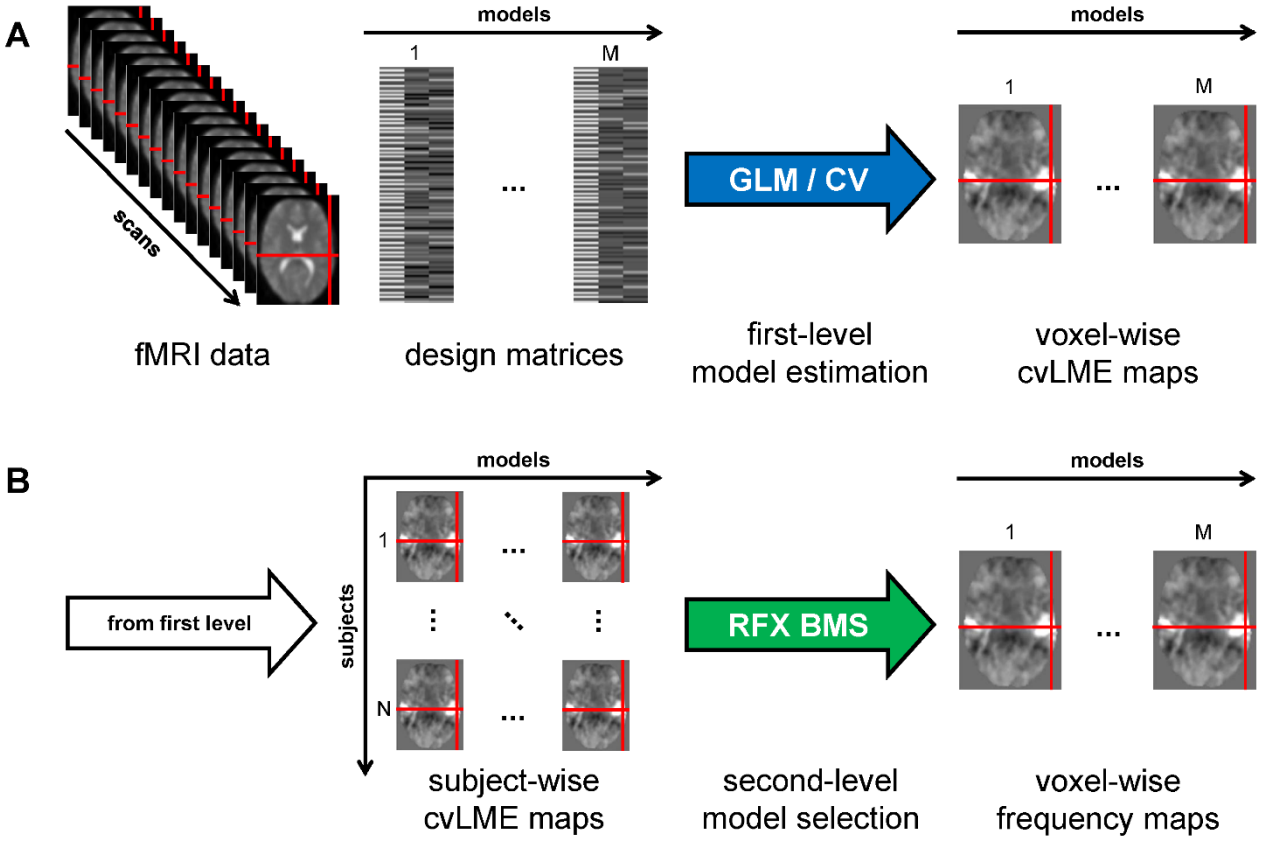
$$p(y|m) = \prod_{i=1}^{N} p(y_i|m_i) = \prod_{i=1}^{N} \prod_{j=1}^{M} p(y_i|e_j)^{m_{ij}}$$

$$p(m|r) = \prod_{i=1}^{N} \mathrm{Mult}(m_i; 1, r) = \prod_{i=1}^{N} \prod_{j=1}^{M} r_j^{m_{ij}}$$

$$p(r|\alpha) = \mathrm{Dir}(r; \alpha) = \frac{\Gamma(\sum \alpha_j)}{\sum \Gamma(\alpha_j)} \prod_{j=1}^{M} r_j^{\alpha_j - 1}$$

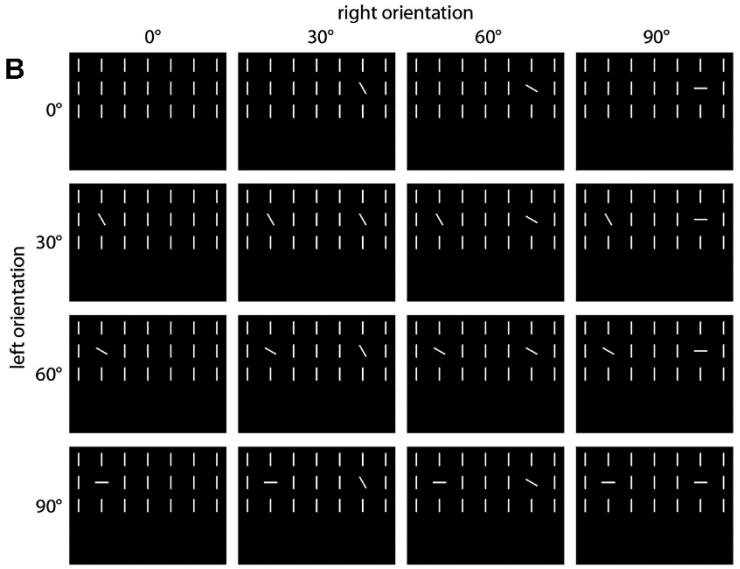
Soch et al., OHBM, 2015; Soch et al., NeuroImage, 2016; Stephan et al., NeuroImage, 2009; Penny et al., PLoS CB, 2010

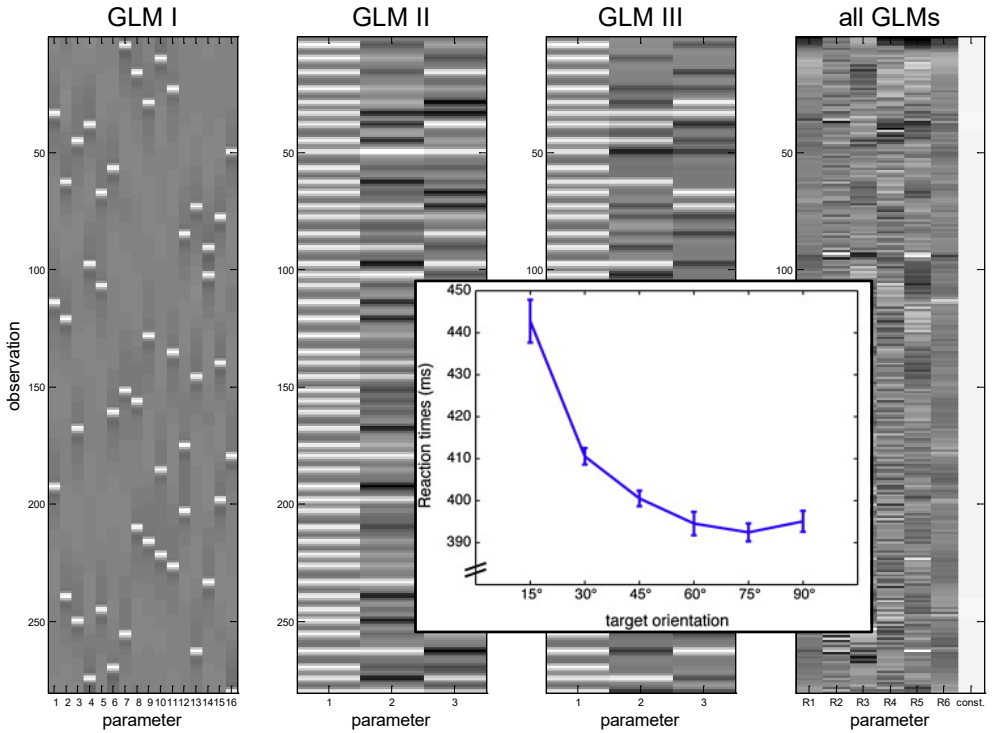
The cvBMS approach in practice



Soch et al., NeuroImage, 2016

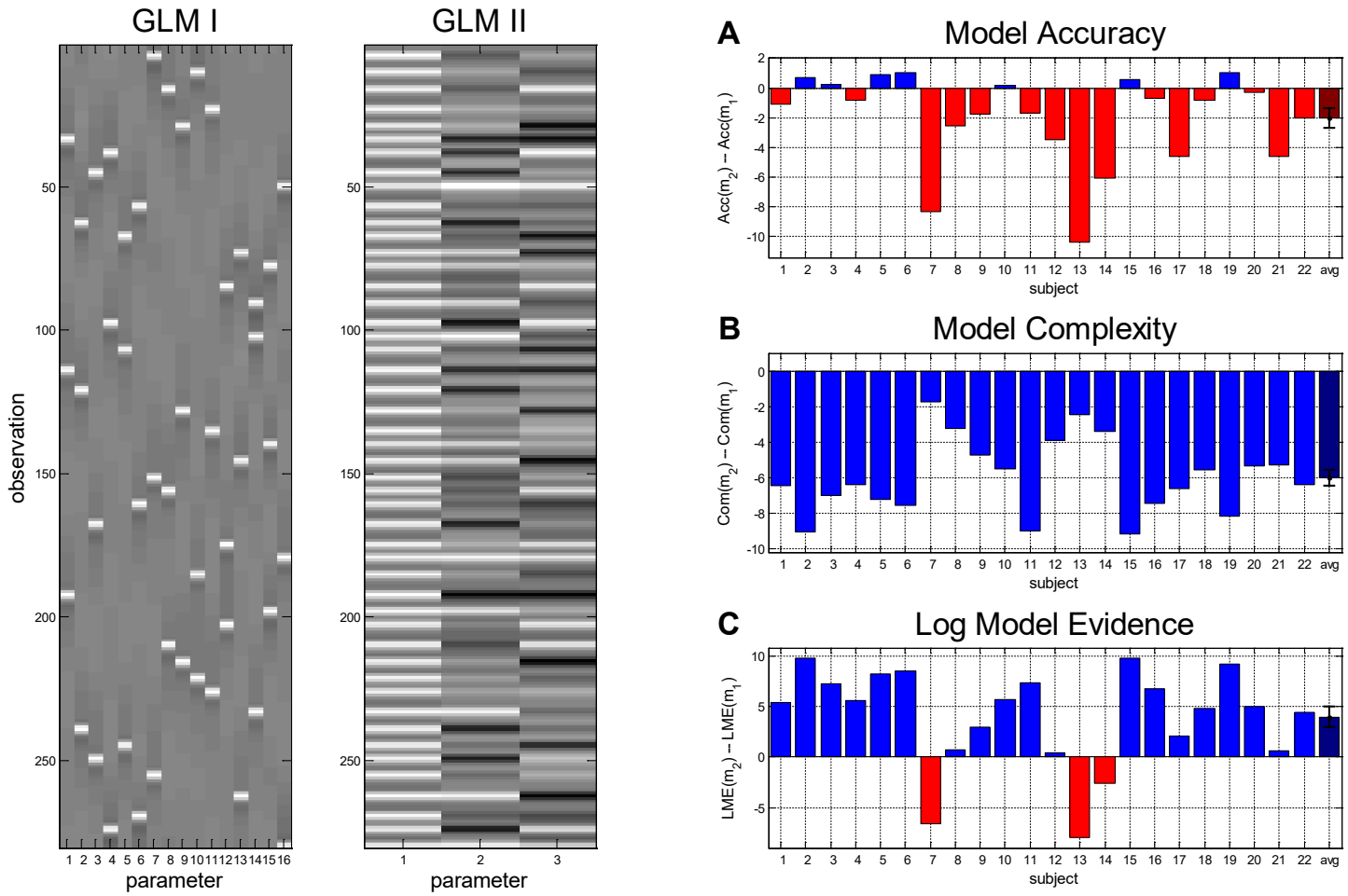
Empirical example: model space





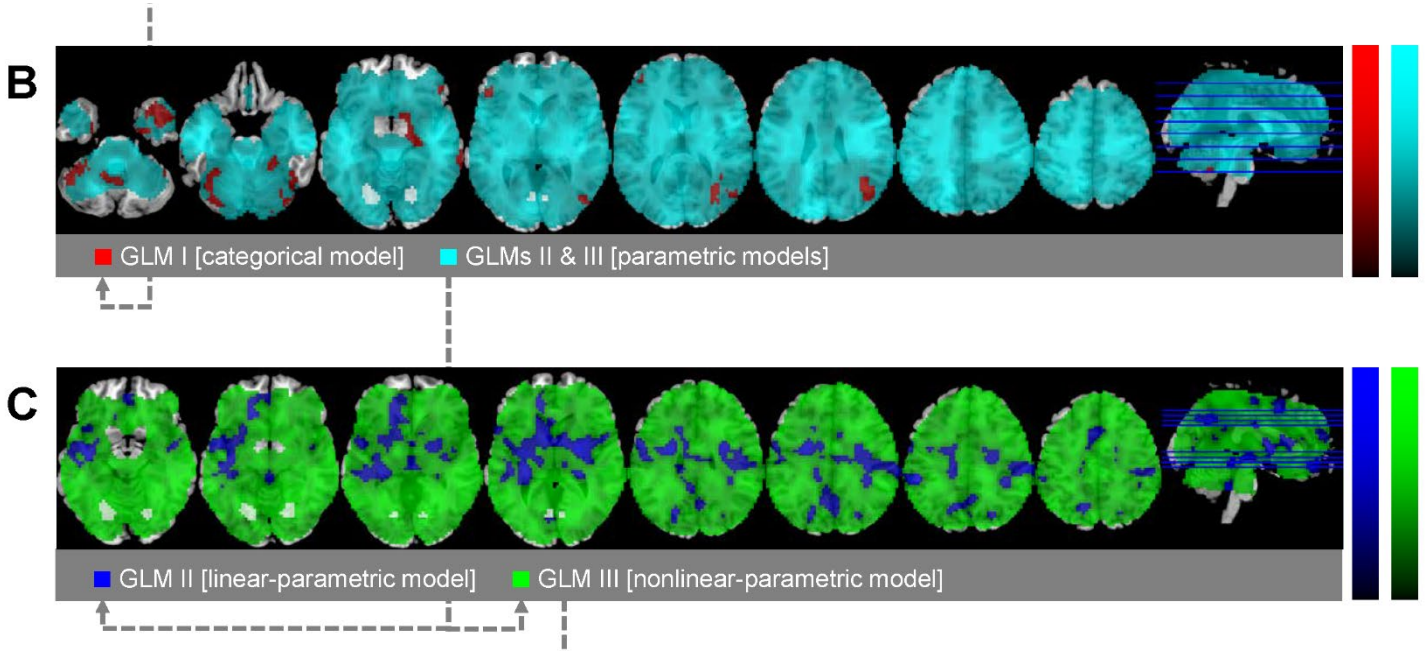
Bogler et al., NeuroImage, 2013; Soch et al., OHBM, 2015

Model selection: GLMs I/II



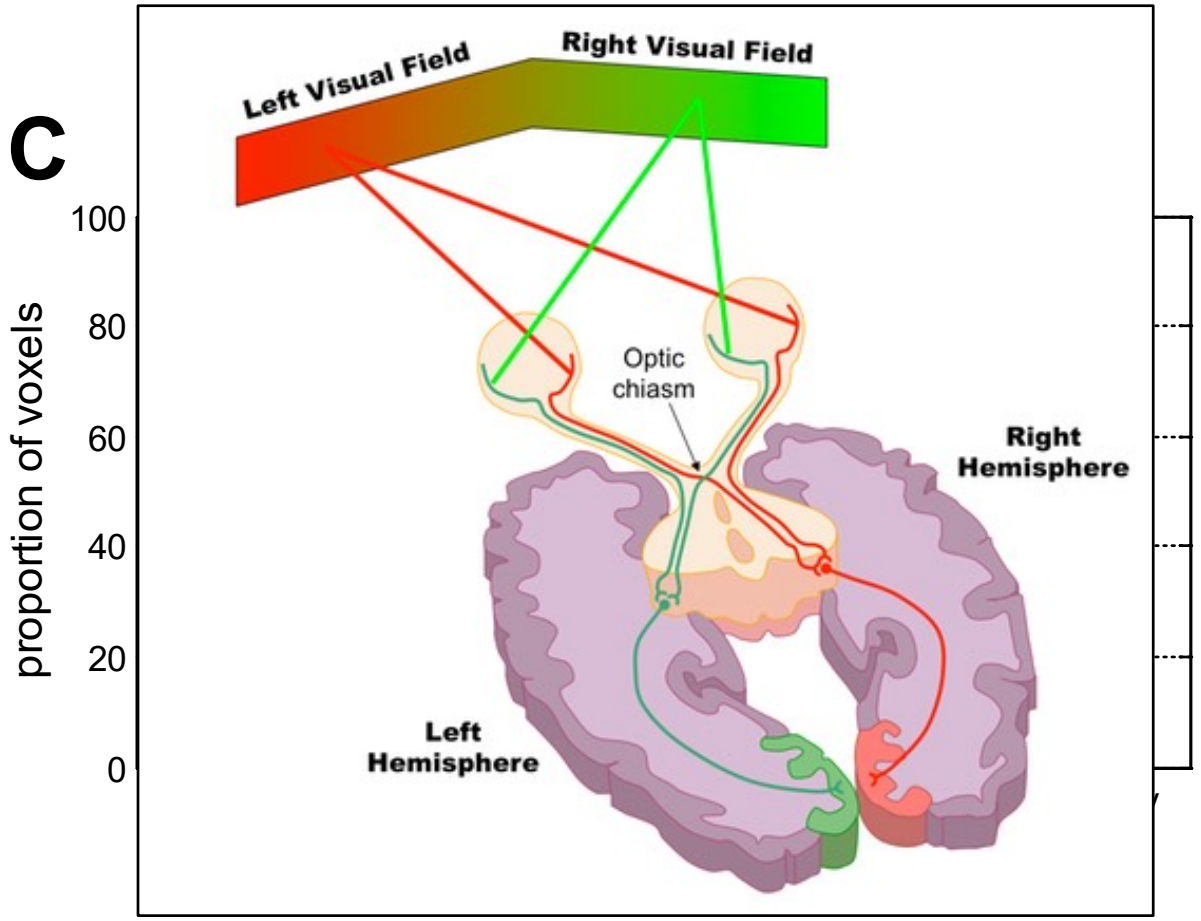
Soch et al., NeuroImage, 2016

Model selection: GLMs I-III



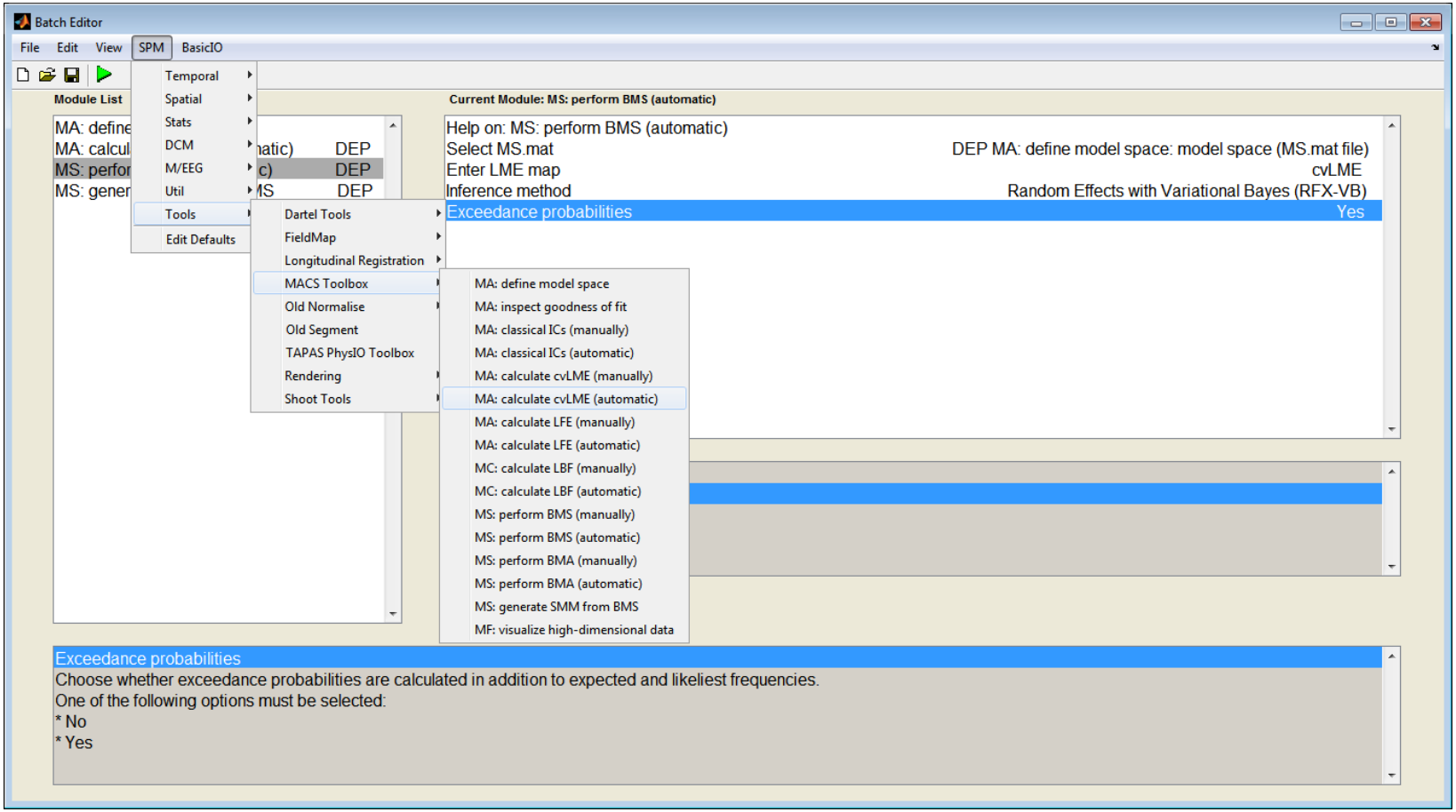
Soch et al., NeuroImage, 2016

Model selection: GLMs III-l/r



Soch et al., NeuroImage, 2016

The MACS toolbox for SPM



Soch & Allefeld, JNeuMeth, 2018; URL: https://github.com/JoramSoch/MACS

5.

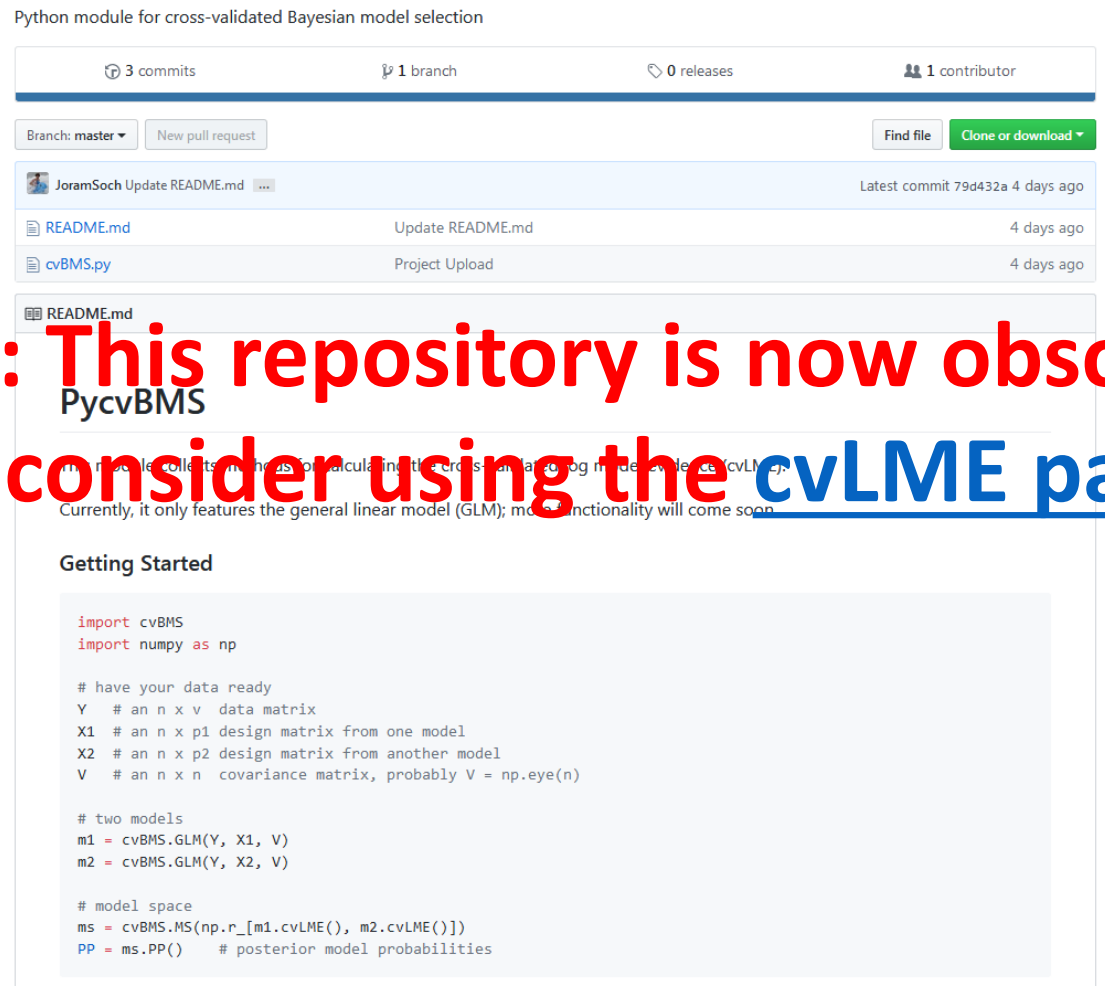
Theory Q&A

Part II: Practice

6.

Code: PycvBMS – a Python module for model selection

PycvBMS – available from GitHub



Python module for cross-validated Bayesian model selection

3 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Find file | Clone or download

JoramSoch Update README.md ... Latest commit 79d432a 4 days ago

README.md	Update README.md	4 days ago
cvBMS.py	Project Upload	4 days ago

README.md

PycvBMS

Currently, it only features the general linear model (GLM); model comparison functionality will come soon.

Getting Started

```
import cvBMS
import numpy as np

# have your data ready
Y   # an n x v  data matrix
X1  # an n x p1 design matrix from one model
X2  # an n x p2 design matrix from another model
V   # an n x n  covariance matrix, probably V = np.eye(n)

# two models
m1 = cvBMS.GLM(Y, X1, V)
m2 = cvBMS.GLM(Y, X2, V)

# model space
ms = cvBMS.MS(np.r_[m1.cvLME(), m2.cvLME()])
PP = ms.PP()    # posterior model probabilities
```

Note: This repository is now obsolete.
Please consider using the cvLME package!

URL: ~~https://github.com/JoramSoch/PycvBMS~~; https://github.com/JoramSoch/cvLME

PycvBMS – four easy steps

1. define GLM(s)
2. calculate cvLMEs
3. define model space
4. calculate PPs

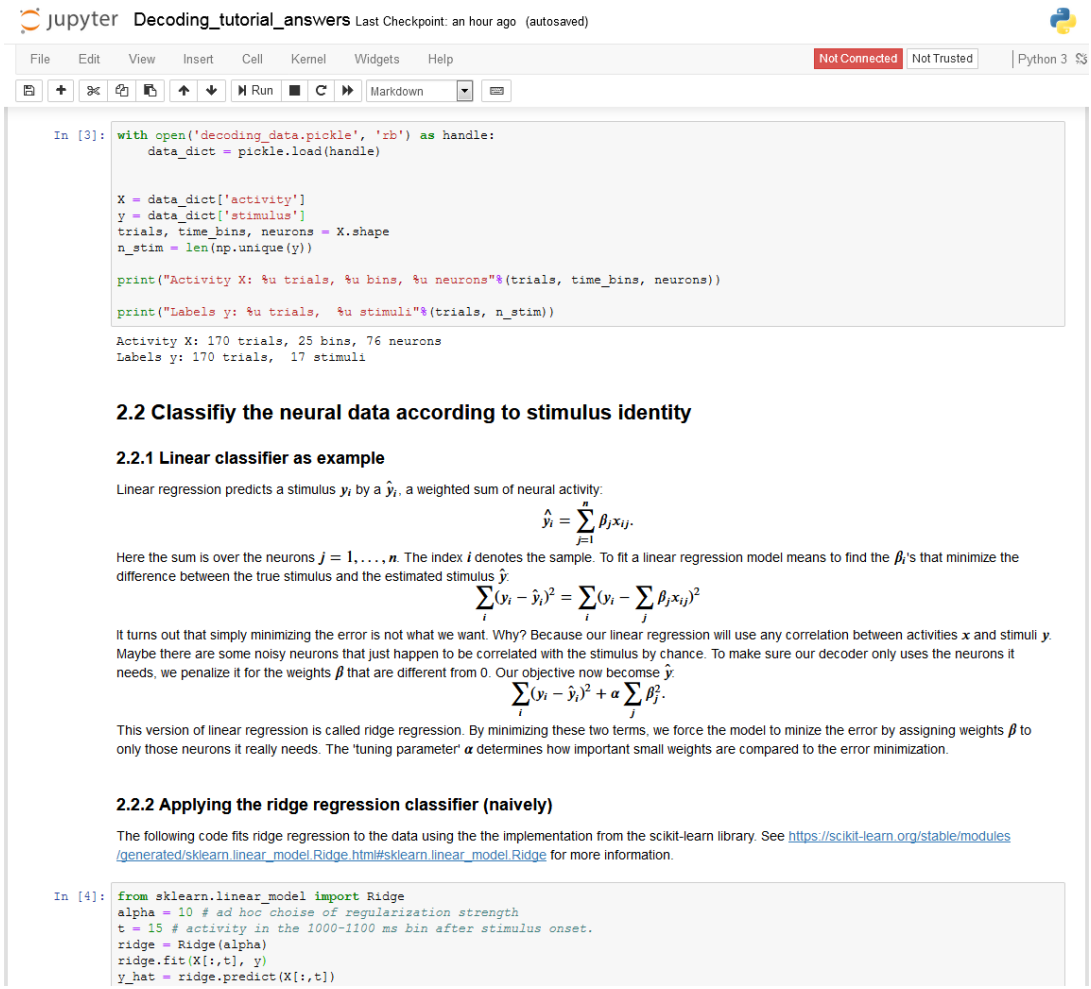
PycvBMS – a little demo

[switch to Anaconda/Spyder]

7.

Demo: Bayesian model selection for Neuropixel probes

What do we analyze?



jupyter Decoding_tutorial_answers Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Not Connected — Not Trusted — Python 3

Run — Markdown

In [3]:
```
with open('decoding_data.pickle', 'rb') as handle:
    data_dict = pickle.load(handle)


X = data_dict['activity']
y = data_dict['stimulus']
trials, time_bins, neurons = X.shape
n_stim = len(np.unique(y))

print("Activity X: %u trials, %u bins, %u neurons"%(trials, time_bins, neurons))

print("Labels y: %u trials,  %u stimuli"%(trials, n_stim))
```

```
Activity X: 170 trials, 25 bins, 76 neurons
Labels y: 170 trials,  17 stimuli
```

2.2 Classifiy the neural data according to stimulus identity

2.2.1 Linear classifier as example

Linear regression predicts a stimulus y_i by a $\hat{y}_i$, a weighted sum of neural activity:

$$\hat{y}_i = \sum_{j=1}^{n} \beta_j x_{ij}.$$

Here the sum is over the neurons $j = 1, \ldots, n$. The index i denotes the sample. To fit a linear regression model means to find the β_i's that minimize the difference between the true stimulus and the estimated stimulus $\hat{y}$.

$$\sum_i (y_i - \hat{y}_i)^2 = \sum_i (y_i - \sum_j \beta_j x_{ij})^2$$

It turns out that simply minimizing the error is not what we want. Why? Because our linear regression will use any correlation between activities x and stimuli y. Maybe there are some noisy neurons that just happen to be correlated with the stimulus by chance. To make sure our decoder only uses the neurons it needs, we penalize it for the weights β that are different from 0. Our objective now becomse $\hat{y}$.

$$\sum_i (y_i - \hat{y}_i)^2 + \alpha \sum_j \beta_j^2.$$

This version of linear regression is called ridge regression. By minimizing these two terms, we force the model to minize the error by assigning weights β to only those neurons it really needs. The 'tuning parameter' α determines how important small weights are compared to the error minimization.

2.2.2 Applying the ridge regression classifier (naively)

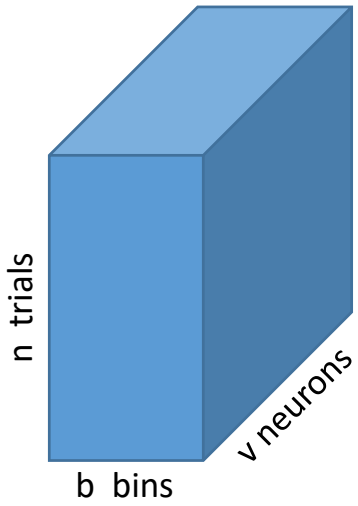
The following code fits ridge regression to the data using the the implementation from the scikit-learn library. See https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html#sklearn.linear_model.Ridge for more information.

In [4]:
```
from sklearn.linear_model import Ridge
alpha = 10 # ad hoc choise of regularization strength
t = 15 # activity in the 1000-1100 ms bin after stimulus onset.
ridge = Ridge(alpha)
ridge.fit(X[:,t], y)
y_hat = ridge.predict(X[:,t])
```

URL: http://data.cortexlab.net/dualPhase3/

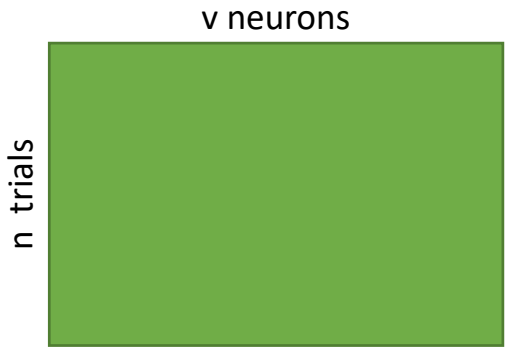
What data were acquired?

Data come as a



3D NumPy array of spike counts/100 ms

and we will analyze the



matrix belonging to the k = 16-th time bin, i.e. 1000-1100 ms after stimulus onset (out of 1500 ms total stimulation time).

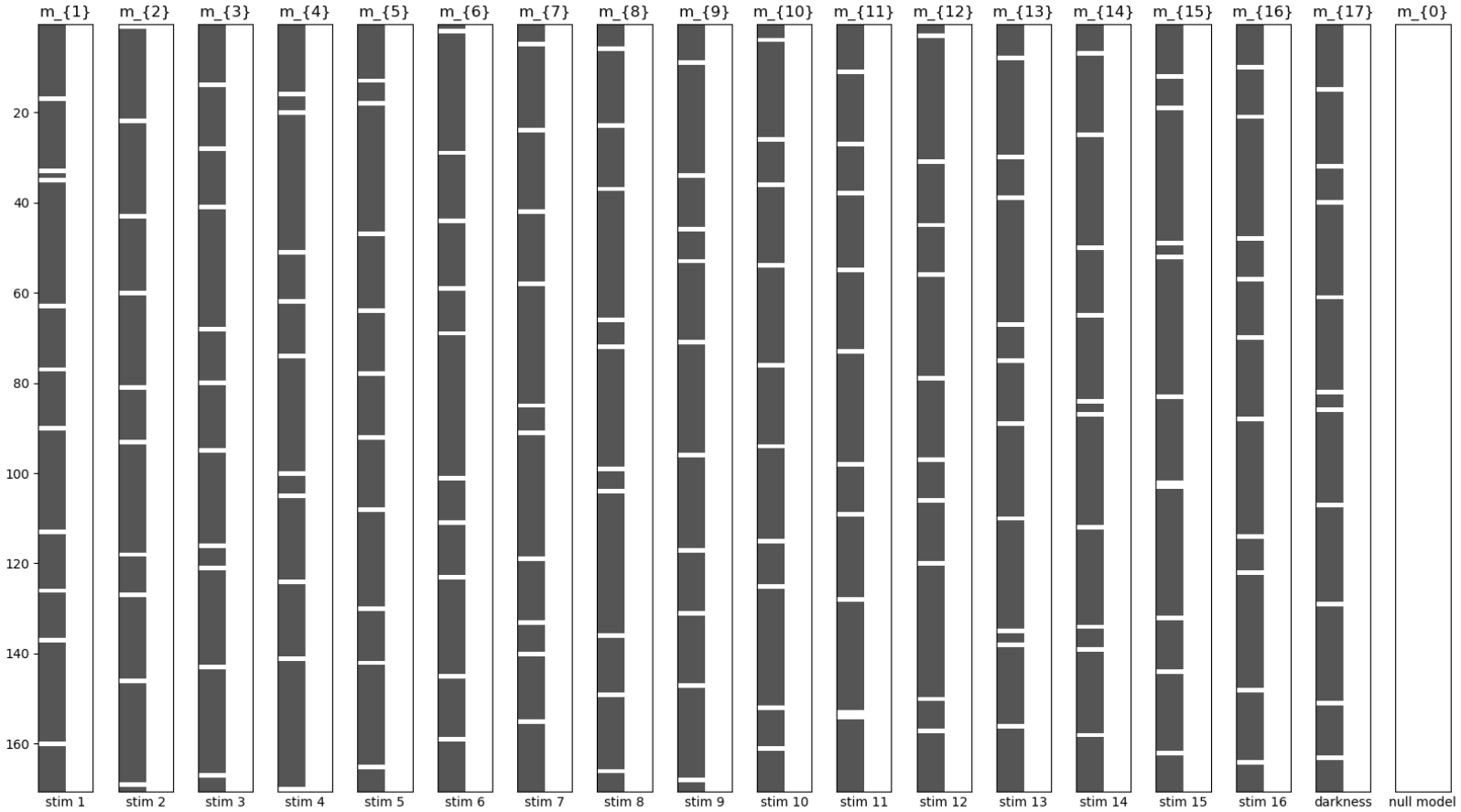
Which stimuli were presented?

Each trial is coded as ...	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
correspoding to ...	0°		45°		90°		135°		180°		225°		270°		315°		dark-ness

Research Question No. 1

„Which neuron reacts to which stimulus?“

Q1: model space

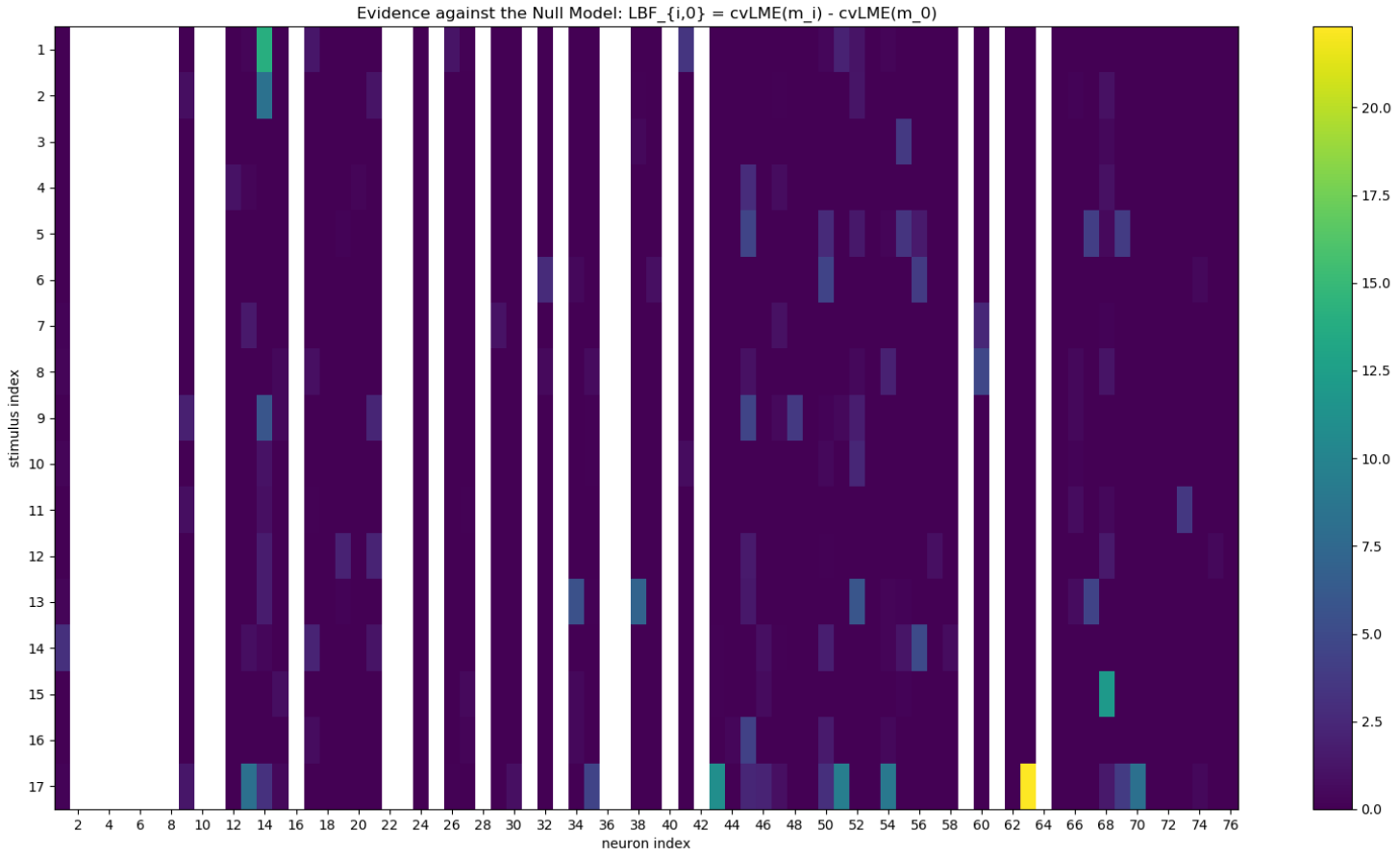


$$m_i: \; y_j = \beta_0 + \beta_1 s_{ij} + \varepsilon_{ij}$$

Q1: model selection tool

$$\mathrm{LBF}_{12} = \log \mathrm{BF}_{12} = \log \frac{p(y|m_1)}{p(y|m_2)} = \mathrm{LME}(m_1) - \mathrm{LME}(m_2)$$

Q1: model comparison



$$\mathrm{LBF}_{i,0} = \mathrm{cvLME}(m_i) - \mathrm{cvLME}(m_0)$$

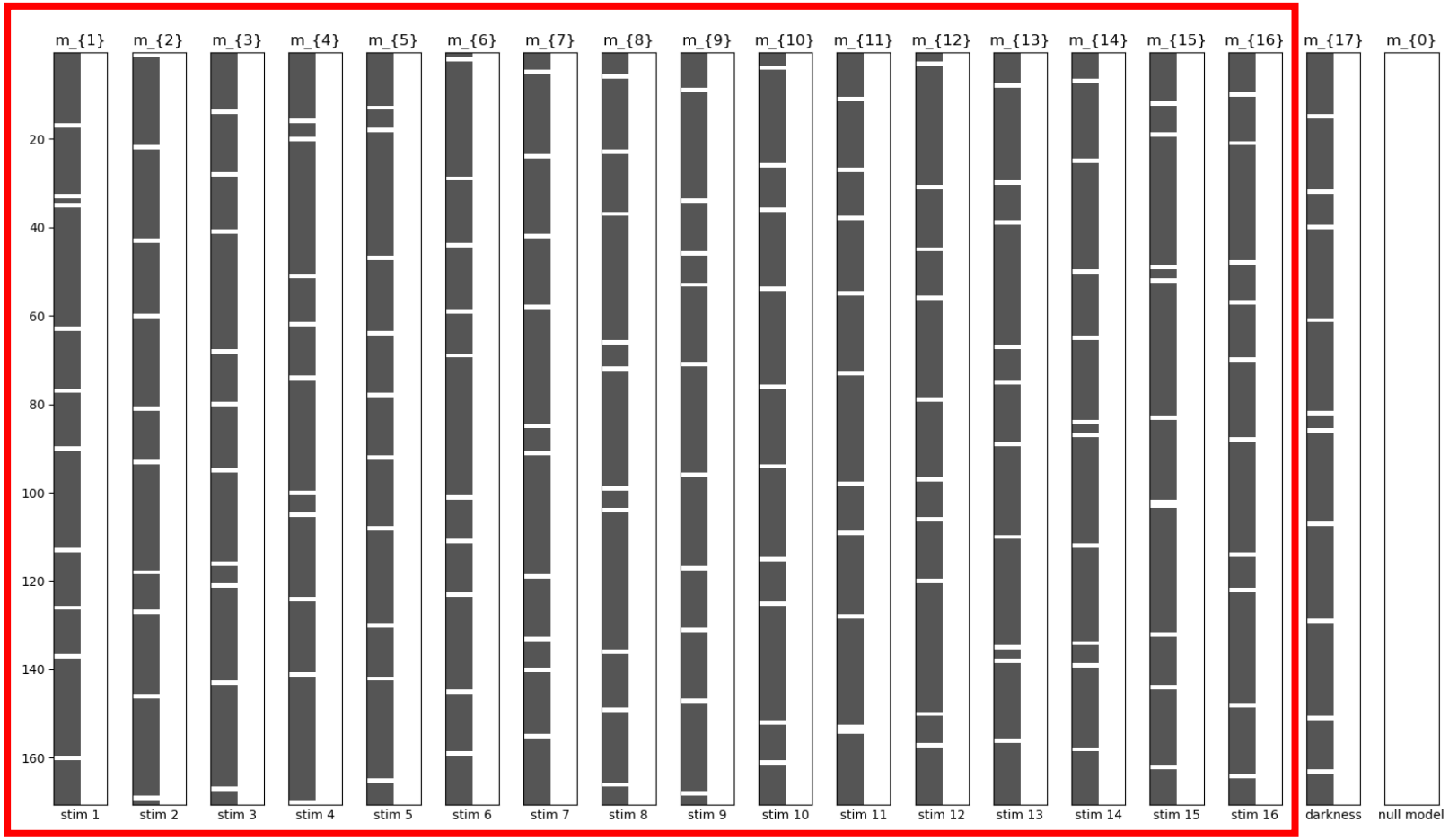
Research Question No. 2

„Does the neuron react to visual stimulation *per se*?“

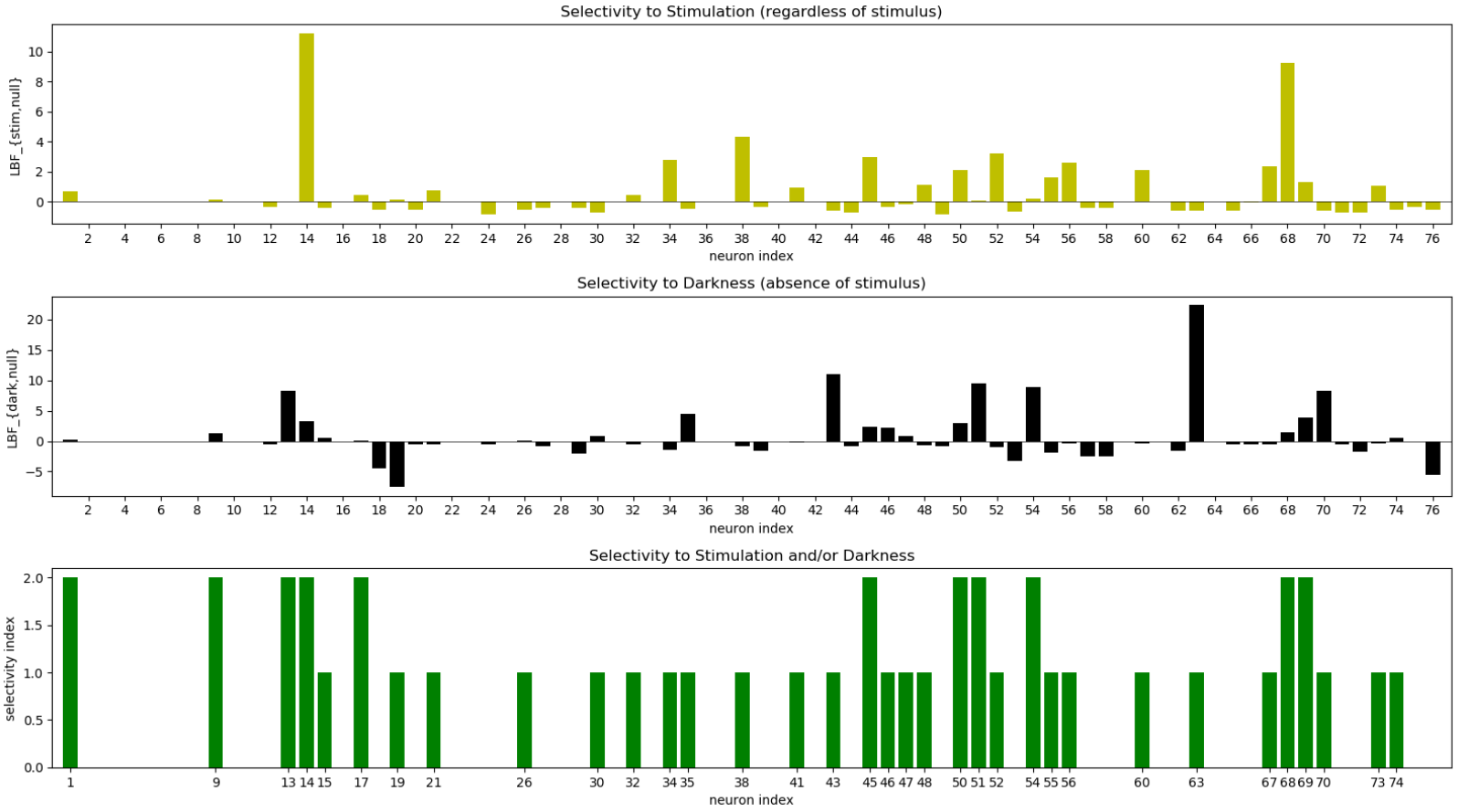
Q2: model selection tool

$$\mathrm{LFE}(f) = \log p(y|f) = \log \sum_{m \in f} p(y|m)\, p(m|f)$$

Q2: model space

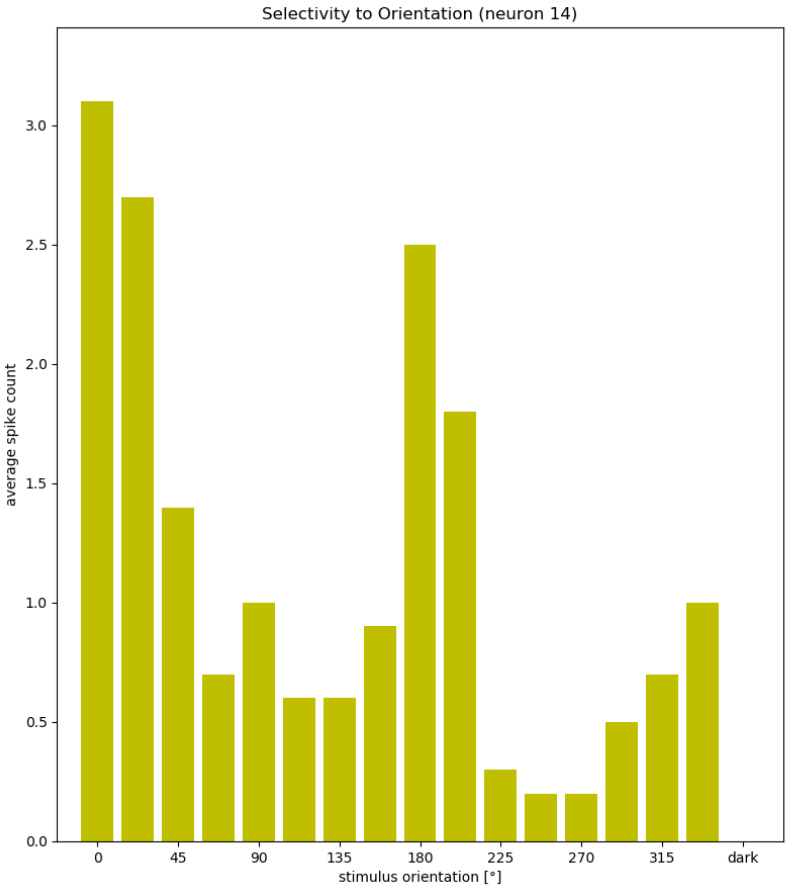


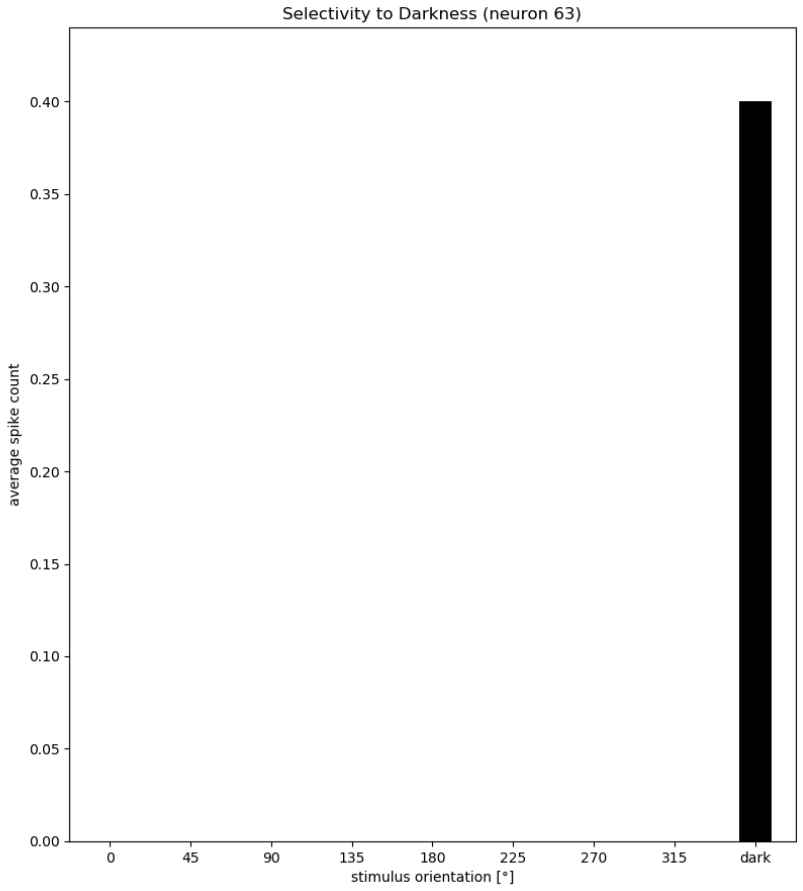
Q2: family comparison



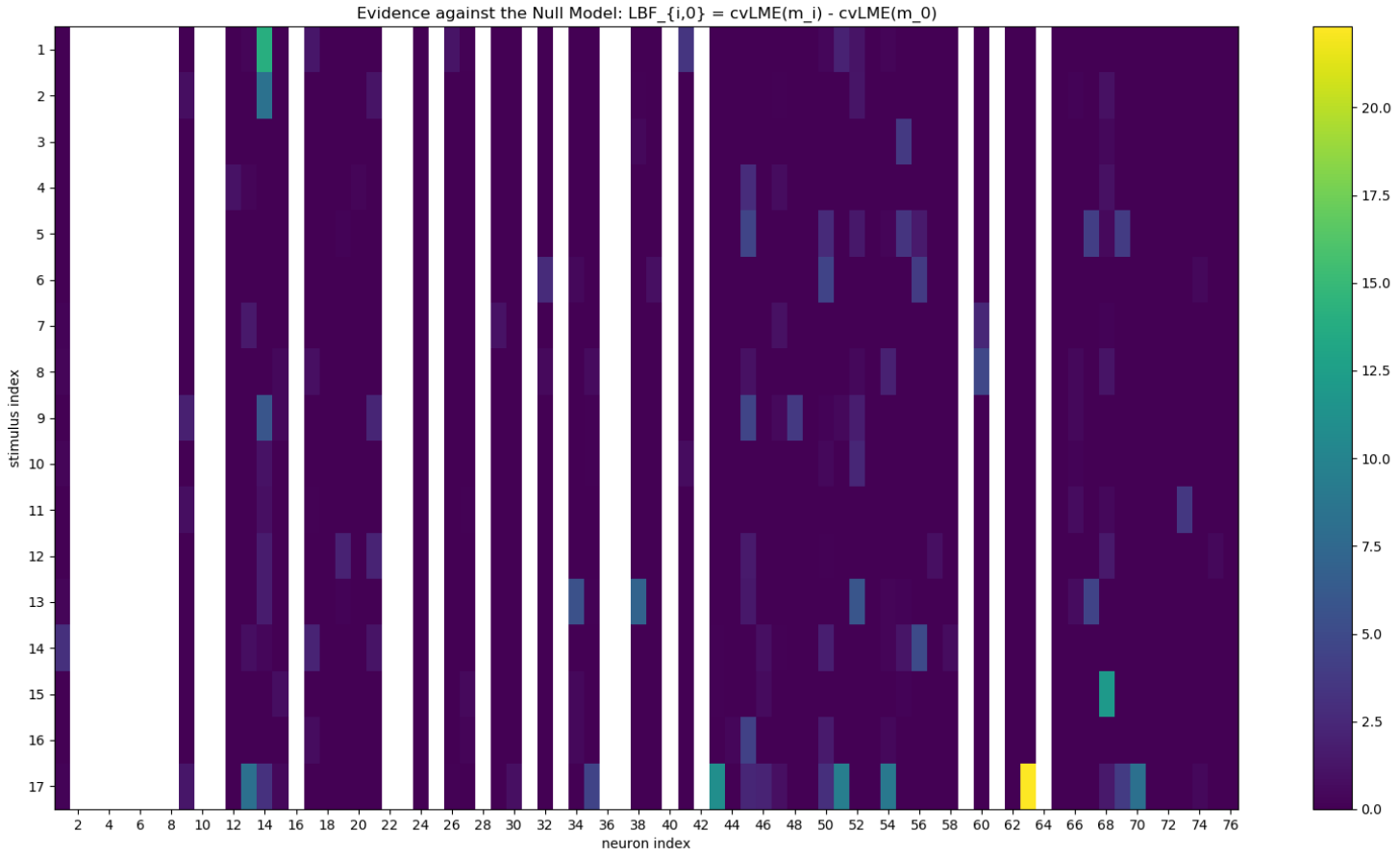
$$\mathrm{LBF}_{\mathrm{stim,null}} = \mathrm{LFE}(f_{1-16}) - \mathrm{cvLME}(m_0)$$
$$\mathrm{LBF}_{\mathrm{dark,null}} = \mathrm{cvLME}(m_{17}) - \mathrm{cvLME}(m_0)$$

Q2: data inspection





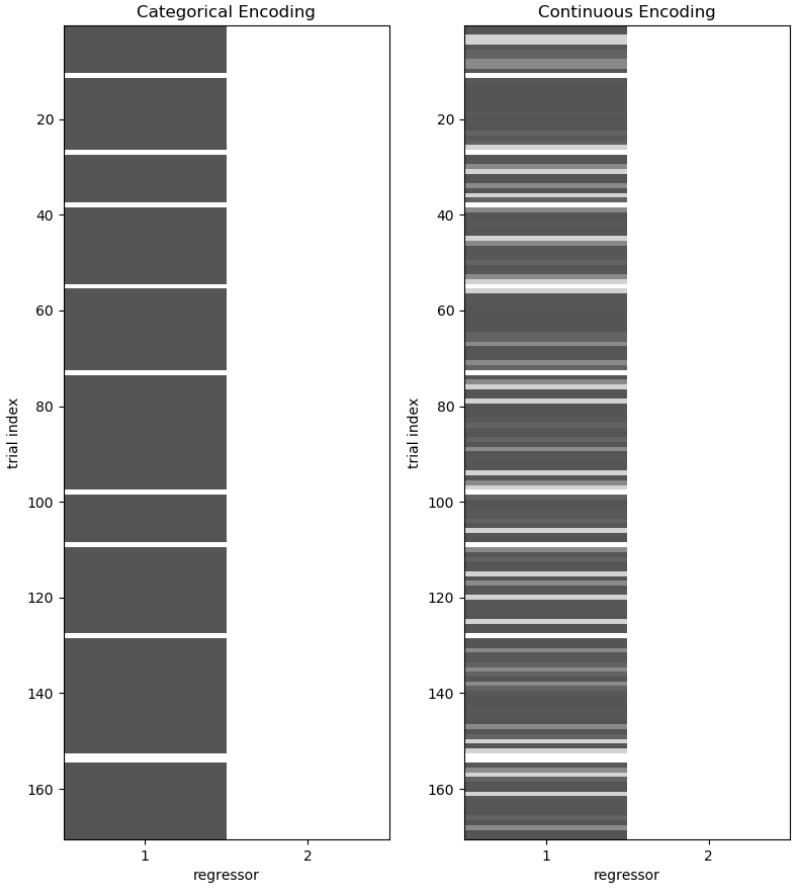
Q2: data inspection

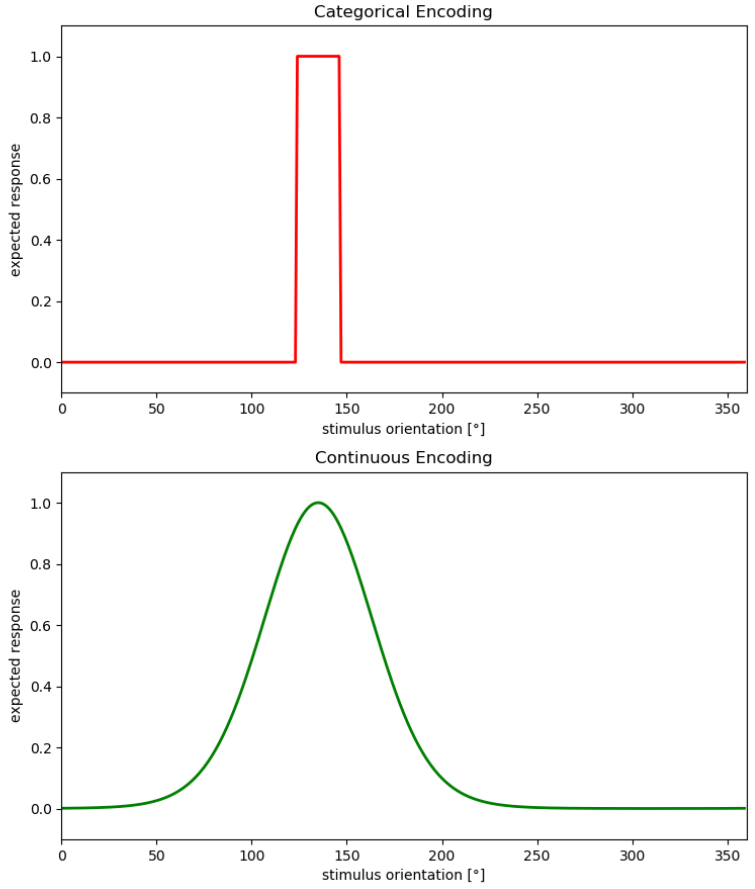


Research Question No. 3

„Does the neuron respond categorically or does it exihibit some tuning function?“

Q3: model space

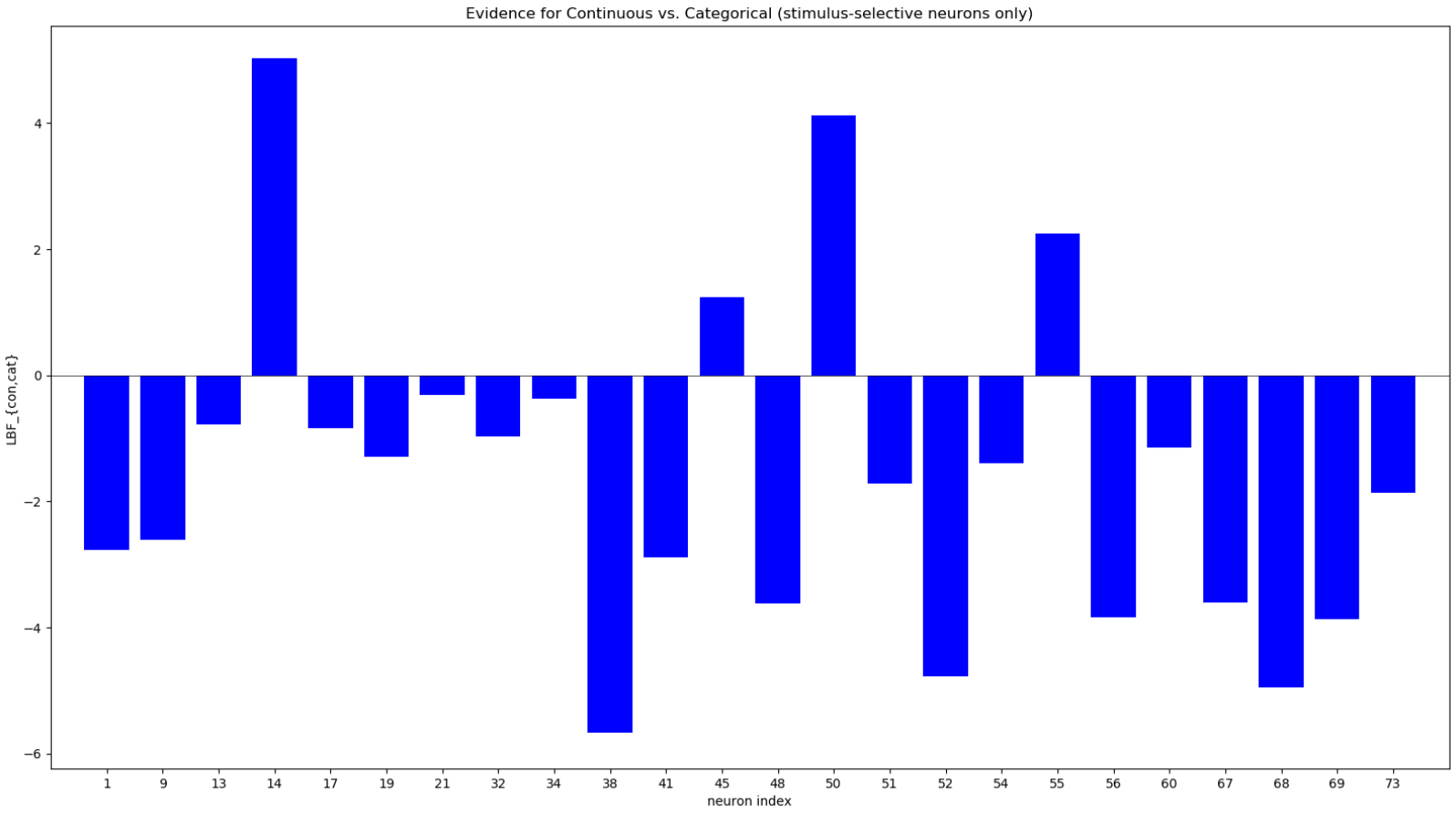




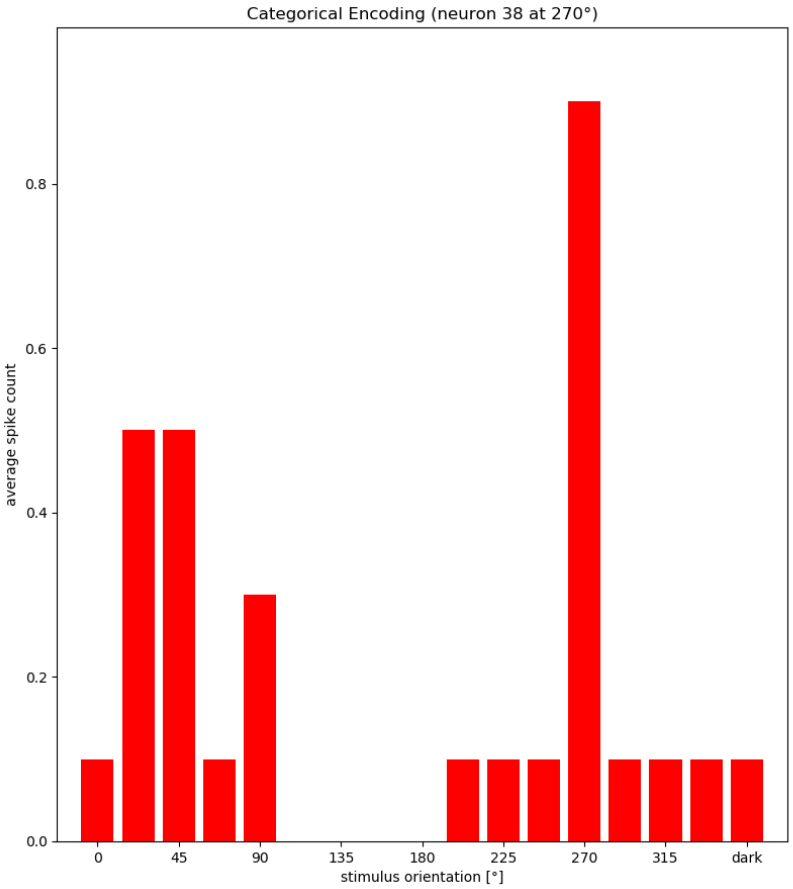
Q3: model selection tool

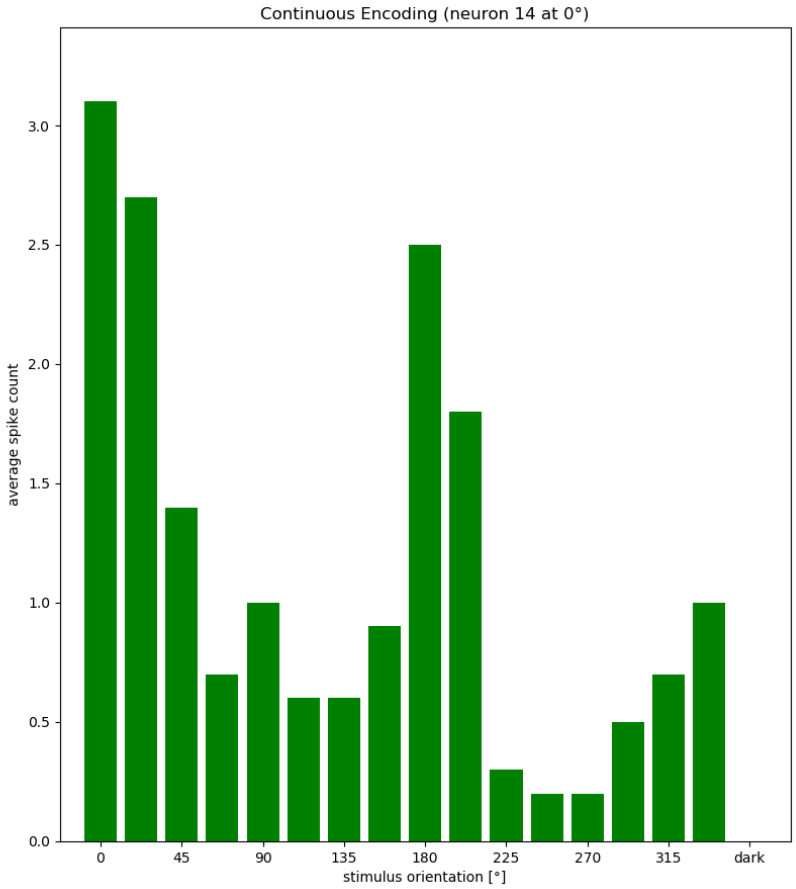
$$\mathrm{LBF}_{12} = \log \mathrm{BF}_{12} = \log \frac{p(y|m_1)}{p(y|m_2)} = \mathrm{LME}(m_1) - \mathrm{LME}(m_2)$$

Q3: model comparison



Q3: data inspection

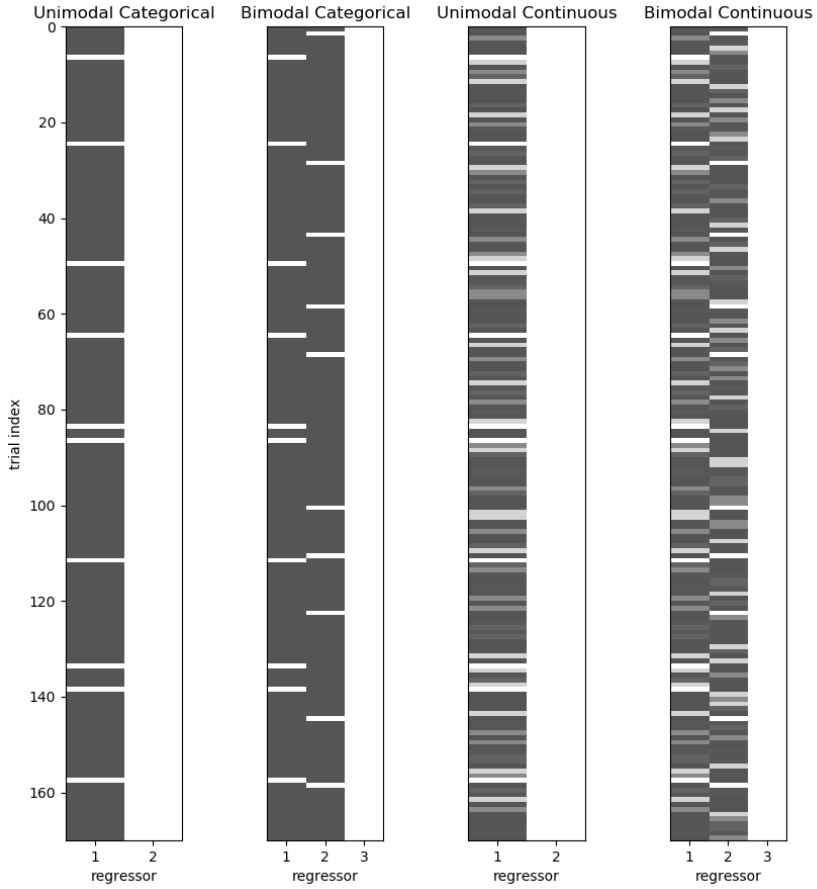


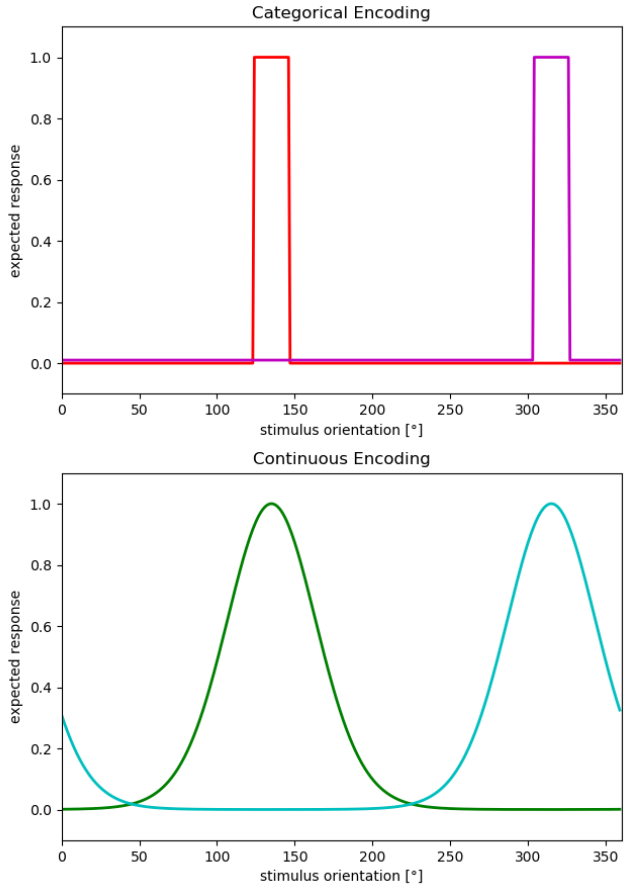


Research Question No. 4

„Does the neuron also respond to the exact opposite of the prefered direction?“

Q4: model space

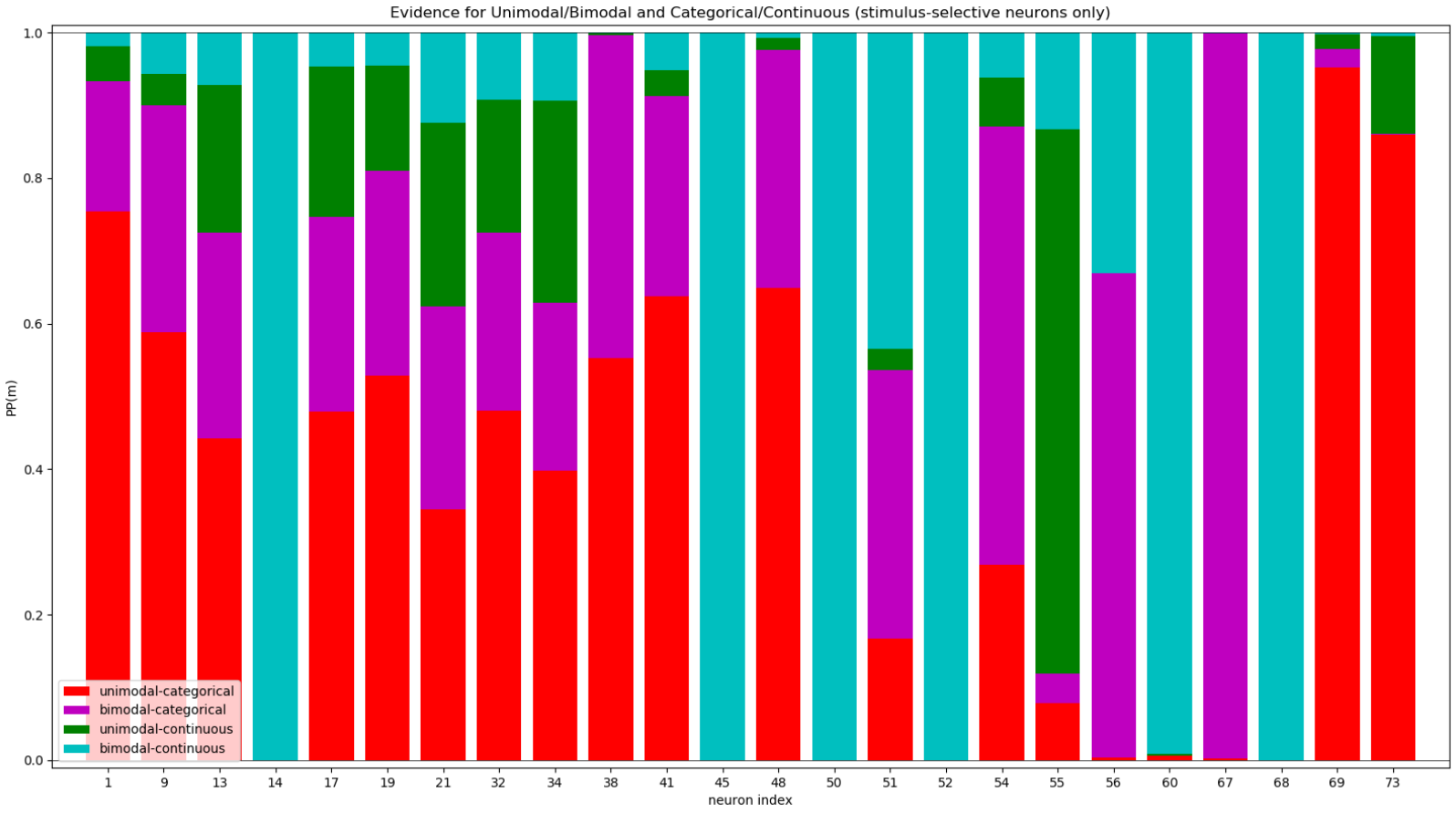




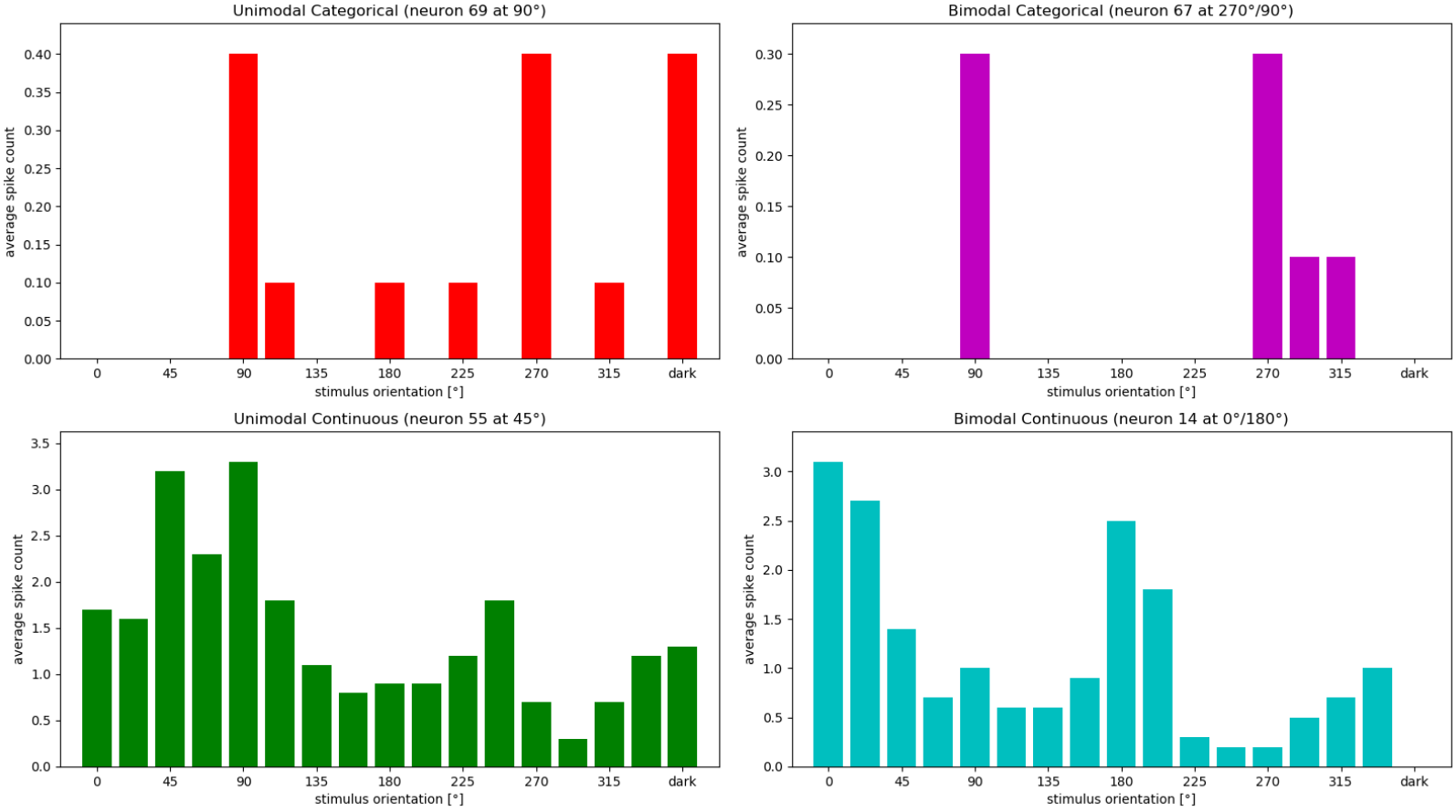
Q4: model selection tool

$$p(m_i|y) = \frac{p(y|m_i)\,p(m_i)}{\sum_{j=1}^{M} p(y|m_j)\,p(m_j)} = \frac{\exp[\mathrm{cvLME}(m_i)]\,p(m_i)}{\sum_{j=1}^{M} \exp[\mathrm{cvLME}(m_j)]\,p(m_j)}$$

Q4: model selection



Q4: data inspection



Possible Further Research Questions

„Does the neuron have a baseline response
or is it zero (i.e. no constant regressor)?“

„Are discrete spike counts normally distributed or
are they rather Poisson-distributed?“ (probably yes)

8.

What do I need to perform (Bayesian) model selection?

It depends! ;-)

If you have ...

... an objective function

$$\hat{\theta} = \arg\max f(y, \theta, m)$$

... a likelihood function

$$p(y|\theta, m)$$

... plus a prior distribution

$$p(\theta|m)$$

... plus an analytic posterior

$$p(\theta|y, m) \propto p(y|\theta, m)\, p(\theta|m)$$

then you can use ...

... any measure of fit

$$\mathrm{GoF} = g(y, \hat{y})$$

... classical information criteria

$$\mathrm{IC} = -2\,\mathrm{MLL} + f_{\mathrm{IC}}(n, k)$$

... the log model evidence

$$\mathrm{LME}(m) = \log p(y|m)$$

... the cross-validated log model evidence

$$\mathrm{cvLME}(m) = \sum_{i=1}^{S} \log p(y_i|y_{\neg i}, m)$$

Soch & Allefeld, JNeuMeth, 2018

9.

Practice Q&A

John-Dylan Haynes

Carsten Allefeld

Carsten Bogler

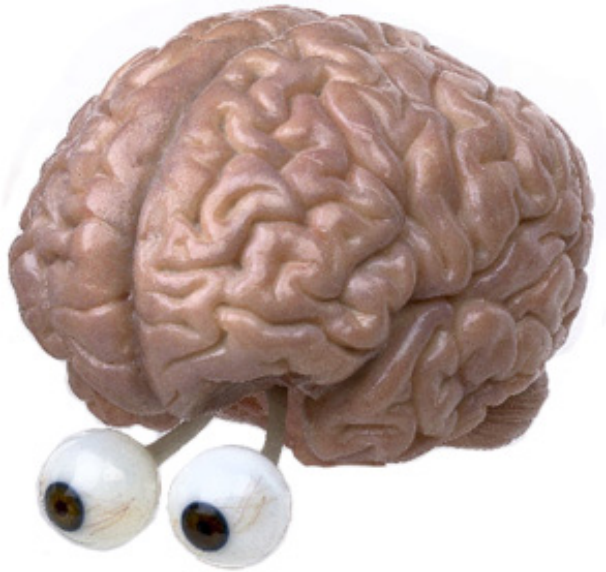
Achim Meyer

THANKS TO MY COLLABORATORS!

THANK YOU!

QUESTIONS?

joram.soch@bccn-berlin.de



Appendix

Model quality control for fMRI

	cvBMS	**cvBMA**	**MACS**
Poster	F1000 Research	OHBM 2017	OHBM 2018
Paper	NeuroImage	NeuroImage	JNeuMeth
Code	obsolete	obsolete	GitHub | Demo

E-Mail: joram.soch@bccn-berlin.de; WWW: orcid.org/0000-0002-8879-5666.

Derivatives of the LME

$$p(y|m) = \int p(y|\theta, m)\, p(\theta|m)\, \mathrm{d}\theta$$

$$p(m|y) = \frac{p(y|m)\, p(m)}{\sum_{j=1}^{M} p(y|m_j)\, p(m_j)}$$

$$\mathrm{LME}(m) = \log p(y|m)$$

$$p(f|y) = \frac{p(y|f)\, p(f)}{\sum_{k=1}^{F} p(y|f_k)\, p(f_k)}$$

$$p(y|f) = \sum_{m \in f} p(y|m)\, p(m|f)$$

$$\mathrm{LFE}(f) = \log p(y|f)$$

$$\mathrm{BF}_{12} = \frac{p(y|m_1)}{p(y|m_2)}$$

$$\mathrm{LBF}_{12} = \mathrm{LME}(m_1) - \mathrm{LME}(m_2)$$

$$p(m|Y) = \frac{p(Y|m)\, p(m)}{\sum_{j=1}^{M} p(Y|m_j)\, p(m_j)}$$

$$p(Y|m) = \prod_{i=1}^{N} p(y_i|m)$$

$$\log p(Y|m) = \sum_{i=1}^{N} \log p(y_i|m)$$

$$\mathrm{GBF}_{12} = \frac{p(Y|m_1)}{p(Y|m_2)}$$

$$\mathrm{LGBF}_{12} = \log p(Y|m_1) - \log p(Y|m_2)$$

Stephan et al., 2009; Penny et al., 2010; Penny, 2012; Soch et al., 2016; Soch et al., 2017; all NeuroImage

Accuracy & Complexity in the LME

$$p(\theta|y,m) = \frac{p(y|\theta,m)\,p(\theta|m)}{p(y|m)}$$

$$p(y|m) = \frac{p(y|\theta,m)\,p(\theta|m)}{p(\theta|y,m)}$$

$$\log p(y|m) = \log p(y|\theta,m) - \log \frac{p(\theta|y,m)}{p(\theta|m)}$$

$$\log p(y|m) = \int p(\theta|y,m)\,\log p(y|\theta,m)\,\mathrm{d}\theta - \int p(\theta|y,m)\,\log \frac{p(\theta|y,m)}{p(\theta|m)}\,\mathrm{d}\theta$$

$$\mathrm{LME} = \langle \log p(y|\theta,m) \rangle_{p(\theta|y,m)} - \mathrm{KL}\left[p(\theta|y,m)\,||\,p(\theta|m)\right]$$

Penny et al., NeuroImage, 2007; Soch et al., NeuroImage, 2016

The pragmatics of model selection

- model dilution

  → model families
- sequential model selection

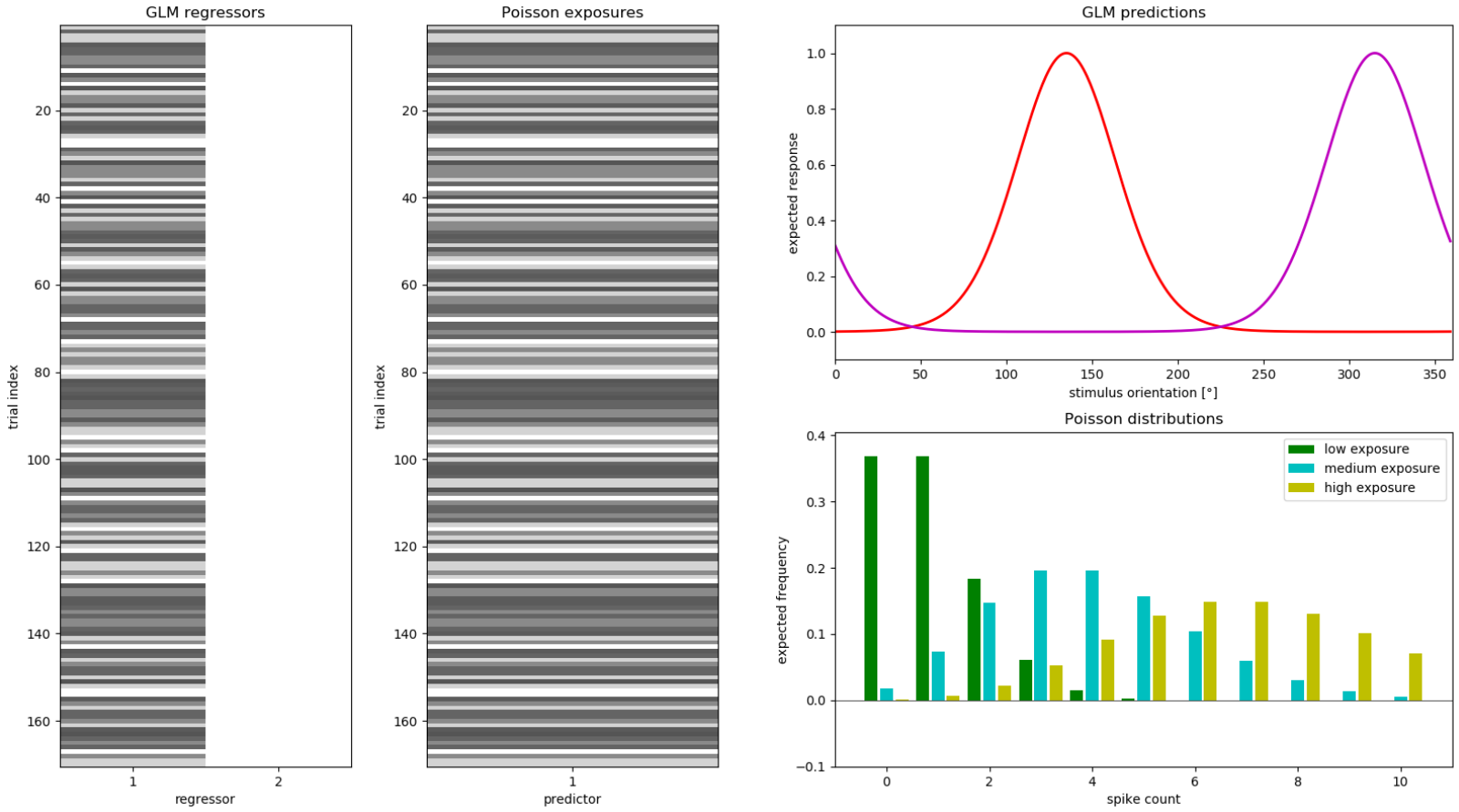
  → assume independence
- methodological control (Q3) vs.

  scientific inference (Q1, Q2, Q4)

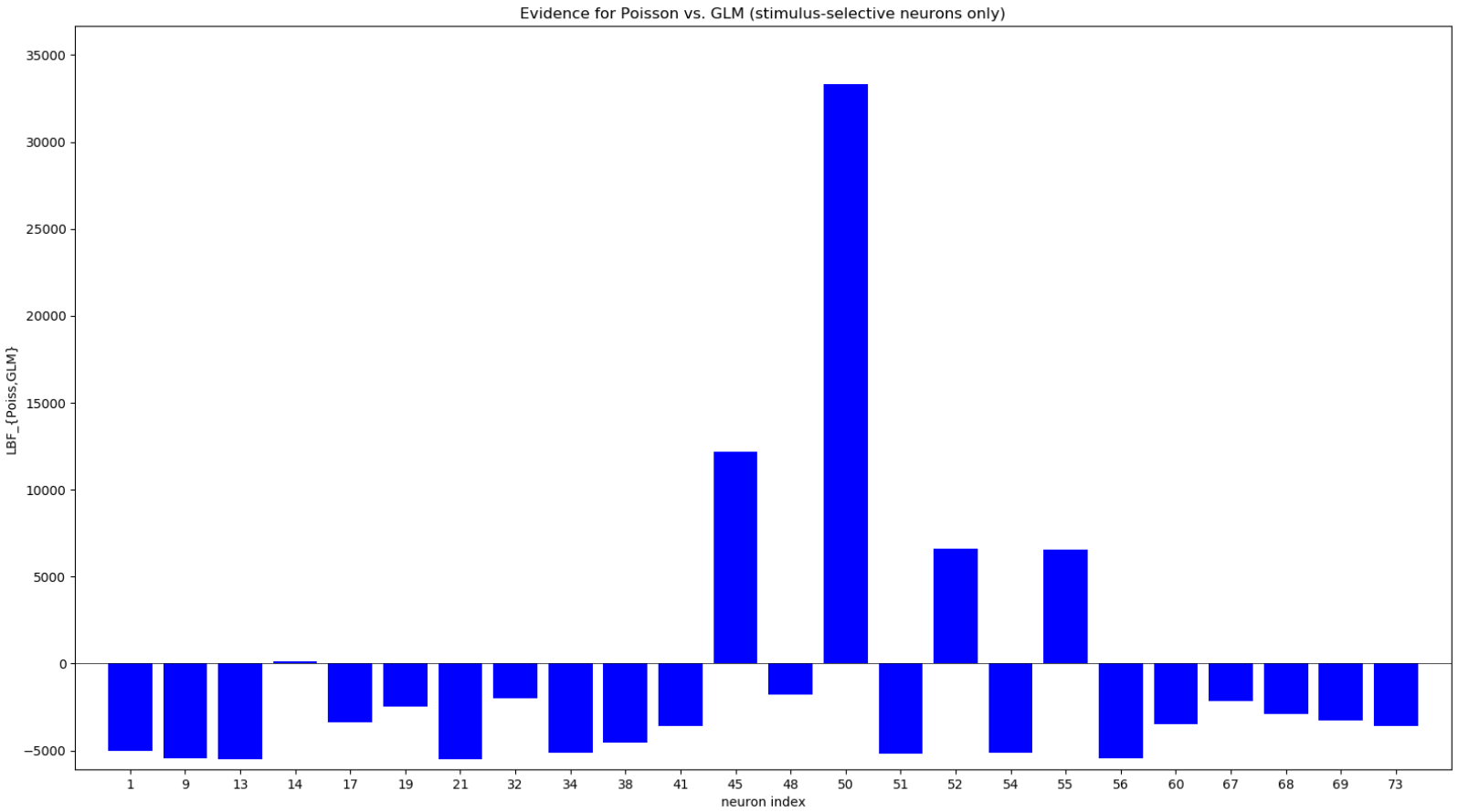
Research Question No. 5

„Are spike counts normally distributed
or are they rather Poisson-distributed?“

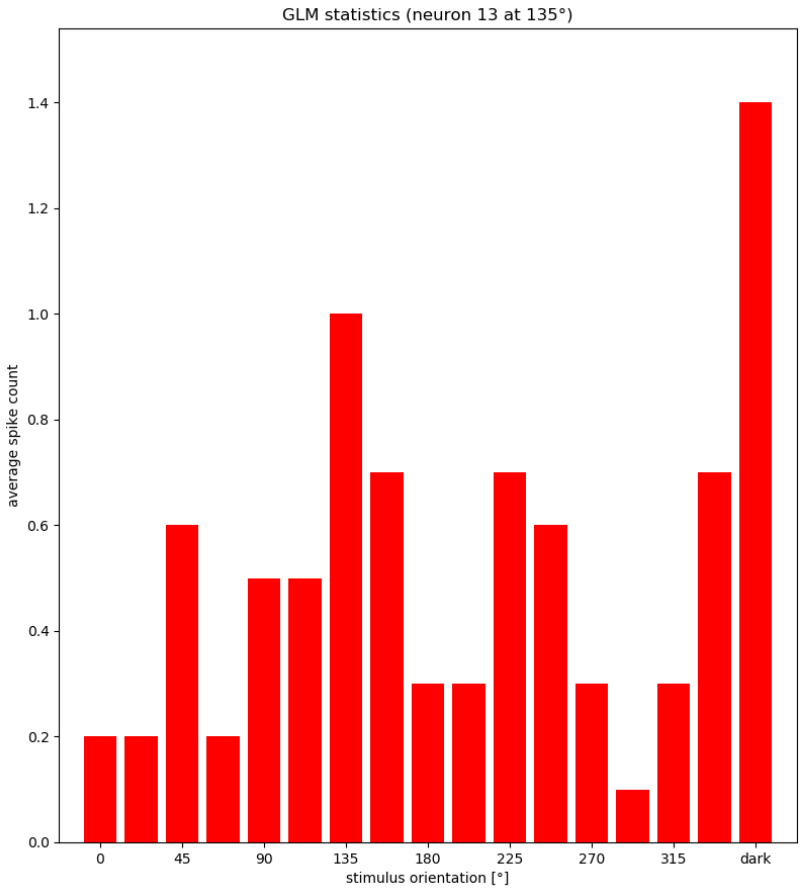
Q5: model space

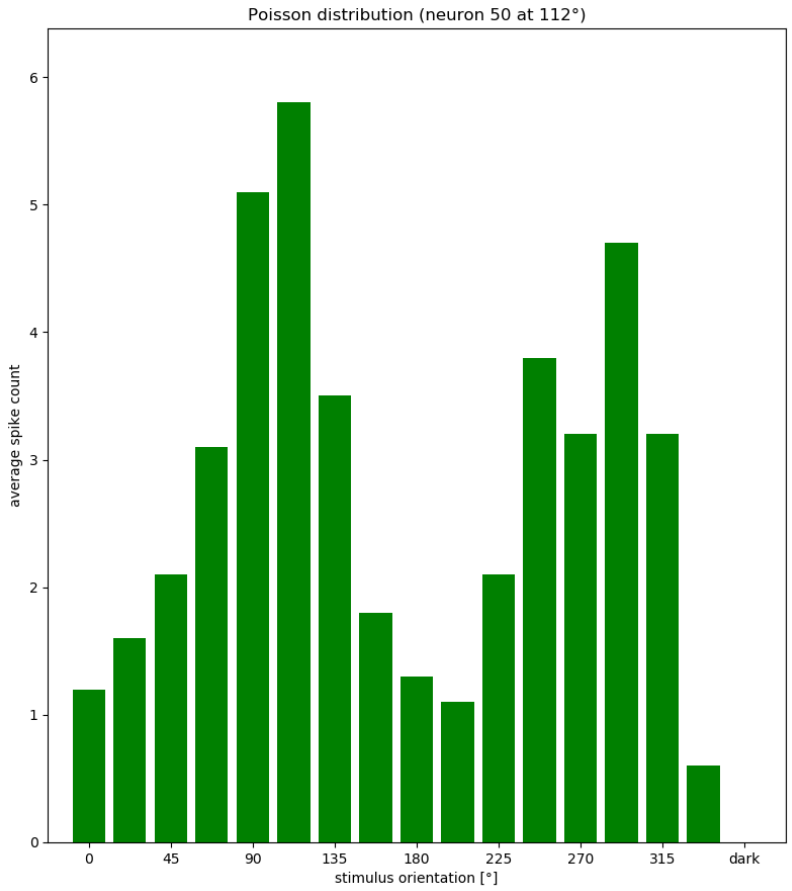


Q5: model comparison



Q5: data inspection





Q5: data inspection