FLATIRON INSTITUTE

Generalized Linear Models (GLM)

A conceptual introduction to GLM

Edoardo Balzani, Sep 23rd 2024

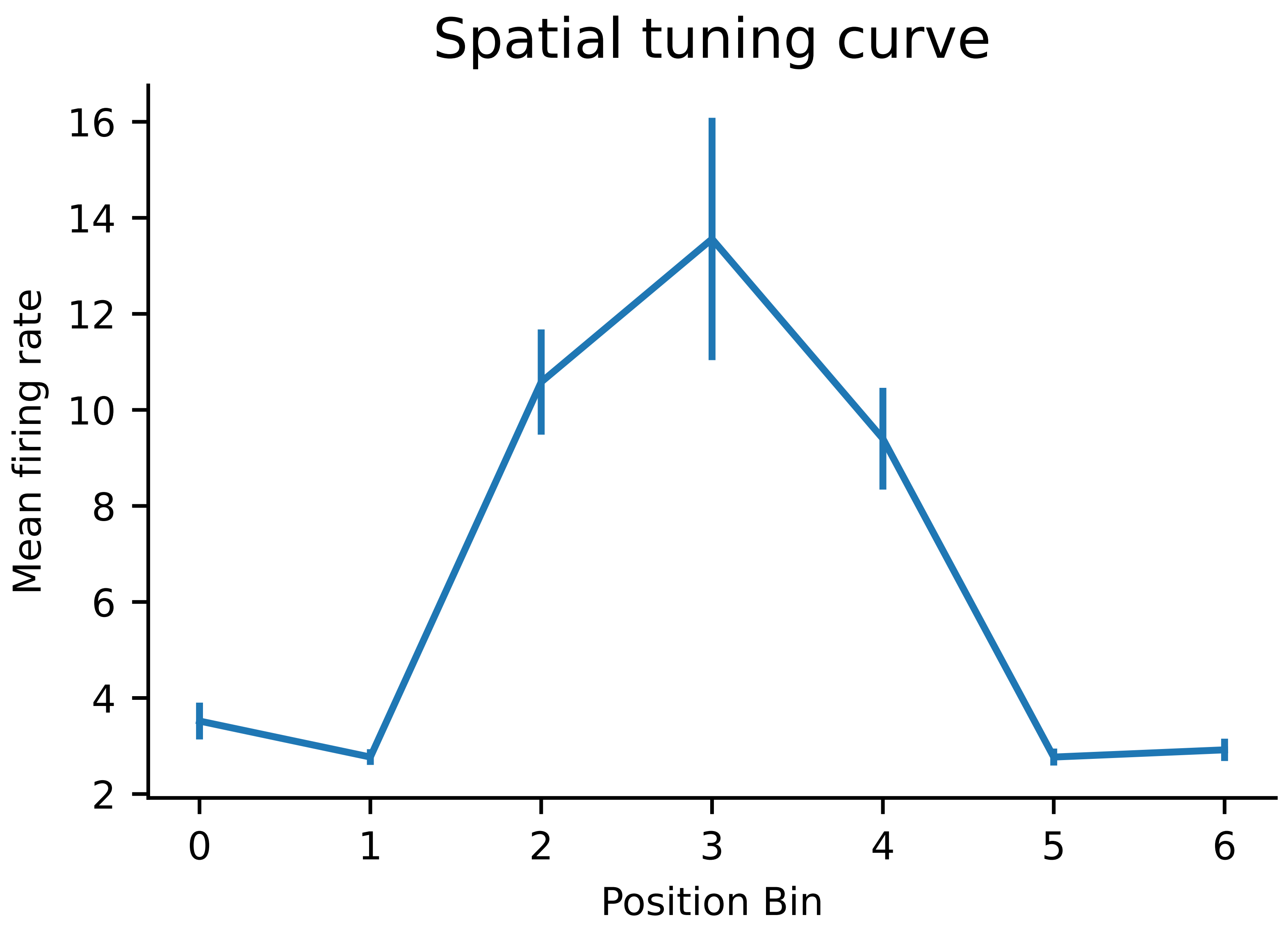
Roadmap

- Why models?
- What are GLMs?
- Why GLMs?
- What can I do with a GLM?
- GLMs In NeMoS
- What features can/should I use?
- Feature construction with Basis
- Summary
- Today’s roadmap

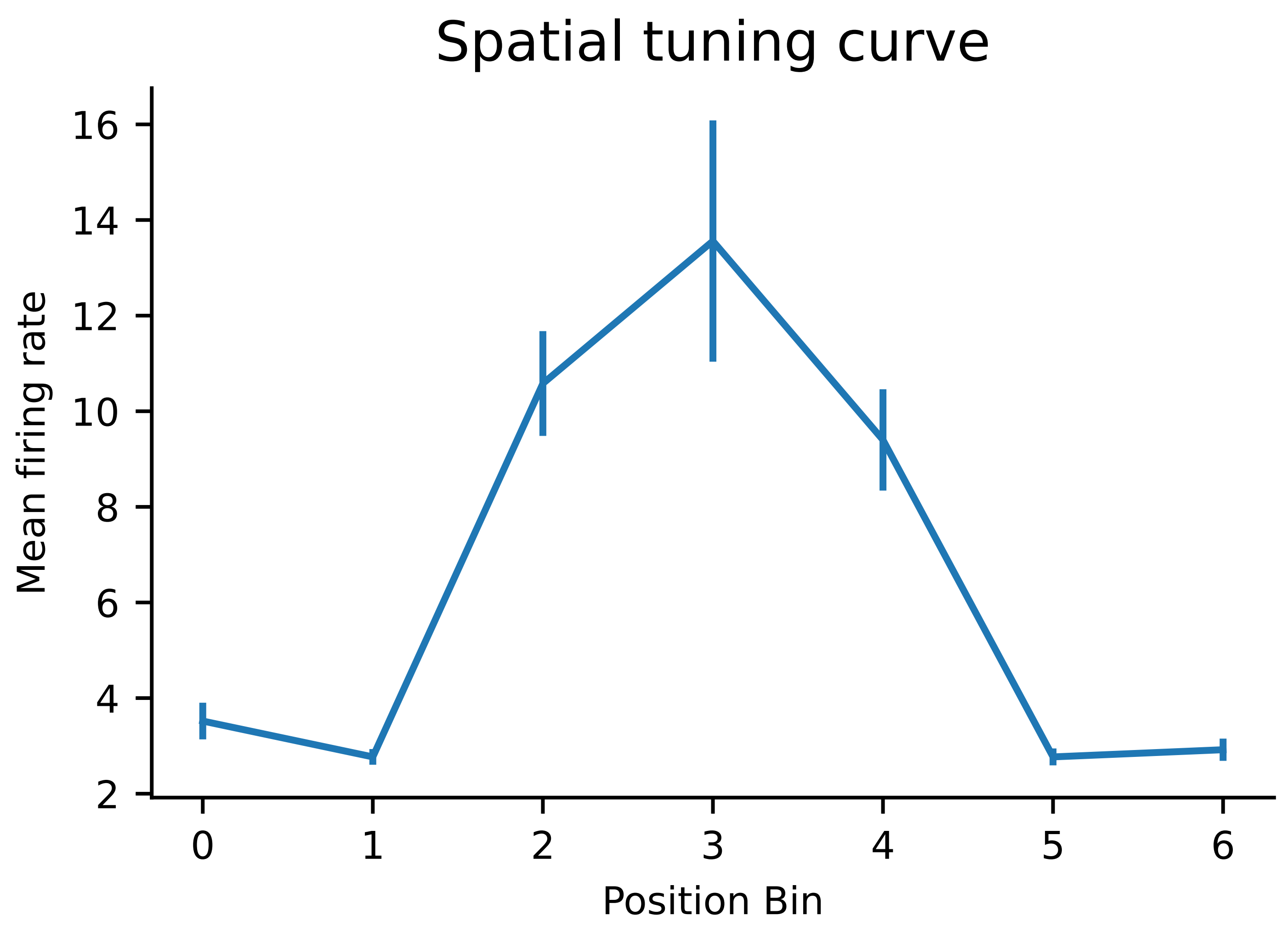
Why models? A hook

linear maze

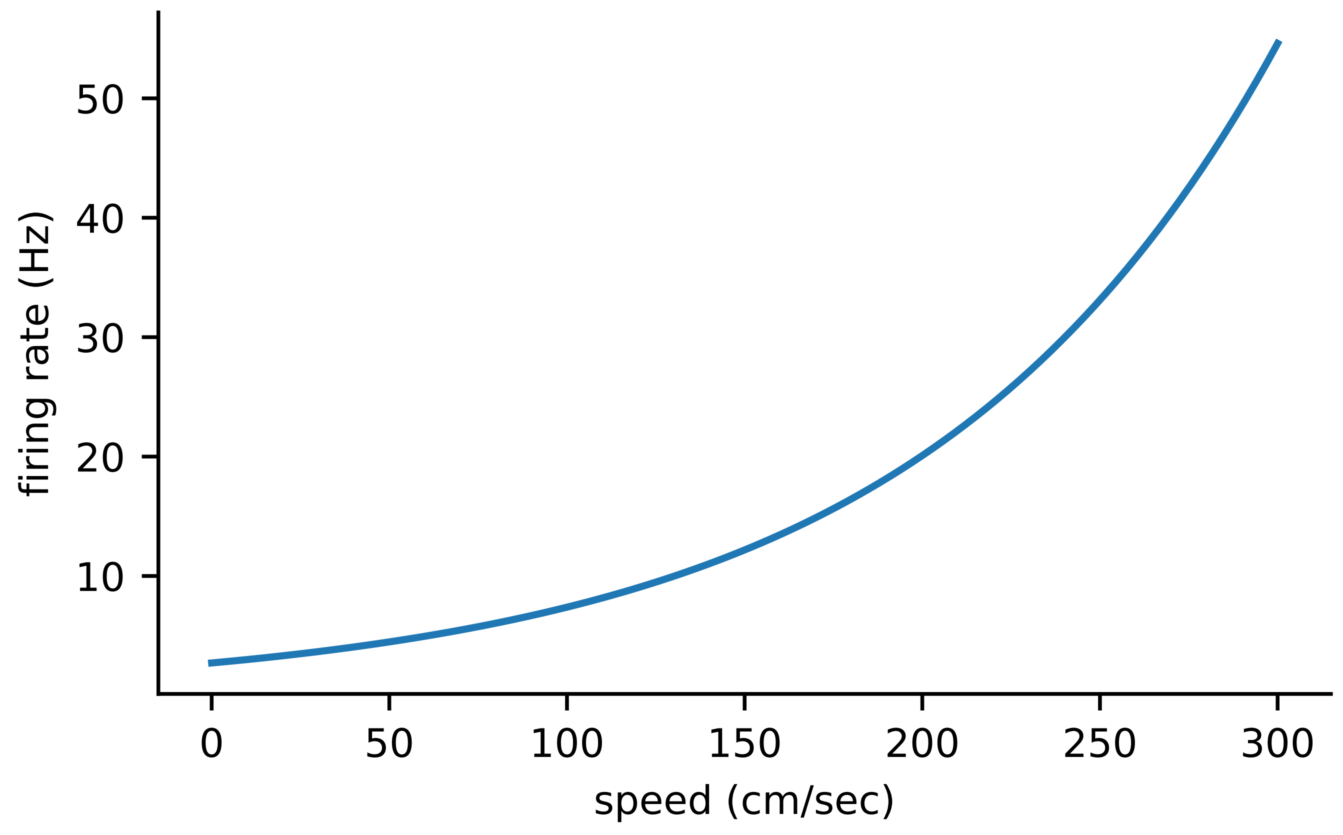
is this neuron encoding the mouse position?



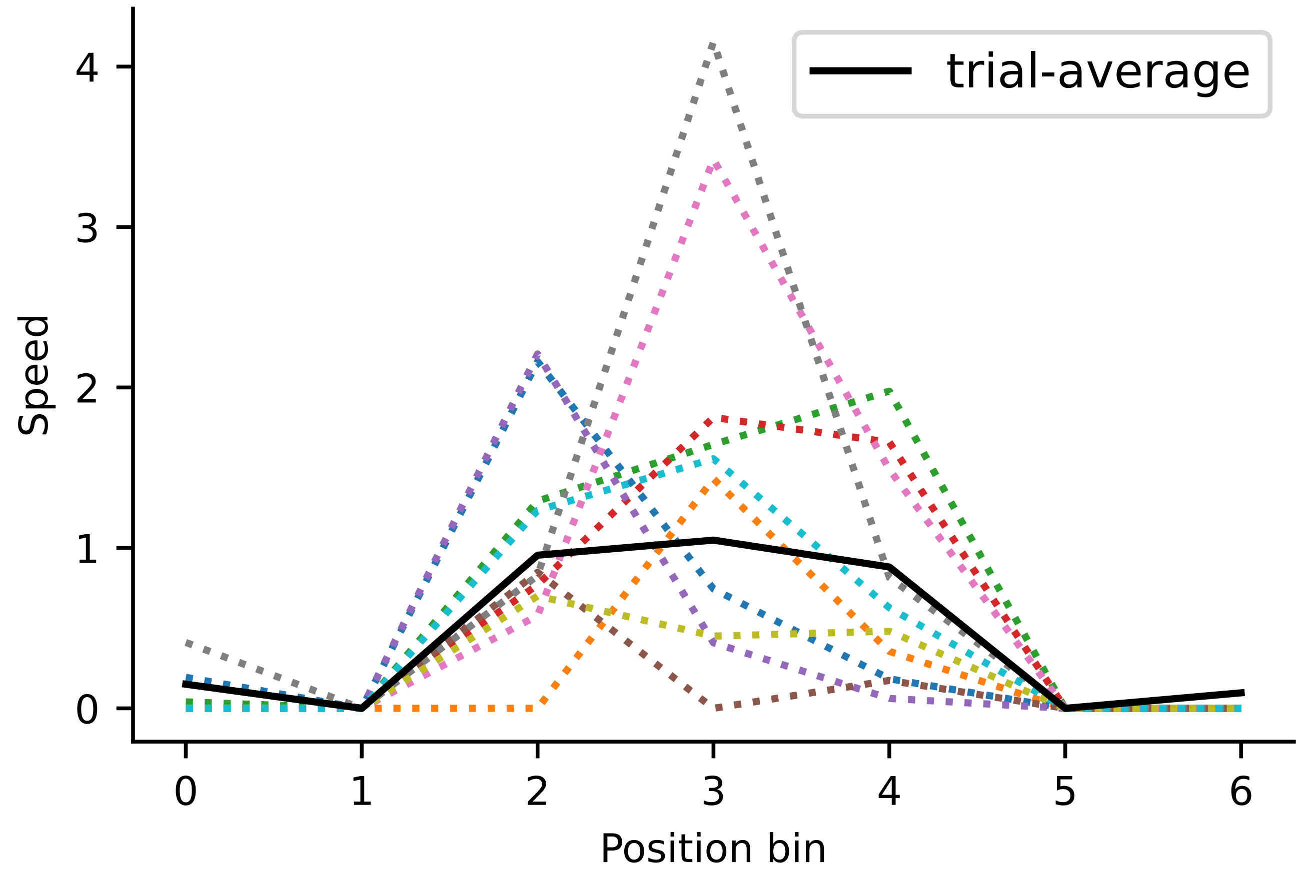
Why models? A hook



..actually, not!



position and speed
are correlated



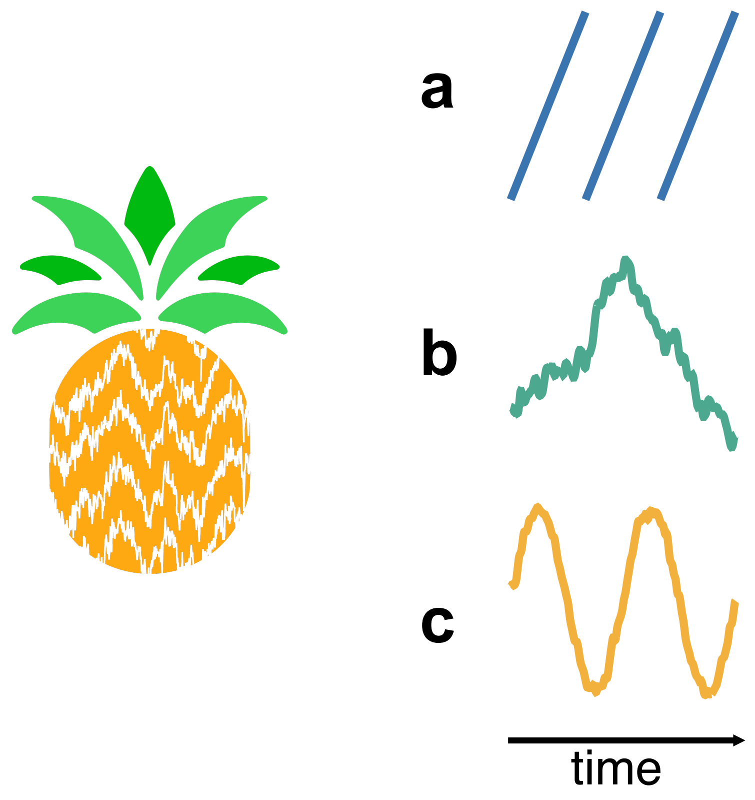
tuning functions don’t tell you the whole story
need better models!

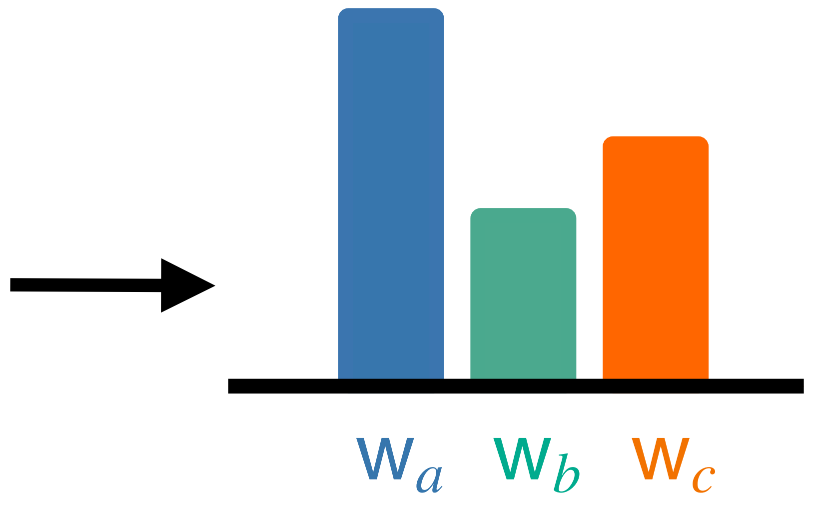
What are GLMs?

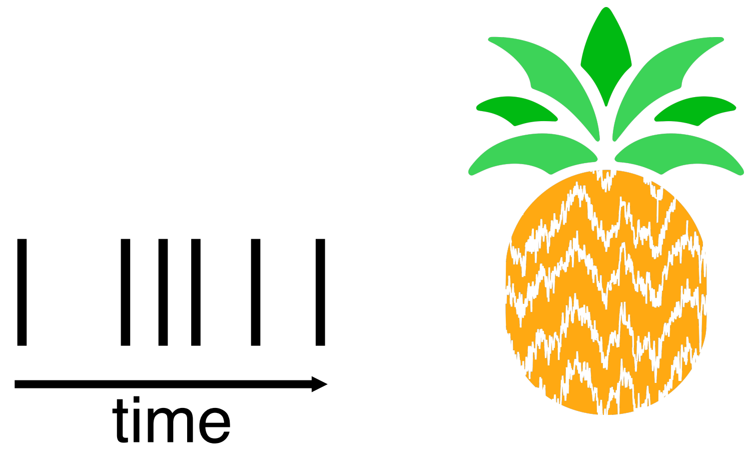
Pre-process

Weights

Observed counts



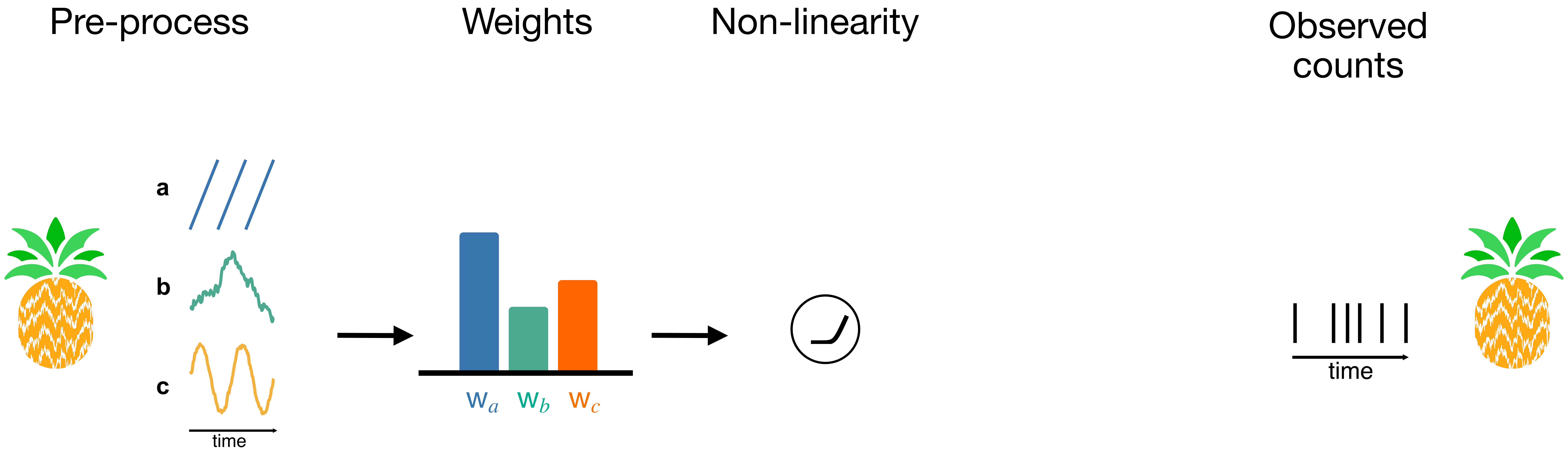




scale the inputs by some weights

$$\mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c$$

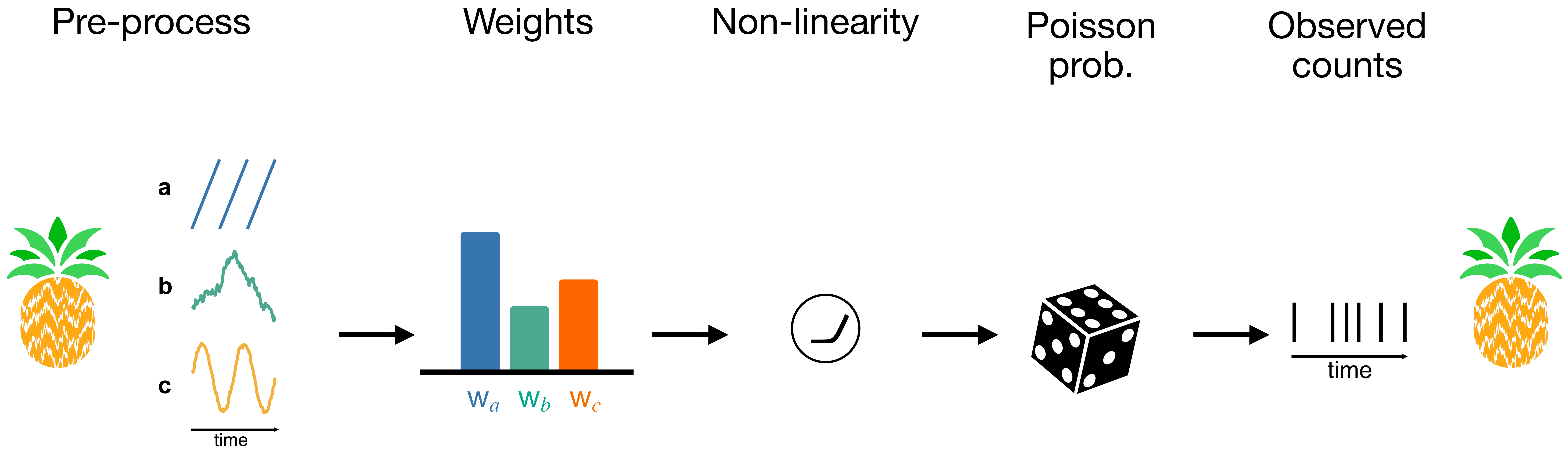
What are GLMs?



non-linearity to make the result positive

firing rate = exp(**a** · w_a + **b** · w_b + **c** · w_c)

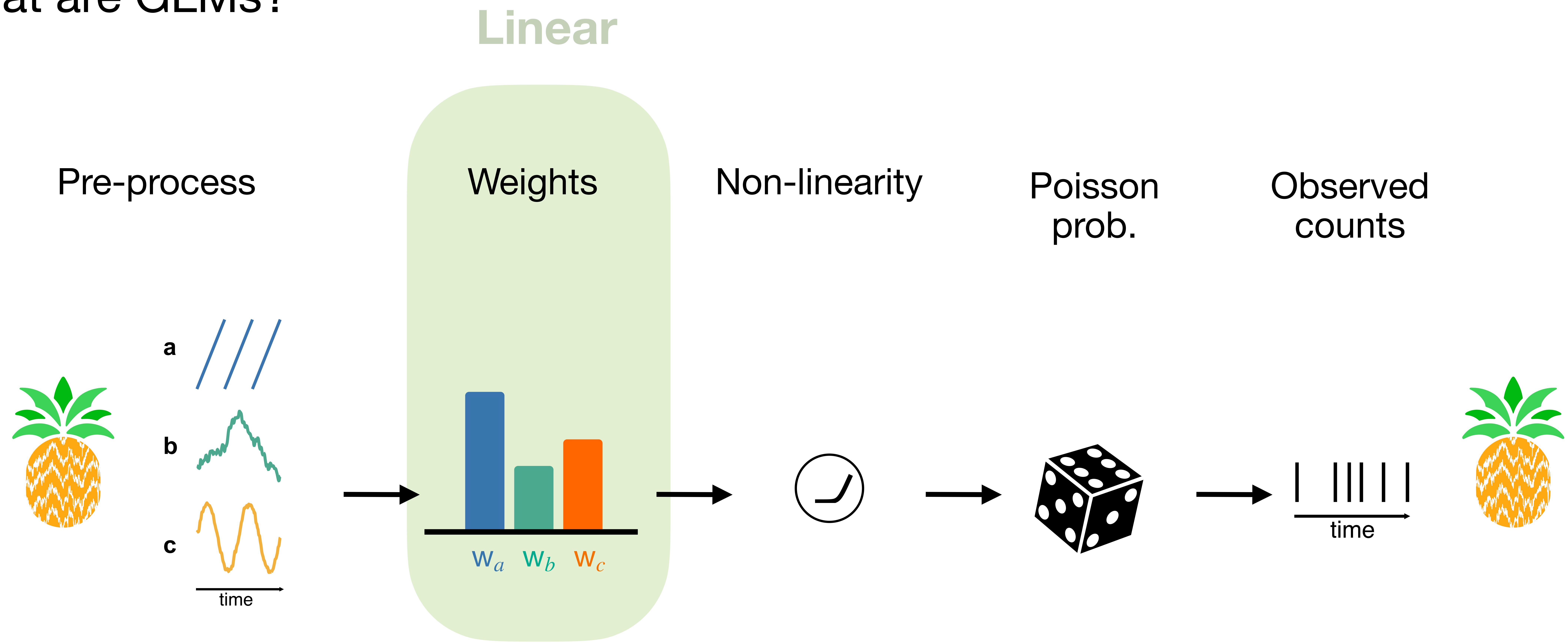
What are GLMs?



firing rate = exp(**a** · w_a + **b** · w_b + **c** · w_c)

probability(spike count $= k$) $=$ Poisson($k \mid$ firing rate)

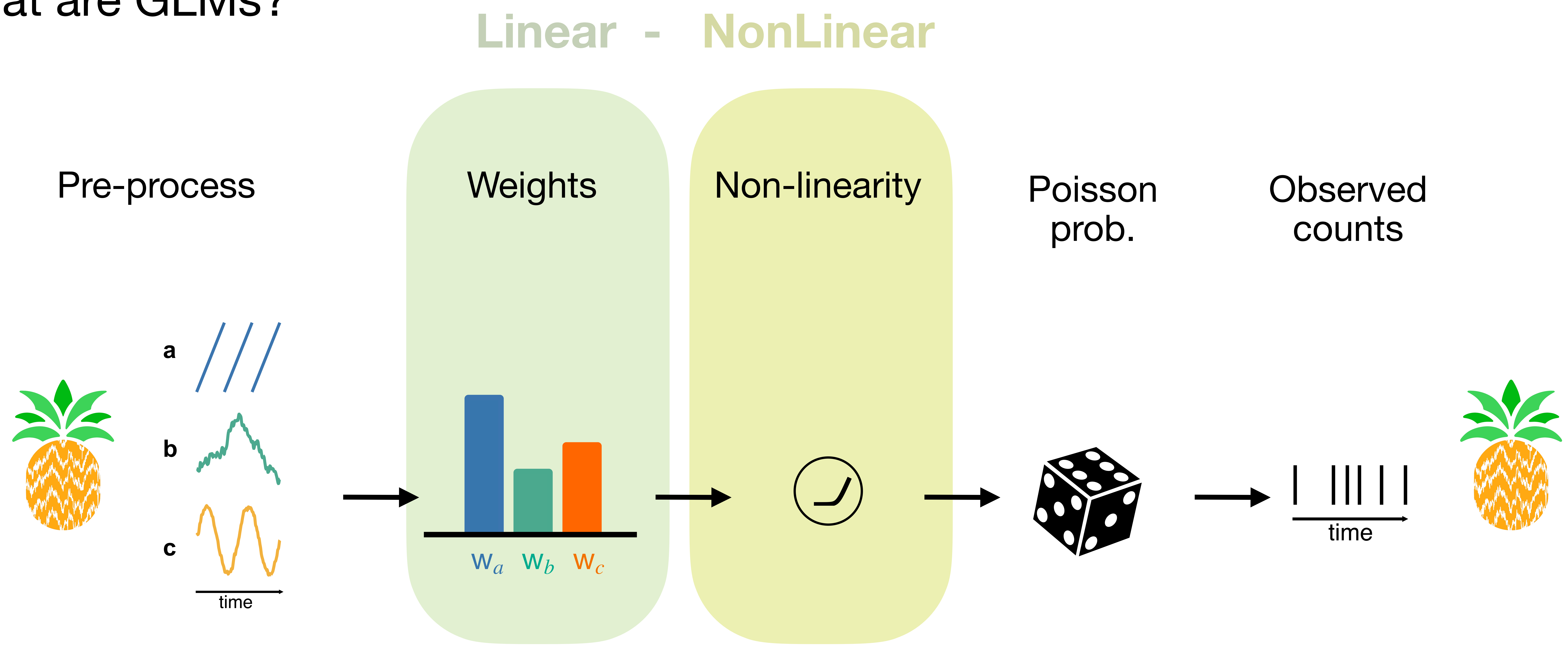
What are GLMs?



$$\text{firing rate} = \exp(\,\mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c\,)$$

$$\text{probability}(\text{spike count} = k) = \text{Poisson}(k \mid \text{firing rate})$$

What are GLMs?

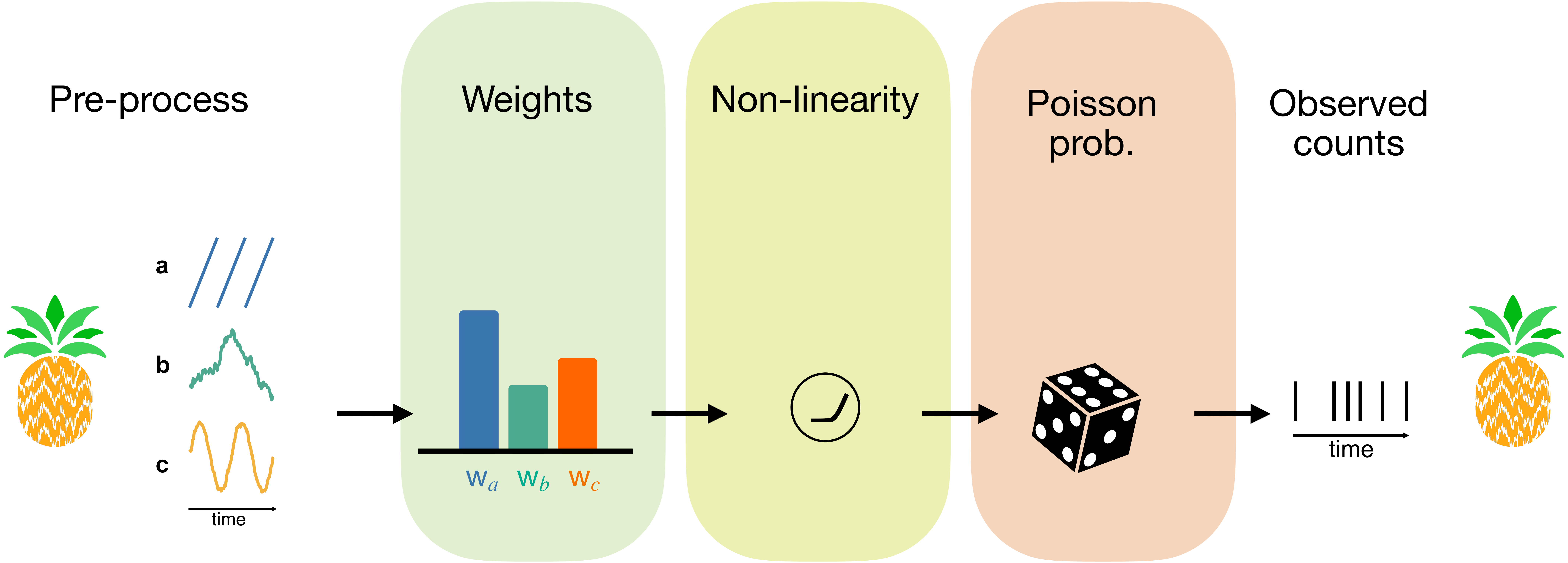


firing rate = exp(**a** $\cdot$ w_a + **b** $\cdot$ w_b + **c** $\cdot$ w_c)

probability(spike count $= k$) $=$ Poisson($k \,|\,$ firing rate)

What are GLMs?

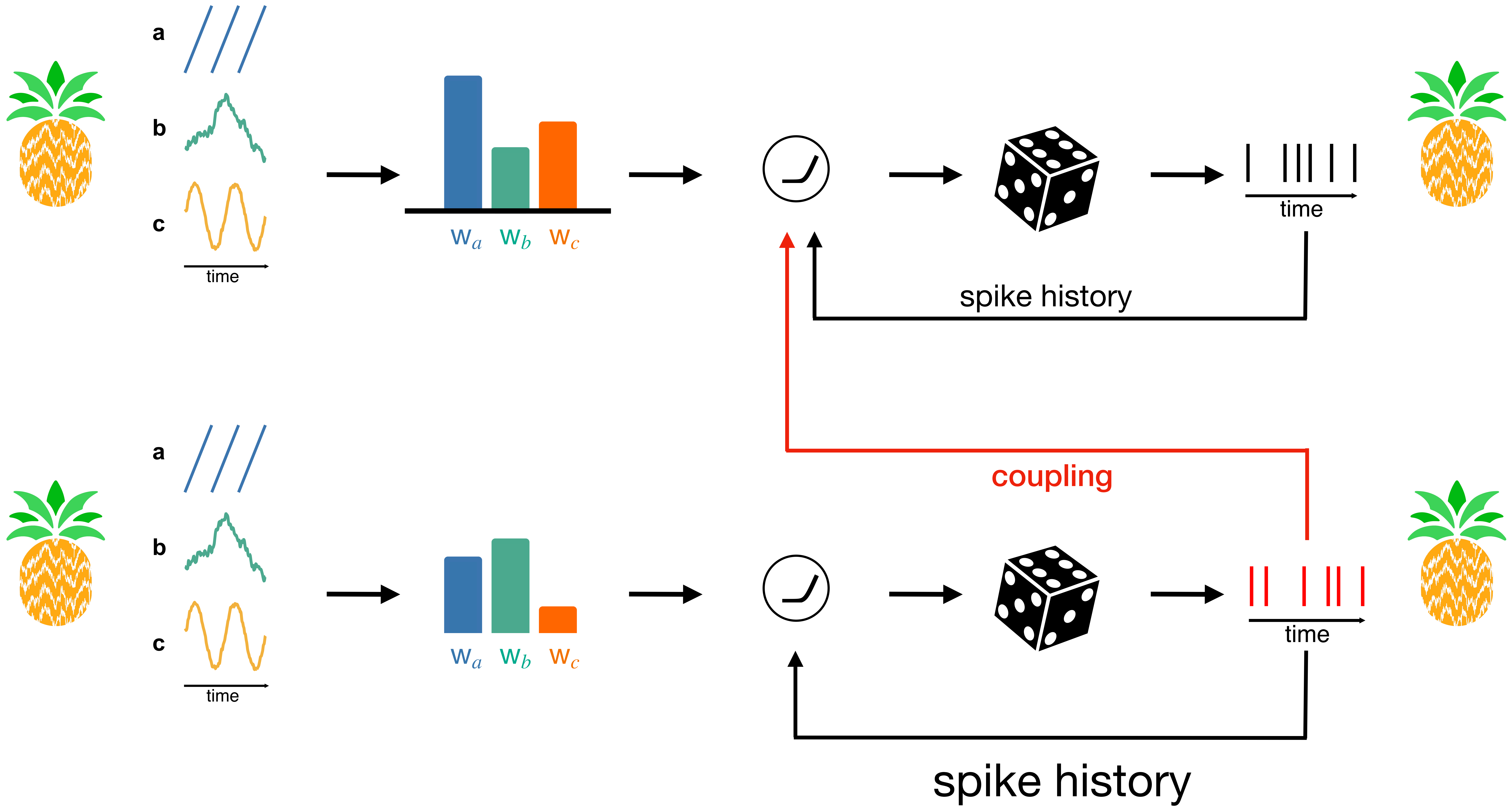
Linear - NonLinear - Poisson (LNP)



$$\text{firing rate} = \exp(\,\mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c\,)$$

$$\text{probability}(\text{spike count} = k) = \text{Poisson}(k \mid \text{firing rate})$$

What are GLMs?



Terminology

$$\text{firing rate} = \exp(\,\mathbf{a} \cdot \mathrm{w}_a + \mathbf{b} \cdot \mathrm{w}_b + \mathbf{c} \cdot \mathrm{w}_c\,)$$

- **a**, **b**, **c** are called **features** or **predictors**

Terminology

$$\text{firing rate} = \exp(\, \mathbf{a} \cdot \mathrm{w}_a + \mathbf{b} \cdot \mathrm{w}_b + \mathbf{c} \cdot \mathrm{w}_c \,)$$

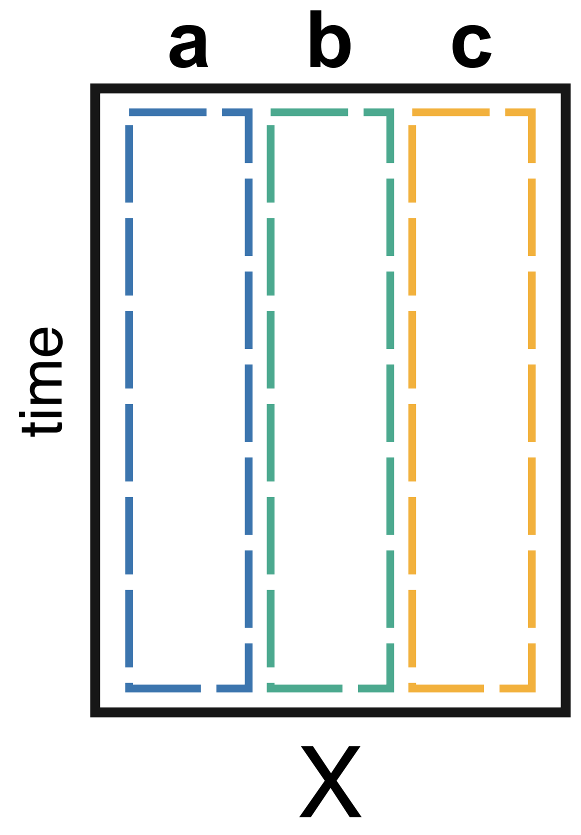
- **a**, **b**, **c** are called **features** or **predictors**
- $\mathbf{w}_a$, $\mathbf{w}_b$, $\mathbf{w}_c$ are called **weights** or **coefficients**

Terminology

$$\text{firing rate} = \exp(\, \mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c \,)$$

- **a**, **b**, **c** are called **features** or **predictors**
- $\mathbf{w}_a$, $\mathbf{w}_b$, $\mathbf{w}_c$ are called **weights** or **coefficients**
- Features are concatenated to form the **design** or **feature** matrix $\mathbf{X} = [\mathbf{a}, \mathbf{b}, \mathbf{c}]$

Feature matrix

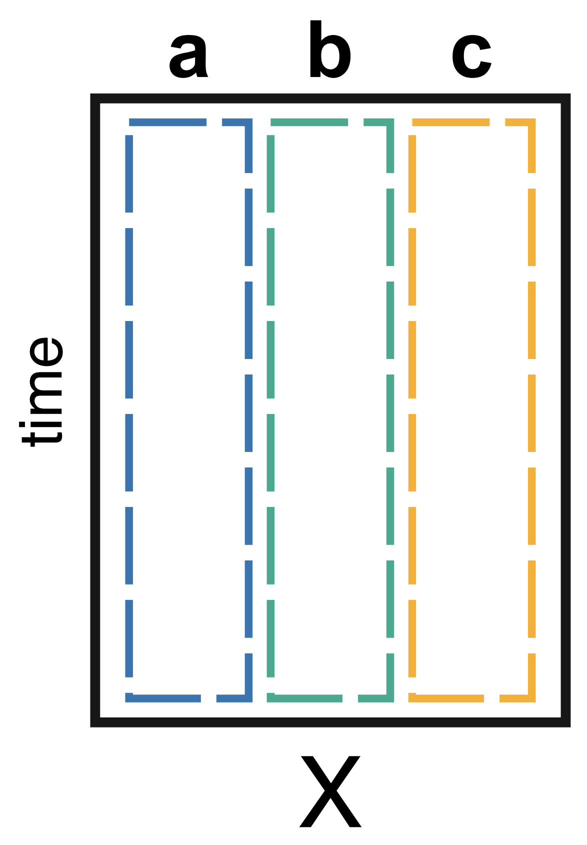


Terminology

$$\text{firing rate} = \exp(\,\mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c\,)$$

- **a**, **b**, **c** are called **features** or **predictors**
- $\mathbf{w}_a$, $\mathbf{w}_b$, $\mathbf{w}_c$ are called **weights** or **coefficients**
- Features are concatenated to form the **design** or **feature** matrix $\mathbf{X} = [\mathbf{a}, \mathbf{b}, \mathbf{c}]$
- The **likelihood** is the probability of observing spike counts given some features and weights.

Design matrix



Likelihood

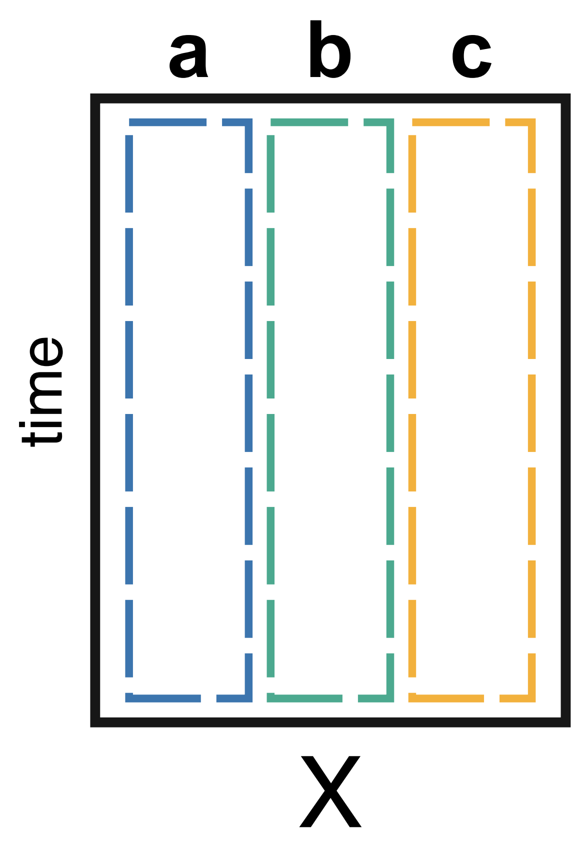
$$\text{probability}(\text{spike count} = k \,|\, \mathbf{X}, \mathbf{w})$$

Terminology

$$\text{firing rate} = \exp(\,\mathbf{a} \cdot w_a + \mathbf{b} \cdot w_b + \mathbf{c} \cdot w_c\,)$$

- **a**, **b**, **c** are called **features** or **predictors**
- $\mathbf{w}_a$, $\mathbf{w}_b$, $\mathbf{w}_c$ are called **weights** or **coefficients**
- Features are concatenated to form the **design** or **feature** matrix $\mathbf{X} = [\mathbf{a}, \mathbf{b}, \mathbf{c}]$
- The **likelihood** is the probability of observing spike counts given some features and weights.
- The **likelihood is a function of the weights** because counts and features are fixed.

Design matrix



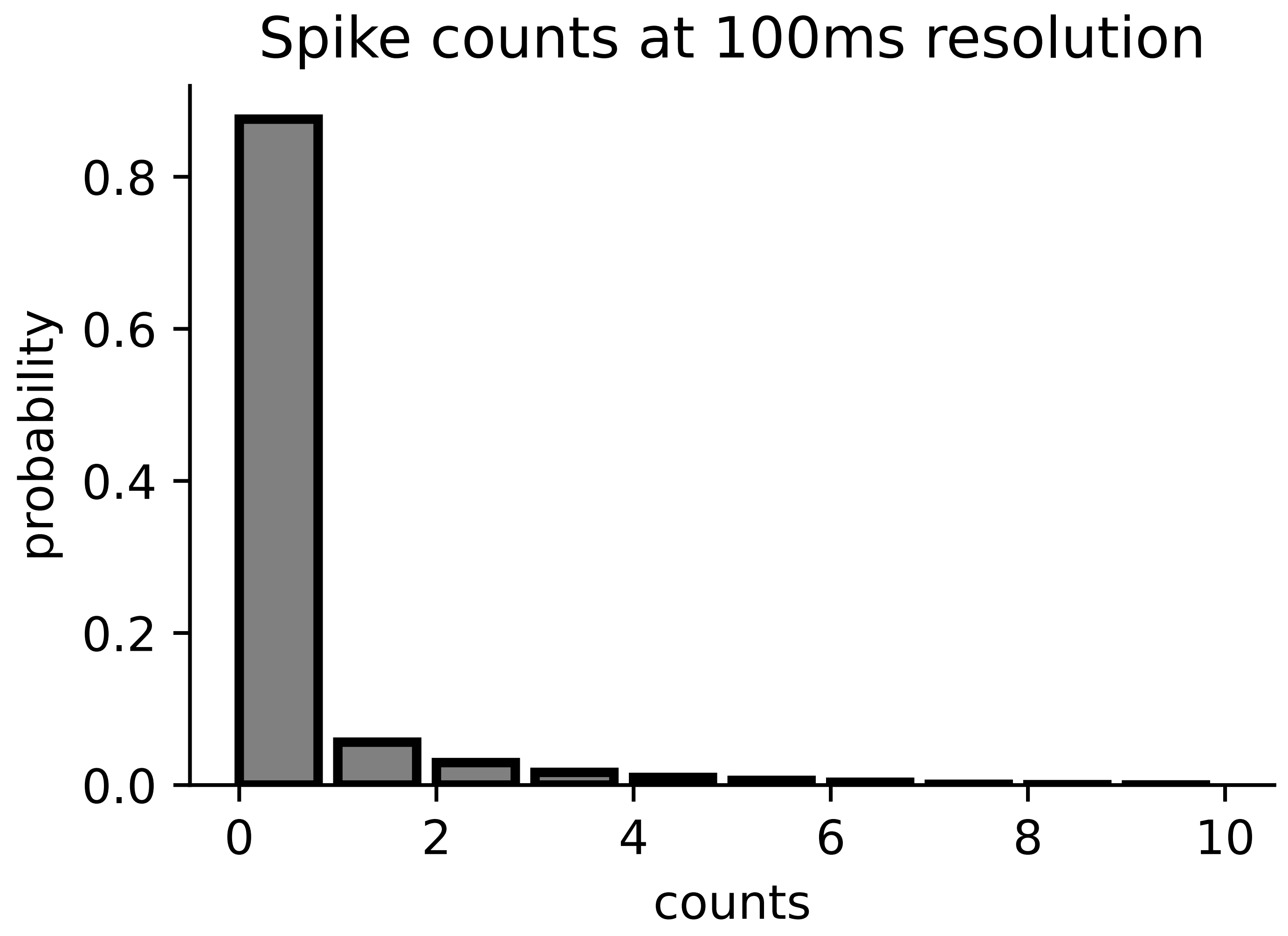
Likelihood

$$\text{probability}(\text{spike count} = k \mid \mathbf{X}, \mathbf{w})$$

Why GLMs?

1. Why not linear regression?
   which assumes normality

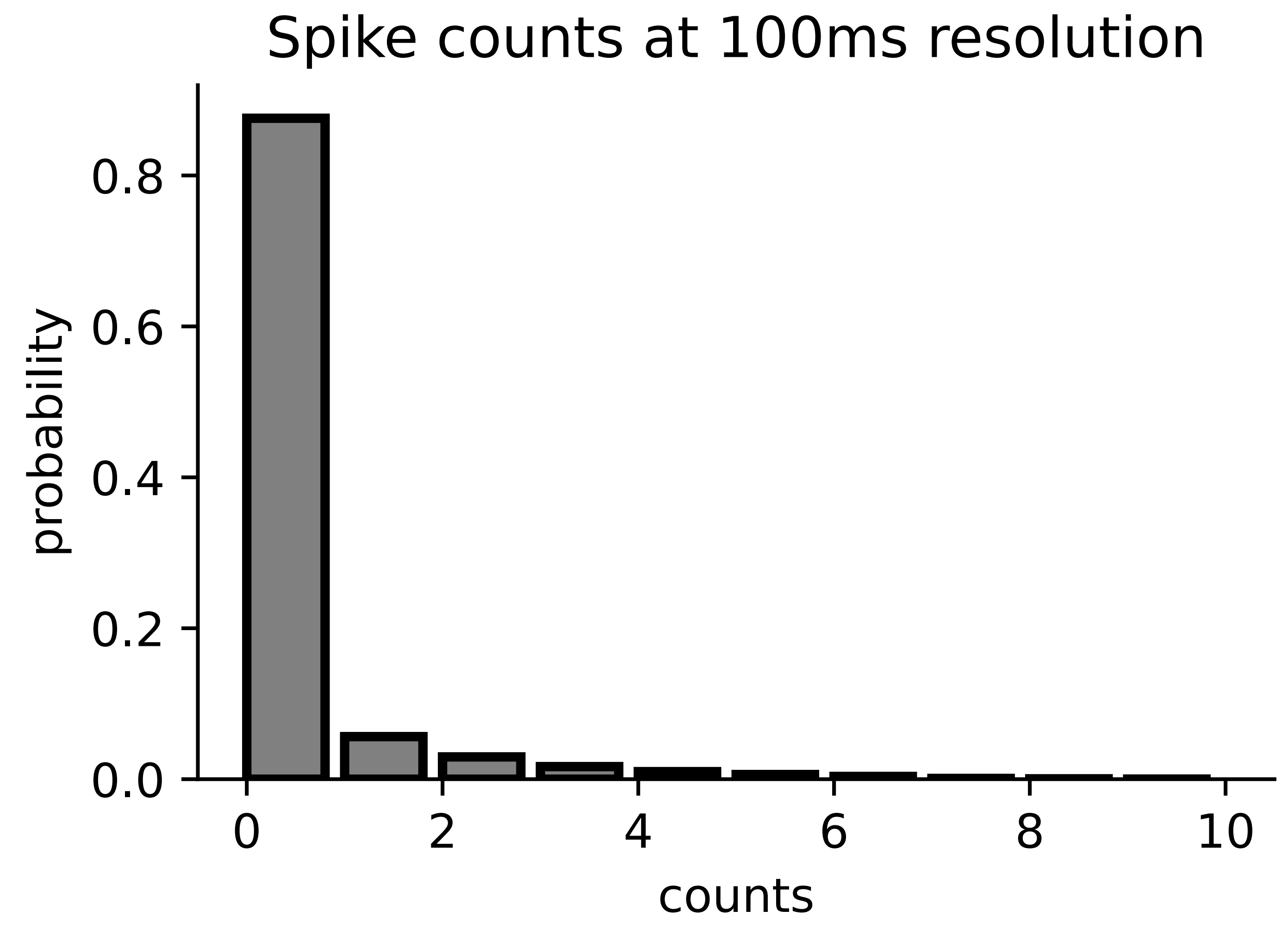
   A. Spike counts are non-Gaussian

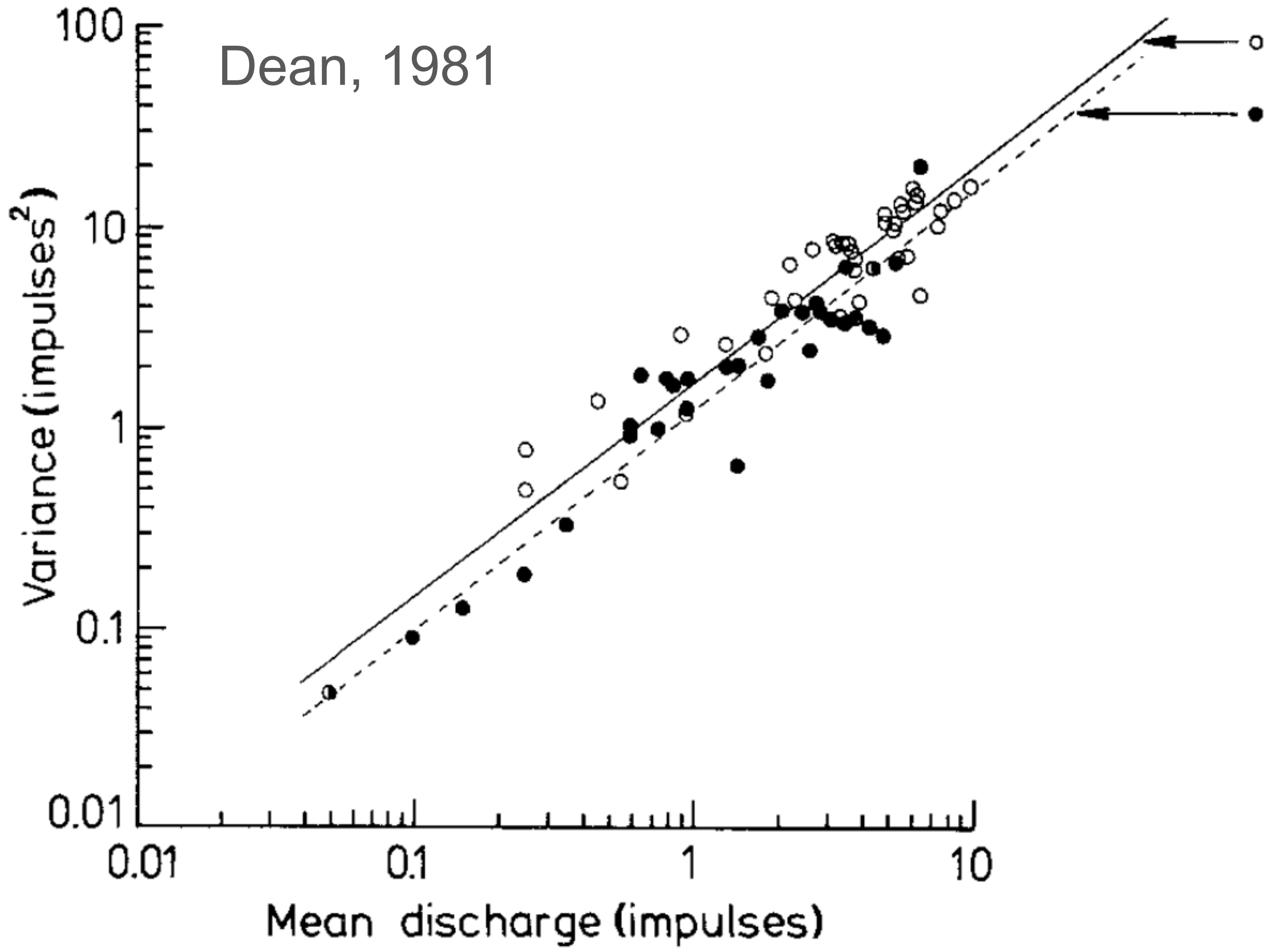


Why GLMs?

1. Why not linear regression?
   which assumes normality

   A. Spike counts are non-Gaussian
   B. Neural activity variance is non-constant





Why GLMs?

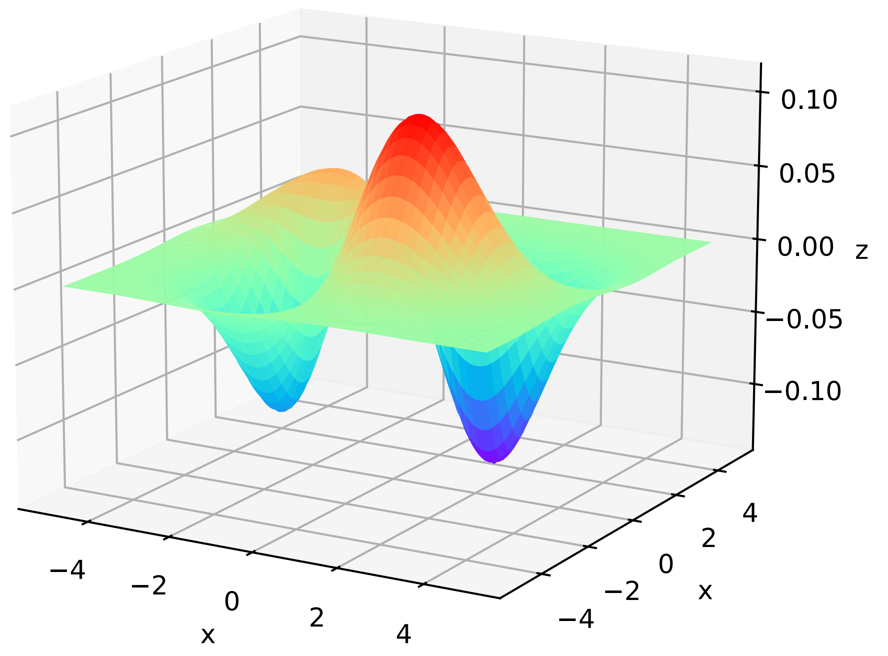
1. Why not linear regression?
   which assumes normality

   A. Spike counts are non-Gaussian

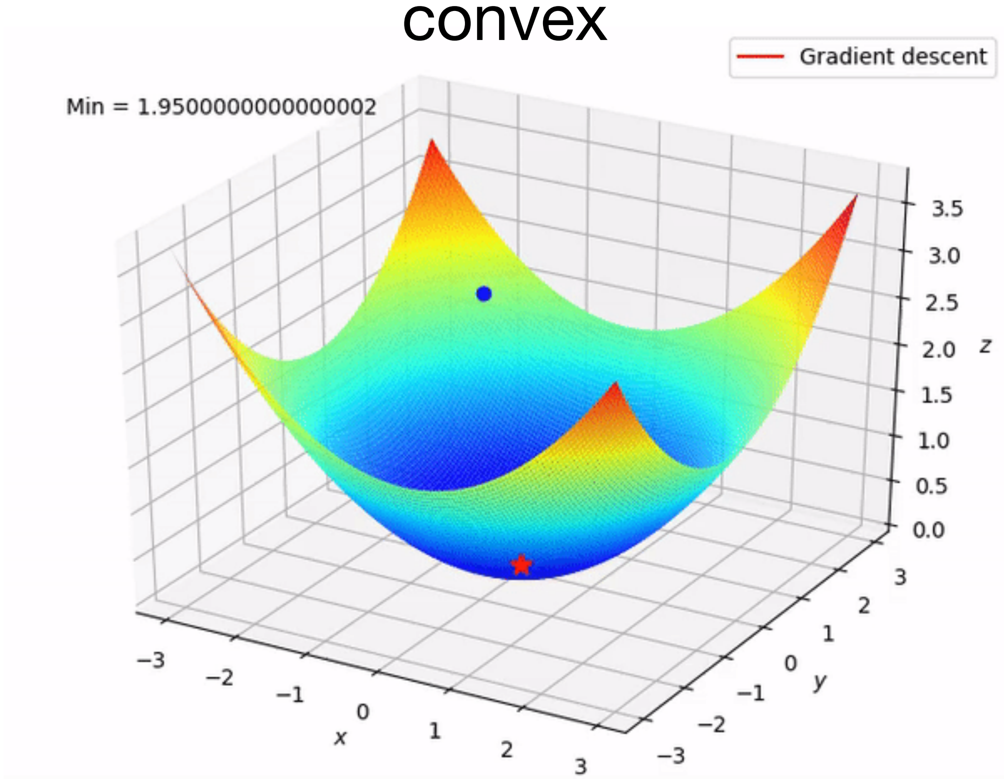
   B. Neural activity variance is non-constant

2. GLM are as **easy to fit** as linear regression
   convex, unique optimal solution

non-convex



convex



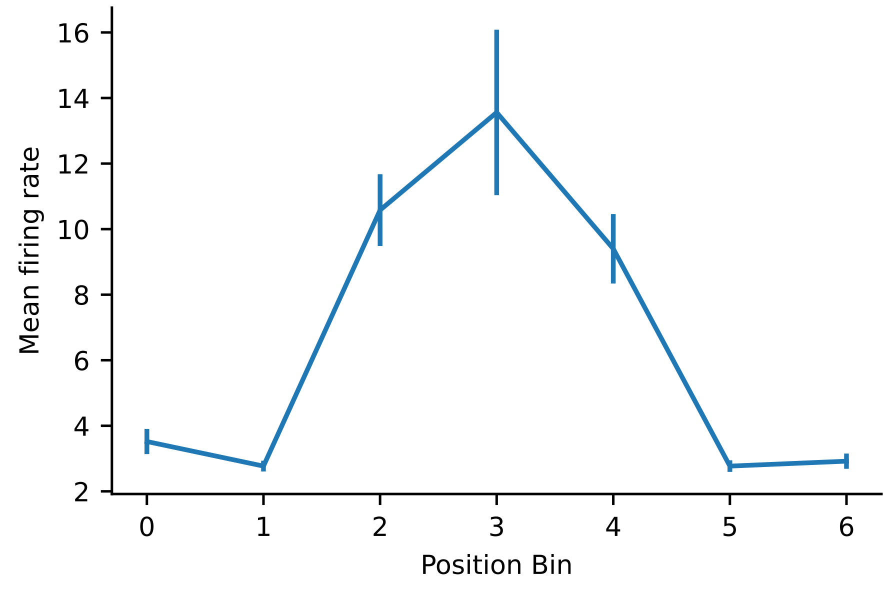
Why GLMs?

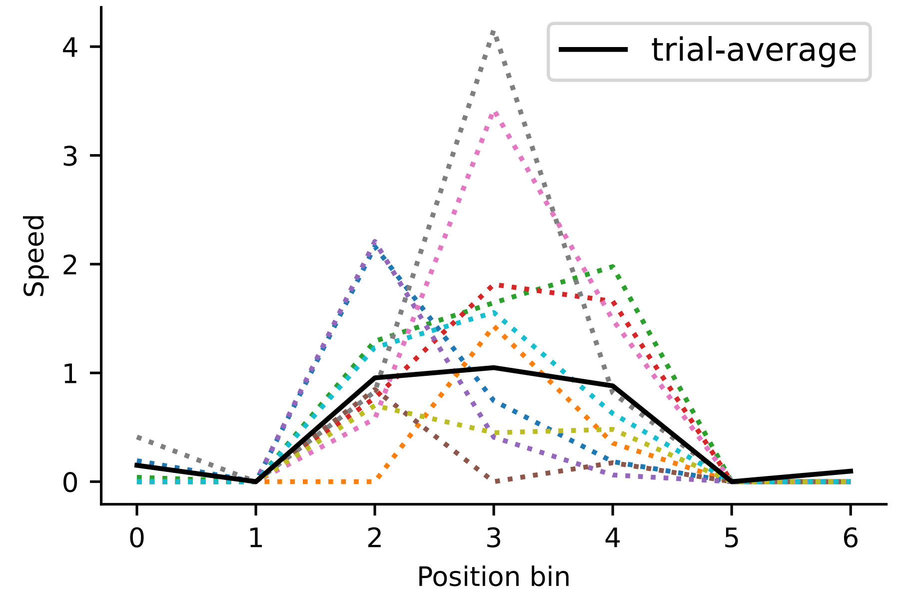
1. Why not linear regression?
   which assumes normality

   A. Spike counts are non-Gaussian
   B. Neural activity variance is non-constant

2. GLM are as **easy to fit** as linear regression
   convex, unique optimal solution

3. GLM are **flexible**
   model multiple inputs jointly





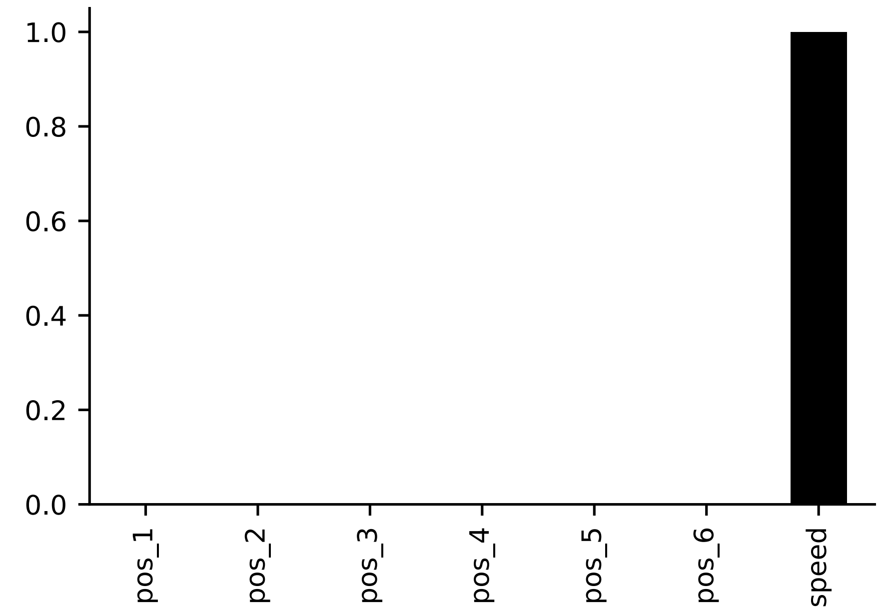
pos 0 pos 1 pos 2 pos 3 pos 4 pos 5 pos 6

Firing rate model:

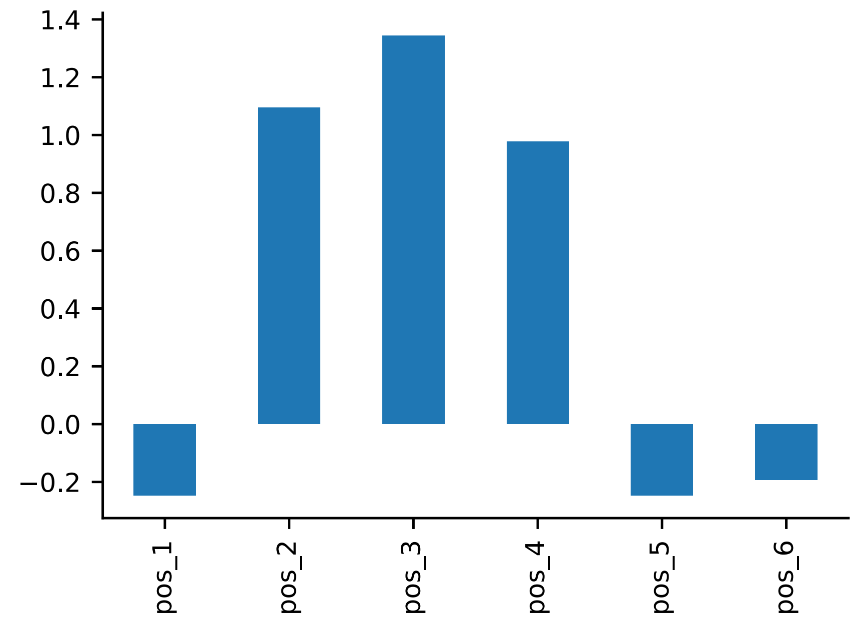
firing rate $= \exp(w_0 \cdot \text{pos}_0(t) + \ldots + w_6 \cdot \text{pos}_6(t))$

$$\text{pos}_i(t) = \begin{cases} 1 & \text{if mouse is in position } i \text{ at time} t \\ 0 & \text{otherwise} \end{cases}$$

true weights



GLM position



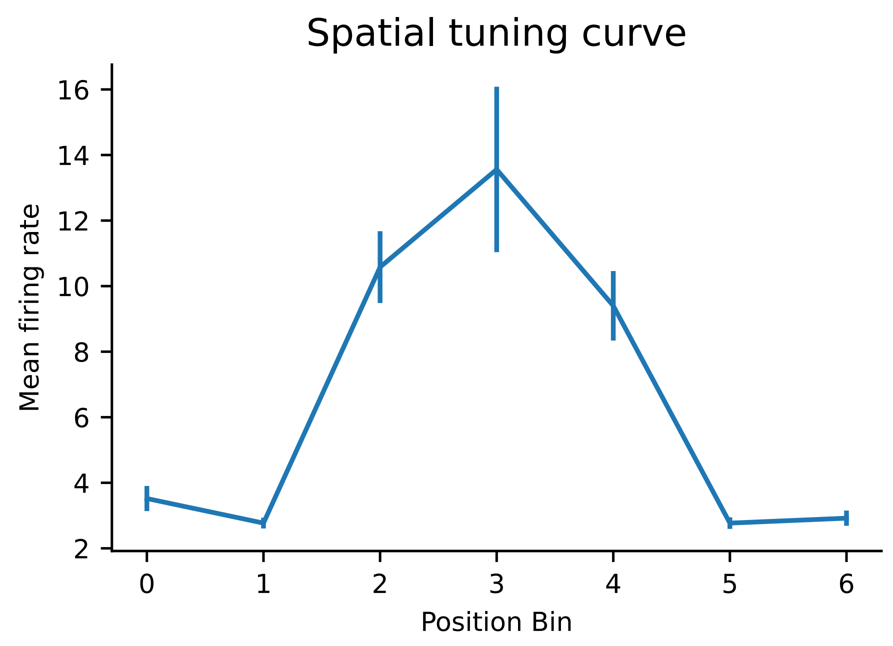
Why GLMs?

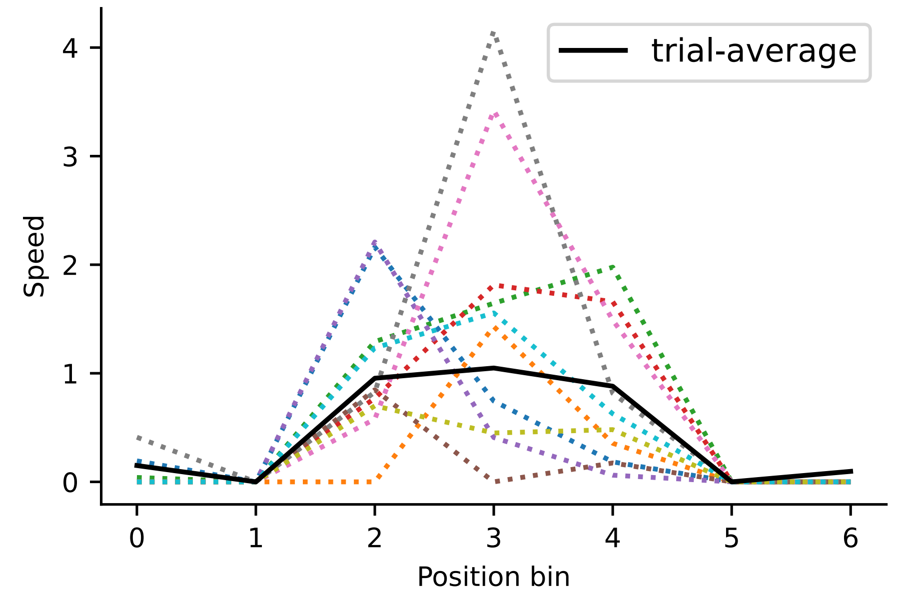
1. Why not linear regression?
   which assumes normality

   A. Spike counts are non-Gaussian
   B. Neural activity variance is non-constant

2. GLM are as **easy to fit** as linear regression
   convex, unique optimal solution

3. GLM are **flexible**
   model multiple inputs jointly

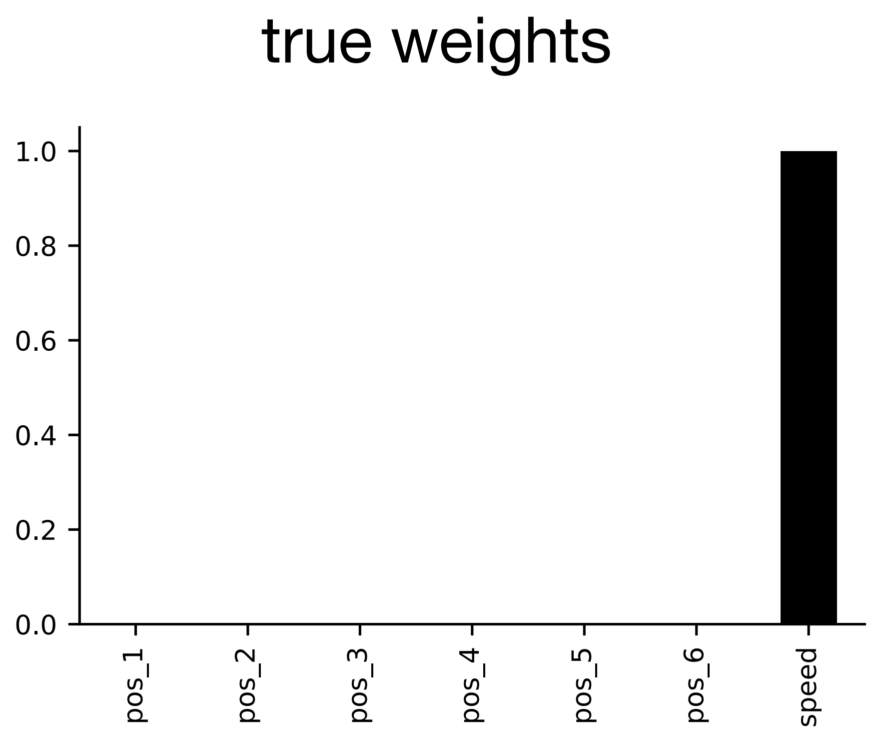


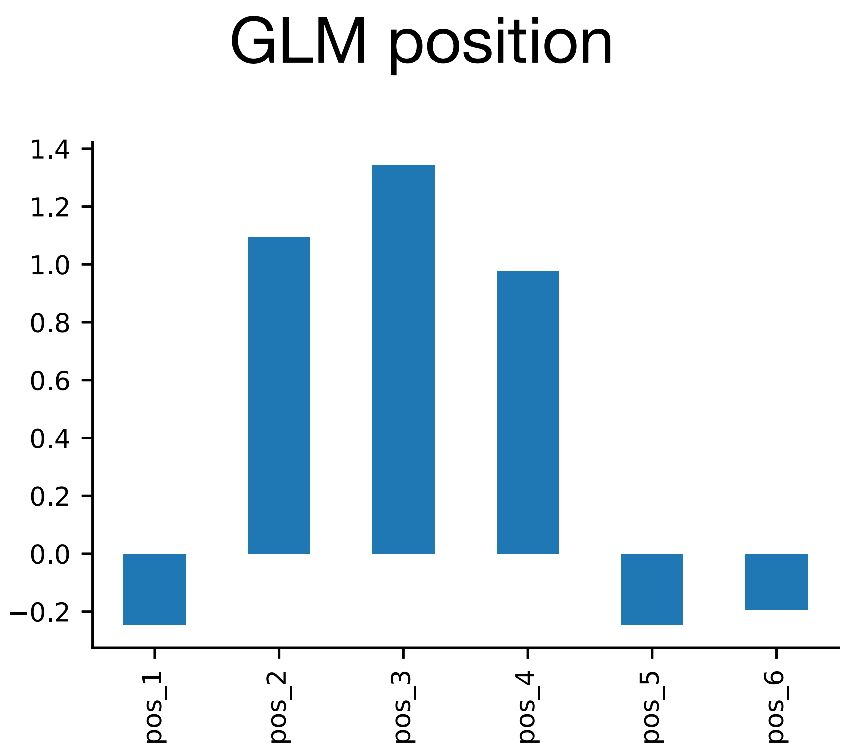


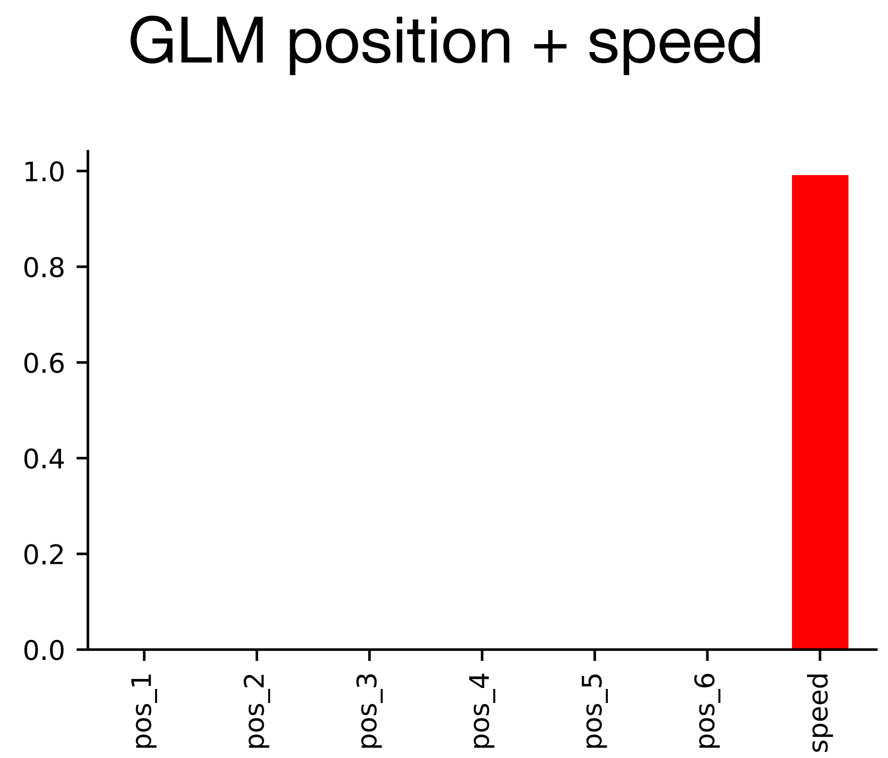
Firing rate model:

$$\text{firing rate} = \exp(w_0 \cdot \text{pos}_0(t) + \ldots + w_6 \cdot \text{pos}_6(t) + w_s \cdot \text{speed}(t))$$

$$\text{pos}_i(t) = \begin{cases} 1 & \text{if mouse is in position } i \text{ at time} t \\ 0 & \text{otherwise} \end{cases}$$







What can I do with a GLM?

1. Model responses to high dimensional inputs

   images, videos, 2D/3D positions…

<u>Pillow at al., 2008</u>
Retina Macaques

<u>Hardcastle et al., 2018</u>
MEC mice

<u>Gardner et al. 2019</u>
MEC rats

<u>Park et al. 2019</u>
LIP Macaques

<u>Weber & Pillow 2017</u>
simulations

<u>Peyrache et al., 2018</u>
ADN mice

What can I do with a GLM?

1. Model responses to high dimensional inputs

   images, videos, 2D/3D positions…

2. Non-linear responses

   place cells, head-direction, grid cells

<u>Pillow at al., 2008</u>
Retina Macaques

<u>Hardcastle et al., 2018</u>
MEC mice

<u>Gardner et al. 2019</u>
MEC rats

<u>Park et al. 2019</u>
LIP Macaques

<u>Weber & Pillow 2017</u>
simulations

<u>Peyrache et al., 2018</u>
ADN mice

What can I do with a GLM?

1. Model responses to high dimensional inputs

   images, videos, 2D/3D positions…

2. Non-linear responses

   place cells, head-direction, grid cells

3. Functional connectivity

   and other time-dependent effects

Pillow at al., 2008
Retina Macaques

Hardcastle et al., 2018
MEC mice

Gardner et al. 2019
MEC rats

Park et al. 2019
LIP Macaques

Weber & Pillow 2017
simulations

Peyrache et al., 2018
ADN mice

What can I do with a GLM?

1. Model responses to high dimensional inputs

   images, videos, 2D/3D positions…

2. Non-linear responses

   place cells, head-direction, grid cells

3. Functional connectivity

   and other time-dependent effects

4. Generate surrogate dataset

<u>Pillow at al., 2008</u>
Retina Macaques

<u>Hardcastle et al., 2018</u>
MEC mice

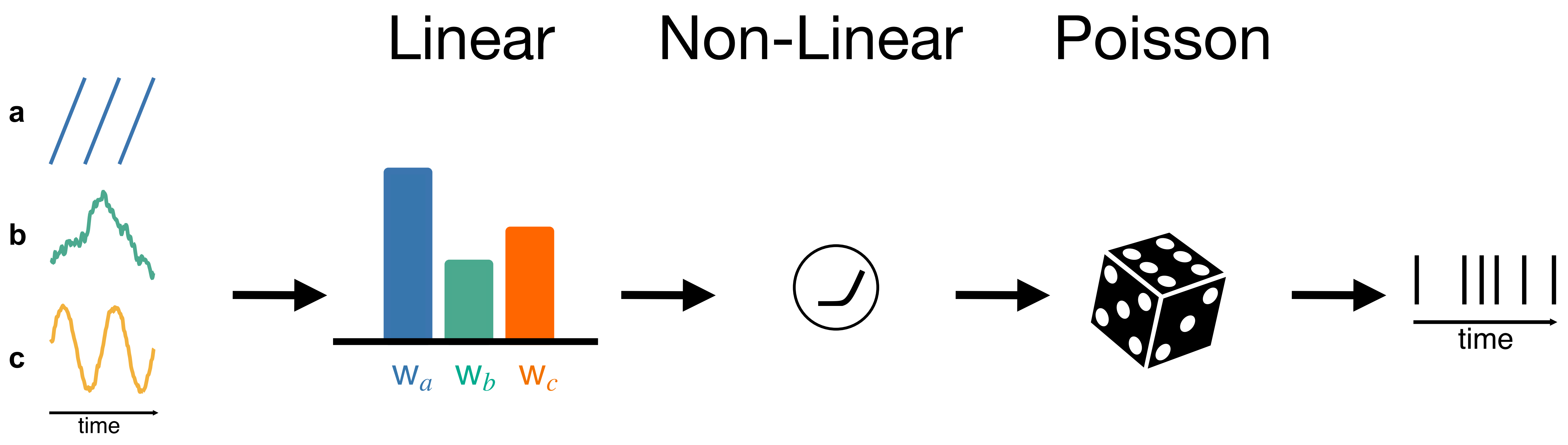
<u>Gardner et al. 2019</u>
MEC rats

<u>Park et al. 2019</u>
LIP Macaques

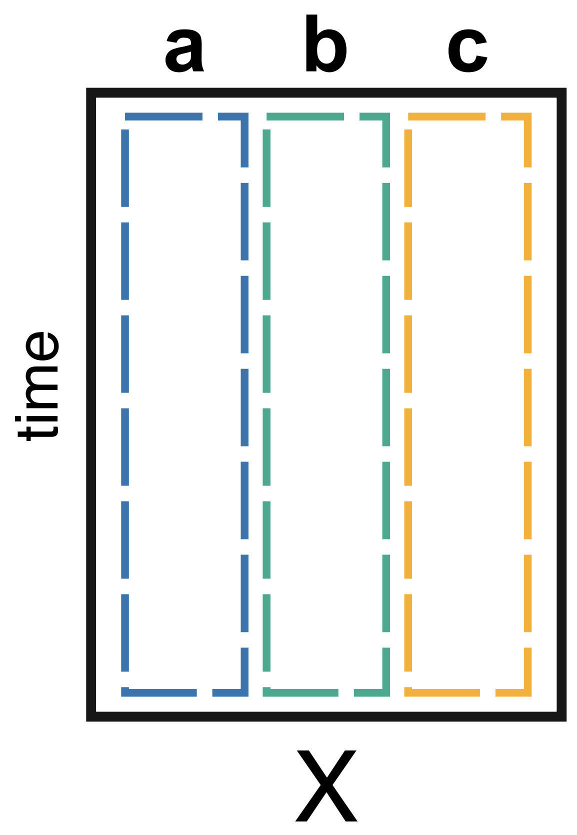
<u>Weber & Pillow 2017</u>
simulations

<u>Peyrache et al., 2018</u>
ADN mice

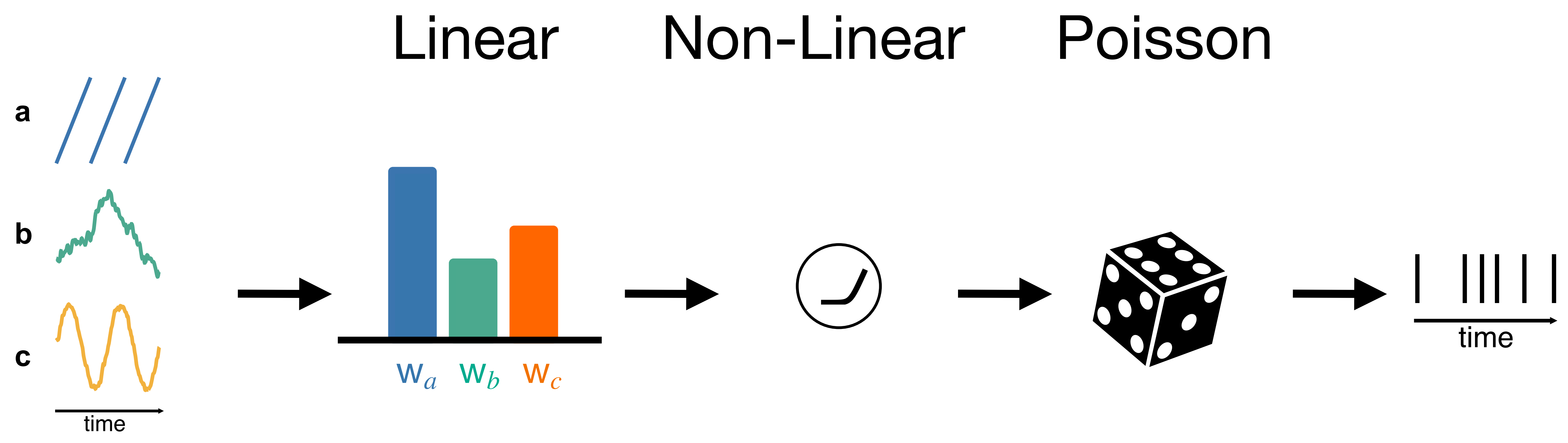
GLM in NeMoS



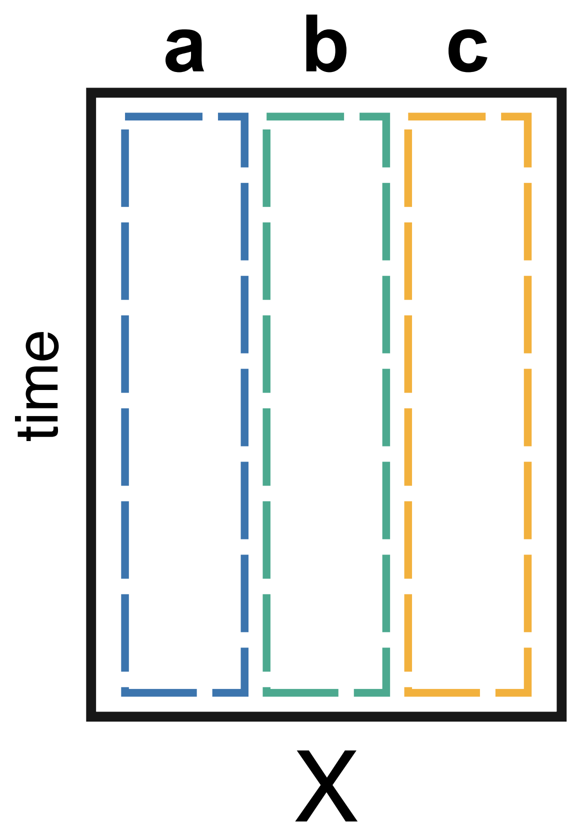
Feature matrix



GLM in NeMoS



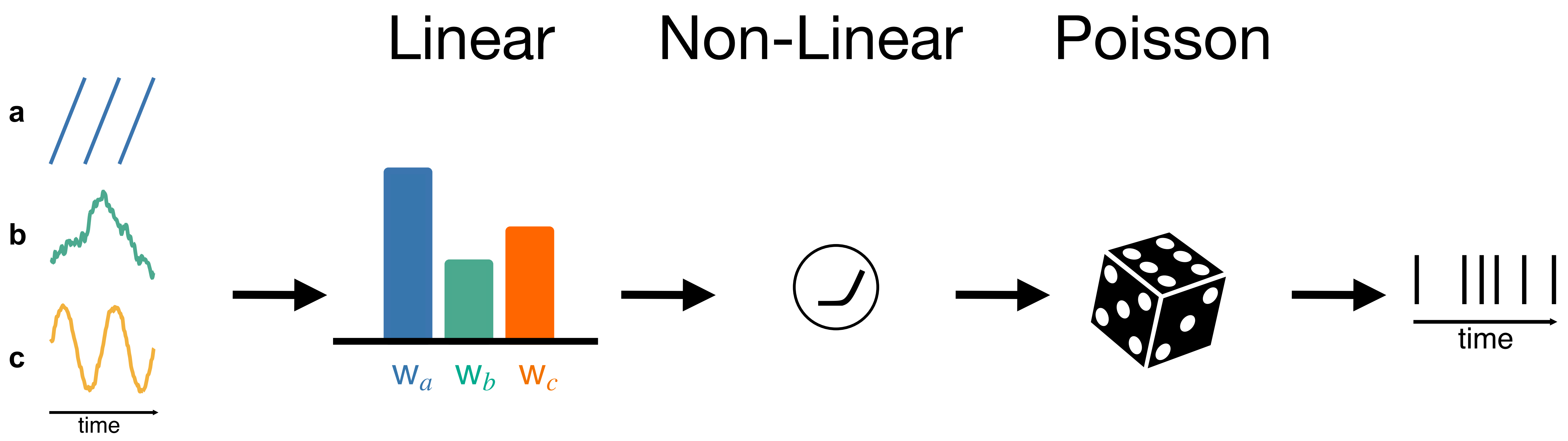
Feature matrix



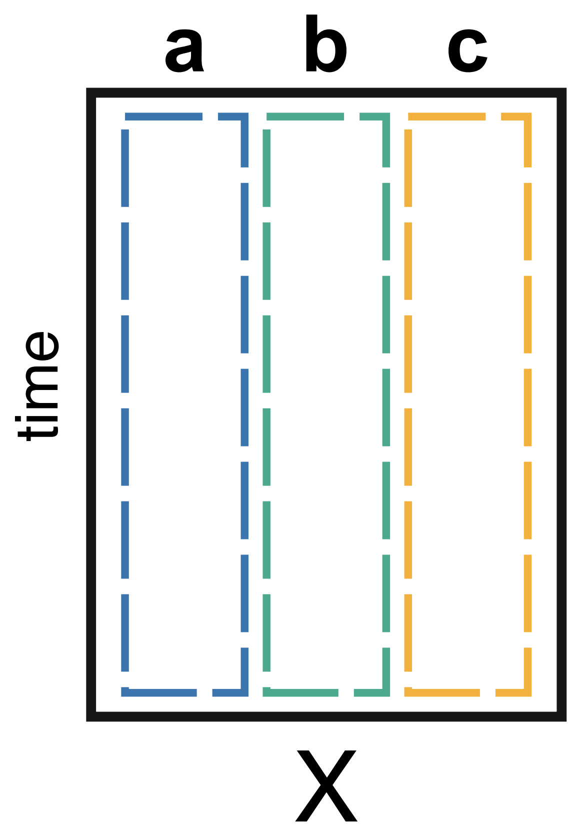

```python
import nemos as nmo

model = nmo.glm.GLM()
```

Define the model

GLM in NeMoS



Feature matrix

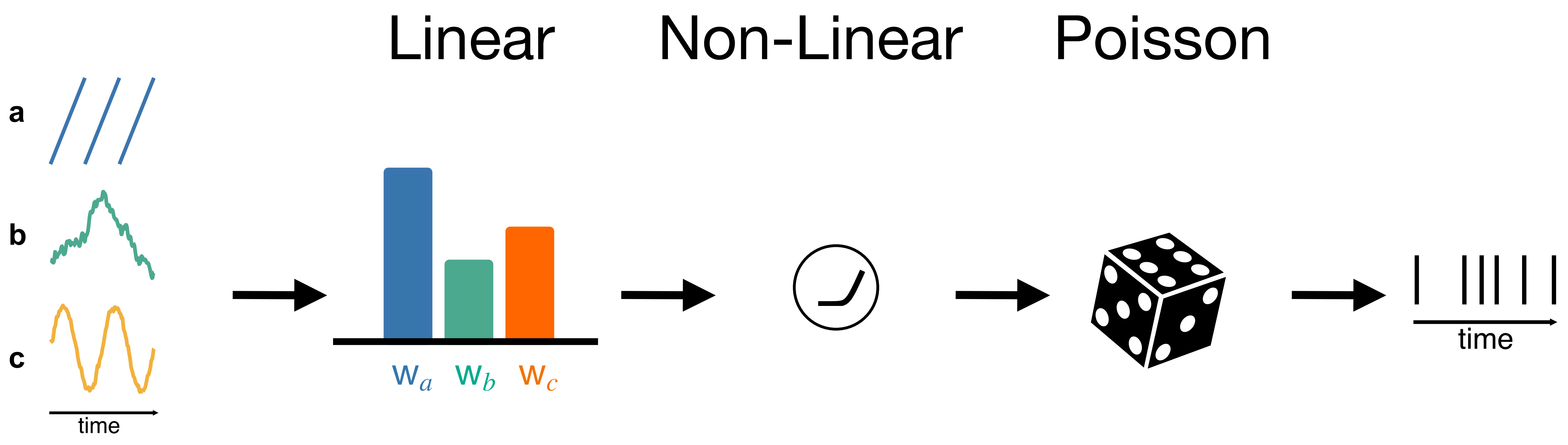


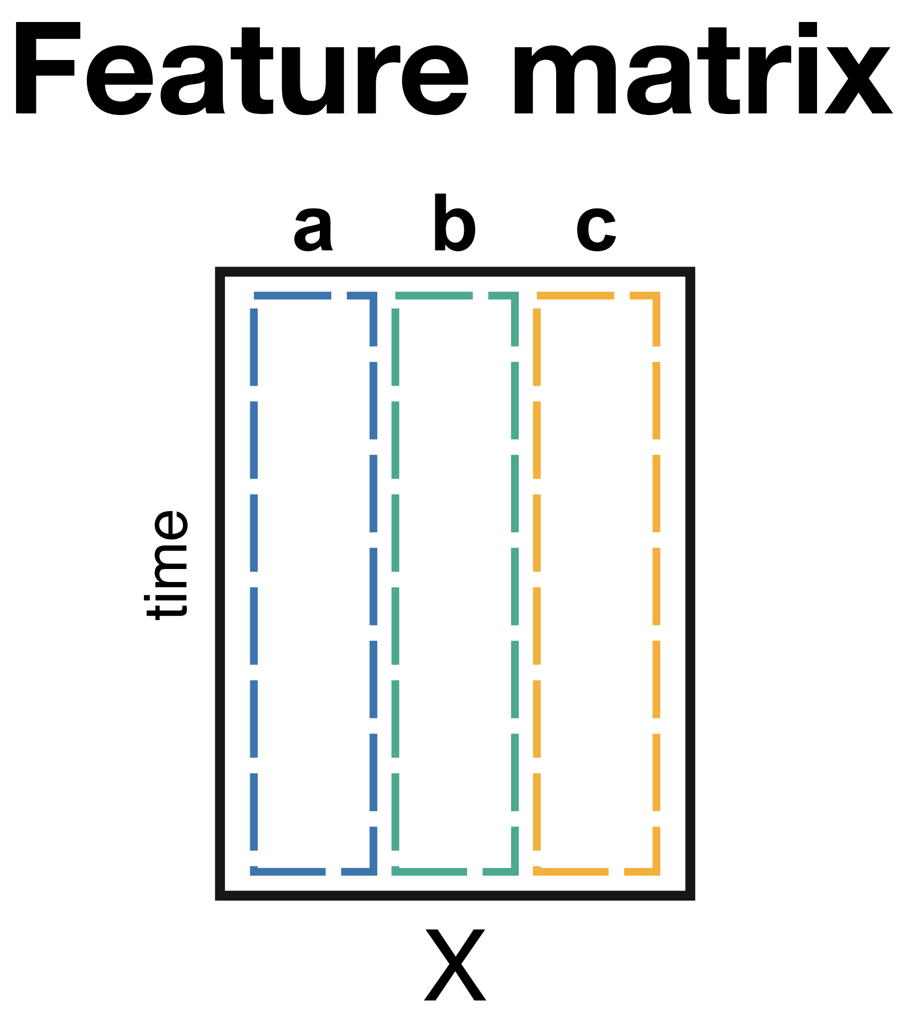

```python
import nemos as nmo

model = nmo.glm.GLM()

model.fit(X, y)
```

Define the model

Fit the GLM (learn w_a, w_b, w_c)

GLM in NeMoS






```python
import nemos as nmo

model = nmo.glm.GLM()

model.fit(X, y)

firing_rate = model.predict(X)
```

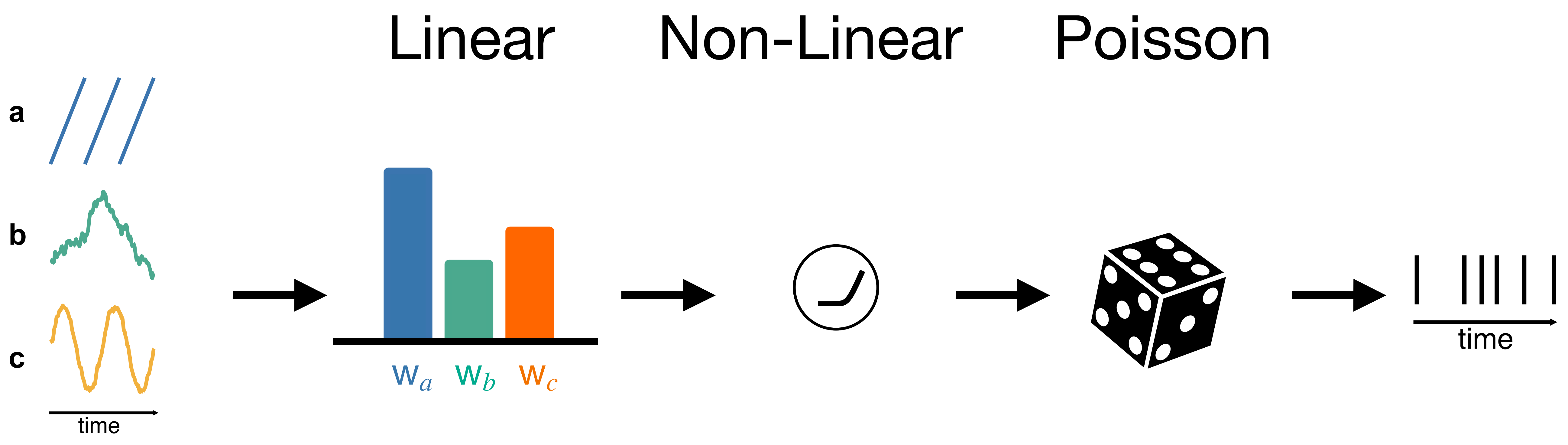
Define the model

Fit the GLM (learn w_a, w_b, w_c)

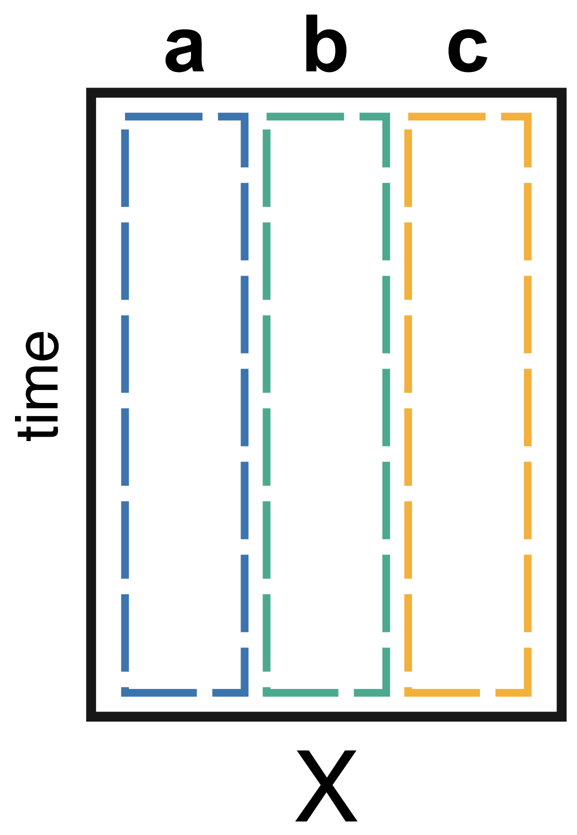
Predict the firing rate

$\exp(\, a \cdot w_a + b \cdot w_b + c \cdot w_c \,)$

GLM in NeMoS



Feature matrix




```python
import nemos as nmo

model = nmo.glm.GLM()

model.fit(X, y)

firing_rate = model.predict(X)

log_likelihood = model.score(X, y)
```

Define the model

Fit the GLM (learn w_a, w_b, w_c)

Predict the firing rate

$\exp(a \cdot w_a + b \cdot w_b + c \cdot w_c)$

Compute the log-likelihood

What features can/should I use?

- It's up to the scientist!
- Choosing features is a way to formulate hypothesis about the neural encoding.
- Any fixed (not learned) transformation of your data is valid* (counting, binning, projecting into Principal Components, filtering, squaring …)

**as long as the resulting time axis matches that of the spike counts*

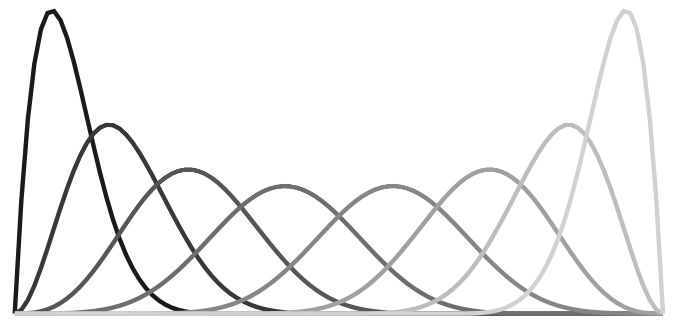
Constructing Features in NeMoS

- NeMoS provides the **basis** module for feature construction

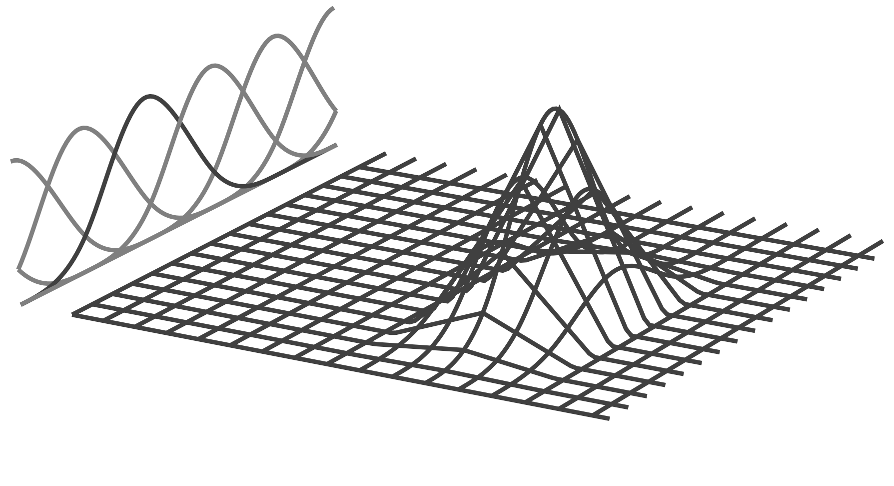
Constructing Features in NeMoS

- NeMoS provides the **basis** module for feature construction
- Basis are **fixed non-linearities**

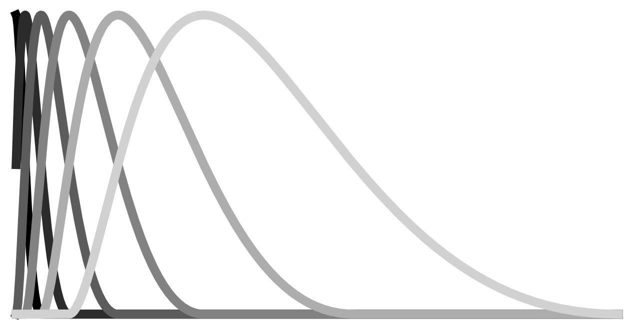
1D



2D



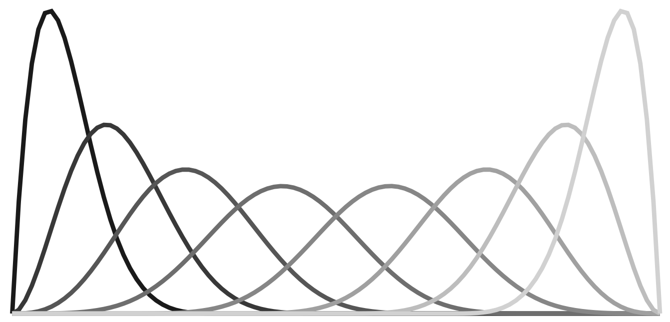
log-stretched



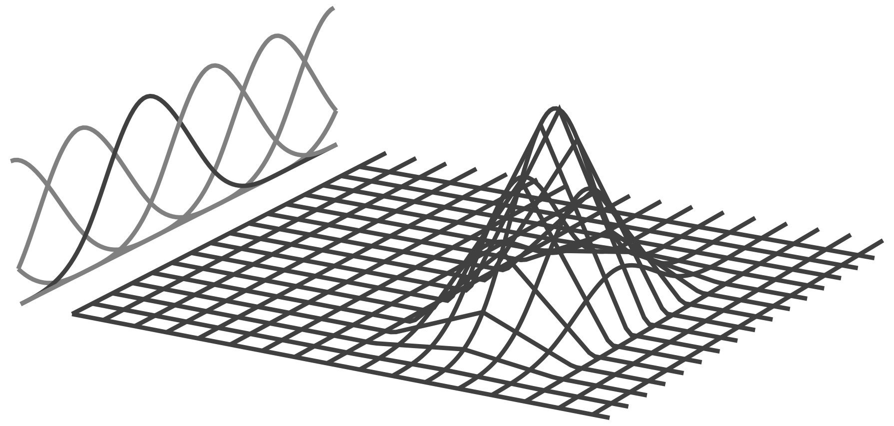
Constructing Features in NeMoS

- NeMoS provides the **basis** module for feature construction
- Basis are **fixed non-linearities**
- Assume that **firing rate varies smoothly/ gradually**

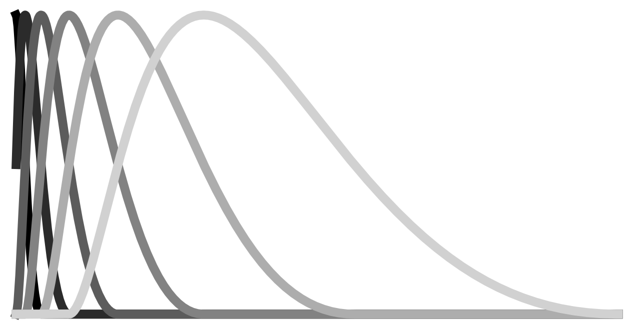
1D



2D



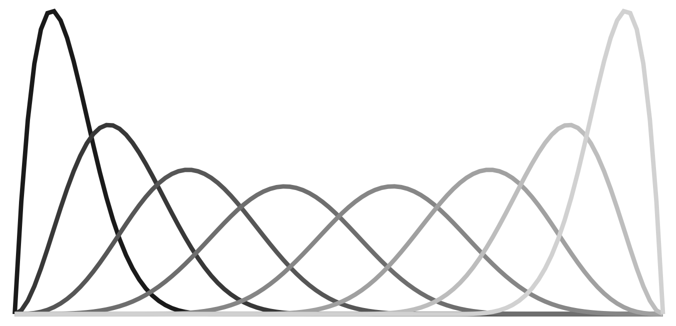
log-stretched



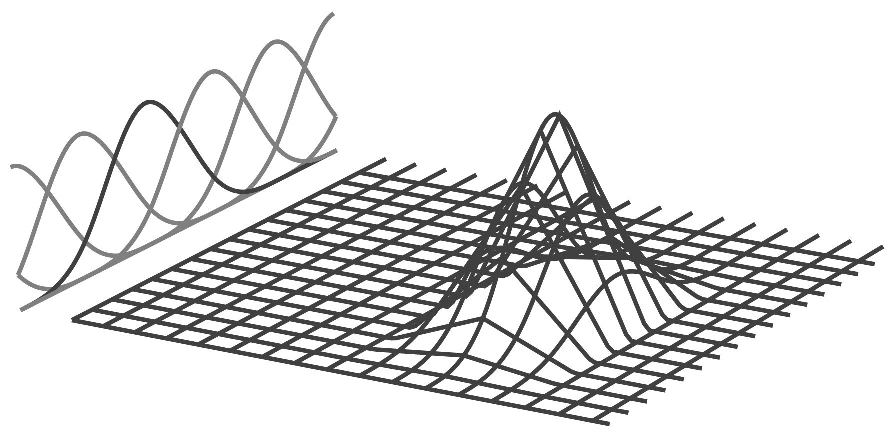
Constructing Features in NeMoS

- NeMoS provides the **basis** module for feature construction
- Basis are **fixed non-linearities**
- Assume that **firing rate varies smoothly/ gradually**
- Used for:
    1. Reducing dimensionality
    2. Non-linear firing rate modulation
    3. Time dependent effects

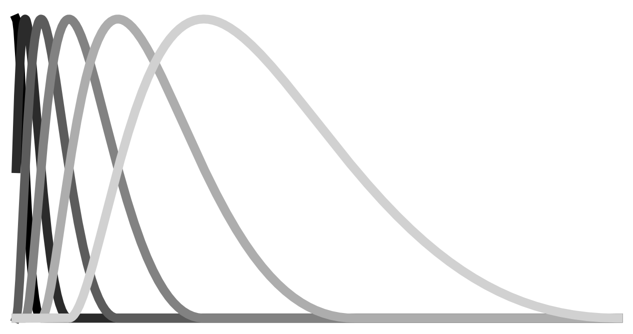
1D



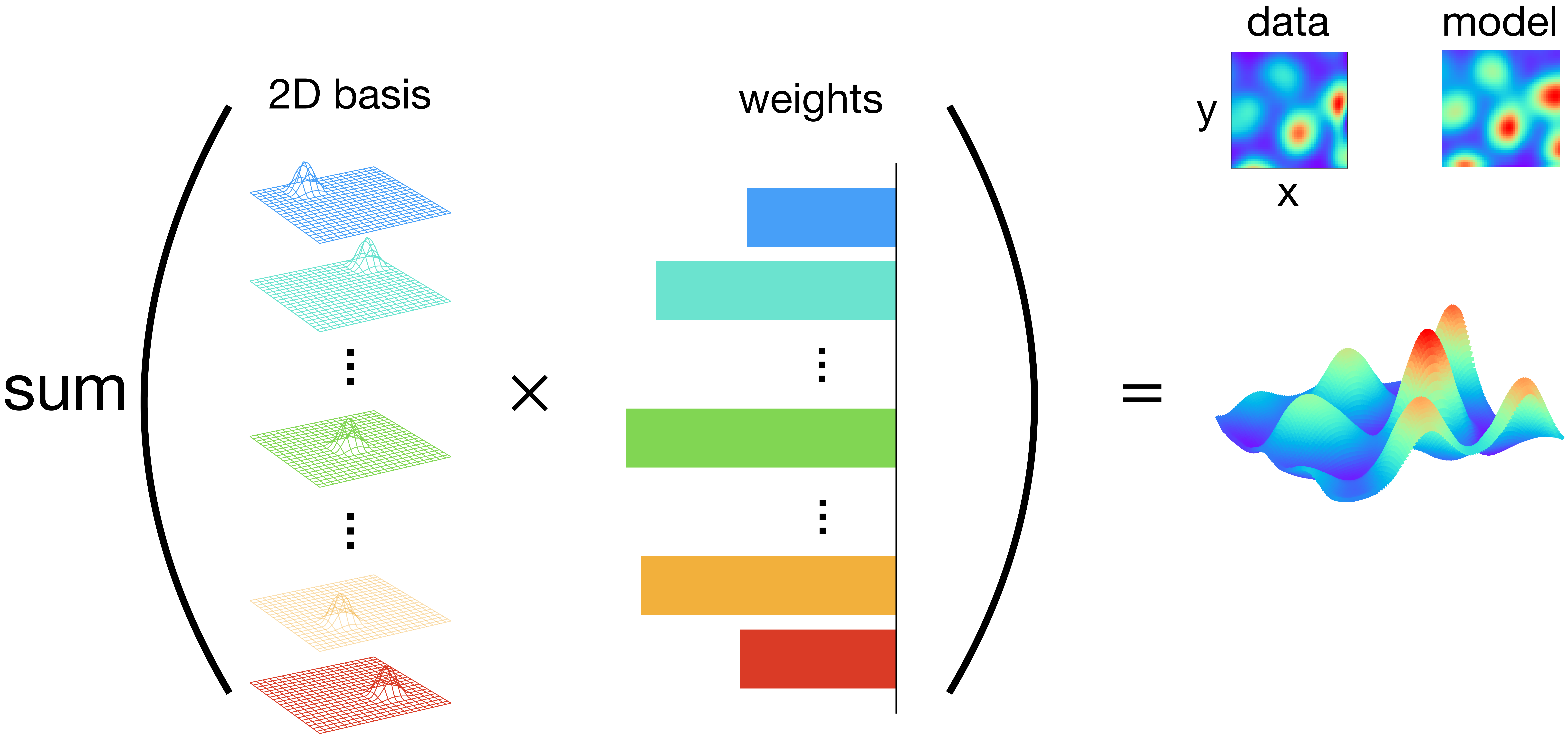
2D



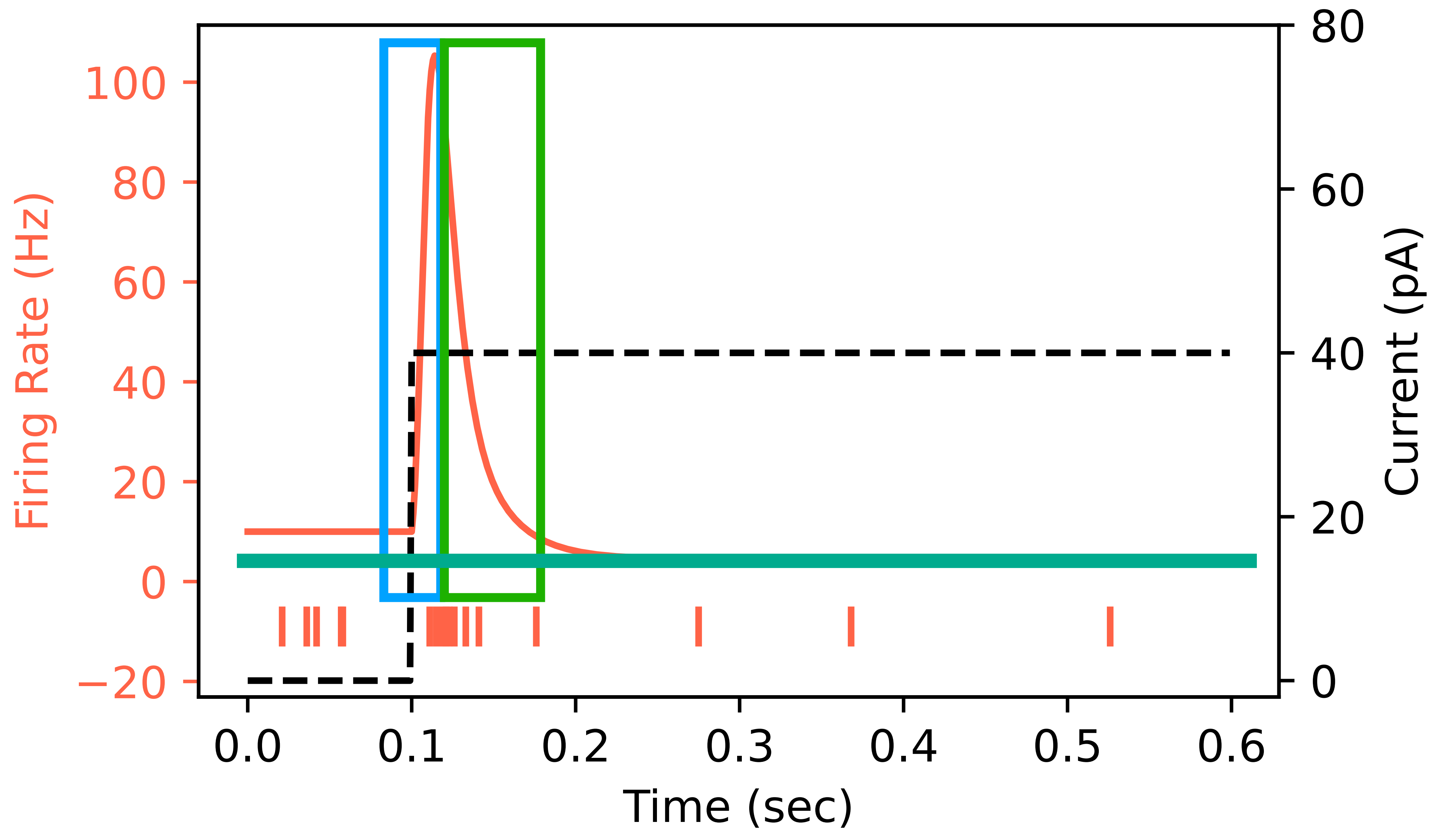
log-stretched



Example: Non-Linear Rate Map



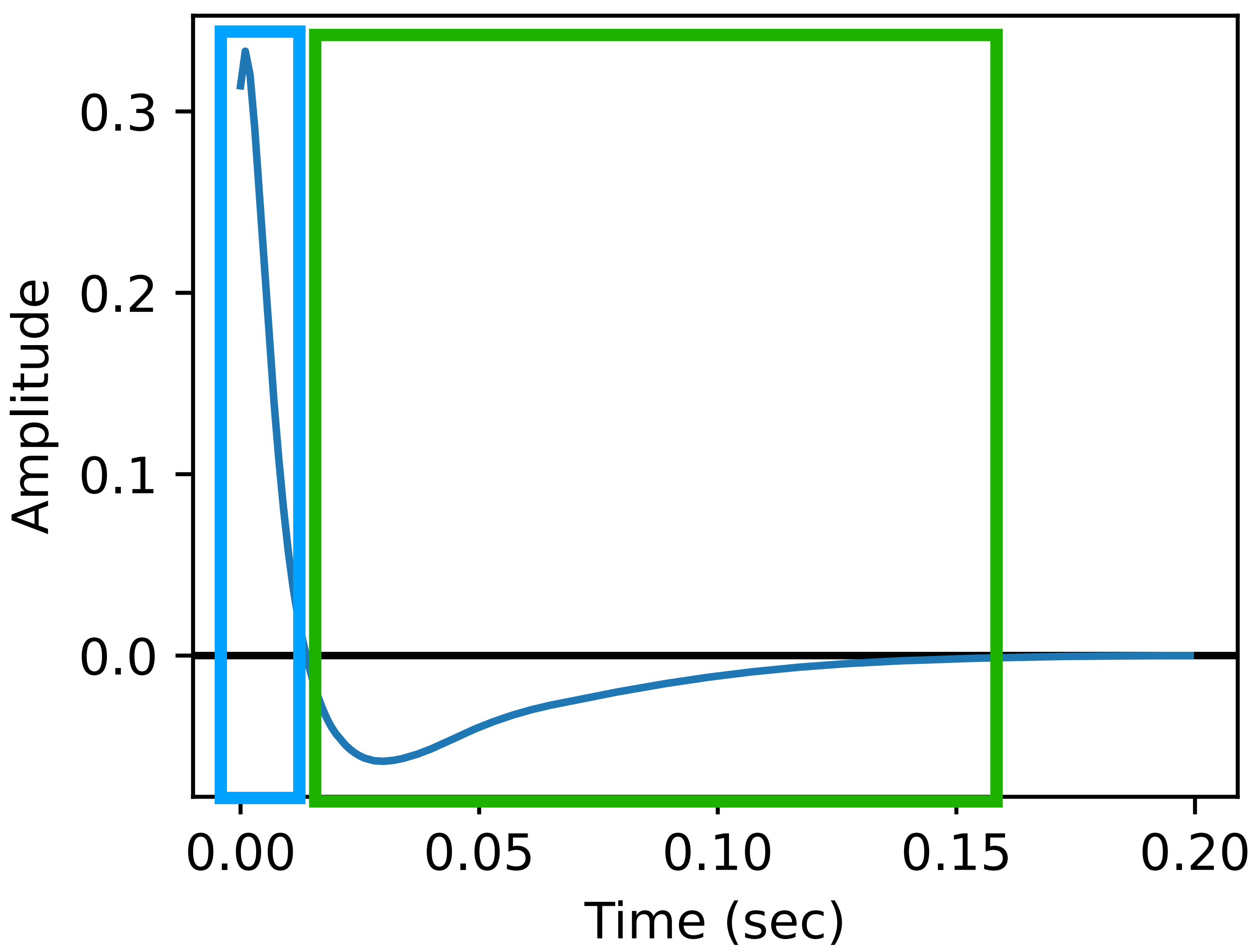
Example 2: Capturing Temporal Effects



- Input: constant current

Example 2: Capturing Temporal Effects

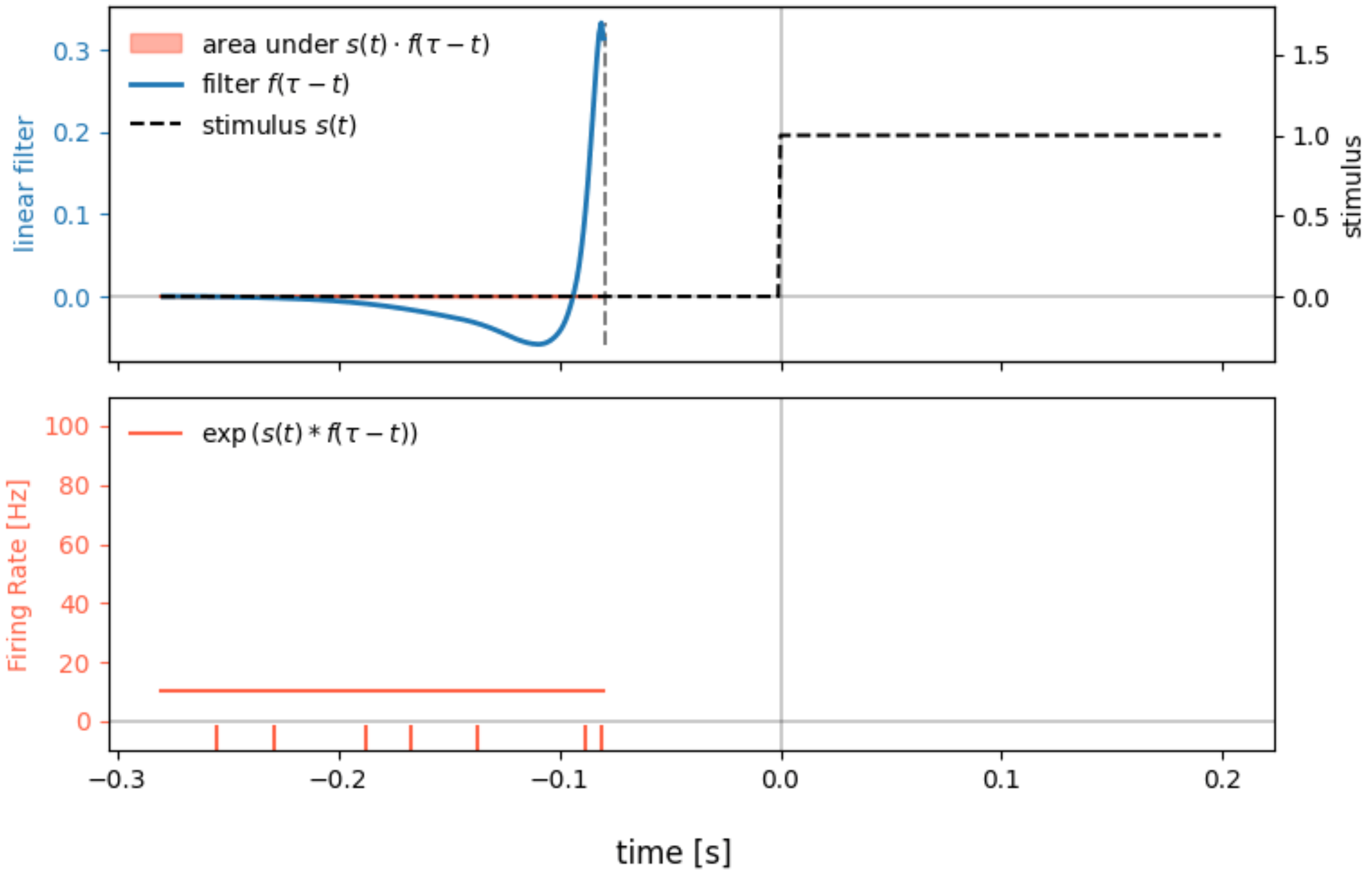
Linear filter



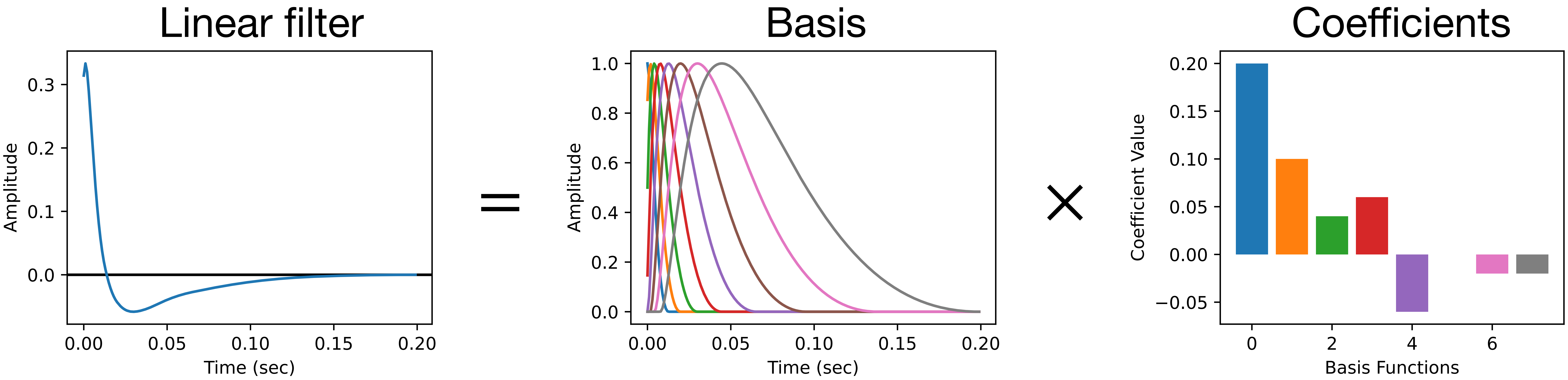
Response to a current impulse

Example 2: Capturing Temporal Effects

Linear filter convolved with the current + non linearity

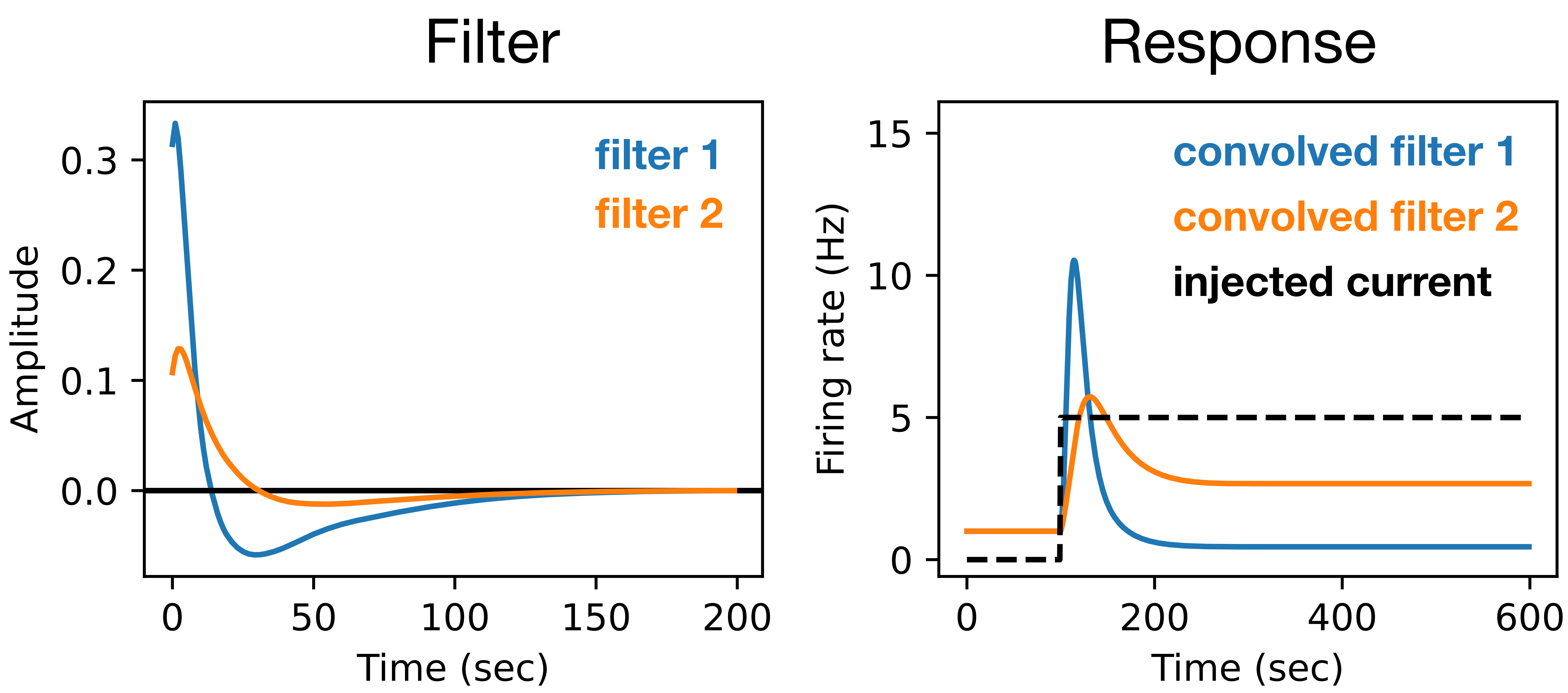


Example 2: Capturing Temporal Effects



- 1ms resolution, for 200ms window => 200 numbers to describe the filter
- With basis you need only 8 numbers

Example 2: Capturing Temporal Effects



Many different responses can be captured by a linear filter

Summary

- Tuning functions do not fully characterize neural encoding.

Summary

- Tuning functions do not fully characterize neural encoding.
- GLMs retain many of the advantageous properties of linear regression *(easy to fit, unique solution)*

Summary

- Tuning functions do not fully characterize neural encoding.
- GLMs retain many of the advantageous properties of linear regression *(easy to fit, unique solution)*
- Better suited for non-normally distributed data.

Summary

- Tuning functions do not fully characterize neural encoding.
- GLMs retain many of the advantageous properties of linear regression *(easy to fit, unique solution)*
- Better suited for non-normally distributed data.
- Rich framework: model jointly many features, flexible design…

Today's roadmap

- **Current injection notebook:**
  - Load and explore a intracellular recordings from the Allen Brain Map with pynapple.
  - Fit an LNP model.
  - Capture temporal effects using NeMoS' basis.
- **Head direction notebook**
  - Capture spike history effects with a recurrently connected GLM.
  - Functional connectivity with a coupled GLM.
- **Place cell notebook**
  - Introduction to model selection by cross-validation.
  - Model selection with NeMoS and scikit-learn.

FLATIRON INSTITUTE

Pynapple

NeMoS

Documentation Website

https://nemos.readthedocs.io/en/stable/

@nemos_neuro

https://pynapple.org/

@thepynapple