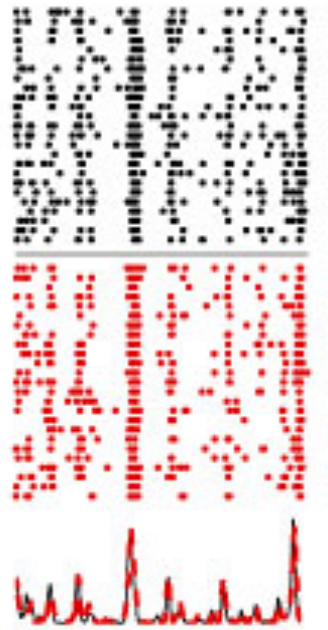
Statistical models for neural coding: GLMs & beyond



Jonathan Pillow

Princeton Neuroscience Institute, Psychology,
Center for Statistics & Machine Learning
Princeton University

CSHL summer course 2024

scientific influences

Rich Zemel

University of Arizona

(Now Columbia)

Eero Simoncelli

NYU

Peter Dayan

Peter Latham

Gatsby Computational Neuroscience Unit

scientific influences

Alex Huk

UT Austin

Mala Murthy

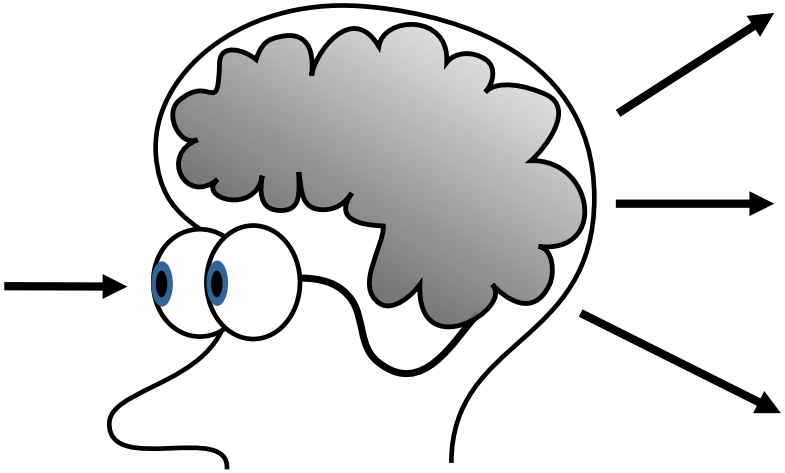
Ilana Witten

Princeton Univeristy

neural coding problem

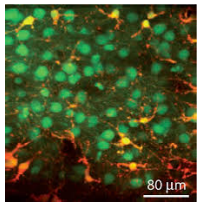
stimulus

x



membrane potential

spikes



imaging

$P(y|x)$

"codebook"

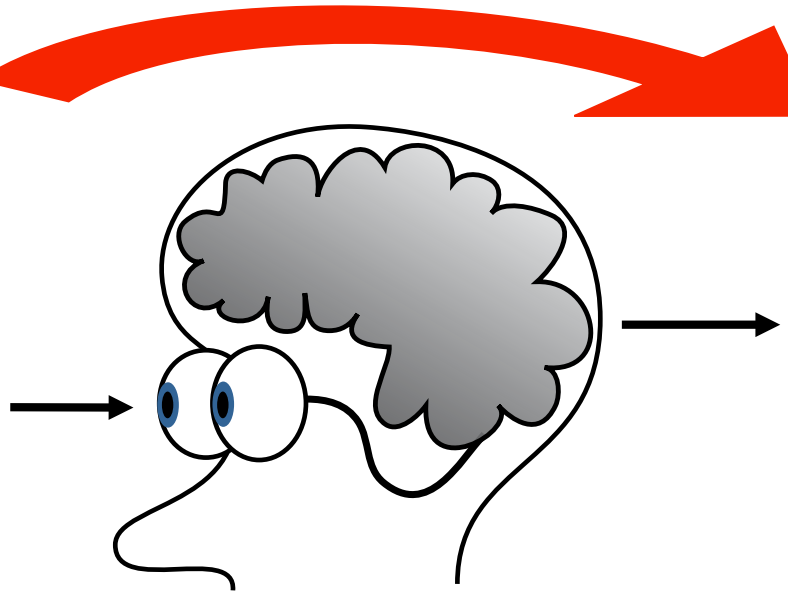
neural activity

y

Q: What is the probabilistic relationship between stimuli and neural activity?

encoding

x



?

$P(y|x)$

"codebook"

neural activity

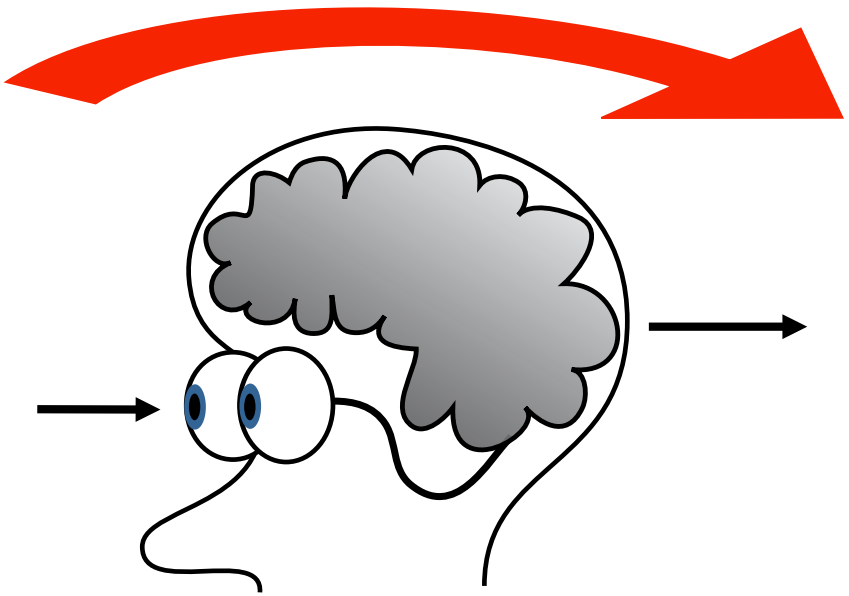
y

uses:

- insight into sensory processing
- sensory prosthetics

encoding

x



?

$P(y|x)$

"codebook"

neural activity

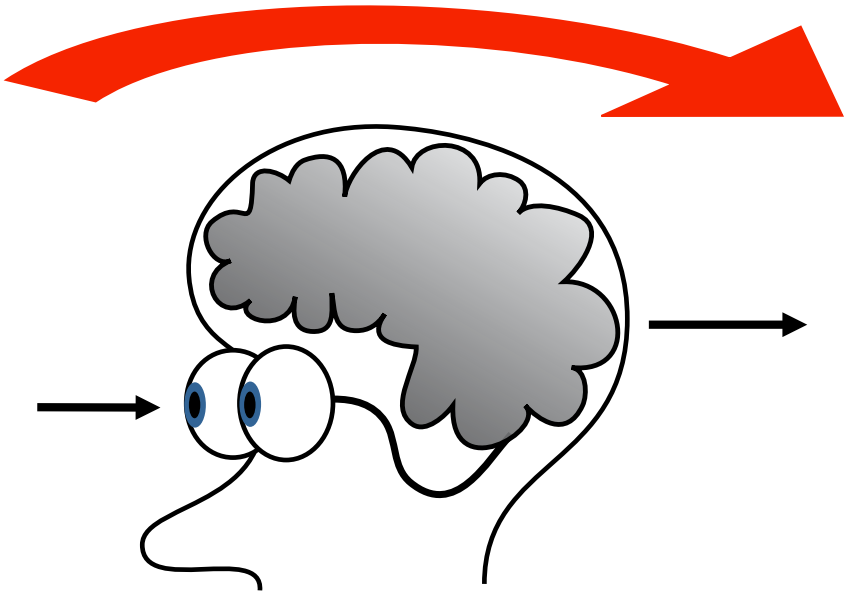
y

uses:

- insight into sensory processing
- sensory prosthetics

encoding

x



?

$P(y|x)$

"codebook"

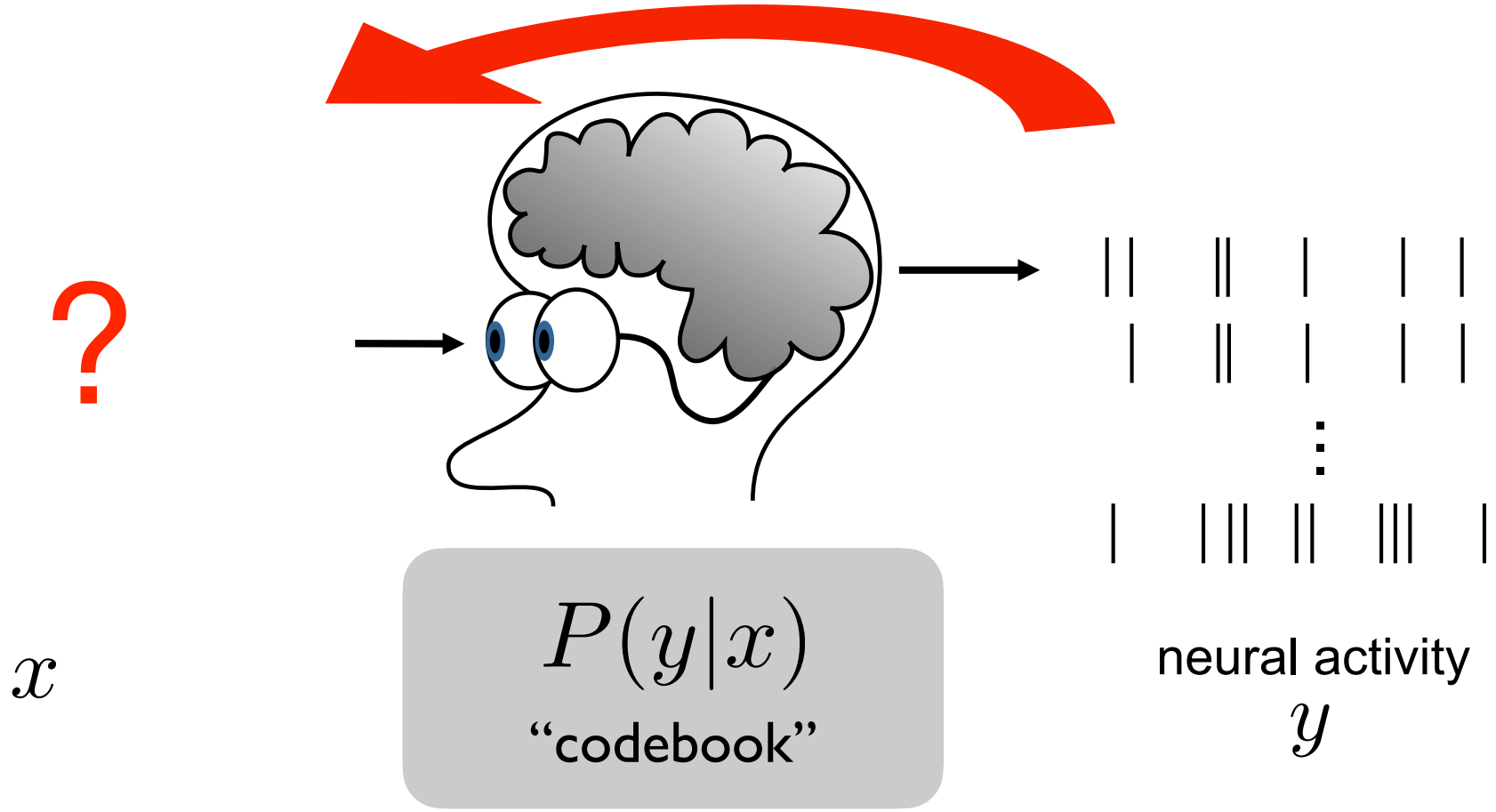
neural activity

y

uses:

- insight into sensory processing
- sensory prosthetics

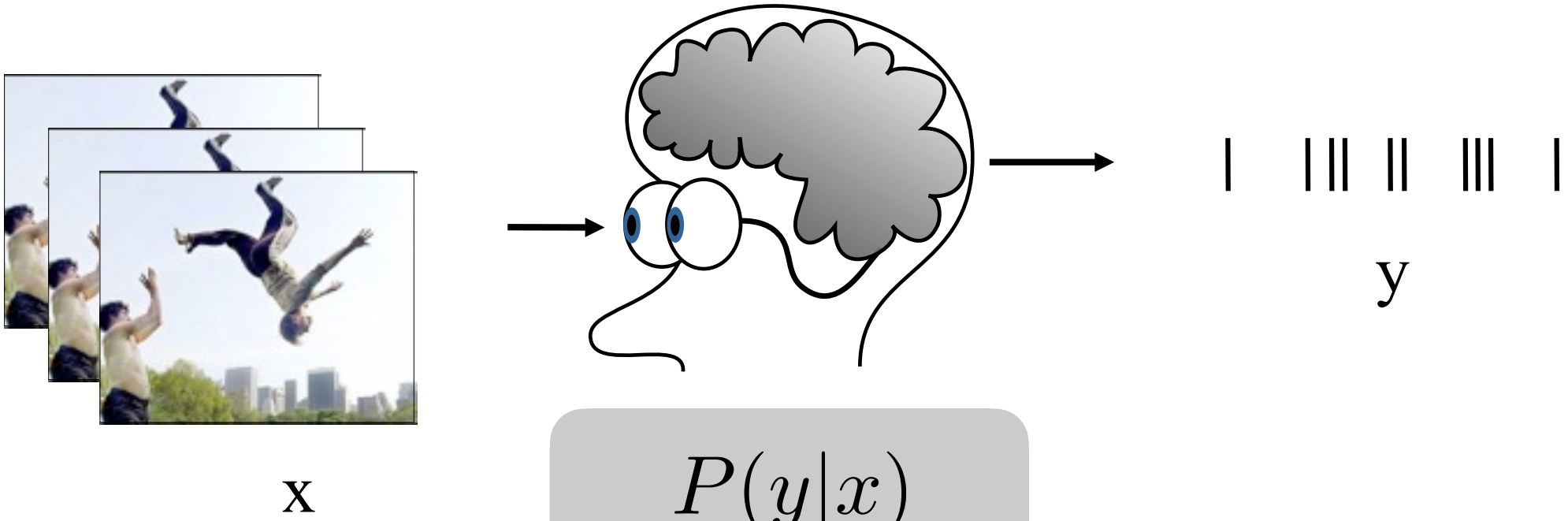
decoding



Bayes' Rule: $P(x|y) \propto P(y|x)P(x)$ (posterior, likelihood, prior)

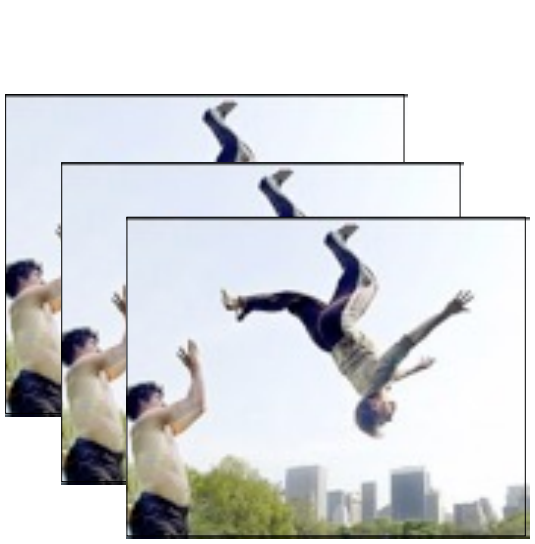
uses:
- constraints on neural readout of pop codes
- motor prosthetics

general name: "regression models"

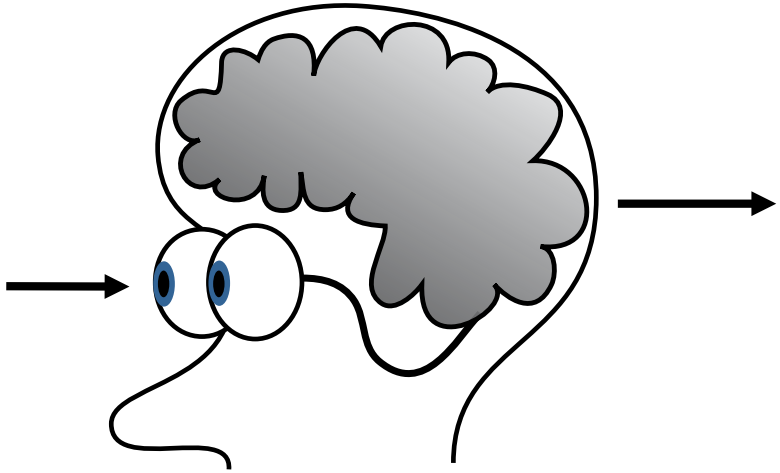


- compare to "latent variable models", which focus only on P(y) (e.g., dimensionality reduction, clustering, state space models)
- Lea to present these tomorrow!

model-based approach



x



| | || || ||| |

y

$P(y|x)$

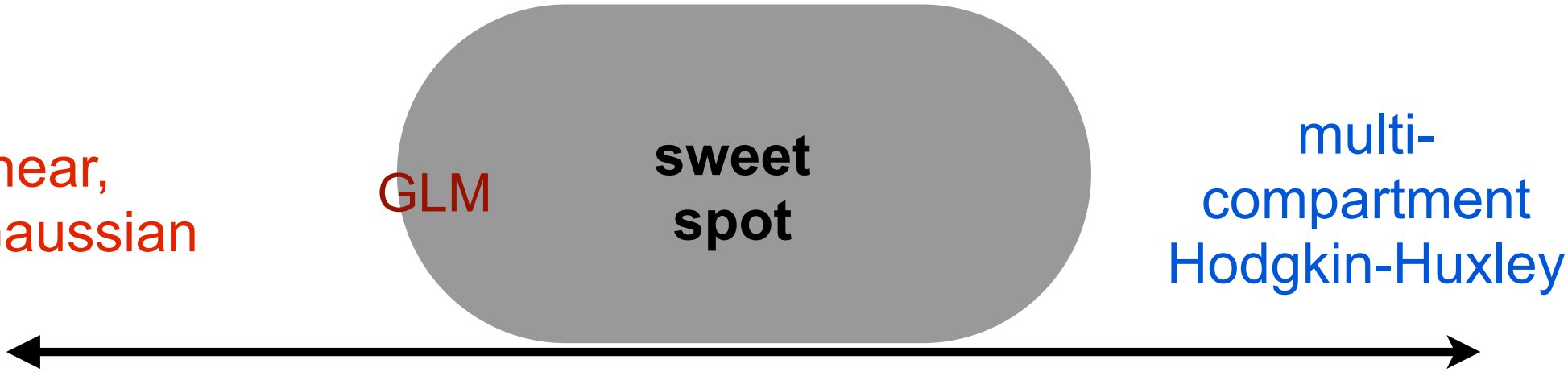
"codebook"

model: $P_\theta(y|x) \approx P(y|x)$

Goal: find model that closely approximates true encoding distribution

Question: what criteria for picking a model?

model desiderata



fittability /
tractability

(can be fit to data)

richness /
flexibility

(capture realistic neural properties)

Outline

1. maximum likelihood for simple models

- spike-count codes

$$P(y|x,\theta)$$

2. Multi-neuron models

- Generalize linear model (GLM)

$$P(y_1, y_2, \ldots, y_n|x,\theta)$$

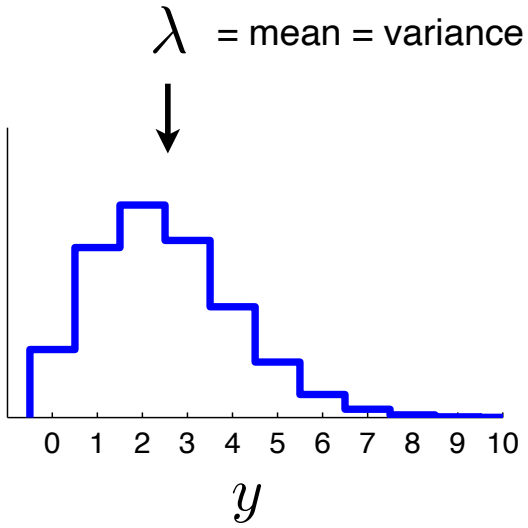
Example 1: linear Poisson neuron

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

spike rate $\lambda = \theta x$

parameter

stimulus

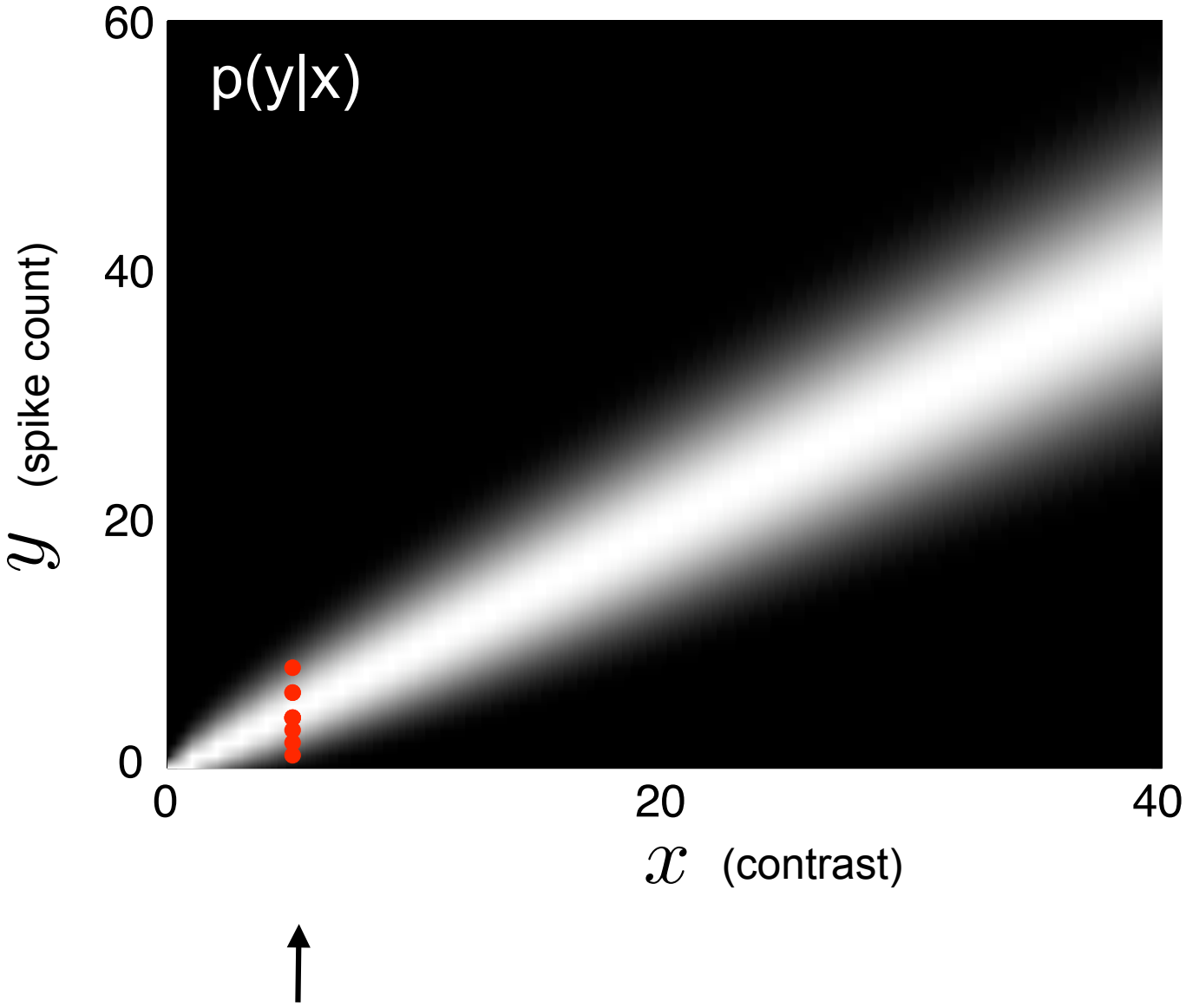


encoding model:

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

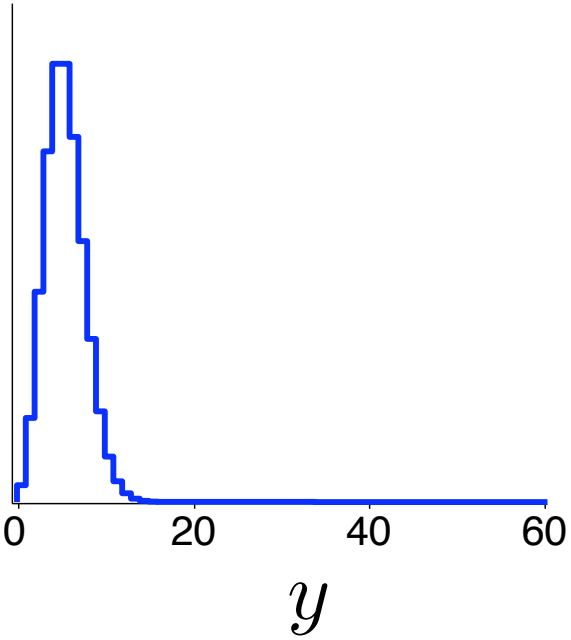
$$= \frac{1}{y!}(\theta x)^y e^{-(\theta x)}$$

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

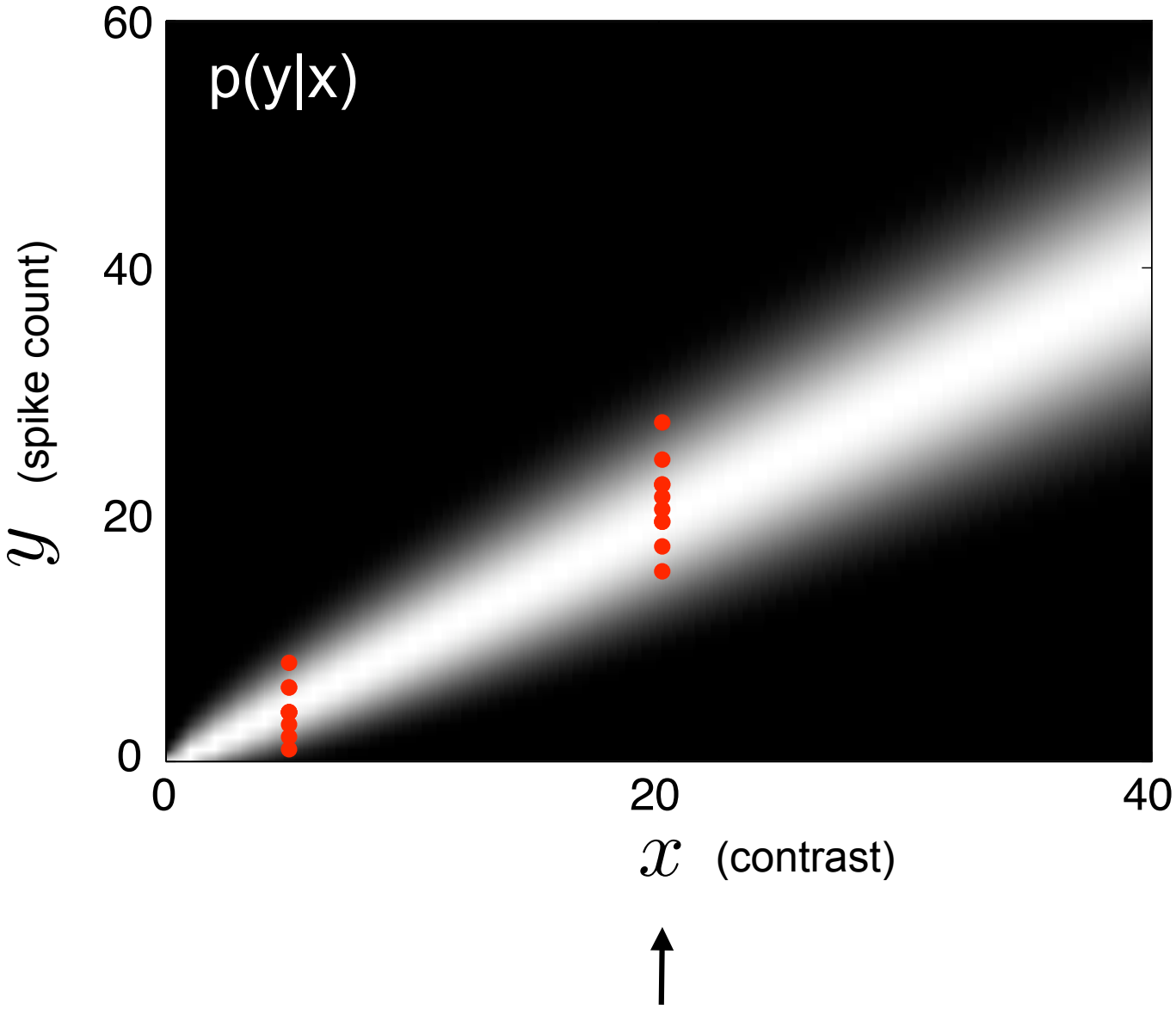


conditional distribution

$$p(y|x = 5)$$

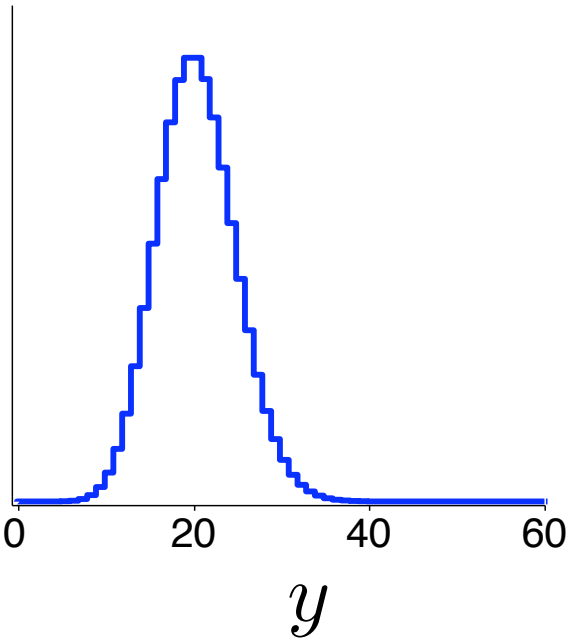


$$\text{mean}(y) = \theta x$$
$$\text{var}(y) = \theta x$$

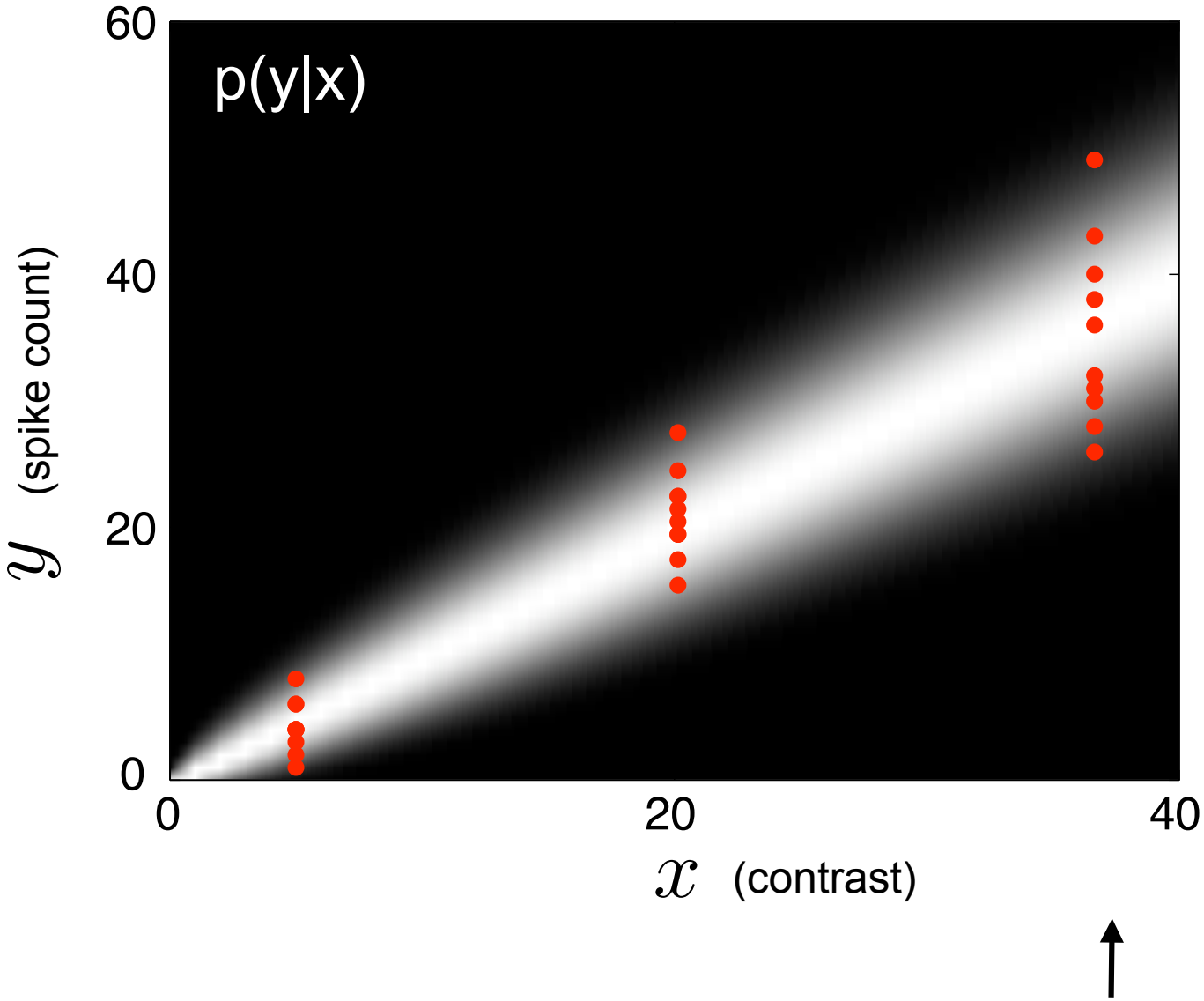


conditional distribution

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

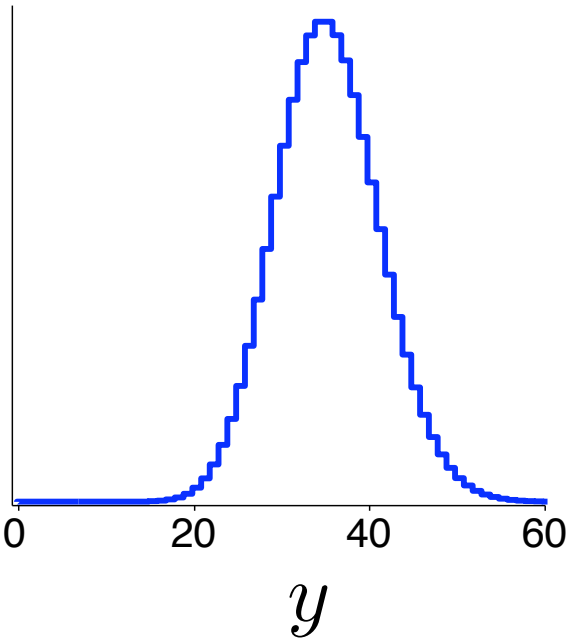


$$\text{mean}(y) = \theta x$$
$$\text{var}(y) = \theta x$$



conditional distribution

$$p(y|x = 35)$$



Maximum Likelihood Estimation:

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

all spike counts

all stimuli

parameters

$$P(Y|X,\theta) = \prod_{i=1}^{N} P(y_i|x_i,\theta)$$

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 $P(Y|X,\theta)$

all spike counts — all stimuli — parameters

$$P(Y|X,\theta) = \prod_{i=1}^{N} P(y_i|x_i,\theta)$$

single-trial probability

Q: what assumption are we making about the responses?

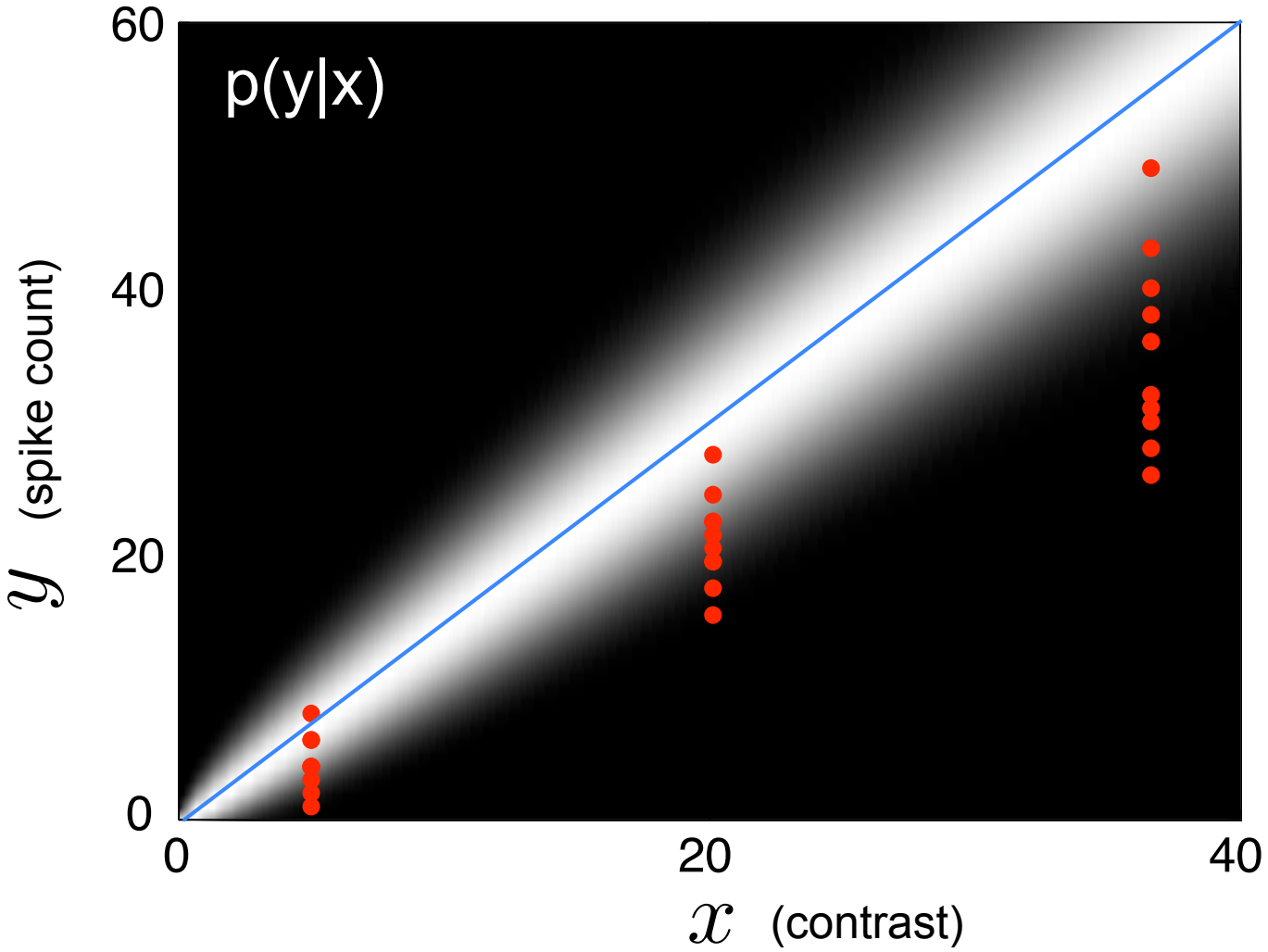
A: conditional independence across trials!

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 $P(Y|X, \theta)$



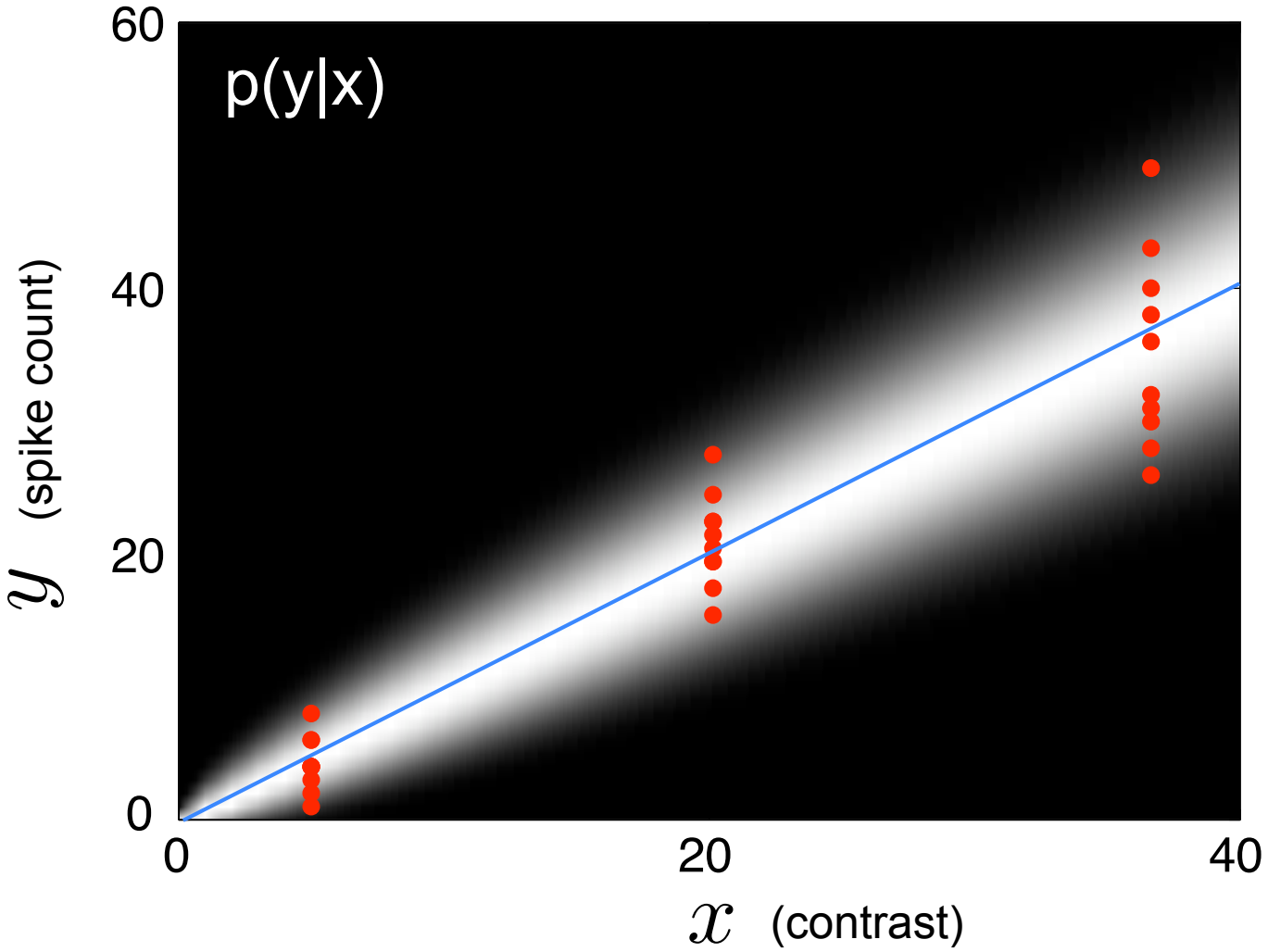
$$y \sim \text{Poiss}(\theta x)$$

$$\theta = 1.5$$

- could in theory do this by turning a knob

Maximum Likelihood Estimation:

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



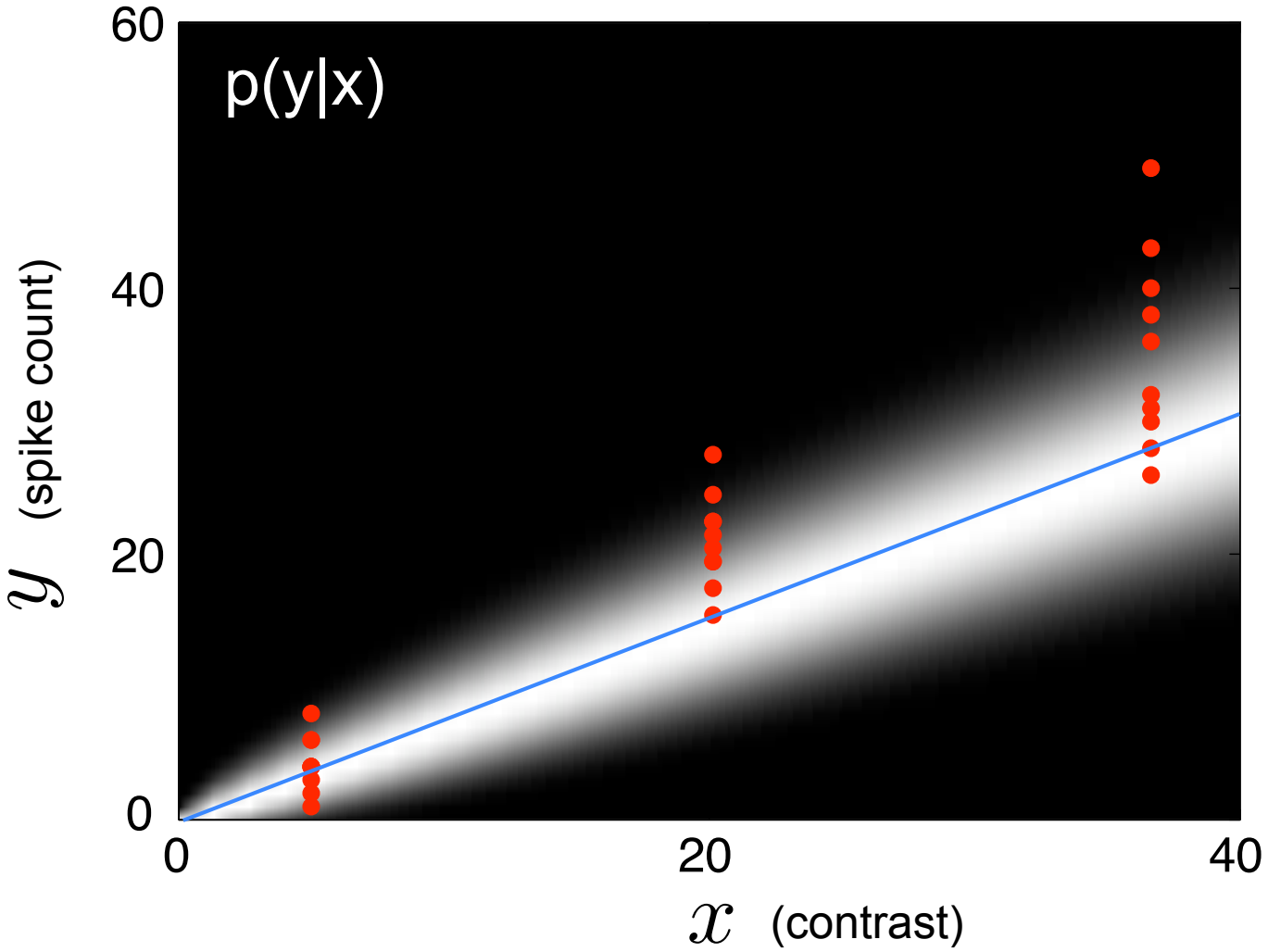
$$y \sim \text{Poiss}(\theta x)$$

$$\theta = 1$$

- could in theory do this by turning a knob

Maximum Likelihood Estimation:

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

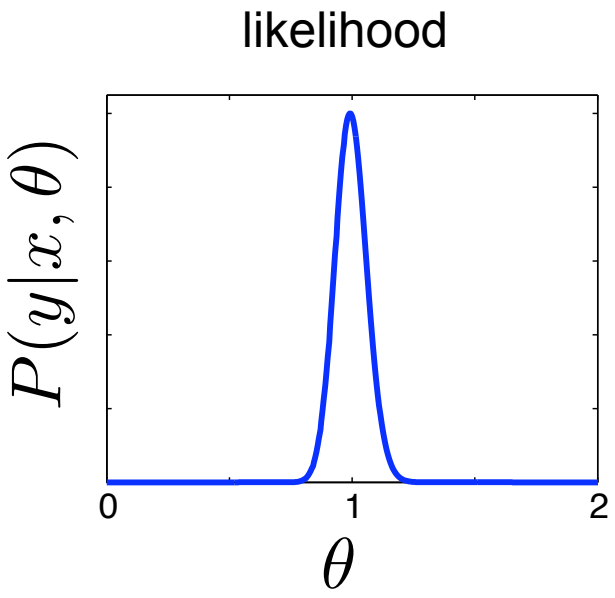


$$y \sim \text{Poiss}(\theta x)$$

$$\theta = 0.5$$

- could in theory do this by turning a knob

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



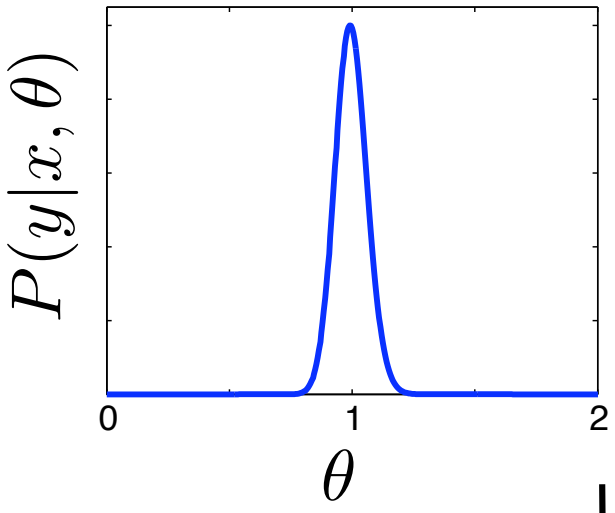
Because data are independent:

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

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

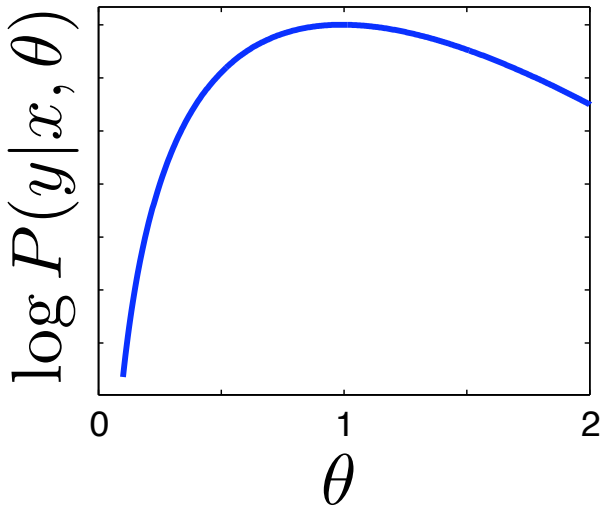
Likelihood function: $P(Y|X,\theta)$ as a function of θ

likelihood



log

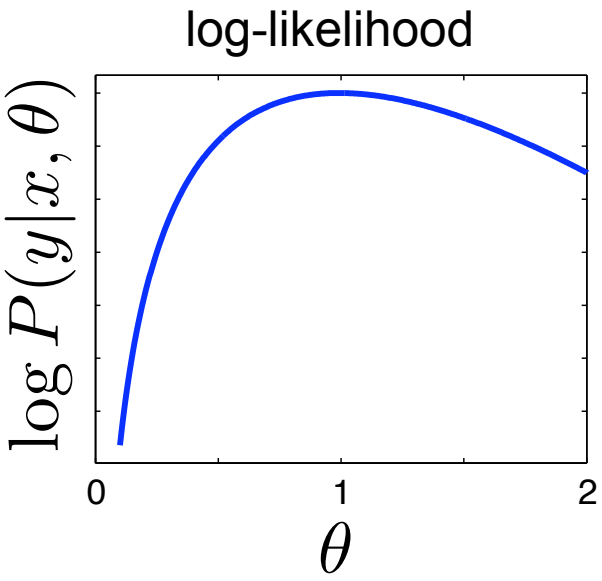
log-likelihood



Because data are independent:

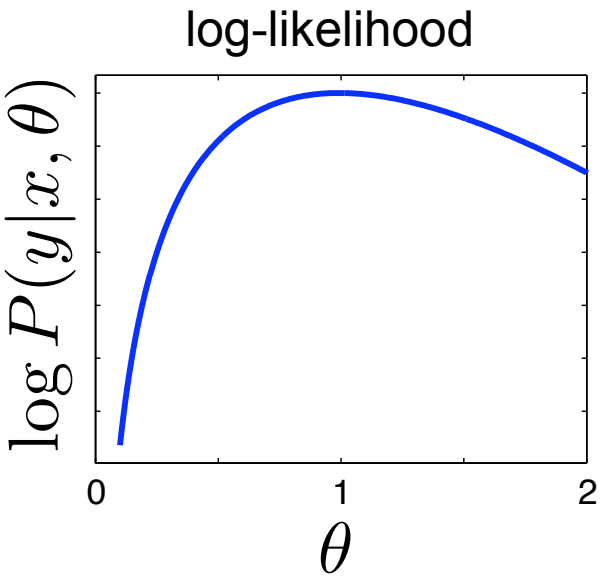
$$P(Y|X,\theta) = \prod_i P(y_i|x_i,\theta)$$
$$= \prod \frac{1}{y_i!}(\theta x_i)^{y_i} e^{-(\theta x_i)}$$

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



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

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"

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

$$\Longrightarrow \quad \hat{\theta}_{ML} = \frac{\sum y_i}{\sum x_i}$$

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)

Example 2: linear Gaussian neuron

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

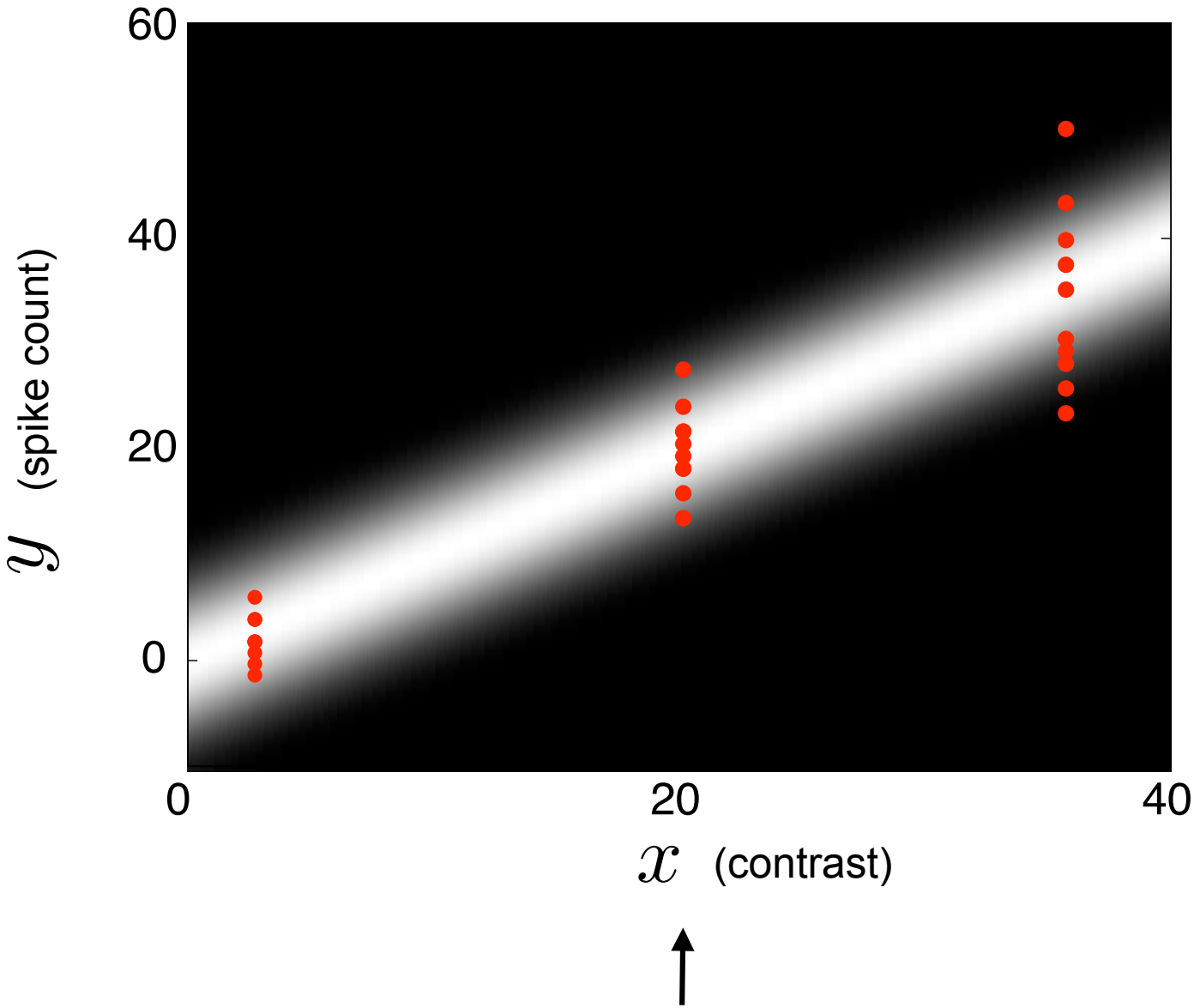
spike rate $\mu = \theta x$

parameter

stimulus

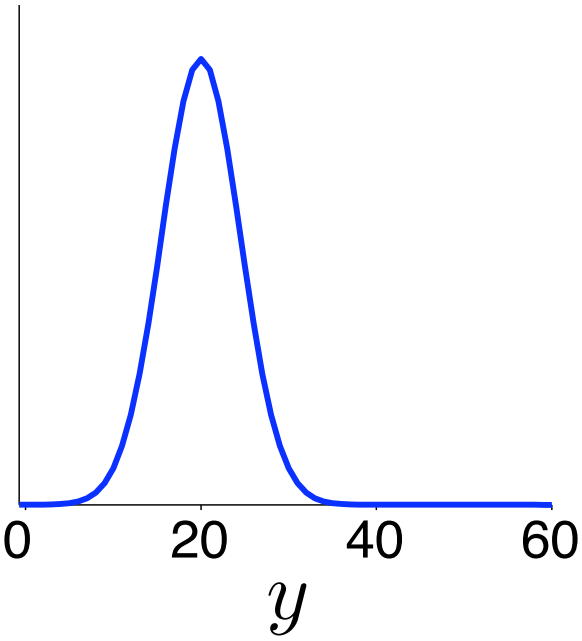
encoding model: $P(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-\theta x)^2}{2\sigma^2}}$

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



encoding distribution

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



All slices have same width

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

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

Do it: differentiate, set to zero, and solve.

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

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

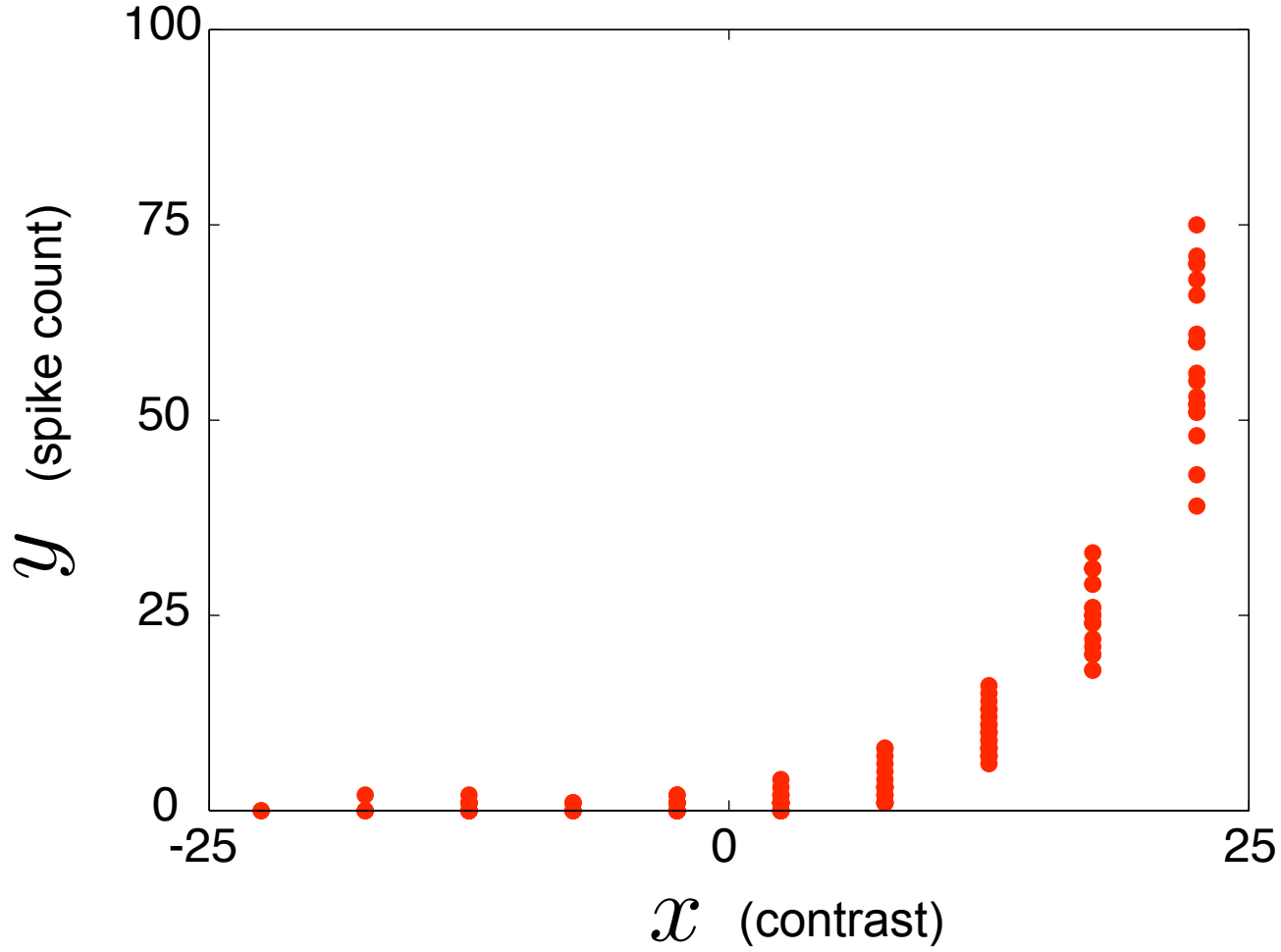
$$\frac{d}{d\theta} \log P(Y|X,\theta) = -\sum \frac{(y_i - \theta x_i)x_i}{\sigma^2} = 0$$

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

("Least squares regression" solution)

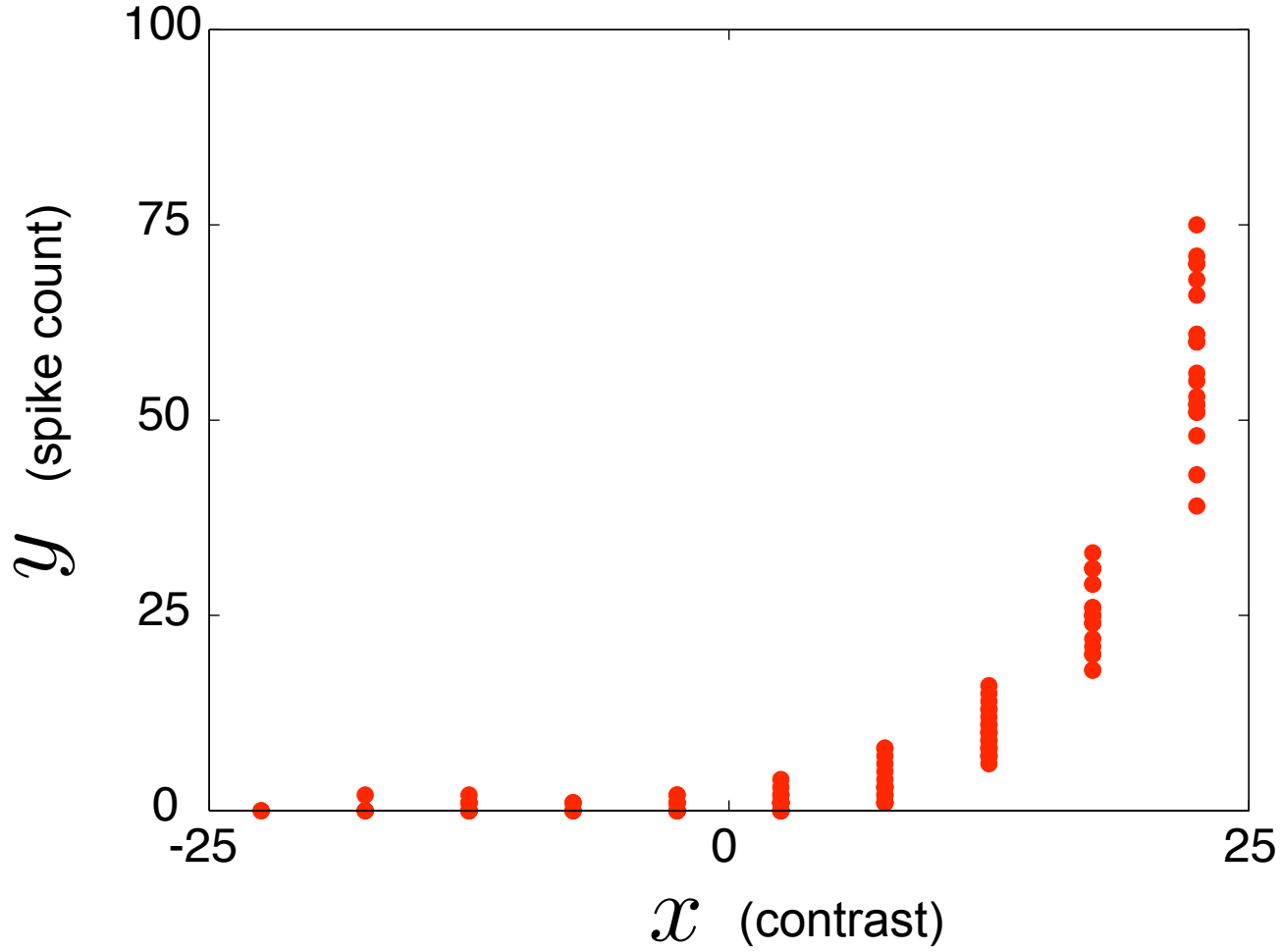
(Recall that for Poisson, $\hat{\theta}_{ML} = \frac{\sum y_i}{\sum x_i}$)

Example 3: unknown neuron



Be the computational neuroscientist: what model would you use?

Example 3: unknown neuron



More general setup: $y \sim Poiss(\lambda)$

$\lambda = f(\theta x)$

for some nonlinear function f

This is a GLM!

GLMs

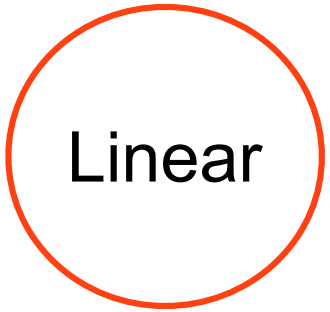
- Be careful about terminology:

GLM ≠ GLM

General Linear Model

Generalized Linear Model

(Nelder 1972)



2003 interview with John Nelder...

Stephen Senn: I must confess to having some confusion when I was a young statistician between general linear models and generalized linear models. Do you regret the terminology?

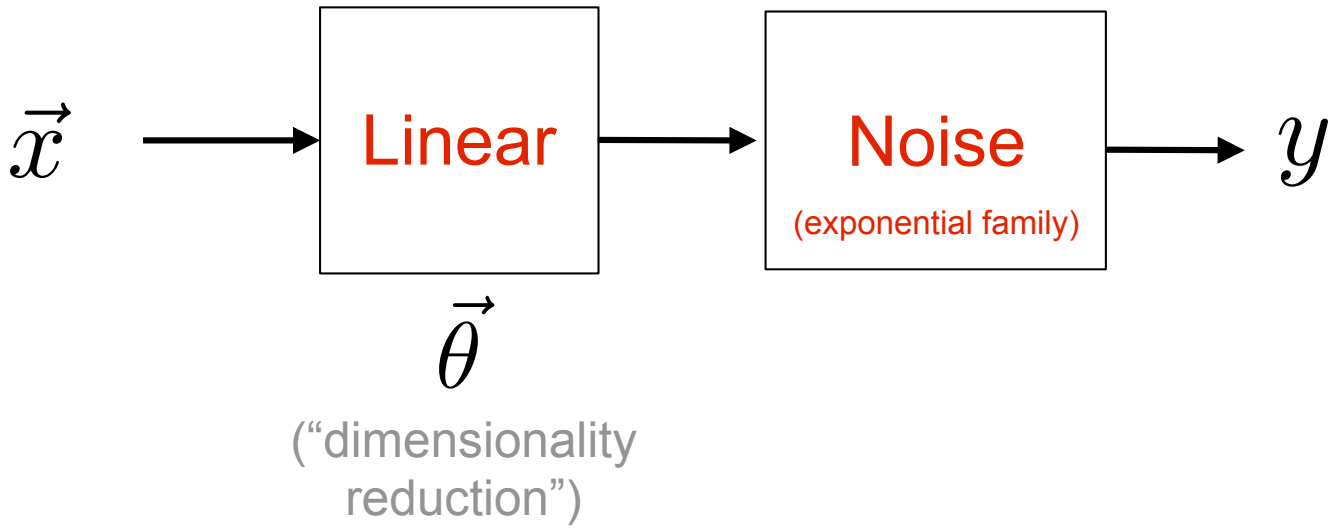
John Nelder: I think probably I do. I suspect we should have found some more fancy name for it that would have stuck and not been confused with the general linear model, although general and generalized are not quite the same. I can see why it might have been better to have thought of something else.

Senn, (2003). *Statistical Science*

Moral:

Be careful when naming your model!

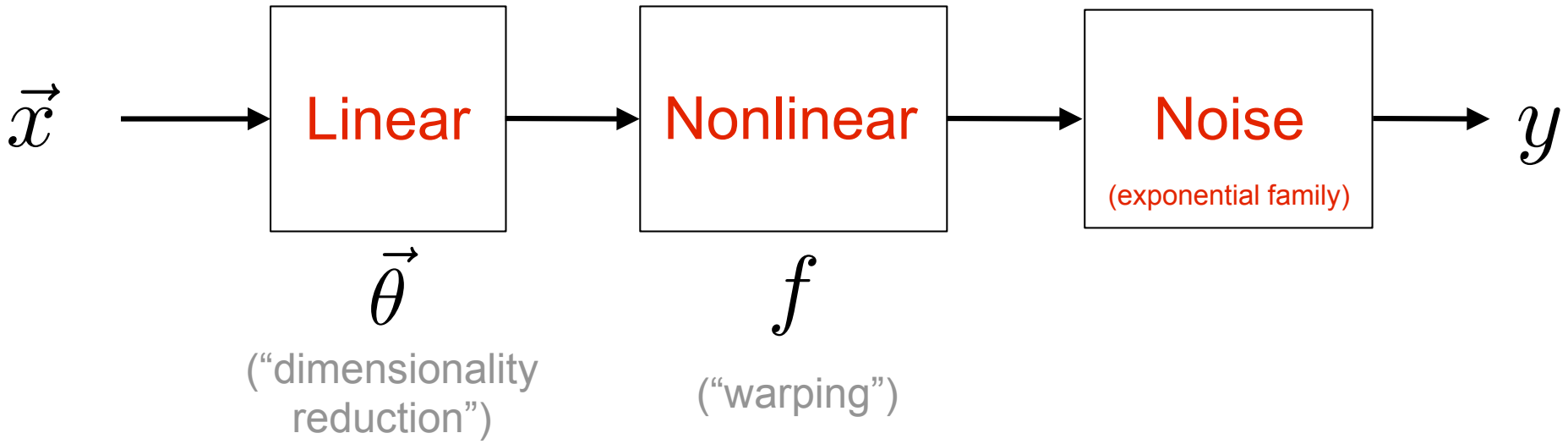
1. General Linear Model (GLM)



Examples:

1. Gaussian $y = \vec{\theta} \cdot \vec{x} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$
2. Poisson $y \sim \mathrm{Poiss}(\vec{\theta} \cdot \vec{x})$

2. Generalized Linear Model



Examples:

1. Gaussian $y = f(\vec{\theta} \cdot \vec{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$
2. Poisson $y \sim \text{Poiss}(f(\vec{\theta} \cdot \vec{x}))$
3. Bernoulli $y \sim \text{Ber}(f(\vec{\theta} \cdot \vec{x}))$

2. Generalized Linear Model

Question: what restrictions (if any) must be placed on the nonlinearity f in each of the GLMs below?

Examples:

1. Gaussian $y = f(\vec{\theta} \cdot \vec{x}) + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$
2. Poisson $y \sim \text{Poiss}(f(\vec{\theta} \cdot \vec{x}))$
3. Bernoulli $y \sim \text{Ber}(f(\vec{\theta} \cdot \vec{x}))$

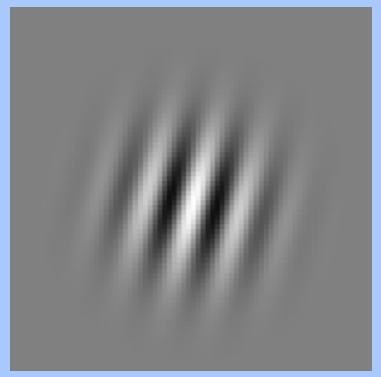
2. Generalized Linear Model

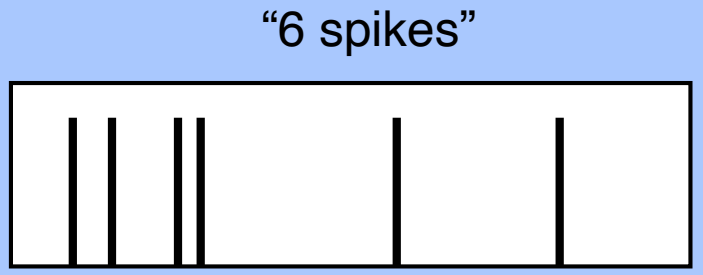
Question: what restrictions (if any) must be placed on the nonlinearity f in each of the GLMs below?

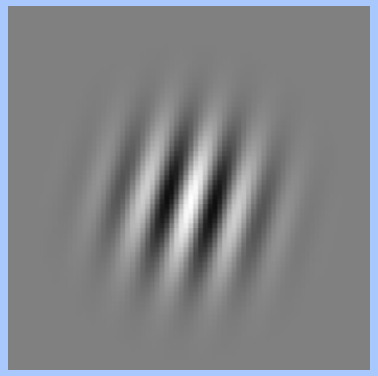
$\mathcal{N}(0, \sigma^2)$

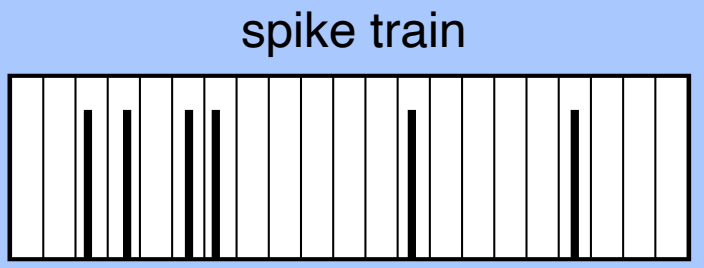
none	1. Gaussian	$y = f(\vec{\theta} \cdot \vec{x}) + \epsilon$
non-negative	2. Poisson	$y \sim \text{Poiss}(f(\vec{\theta} \cdot \vec{x}))$
bounded between 0 and 1	3. Bernoulli	$y \sim \text{Ber}(f(\vec{\theta} \cdot \vec{x}))$

Part II: from spike counts to spike trains!









response at time t

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

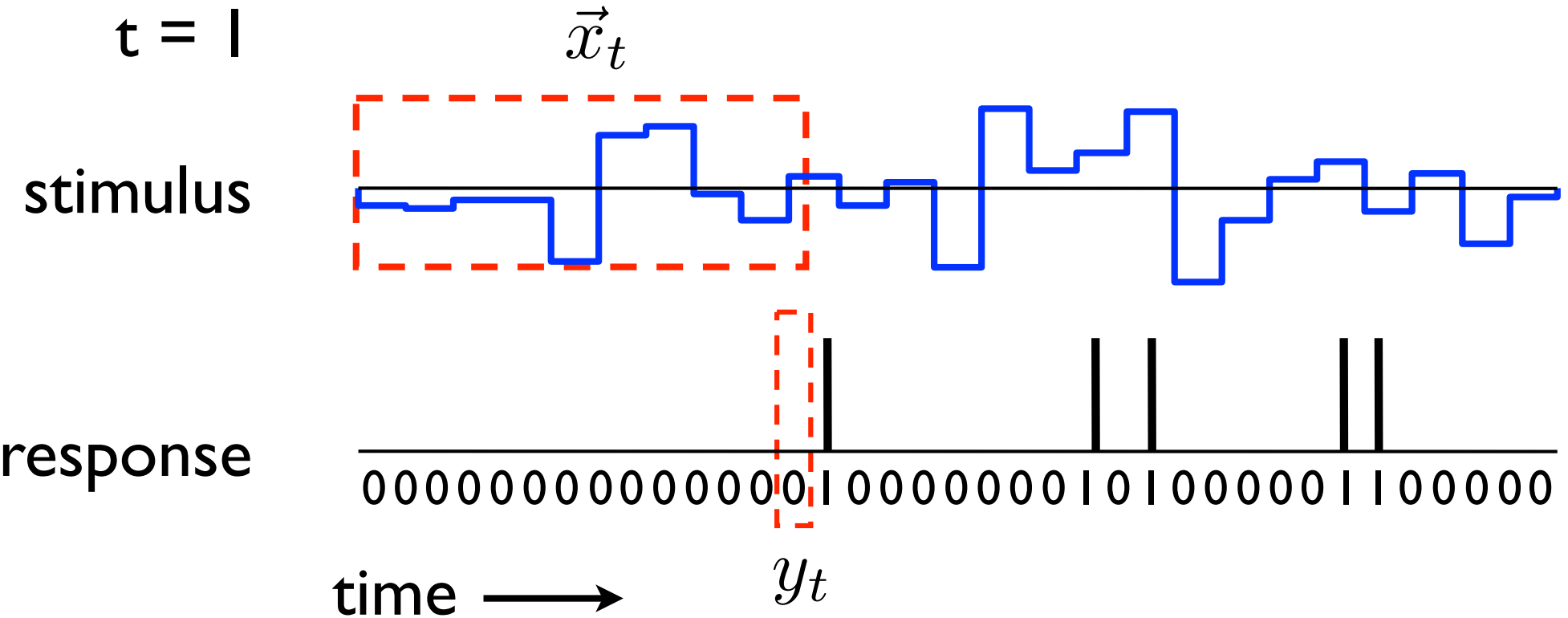
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 1



response at time t

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

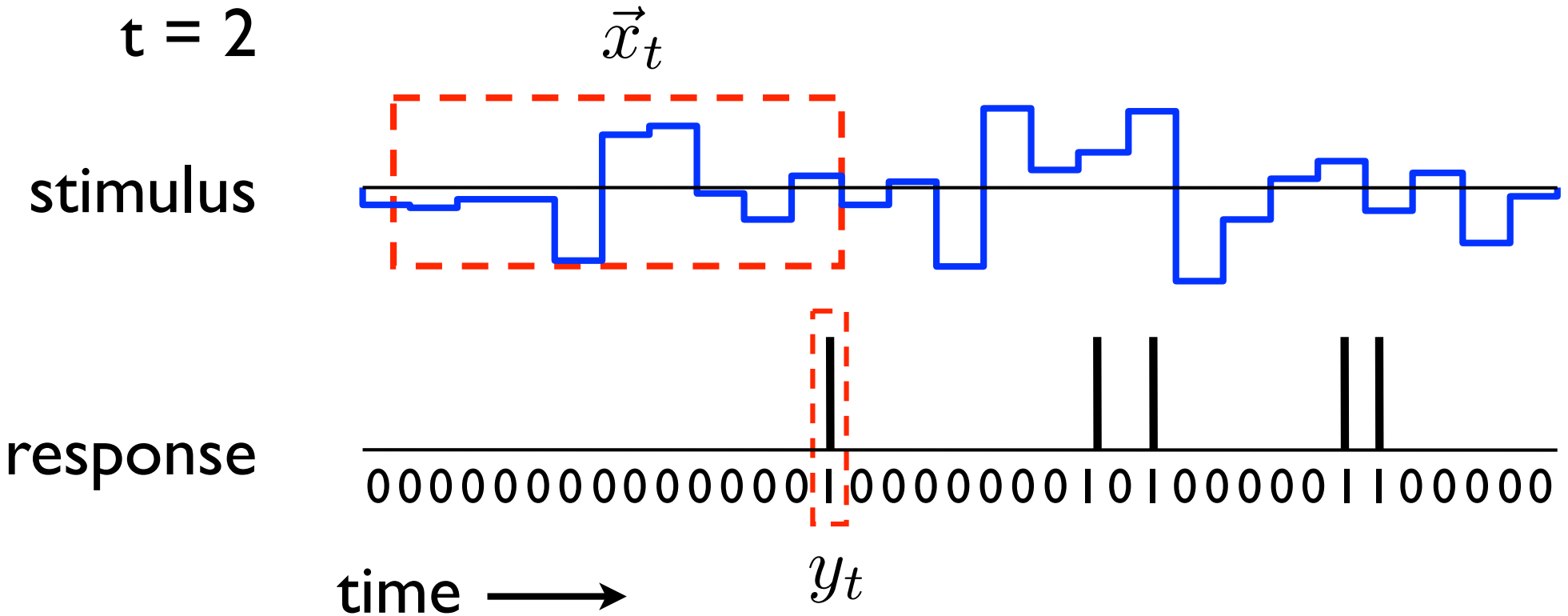
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 2



response at time t

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

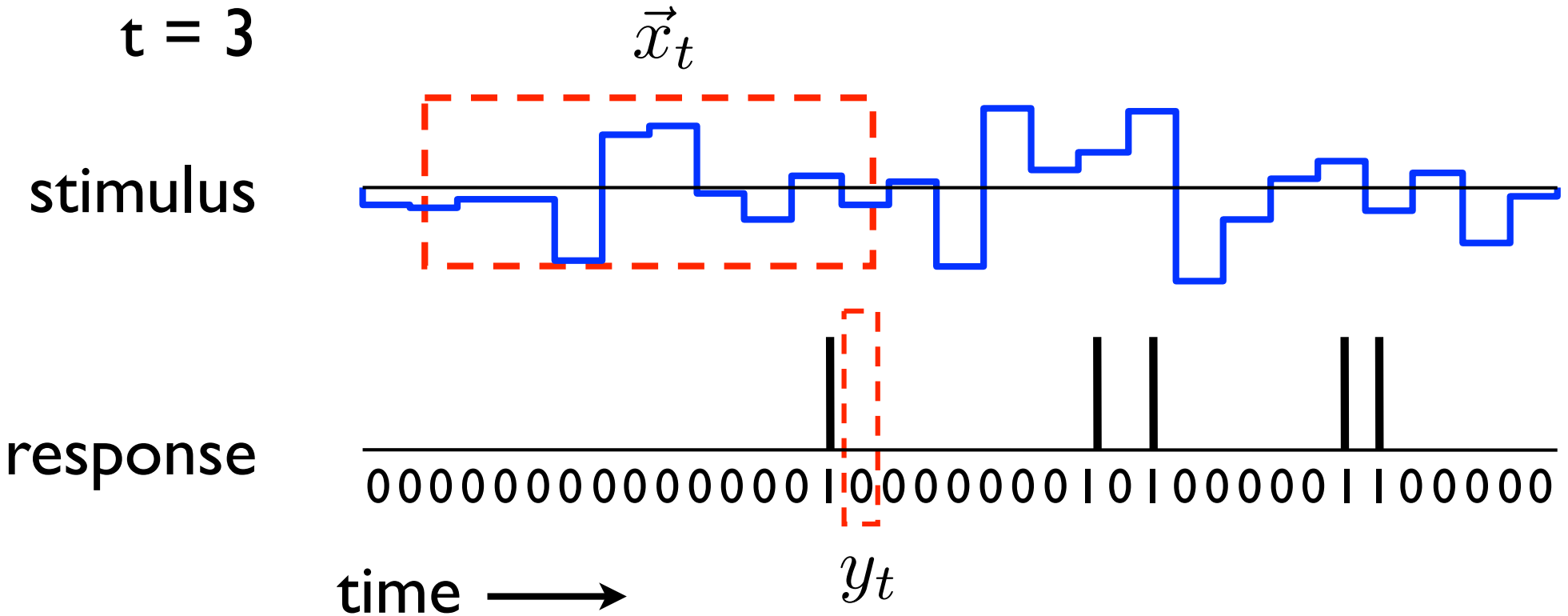
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 3



response at time t

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

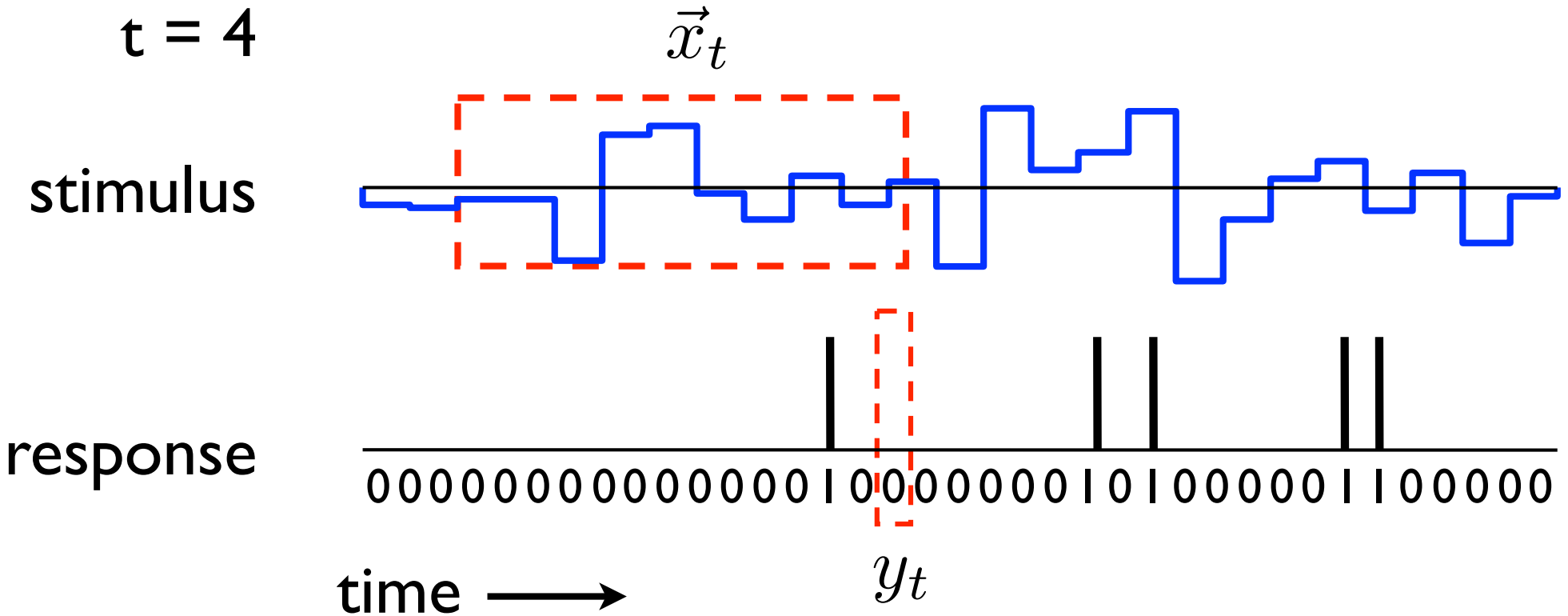
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 4



response at time t

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

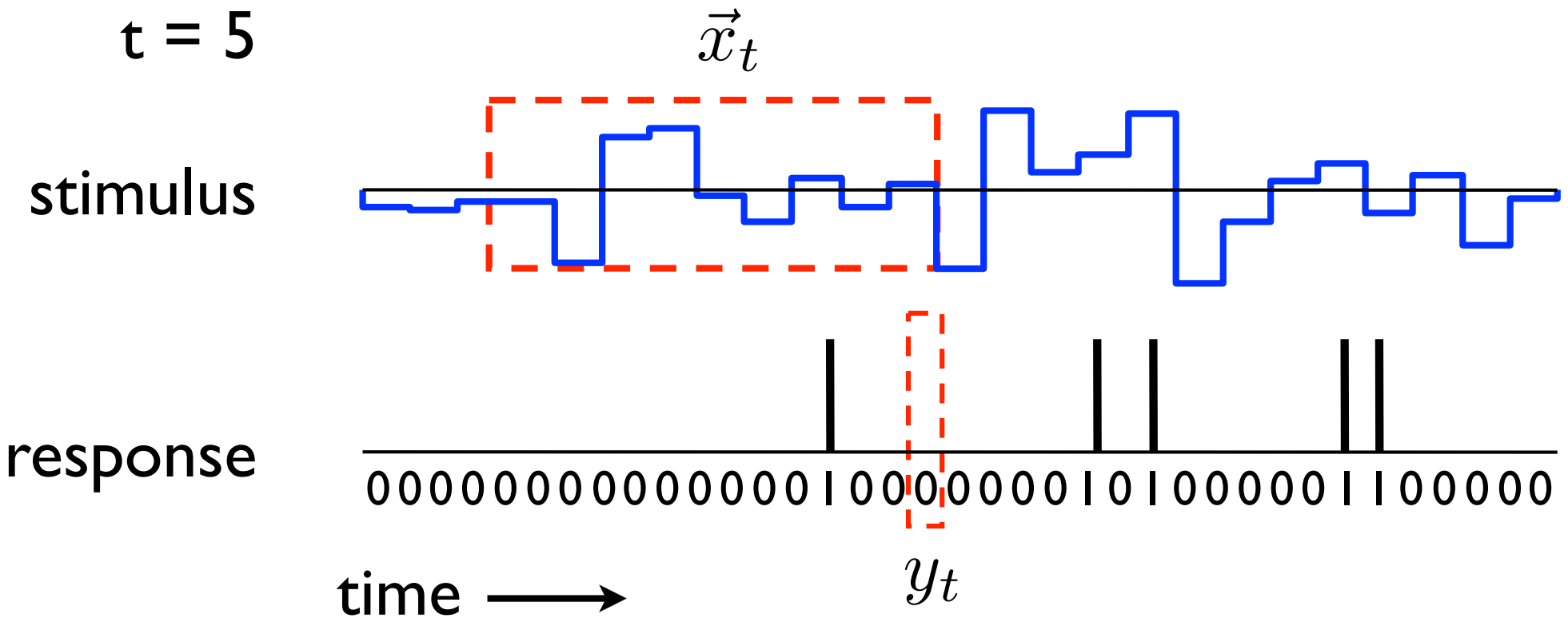
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 5



response at time t

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

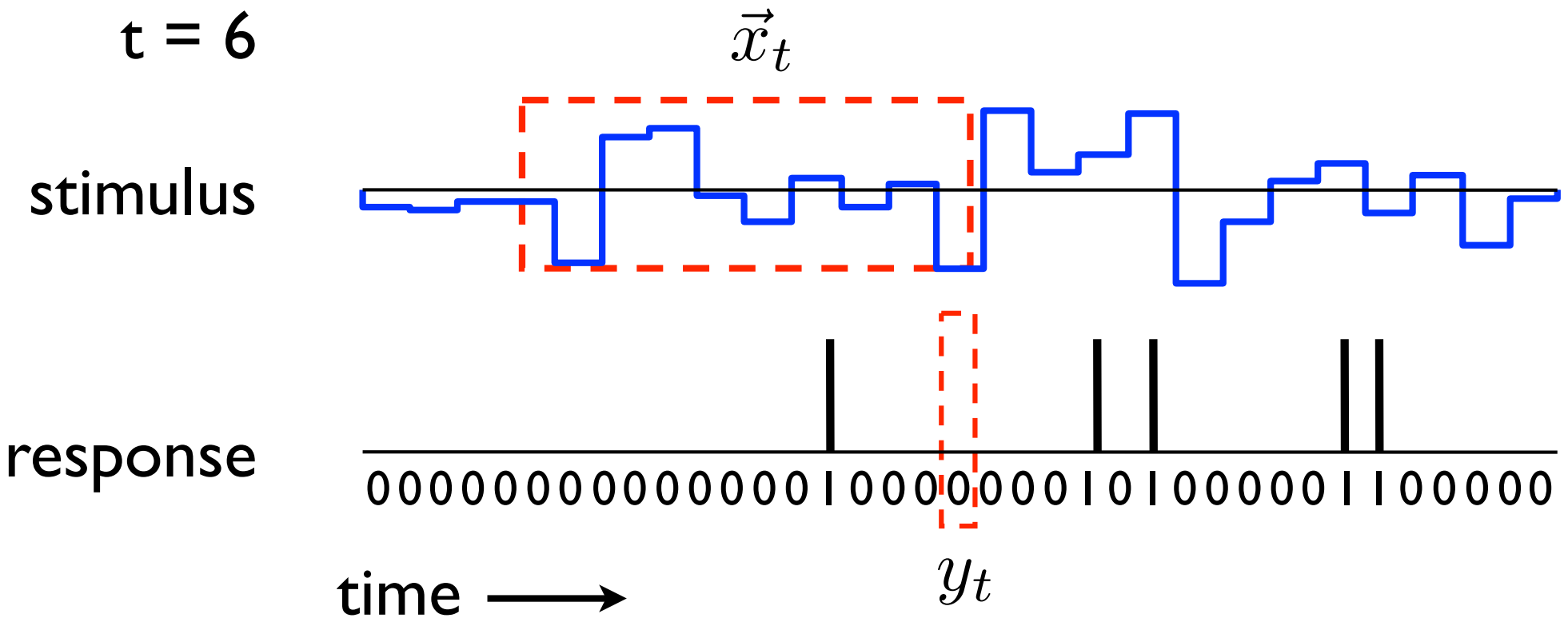
$N(0, \sigma^2)$

linear filter

vector stimulus at time t

walk through the data one time bin at a time

t = 6

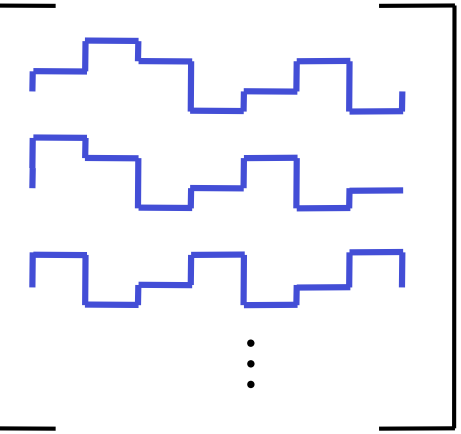


Build up to following matrix version:

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

time ↓

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \\ & \vdots & \end{bmatrix} \begin{bmatrix} \\ \vec{k} \\ \\ \end{bmatrix}$$



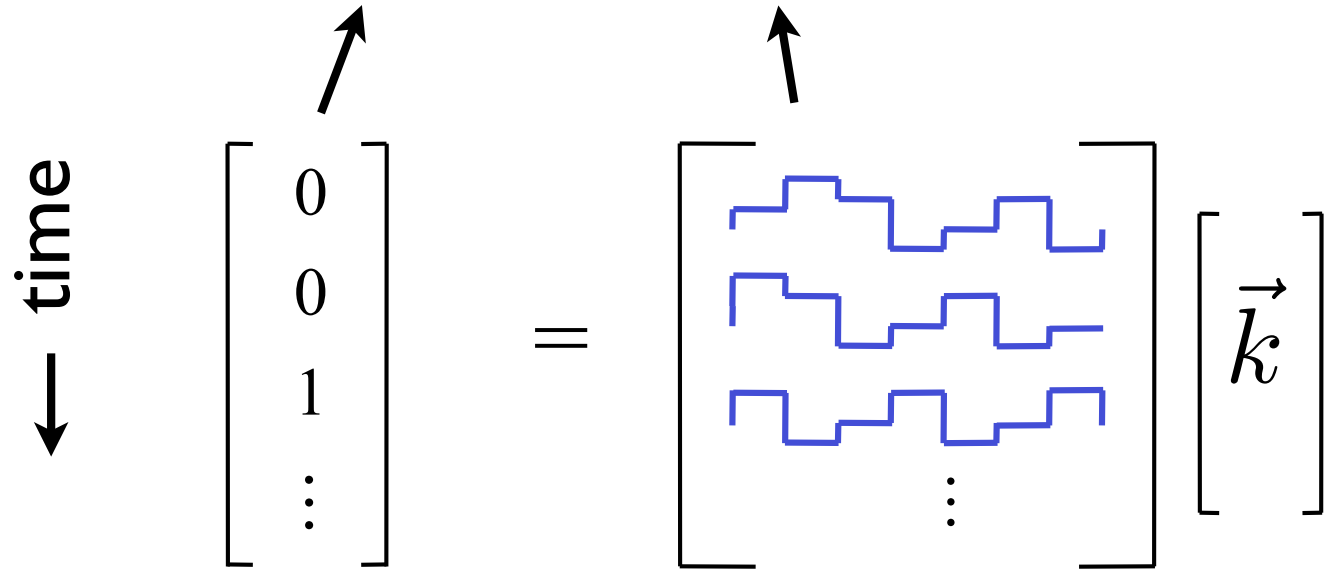
design matrix

"Hankel matrix"
(every row is a shifted copy of the row above)

(see also: Toeplitz matrix)

Build up to following matrix version:

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



"Linear-Gaussian" GLM:

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

(aka "least-squares regression")

stimulus covariance

spike-triggered avg (STA)

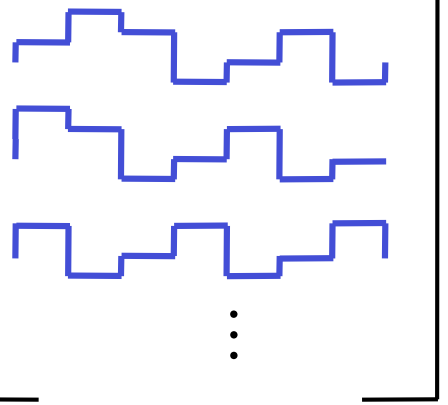
Formally:

$$Y = X\vec{k} + \vec{\epsilon}$$

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

iid Gaussian noise vector

time ↓

$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \end{bmatrix} = \begin{bmatrix} \text{  } \end{bmatrix} \begin{bmatrix} \vec{k} \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \end{bmatrix}$$

equivalent to writing: $Y|X, \vec{k} \sim \mathcal{N}(X\vec{k}, \sigma^2 I)$

or

$$P(Y|X, \vec{k}) = \frac{1}{|2\pi\sigma^2 I|^{\frac{T}{2}}} \exp\left(-\frac{1}{2\sigma^2}(Y - X\vec{k})^\top (Y - X\vec{k}) \right)$$

Take log, differentiate and set to zero.

Formally:

$$Y = X\vec{k} + \vec{\epsilon}$$

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

iid Gaussian noise vector

time ↓

$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \end{bmatrix} = \begin{bmatrix} \text{⎍} \\ \text{⎍} \\ \text{⎍} \\ \vdots \end{bmatrix} \begin{bmatrix} \vec{k} \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \end{bmatrix}$$

equivalent to w

$^2 I)$

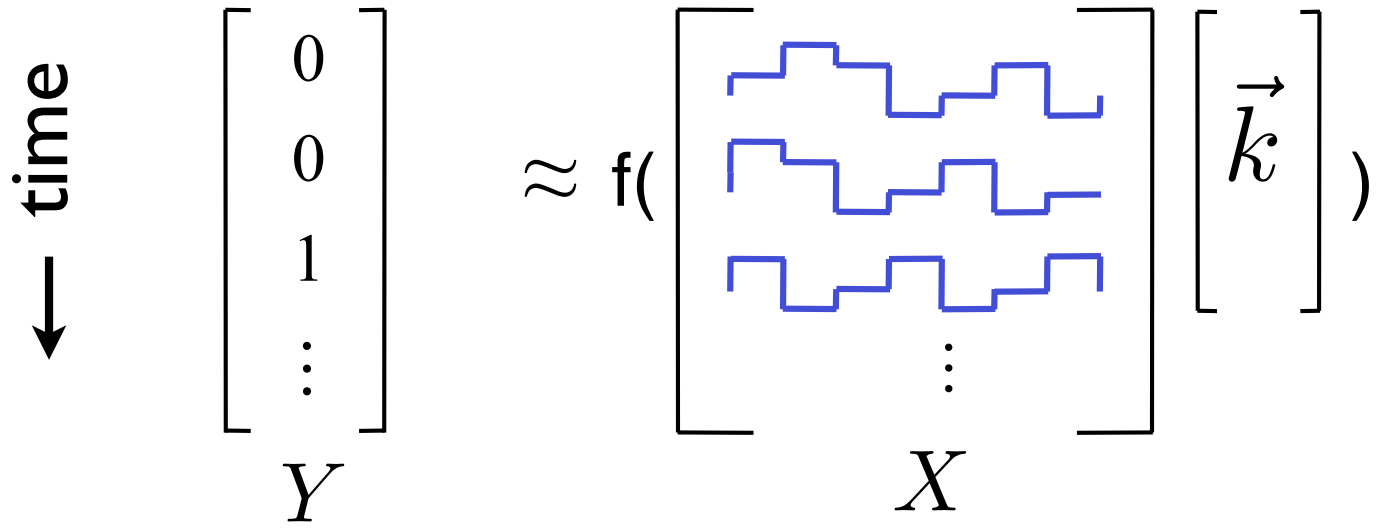
$P(Y|X,$

$X\vec{k})^\top (Y - X\vec{k})\Big)$

General point: minimizing a sum of squares is *always* equivalent to maximizing likelihood under a Gaussian noise model!

Take log, differentiate and set to zero.

But noise is *not* Gaussian: binary responses



probability of spike at bin t

nonlinearity

Bernoulli GLM:

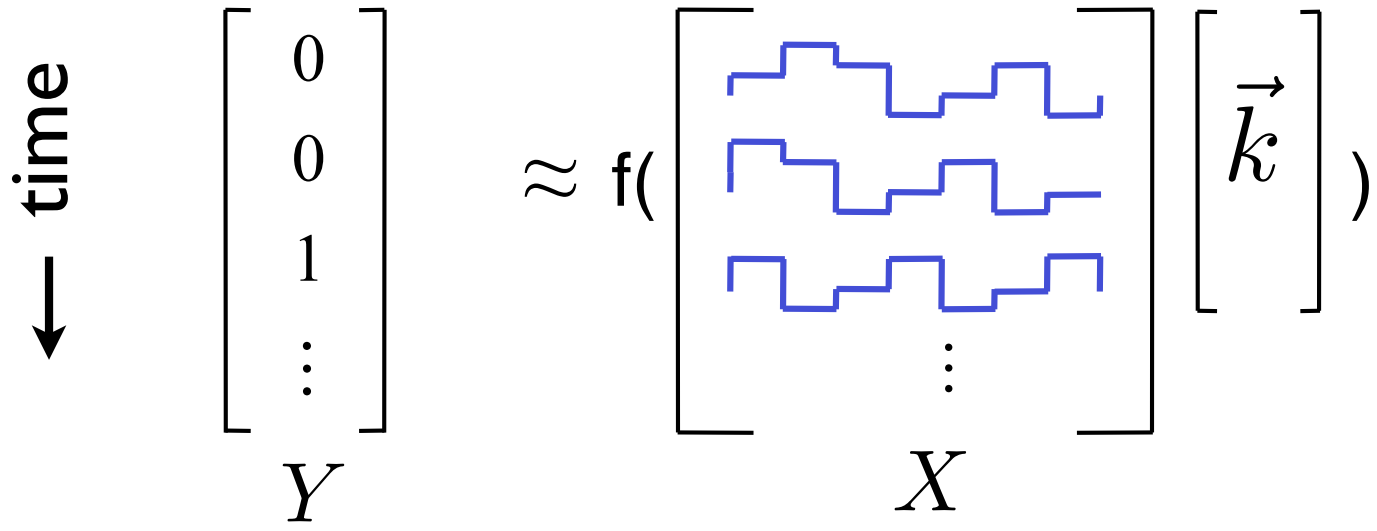
(coin flipping model
y = 0 or 1)

$$p(y_t = 1|\vec{x}_t) = p_t$$

$$p(y_t = 0|\vec{x}_t) = 1 - p_t$$

$$p_t = f(\vec{x}_t \cdot \vec{k})$$

But noise is *not* Gaussian: binary responses



probability of spike at bin t

nonlinearity

Bernoulli GLM:

(coin flipping model y = 0 or 1)

$$p(y_t = 1|\vec{x}_t) = p_t$$

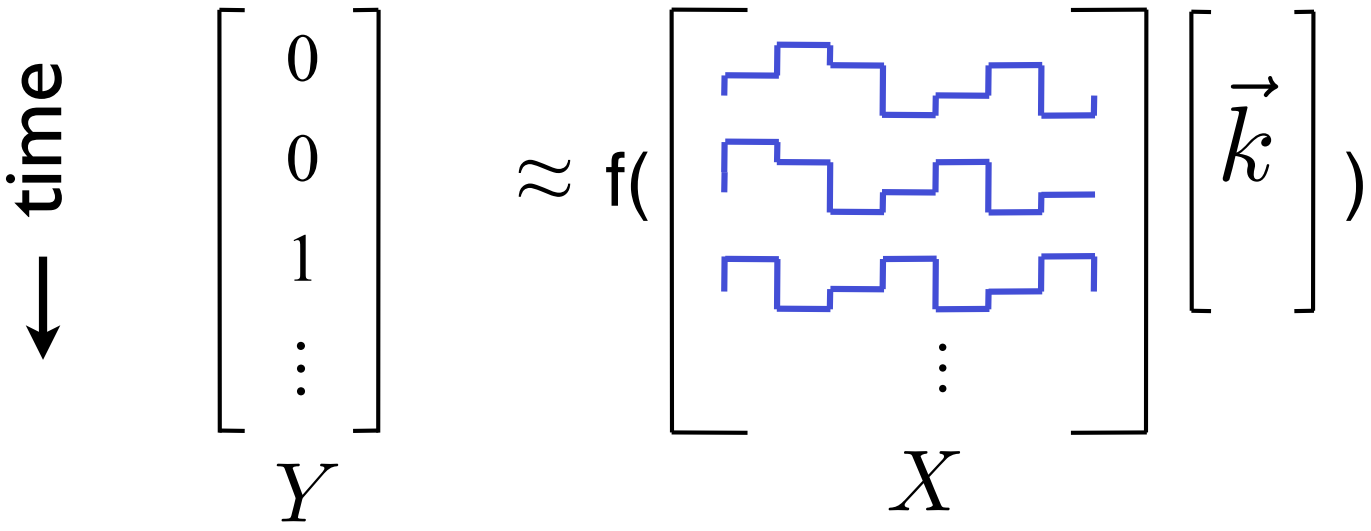
$$p(y_t = 0|\vec{x}_t) = 1 - p_t$$

$$p_t = f(\vec{x}_t \cdot \vec{k})$$

logistic regression: $f(z) = \dfrac{1}{1 + e^{-z}}$ logistic function

- so logistic regression is a special case of a Bernoulli GLM!

But noise is *not* Gaussian: binary responses



probability of spike at bin t

nonlinearity

Bernoulli GLM:

(coin flipping model
y = 0 or 1)

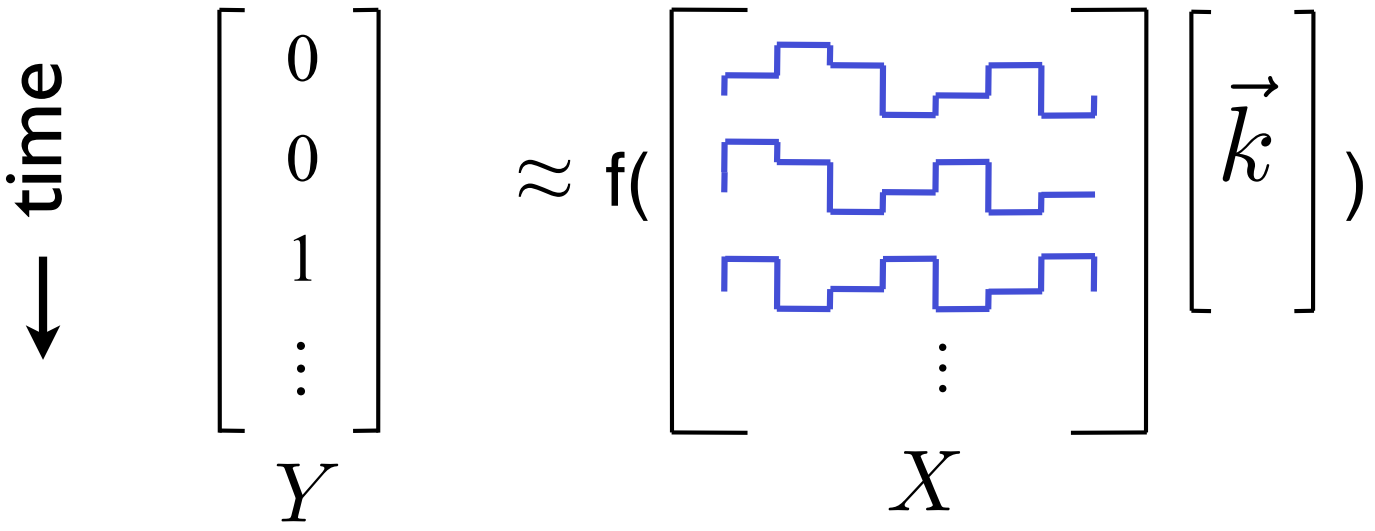
$$p(y_t = 1|\vec{x}_t) = p_t$$

$$p_t = f(\vec{x}_t \cdot \vec{k})$$

$$p(y_t = 0|\vec{x}_t) = 1 - p_t$$

$$p(y_t|\vec{x}_t) = p_t^{y_t}(1 - p_t)^{1-y_t}$$

But noise is *not* Gaussian: binary responses



probability of spike at bin t

nonlinearity

Bernoulli GLM:

(coin flipping model
y = 0 or 1)

$$p(y_t = 1|\vec{x}_t) = p_t$$

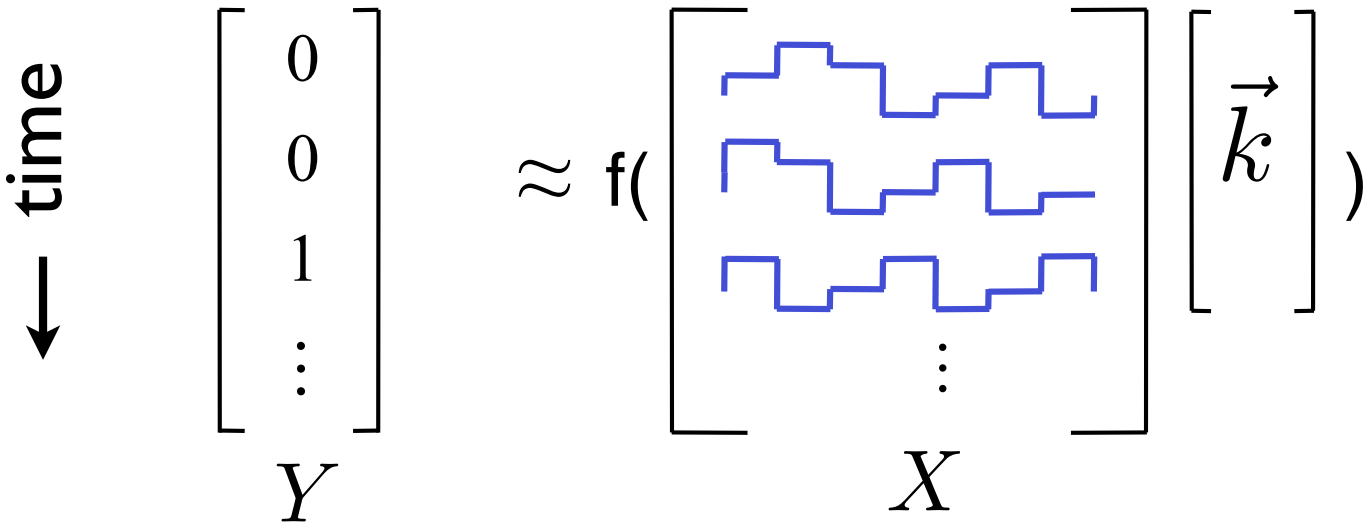
$$p(y_t = 0|\vec{x}_t) = 1 - p_t$$

$$p(y_t|\vec{x}_t) = p_t^{y_t}(1 - p_t)^{1-y_t}$$

$$p_t = f(\vec{x}_t \cdot \vec{k})$$

log-likelihood: $\mathcal{L} = \sum_{t=1}^{T} \left(y_t \log f(\vec{x}_t \cdot \vec{k}) + (1 - y_t) \log(1 - f(\vec{x}_t \cdot \vec{k})) \right)$

But noise is *not* Gaussian: binary responses



Matrix-vector version:

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

in python:

```
z = f(X @ k)  # compute P(heads) for each time bin
L = np.dot(y, np.log(z)) + np.dot(1-y, np.log(1-z))
```

log-likelihood: $\mathcal{L} = \sum_{t=1}^{T} \left(y_t \log f(\vec{x}_t \cdot \vec{k}) + (1-y_t) \log(1 - f(\vec{x}_t \cdot \vec{k})) \right)$

non-negative integer responses
("Poisson regression")

Poisson GLM:
(integer y≥0)

nonlinearity

firing rate
(aka "conditional intensity")

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

$$y_t|\vec{x}_t, \vec{k} \sim \text{Poiss}(\Delta\lambda_t)$$

time bin size

encoding distribution:

$$p(y_t|\vec{x}_t, \vec{k}) = \frac{(\Delta\lambda_t)^{y_t}}{y_t!} e^{-\Delta\lambda_t}$$

log-likelihood:

$$\mathcal{L} = \log p(Y|X, \vec{k}) = \sum_t \left(y_t \log f(\vec{x}_t \cdot \vec{k}) - \Delta f(\vec{x}_t \cdot \vec{k}) \right) + \ const$$

python:

```
z = f(X @ k)  # conditional intensity
L = np.dot(y, np.log(z)) - dt*np.sum(z)
```

GLM: Summary:

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

(note: assuming "identity" nonlinearity)

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

```
k_hat = np.linalg.solve(X.T@X, X.T@Y)
```

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

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

```
z = f(X @ k)  # compute P(heads) for each time bin
L = np.dot(y, np.log(z)) + np.dot(1-y, np.log(1-z))
```

3. Poisson GLM: $y_t|\vec{x}_t,\vec{k} \sim \mathrm{Poiss}(\Delta\lambda_t)$

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

```
z = f(X @ k)  # conditional intensity
L = np.dot(y, np.log(z)) - dt*np.sum(z)
```

GLM tutorial:

MATLAB: https://github.com/pillowlab/GLMspiketraintutorial

python: https://github.com/pillowlab/GLMspiketraintutorial_python

- **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.

Do it: tutorial1_PoissonGLM.m

1. cd into correct directory ('GLMspiketraintutorial')
2. open tutorial: use `ctrl-enter`, `ctrl-down`

Dataset:

- retinal ganglion cell spike trains [Uzzell & Chichilnisky 2004]
- 2 OFF cells, 2 off cells
- full-field, binary white noise stimuli

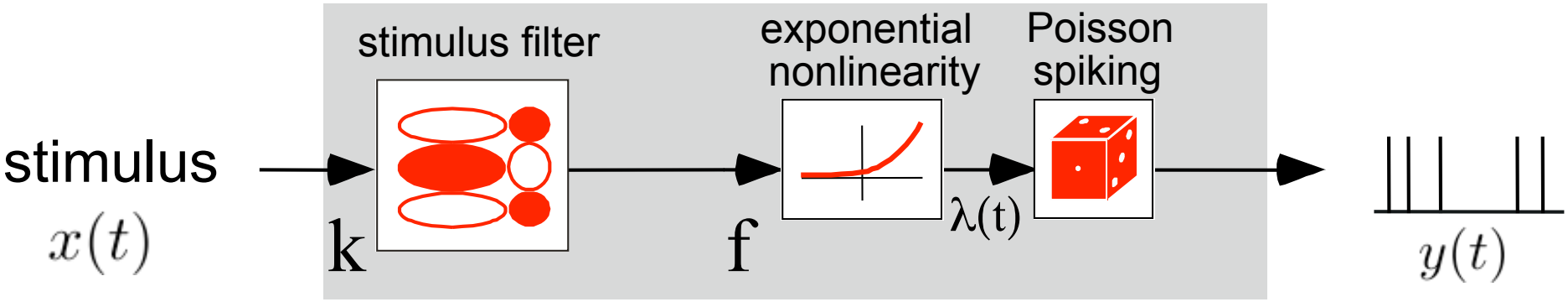
Do it: tutorial1_PoissonGLM.m

1. cd into correct directory ('GLMspiketraintutorial')
2. open tutorial: use `ctrl-enter`, `ctrl-down`

Topics:

1-2. load data, bin spike trains
3. build design matrix (slow and fast versions)
4a. compute STA (for vizualization)
4b. *linear-Gaussian GLM* ("least-squares regression")
5. *Poisson GLM*: fit assuming exponential nonlinearity
6. *Poisson GLM*: non-parametric estimate of nonlinearity
7. model comparison: log-likelihood & mutual information
8. model comparison: AIC
9. simulate from fitted model

GLM (aka Linear-Nonlinear-Poisson)



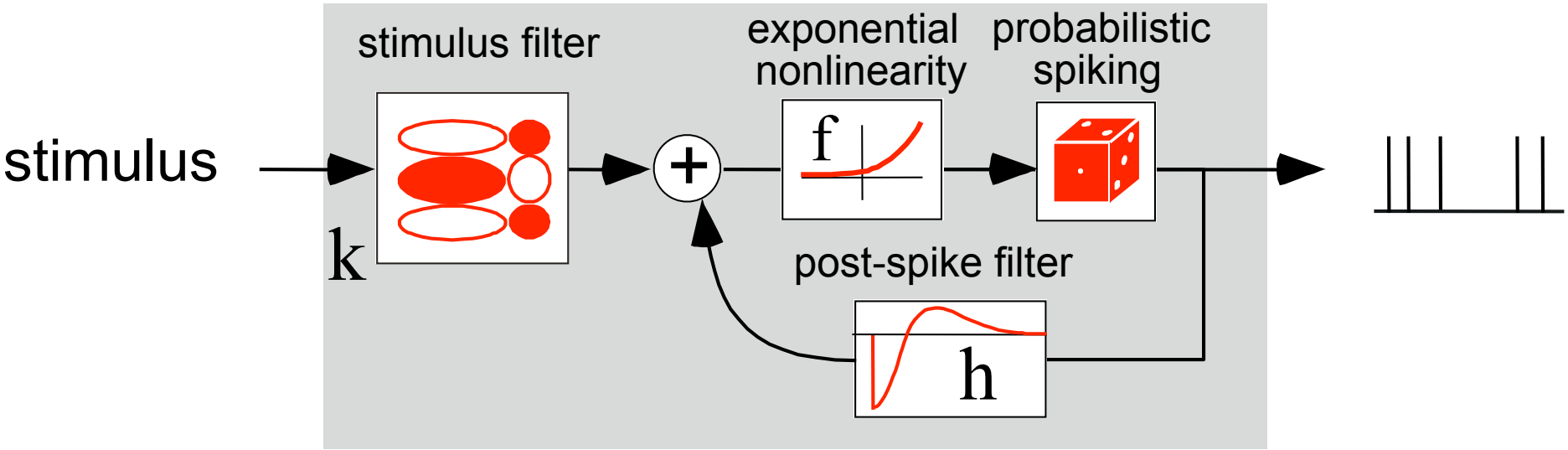
conditional intensity (spike rate)

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

- output: Poisson process
- problem: assumes spiking depends only on stimulus!

(Truccolo et al 04)

Poisson GLM with spike-history dependence



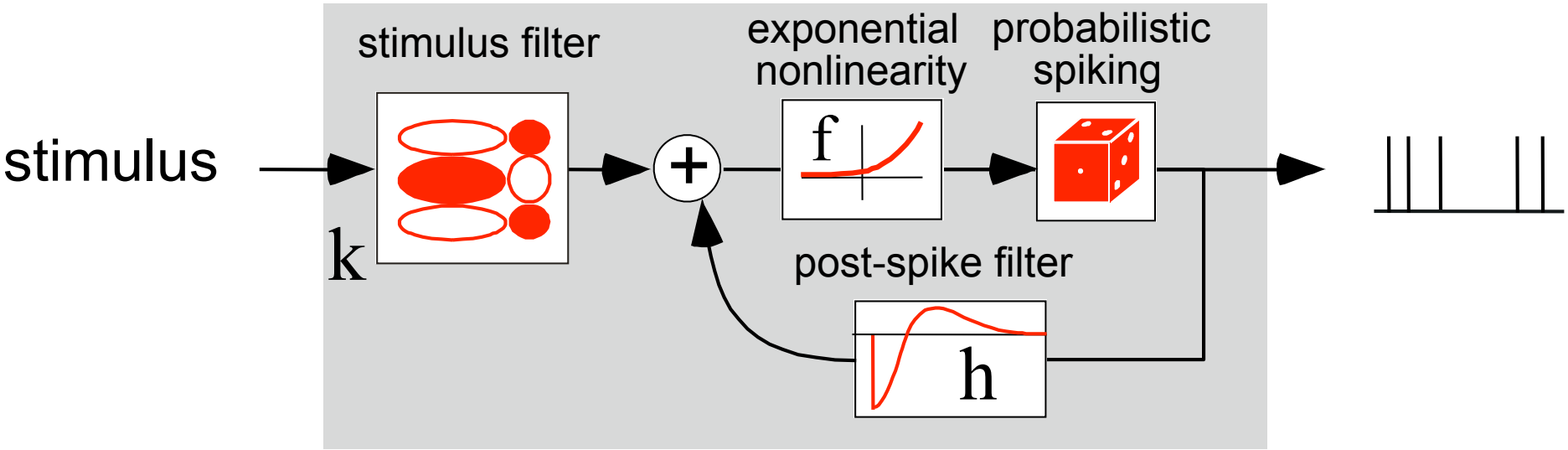
conditional intensity
(spike rate)

$$\lambda(t) = f\left(\vec{k}\cdot\vec{x}(t) + \vec{h}\cdot\vec{y}_{hst}(t)\right)$$

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

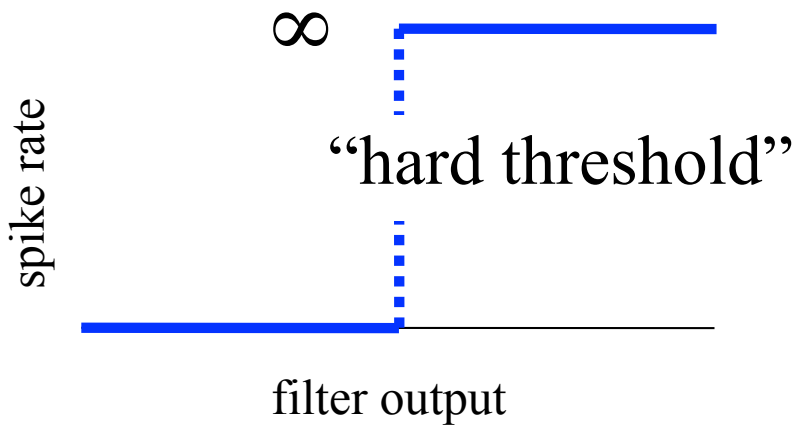
- output: no longer a Poisson process

Poisson GLM with spike-history dependence

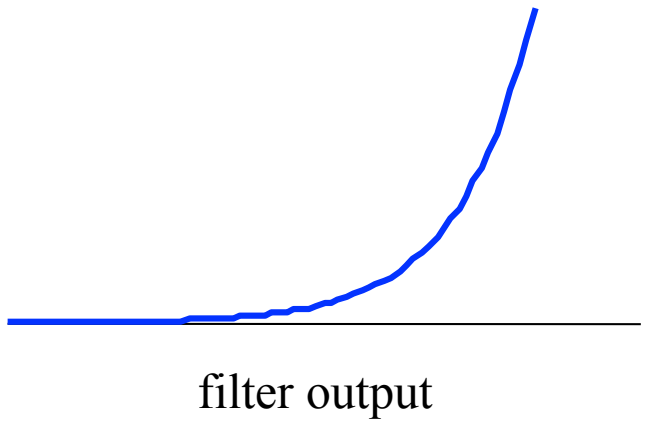


- interpretation: "soft-threshold" integrate-and-fire model

traditional IF

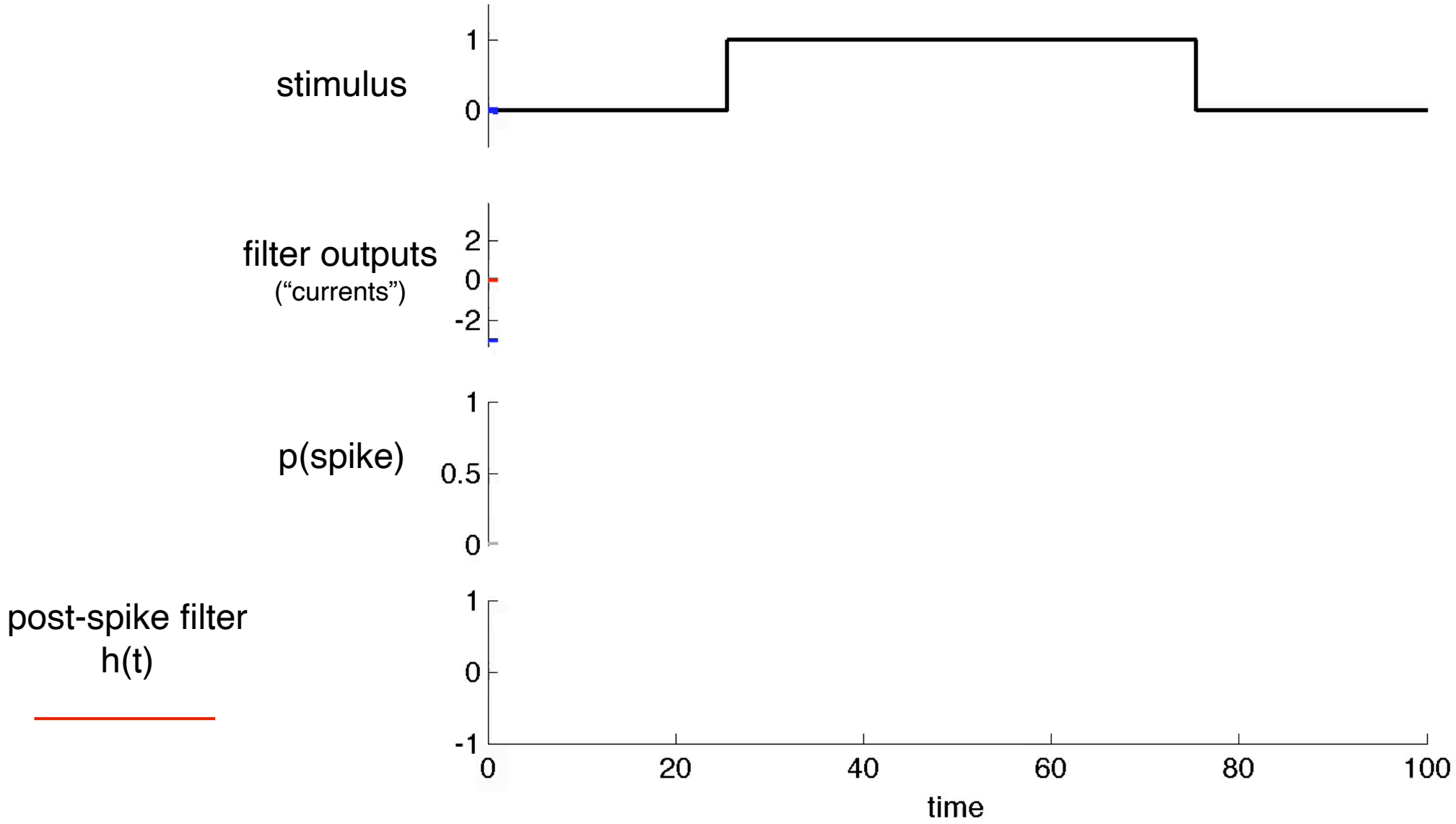


"soft-threshold" IF



GLM dynamic behaviors

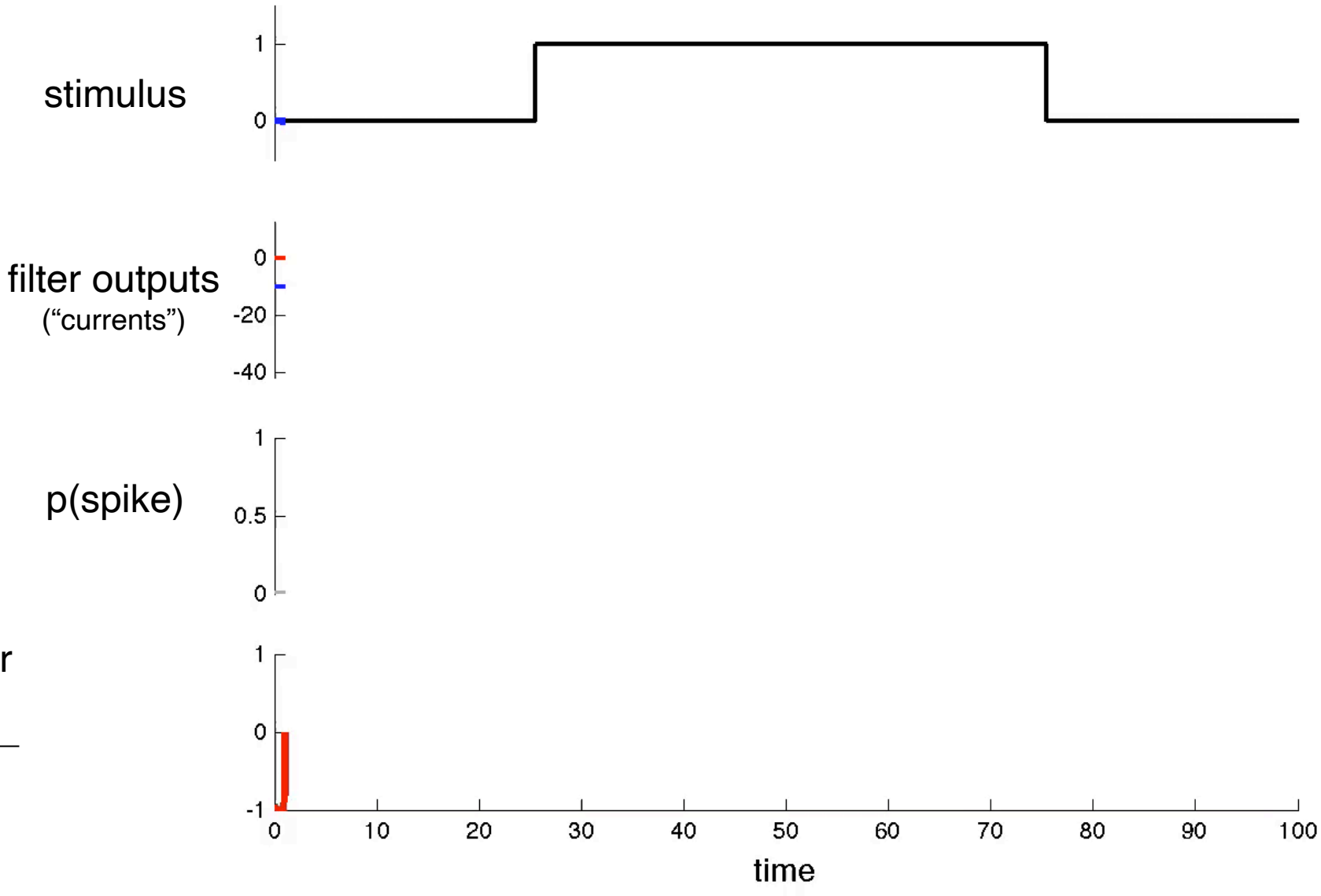
- irregular spiking



GLM dynamic behaviors

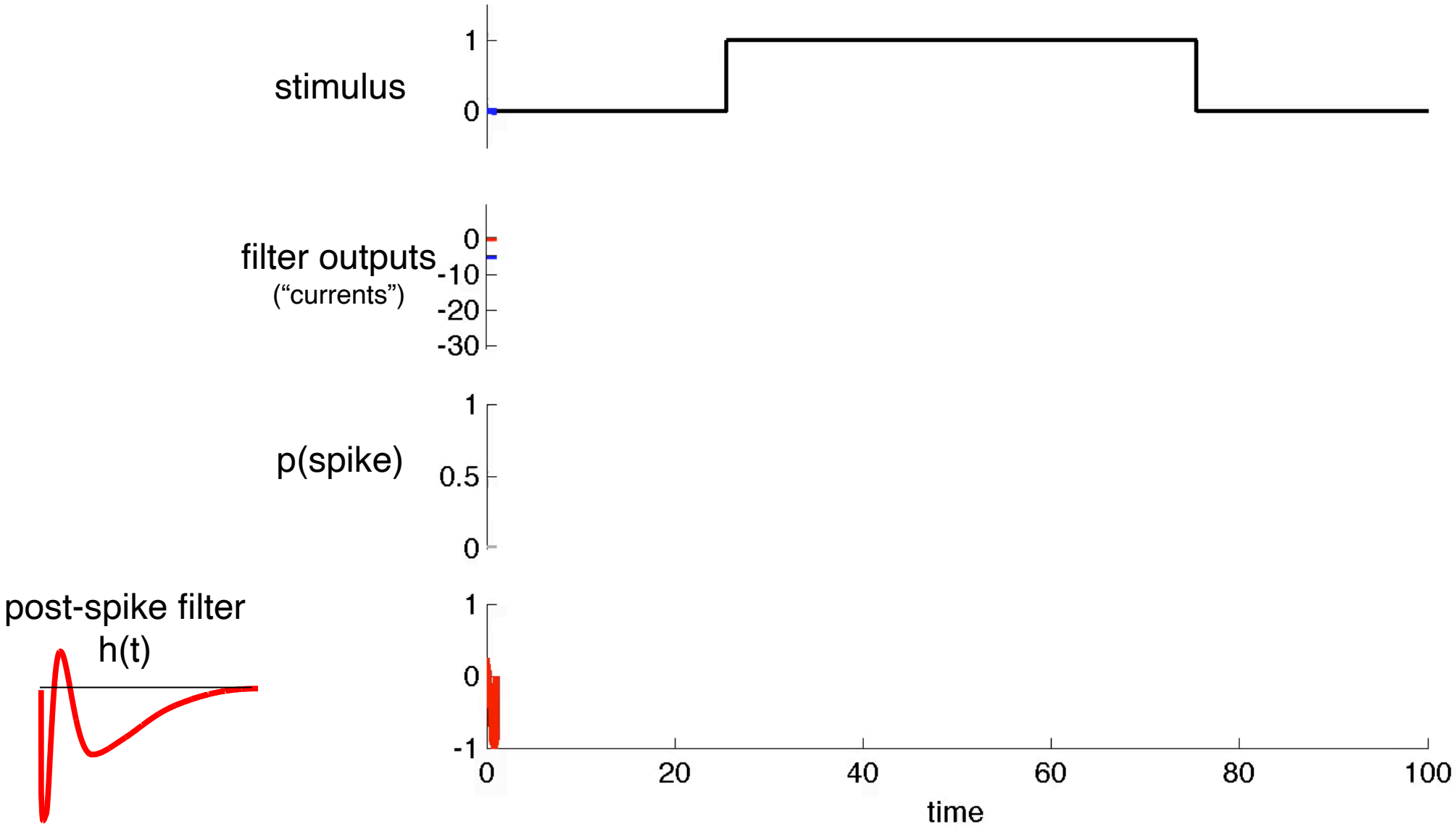
(Weber & Pillow 2016)

- regular spiking



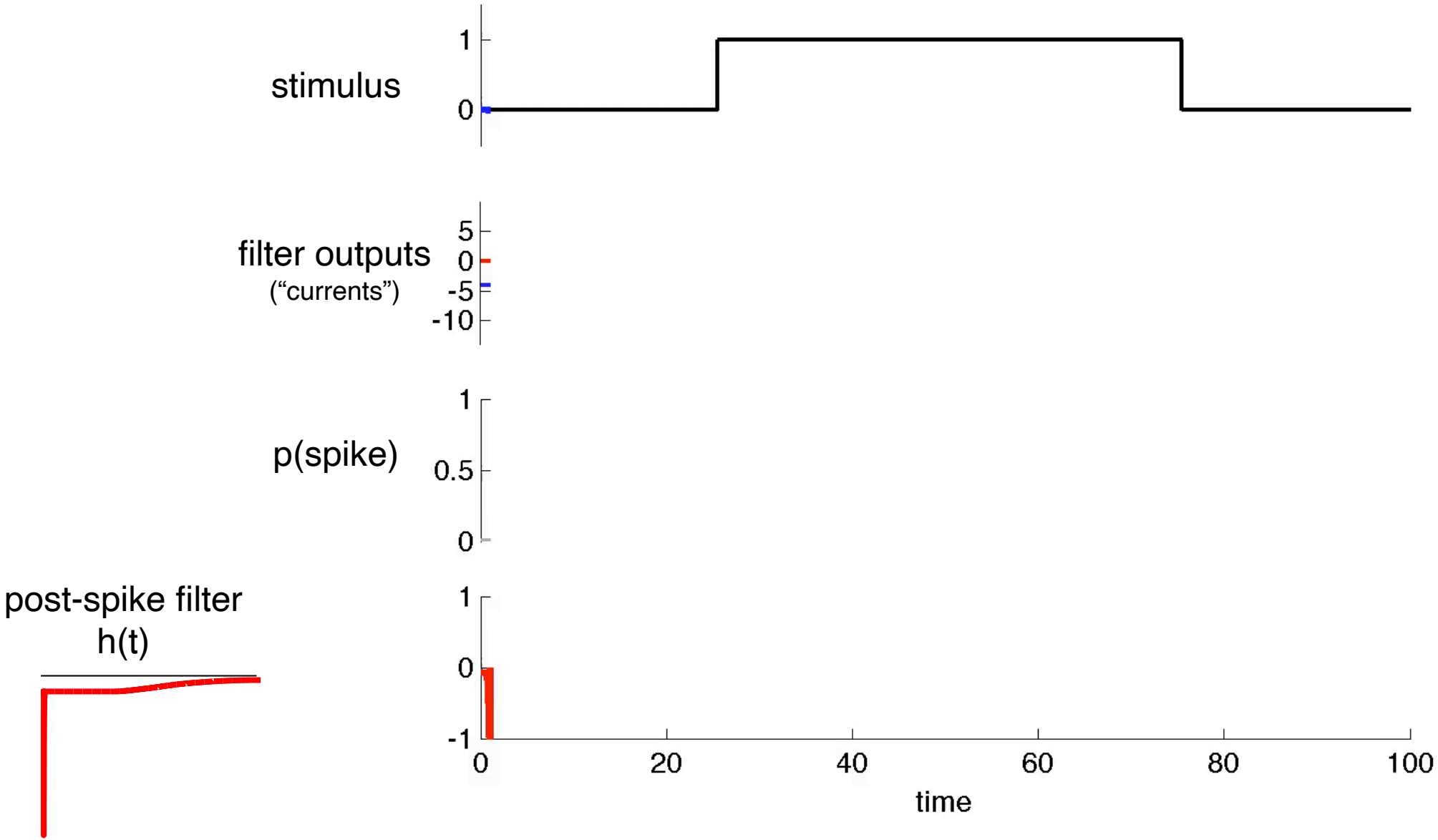
GLM dynamic behaviors

- bursting



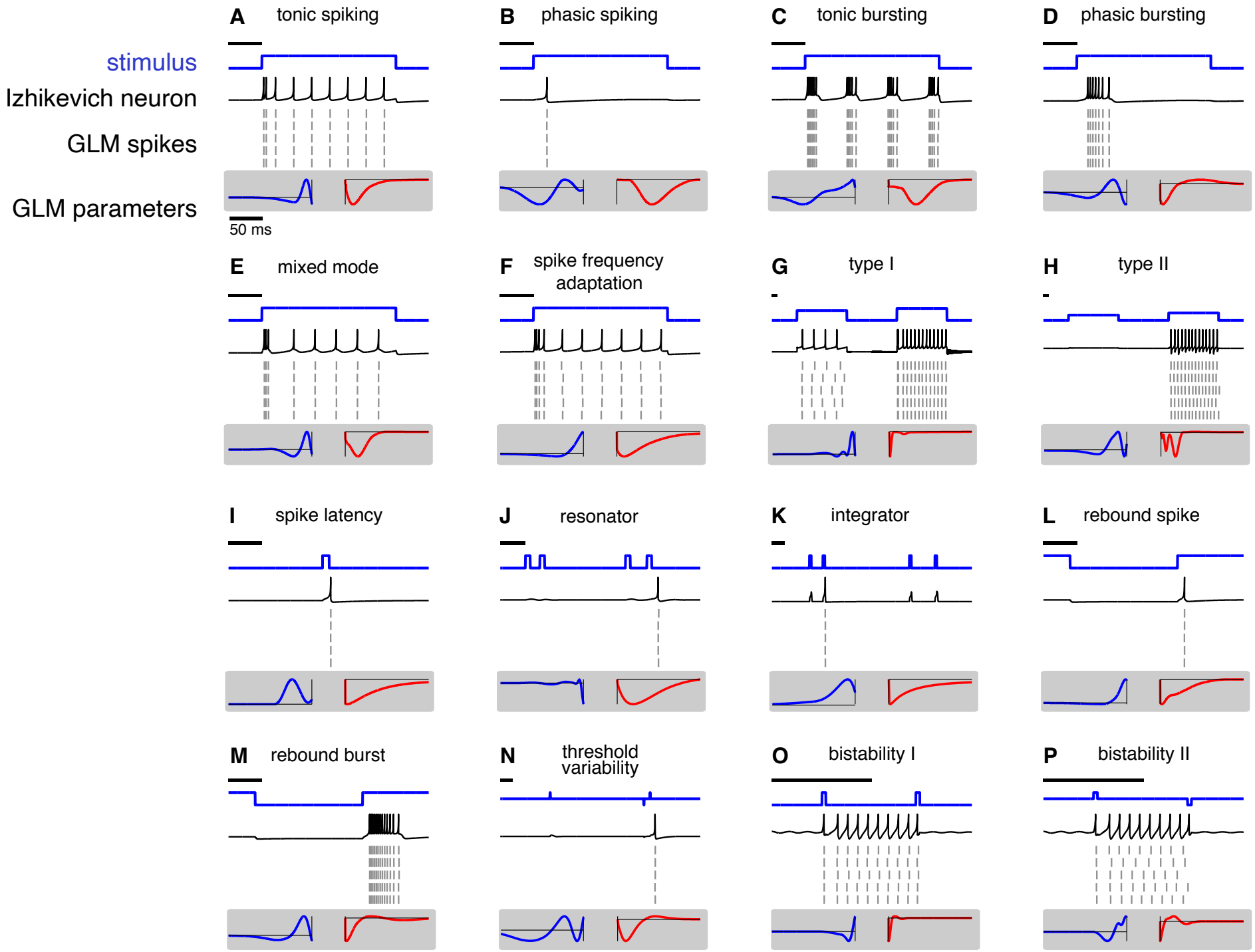
GLM dynamic behaviors

- adaptation

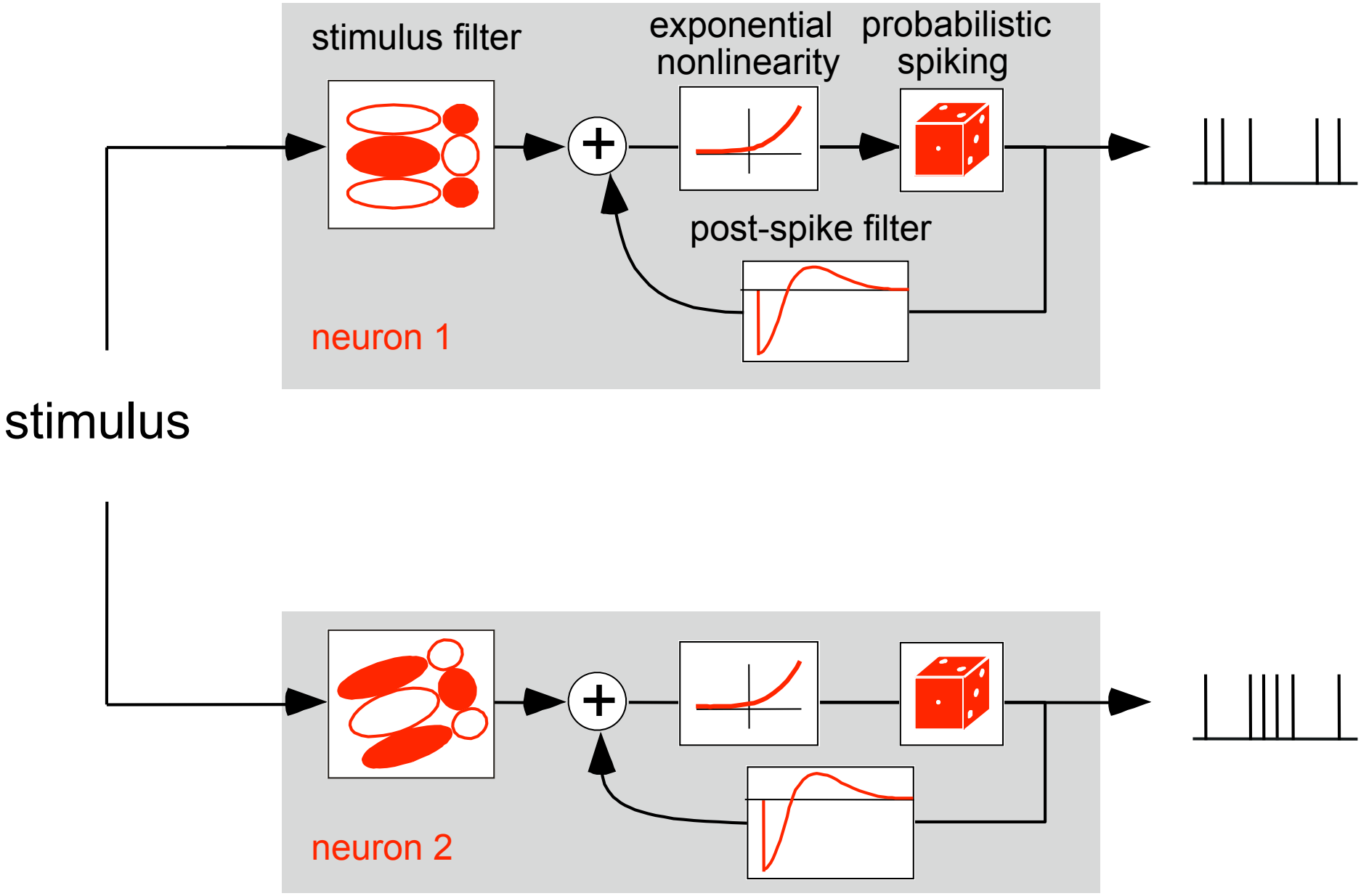


GLM dynamic behaviors (from Izhikevich)

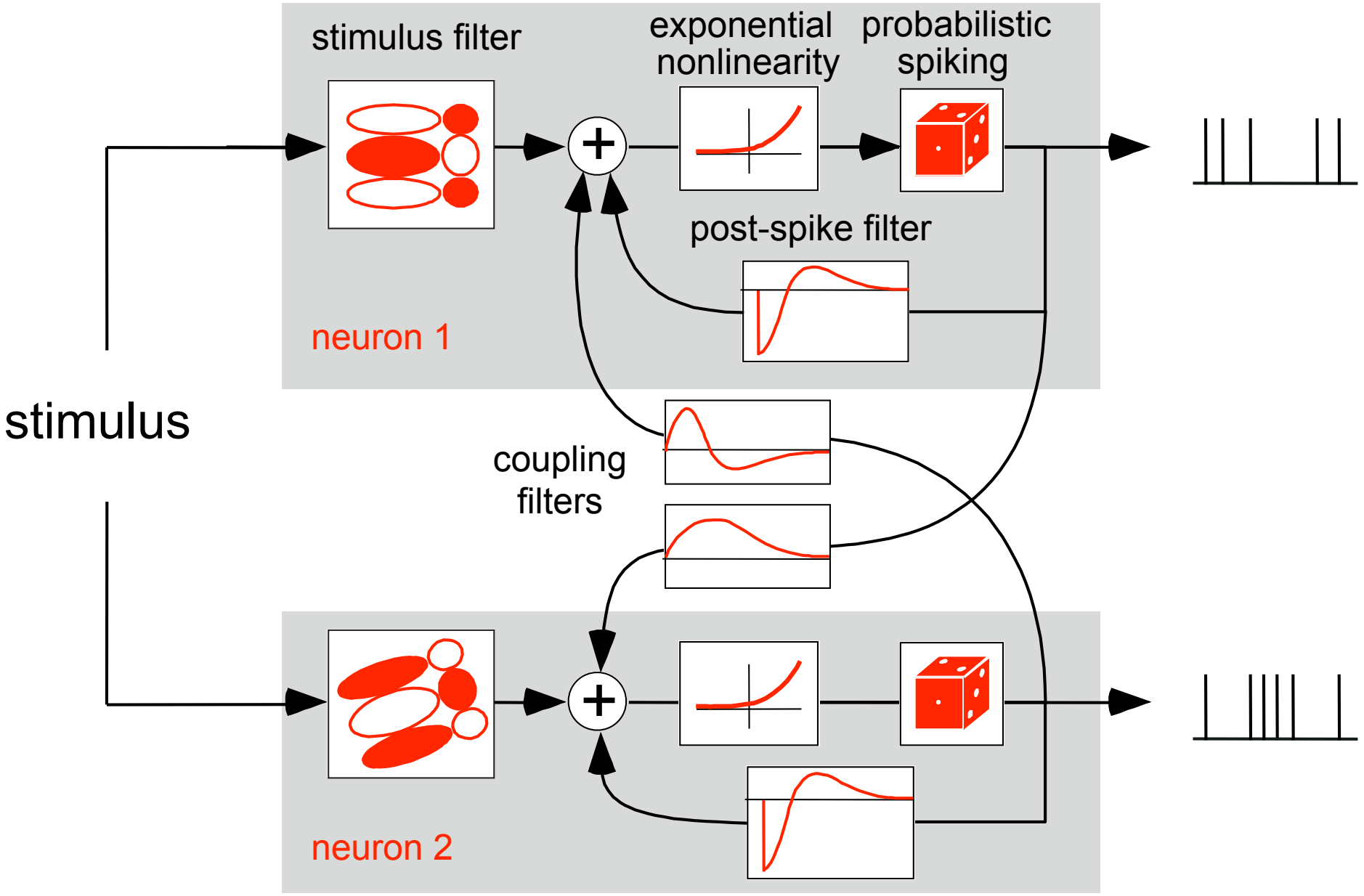
(Weber & Pillow 2017)



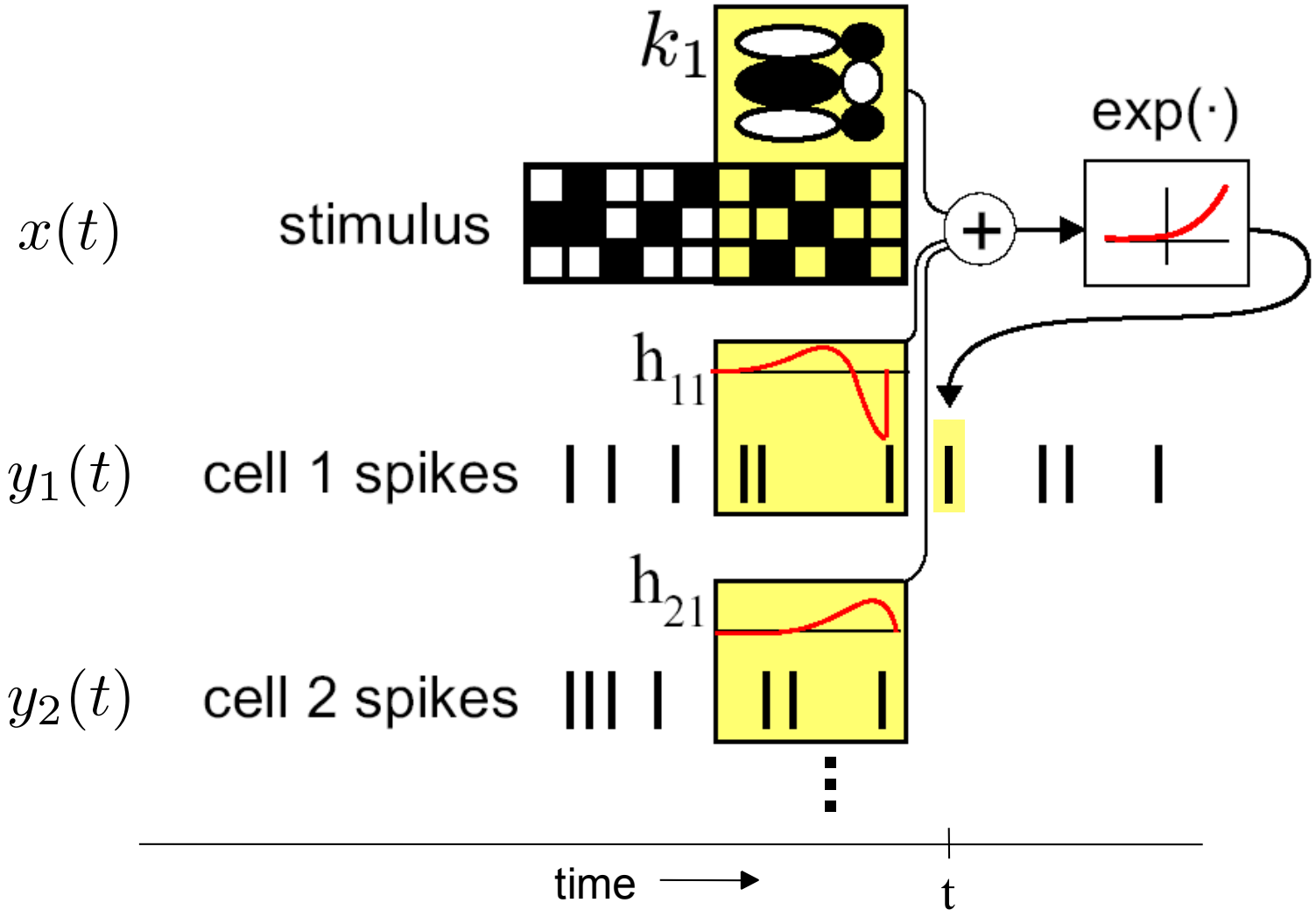
multi-neuron GLM



multi-neuron GLM



GLM equivalent diagram:



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

Do it: tutorial2_spikehistcoupledGLM.m

Topics:

1-2. load data & bin spike trains

3. build design matrix, now including spike history!

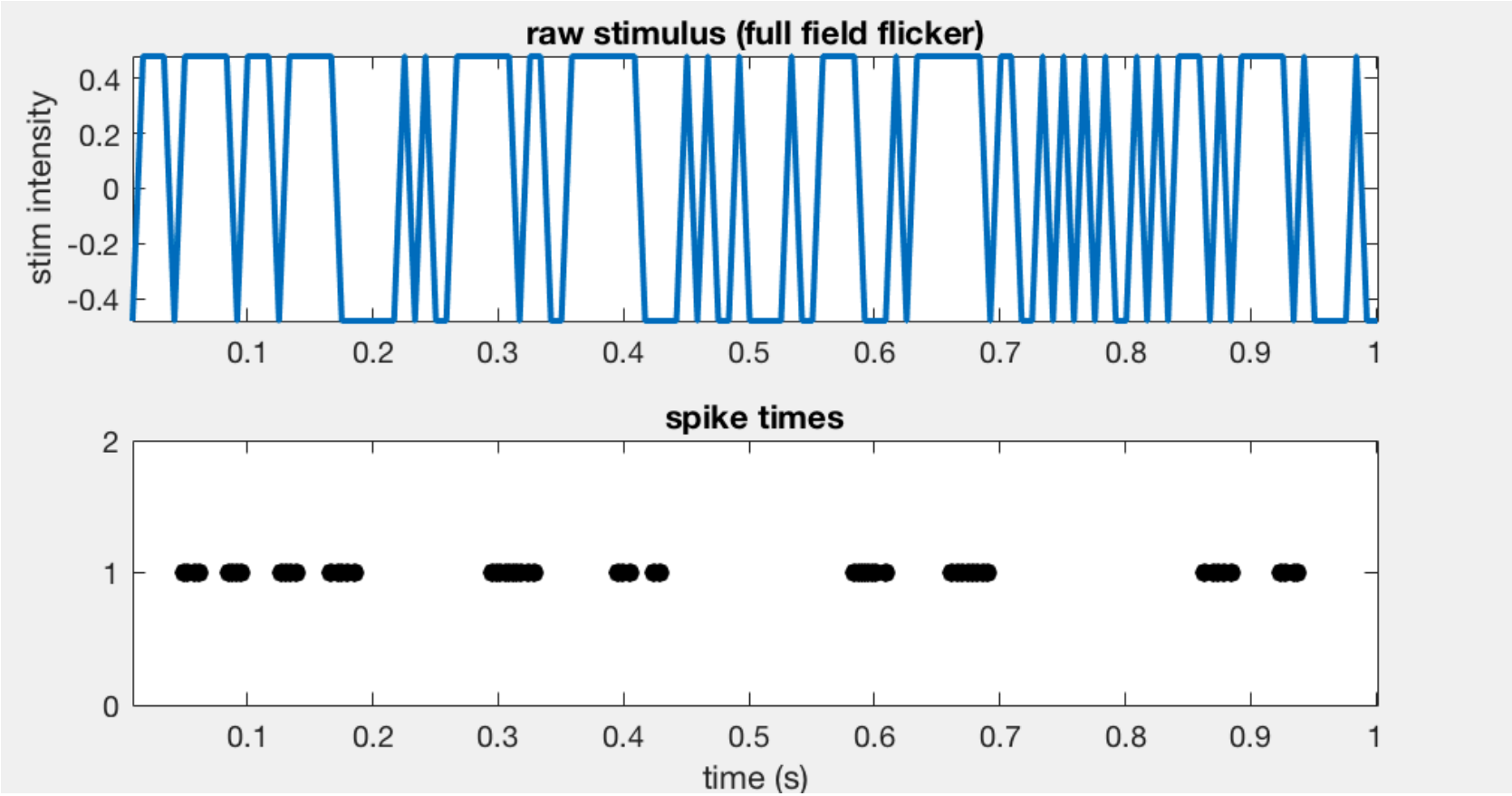
4. fit Poisson GLM with spike-history

5. fit coupled Poisson GLM (4 neuron spike-history)

6. model comparison: log-likelihood & AIC

Example dataset

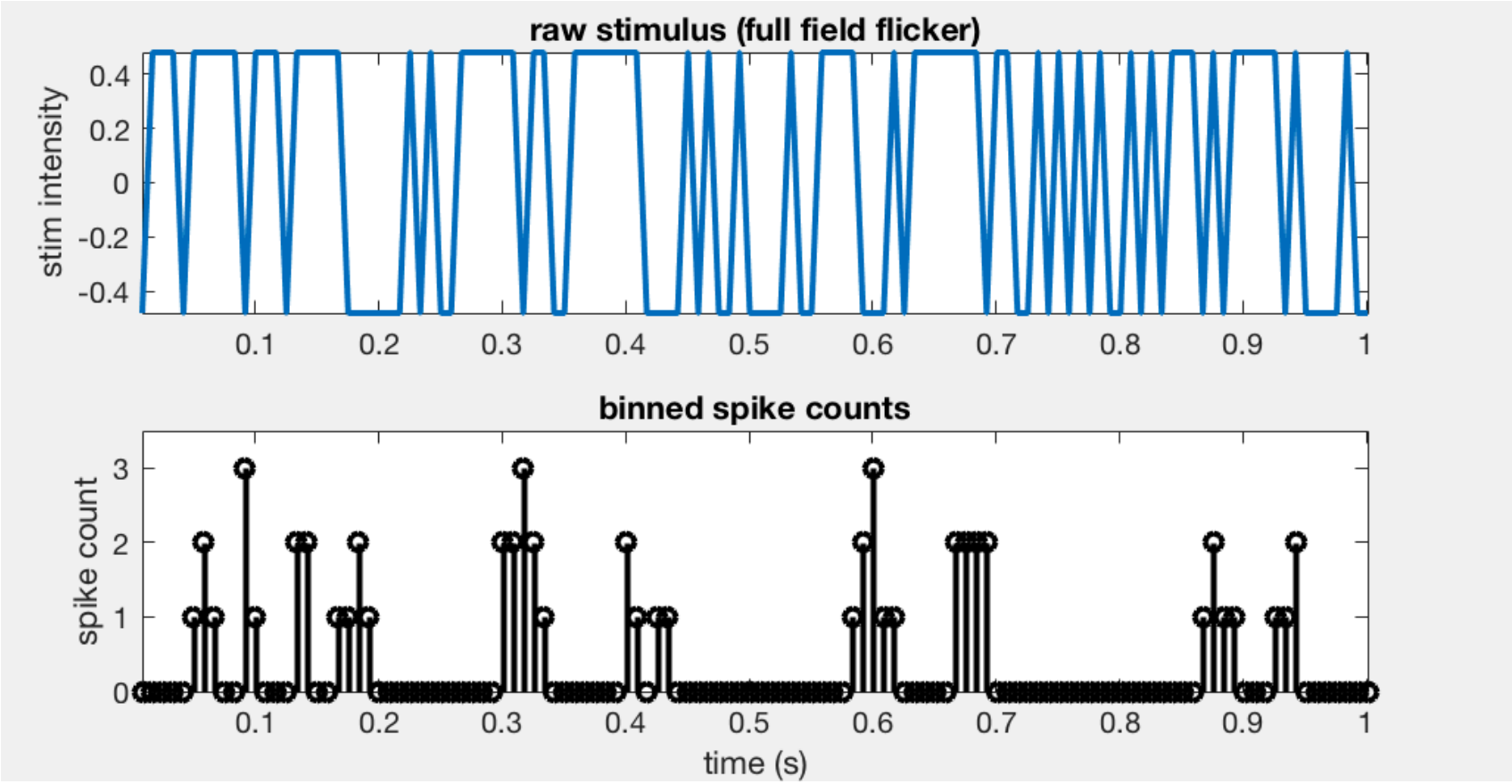
- stimulus = binary flicker
- parasol retinal ganglion cell spike responses



Uzzell et al (J Neurophys 04)

Example dataset

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

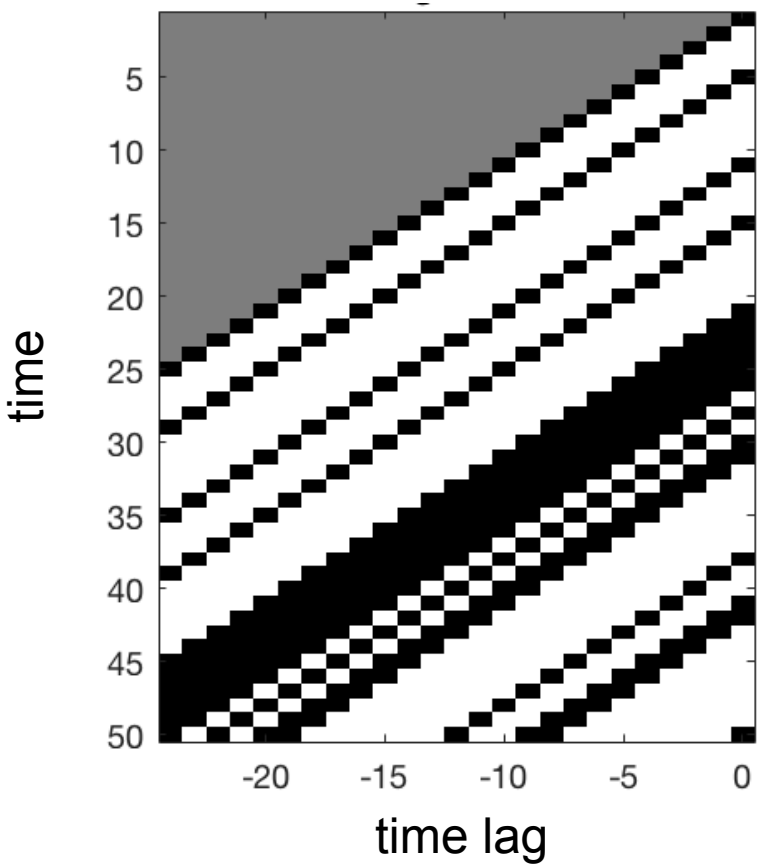


Uzzell et al (J Neurophys 04)

Stimulus-only GLM

design matrix

X



model

$P(Y|X)$

spike response

Y

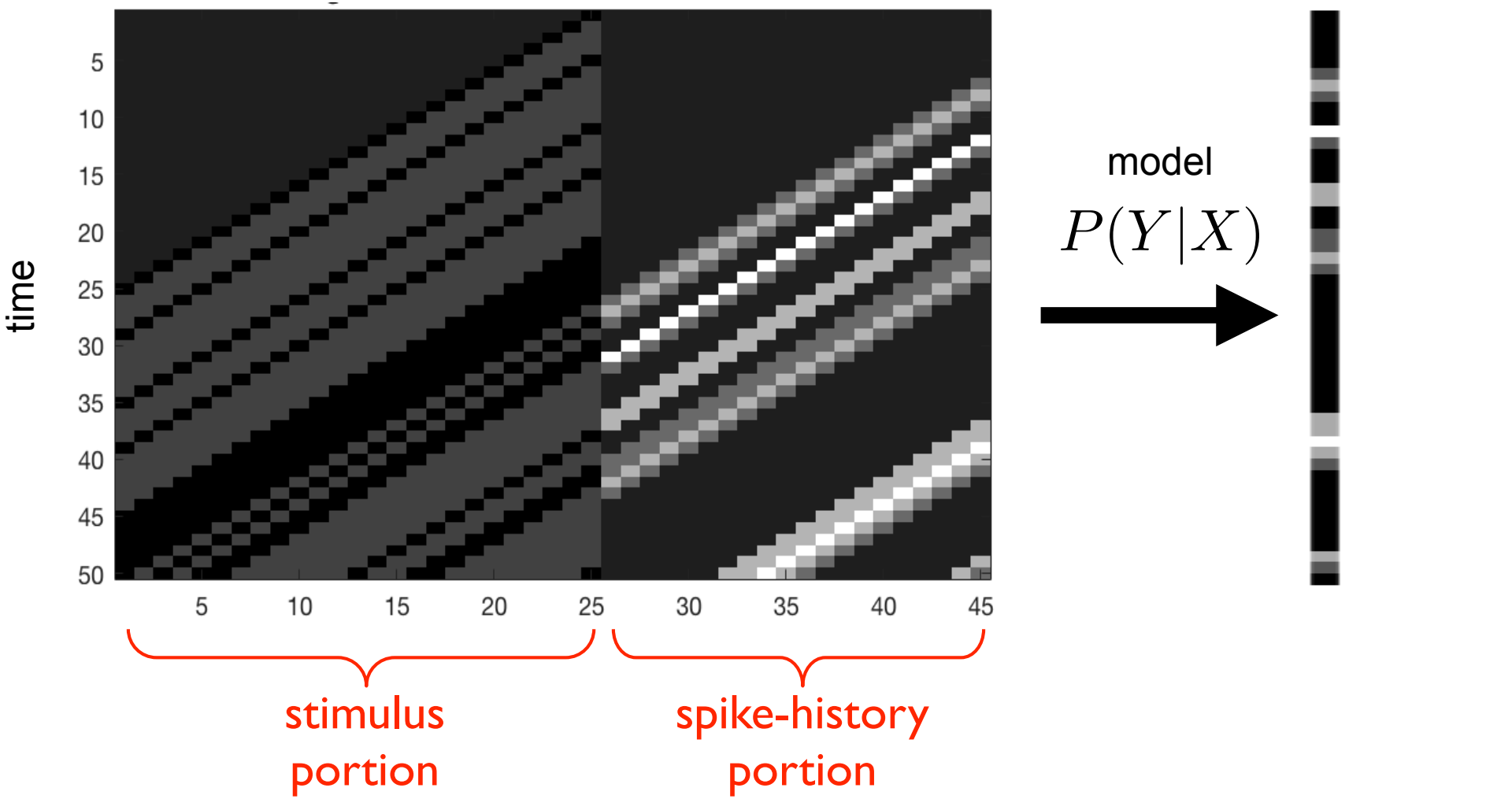
Stimulus + SpikeHistory GLM

design matrix

X

spike response

Y



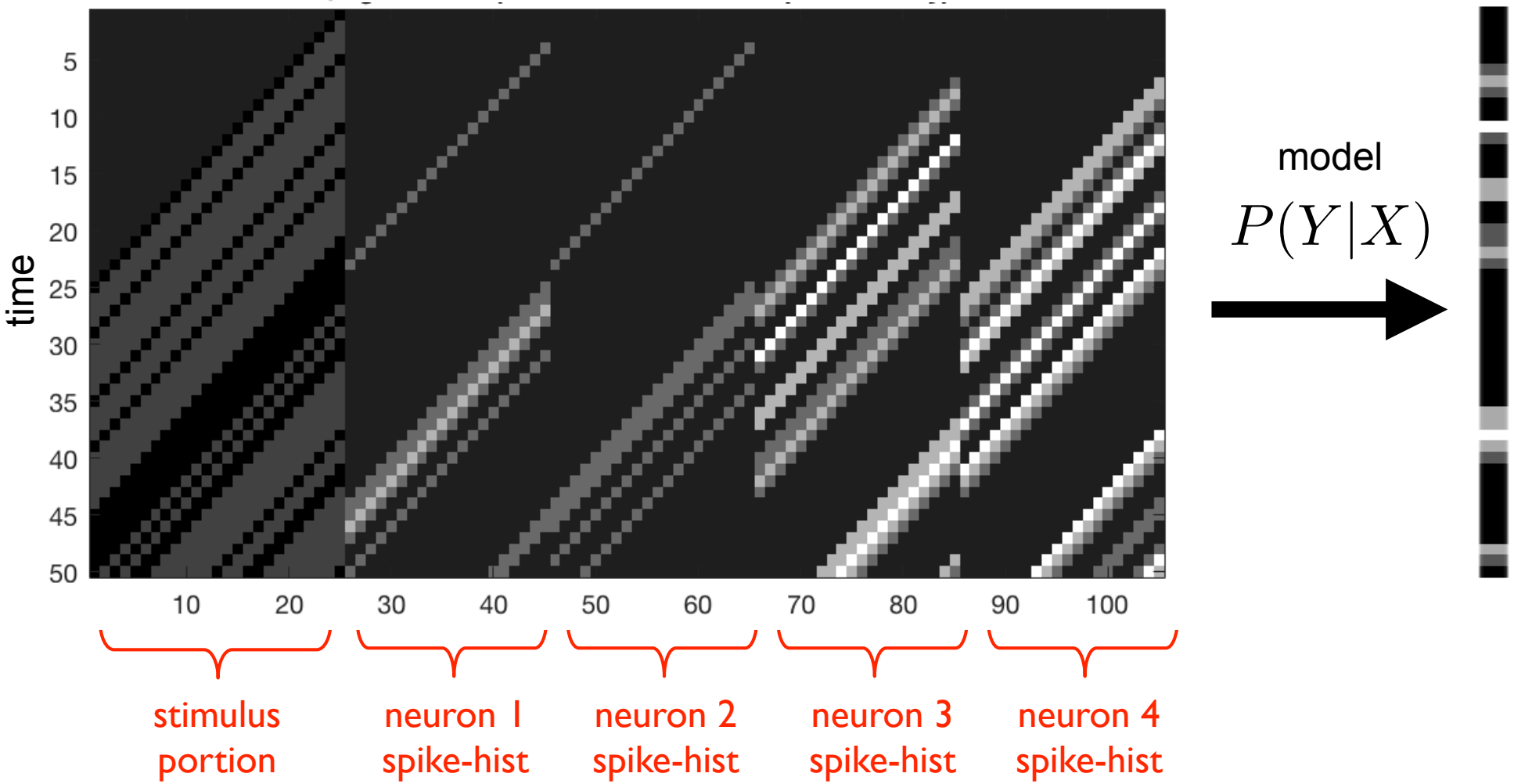
Stimulus + History + 3 Neuron Coupling GLM

design matrix

X

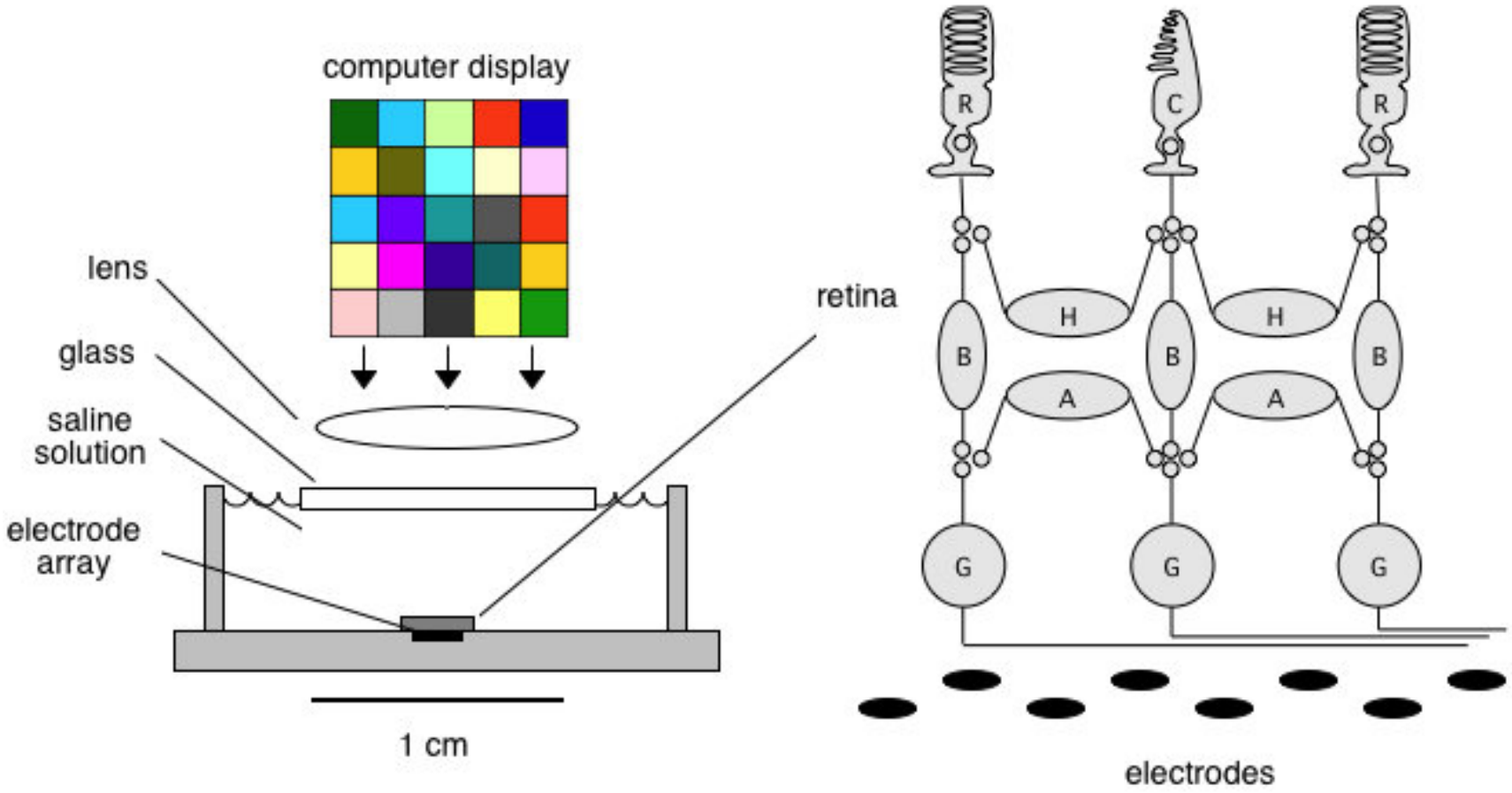
spike response

Y



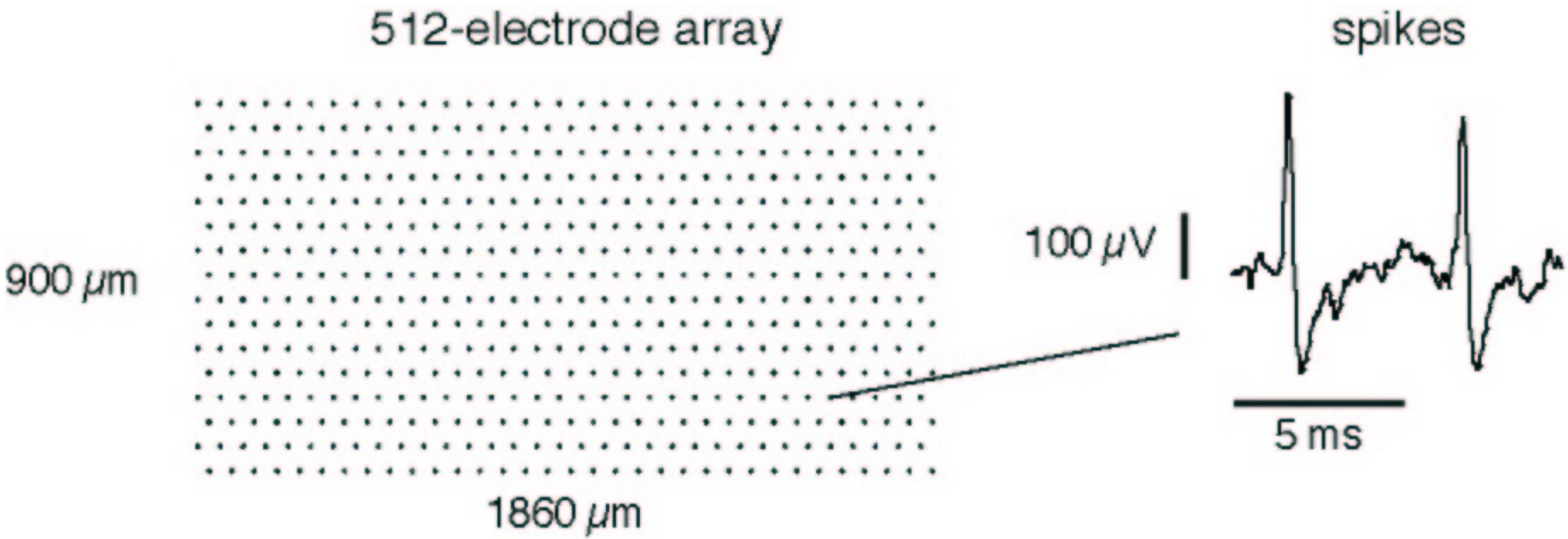
Application of GLM to Retinal Data

Application to Data: primate retina



Data: Chichilnisky Lab, The Salk Institute

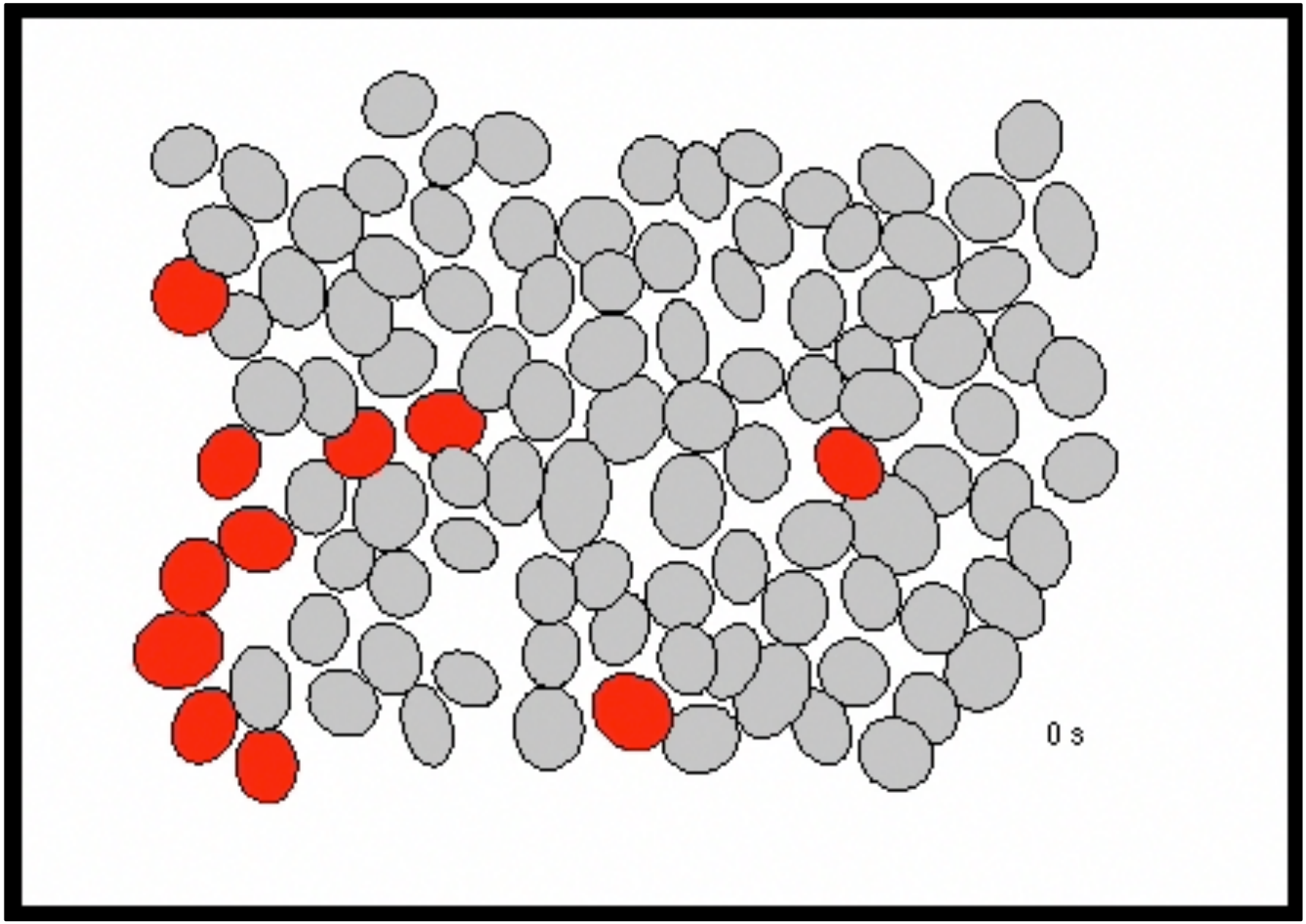
Application to Data: primate retina



Data: Chichilnisky Lab, The Salk Institute

Retinal responses to white noise

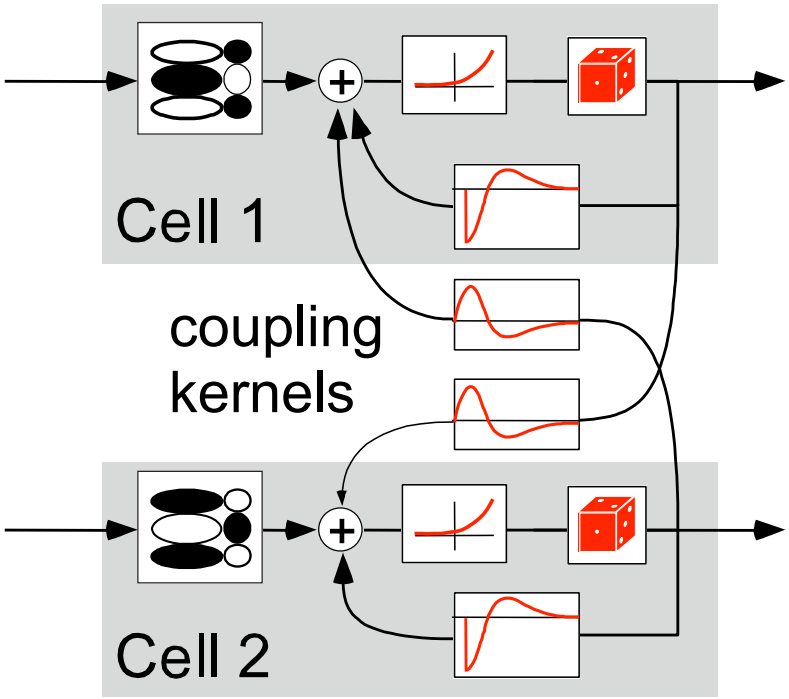
(ON parasol cells)



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

Approach: examine two versions of the GLM:

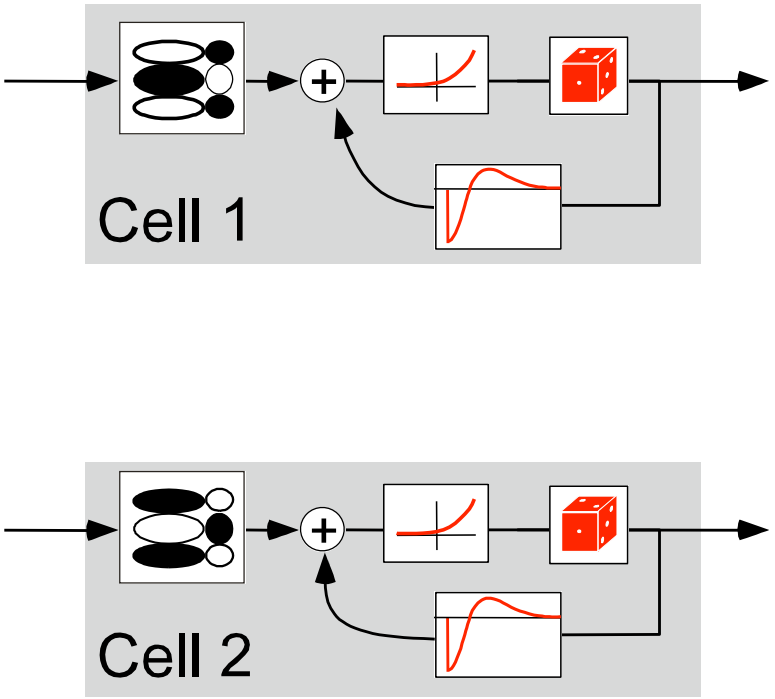
"full" model



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

joint encoding

"uncoupled" model



$$P(y_1|x)P(y_2|x)\cdots P(y_n|x)$$

independent encoding

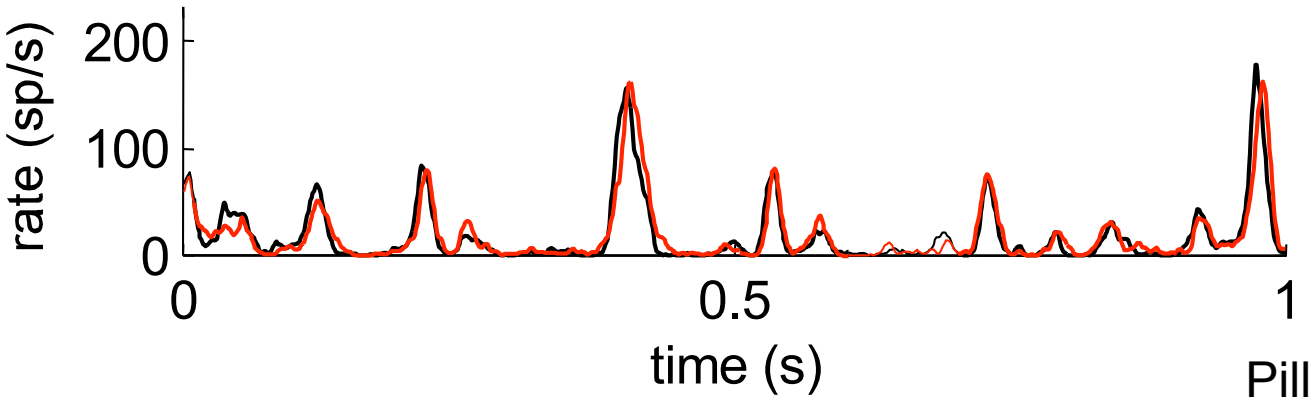
Predicting single-cell responses: $P(y_i|x)$

novel stimulus

retinal ganglion cell

simulation from fitted GLM

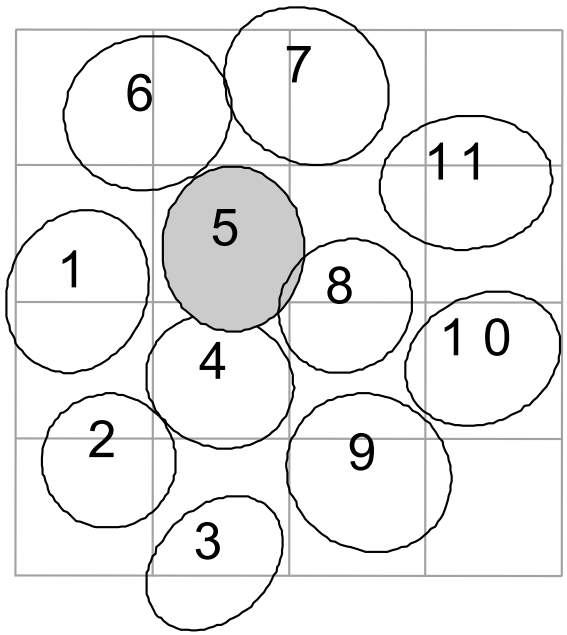
mean: psth (peristimulus time histogram)

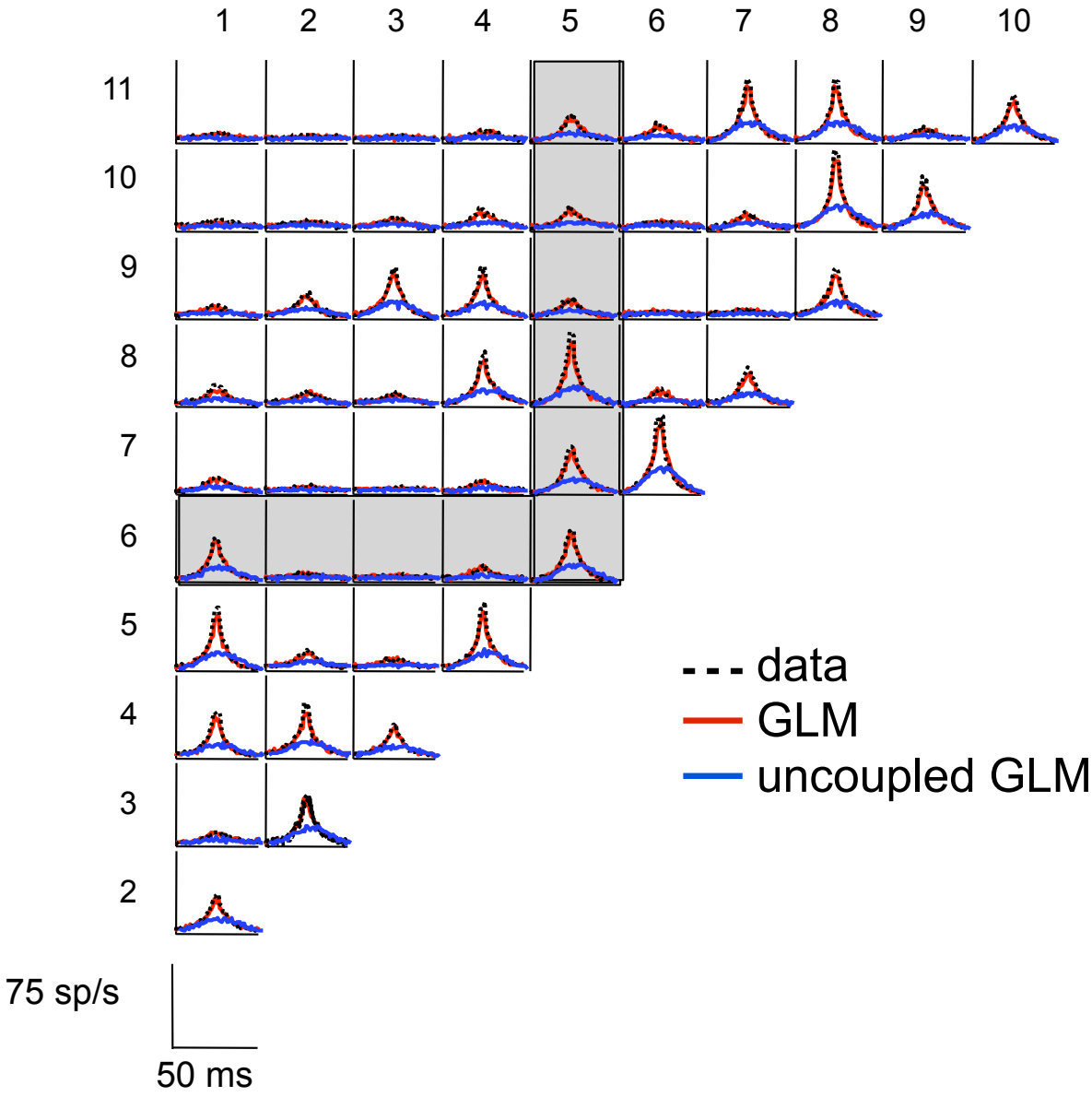


Pillow et al 2008

modeling correlation structure in neural spike trains

receptive fields



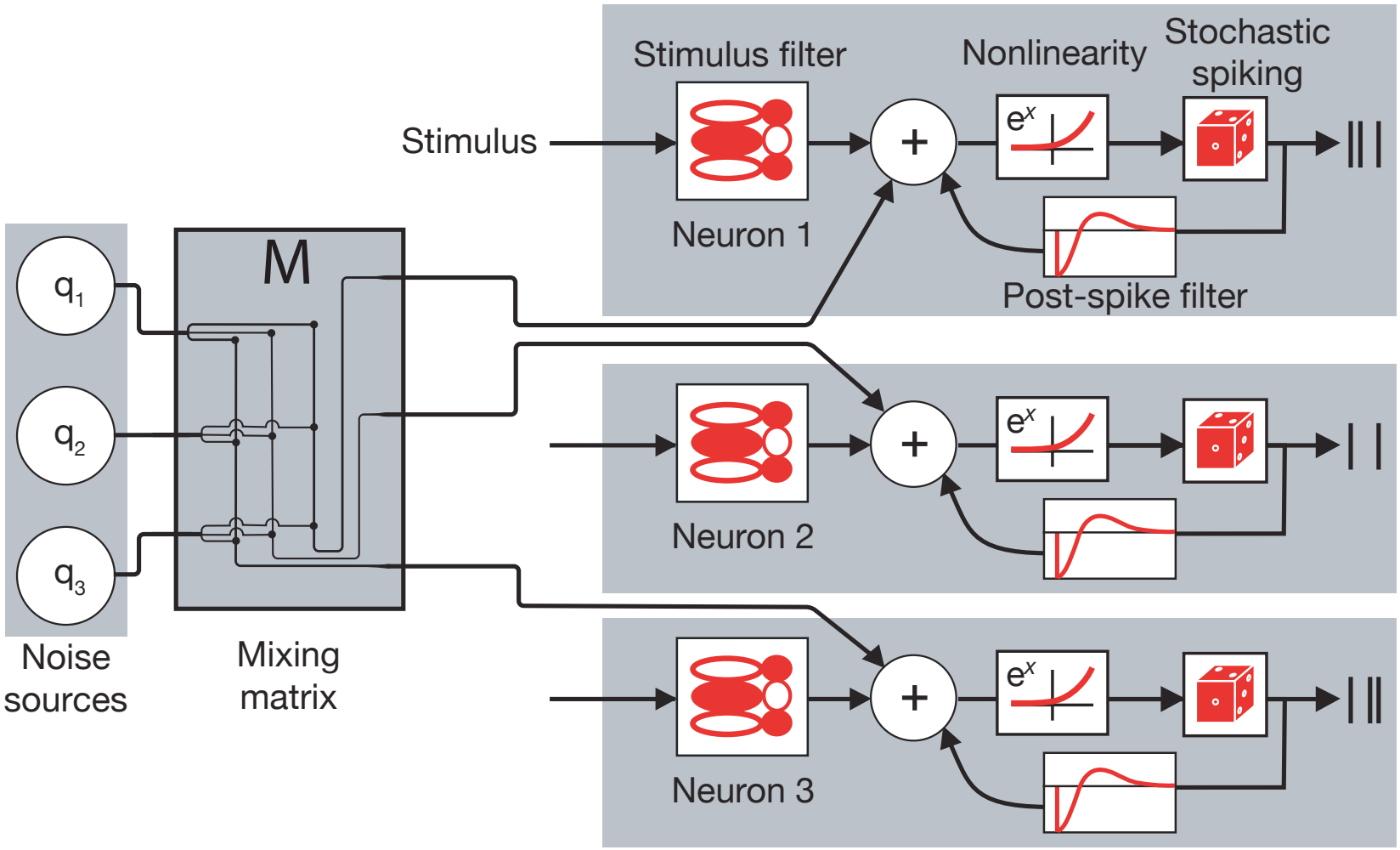


Pillow et al 2008

Beyond GLM

“shared noise” (latent variable) models

Macke et al 2011
Vidne et al 2012
Archer et al 2014

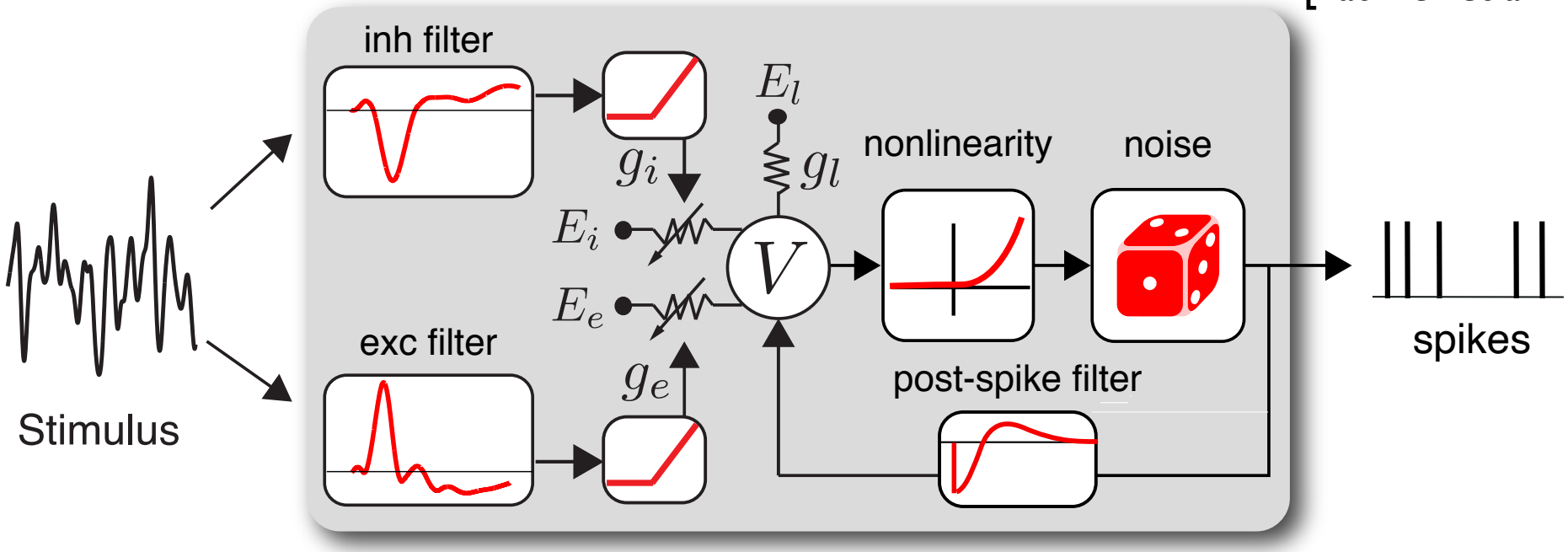


$$\lambda_t^i = \exp(\mu^i + \mathbf{k}^i \cdot \mathbf{x}_t^i + \mathbf{h}^i \cdot \mathbf{y}_t^i + \mathbf{M} \cdot q_t)$$

shared noise

extending GLM to conductance-based model

[Latimer et al 2019]



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

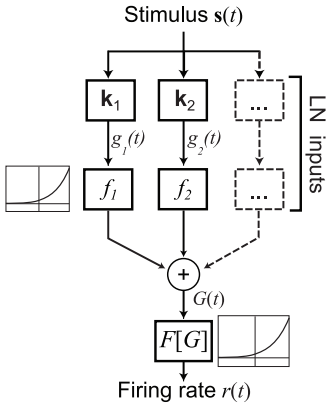
$$\frac{dV}{dt} = g_l(E_l - V) + g_e(E_e - V) + g_i(E_i - V)$$

spike rate

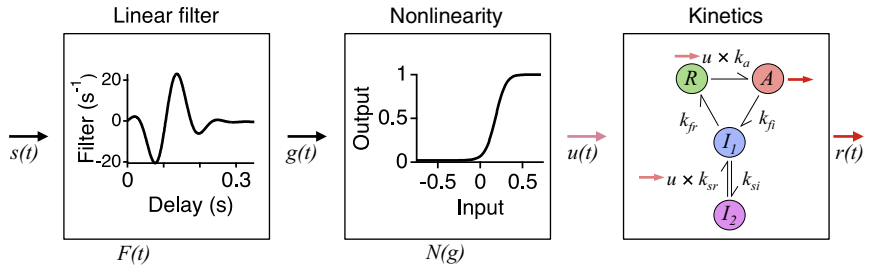
$$\lambda(t) = f(V(t))$$

- shunting inhibition
- adaptive changes in dynamics

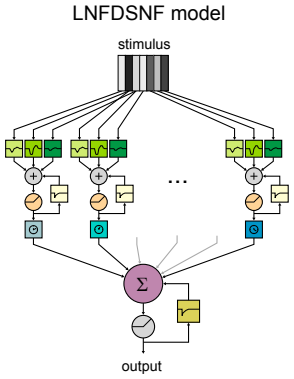
many other biophysically oriented extensions



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



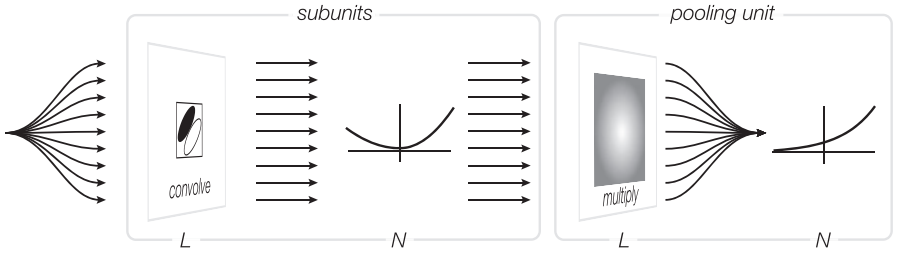
Linear-Nonlinear-Kinetics (LNK)
[Ozuysal & Baccus 2014]



LNFDSNF model

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

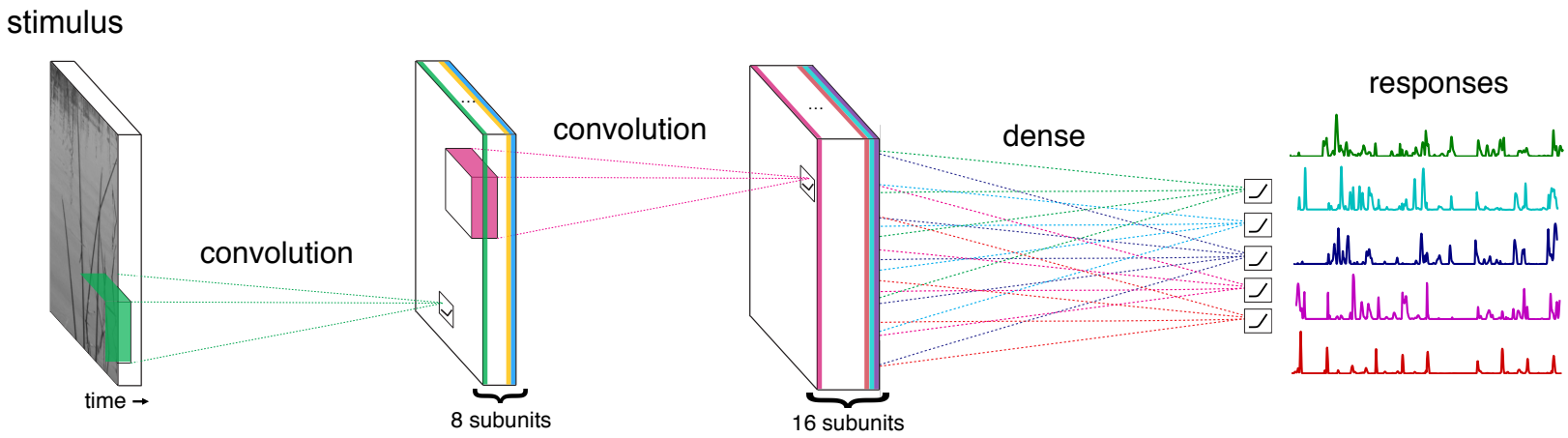
[Real, Asari, Gollisch & Meister 2017]



convolutional subunit model

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

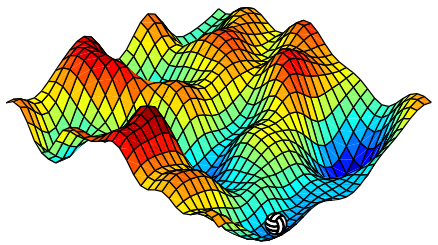
deep learning / deep neural networks (DNNs)



[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 a bunch of tricks (batches, noise, SGD, dropout, ….)
- do NOT worry about local maxima!



Summary

- GLM: 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)

Big Picture

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