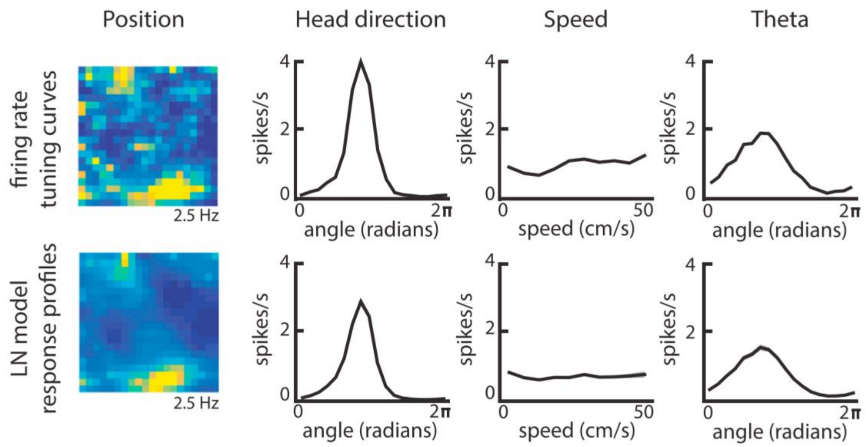
Generalized linear models



$$\mathbf{W}\mathbf{x} = \mathbf{W}^P\mathbf{x}_P + \mathbf{W}^H\mathbf{x}_H + \mathbf{W}^S\mathbf{x}_S + \mathbf{W}^T\mathbf{x}_T$$

Machine Learning from Scratch seminar

John Vastola

2/20/24

Q. What is **Machine Learning from Scratch**?

A. Occasional HMS seminar focused on learning about ML fundamentals.
Topics might be related to neuro research, or might just be cool!

`https://github.com/DrugowitschLab/ML-from-scratch-seminar`

Seminars happen over 2 evenings:

first is **theory-focused**
second is **coding-focused**

~ **Four** seminars planned for Spring 2024:

Feb | GLMs (w/ Kiah Hardcastle)
Mar | RNNs (w/ Siyan Zhou)
Apr | TBD
May | TBD

Useful resources for learning about GLMs

Tutorials

2016 SFN tutorial on GLMs by Jesse Kaminsky and Jonathan Pillow
https://github.com/pillowlab/GLMspiketraintutorial_python

Neuromatch Academy GLM tutorial by Fiquet et al.
https://compneuro.neuromatch.io/tutorials/W1D3_GeneralizedLinearModels/student/W1D3_Tutorial1.html

GLM_Tensorflow2 repository by Shih-Yi Tseng
https://github.com/sytseng/GLM_Tensorflow_2

See this session’s GitHub page for more!

Useful resources for learning about GLMs

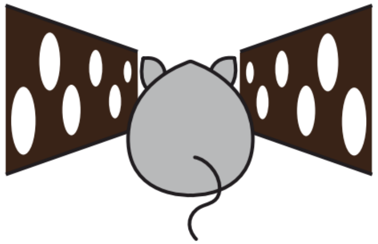
Example papers

[2014 Park et al.] “Encoding and decoding in parietal cortex during sensorimotor decision-making”
application to data from macaque lateral intraparietal area
https://www.nature.com/articles/nn.3800

[2017 Hardcastle et al.] “A Multiplexed, Heterogeneous, and Adaptive Code for Navigation in Medial Entorhinal Cortex”
application to data from mouse medial entorhinal cortex
https://doi.org/10.1016/j.neuron.2017.03.025

[2022 Tseng and Chettih et al.] “Shared and specialized coding across posterior cortical areas for dynamic navigation decisions”
application to data from mouse posterior cortex
https://doi.org/10.1016/j.neuron.2022.05.012

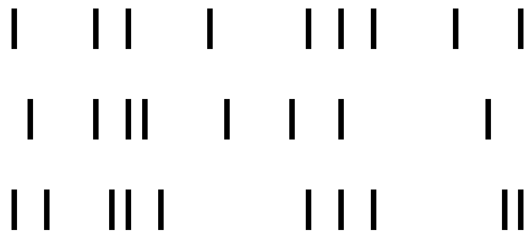
See this session’s GitHub page for more!



perception
beliefs
actions

$p(\mathbf{x}|\mathbf{r})$ e.g., predicting choices from neural activity

e.g., predicting activity from behavior $p(\mathbf{r}|\mathbf{x})$



Q. How do we estimate p(r | x) using a mix of neural and behavioral data?

Statistical models
"Just fit a line!"

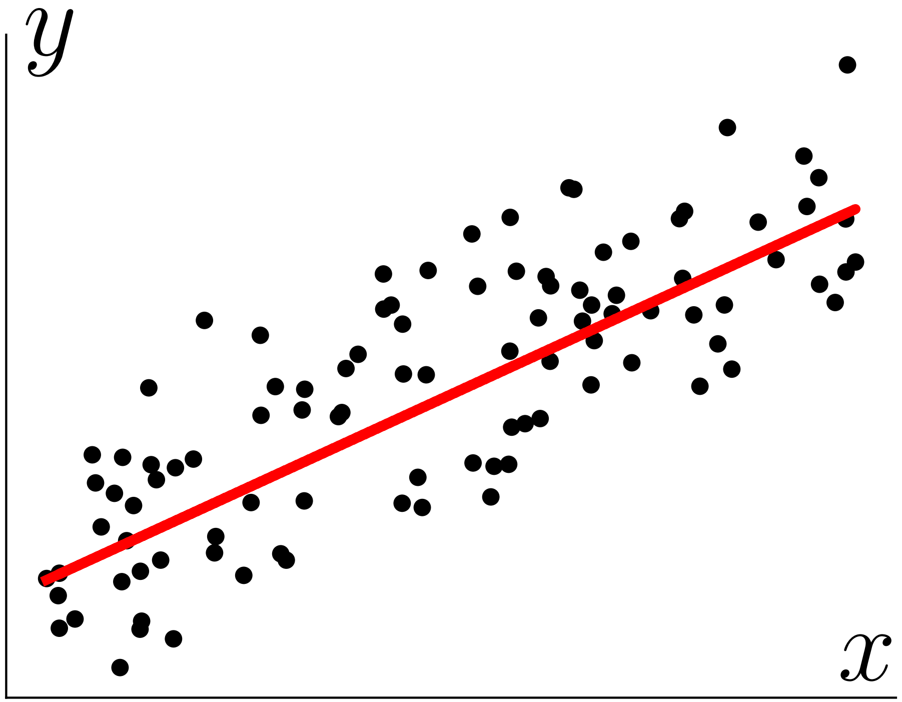
GLMs

RNNs
(next
session!)

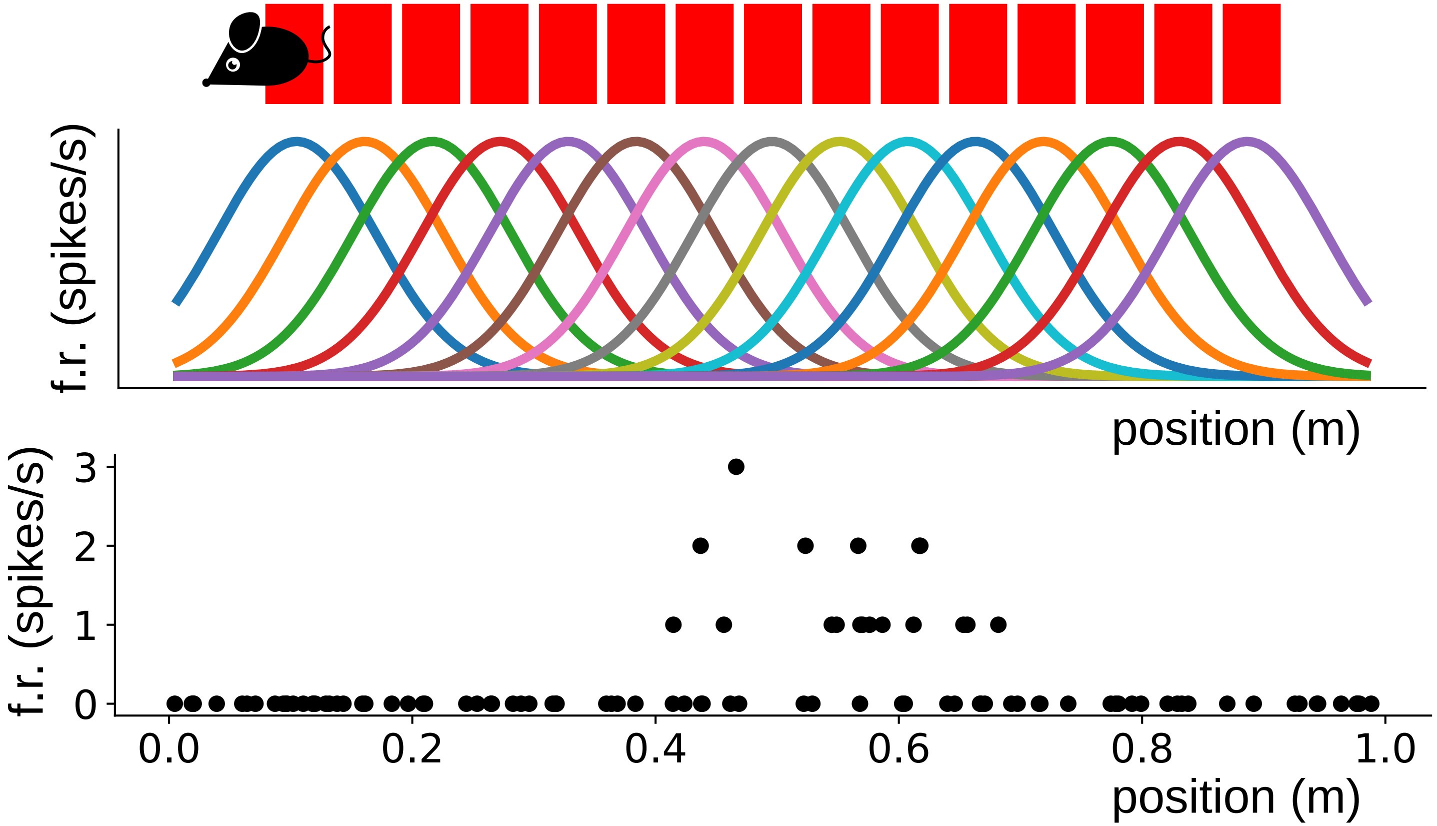
Biophysical models
"All details matter!"

increasing sophistication / realism

Lines are simple + interpretable…



…but not necessarily a good fit for neural data!



Solution

Need to add nonlinearity to account for, e.g., nonnegativity.

Also need to allow for non-Gaussian (in particular, Poisson-like) noise.

GLMs from a generative modeling perspective

noise model

$$\mathbf{x} \xrightarrow[\text{}]{\text{linear filter}} \mathbf{Wx} \xrightarrow{\text{apply nonlinearity}} f(\mathbf{Wx}) \xrightarrow{p(\mathbf{r}|f(\mathbf{Wx}))} \mathbf{r}$$

input
(percepts, beliefs, actions…)

distribution
parameters

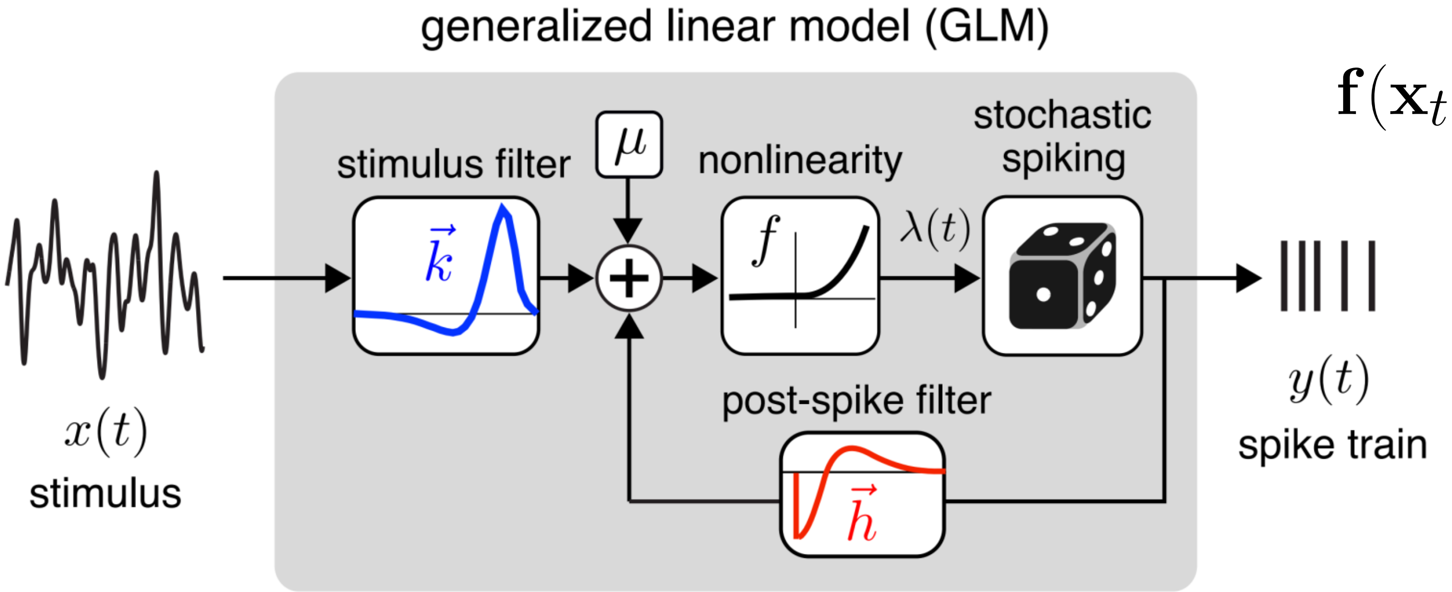
output
(spikes, Ca traces,…)

Examples:

- **linear regression** $\quad f(\mathbf{z}) = \mathbf{z} \quad p(\mathbf{r}|\boldsymbol{\mu}) = \mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I})$
- **logistic regression** $\quad f(\mathbf{z}) = 1/(1 + e^{-\mathbf{z}}) \quad p(r|p) = p^r(1-p)^{1-r}$
- **Poisson regression** $\quad f(\mathbf{z}) = \exp(\mathbf{z}) \quad p(\mathbf{r}|\boldsymbol{\mu}) = \mathcal{P}(\boldsymbol{\mu})$

Note for math lovers: These approaches can be unified using the idea of exponential family distributions.
See Murphy, “Probabilistic Machine Learning” Book 1, Ch. 12. https://probml.github.io/pml-book/

Neuroscience GLMs in practice: linear-nonlinear-Poisson models



$$\mathbf{f}(\mathbf{x}_t, \{\mathbf{y}_{t'}\}) = \exp[\ \mathbf{W}\mathbf{x}_t + \sum_{t'<t} \mathbf{W}^{t'} \mathbf{y}_{t'}\ ]$$

$$\mathbf{y}_t \sim \mathcal{P}(\ \mathbf{f}(\mathbf{x}_t, \{\mathbf{y}_{t'}\})\Delta t\ )$$

Poisson regression (i.e., using an independent Poisson noise model) most commonly used GLM.

The canonical choice of **f**() is exponential (see exp. family math) but others sometimes used.

Using recent spiking history as input is optional, but helps capture, e.g., bursting, refractory period…

Weber and Pillow, *Neur. Comp.* (2017)

Poisson regression details: likelihood and gradients

Log-likelihood

$$\log p = \sum_{n,i} r_i^{(n)} \sum_j W_{ij} x_j^{(n)} - f(\sum_k W_{ik} x_k^{(n)}) \Delta t$$

n: samples, i: neurons, j: inputs

Gradient wrt weights

$$\frac{\partial \log p}{\partial W_{ij}} = \sum_n \left[r_i^{(n)} - f(\sum_k W_{ik} x_k^{(n)}) \Delta t \right] x_j^{(n)}$$

$$= \sum_n \left[r_i^{(n)} - \hat{r}_i^{(n)} \right] x_j^{(n)}$$

Note simplicity of expression:
the nth sample contributes a term (true – estimated)*input. Two-factor learning rule!

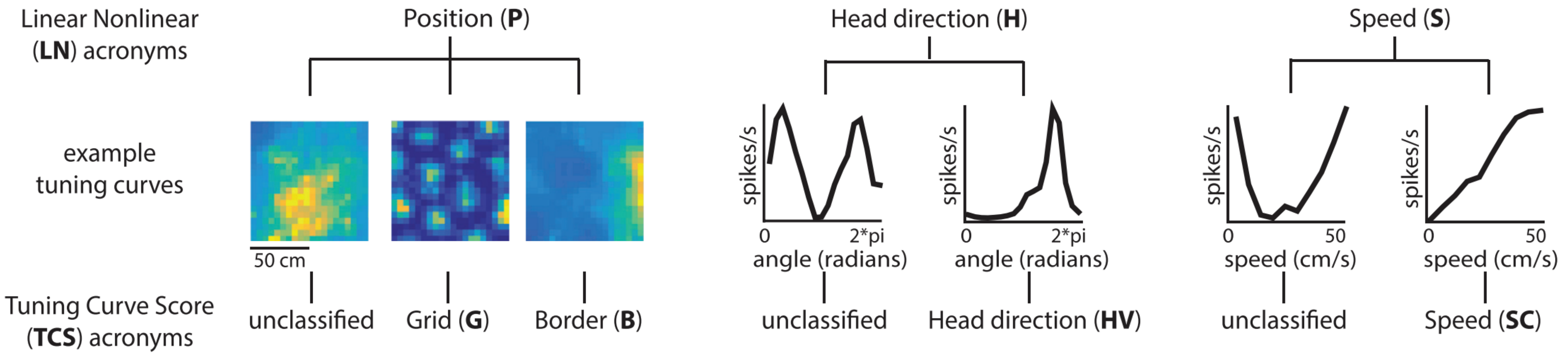
This holds for more general GLMs with canonical activation functions too. See Murphy.

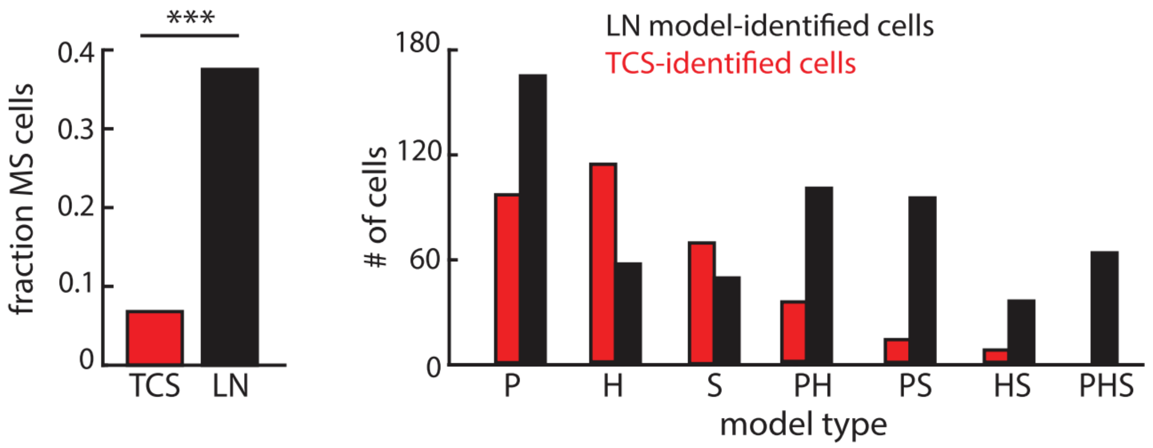
Example 1: Fitting data from mouse medial entorhinal cortex

Hardcastle et al., *Neuron* (2017)

“A Multiplexed, Heterogeneous, and Adaptive Code
for Navigation in Medial Entorhinal Cortex”

Example 1: Fitting data from mouse medial entorhinal cortex

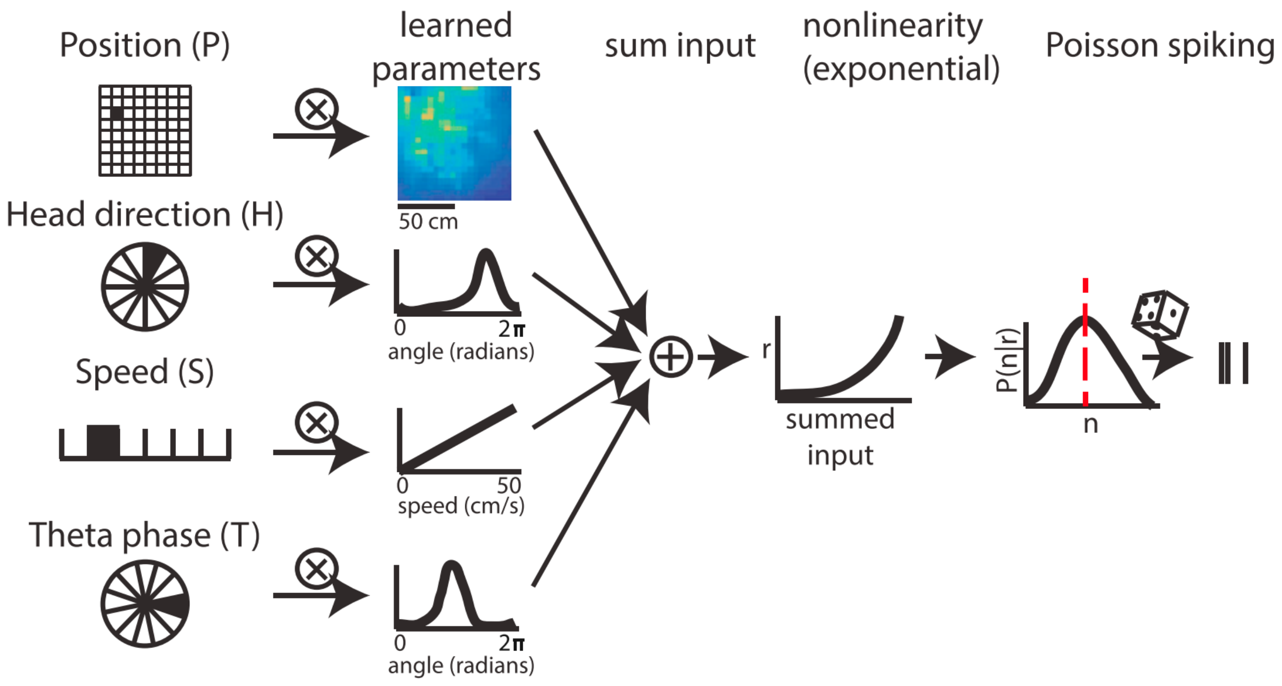


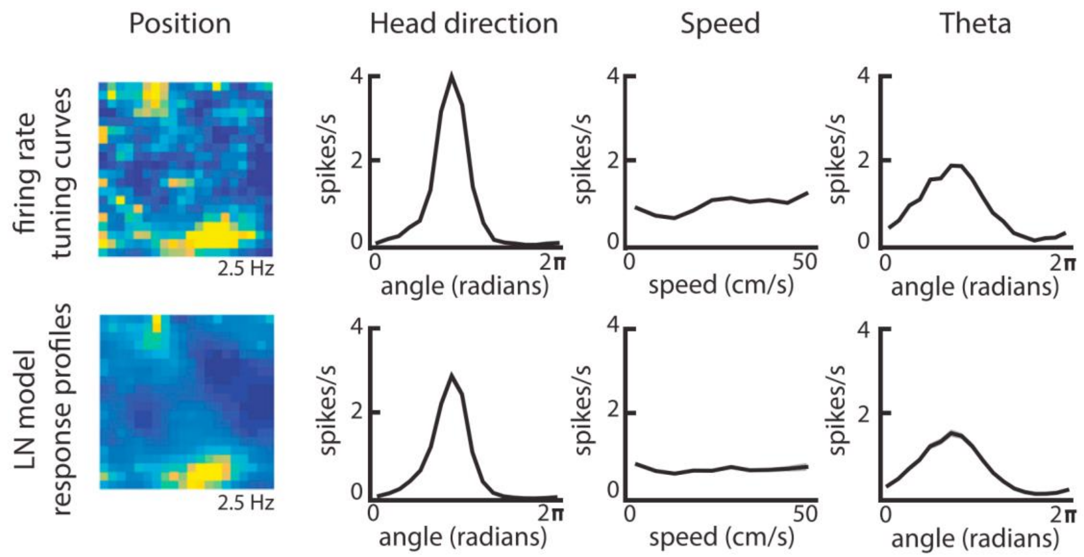


Characterizing cells via GLMs works better!

Hardcastle et al., *Neuron* (2017)

Example 1: Fitting data from mouse medial entorhinal cortex





$$p(\mathbf{r}|\mathbf{x}) = \mathcal{P}(e^{\mathbf{W}\mathbf{x}}\Delta t)$$

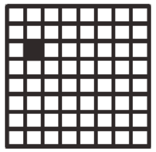
Poisson observations,
exponential activation function

$$\mathbf{W}\mathbf{x} = \mathbf{W}^P\mathbf{x}_P + \mathbf{W}^H\mathbf{x}_H + \mathbf{W}^S\mathbf{x}_S + \mathbf{W}^T\mathbf{x}_T$$

position head-direction speed theta phase

Example 1: Fitting data from mouse medial entorhinal cortex

Position (P)



Head direction (H)

Speed (S)

Theta phase (T)

Important fitting detail:
Since behavior encodings are **'one-hot'**, need to regularize objective with smoothness penalty that ensures firing rates for nearby bins are related:

$$\beta_P \sum_{i,j} \frac{(W^P_{ij} - W^P_{i,j+1})^2}{2} + \beta_H \sum_{i,j} \frac{(W^H_{ij} - W^H_{i,j+1})^2}{2}$$
$$+\beta_S \sum_{i,j} \frac{(W^S_{ij} - W^S_{i,j+1})^2}{2} + \beta_T \sum_{i,j} \frac{(W^T_{ij} - W^T_{i,j+1})^2}{2}$$

Beta coefficients control strength of smoothness penalty for each variable.

Example 1 Takeaways

- GLMs provide a **statistically unbiased*** way to characterize tuning of neurons. Better way to characterize MEC cells than with grid scores, border scores, etc.

  Caveat: depends somewhat on state representation used as input.
  Hardcastle et al. use maximally expressive one-hot behavior encodings to avoid issues.

- Regularization is important for avoiding artifacts since data is limited. Without it, would tend to get weird/discontinuous tuning maps.

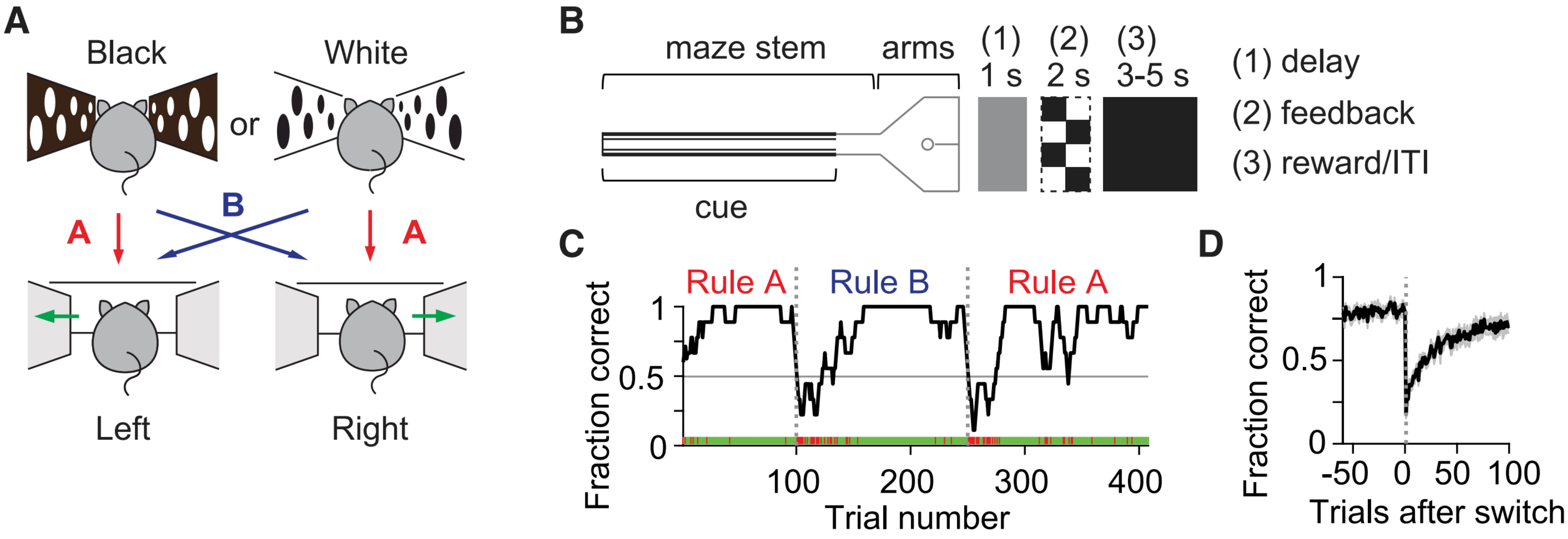
  Caveat: as we will see, can also address smoothness using a different state representation.

Example 2: Fitting data from mouse posterior cortex

Tseng and Chettih et al., *Neuron* (2022)

“Shared and specialized coding across posterior cortical areas for dynamic navigation decisions”

Example 2: Fitting data from mouse posterior cortex

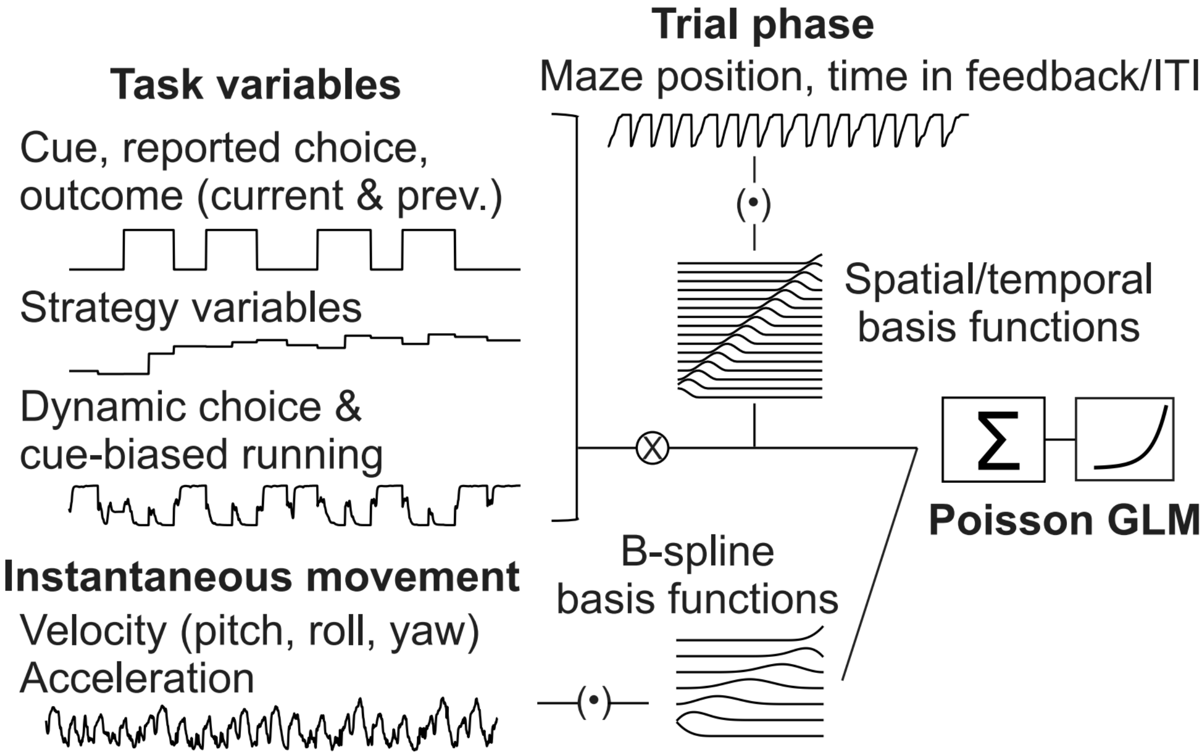


GLMs can also be used to characterize tuning in **dynamic tasks with nontrivial structure**!

See also tutorial slides by Shih-Yi at:
https://github.com/sytseng/GLM_Tensorflow_2

Tseng and Chettih et al., *Neuron* (2022)

Example 2: Fitting data from mouse posterior cortex



Much more complex for a few reasons:

1. Multiple categories of variables are relevant.

Dynamic navigation involves tracking:
Movement (e.g., velocity), time (e.g., task phase), belief (go left or right?), strategy (e.g., be lazy?) …

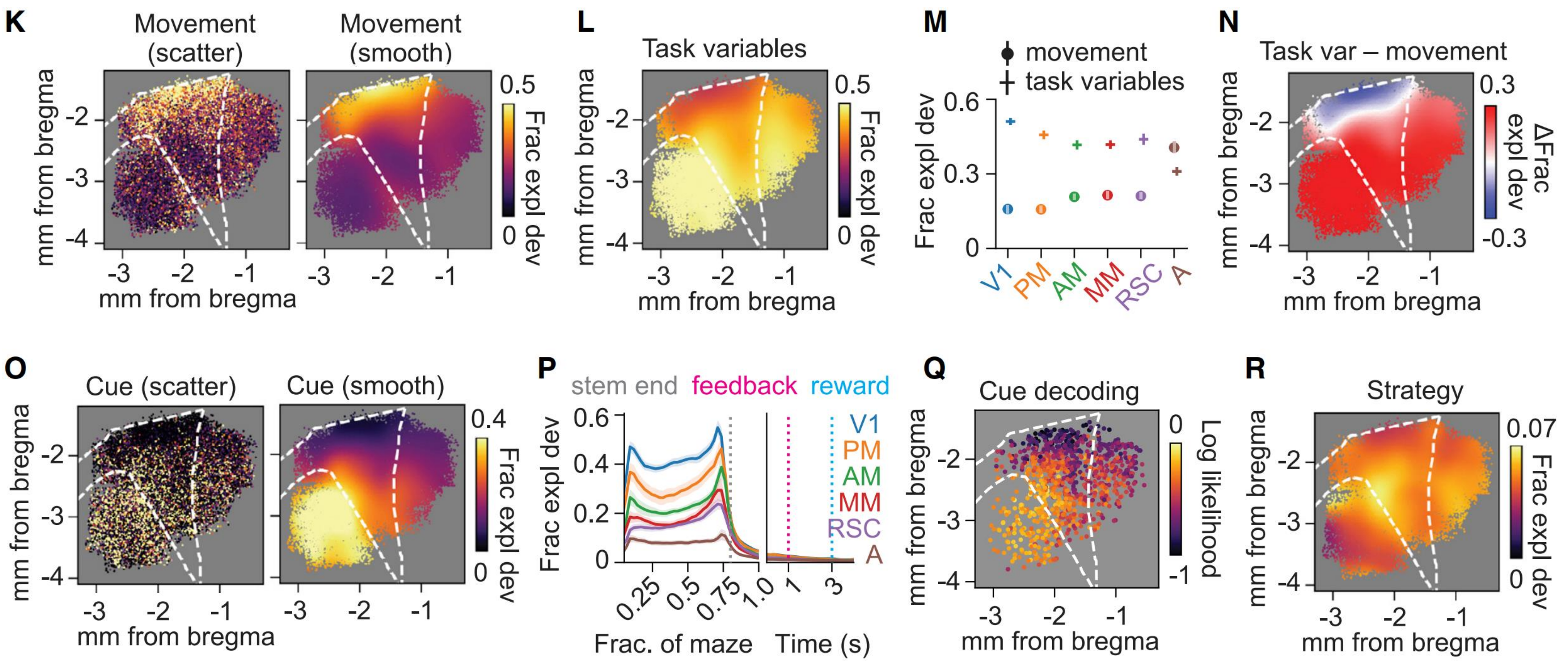
2. Time-dependence of firing rates is important.

e.g., evidence accumulation occurs over time, address using temporal basis functions

3. Parameter smoothness via state representation.

address using smooth basis functions instead of via regularization

Example 2: Fitting data from mouse posterior cortex



Example 2: Fitting data from mouse posterior cortex

Important fitting detail:
Number of parameters is HUGE (~ > 1000 per neuron), so model highly underconstrained.
Tseng et al. used a group lasso penalty to regularize the model:

$$\lambda \sum_i \sqrt{g_i} \|\mathbf{w}_i\|_2$$

Idea: "encourages sparsity between tuning to different variables,
but non-sparse L2 regularization on the within-variable bases"

Can also use some combination of:

$$\sum_j \frac{w_j^2}{2}$$

encourages small magnitudes

$$\sum_j |w_j|$$

encourages sparsity

Example 2: Fitting data from mouse posterior cortex

Q. How do we measure goodness-of-fit?

For Poisson regression, better to use “Poisson deviance” than squared error.

$$D_{model} = 2\sum_n \left[\, y_n \left(\mathbf{w}^T\mathbf{x}_n + w_0 - \log\langle y\rangle\right) - \left(e^{\mathbf{w}^T\mathbf{x}_n + w_0} - \langle y\rangle\right) \right]$$

Can also consider a quantity analogous to “variance explained”: *deviance* explained!

$$R^2_{GLM} := \frac{D_{model}}{D_{tot}} \geq 0$$

(see lecture notes for more details)

Idea: Compare fitted model to two extremes. One extreme: only fit mean of data. (null model)
Other extreme: # parameters = # data points (saturated model)

Example 2: Fitting data from mouse posterior cortex

Q. How do we choose hyperparameters (e.g., regularization strengths)?

Can't just check which parameter fits better on test set!
(because then you're using the test set as a training set…).

Instead, can use a strategy like k-fold cross validation.
Split the training data up into pieces, and test model fits on one piece against fits on other pieces.

Example 2 Takeaways

- Complex state representations allow GLMs to capture complicated spiking dynamics in a task that mixes dynamic navigation and decision-making.

  On the other hand, additional model complexity has predictable tradeoffs,
  re: interpretability, time required to fit model, etc.

- Very large number of parameters means careful model selection is crucial.

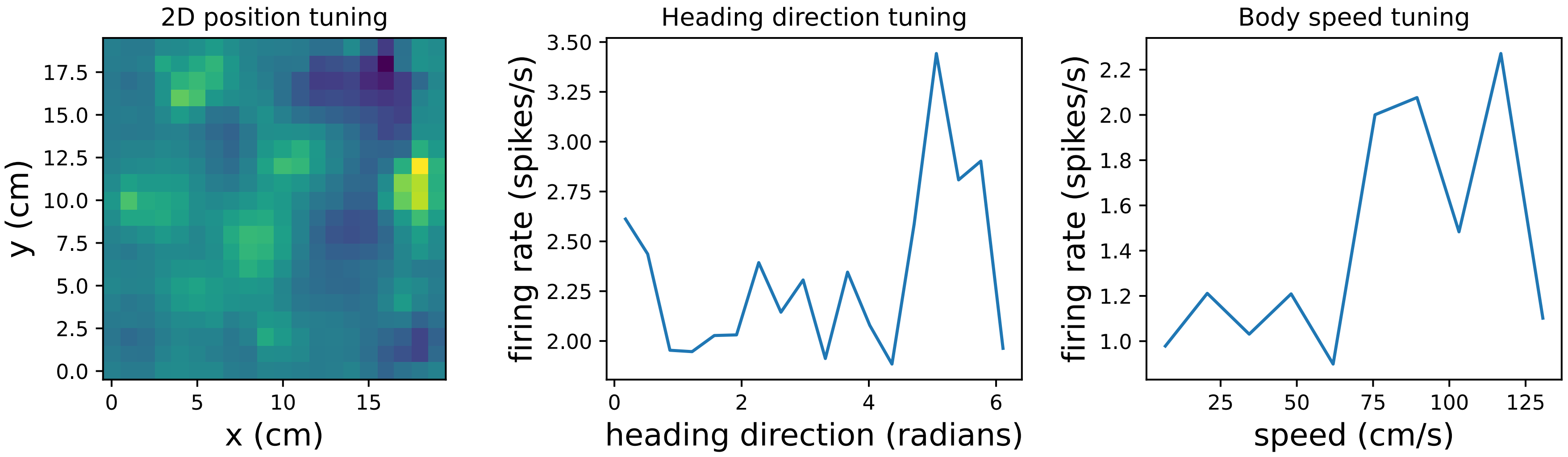
  In practice, ought to use tricks like regularization, k-fold cross-validation, etc…
  Be careful about measuring goodness-of-fit; deviance more appropriate than squared error!

See Shih-Yi’s GitHub repo for a research-level implementation of GLMs
https://github.com/sytseng/GLM_Tensorflow_2

Coding exercise: your turn!

https://github.com/DrugowitschLab/ML-from-scratch-seminar

Goal: Train Poisson GLM to fit data from 4 mouse MEC neurons



Data from Mallory and Hardcastle et al., *Nat Comm* (2021)

https://figshare.com/authors/Lisa_Giocomo/9864194