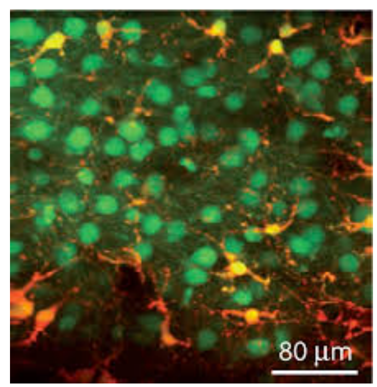
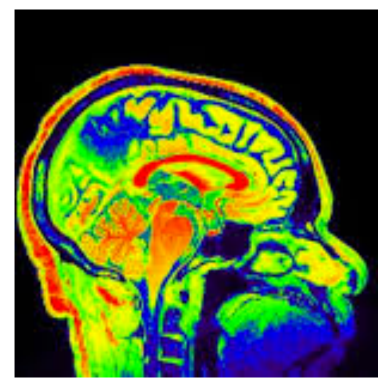
Statistical Models for Neural Data: from Regression / GLMs to Latent Variables

Jonathan Pillow

Princeton Neuroscience Institute



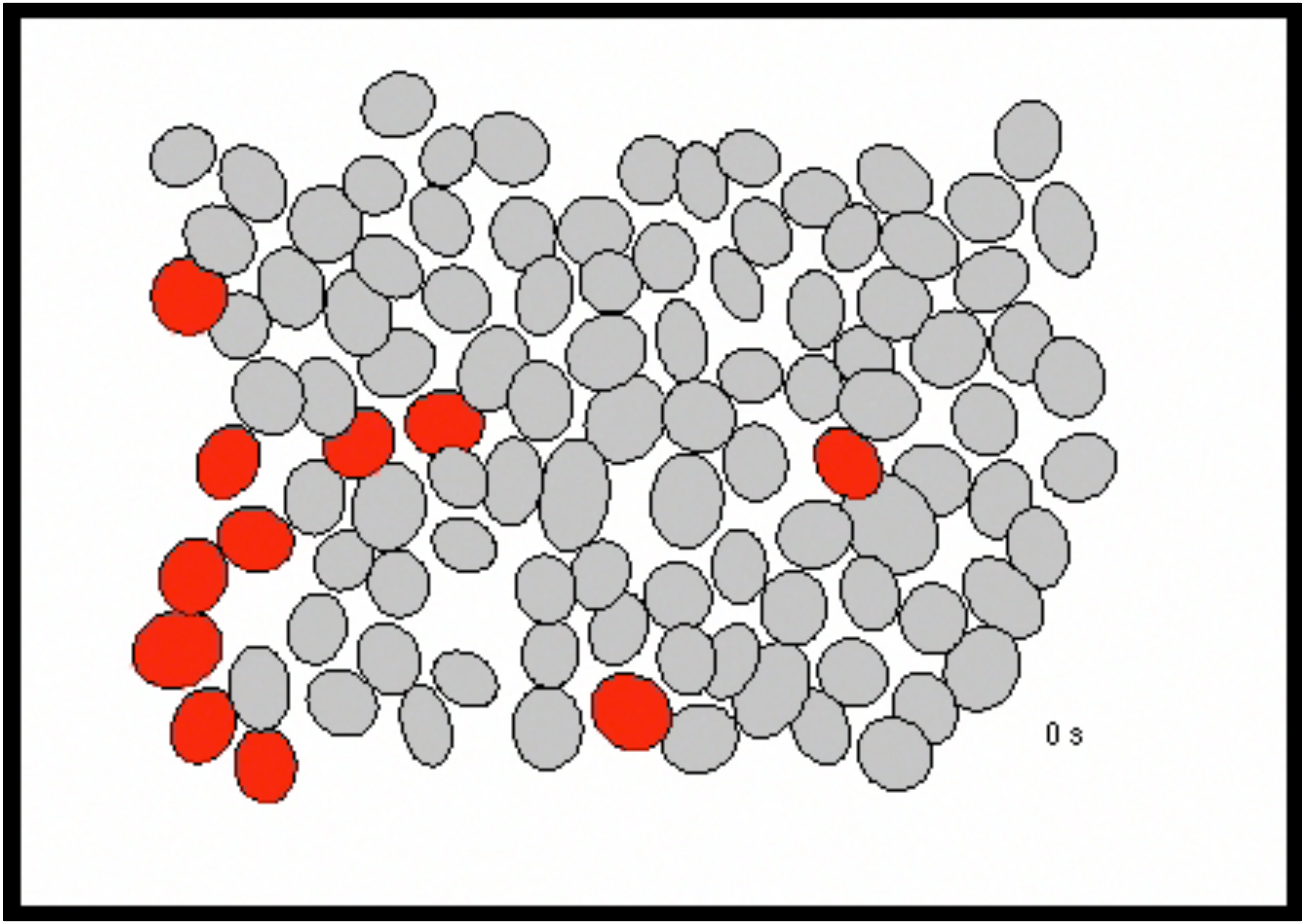


Tutorial

Cosyne 2018

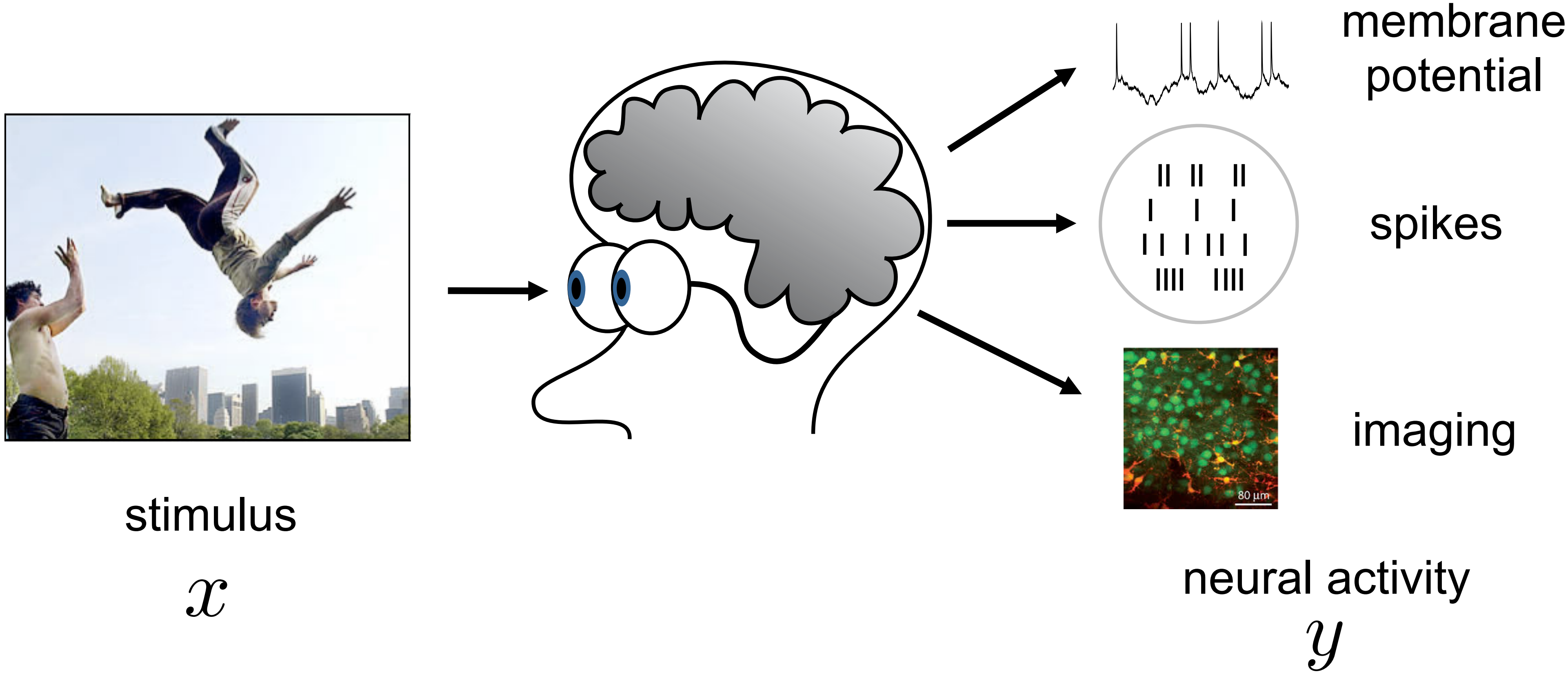
Retinal responses to white noise

(ON parasol cells)



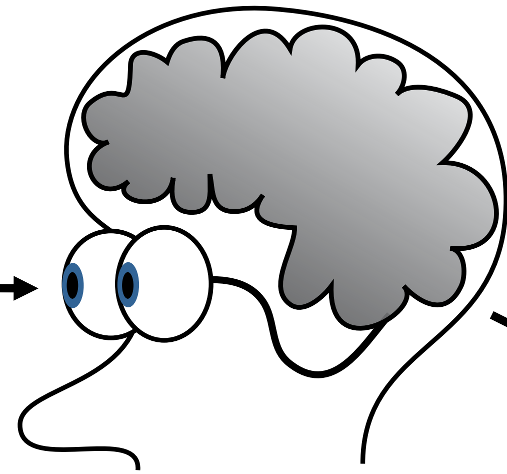
Shlens, Field, Gauthier, Greschner, Sher , Litke & Chichilnisky (2009).

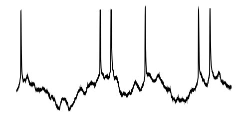
neural coding problem



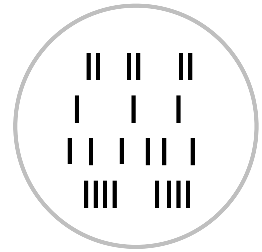
- How are stimuli and actions encoded in neural activity?
- What aspects of neural activity carry information?

neural coding problem

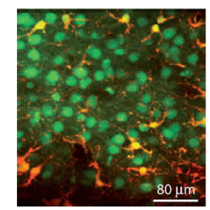




membrane
potential



spikes



imaging

stimulus

x

$P(y|x)$

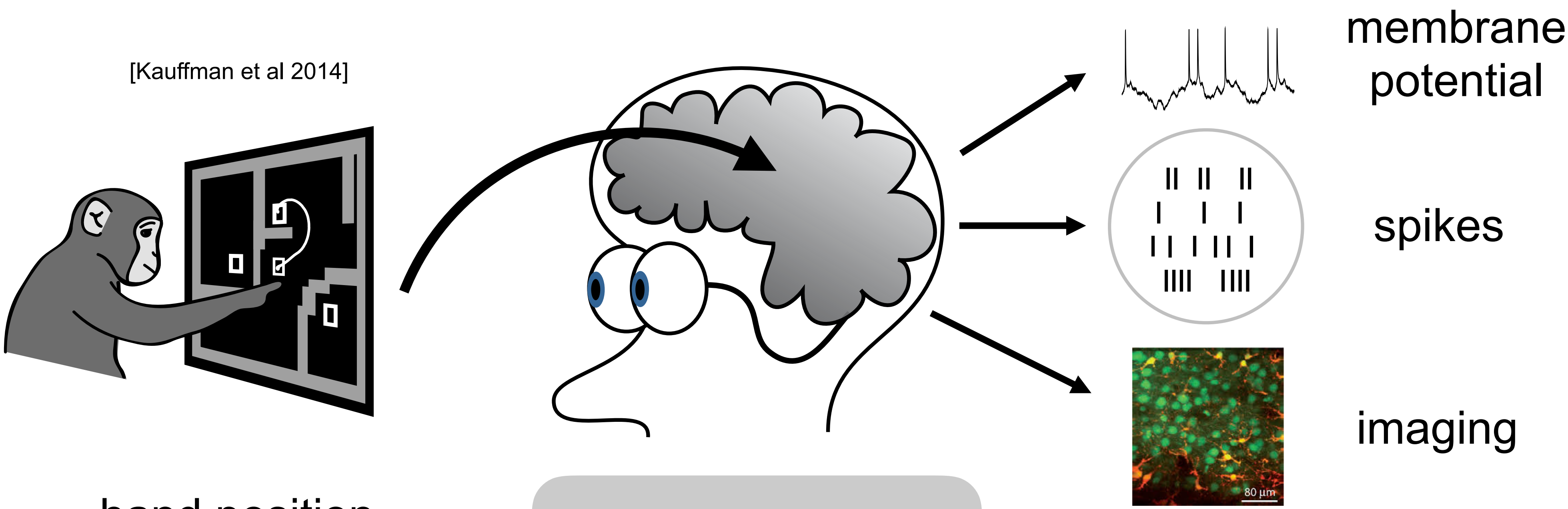
encoding models

neural activity

y

Approach:
- develop flexible statistical models of P(y|x)
- quantify information carried in neural responses

neural coding problem



hand position
x

$P(y|x)$
encoding models

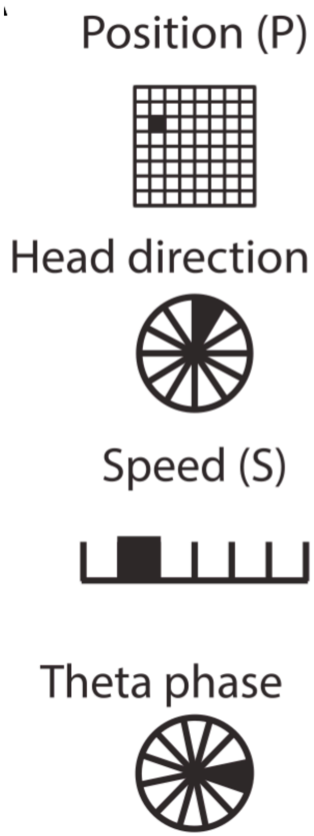
neural activity
y

“regression models”

- not restricted to sensory variables

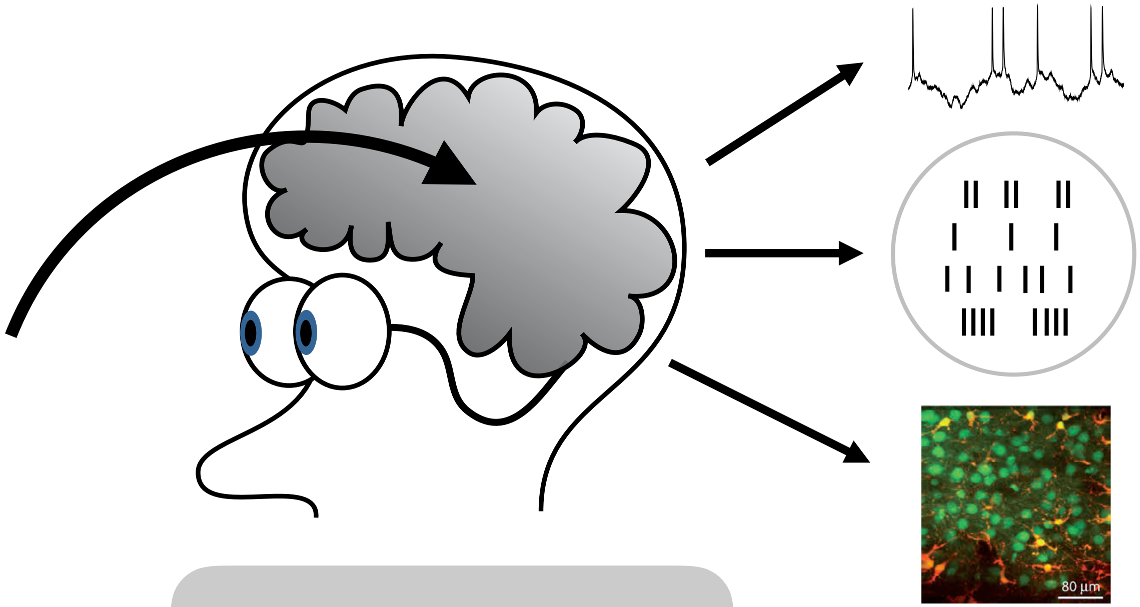
neural coding problem

[Hardcastle et al 2015]



x

"external covariates"



membrane potential

spikes

imaging

neural activity

y

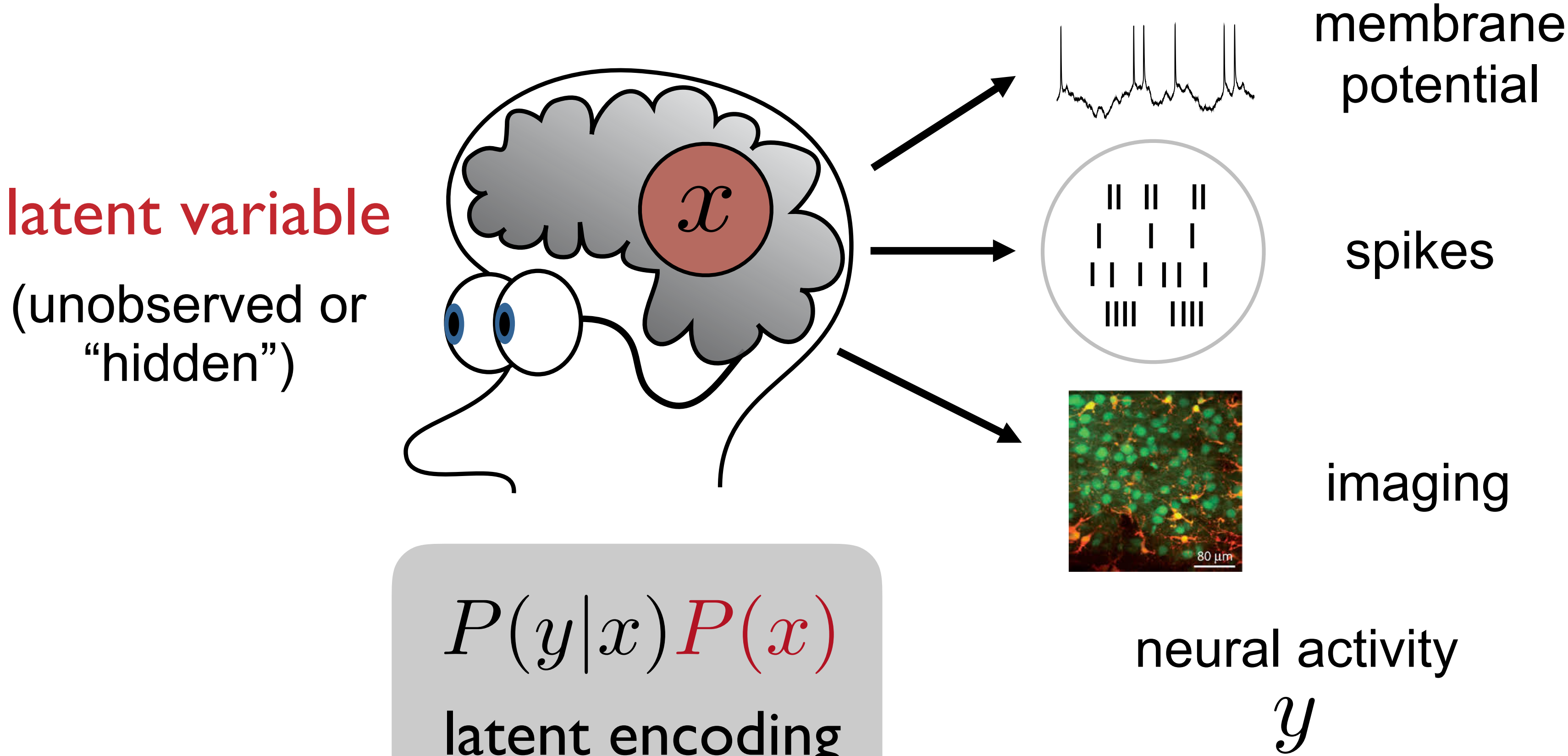
$P(y|x)$

encoding models

"regression models"

- not restricted to sensory variables

latent variable models



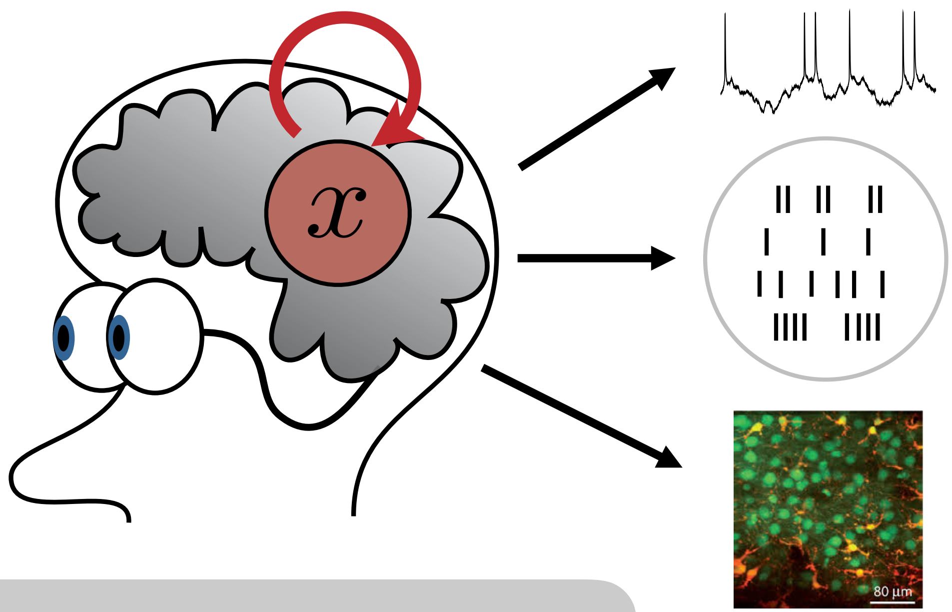
$$P(y|x)P(x)$$

latent encoding models

- capture hidden structure underlying neural activity

  (eg. low-dimensional or discrete states)

latent variable models

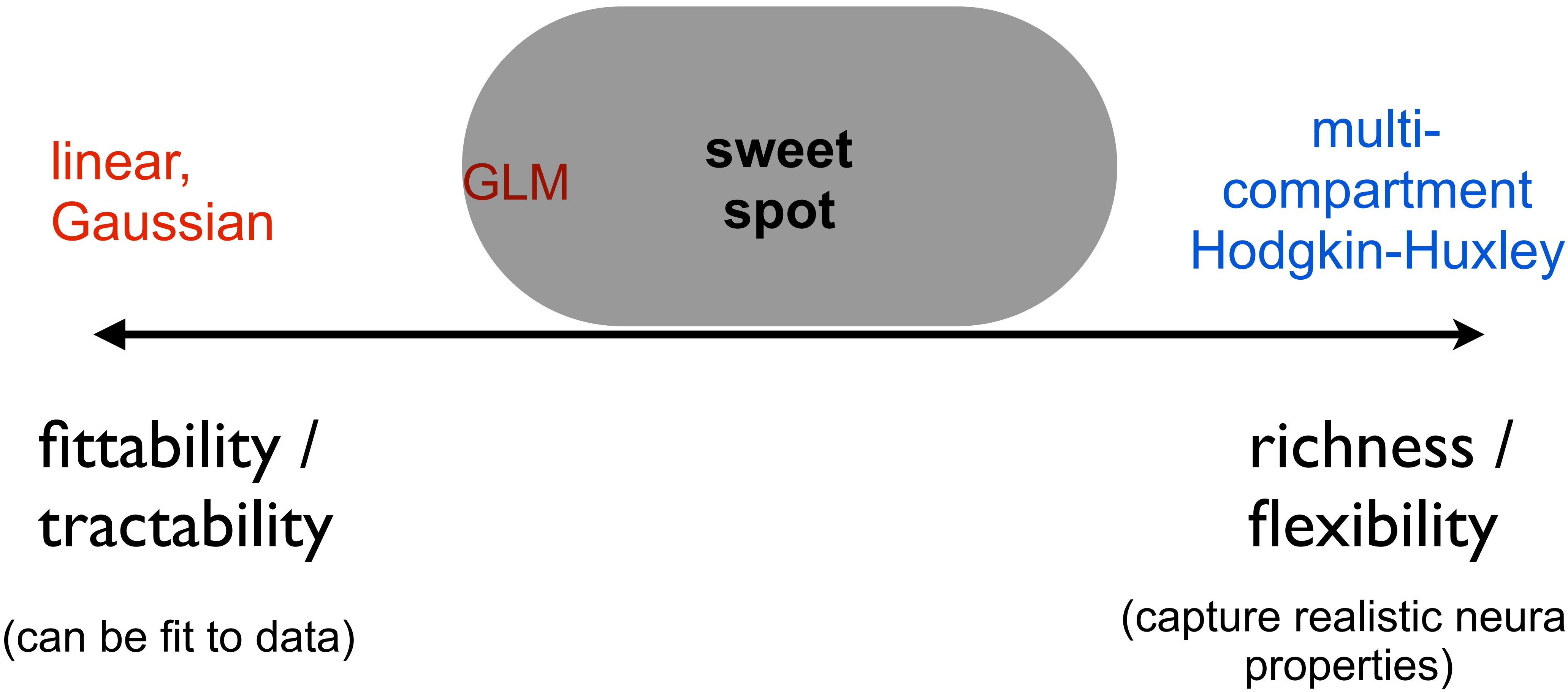


$$P(y_t|x_t)P(x_t|x_{t-1})$$

latent dynamical
encoding models

- capture hidden dynamics underlying neural activity

model desiderata



normative theories
(e.g. "efficient coding")

Why does the code take this form?

descriptive statistical models

$$P(y|x)$$

What is the code?

anatomy, biophysics

How is it implemented?

Outline

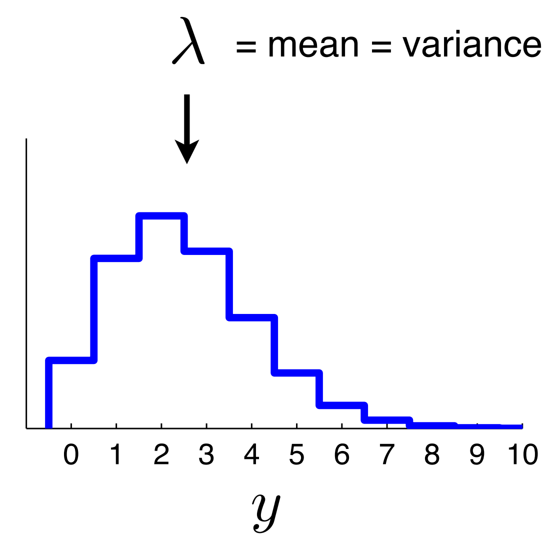
1. Spike count models & Maximum Likelihood
2. Spike train models (GLMs with spike history)
3. Multiple Spike Train Models (GLMs with coupling)
4. Regularization
5. Beyond GLM
6. Latent variable models

simple example #1: linear Poisson neuron

parameter stimulus

spike rate $\lambda = \theta x$

spike count $y \sim Poiss(\lambda)$

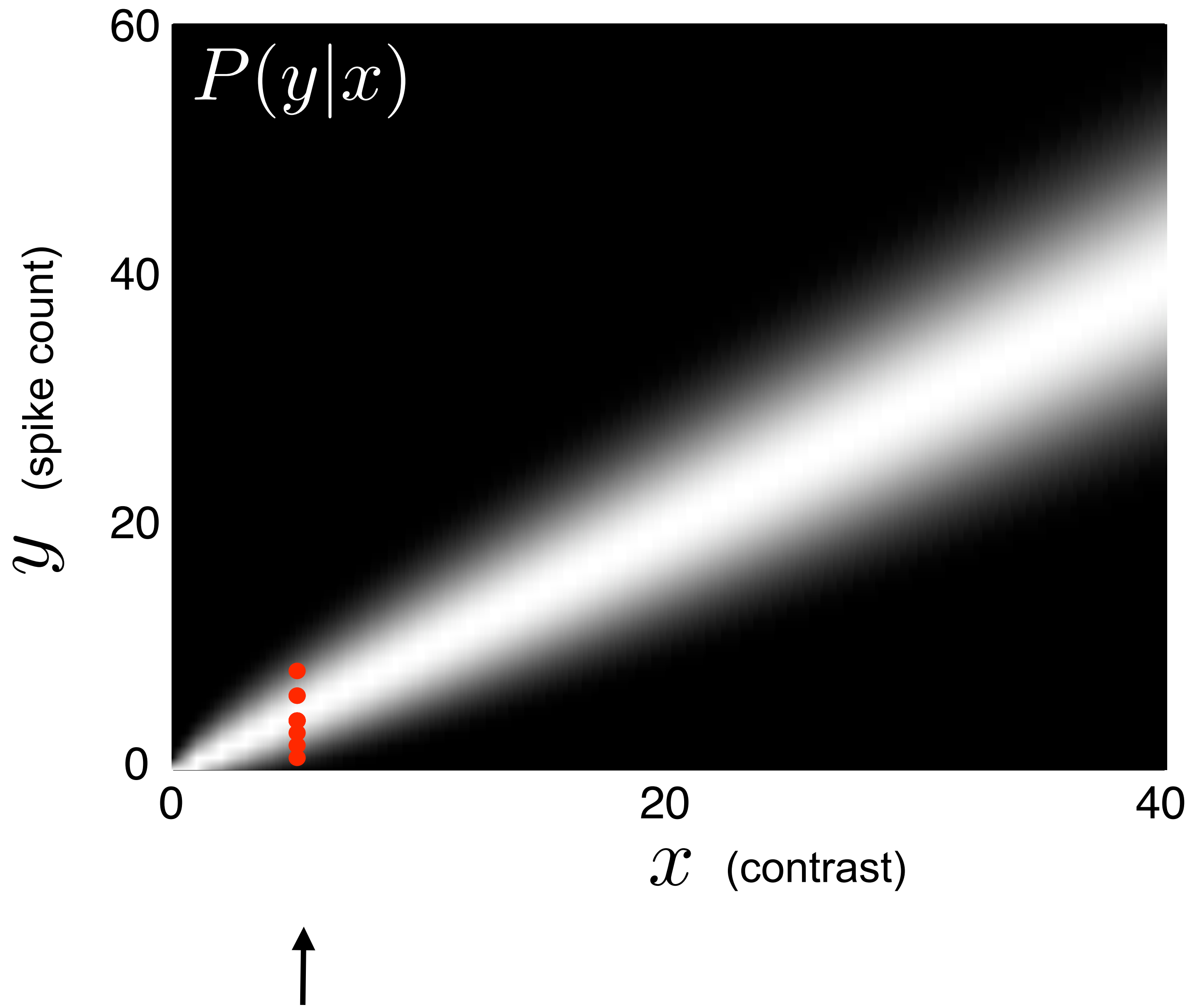


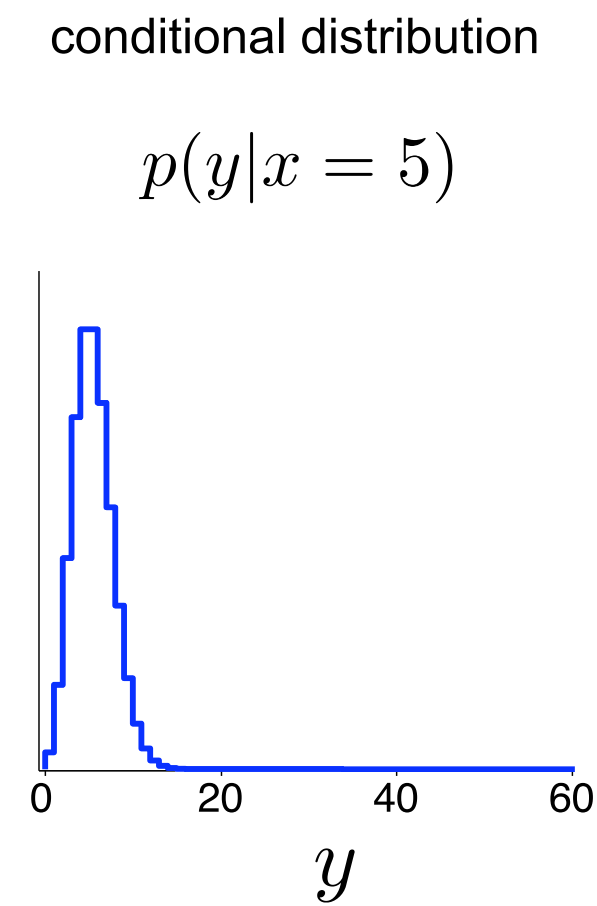
encoding model:

$$P(y|x,\theta) = \frac{1}{y!}\lambda^y e^{-\lambda}$$

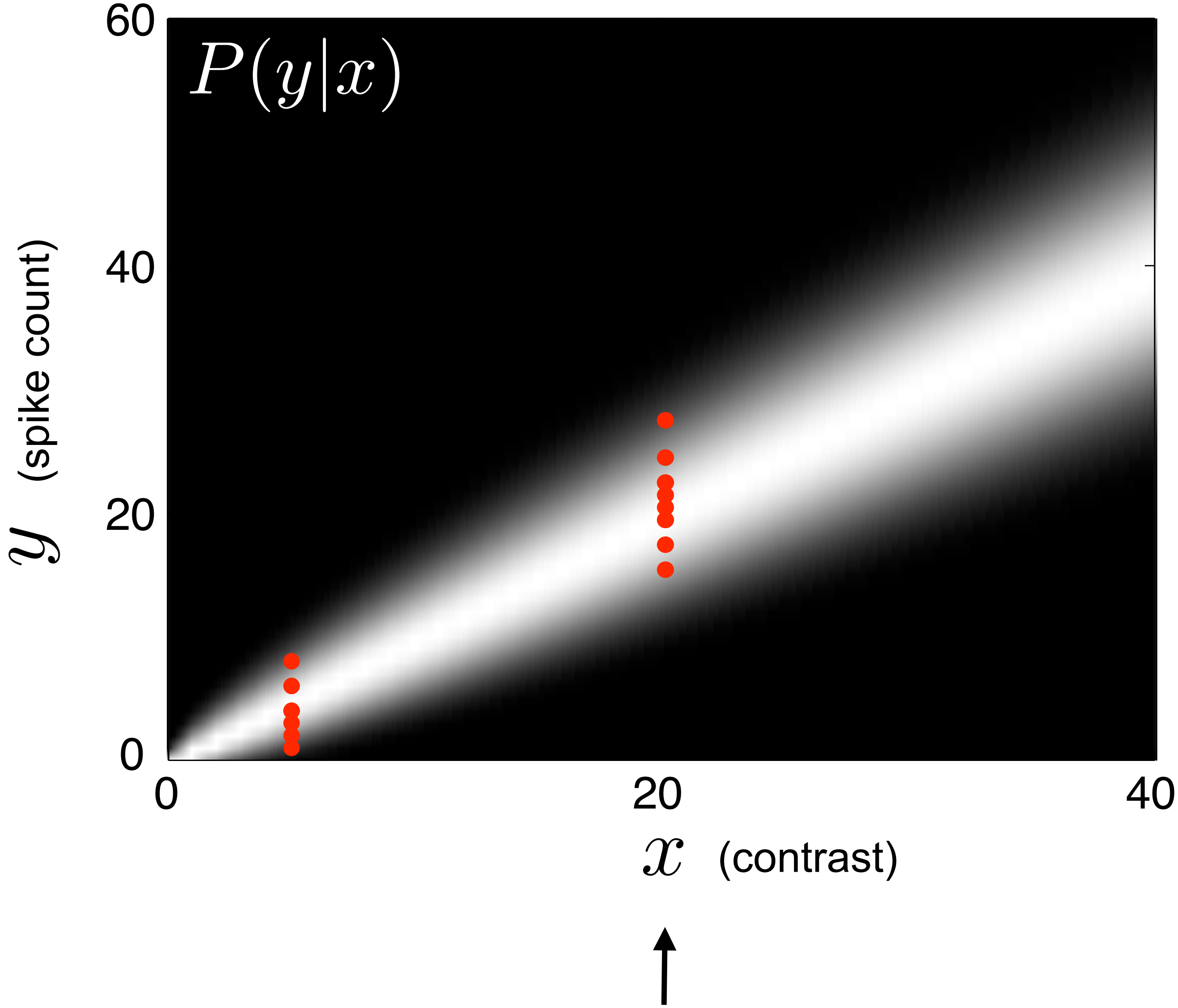
$$= \frac{1}{y!}(\theta x)^y e^{-(\theta x)}$$

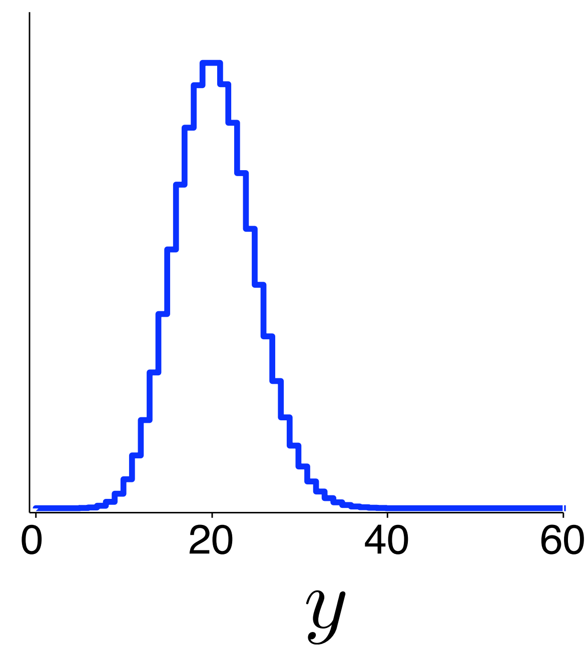
$$\mathrm{mean}(y) = \theta x$$
$$\mathrm{var}(y) = \theta x$$



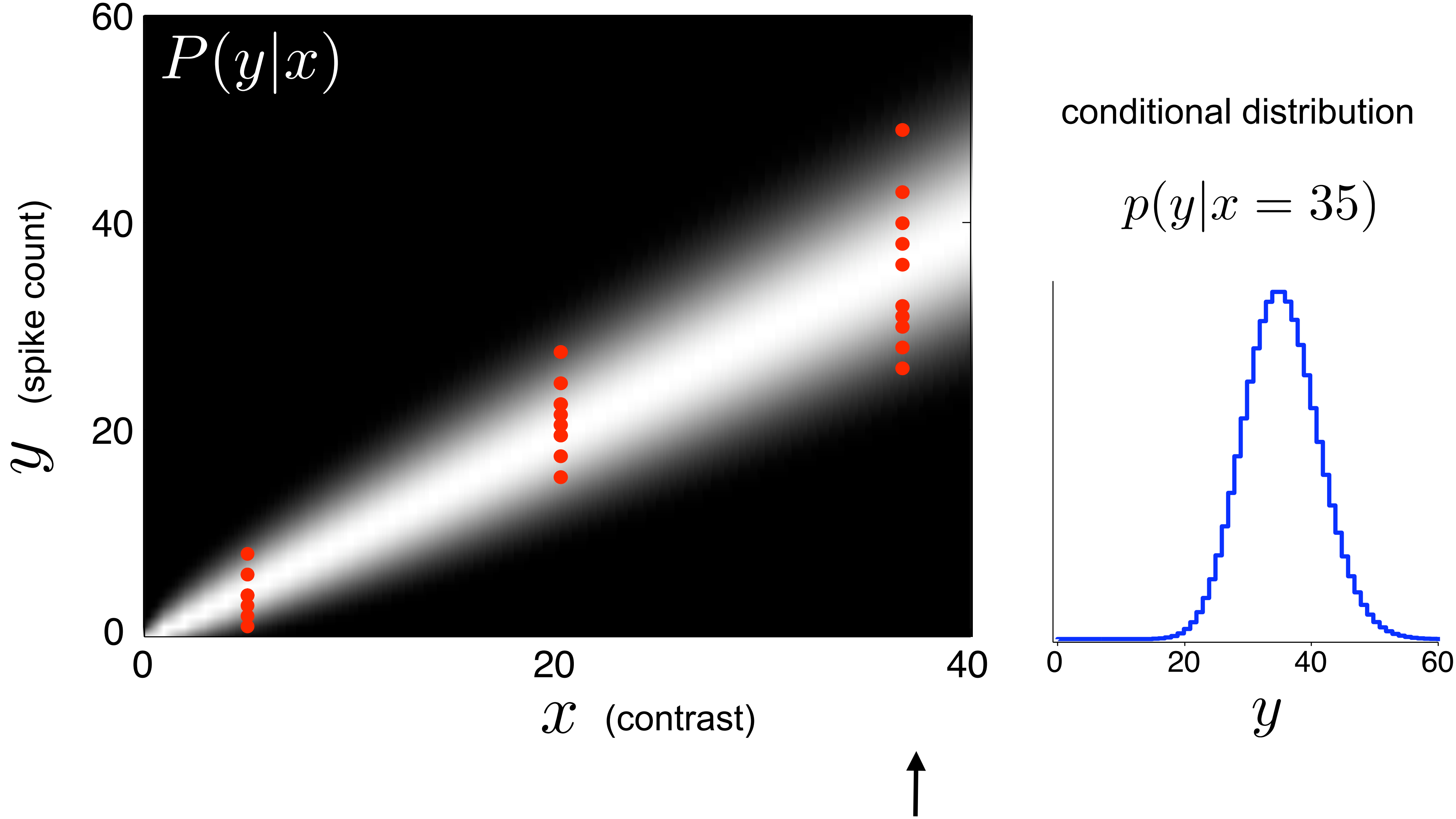


$$\mathrm{mean}(y) = \theta x$$
$$\mathrm{var}(y) = \theta x$$





$$\mathrm{mean}(y) = \theta x$$
$$\mathrm{var}(y) = \theta x$$



Maximum Likelihood Estimation:

- given observed data (Y, X), find θ that maximizes $P(Y|X,\theta)$

all spike counts

all stimuli

parameters

$$P(Y|X,\theta) = \prod_{i=1}^{N} \underbrace{P(y_i|x_i,\theta)}_{\text{single-trial probability}}$$

Q: what assumption are we making about the responses?

A: conditional independence across trials!

Maximum Likelihood Estimation:

- given observed data (Y, X), find θ that maximizes $P(Y|X, \boxed{\theta})$

all spike counts

all stimuli

parameters

$$P(Y|X,\theta) = \prod_{i=1}^{N} \underbrace{P(y_i|x_i,\theta)}_{\text{single-trial probability}}$$

Q: what assumption are we making about the responses?

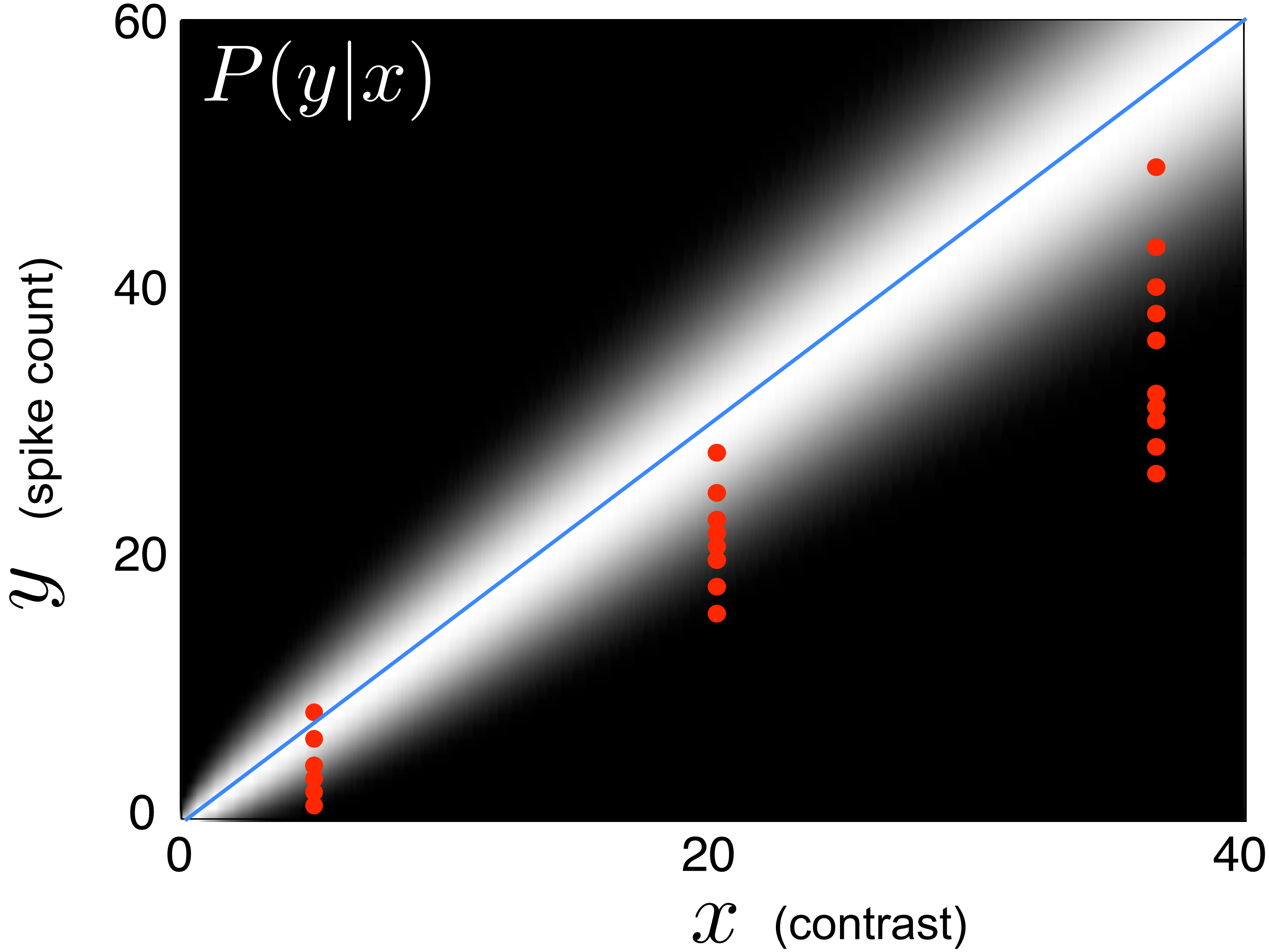
A: conditional independence across trials!

Q: when do we call $P(Y|X,\theta)$ a *likelihood*?

A: when considering it as a function of θ !

Maximum Likelihood Estimation:

- given observed data (Y, X), find θ that maximizes $P(Y|X, \theta)$



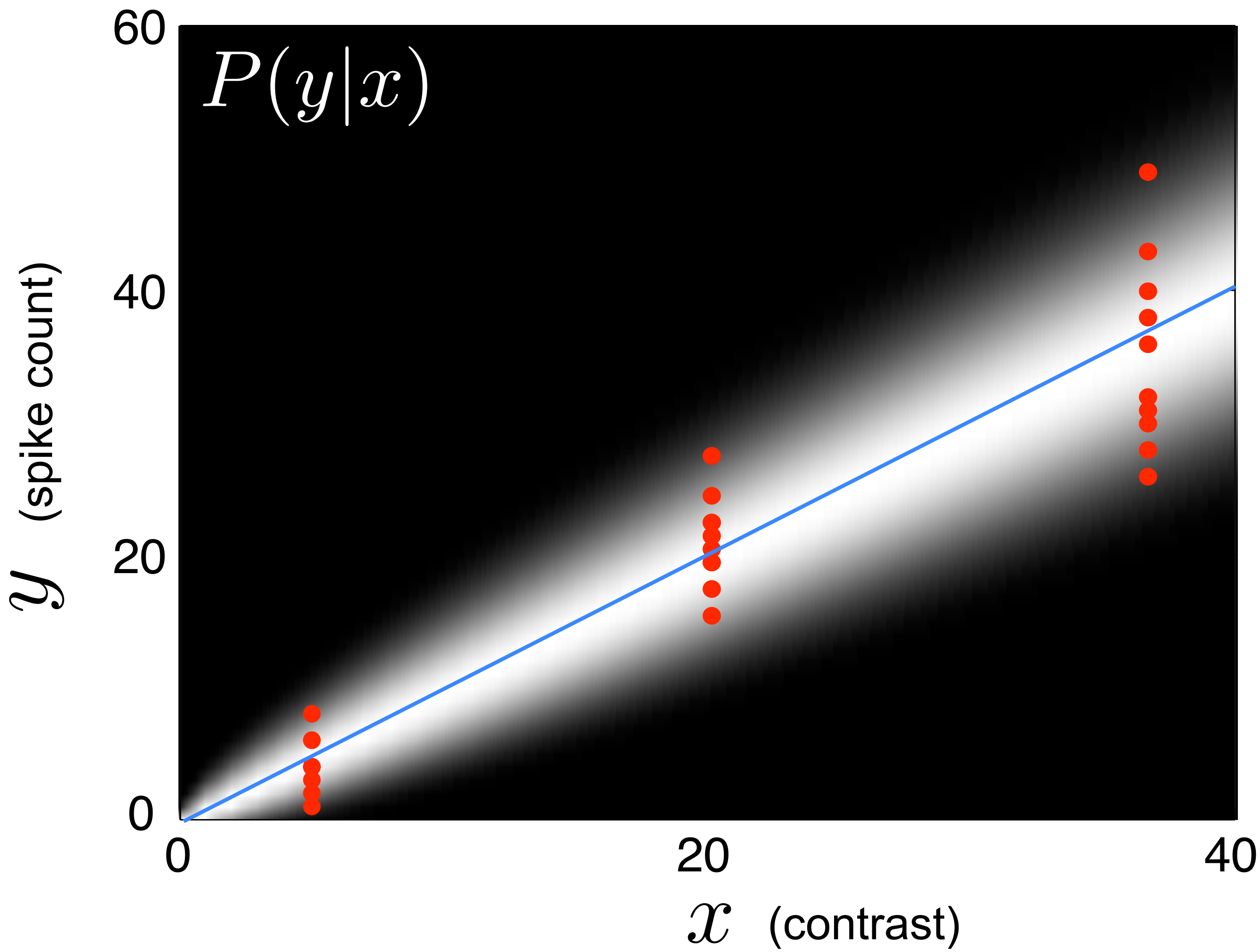
$y \sim \mathrm{Poiss}(\theta x)$

$\theta = 1.5$

- could in theory do this by turning a knob

Maximum Likelihood Estimation:

- given observed data (Y, X), find θ that maximizes $P(Y|X,\theta)$



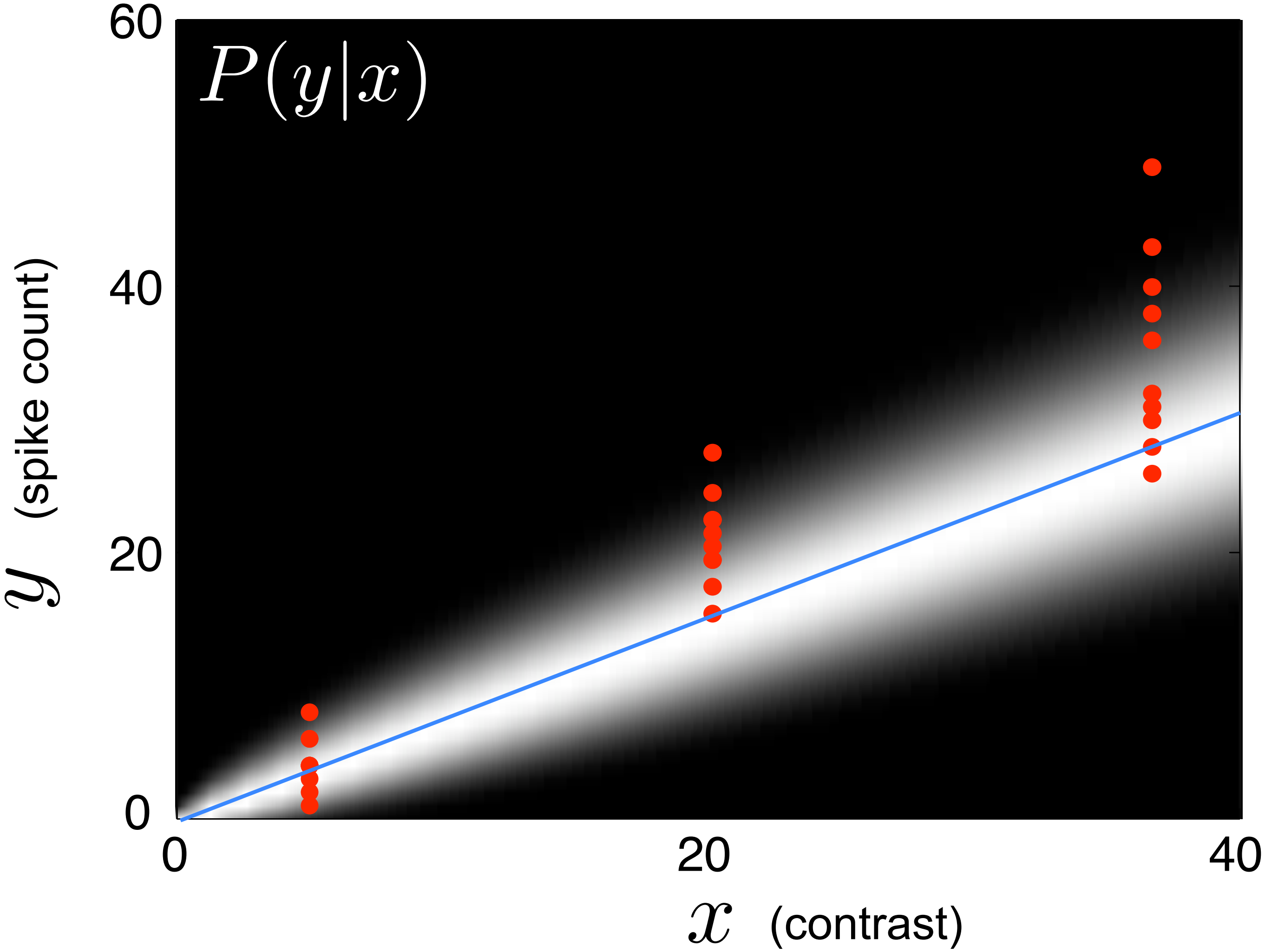
$y \sim \mathrm{Poiss}(\theta x)$

$\theta = 1$

- could in theory do this by turning a knob

Maximum Likelihood Estimation:

- given observed data (Y, X), find θ that maximizes $P(Y|X, \theta)$

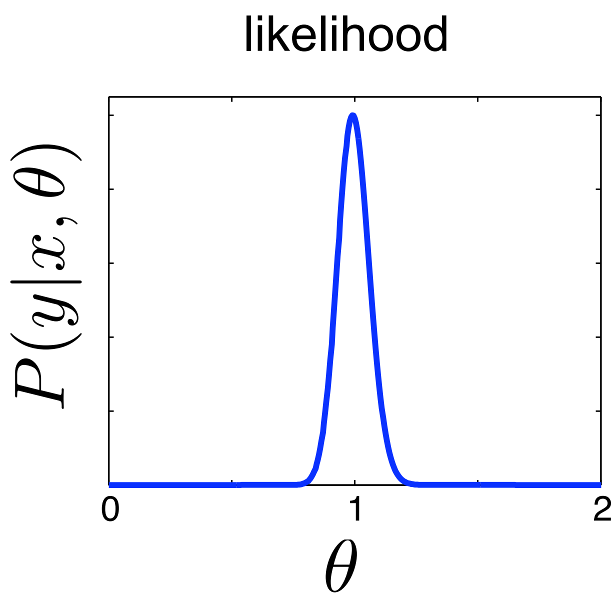


$y \sim \mathrm{Poiss}(\theta x)$

$\theta = 0.5$

- could in theory do this by turning a knob

Likelihood function: $P(Y|X,\theta)$ as a function of θ



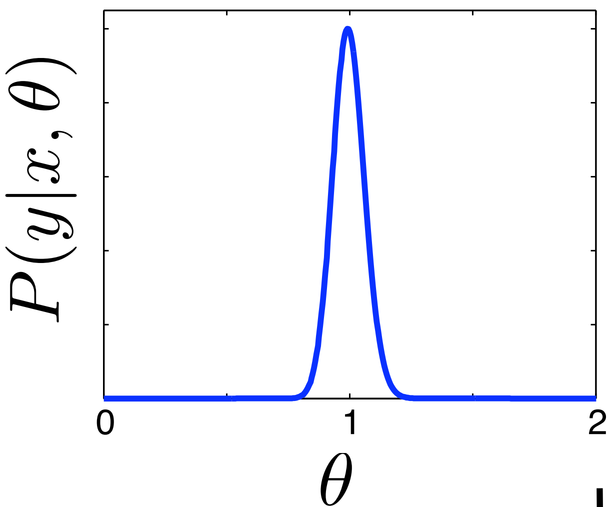
Because data are independent:

$$P(Y|X,\theta) = \prod_i P(y_i|x_i,\theta)$$

$$= \prod \frac{1}{y_i!}(\theta x_i)^{y_i} e^{-(\theta x_i)}$$

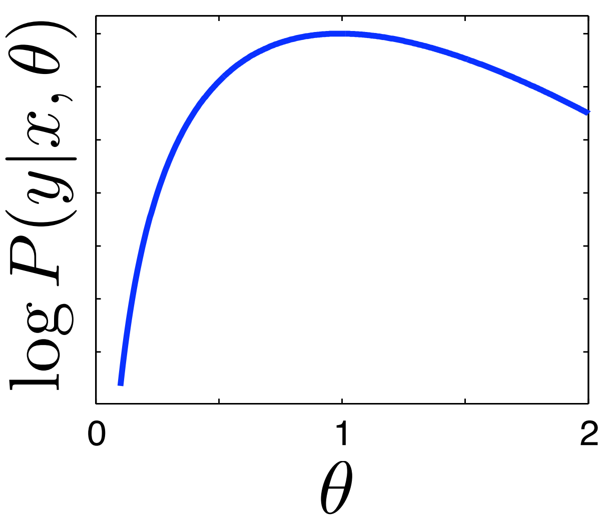
Likelihood function: $P(Y|X,\theta)$ as a function of θ

likelihood



log

log-likelihood



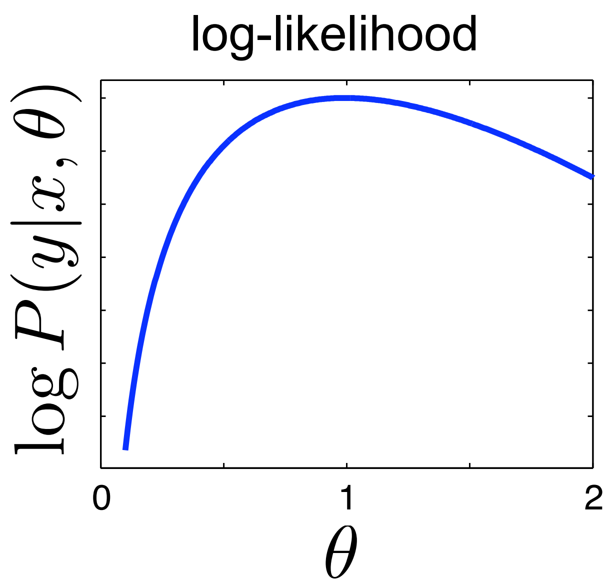
Because data are independent:

$$P(Y|X,\theta) = \prod_i P(y_i|x_i,\theta)$$

$$= \prod \frac{1}{y_i!}(\theta x_i)^{y_i} e^{-(\theta x_i)}$$

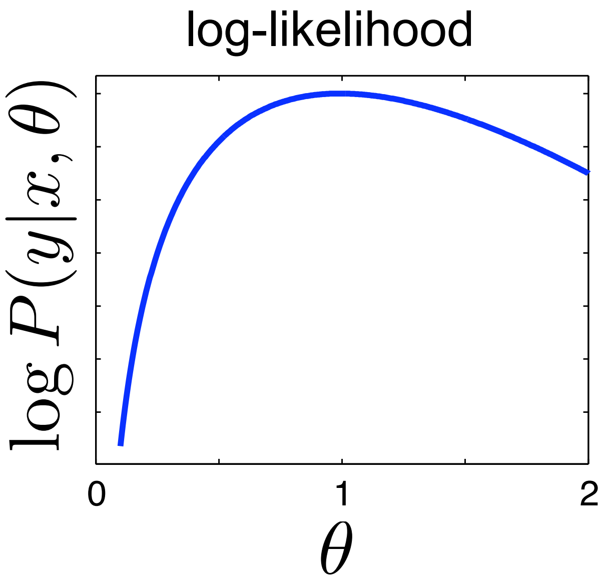
$$\log P(Y|X,\theta) = \sum_i \log P(y_i|x_i,\theta)$$

$$= \sum y_i \log\theta - \theta x_i + c$$



$$\log P(Y|X,\theta) = \sum_i \log P(y_i|x_i,\theta)$$
$$= \sum y_i \log \theta - \theta x_i + c$$
$$= \log \theta (\sum y_i) - \theta (\sum x_i)$$

Do it: solve for θ



$$\log P(Y|X,\theta) = \sum_i \log P(y_i|x_i,\theta)$$
$$= \sum y_i \log\theta - \theta x_i + c$$
$$= \log\theta(\sum y_i) - \theta(\sum x_i)$$

- Closed-form solution when model in “exponential family”

$$\frac{d}{d\theta}\log P(Y|X,\theta) = \frac{1}{\theta}\sum y_i - \sum x_i = 0$$
$$\implies \quad \hat{\theta}_{ML} = \frac{\sum y_i}{\sum x_i}$$

Properties of the MLE (maximum likelihood estimator)

- consistent
  (converges to true θ in limit of infinite data)

- efficient
  (converges as quickly as possible,
  i.e., achieves minimum possible asymptotic error)

simple example #2: linear Gaussian neuron

parameter

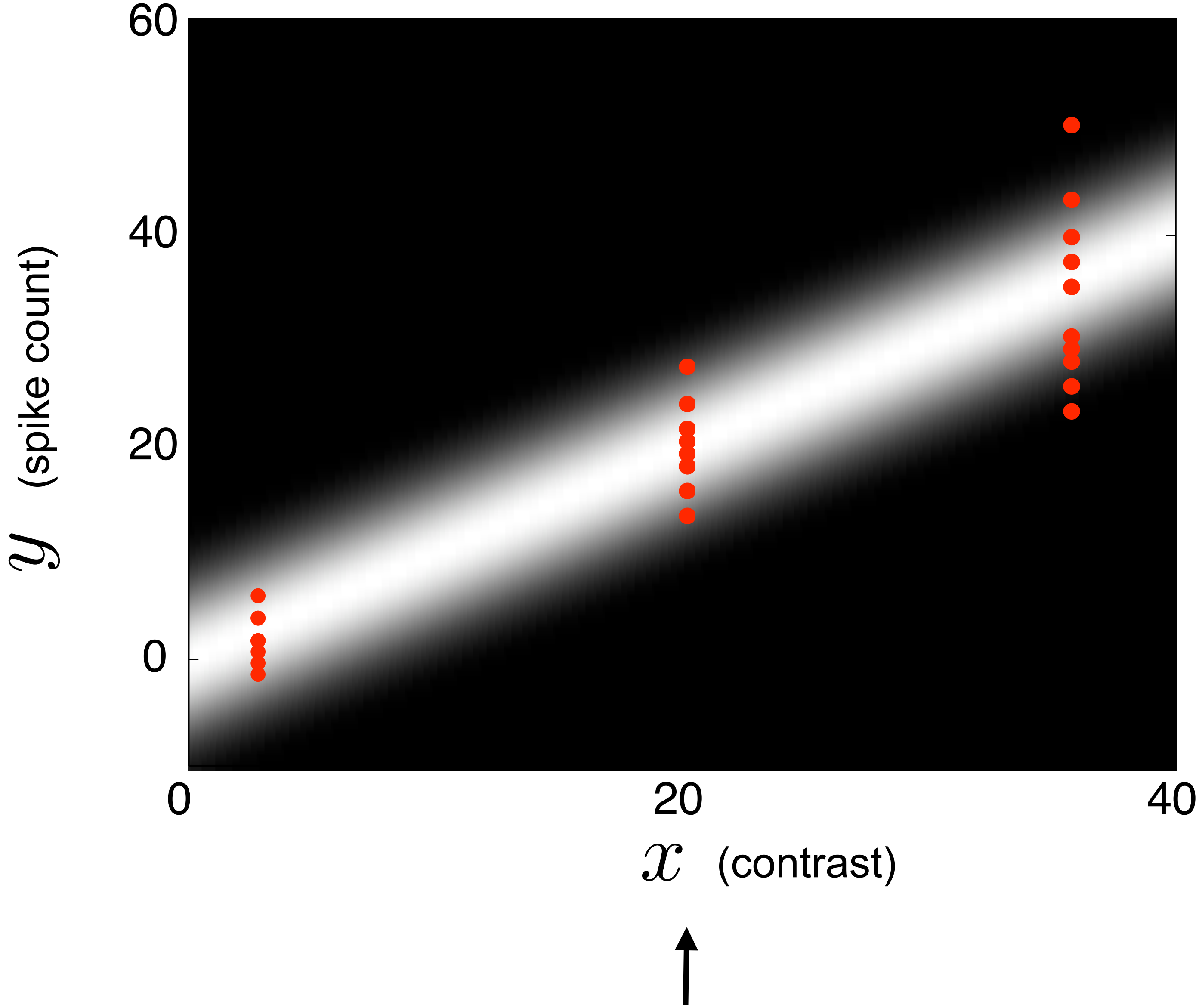
stimulus

spike rate $\mu = \theta x$

spike count $y \sim \mathcal{N}(\mu, \sigma^2)$

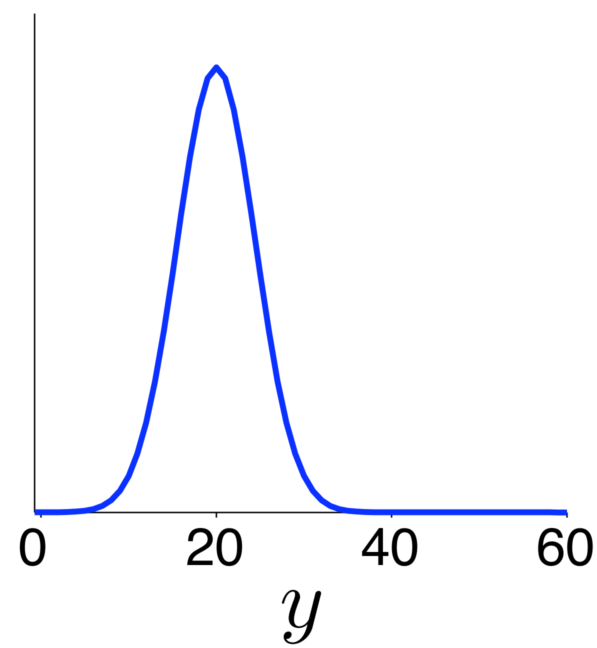
encoding model: $$P(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-\theta x)^2}{2\sigma^2}}$$

$$\mathrm{mean}(y) = \theta x$$
$$\mathrm{var}(y) = \sigma^2$$



encoding distribution

$$p(y|x = 20)$$



All slices have same width

$$\boxed{P(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-\theta x)^2}{2\sigma^2}}}$$

Log-Likelihood $$\log P(Y|X,\theta) = -\sum \frac{(y_i - \theta x_i)^2}{2\sigma^2} + c$$

Differentiate, set to zero, and solve for θ

$$\boxed{P(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-\theta x)^2}{2\sigma^2}}}$$

Log-Likelihood $\quad \log P(Y|X,\theta) = -\sum \frac{(y_i - \theta x_i)^2}{2\sigma^2} + c$

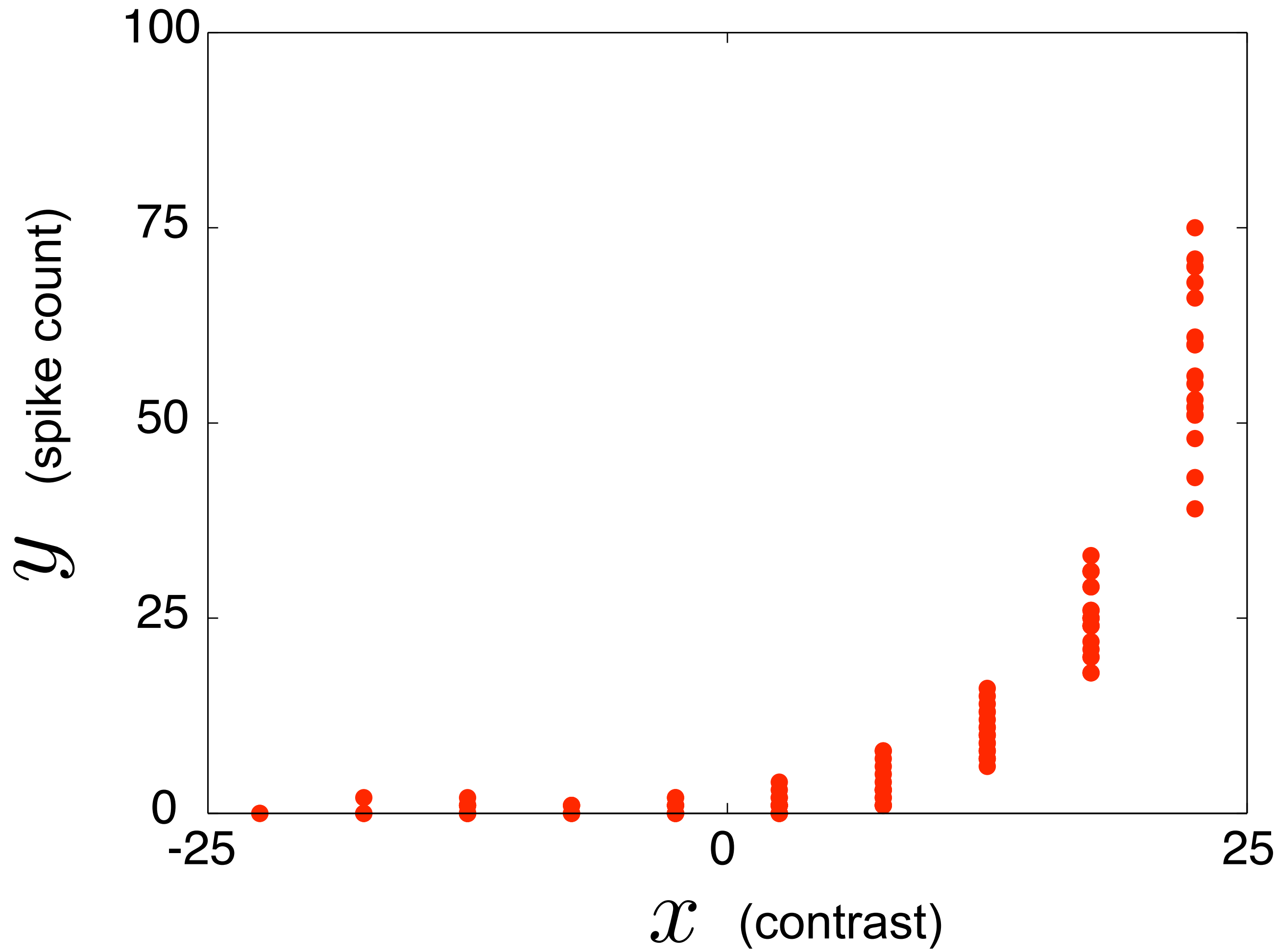
$$\frac{d}{d\theta} \log P(Y|X,\theta) = -\sum \frac{(y_i - \theta x_i)x_i}{\sigma^2} = 0$$

Maximum-Likelihood Estimator: $\quad \hat{\theta}_{ML} = \frac{\sum y_i x_i}{\sum x_i^2}$

("Least squares regression" solution)

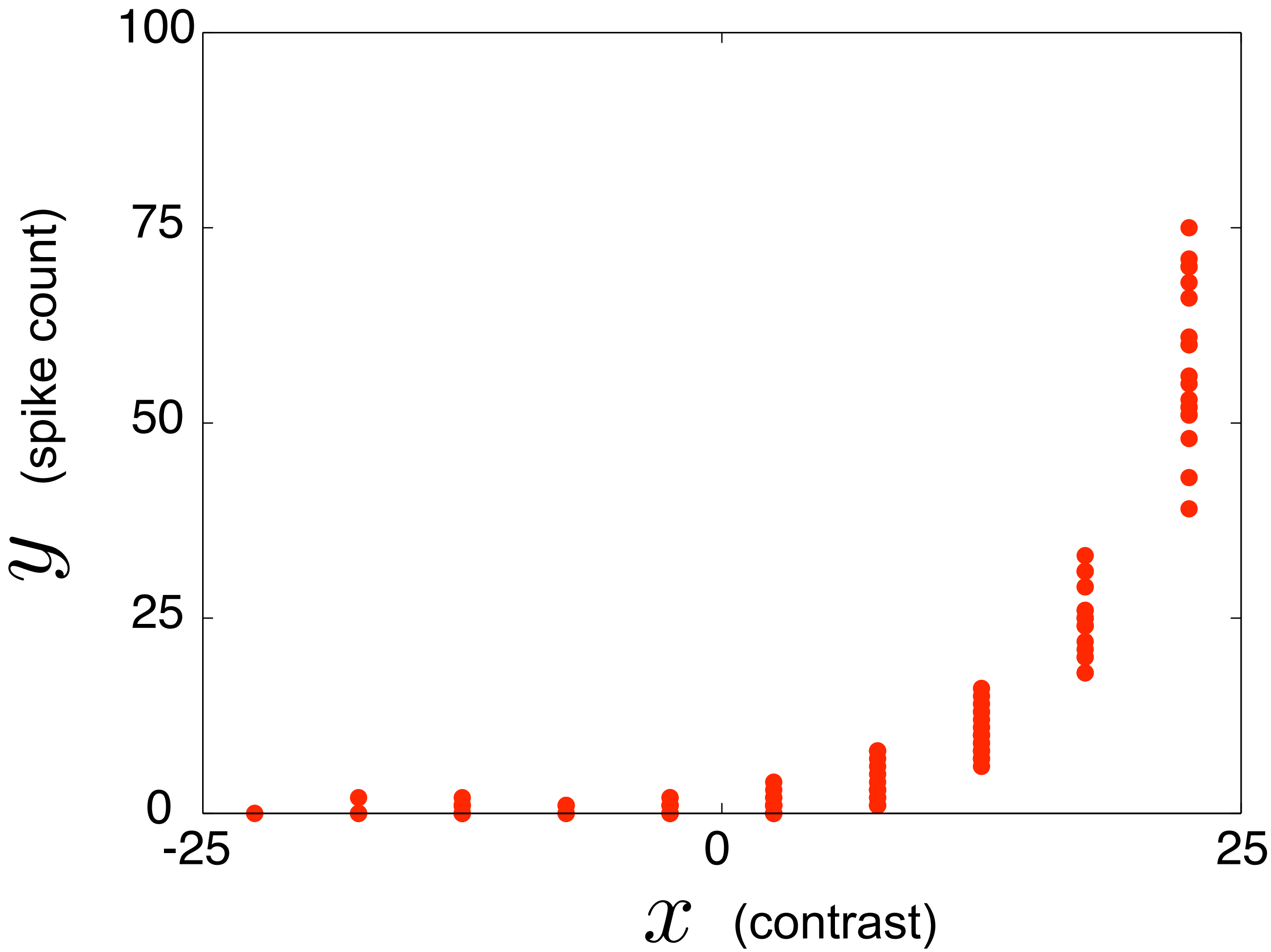
(Recall that for Poisson, $\hat{\theta}_{ML} = \frac{\sum y_i}{\sum x_i}$)

example #3: unknown neuron



Be the computational neuroscientist: what model would you use?

Example 3: unknown neuron



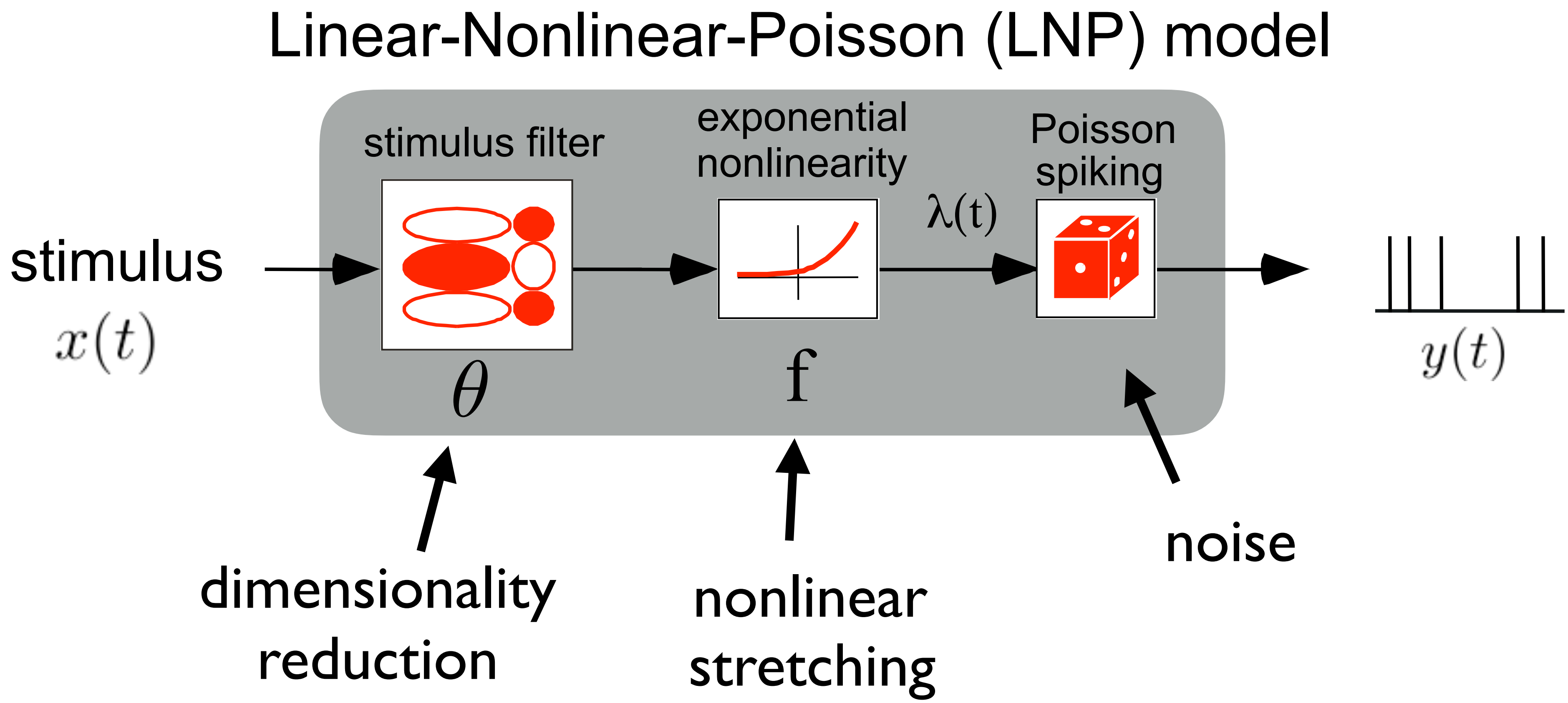
More general setup:

$$\lambda = f(\theta x)$$ firing rate is nonlinear

$$y \sim Poiss(\lambda)$$ Poisson firing

This is a GLM!

"basic" Poisson generalized linear model (GLM)



spike rate $\lambda = f(\vec{k} \cdot \vec{x})$
spike count $y \sim \mathrm{Poiss}(\lambda)$

- also known as a "cascade" model

What is a GLM?

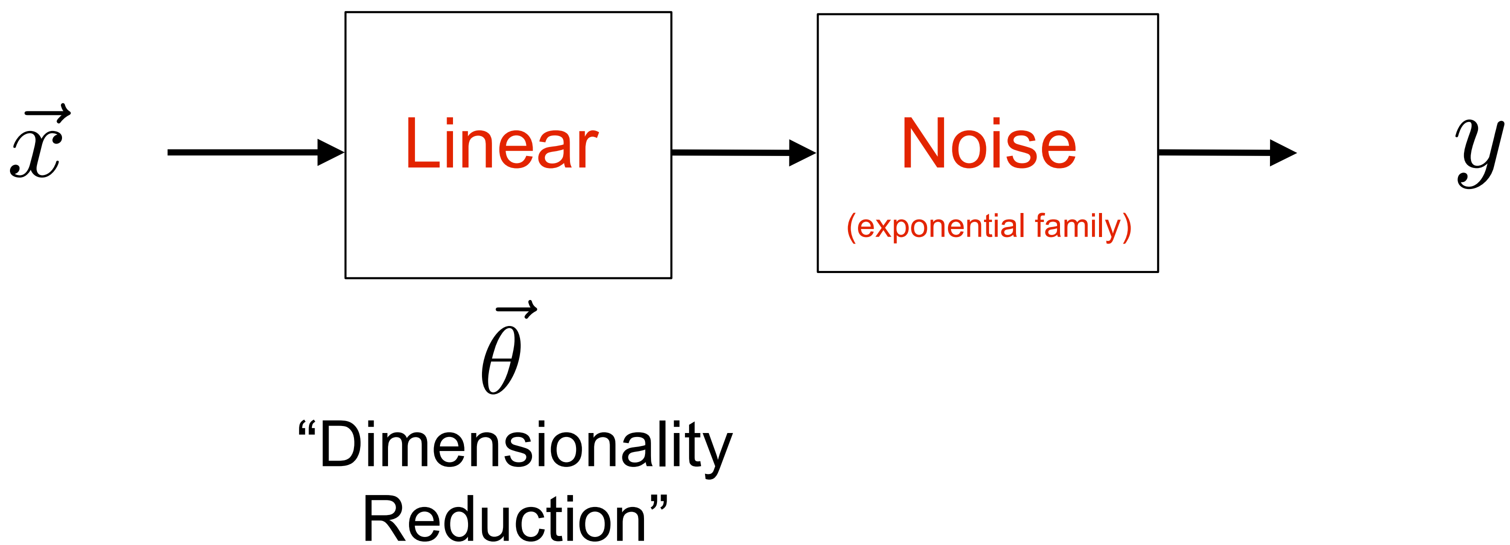
Be careful about terminology:

GLM	$\neq$	GLM
General Linear Model		Generalized Linear Model
		(Nelder 1972)
Linear		~~Linear~~

Moral:

Be careful when naming your model!

1. General Linear Model

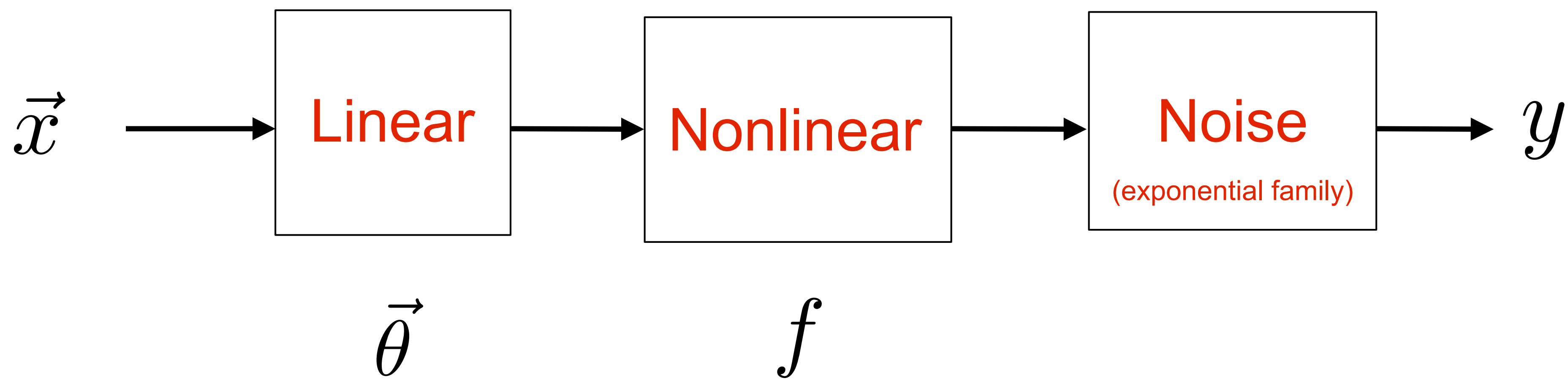


Examples:

1. Gaussian $\quad y = \vec{\theta} \cdot \vec{x} + \epsilon$

2. Poisson $\quad y \sim \mathrm{Poiss}(\vec{\theta} \cdot \vec{x})$

2. Generalized Linear Model



Examples:

1. Gaussian $\quad y = f(\vec{\theta} \cdot \vec{x}) + \epsilon$

2. Poisson $\quad y \sim \mathrm{Poiss}(f(\vec{\theta} \cdot \vec{x}))$

2. Generalized Linear Model

Terminology:

$\vec{\theta}$ "parameters"

f

"distribution function"

f^{-1} = "link function"

f = "the nonlinearity"

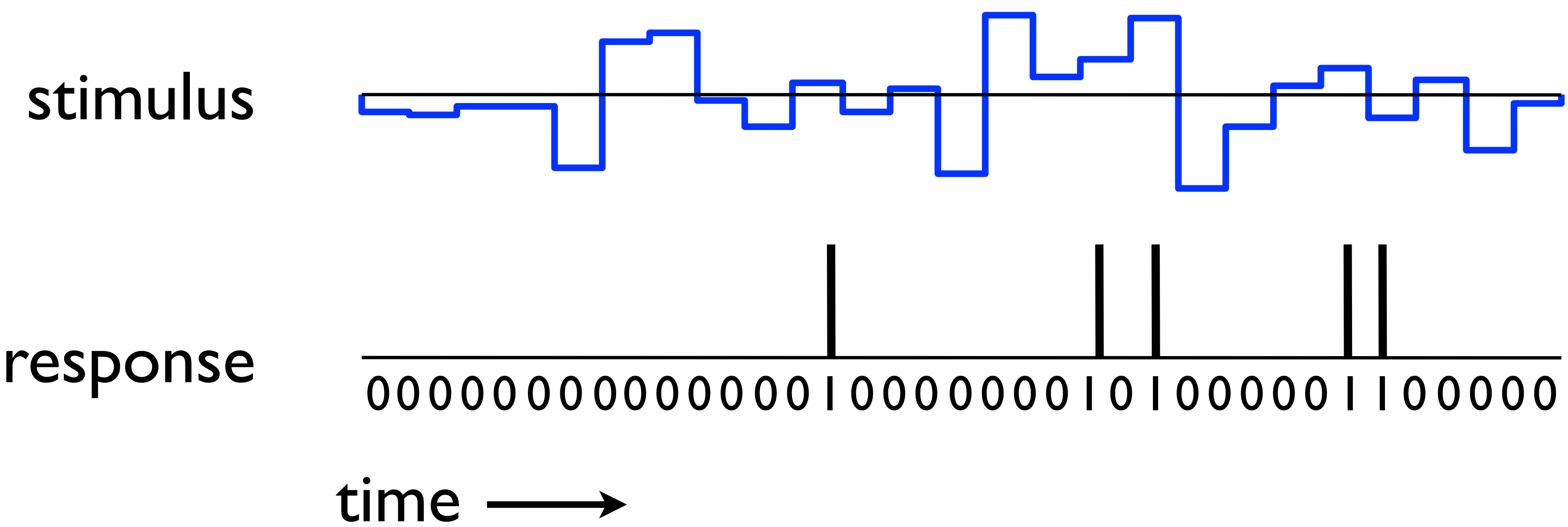
Applying it to data

$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

response at time t

linear filter

vector stimulus at time t



response
at time t

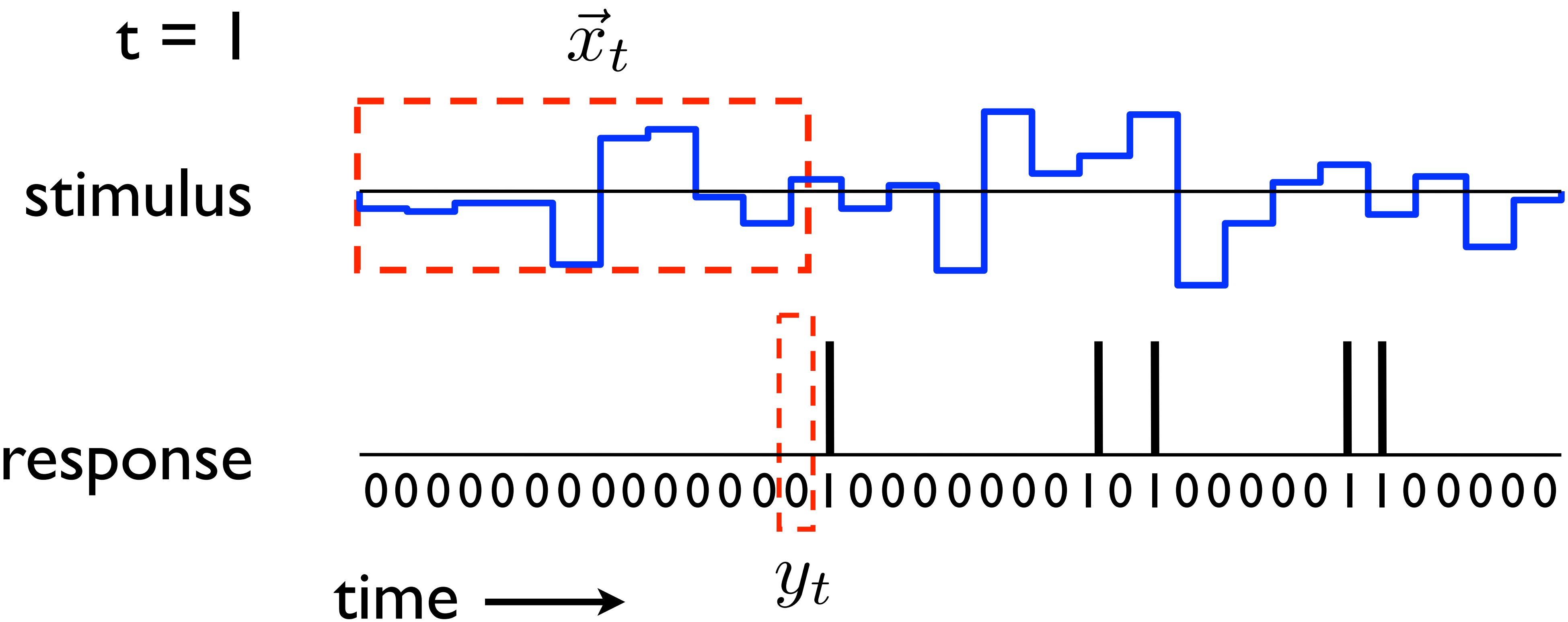
$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

linear
filter

vector stimulus
at time t

walk through the data
one time bin at a time

t = 1



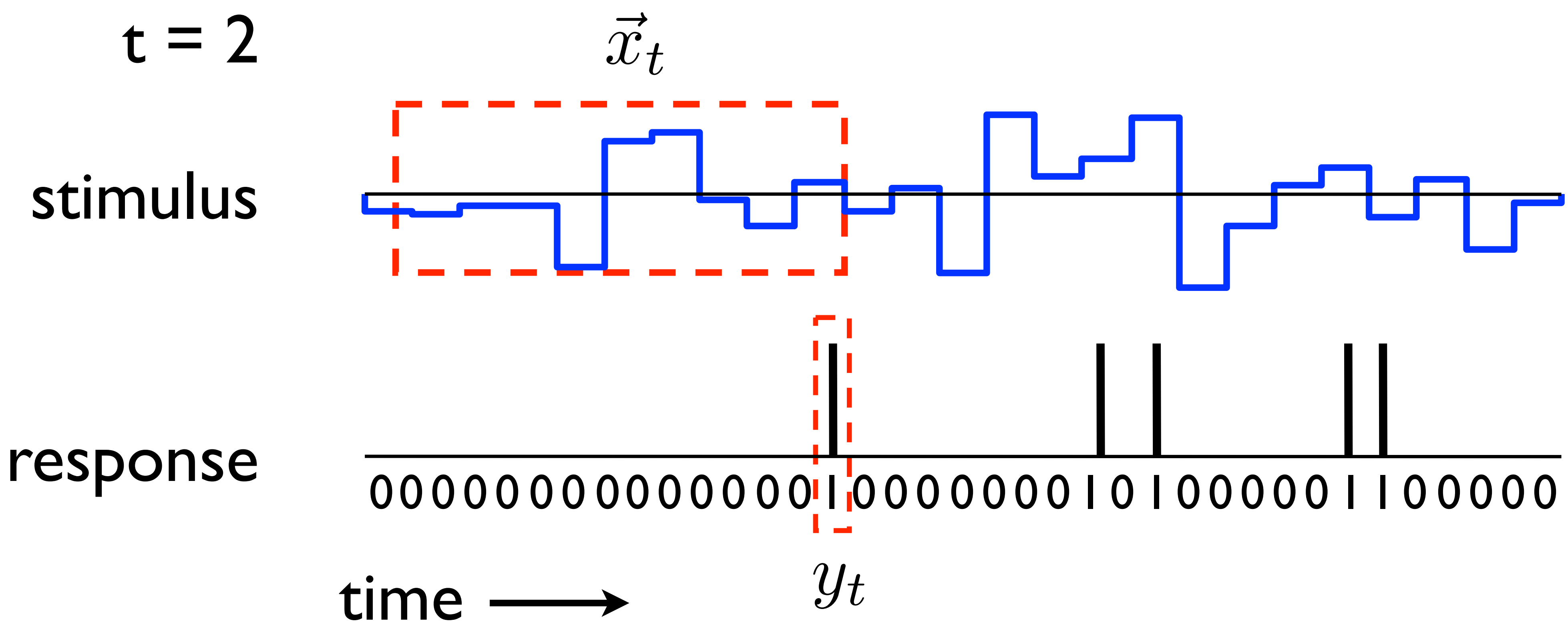
response
at time t

$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

linear
filter

vector stimulus
at time t

walk through the data
one time bin at a time



response
at time t

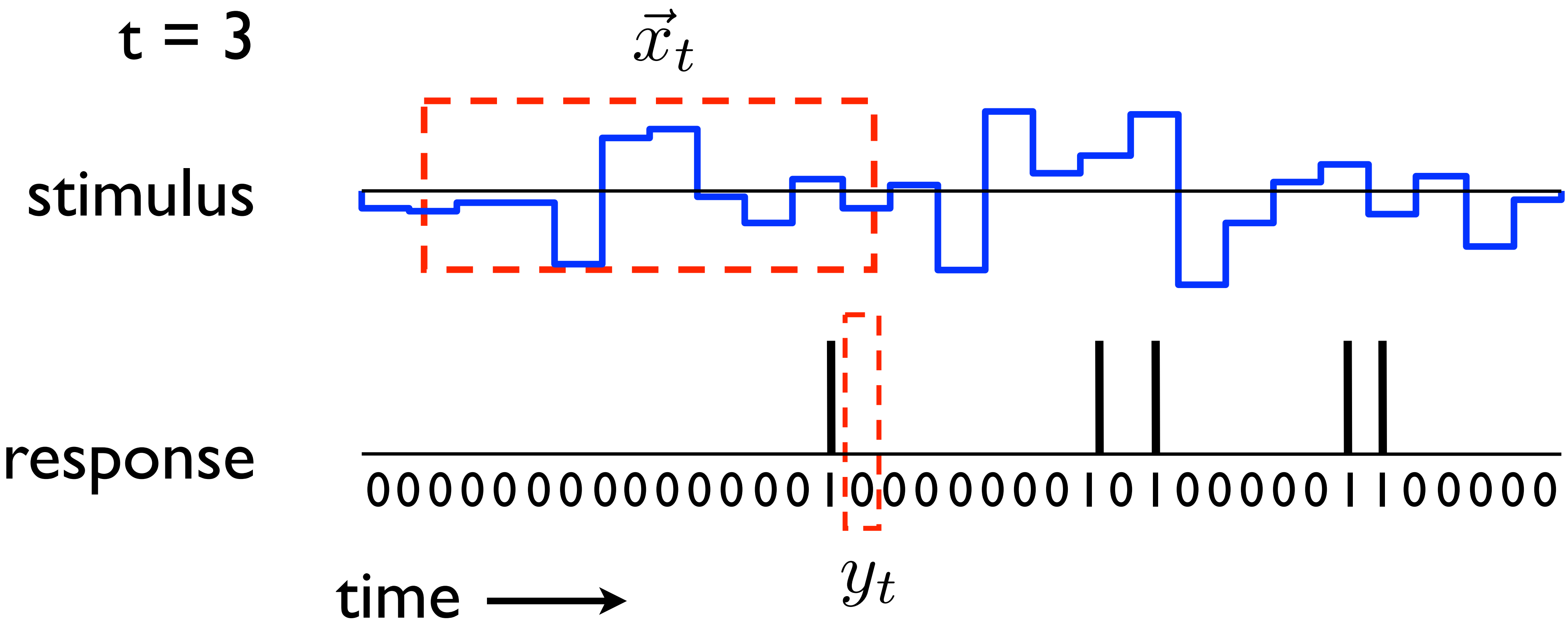
$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

linear
filter

vector stimulus
at time t

walk through the data
one time bin at a time

t = 3



response
at time t

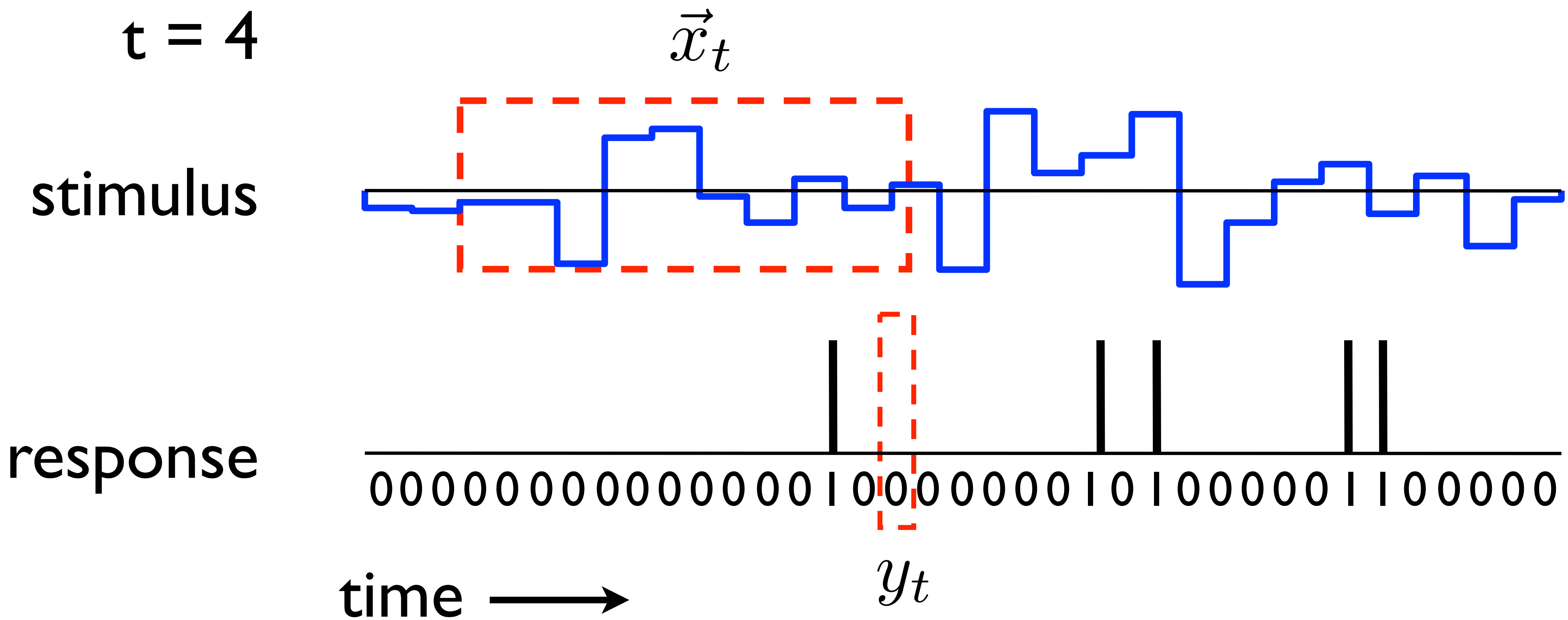
$$y_t = \vec{k} \cdot \vec{x}_t + \text{noise}$$

linear
filter

vector stimulus
at time t

walk through the data
one time bin at a time

t = 4



response at time t

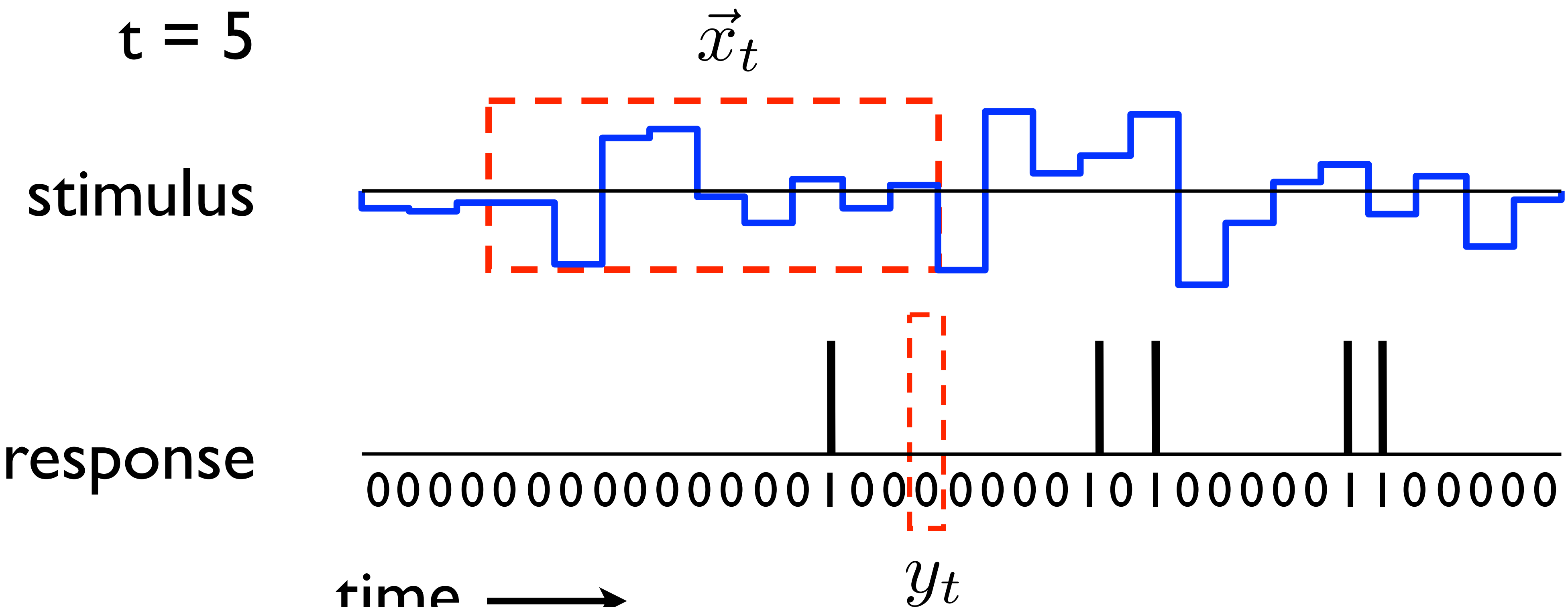
$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 5



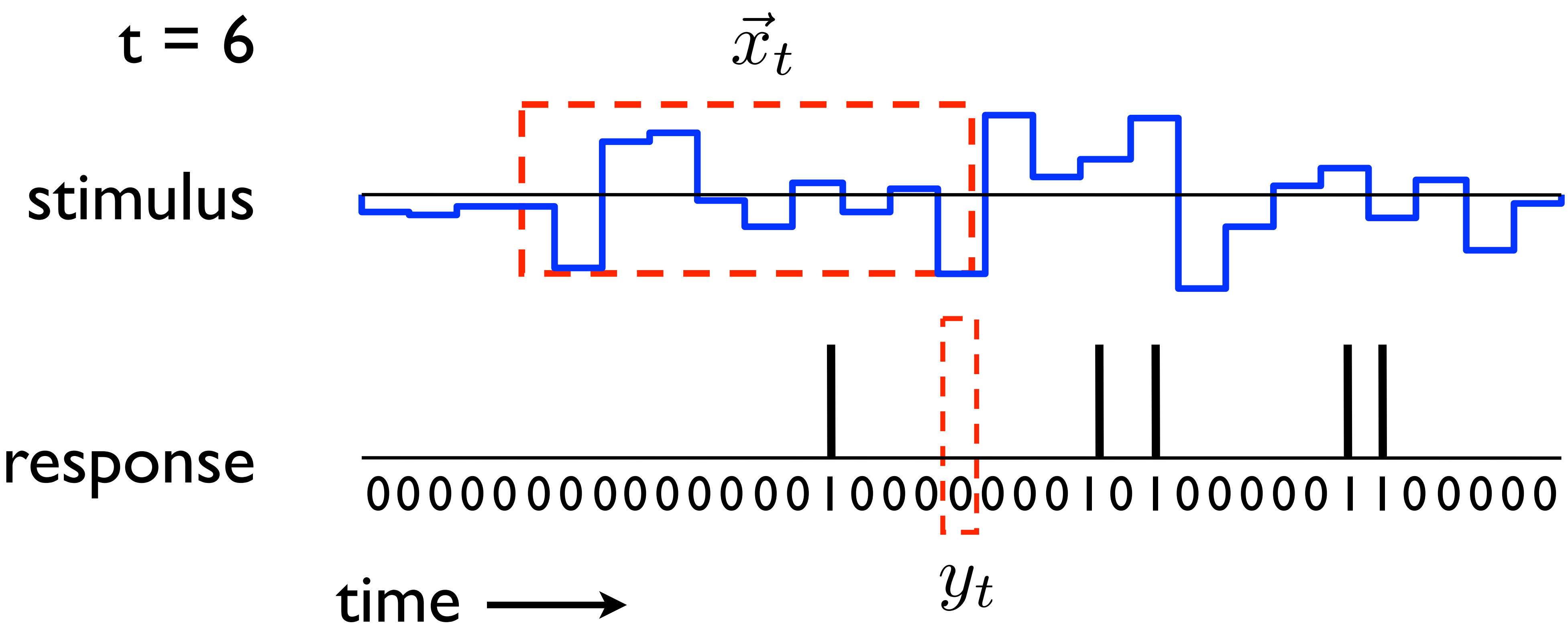
response
at time t

$$y_t = \vec{k} \cdot \vec{x}_t \; + \; \text{noise}$$

linear
filter

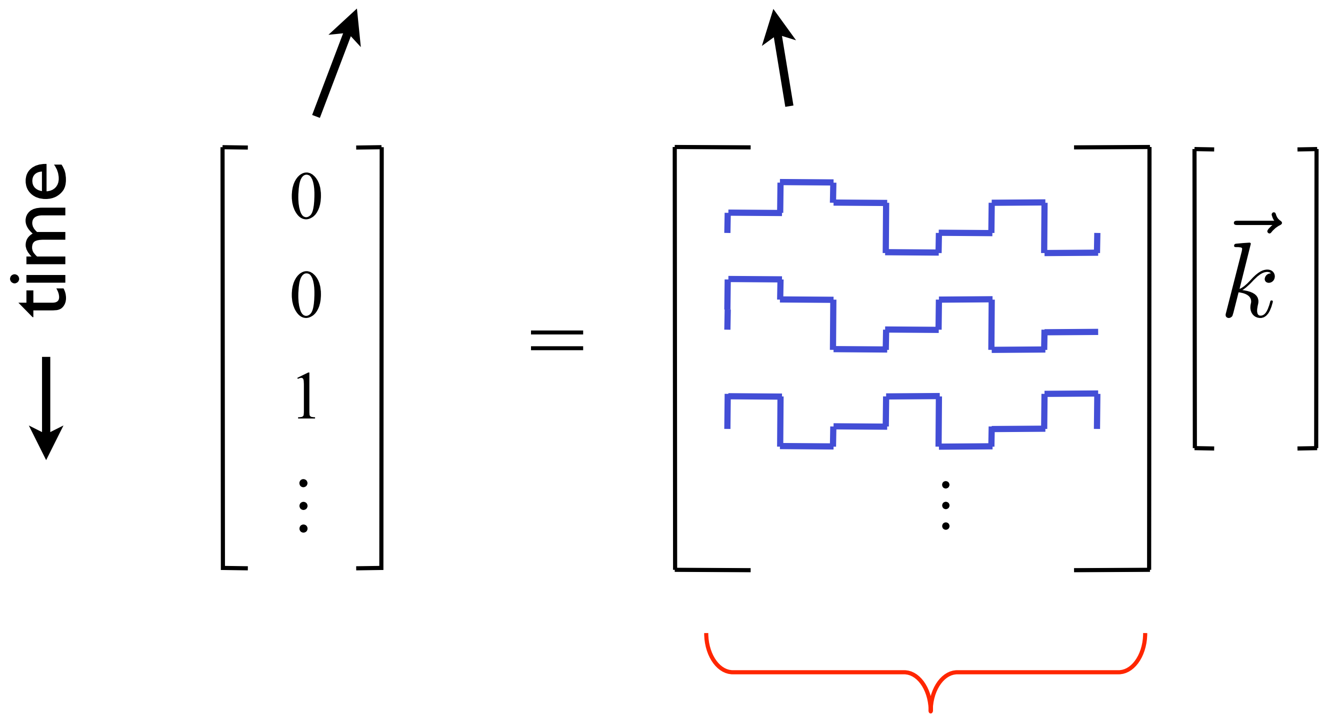
vector stimulus
at time t

walk through the data
one time bin at a time



Build up to following matrix version:

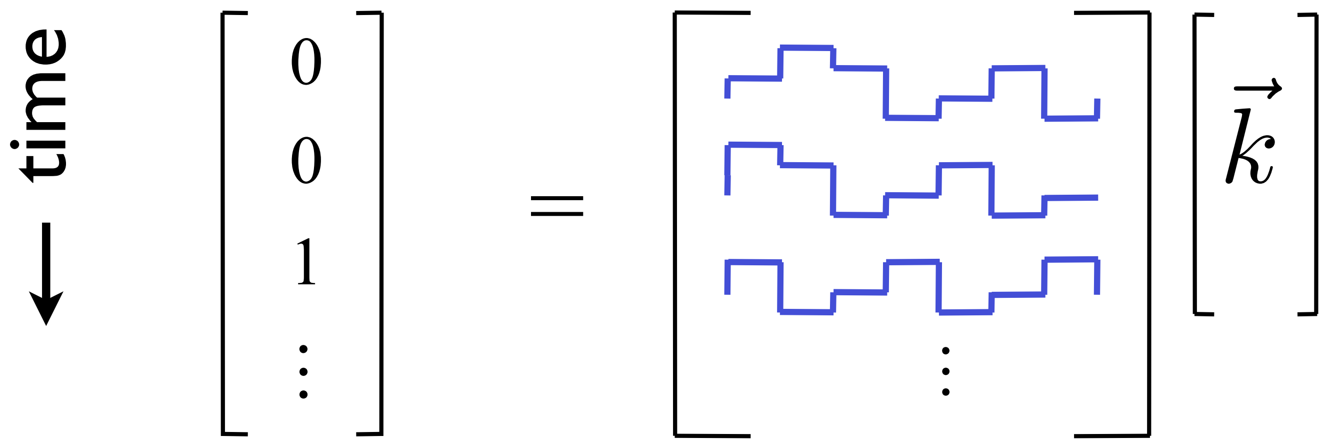
$$Y \quad = \quad X\vec{k} \quad + \quad noise$$



design matrix

Computing maximum likelihood estimate

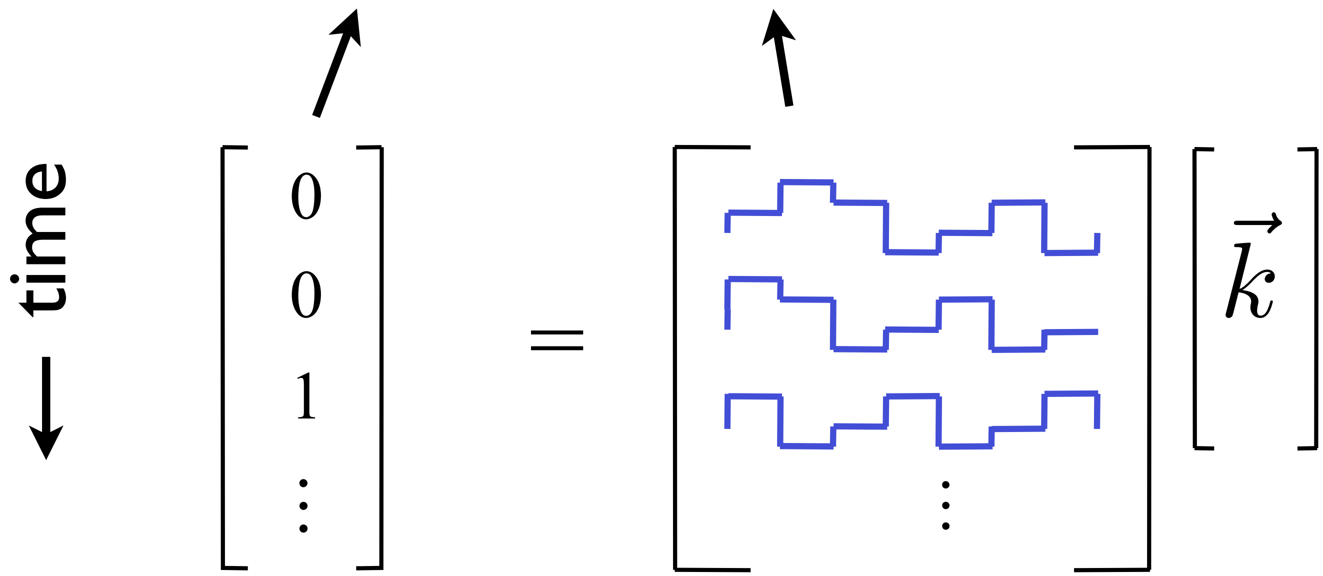
$$Y \quad = \quad X\vec{k} \quad + \quad noise$$



1. <u>"Linear-Gaussian" GLM</u>: $\hat{k} = \underbrace{(X^T X)}_{\text{stimulus covariance}}{}^{-1} \underbrace{X^T Y}_{\text{spike-triggered avg (STA)}}$

Computing maximum likelihood estimate

$$Y = f(X\vec{k}) + noise$$

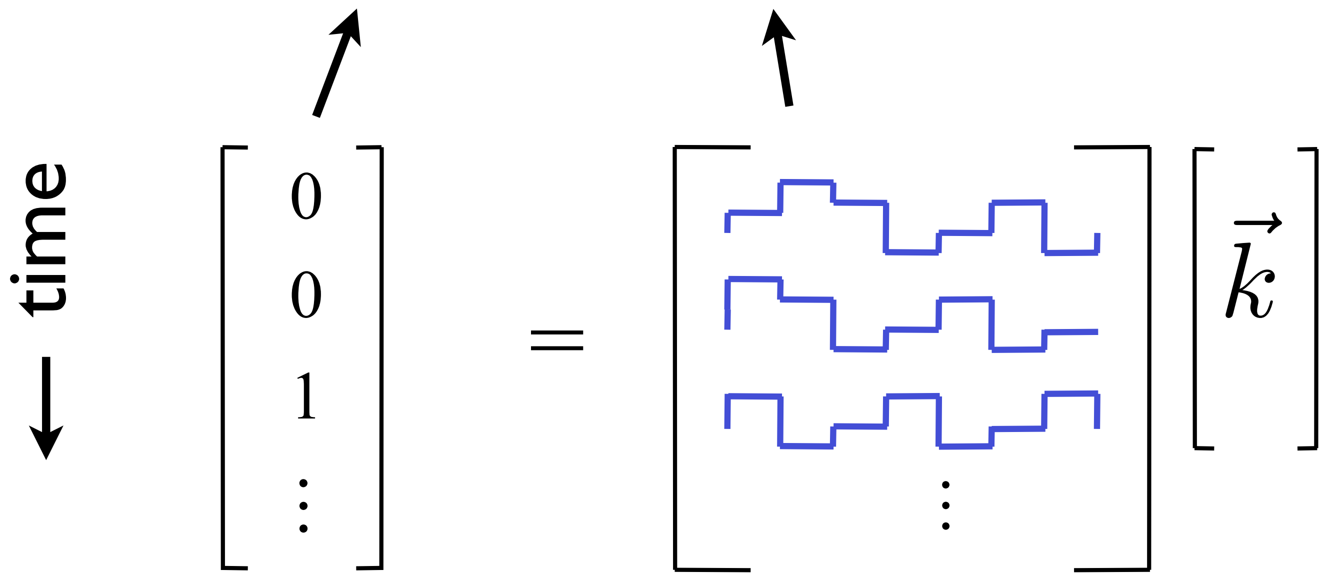


2. <u>Poisson GLM</u>: `k = glmfit(X,Y,'Poisson');`

maximum likelihood fit
(assumes exponential nonlinearity by default)

Computing maximum likelihood estimate

$$Y = f(X\vec{k}) + noise$$



3. <u>Bernoulli GLM</u>: `k = glmfit(X,Y,'binomial');`

outputs 0 and 1 (assumes **logistic** nonlinearity by default)

"logistic regression"

GLM summary

1. Linear-Gaussian GLM: $Y|X,\vec{k} \sim \mathcal{N}(X\vec{k}, \sigma^2 I)$ continuous

log-likelihood: $-\frac{1}{2\sigma^2}(Y - X\vec{k})^\top (Y - X\vec{k}) + const$

MLE: $\hat{k} = (X^T X)^{-1} X^T Y$

2. Poisson GLM: $y|\vec{x},\vec{k} \sim \mathrm{Poiss}\big(f(\vec{x}_t \cdot \vec{k})\big)$ integer counts

log-likelihood: $\mathcal{L} = Y^\top \log f(X\vec{k}) - \mathbf{1}^\top f(X\vec{k})$

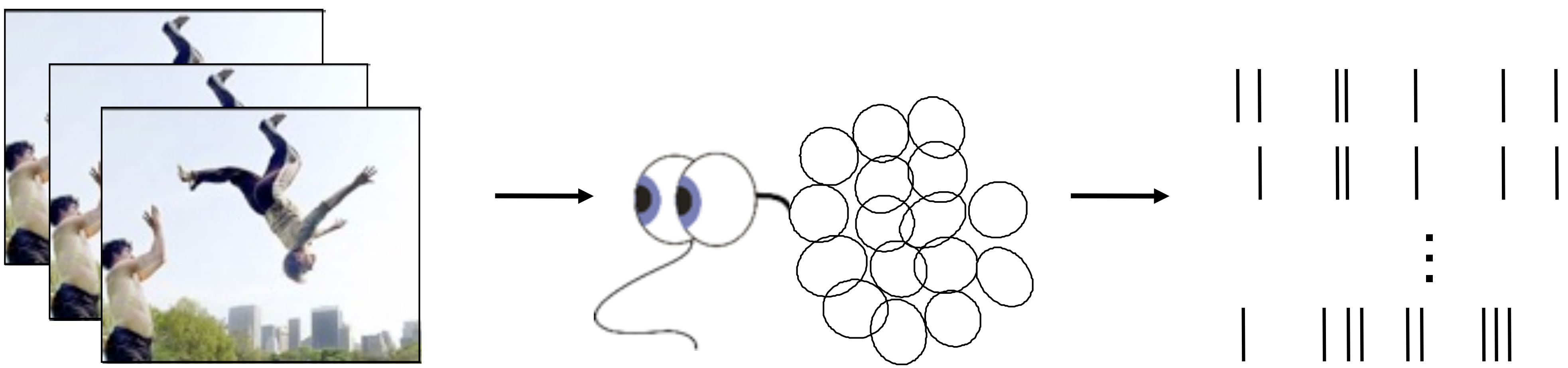
3. Bernoulli GLM: $y_t|\vec{x}_t,\vec{k} \sim \mathrm{Ber}(f(\vec{x}_t \cdot \vec{k}))$ binary counts

log-likelihood: $\mathcal{L} = Y^\top \log f(X\vec{k}) - (1 - Y)^\top \log(1 - f(X\vec{k}))$

"logistic regression" if $f(x) = \dfrac{1}{1 + e^{-x}}$

NEXT:

GLMs with spike-history and coupling



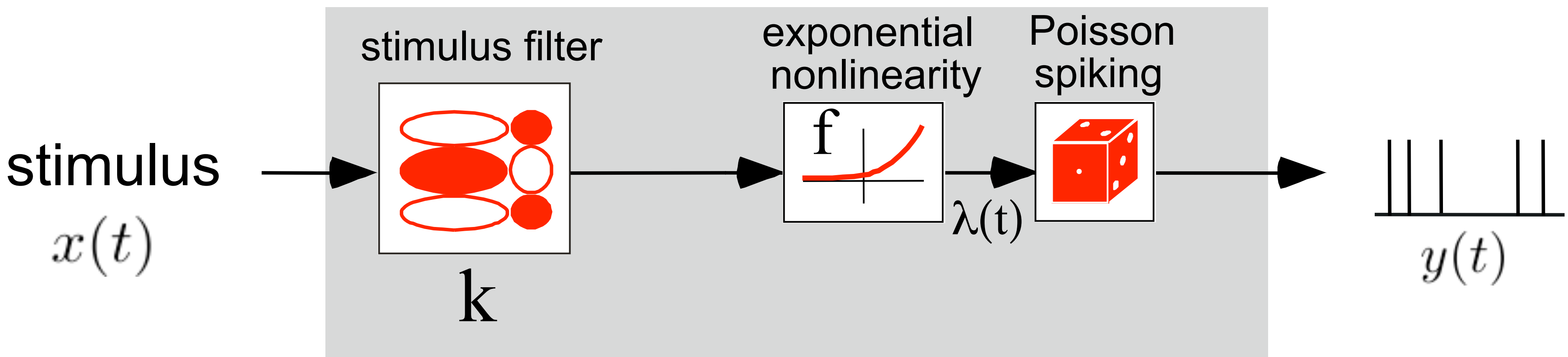
x

stimuli

y

spike trains

Poisson GLM

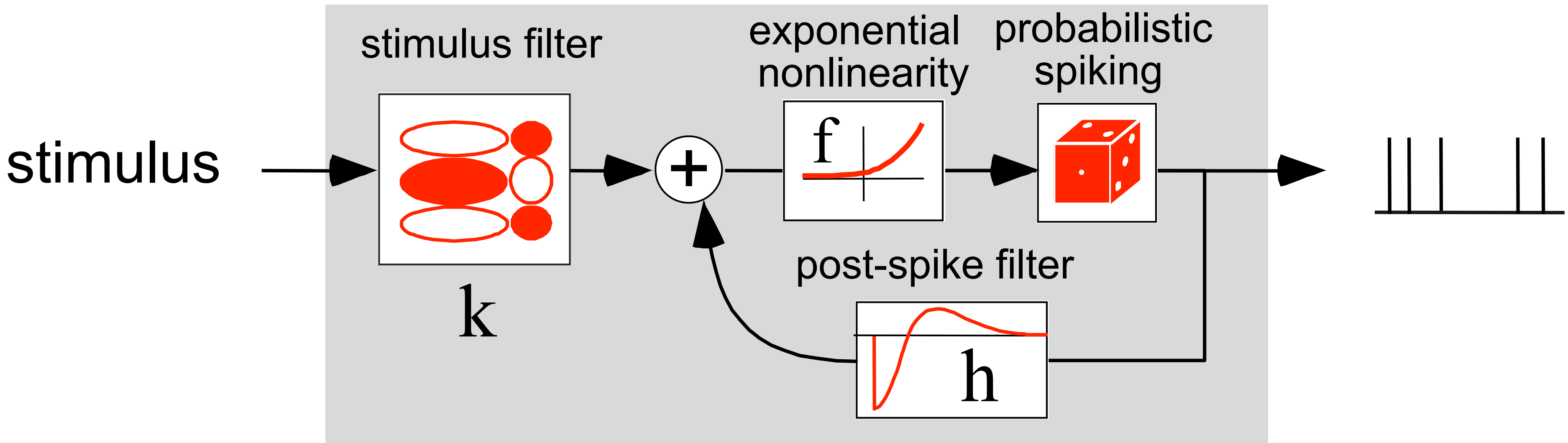


spike rate $\lambda(t) = f(k \cdot x(t))$

- problem: assumes spiking depends only on stimulus!

(Truccolo et al 2004, Gerstner 2001)

Poisson GLM with spike-history dependence



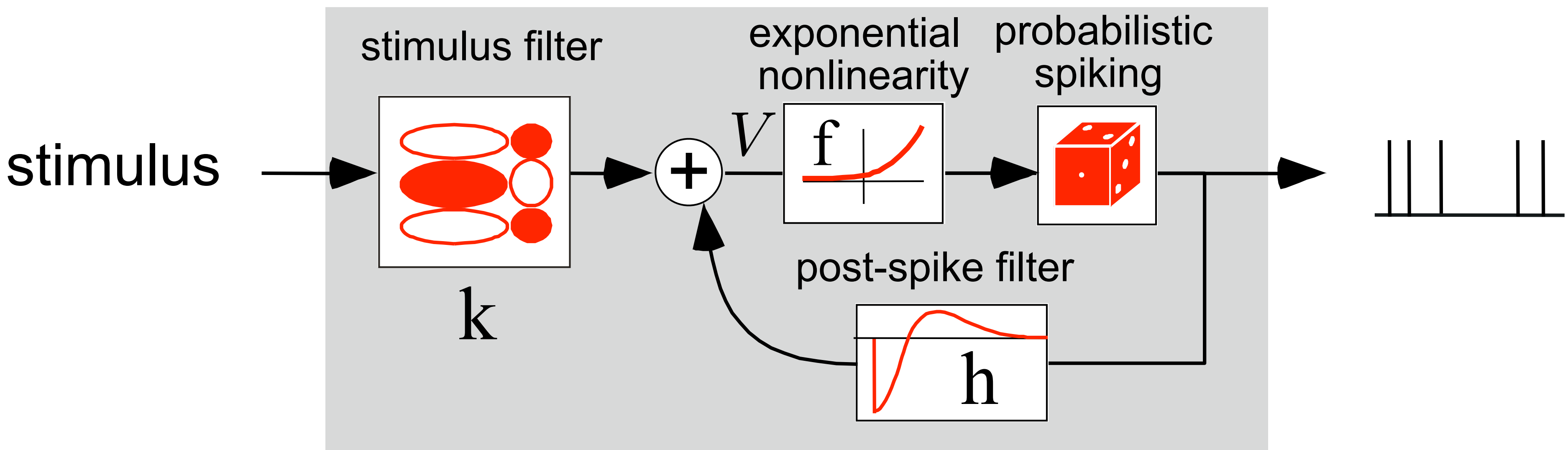
spike rate: $$\lambda(t) = f(\vec{k} \cdot \vec{x}(t) + \vec{h} \cdot \vec{y}_{hst}(t))$$

$$= e^{\vec{k} \cdot \vec{x}(t)} \cdot e^{\vec{h} \cdot \vec{y}_{hst}(t)}$$

- output: no longer a Poisson process
- interpretation: "soft-threshold" integrate-and-fire model

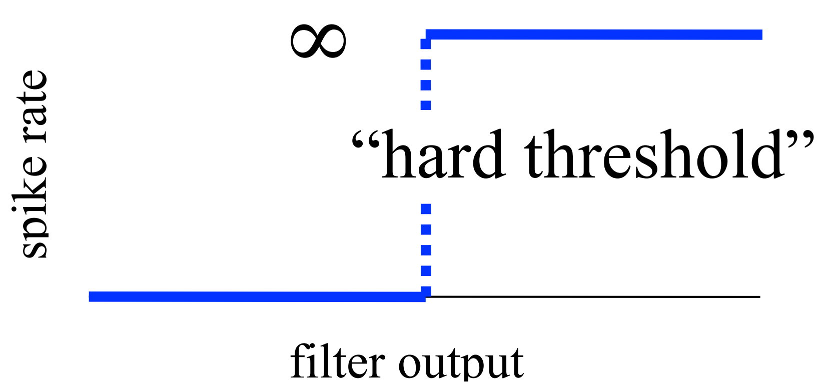
(Truccolo et al 2004, Gerstner 2001)

Poisson GLM with spike-history dependence

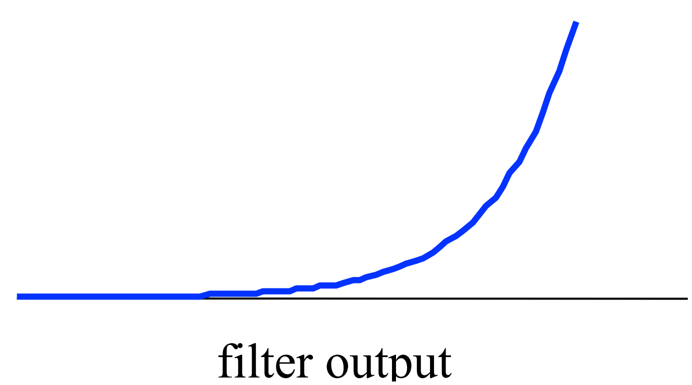


- interpretation: "soft-threshold" integrate-and-fire model

traditional IF

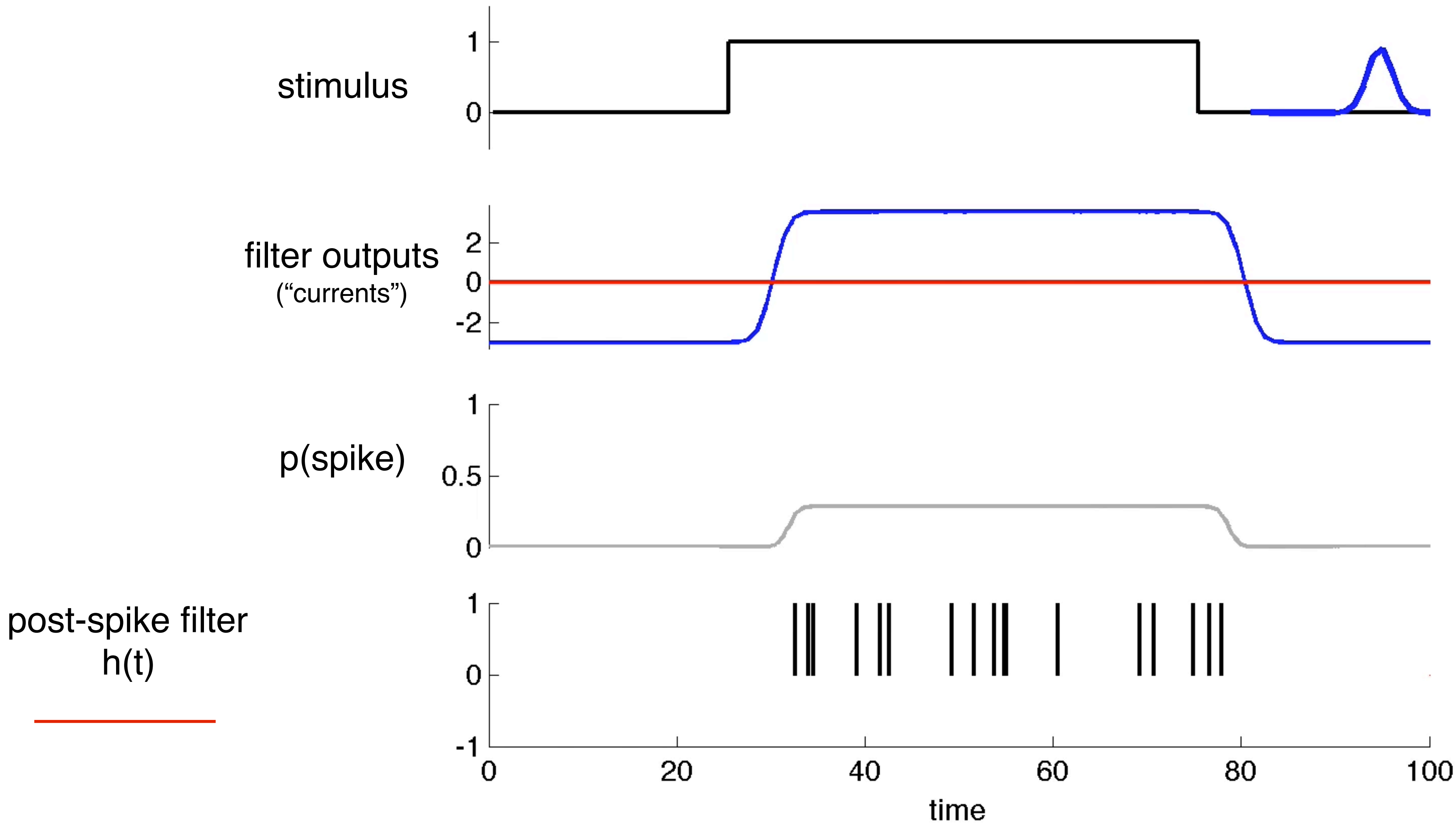


"soft-threshold" IF



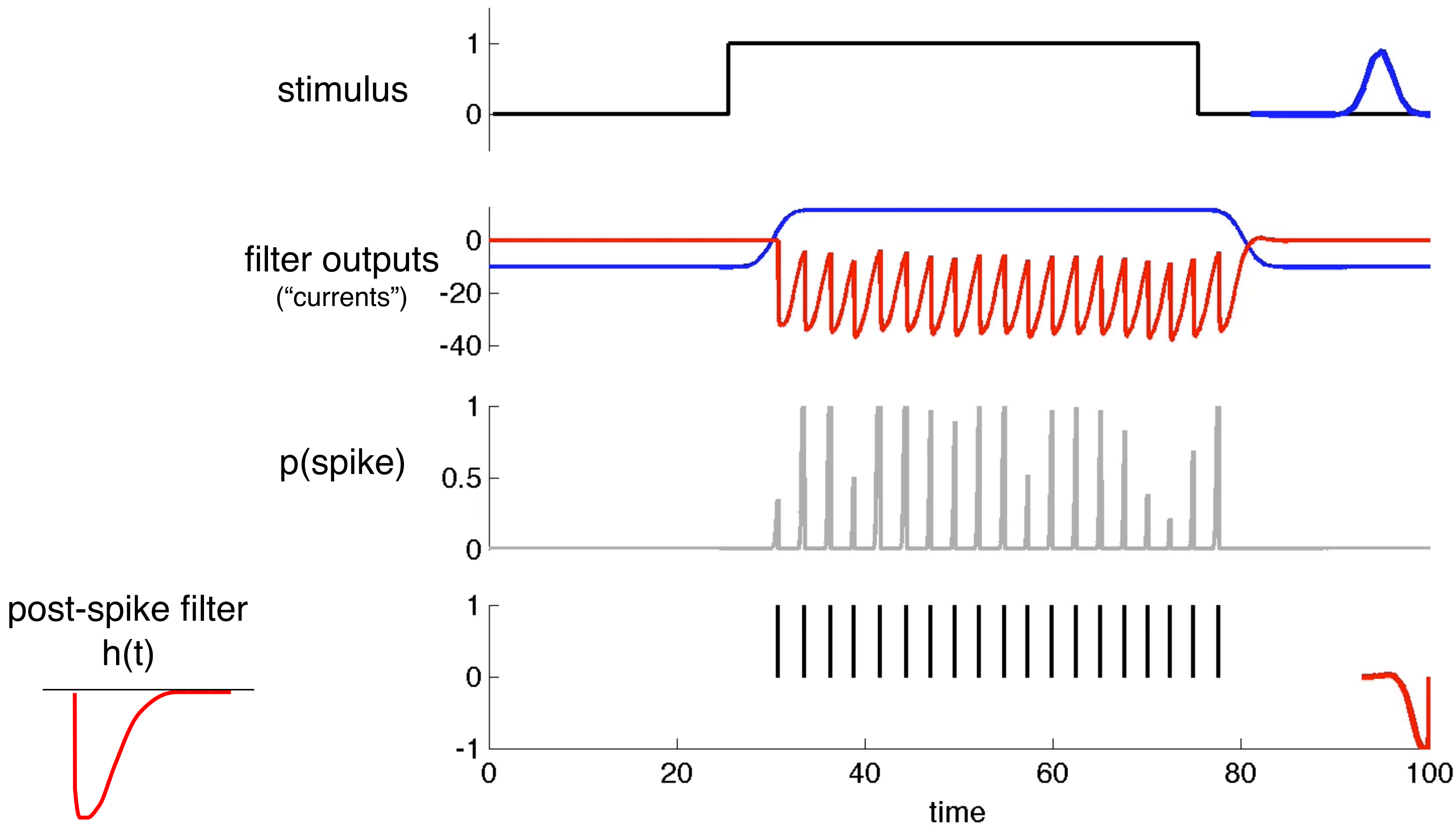
GLM dynamic behaviors

• irregular spiking



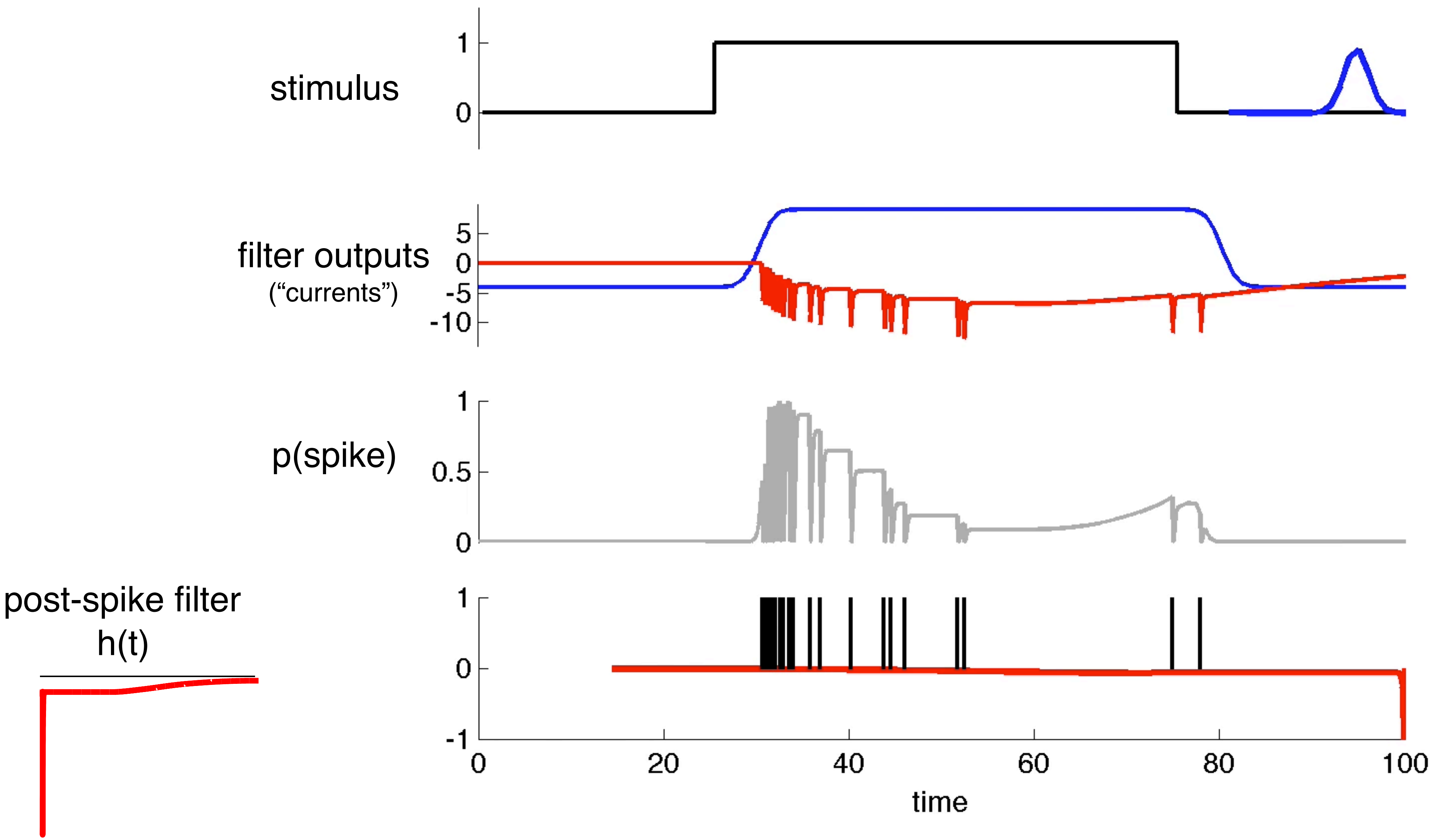
GLM dynamic behaviors

• regular spiking



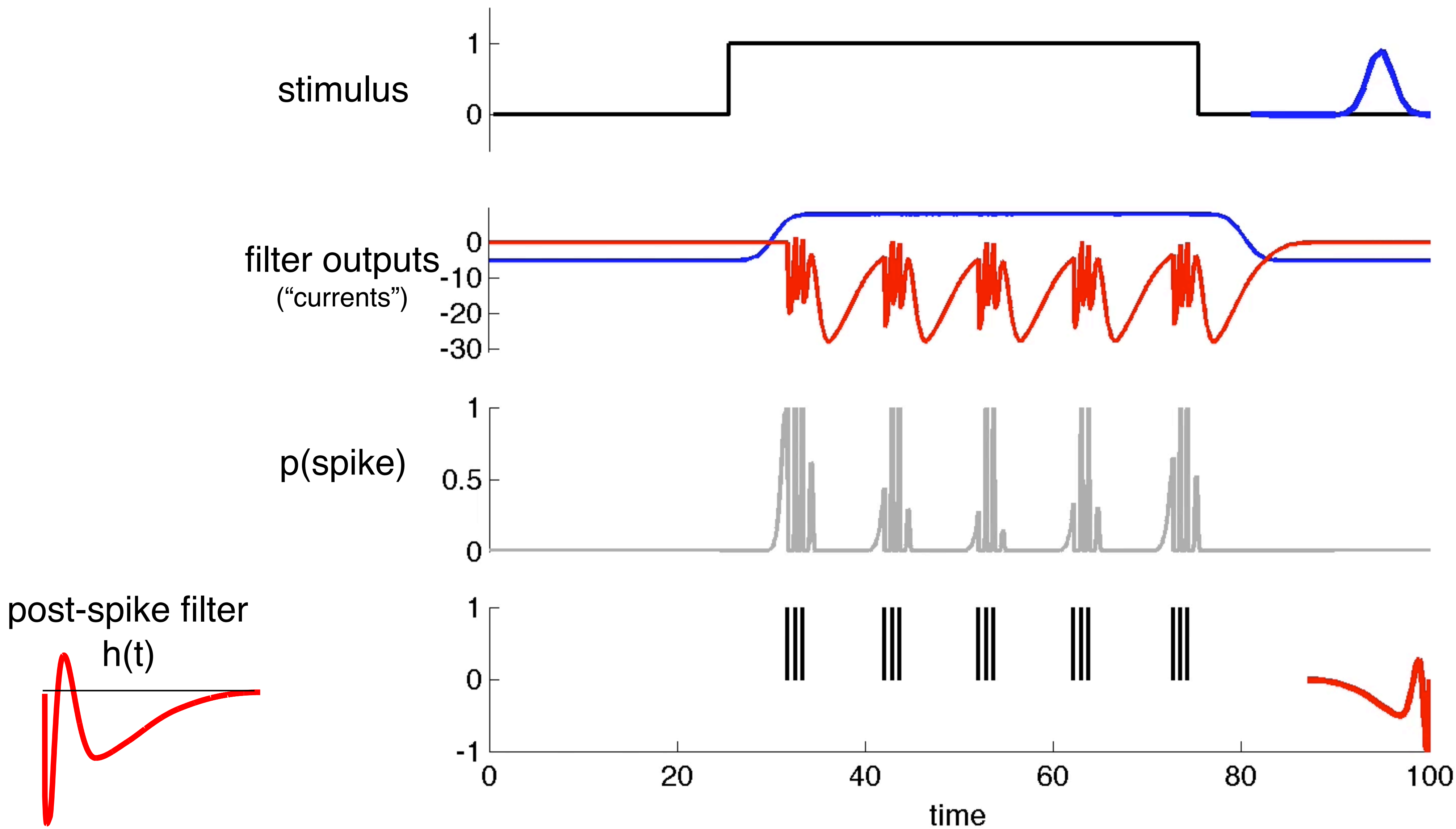
GLM dynamic behaviors

• adaptation



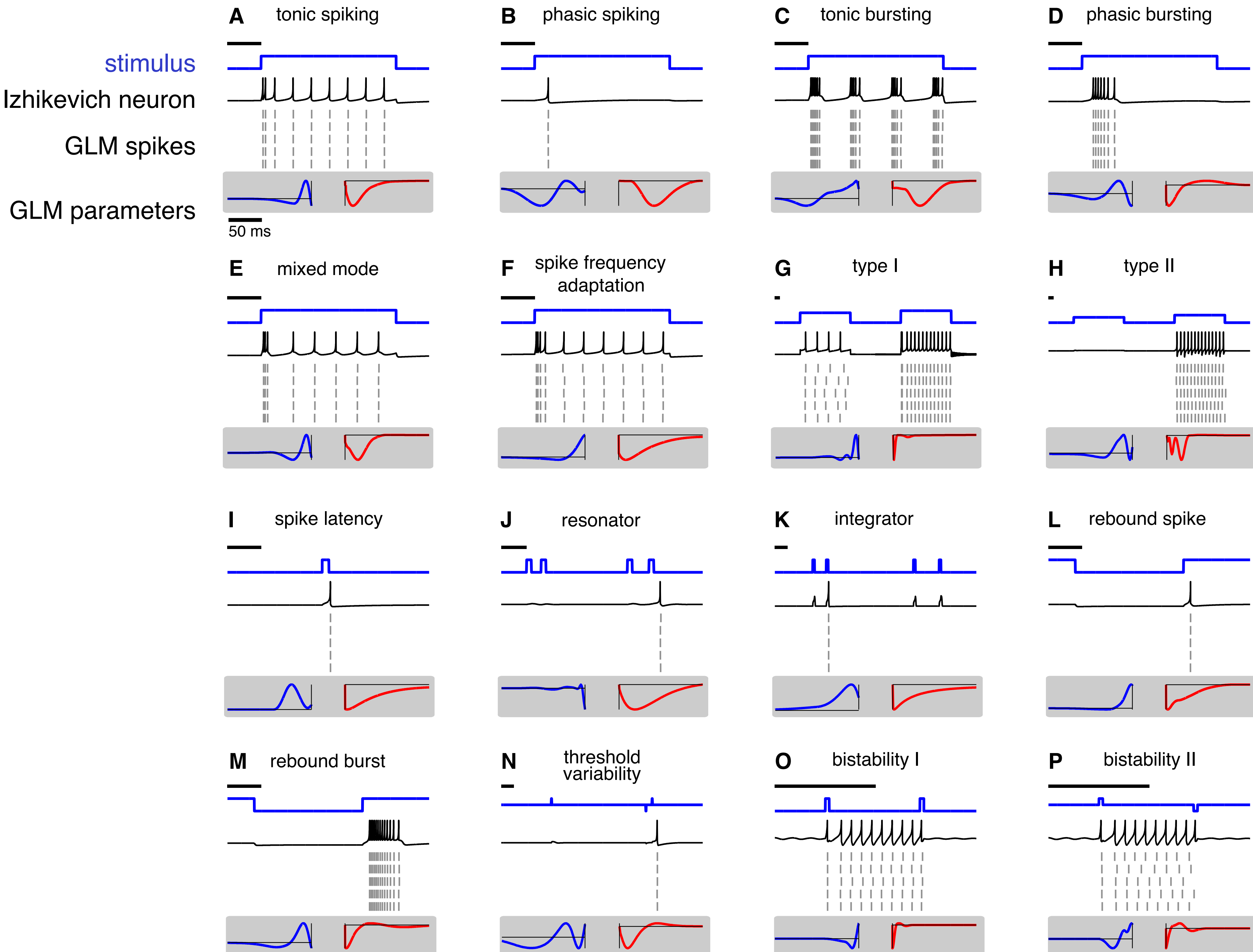
GLM dynamic behaviors

• bursting

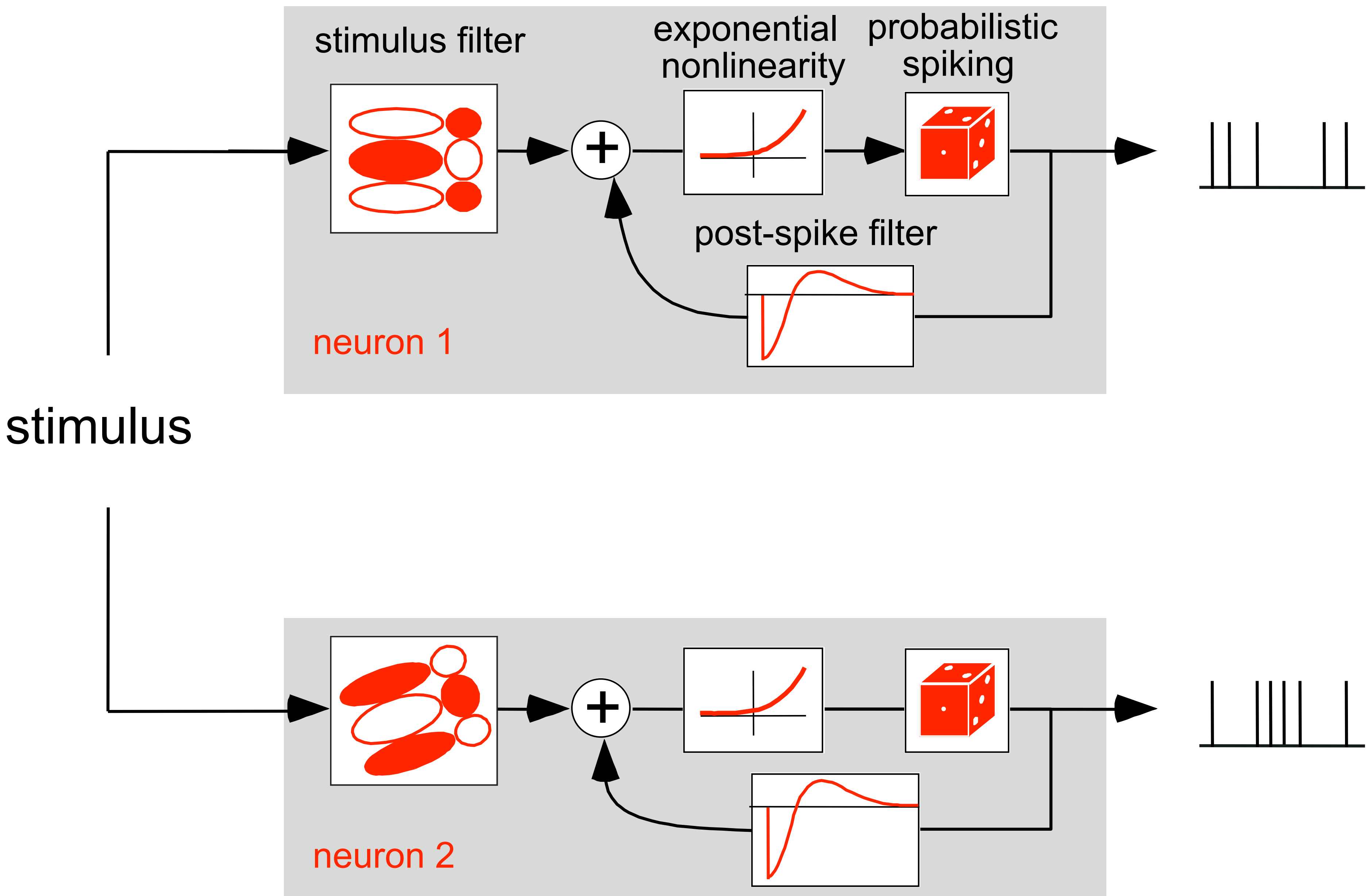


GLM dynamic behaviors (from Izhikevich)

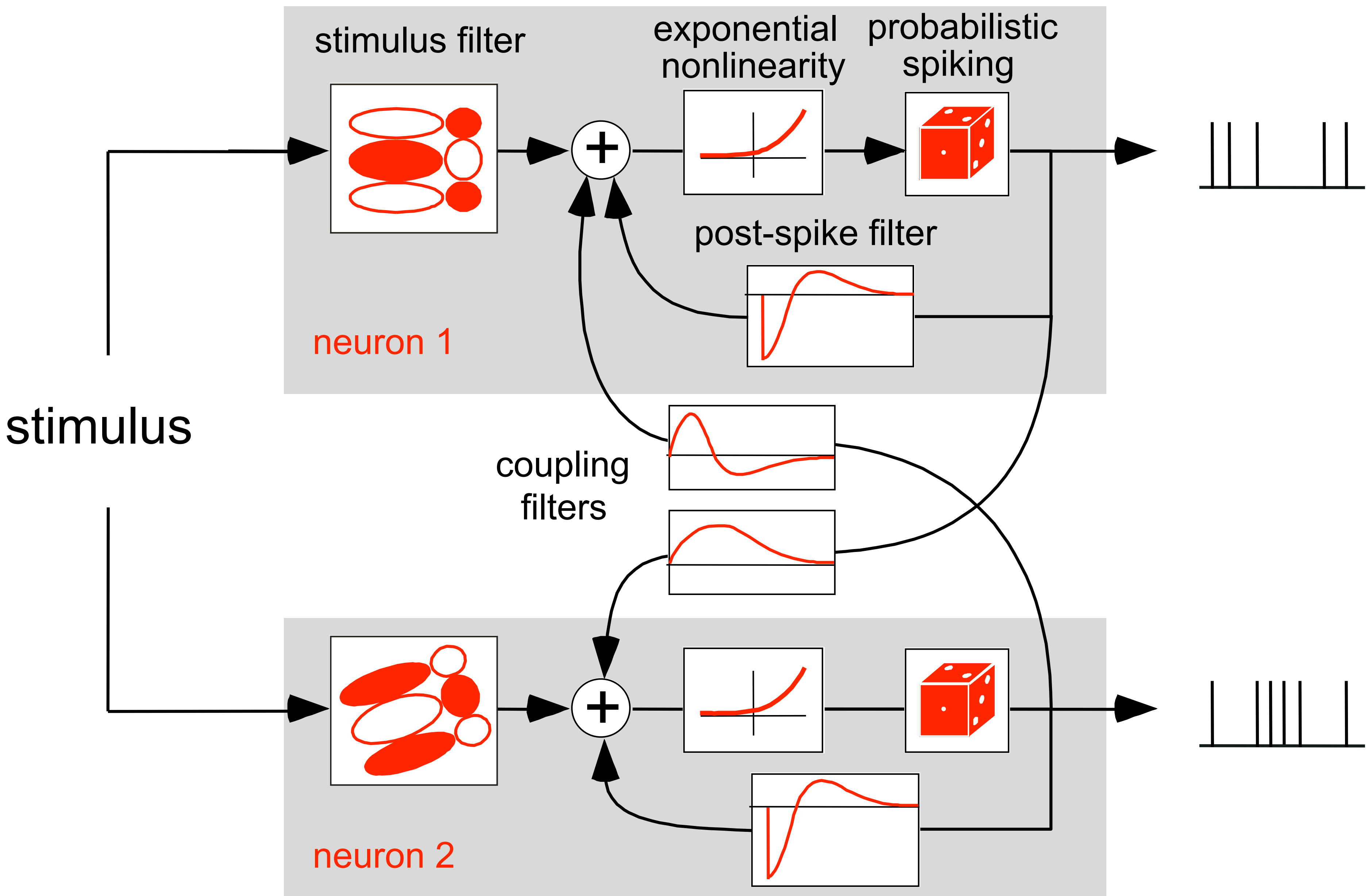
(Weber & Pillow 2017)



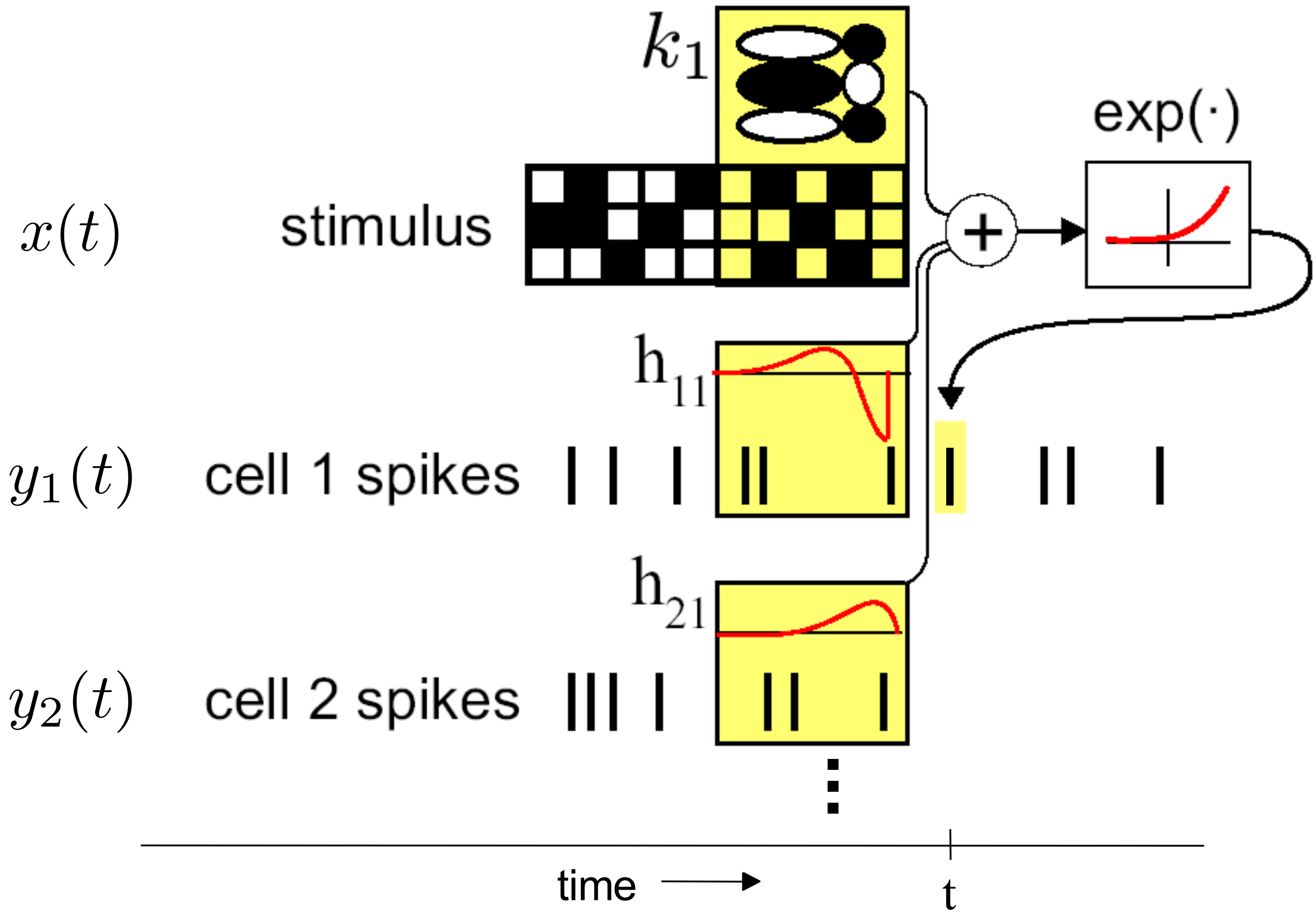
multi-neuron GLM



multi-neuron GLM



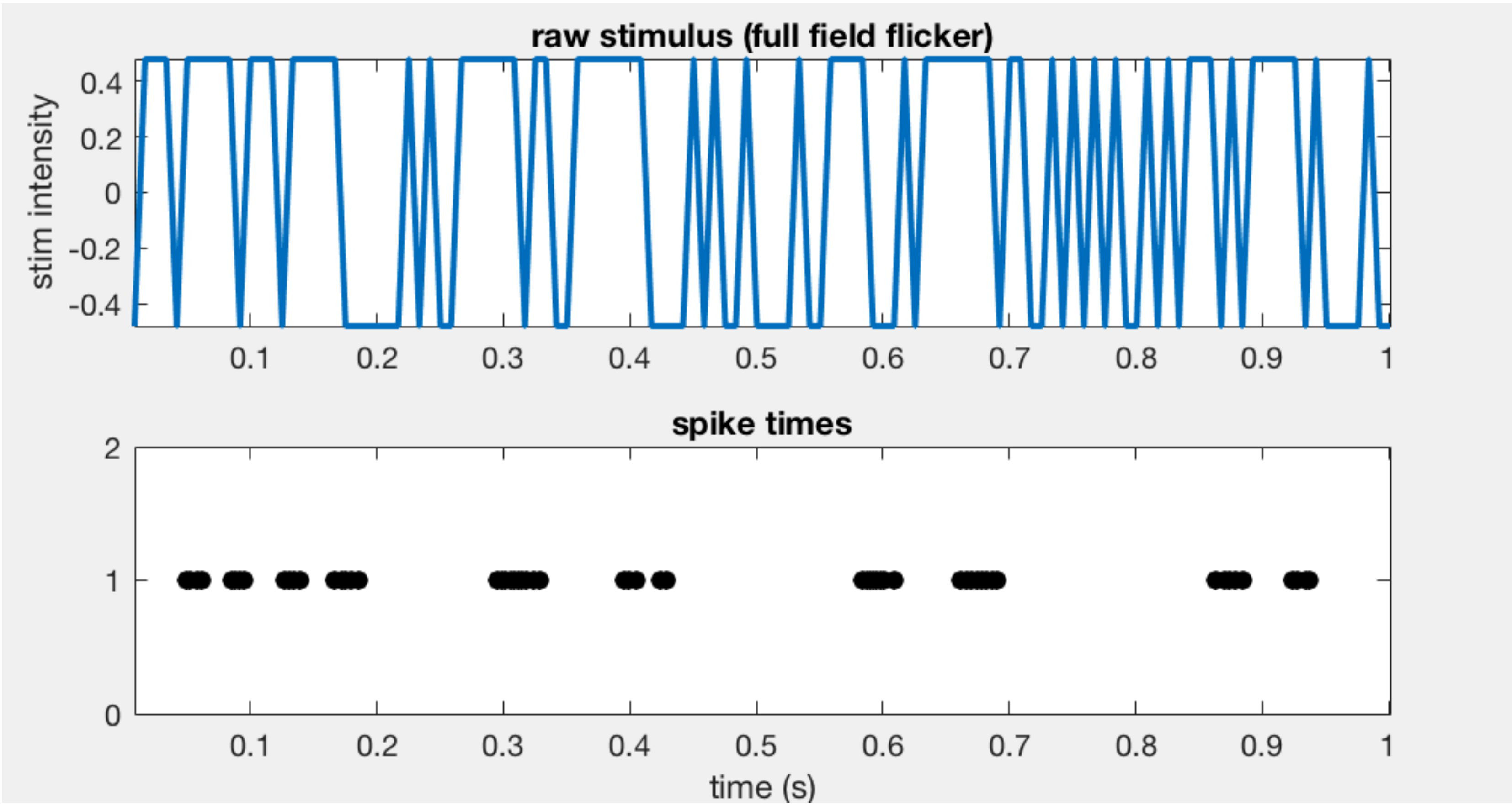
GLM equivalent diagram:



spike rate $\quad \lambda_i(t) = \exp(k_i \cdot x(t) \; + \; \sum_j h_{ij} \cdot y(t))$

Example dataset

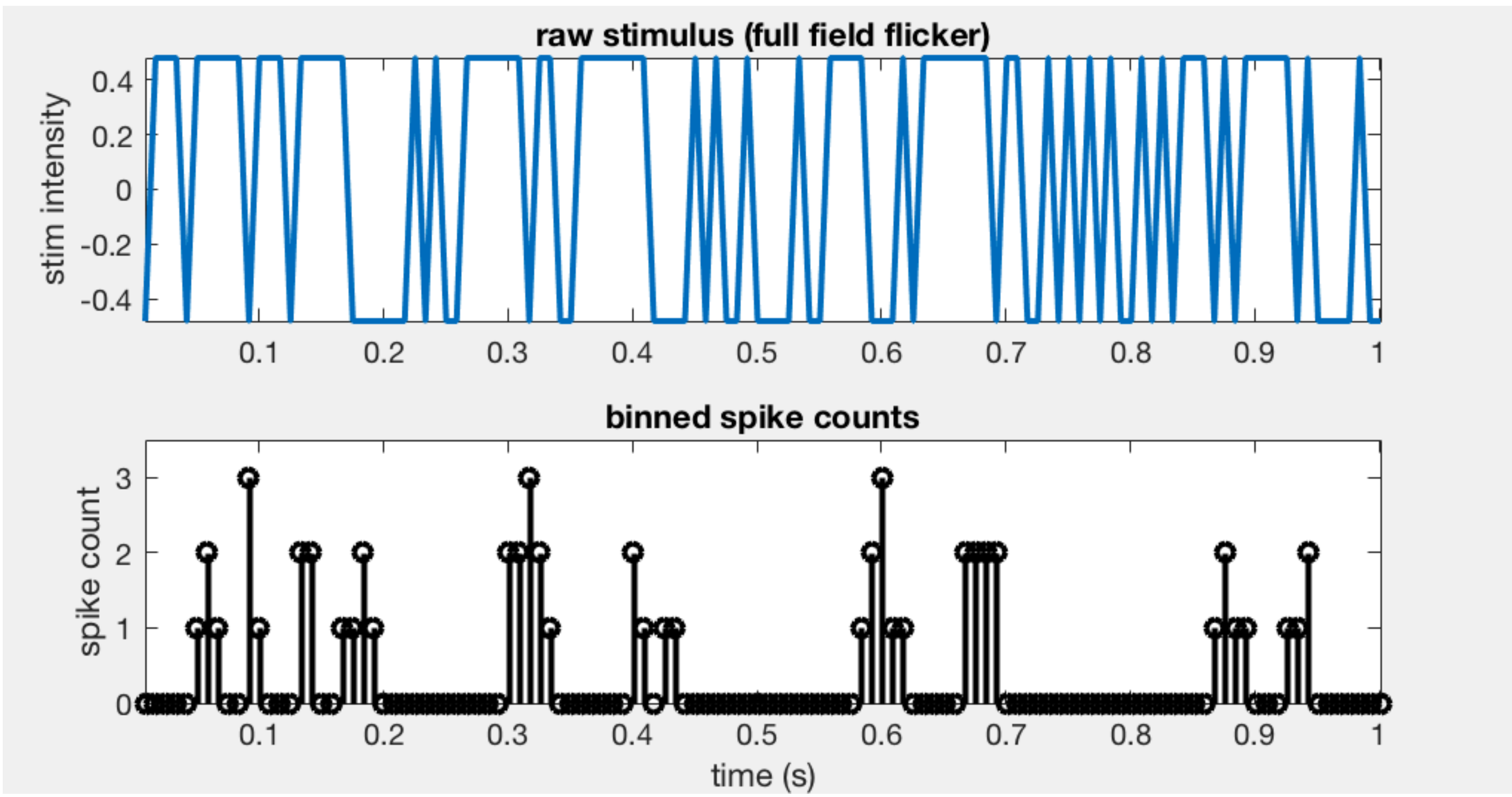
- stimulus = binary flicker
- parasol retinal ganglion cell spike responses



Uzzell et al (J Neurophys 04)

Example dataset

- stimulus = binary flicker
- parasol retinal ganglion cell spike responses

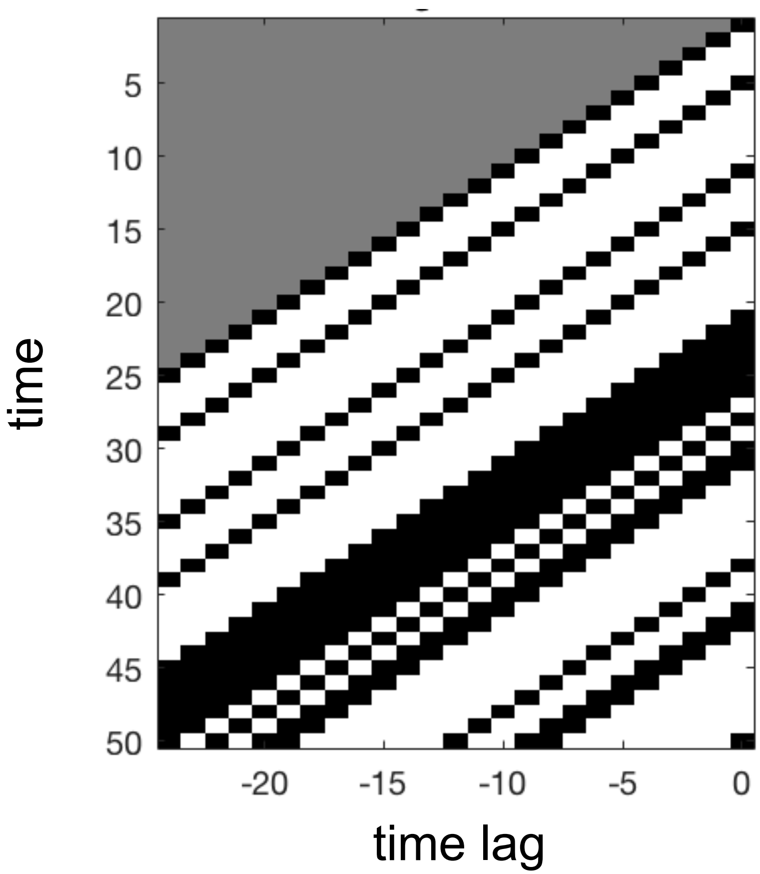


Uzzell et al (J Neurophys 04)

Stimulus-only GLM

design matrix

X



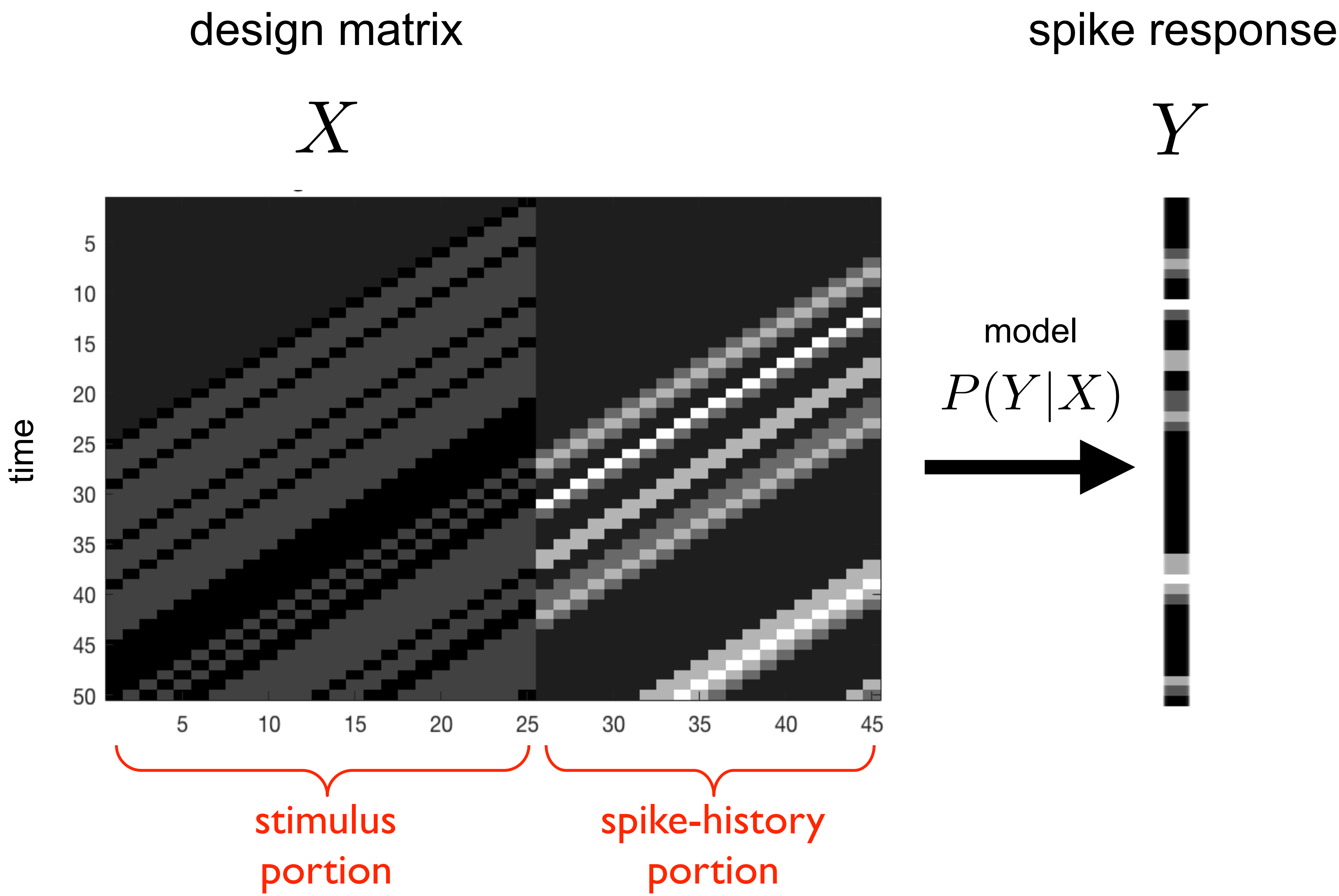
model

$P(Y|X)$

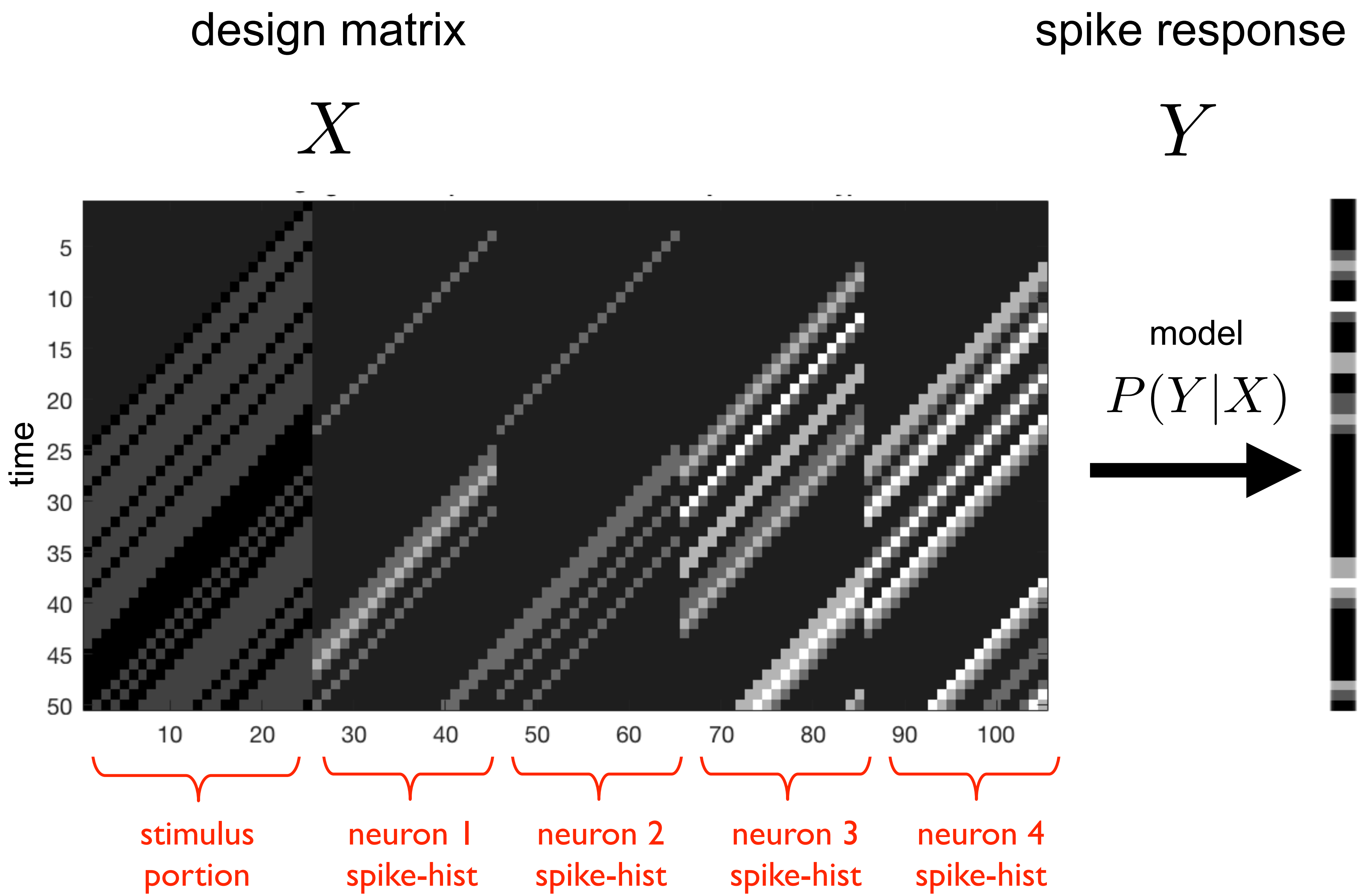
spike response

Y

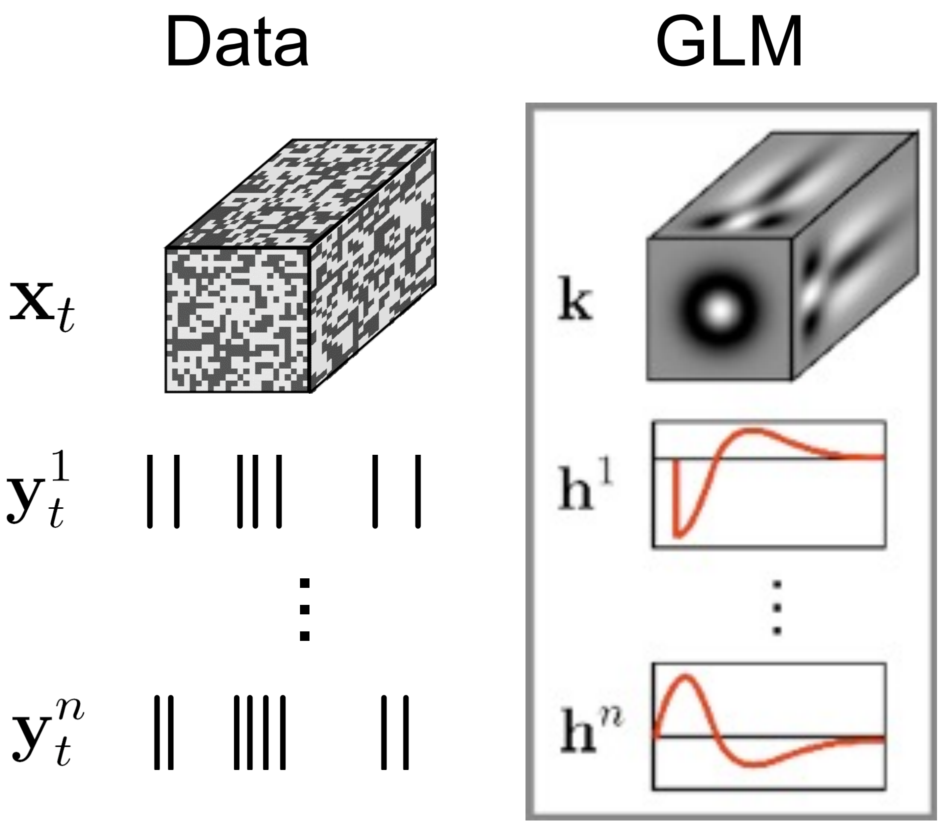
Stimulus + SpikeHistory GLM



Stimulus + History + 3 Neuron Coupling GLM



Fitting: Maximum Likelihood

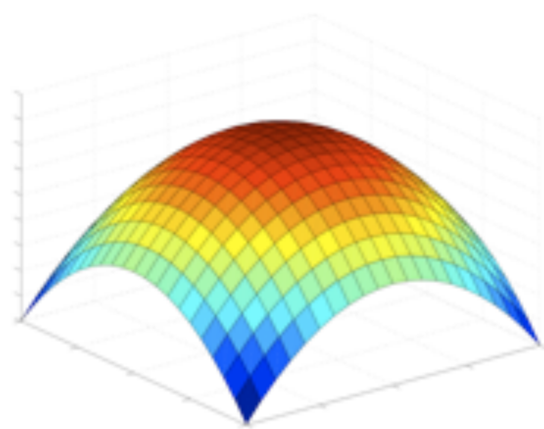


- maximize log-likelihood for filters $\{\mathrm{k}, \mathrm{h}_1, \mathrm{h}_2, \ldots \mathrm{h}_\mathrm{n}\}$

firing rate: $\lambda_t = f(\vec{x}_t \cdot \vec{k})$

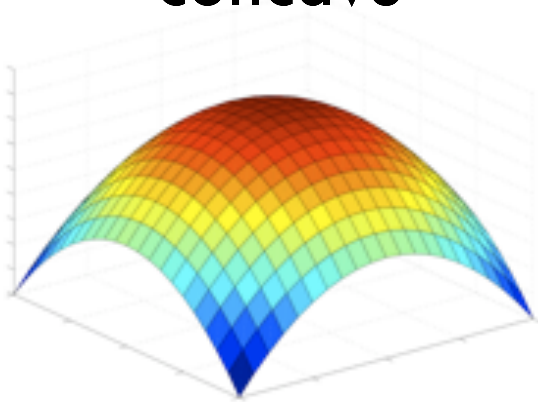
$$\log P(Y|X) = \sum_t y_t \log \lambda_t - \lambda_t$$

- log-likelihood is concave
- no local maxima [Paninski 04]



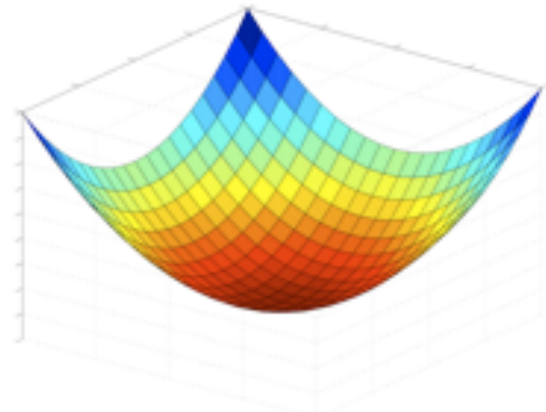
convexity and concavity

concave



- everywhere downward curvature

convex



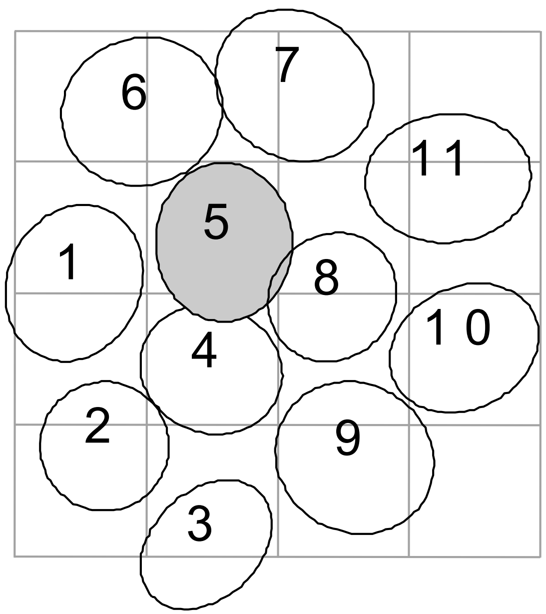
- everywhere upward curvature

- maximizing concave function $\Longleftrightarrow$ minimizing a convex function
- preclude existence of non-global local optima

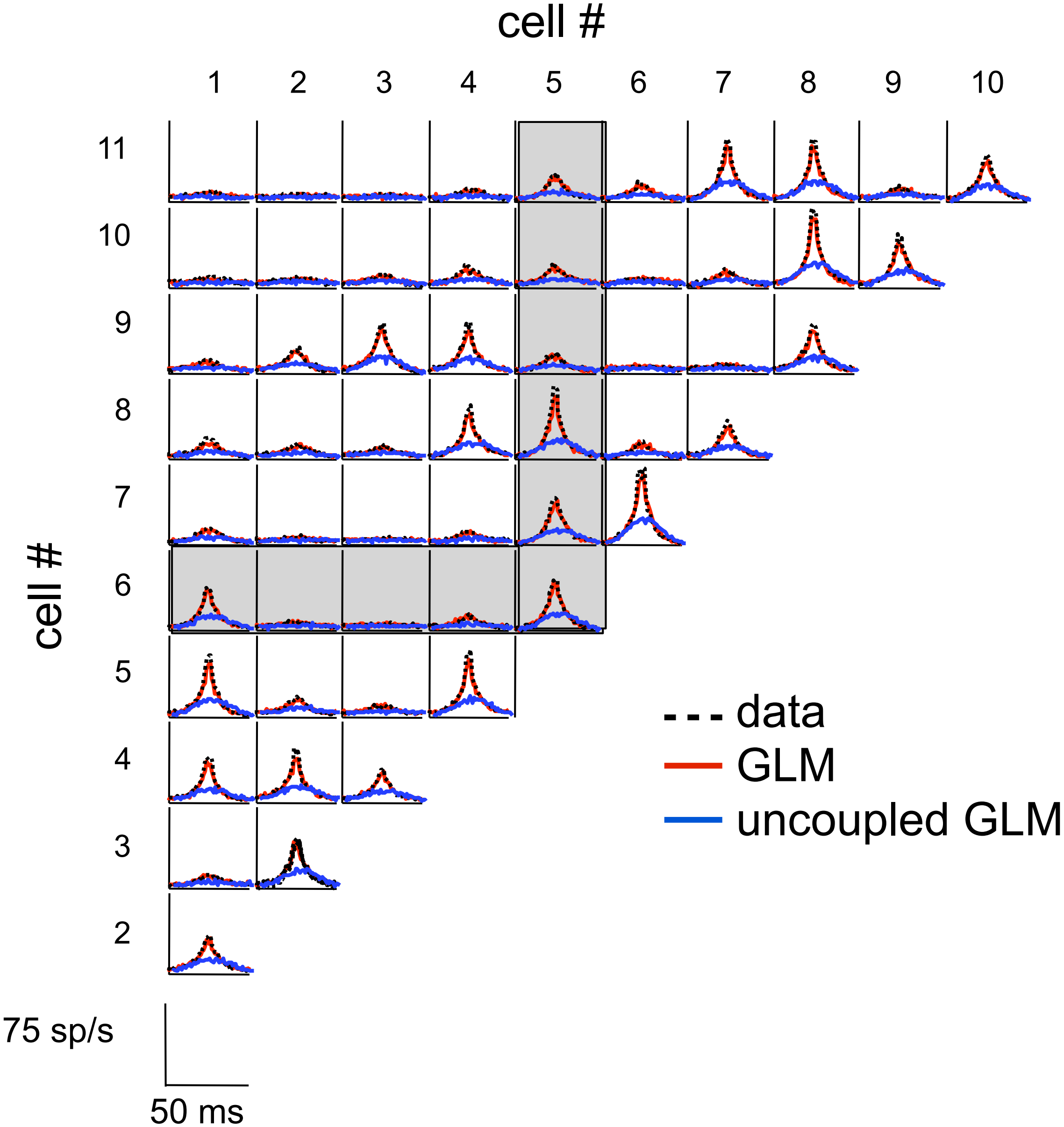
capturing dependencies in multi-neuron responses

[Pillow et al 2008]

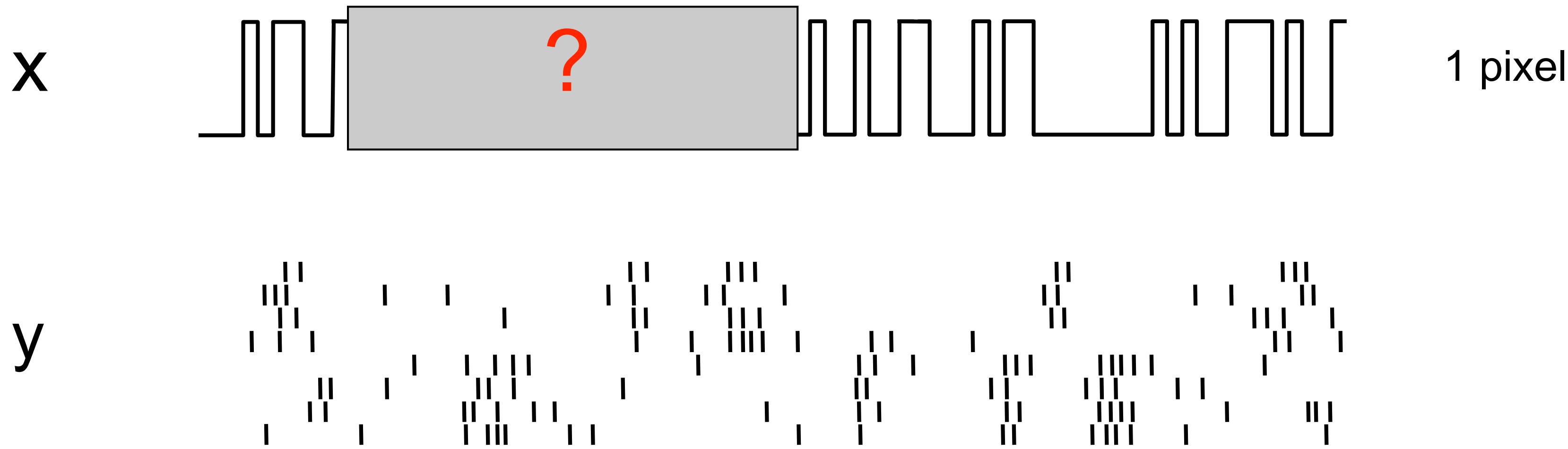
retinal receptive fields



cross-correlations

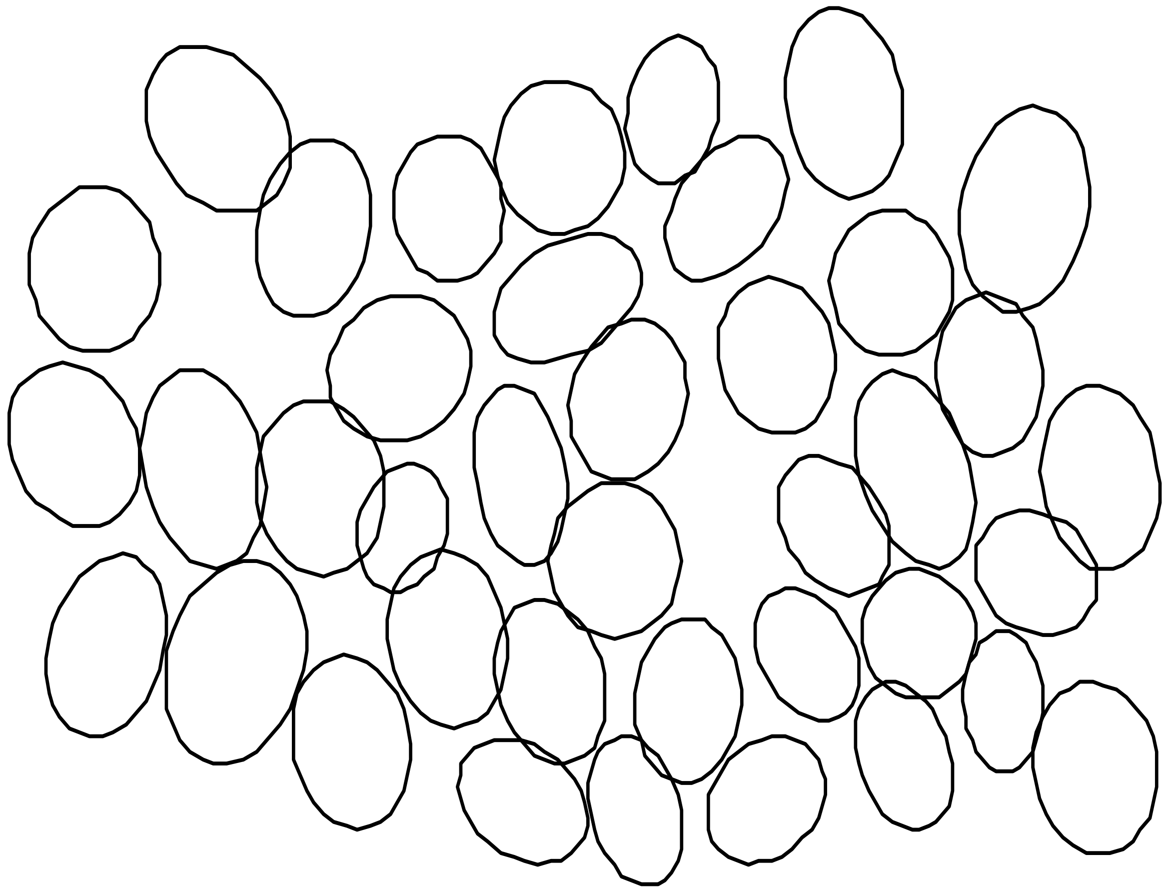


Decoding



- estimate stimuli from the observed spike times
- tool for comparing different encoding models

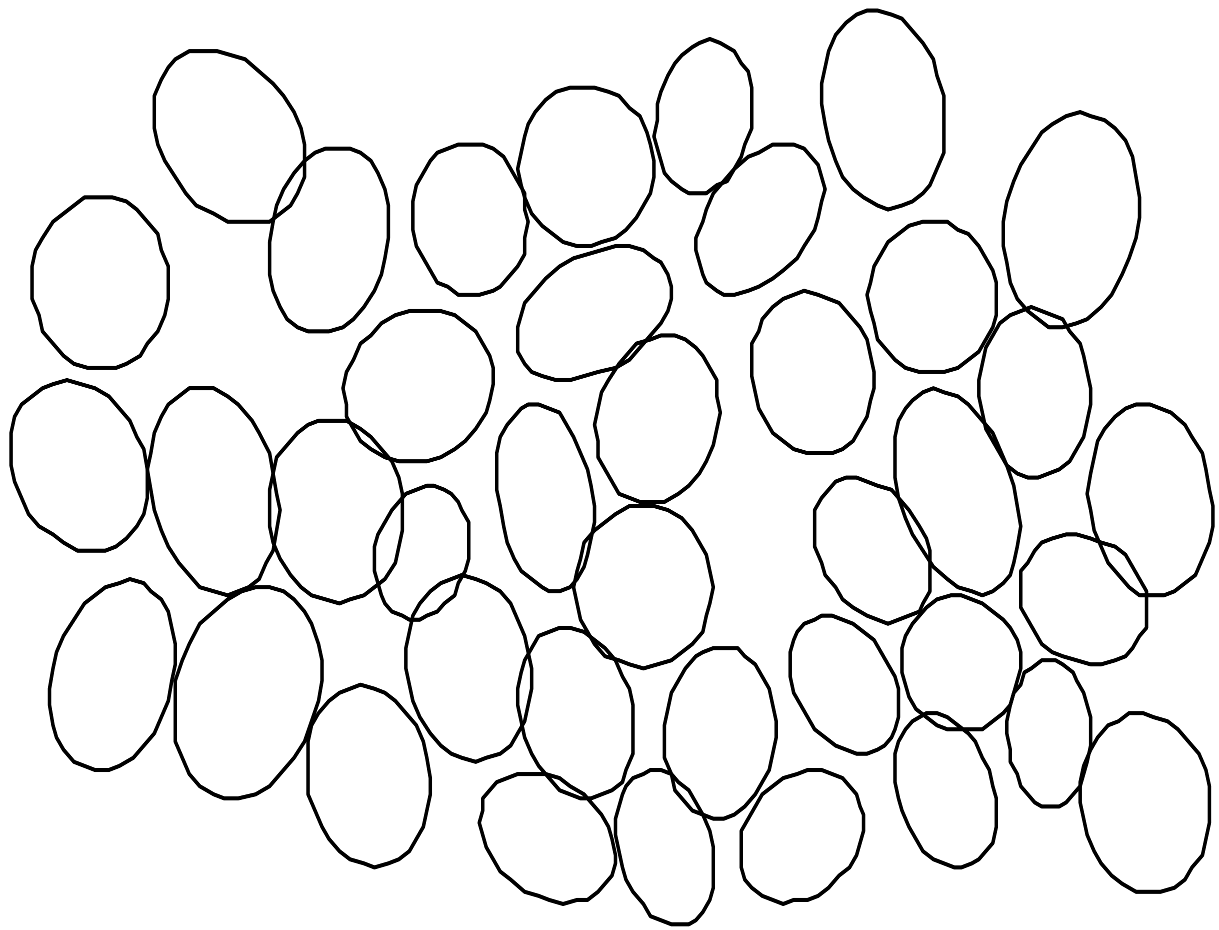
Decode: response 1



Q: what was the stimulus?

Frechette et al, 2005

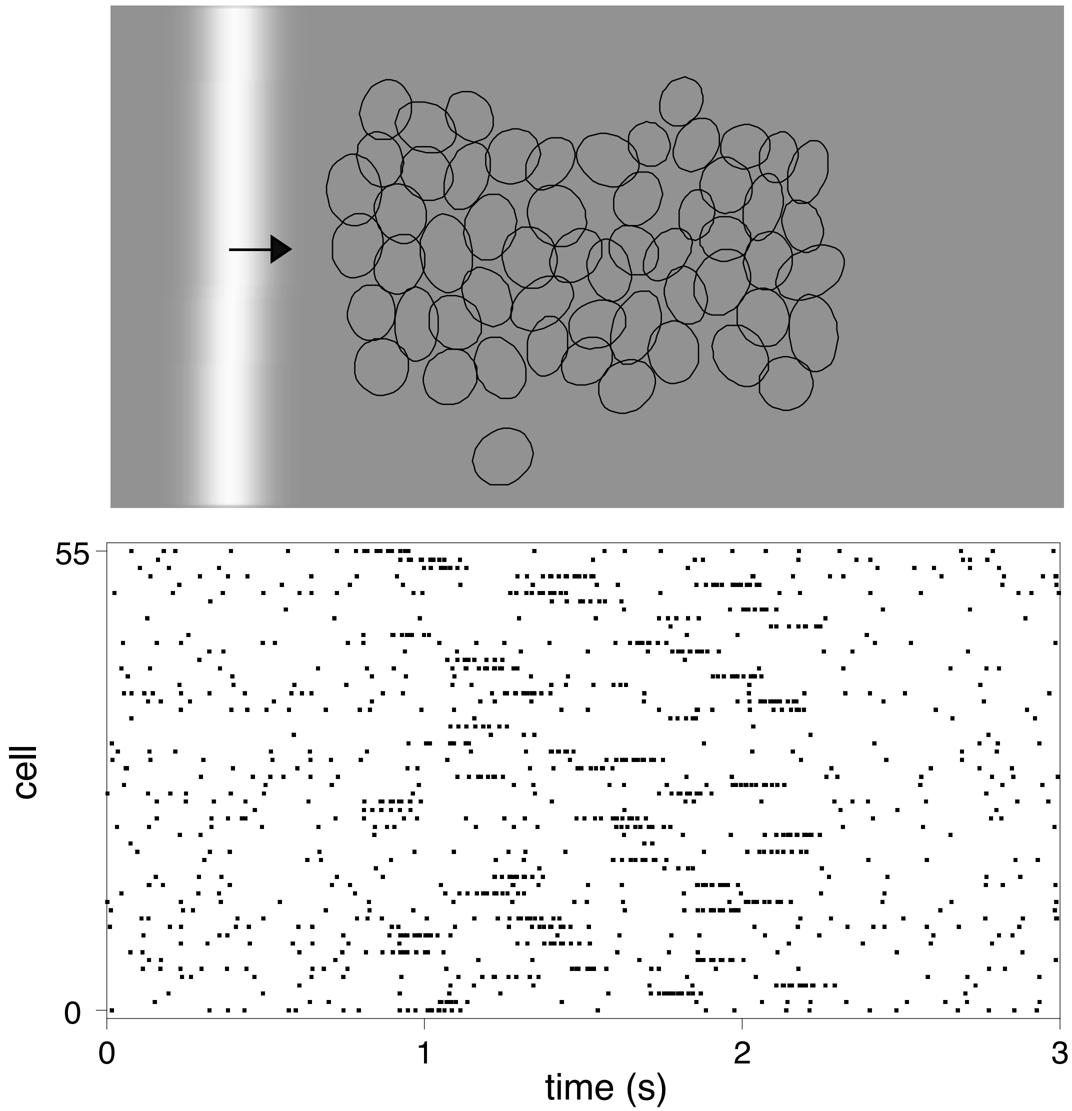
Decode: response 2



Q: what was the stimulus?

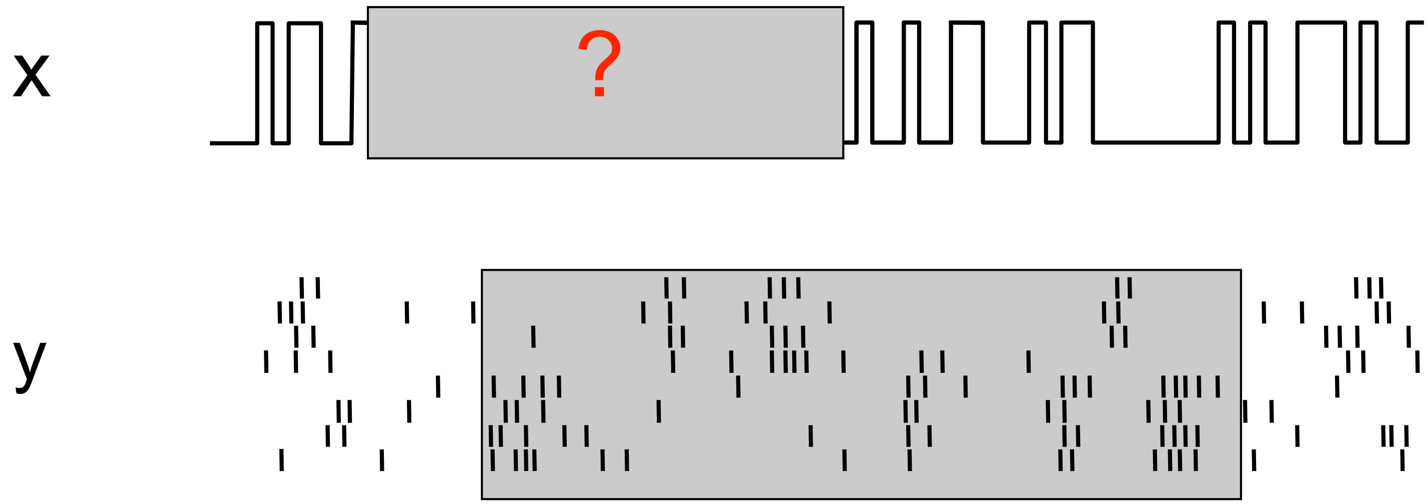
Frechette et al, 2005

Responses to Moving Bar



Frechette et al, 2005

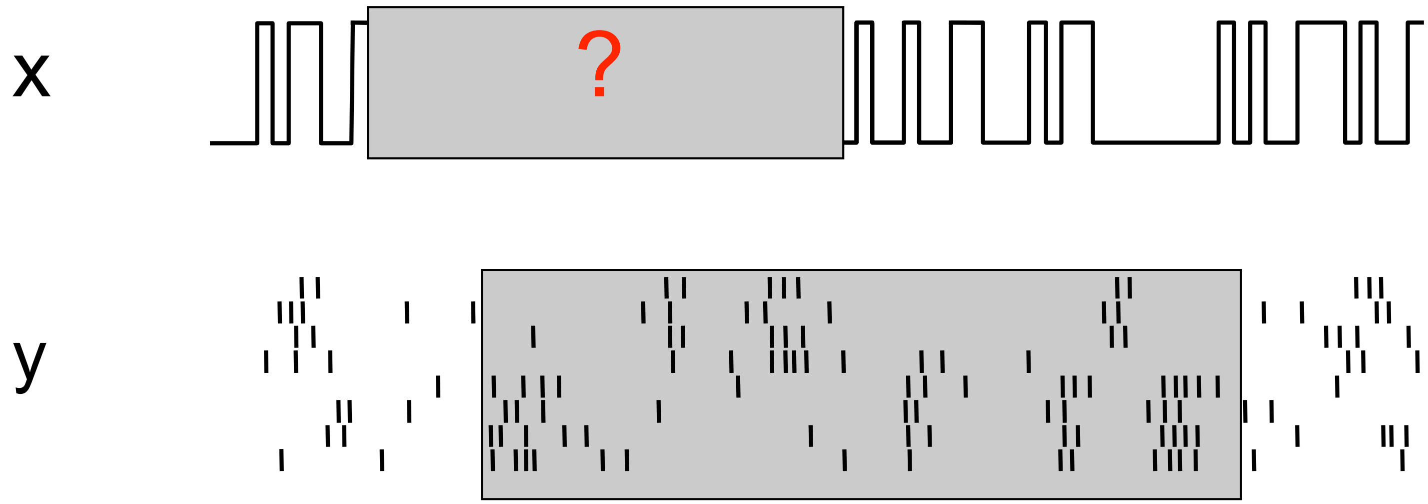
Bayesian Decoding



posterior likelihood prior

Bayes' rule: $P(x|y) \propto P(y|x)P(x)$

Bayesian Decoding



posterior likelihood prior

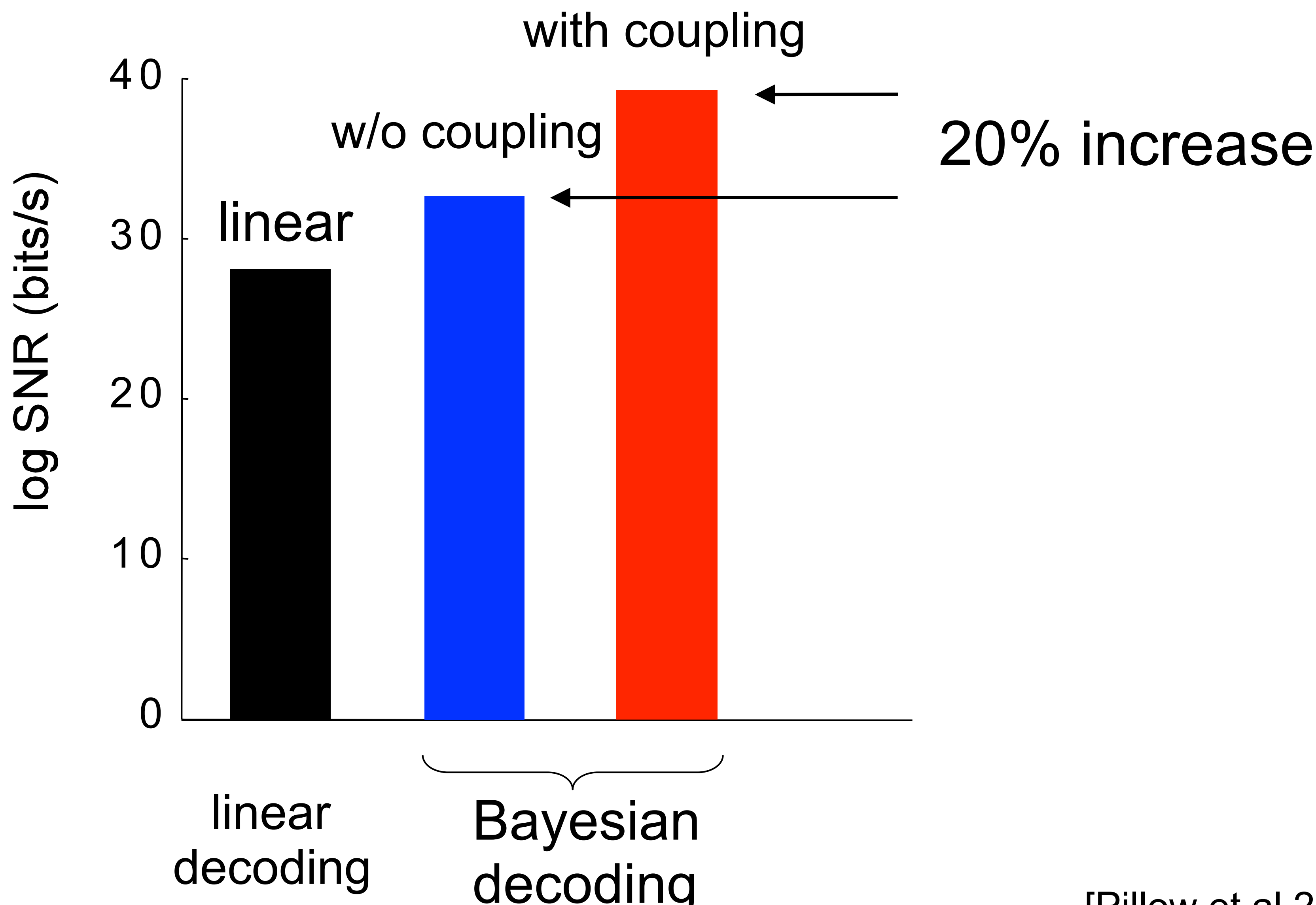
Bayes' rule: $P(x|y) \propto P(y|x)P(x)$

$P(y_1|x) \cdots P(y_n|x)$ vs. $P(y_1, y_2, \ldots, y_n|x)$

"independent"
(uncoupled GLM)

"joint encoding"
(coupled GLM)

Decoding Comparison



[Pillow et al 2008]

Regularization

Modern statistics

- more dimensions than samples $D \geq N$

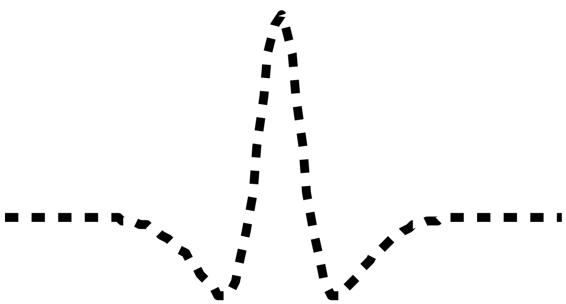
$$\begin{matrix}\text{N} \\ \text{observations}\end{matrix}\left\{\begin{bmatrix} y_1 \\ | \\ y_N \end{bmatrix}\right. = \begin{bmatrix} \overbrace{\text{———} \quad \vec{x}_1 \quad \text{———}}^{\text{D regressors}} \\ \\ \text{———} \quad \vec{x}_N \quad \text{———} \end{bmatrix} \begin{bmatrix} w_1 \\ | \\ | \\ w_D \end{bmatrix} + \text{noise}$$

- fewer equations than unknowns!
- no unique solution

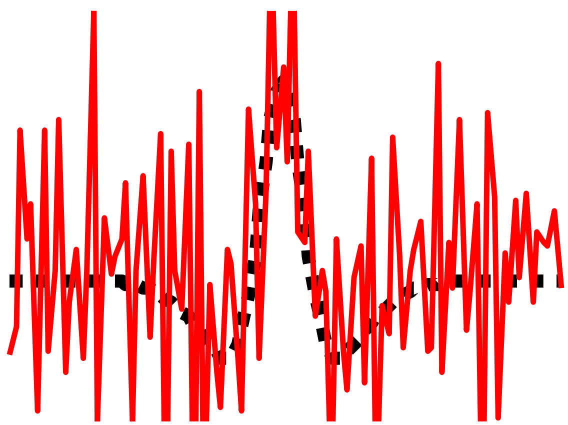
Simulated Example

- 100-element filter (D=100)
- 100 noisy samples (N=100)

true **w**



maximum likelihood



maximize

$$\log p(data|\mathbf{w})$$

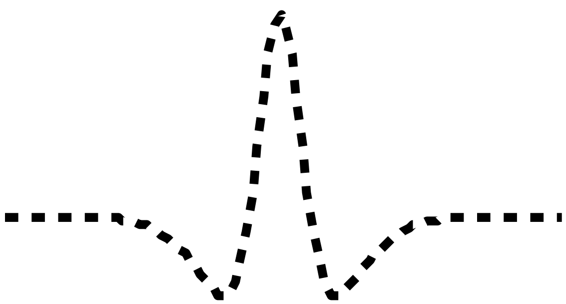
"overfitting" - parameters fit to details in the training data that are not useful for predicting new data

Simulated Example

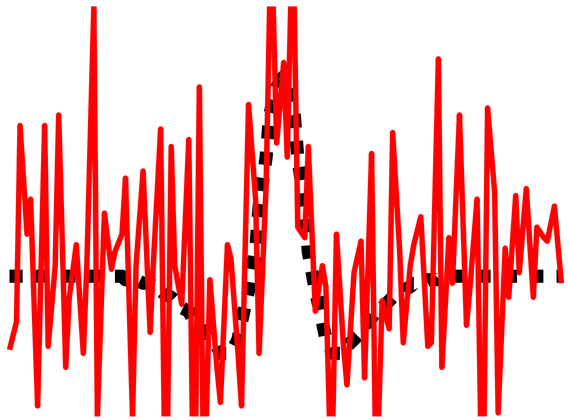
- 100-element filter (D=100)
- 100 noisy samples (N=100)

$$\hat{\mathbf{w}} = (X^\top X + \lambda I)^{-1} X^\top Y$$

true **w**



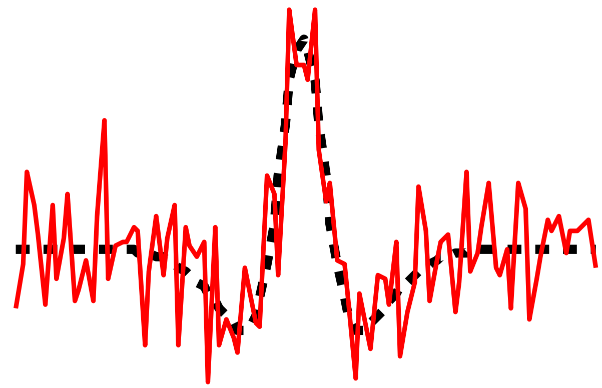
maximum likelihood



maximize

$$\log p(data|\mathbf{w})$$

"ridge regression"



maximize

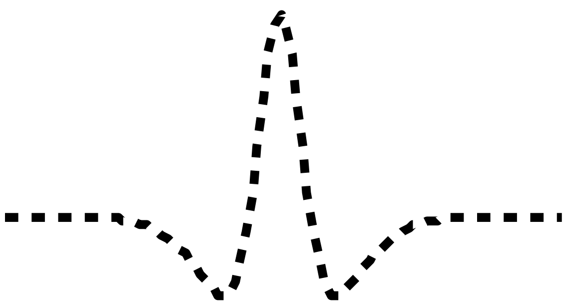
$$\log p(data|\mathbf{w}) \underbrace{- \lambda \sum w_i^2}_{\text{penalty on big weights}}$$

- biased, but gives improved performance for appropriate choice of λ (James & Stein 1960)

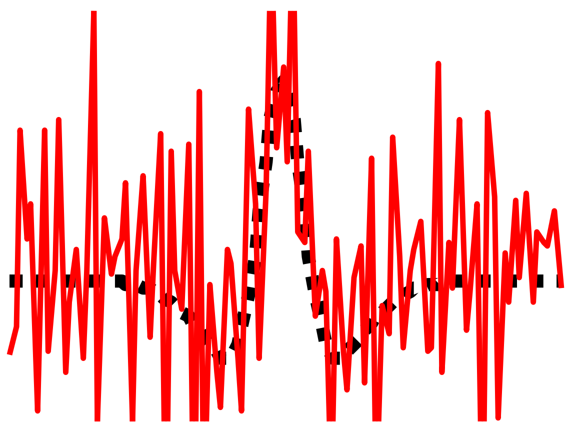
Simulated Example

- 100-element filter (D=100)
- 100 noisy samples (N=100)

true **w**



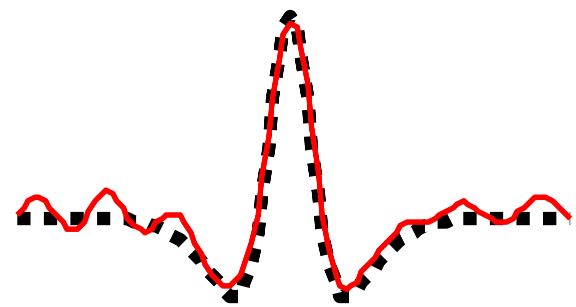
maximum likelihood



maximize

$$\log p(data|\mathbf{w})$$

"smoothed"



maximize

$$\log p(data|\mathbf{w}) \underbrace{-\lambda \sum (w_i - w_{i-1})^2}_{\text{smoothness penalty}}$$

Q: how to set the regularization strength λ ?

Simplest answer: use cross-validation!

GLM tutorial (matlab):

code: https://github.com/pillowlab/GLMspiketraintutorial
data: available on request from pillow@princeton.edu

- **tutorial1_PoissonGLM.m** - fitting of a linear-Gaussian GLM and Poisson GLM (aka LNP model) to RGC neurons stimulated with temporal white noise stimulus.
- **tutorial2_spikehistcoupledGLM.m** - fitting of a Poisson GLM with spike-history and coupling between neurons.
- **tutorial3_regularization_linGauss.m** - regularizing linear-Gaussian model parameters using maximum a posteriori (MAP) estimation under two kinds of priors:
  - (1) ridge regression (aka "L2 penalty");
  - (2) L2 smoothing prior (aka "graph Laplacian").
- **tutorial4_regularization_PoissonGLM.m** - MAP estimation of Poisson-GLM parameters using same two priors as in tutorial3.

GLM summary

- linear (“dim reduction”) + nonlinear + noise
- incorporate spike-history via “spike history” filter
- rich dynamical properties: refractoriness, bursting, adaptation
- incorporate correlations between neurons via “coupling” filters
- flexible tool for encoding & decoding analyses
- regularize to reduce overfitting (essential w/ correlated stimuli)

Beyond GLM

polynomial models

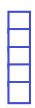
Lee & Schetzen 1965
Marmarelis & Naka 1972
Korenberg & Hunter1986

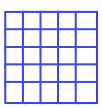
Volterra / Wiener Kernels

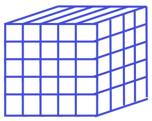
Taylor series expansion of a function f(x) in *n* dimensions

$$y \;=\; k_0 + \vec{k}_1 \cdot \vec{x} + \vec{x}^t K_2 \vec{x} + K_3 \cdot \vec{x}^3 + \ldots$$

const


vector


matrix


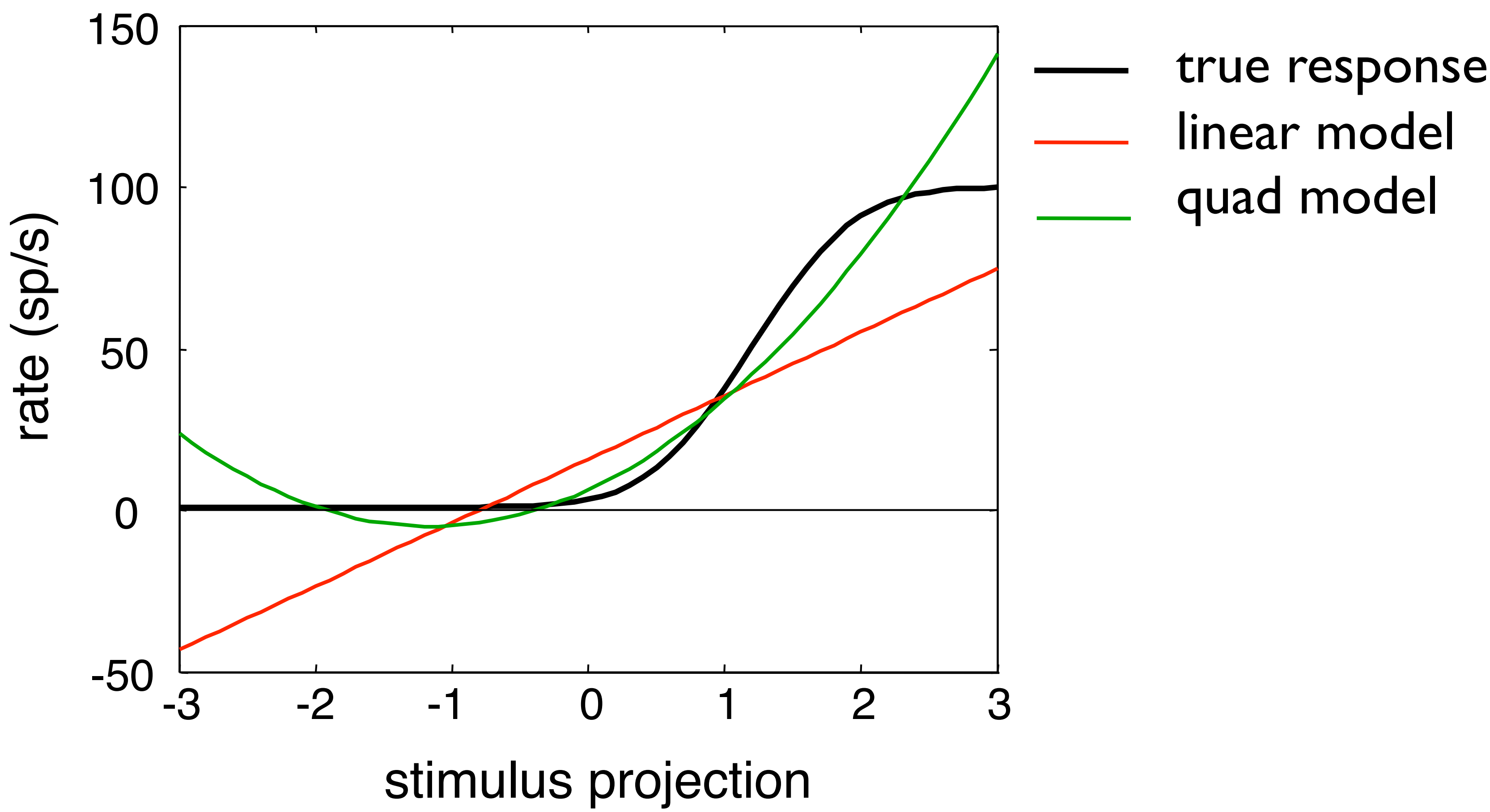
3-tensor


# parameters:	1	n (20)	n^2 (400)	n^3 (8000)

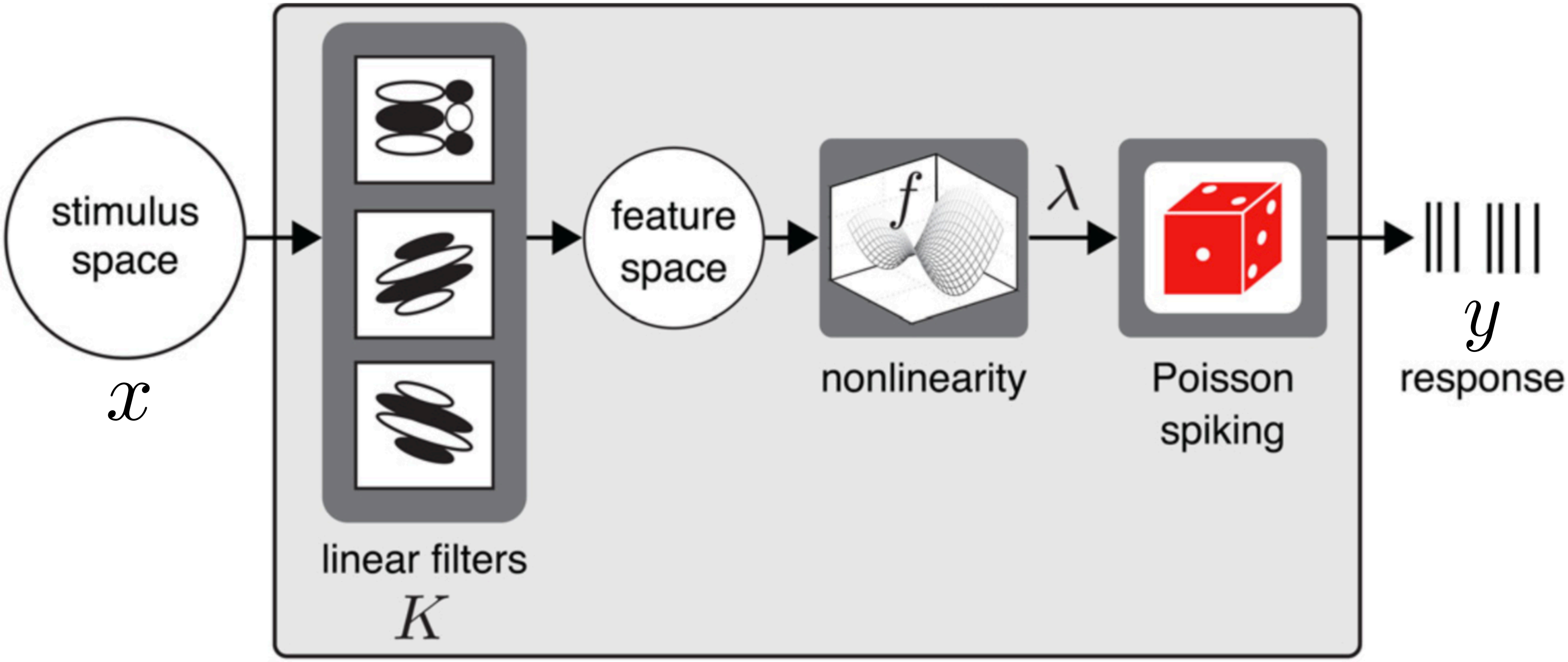
- from "systems identification" literature (1960s-70s)
- white noise stimuli
- estimate kernels using moments of spike-triggered stimuli

Why are Volterra/Wiener models (generally) bad?

- no output nonlinearity
- polynomials give poor fit to neural nonlinearities (e.g., rectifying, saturating)



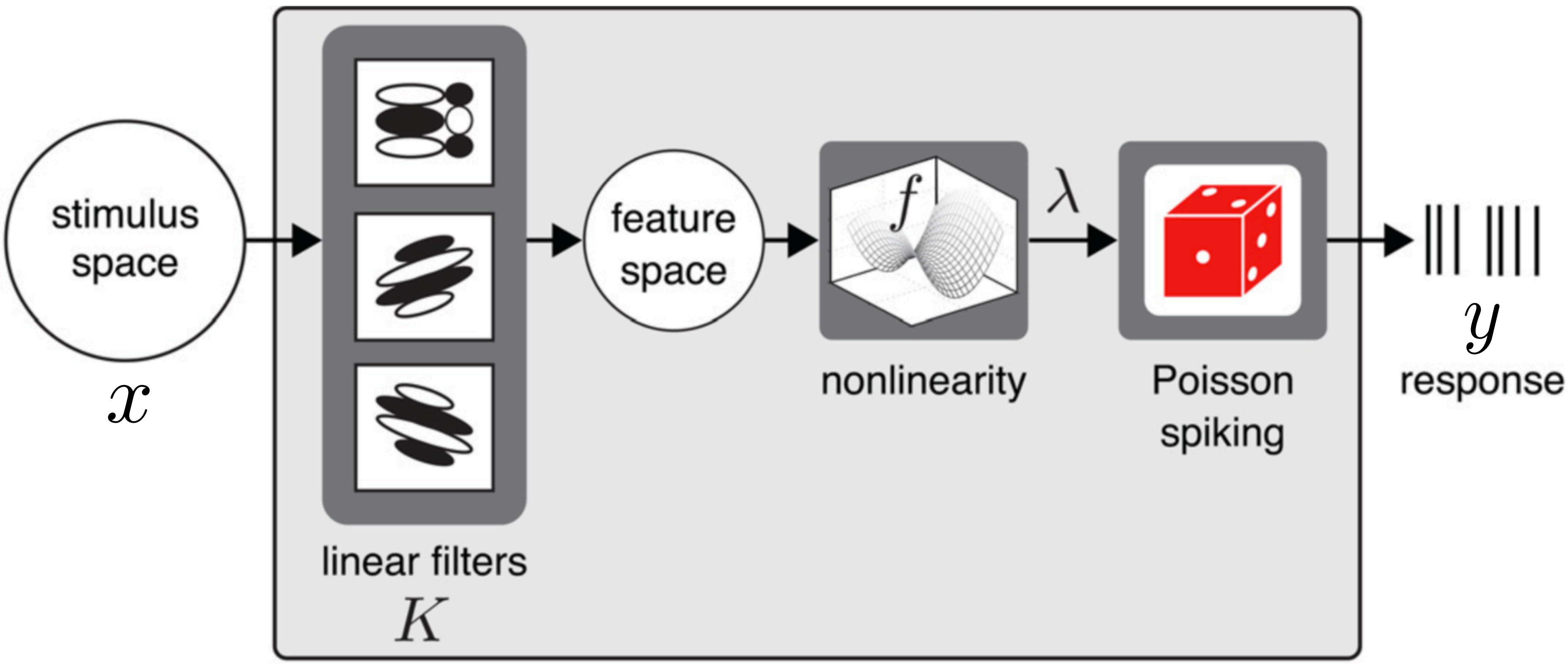
multi-filter LNP



- responses may depend on more than one projection of stimulus!
- emphasis on *dimensionality reduction*

- no longer technically a GLM if fitting nonlinearity *f*

multi-filter LNP



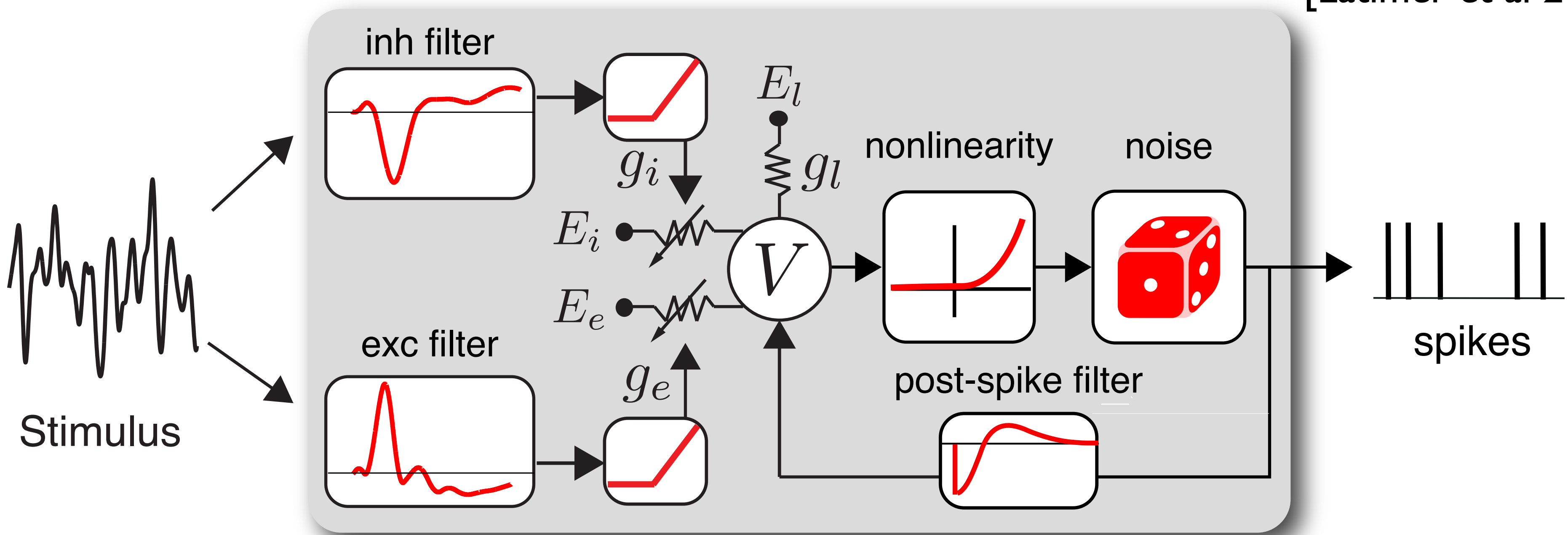
Estimators:

- Spike-triggered covariance (STC) [de Ruyter & Bialek 1998, Schwartz et al 2006]
- Generalized Quadratic Model (GQM) [Park & Pillow 2011; Park et al 2013; Rajan et al 2013]
- maximally informative dimensions (MID) / maximum likelihood

  [Sharpee et al 2004] [Williamson et al 2015]

extending GLM to conductance-based model

[Latimer et al 2014]



conductances

$$g_e(t) = f_c(k_e \cdot \mathbf{x}(t))$$

$$g_i(t) = f_c(k_i \cdot \mathbf{x}(t))$$

membrane dynamics

$$\frac{dV}{dt} = g_l(E_l - V) + g_e(E_e - V) + g_i(E_i - V)$$

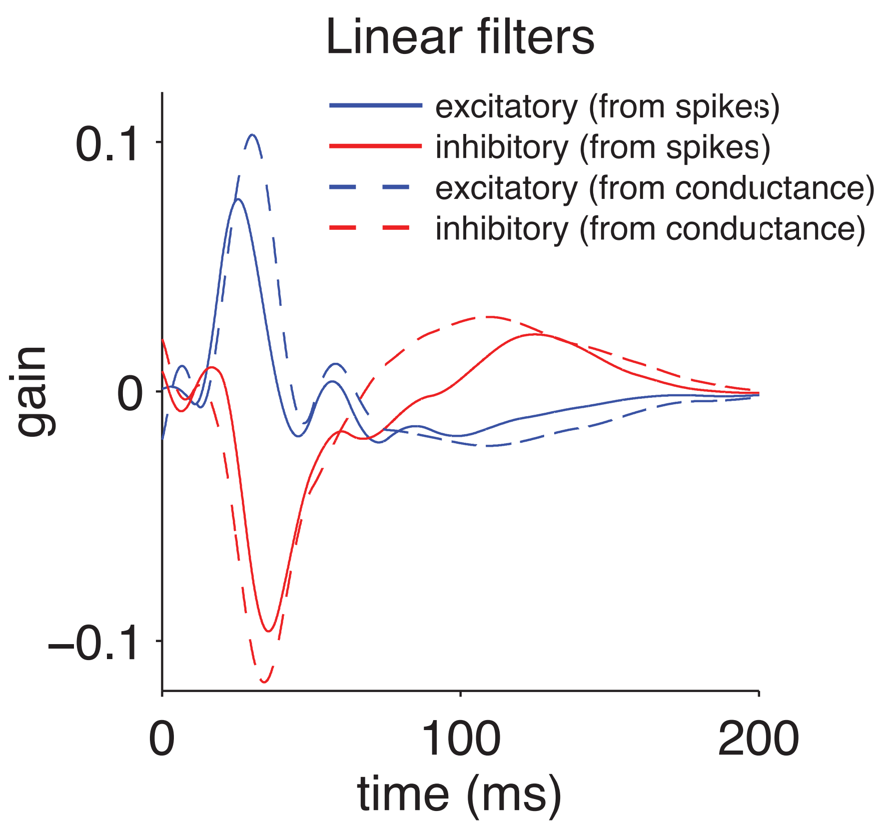
inst. spike rate

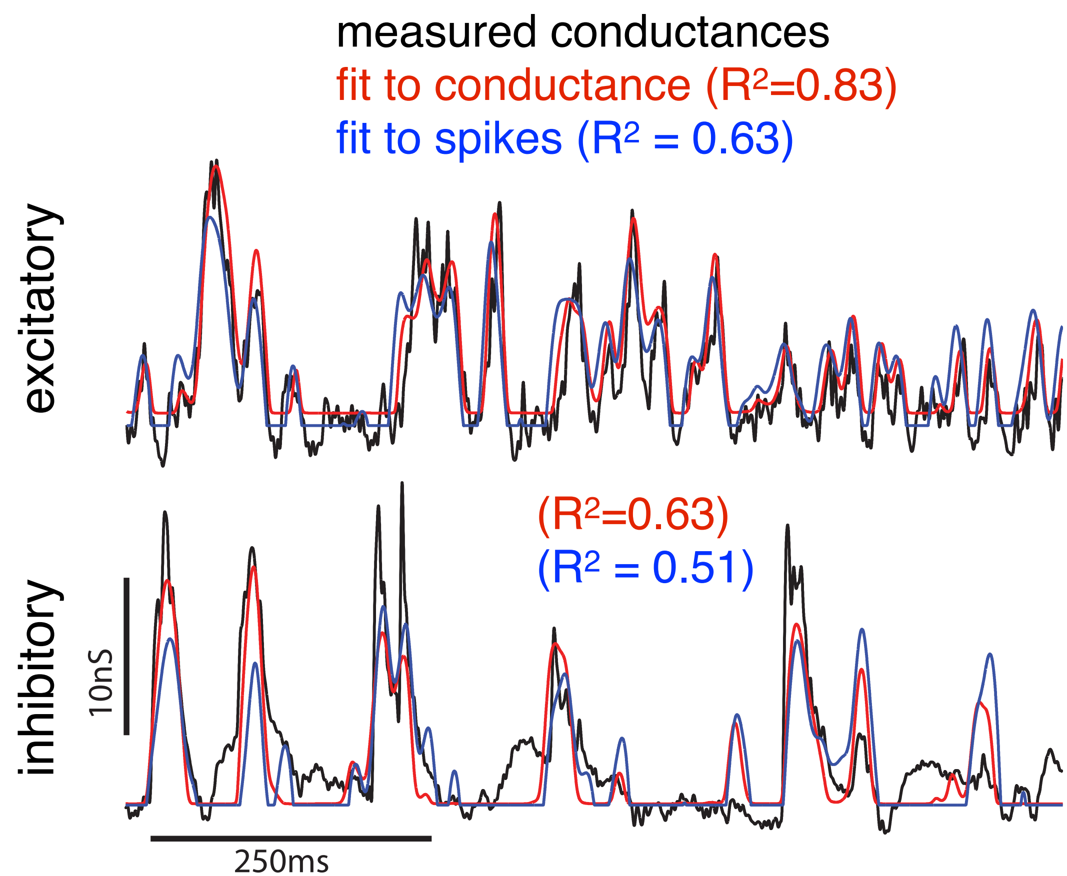
$$\lambda(t) = f(V(t))$$

- shunting inhibition
- adaptive changes in dynamics

extending GLM to conductance-based model

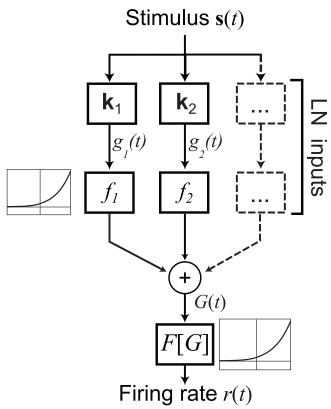
- intracellular recordings in macaque parasol RGCs (Fred Rieke)



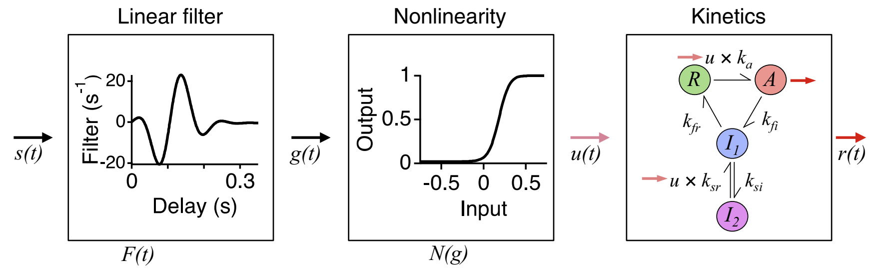


[Latimer et al 2014]

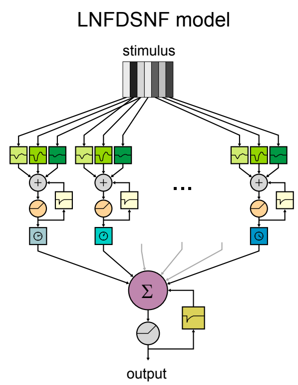
many other biophysically oriented extensions



Nonlinear input model (NIM)
[McFarland, Cui, & Butts 2013]



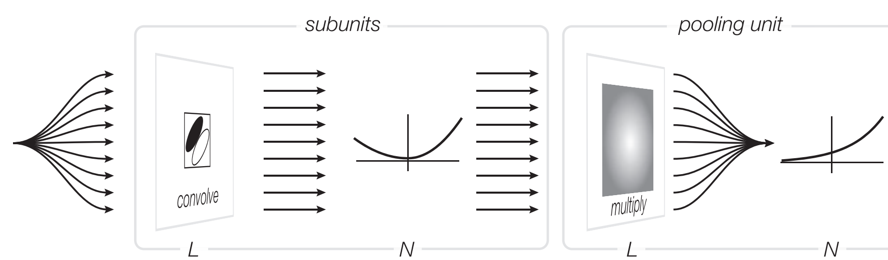
Linear-Nonlinear-Kinetics (LNK)
[Ozuysal & Baccus 2014]



LNFDSNF model

linear-nonlinear-feedback-delayed-sum-nonlinear-feedback

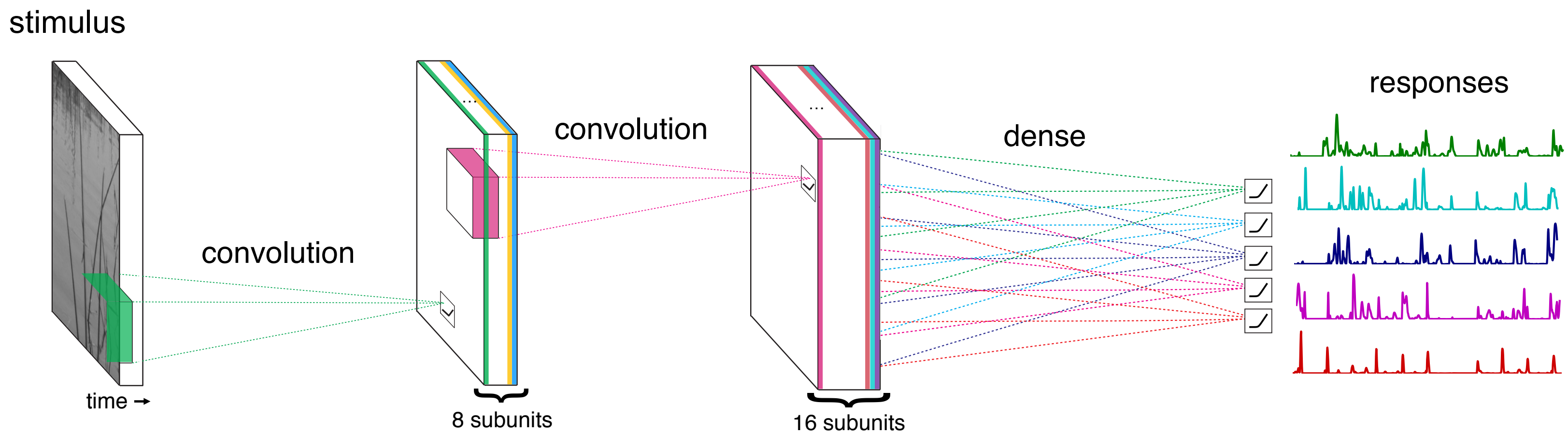
[Real, Asari, Gollisch & Meister 2017]



convolutional subunit model

[Vintch, Movshon & Simoncelli 2015,
Wu, Park & Pillow 2014]

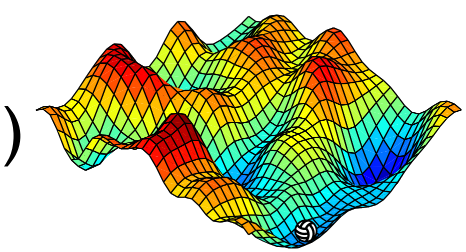
deep learning / deep neural networks (DNNs)



[Yamins et al 2014, Mcintosh et al 2016, Maheswaranathan et al 2017, Benjamin et al 2017, …]

If you understand GLMs… you understand DNNs!

- stack many LNs on top of each other: LN LN LN LN P
- use gradient ascent to maximize likelihood
- use software (tensorflow, theano) to compute gradients (no more computing gradients by hand!)
- use a bunch of tricks (batches, noise, SGD, dropout, ….)
- do NOT worry about local maxima!



[credit: Jakob Macke]

Modern machine learning far outperforms GLMs at predicting spikes

Ari S. Benjamin[1], Hugo L. Fernandes[2], Tucker Tomlinson[3], Pavan Ramkumar[2,4], Chris VerSteeg[1], Lee Miller[1,2,3], Konrad Paul Kording[1,2,3]

macaque M1

Fig 2

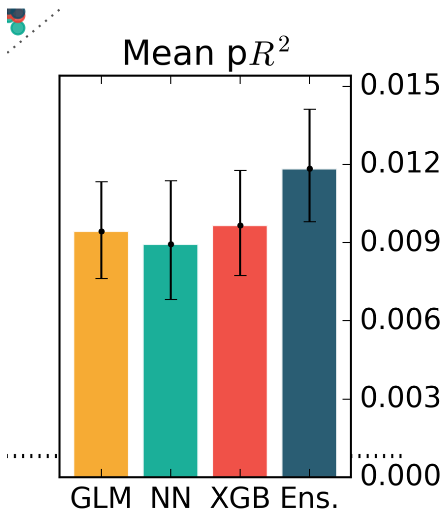


Fig 3

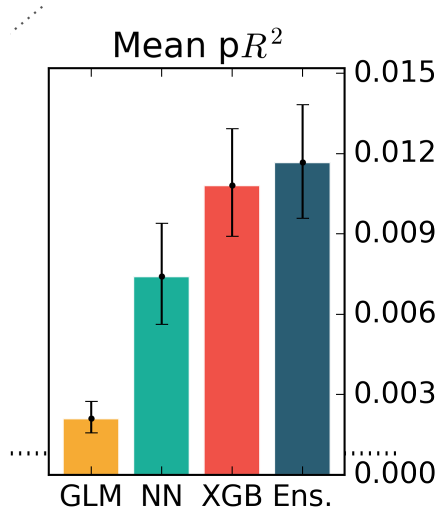


Fig 4

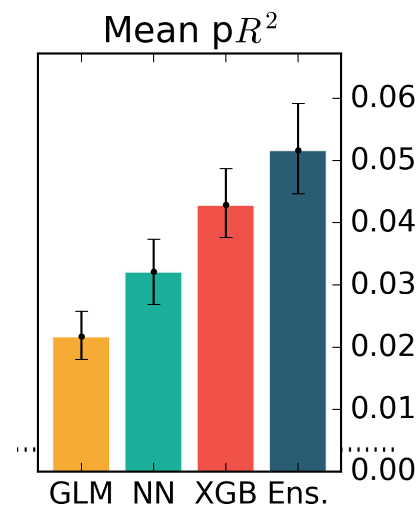


Fig 5

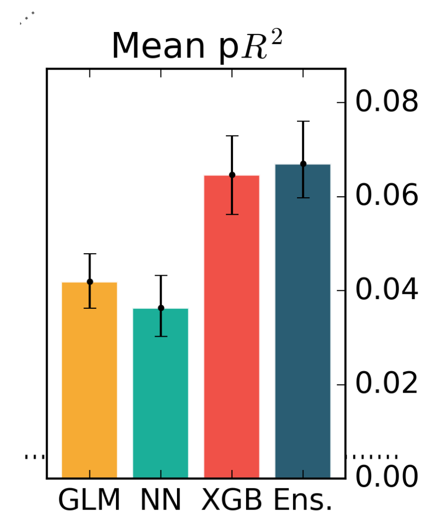
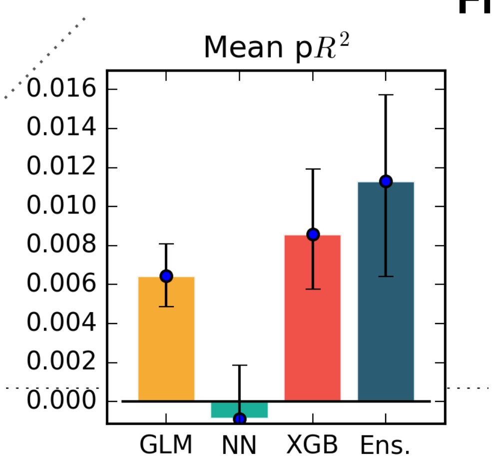
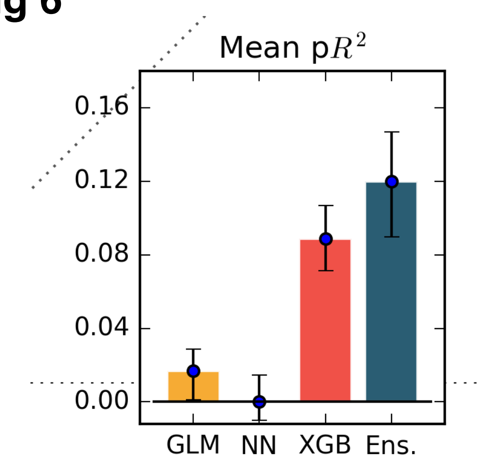


Fig 6



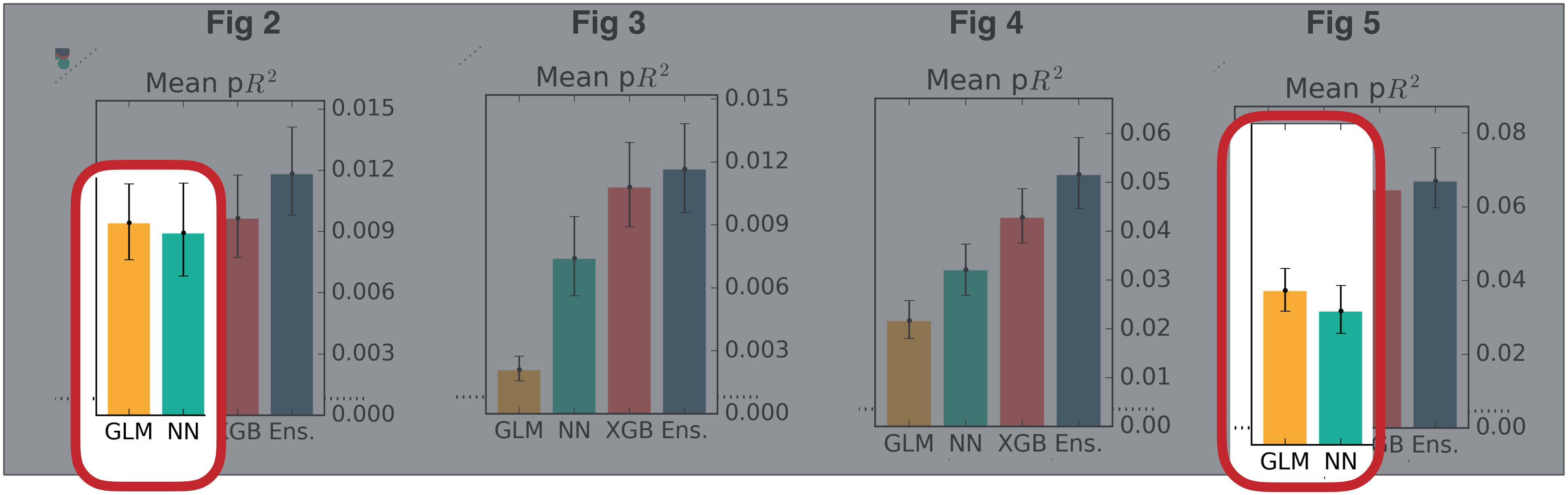
macaque S1

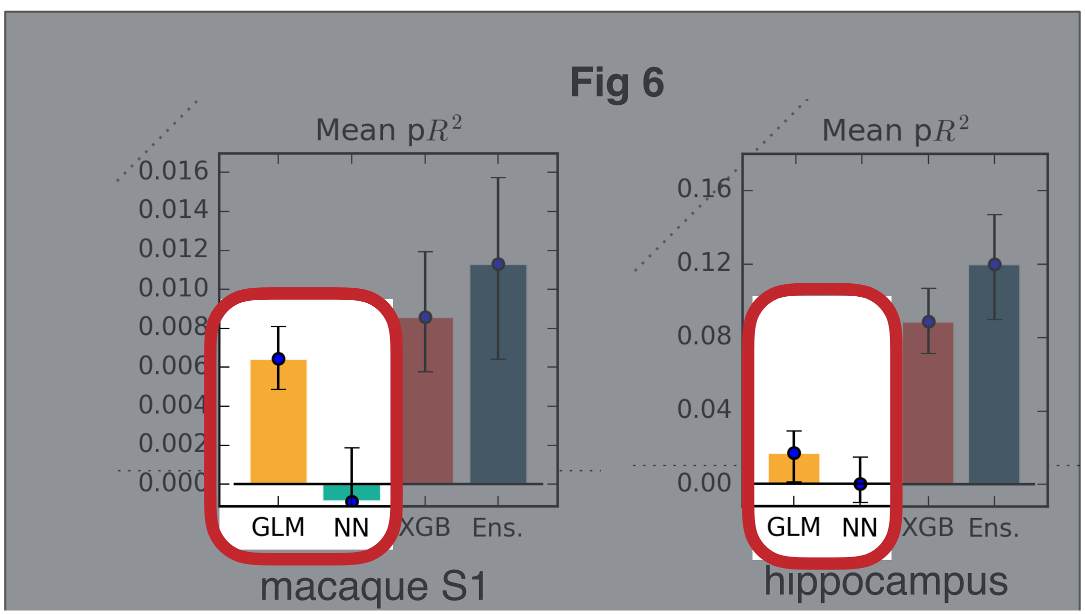


hippocampus

~~Modern machine learning far~~ GLMs outperforms ~~GLMs~~ NNs at predicting spikes ?

Ari S. Benjamin[1], Hugo L. Fernandes[2], Tucker Tomlinson[3], Pavan Ramkumar[2,4], Chris VerSteeg[1], Lee Miller[1,2,3], Konrad Paul Kording[1,2,3]





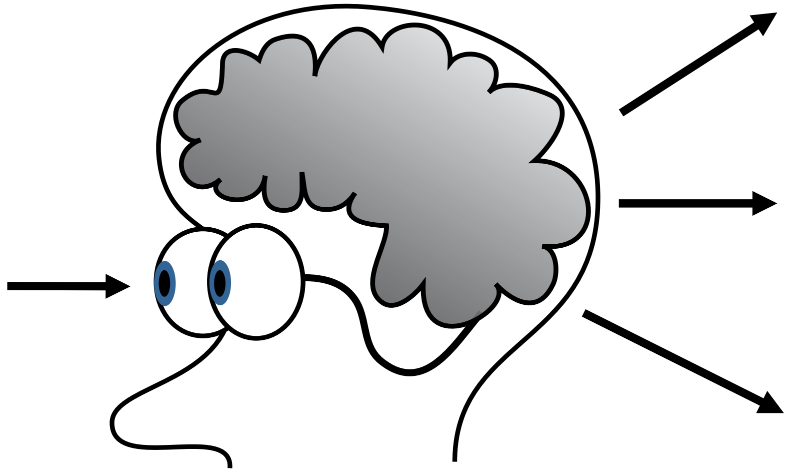
(No of course not!)

GLM is a special case of NN!

encoding models

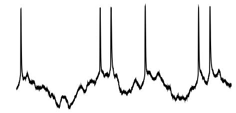
stimulus

x

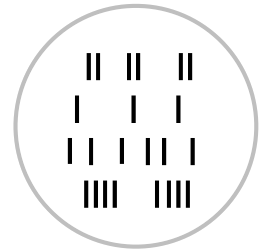


$$P(y|x)$$

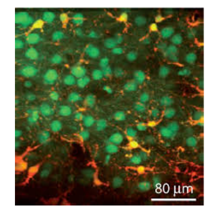
encoding models



membrane potential



spikes



imaging

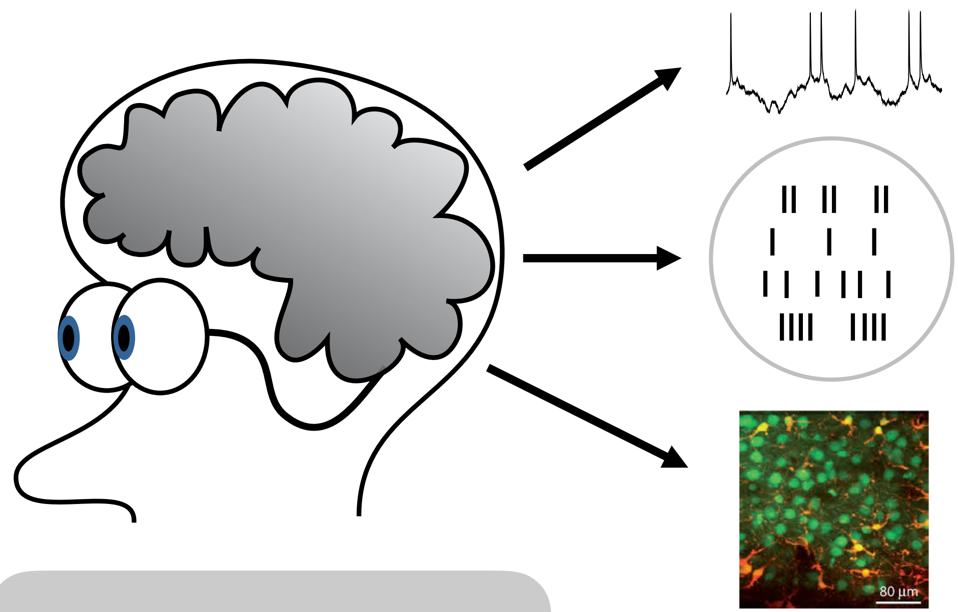
neural activity

y

What if there's no stimulus?

encoding models

What if there's no stimulus?



membrane potential

spikes

imaging

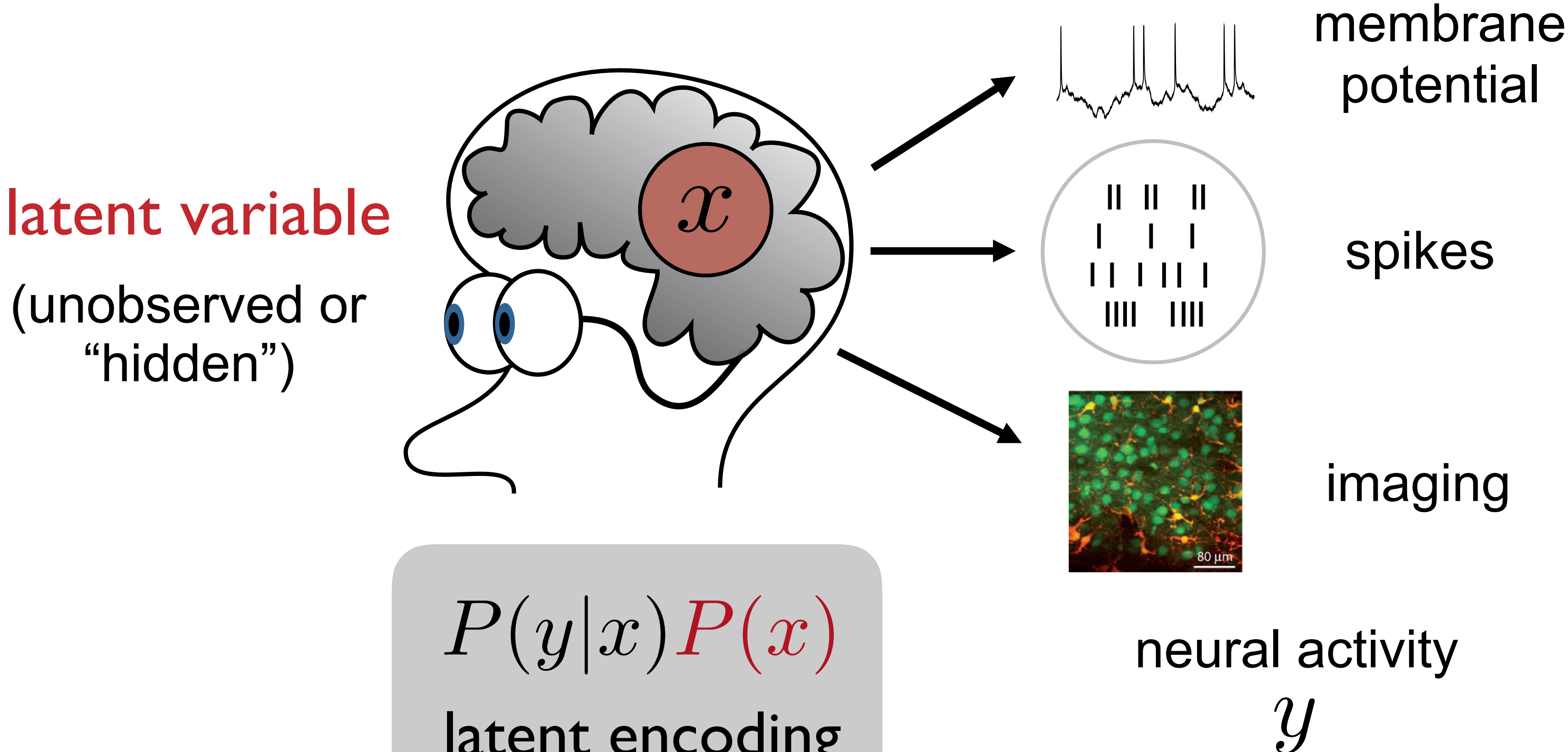
$$P(y|x)$$

encoding models

neural activity

y

latent variable models

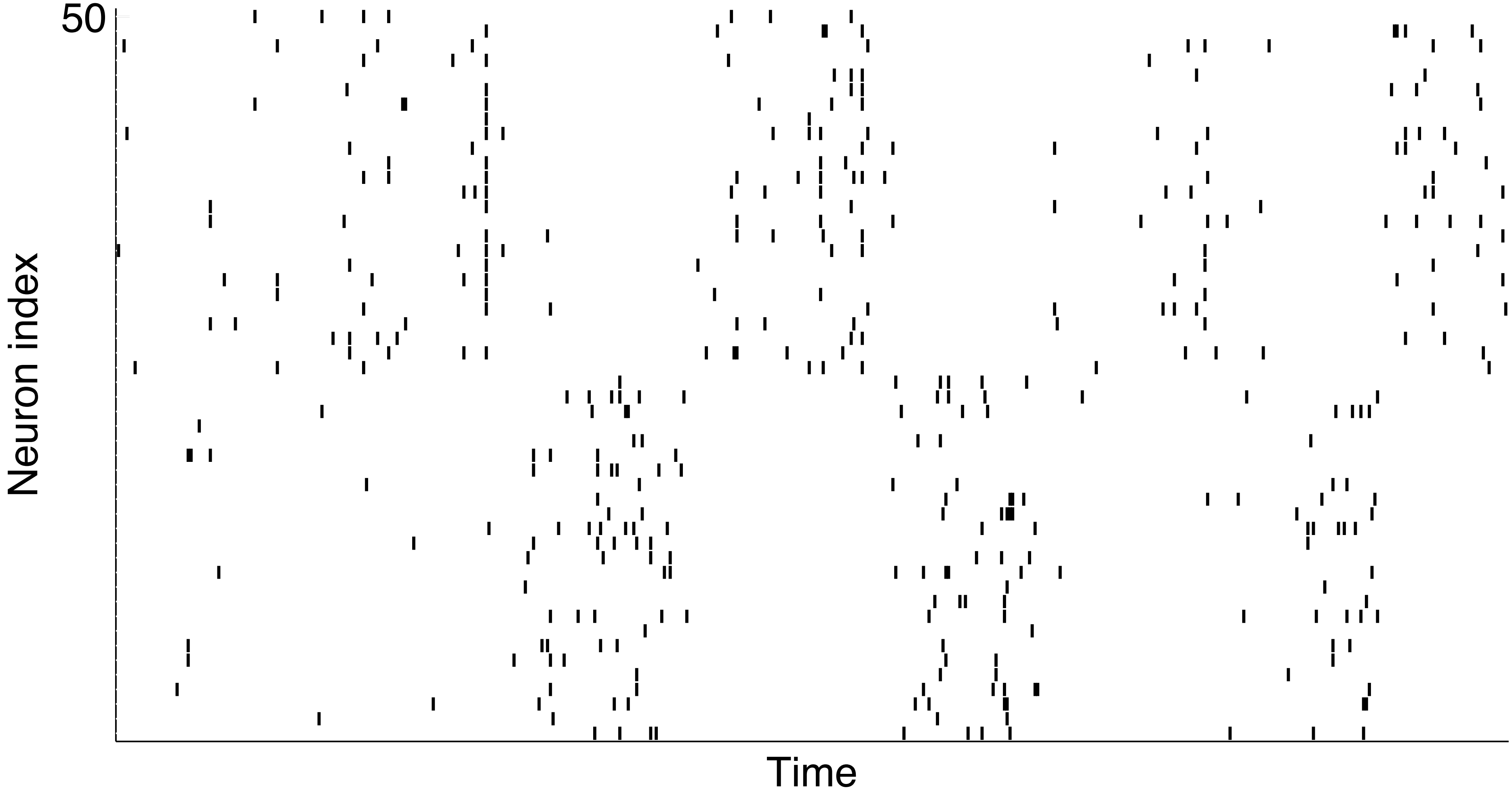


$P(y|x)P(x)$

latent encoding
models

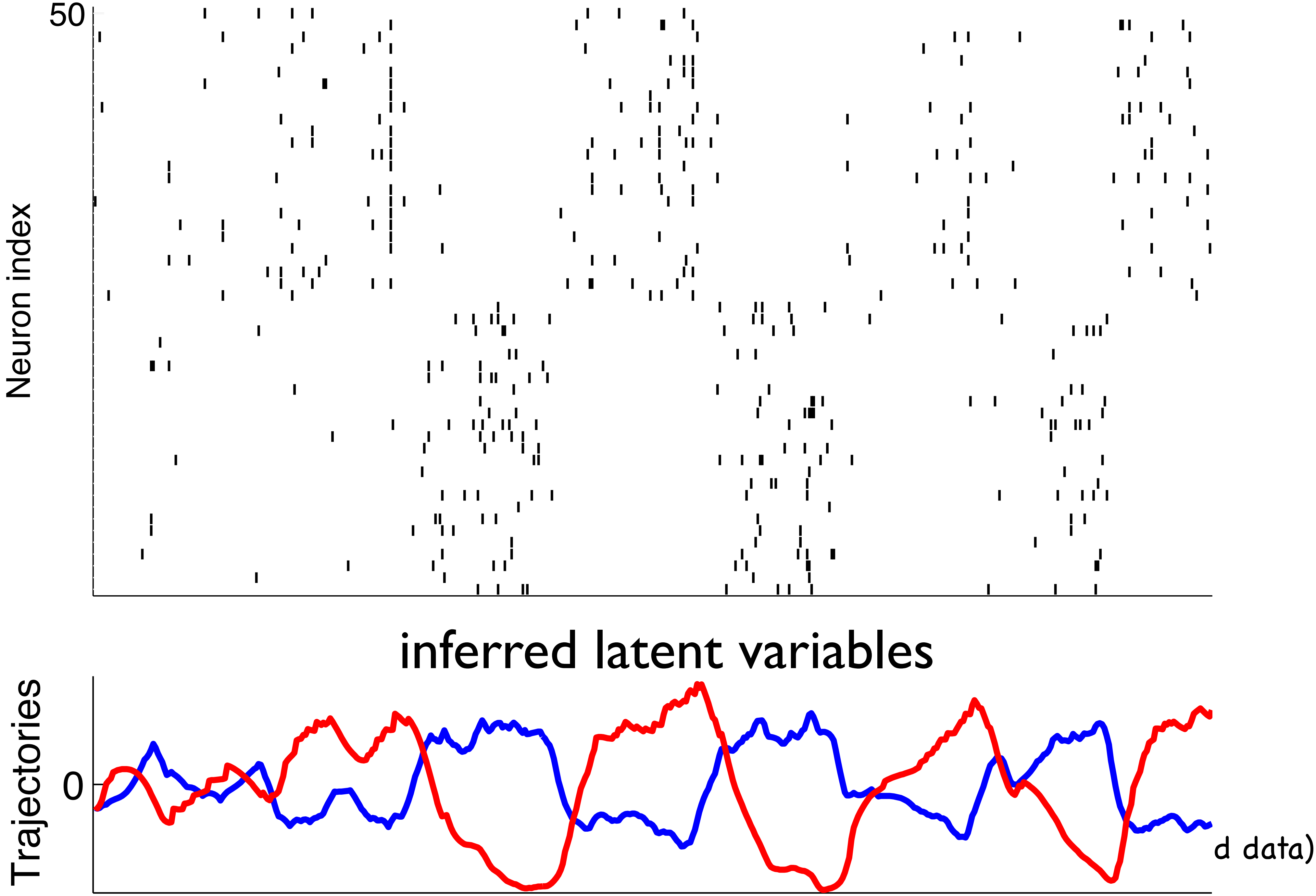
[credit: Jakob Macke]

spike responses



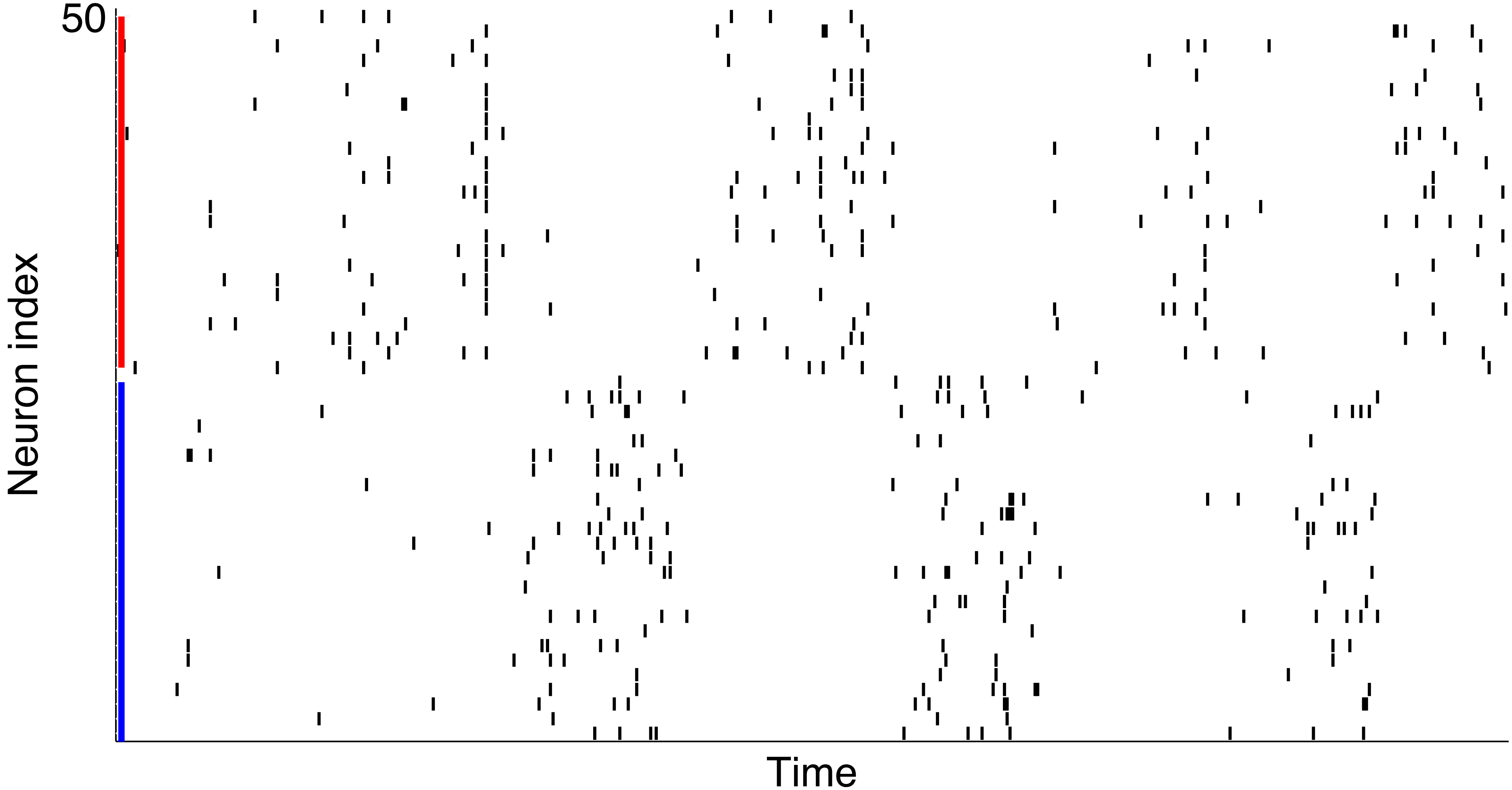
(simulated data)

spike responses

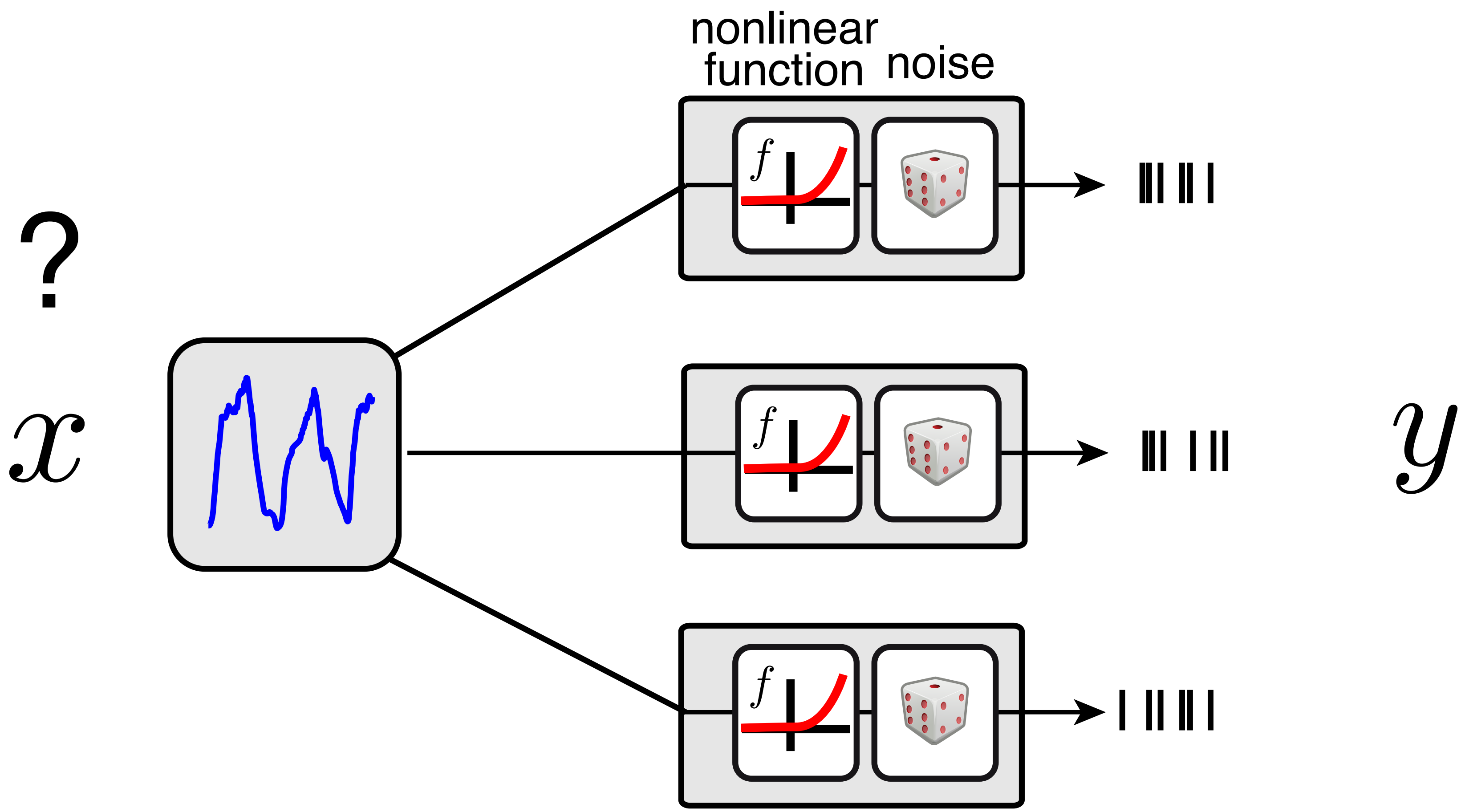


[credit: Jakob Macke]

spike responses



latent variable models = GLMs where we don't know x



goal: find shared structure underlying y

chicken and egg problem

[credit: Jakob Macke]

Why are latent variable models hard to work with?

- hard to compute likelihood!

Why are latent variable models hard to work with?

- hard to compute likelihood!

sensory encoding model

x

$\downarrow$

y

Poisson GLM: fit using:

$$\log P(Y|X) = \sum_t y_t \log \lambda_t - \lambda_t$$

Why are latent variable models hard to work with?

- hard to compute likelihood!

sensory encoding model	**latent variable model**
x	x
$\downarrow$	$\downarrow$
y	y

fit using:

$$\log P(Y|X) = \sum_t y_t \log \lambda_t - \lambda_t$$

fit using: $\log P(Y)$

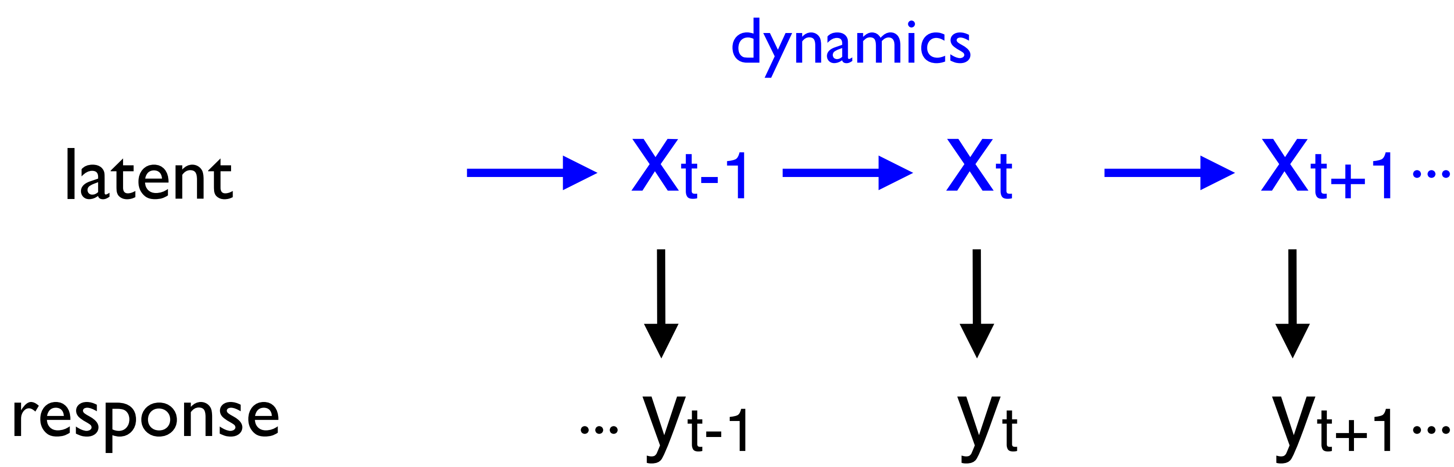
$$= \log \int P(Y|X)P(X)dx$$

requires an integral!

Why are latent variable models hard to work with?

- hard to compute likelihood!

latent
dynamical model



fit using: $\log P(Y)$

$$\int \prod_{t=1}^{T} \Big(\overbrace{p(y_t|x_t)}^{\text{observations}} \overbrace{p(x_t|x_{t-1})}^{\text{dynamics}} \Big) dx_1 dx_2 \cdots dx_T$$

high-dimensional integral

Fitting Latent Variable Models

1. Sampling ("MCMC") - fully Bayesian inference

- procedure for sampling joint distribution: $P\big(\theta, \{\mathbf{x}\} \;\big|\; \{\mathbf{r}\}, \{c\}\big)$

  1) sample latents: $\mathbf{x}^{(i)} \sim p(\mathbf{x}|\mathbf{r}, c, \theta^{(i)})$ conditional over latents

  2) sample parameters: $\theta^{(i+1)} \sim p(\theta|\mathbf{r}, c, \mathbf{x}^{(i)})$ conditional over parameters

Fitting Latent Variable Models

1. Sampling ("MCMC") - fully Bayesian inference

- procedure for sampling joint distribution: $P\big(\theta, \{\mathbf{x}\} \;\big|\; \{\mathbf{r}\}, \{c\}\big)$

  1) sample latents: $\mathbf{x}^{(i)} \sim p(\mathbf{x}|\mathbf{r}, c, \theta^{(i)})$ conditional over latents

  2) sample parameters: $\theta^{(i+1)} \sim p(\theta|\mathbf{r}, c, \mathbf{x}^{(i)})$ conditional over parameters

2. Expectation maximization (EM)

Alternate updating parameters and posterior over latents.

Fitting Latent Variable Models

1. Sampling ("MCMC") - fully Bayesian inference

- procedure for sampling joint distribution: $P\big(\theta, \{\mathbf{x}\} \,\big|\, \{\mathbf{r}\}, \{c\}\big)$

1) sample latents: $\mathbf{x}^{(i)} \sim p(\mathbf{x}|\mathbf{r}, c, \theta^{(i)})$ conditional over latents

2) sample parameters: $\theta^{(i+1)} \sim p(\theta|\mathbf{r}, c, \mathbf{x}^{(i)})$ conditional over parameters

2. Expectation maximization (EM)

Alternate updating parameters and posterior over latents.

3. Variational inference

Optimize a lower bound on posterior over parameters
Easy with modern probabilistic programming languages
(STAN, Edward)

Latent Variable models are defined by two quantities:

latent $P(x)$	**mapping** $P(y\|x)$	$\longrightarrow$ Model
• discrete	• Gaussian	• Mixture of Gaussians ("clustering")
• Gaussian	• linear, Gaussian	• Factor analysis (PCA is special case)
• linear Gaussian dynamics	• linear, Gaussian	• Linear Dynamical Systems (LDS) ("Kalman filter")
• discrete transitions	• any	• Hidden Markov Model ("HMM")

variational latent Gaussian process (vLGP)

[Zhao & Park 2016]

latent

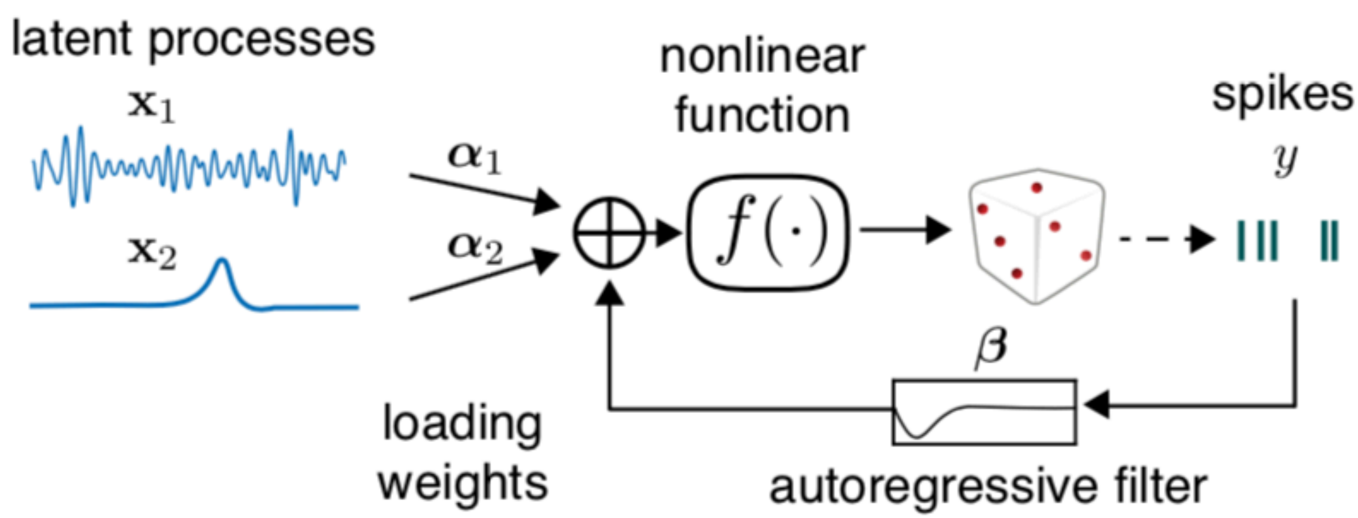
$$P(x)$$

mapping

$$P(y|x)$$

Gaussian Process

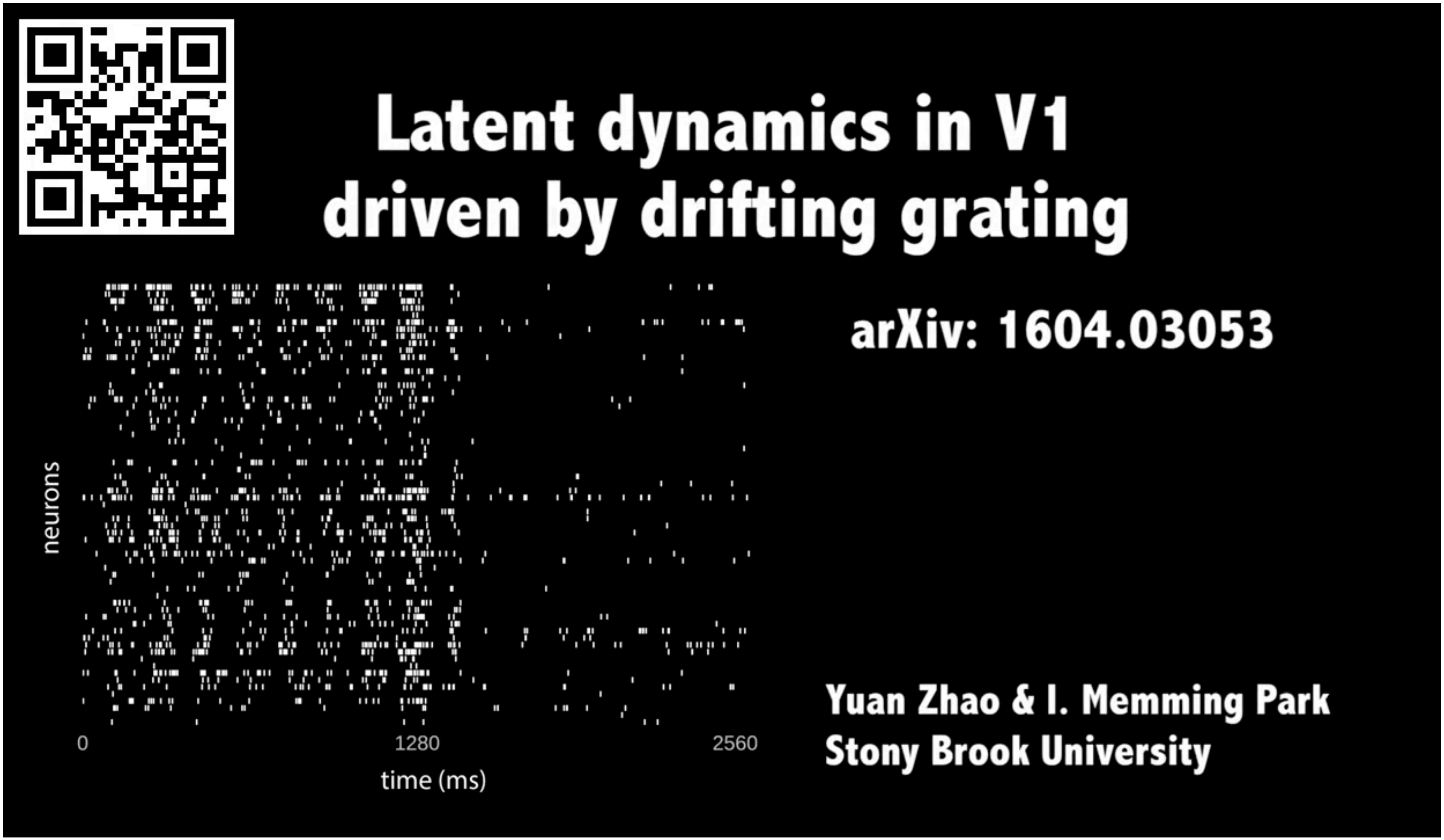
Poisson GLM



[Zhao & Park 2016]

variational latent Gaussian process (vLGP)

- 63 simultaneously-recorded V1 neurons [Graf et al 2011]
- stimuli: drifting sinusoidal gratings

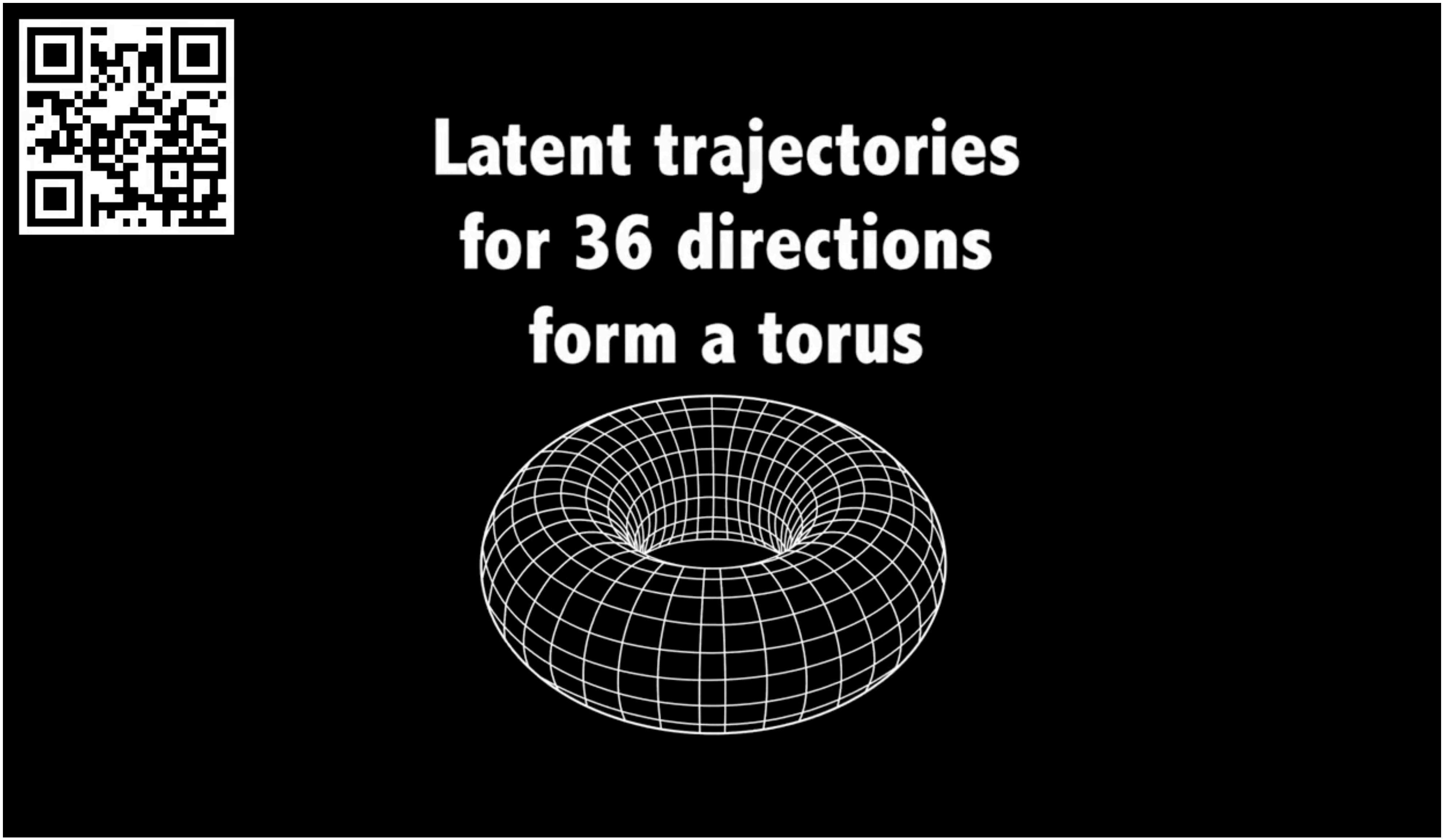


https://www.youtube.com/watch?v=CrY5AfNH1ik

[Zhao & Park 2016]

variational latent Gaussian process (vLGP)

- 63 simultaneously-recorded V1 neurons [Graf et al 2011]
- stimuli: drifting sinusoidal gratings



https://www.youtube.com/watch?v=CrY5AfNH1ik

Summary

- descriptive statistical "encoding" models
- seek to capture structure in data
- formal tools for comparing models
- encoding and decoding analyses via Bayes rule
- models are modular, easy to build /extend / generalize

Big Picture

- large-scale recording technology advancing rapidly
- lots of interesting structure in high-D neural data
- big opportunities in computational / statistical for developing new methods and models to find / exploit this structure!