(Towards)

ACTIVE MEASUREMENT FOR NEUROSCIENCE

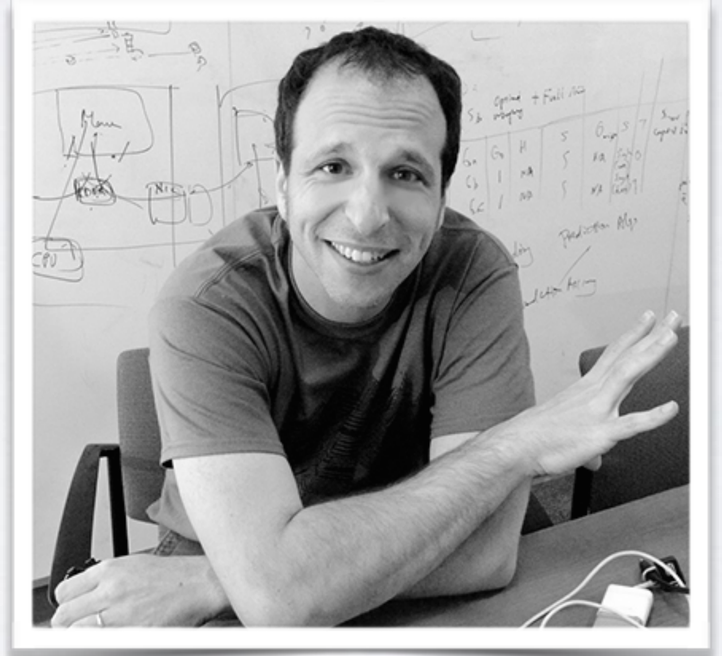
LCCC Focus Period on Large-Scale and Distributed Optimization
June 2017

Ross Boczar
PhD Student
boczar@berkeley.edu

Berkeley
Center for
Computational
Imaging

riselab
UC Berkeley

Eric
Jonas



Ben
Recht

. . .

THIS TALK

- Problem Statement
- Motivating Science
- Motivating Works
- PyWren: A Shameless Plug
- Some Things to Try

PROBLEM STATEMENT

PROBLEM STATEMENT

- **March 2017:**

PROBLEM STATEMENT

- **March 2017:**

I have to give a talk in 3 months.

lateral view
frontal view
dorsal view
0.000 s
dF/F
1
0
100 µm

lateral view
frontal view
dorsal view
0.000 s
dF/F
1
0
100 µm

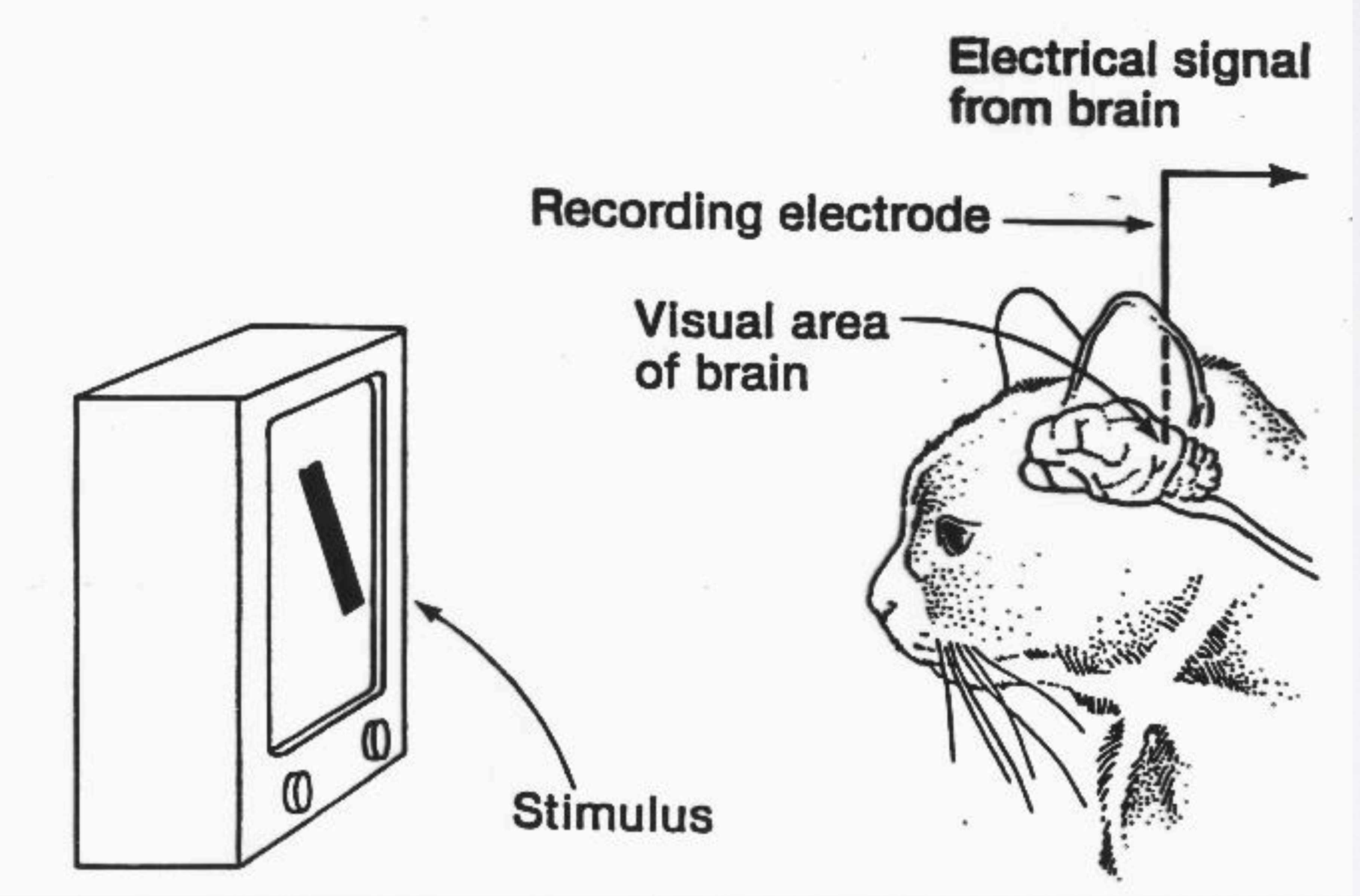
TWO MOTIVATORS

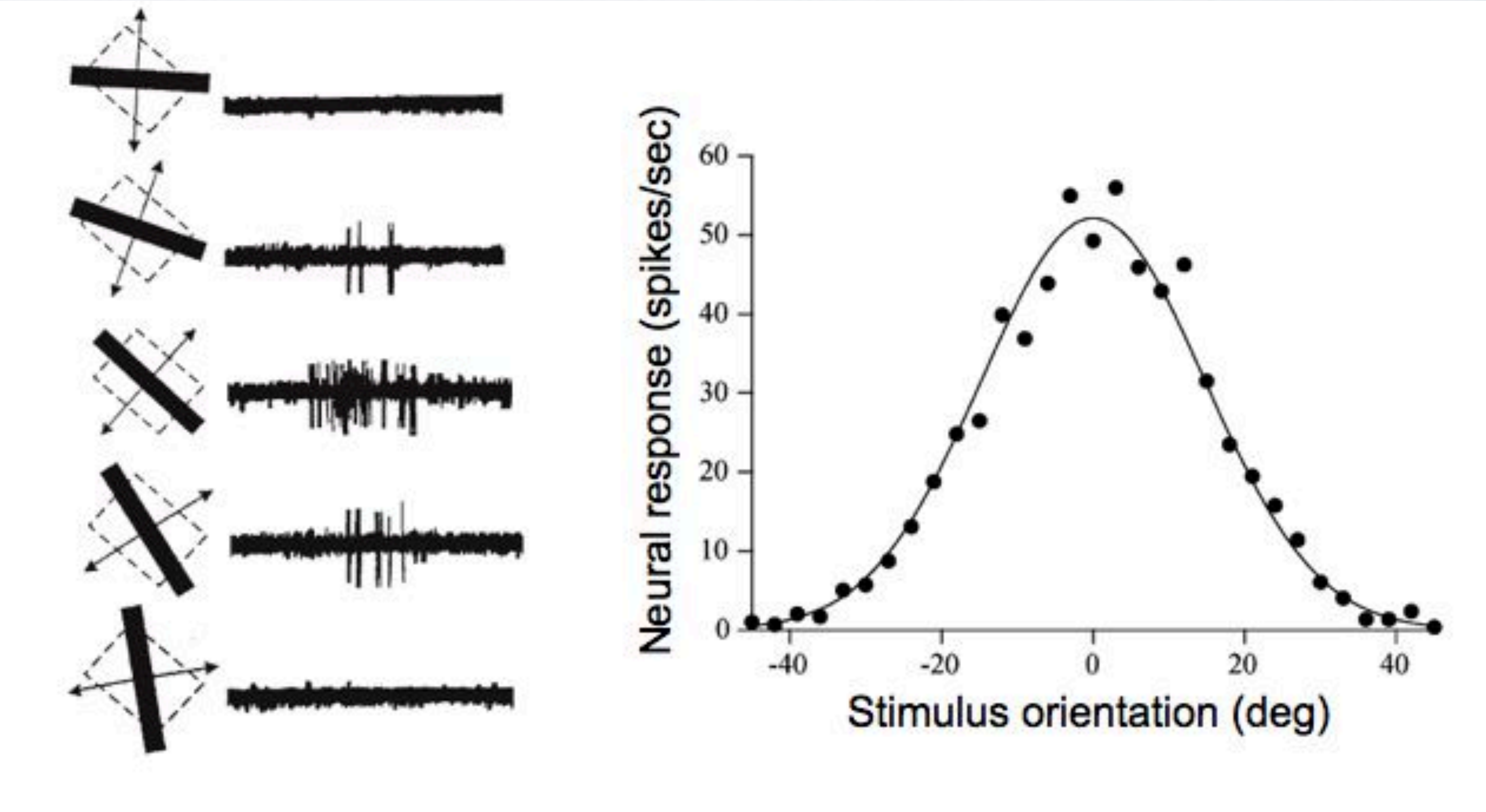
- We have a ton of cells, and finite experimental time! (can record from an organism for a very short period of time)
- We now have fine-grained control over the neurons via optogenetics — we can use lasers to turn on and off individual cells or subpopulations of cells

How do we learn as much about the system as quickly as possible?

Start simple:

SINGLE CELL RESPONDING TO VISUAL INPUT





MOTIVATING WORKS

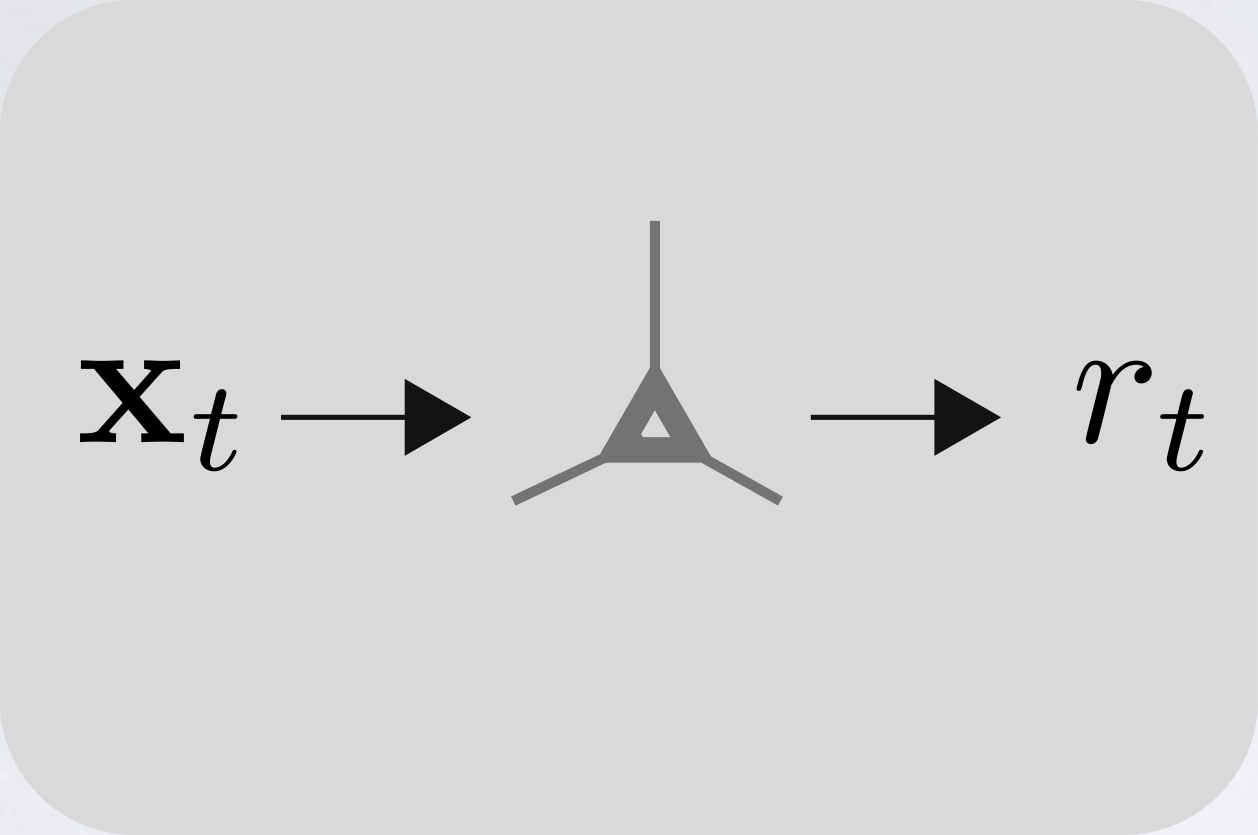
Sequential Optimal Experiment Design for Neurophysiological Experiments
Lewi, Butera, and Paninski 2009

Adaptive Bayesian Methods for Closed-loop Neurophysiology
Pillow and Park 2016

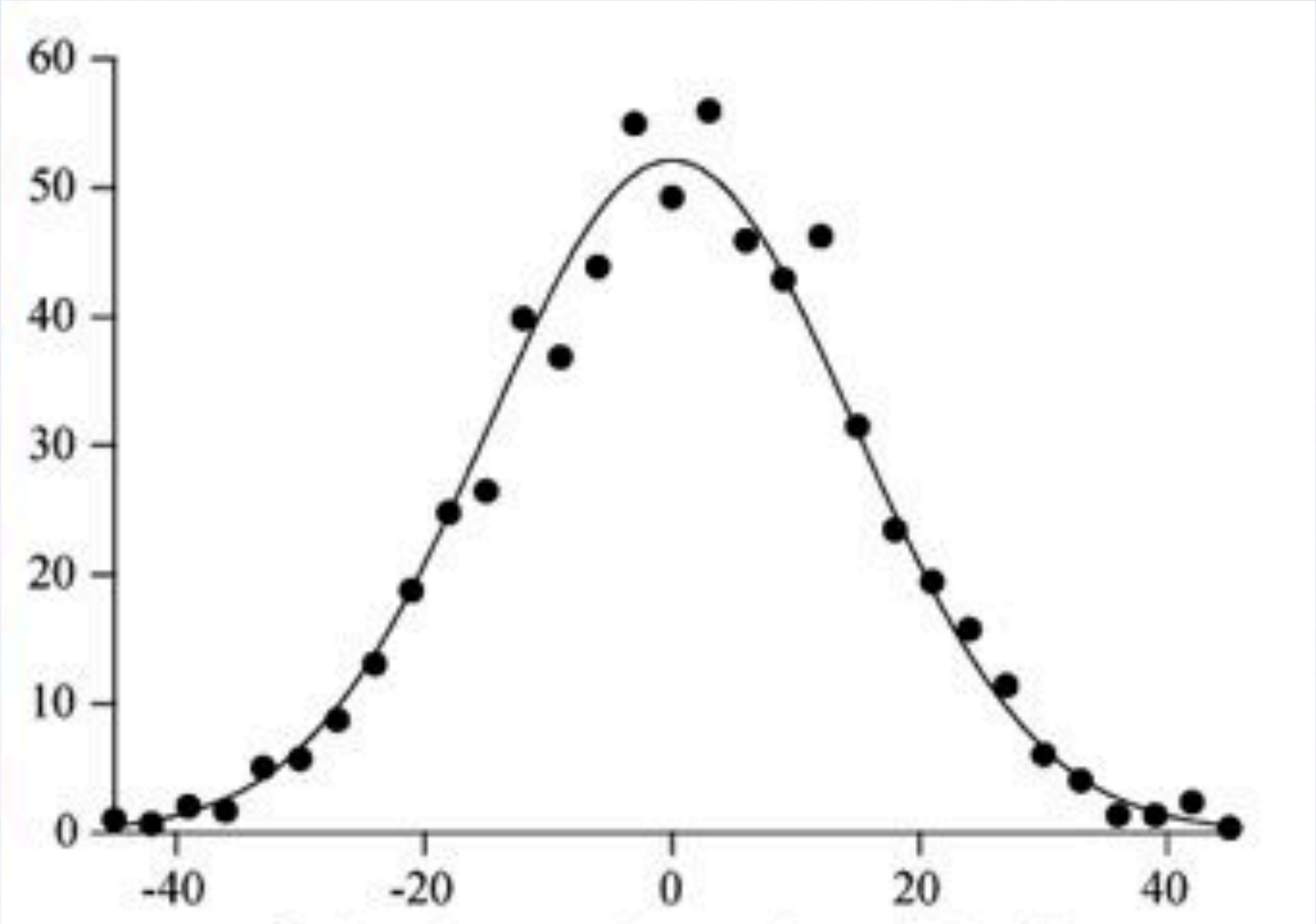
A SIMPLE EXAMPLE

showing the adaptive measurement paradigm

1. present stimulus, observe response



trial t

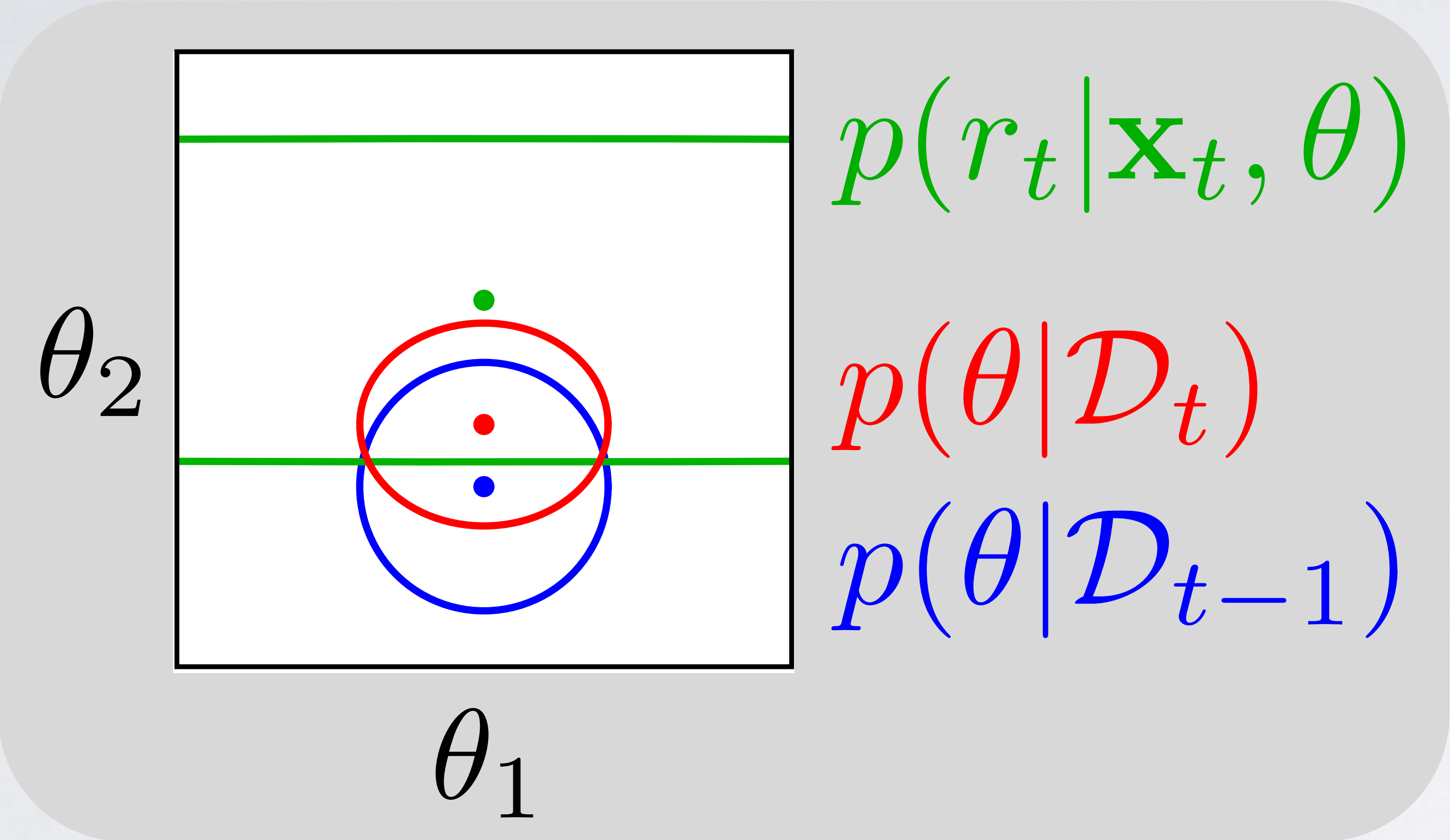


$$f(x;\theta) = b + A\exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)$$

$$\begin{aligned}\lambda &= f(\mathbf{x}) \\ p(r|\mathbf{x}) &= \frac{1}{r!}\lambda^r \mathrm{e}^{-\lambda}\end{aligned}$$

Pillow and Park 2016

2. update posterior



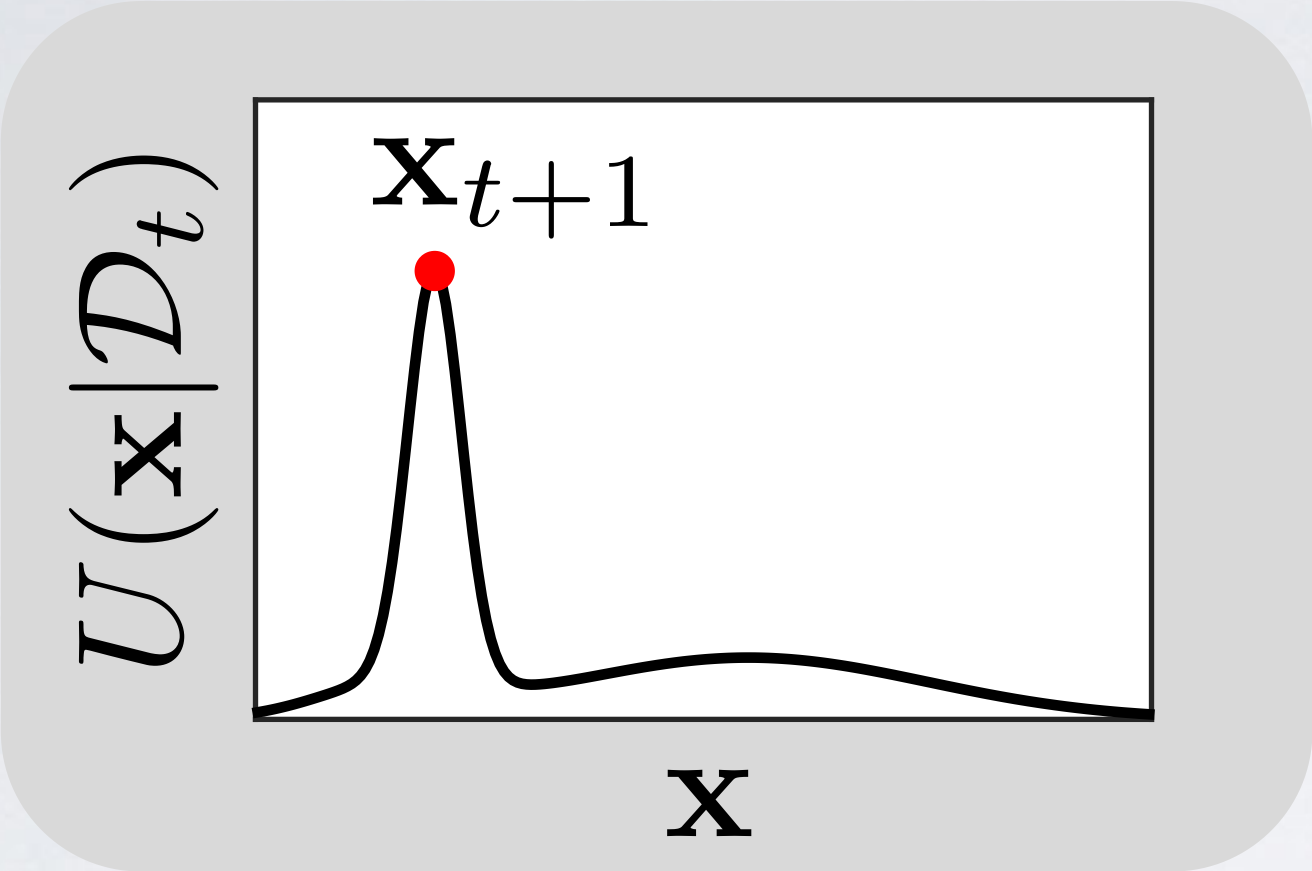
Log-likelihood based on observed responses:

$$\mathcal{L}(\boldsymbol{\lambda}_t|\mathcal{D}_t) = \log p(R_t|\boldsymbol{\lambda}_t) = {R_t}^\top \log \boldsymbol{\lambda}_t - \mathbf{1}^\top \boldsymbol{\lambda}_t,$$

Parameter space small enough to grid in this case (**not typical**)

Pillow and Park 2016

3. maximize expected utility



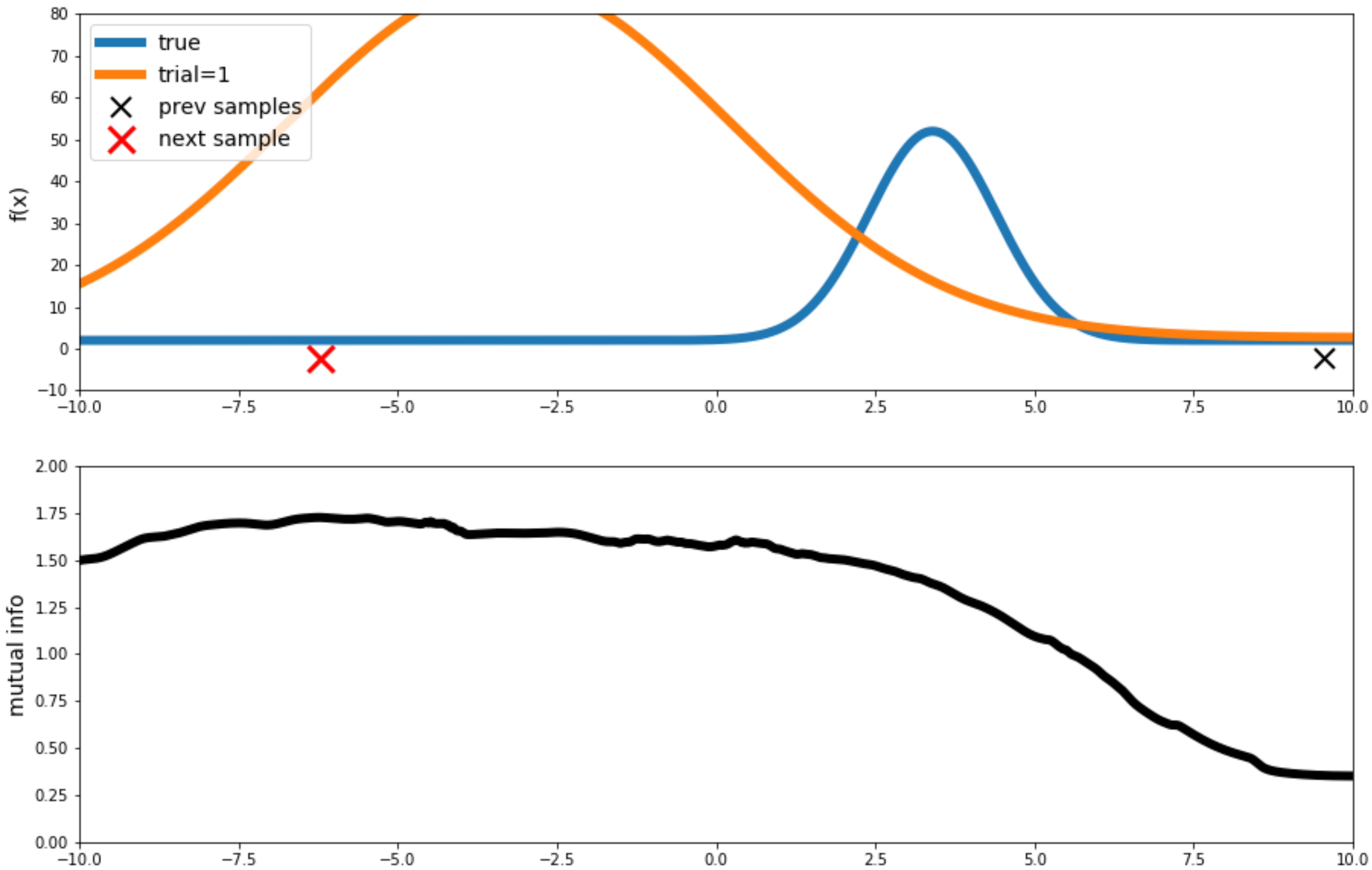
"Infomax learning"

$$U_{\text{infomax}}(\mathbf{x}|\mathcal{D}_t) = \mathbb{E}_{r,\theta}\left[\log \frac{p(\theta|r,\mathbf{x},\mathcal{D}_t)}{p(\theta|\mathcal{D}_t)}\right]$$

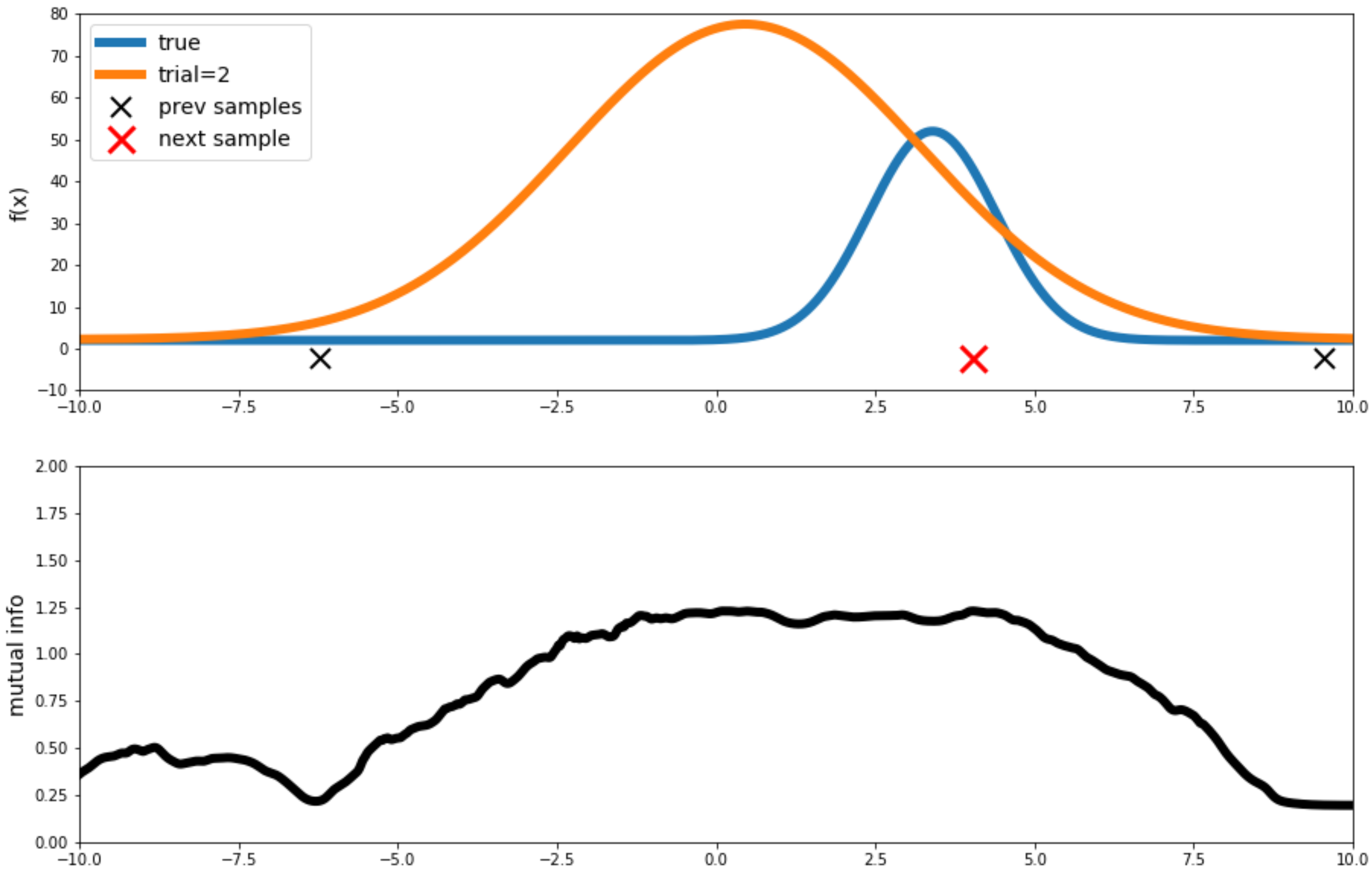
One of multiple criteria to optimize (MMSE, prediction error,…)

Requires integrating over parameter and response spaces, can use MCMC / bag of samples, in this example we can numerically integrate (**not typical**)

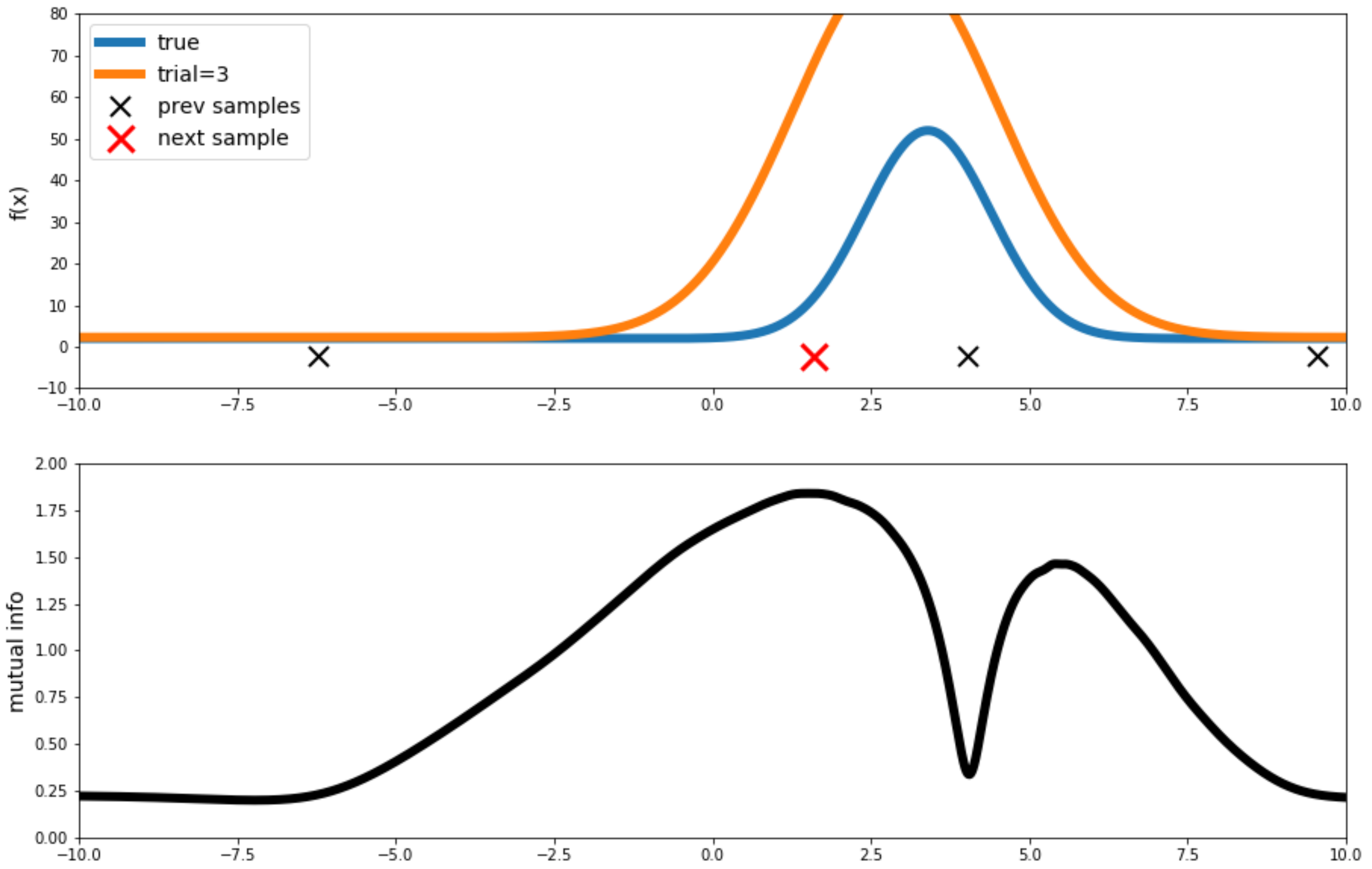
Pillow and Park 2016



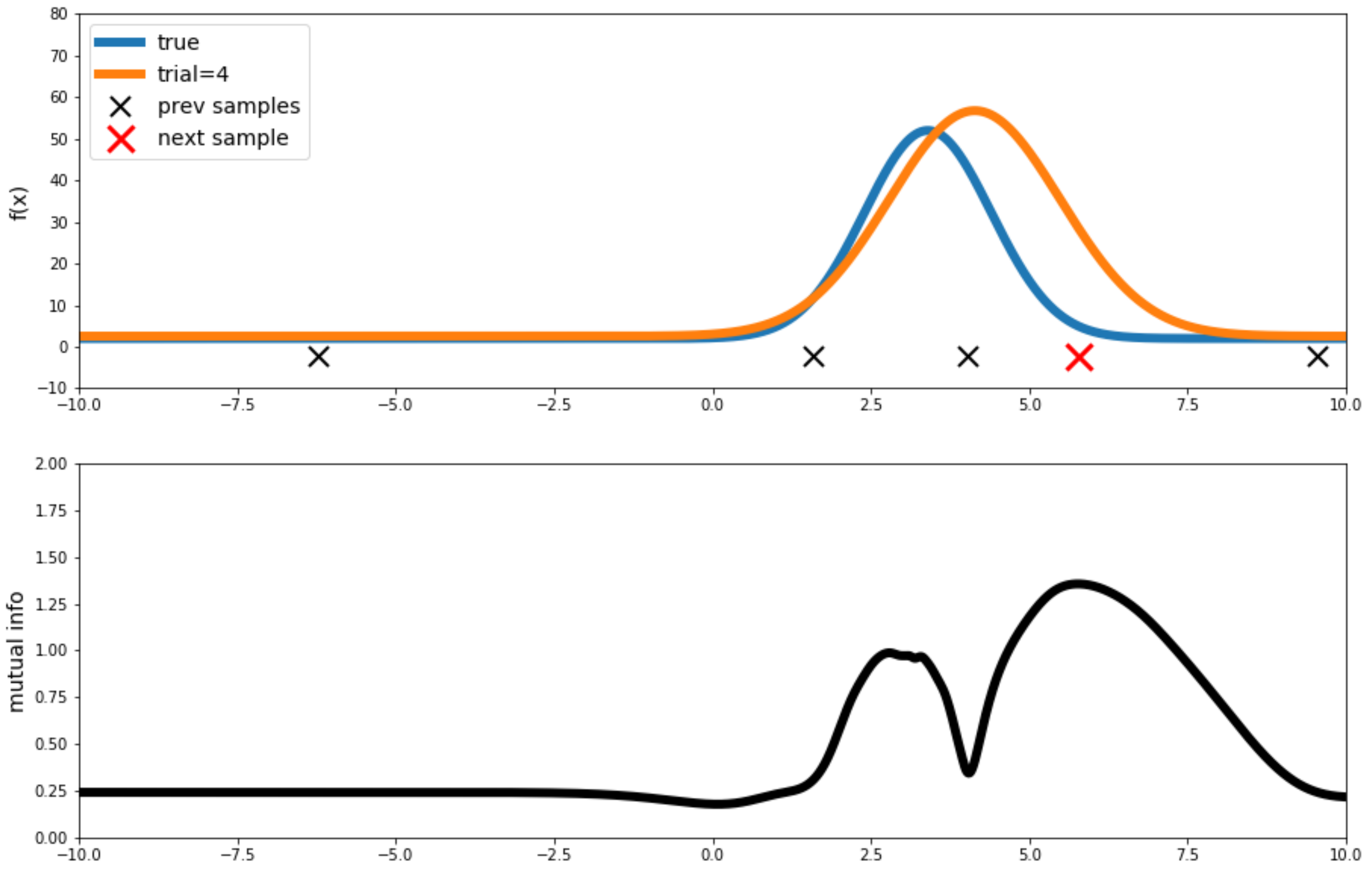
true
trial=1
prev samples
next sample
f(x)
mutual info

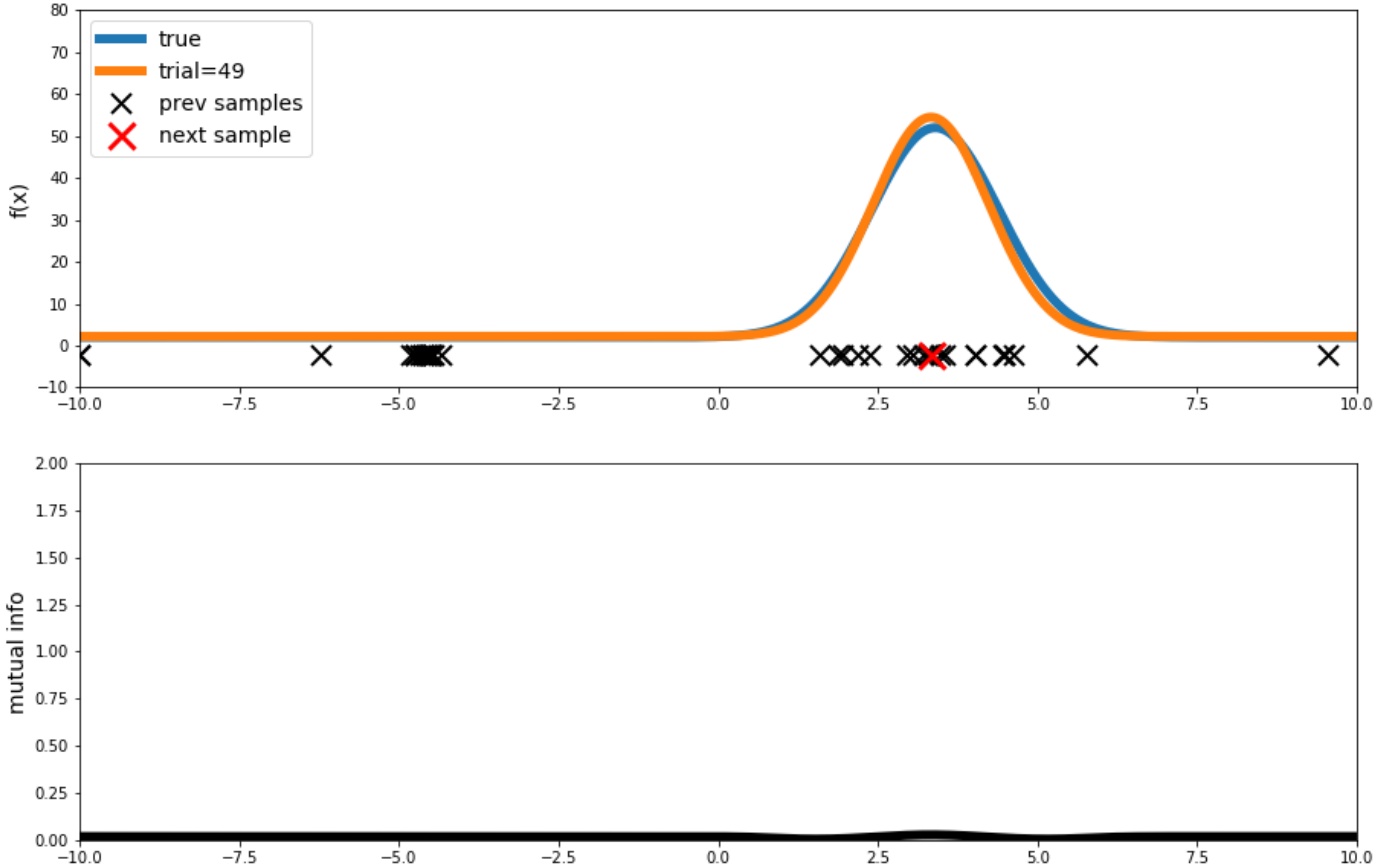


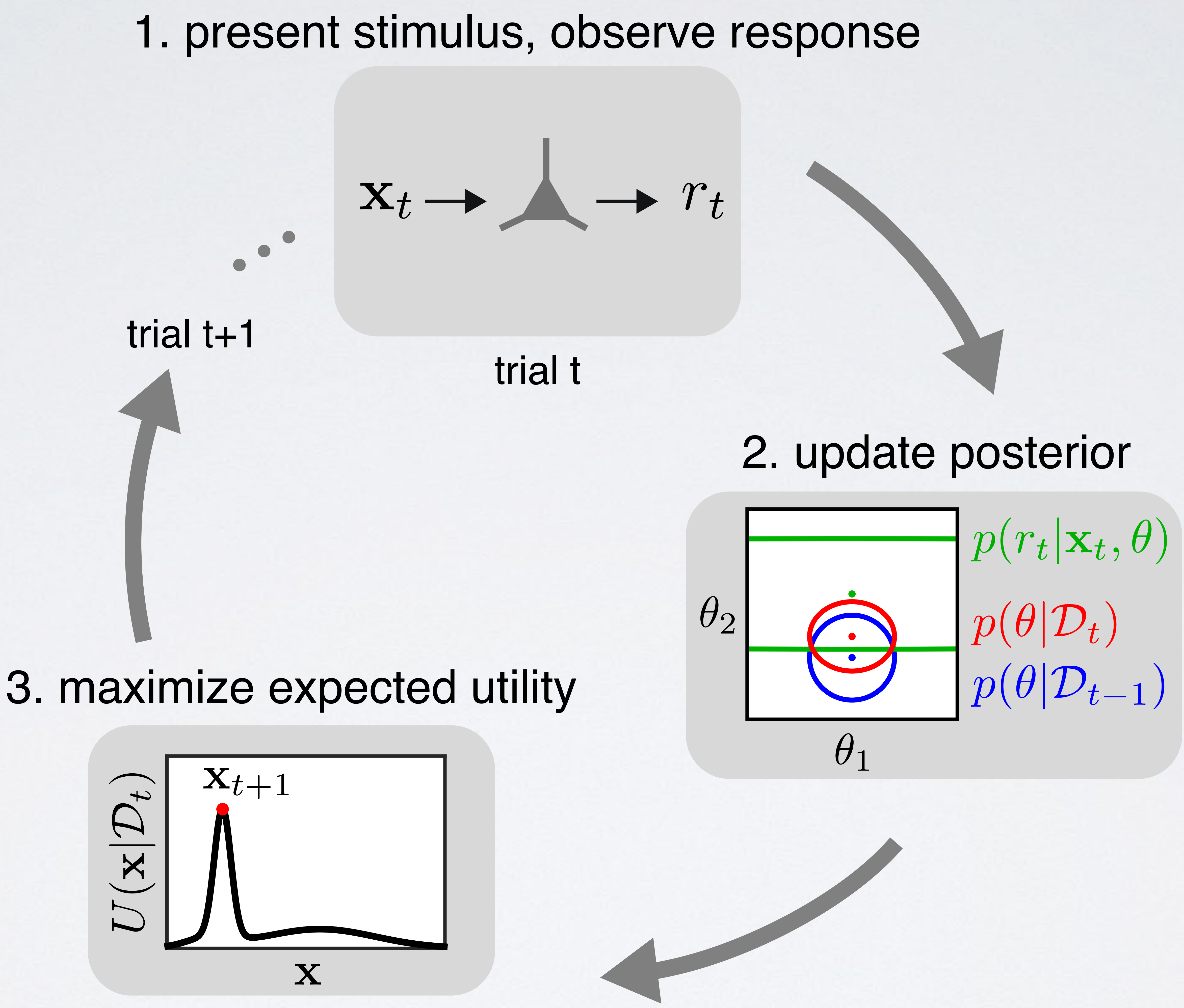
true
trial=2
prev samples
next sample
f(x)
mutual info



true
trial=3
prev samples
next sample
f(x)
80
70
60
50
40
30
20
10
0
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
mutual info
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00

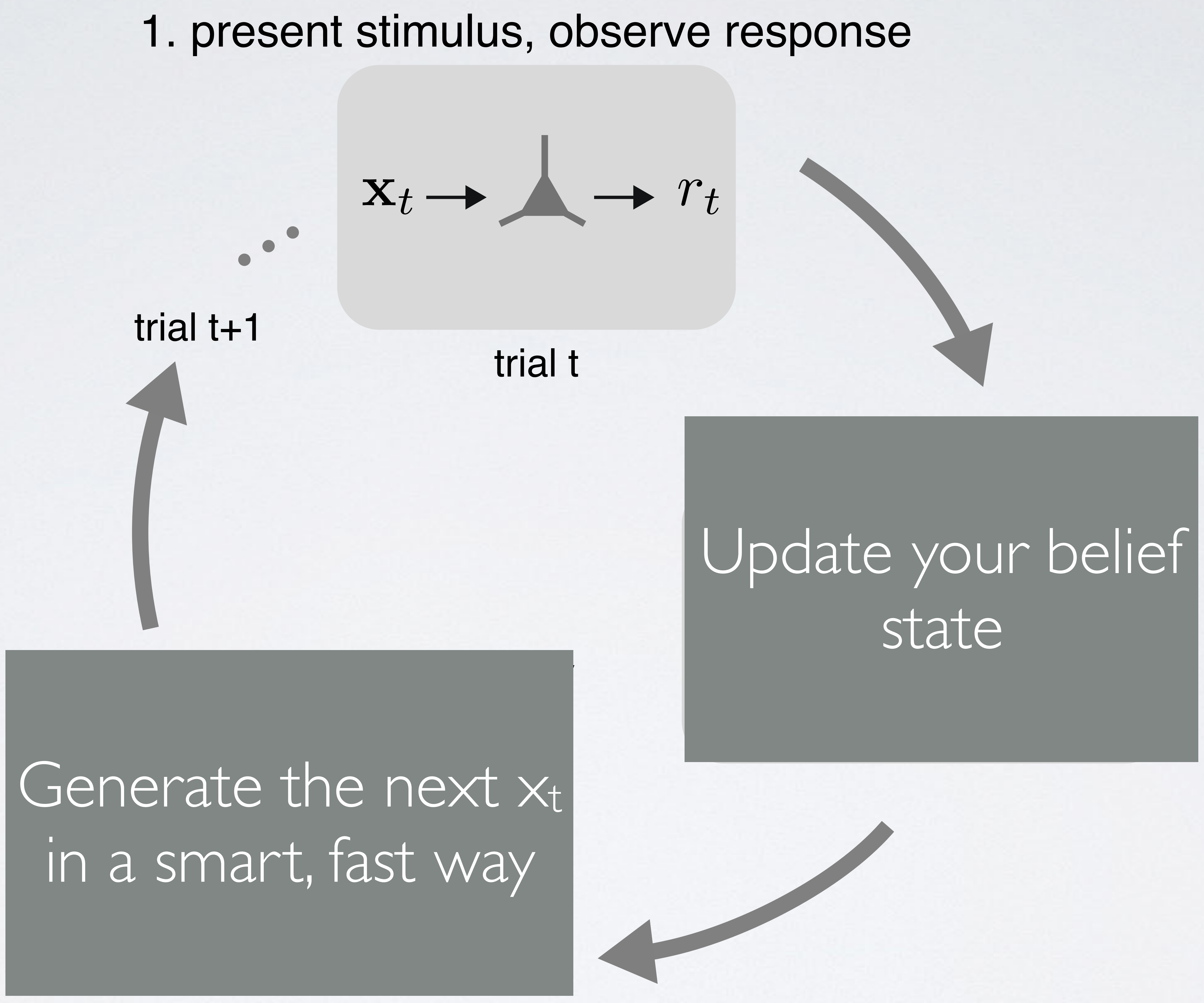

true
trial=4
prev samples
next sample
f(x)
mutual info
80
70
60
50
40
30
20
10
0
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00


true
trial=49
prev samples
next sample
f(x)
mutual info
80
70
60
50
40
30
20
10
0
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00



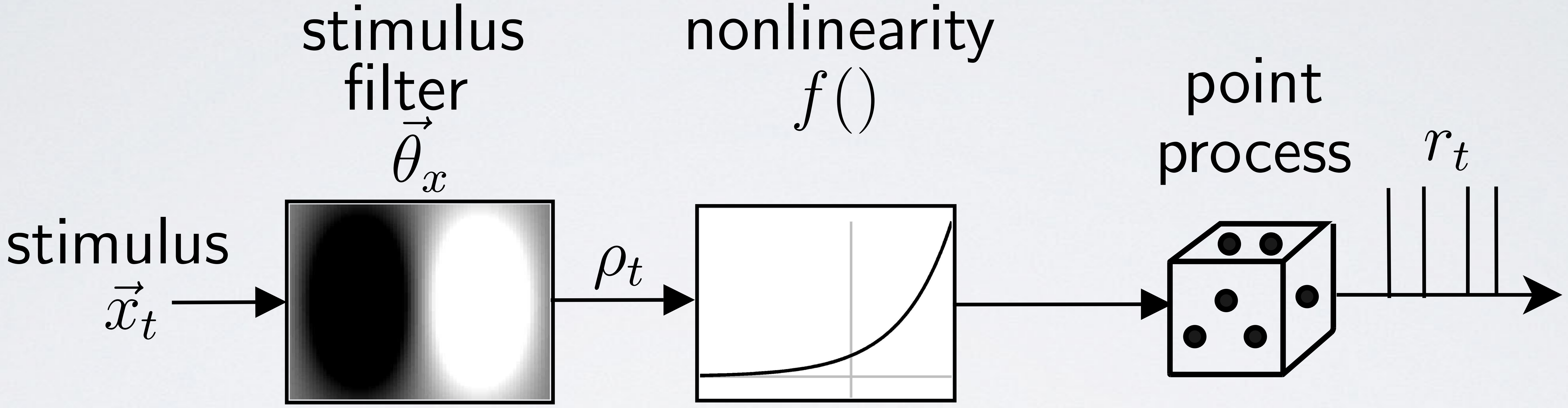
Pillow and Park 2016, Fig. 1

In general:



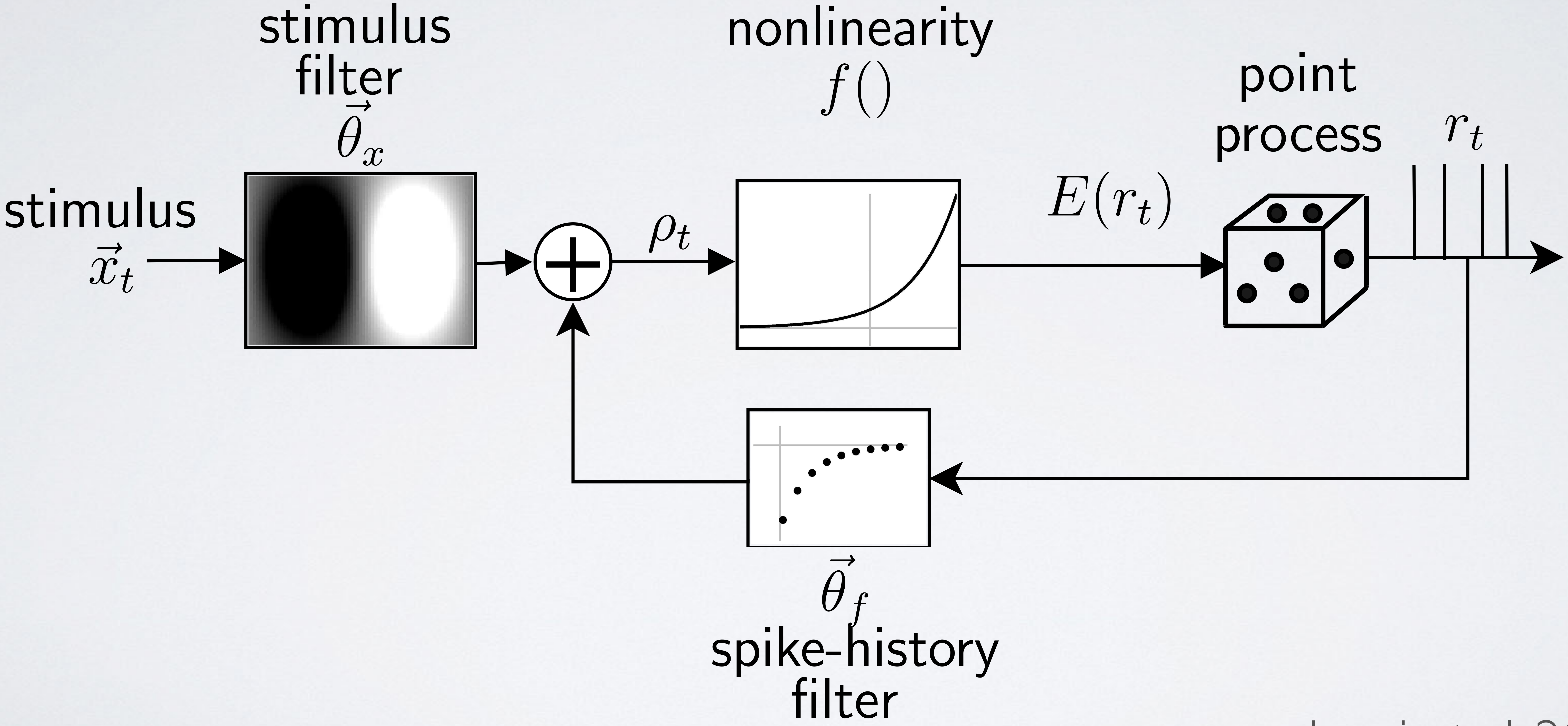
Pillow and Park 2016, Fig. 1

ANOTHER VIEW



Lewi et al. 2009, Fig. 2

ANOTHER VIEW



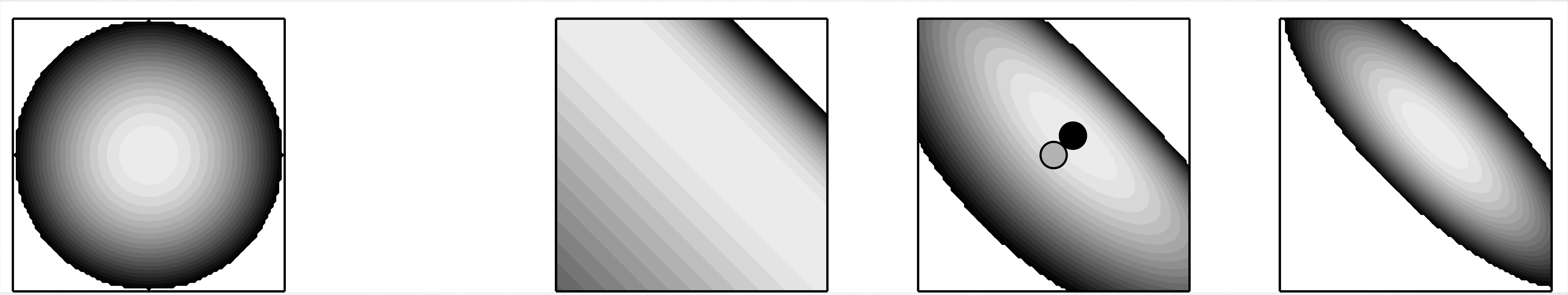
Lewi et al. 2009, Fig. 2

CURRENT APPROACH

- More complicated example: Lewi-09 (visual receptive fields)

- Laplace approximation for belief state (2nd order statistics) gives a compact representation for the parameter distribution

$$logp(\vec{\theta}|\vec{\mu}_{t-1}, C_{t-1}) \quad + \quad logp(r_t|\vec{s}_t, \vec{\theta}) \ = logp(\vec{\theta}|\vec{s}_t, r_t, \vec{\mu}_{t-1}, C_{t-1}) \approx logp(\vec{\theta}|\vec{\mu}_t, C_t)$$



CURRENT APPROACH

- Have to solve high-dimensional non-convex optimization and/or integration to solve for the next x — have to grid or sample based on heuristics (i.i.d. is bad!)

- **Drawbacks:** Curse of dimensionality, problems with EM / MCMC sampling, certain ops can get computationally (and financially!) expensive, would like to deal with more complicated models, …

CURRENT APPROACH

- Would like a lot of cores **now**, suitable for prototyping and exploration for these computationally intensive tasks, many of which are "embarrassingly parallel"

PYWREN:
A POSSIBLE (PARTIAL) PANACEA

PREVIOUSLY, AT COMP IMAGING LUNCH

My background: formerly mostly controls, now mostly ML and optimization

My background: formerly mostly controls, now mostly ML and optimization

Eric: How do you get busy physicists and electrical engineers to give up Matlab?

MATLAB

PyWren

pywren.io

"Most wrens are small and rather inconspicuous, except for their loud and often complex songs."

PYWREN: THE API

```python
import pywren
import numpy as np

def addone(x):
    return x + 1

wrenexec = pywren.default_executor()
xlist = np.arange(10)
futures = wrenexec.map(addone, xlist)

print [f.result() for f in futures]
```

The output is as expected:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

USING “SERVERLESS INFRASTRUCTURE”

ANACONDA®

Powered by Continuum Analytics

+

AWS Lambda

Run code without thinking about servers.
Pay for only the compute time you consume.

Get started with AWS Lambda

(Leptotyphlops carlae)

Want our runtime to include

NumPy

SciPy

Cython

Numba

scikit learn

pillow

Start **1205MB**

conda clean

977 MB

eliminate pkg

946 MB

Delete non-AVX2 MKL

670 MB

strip shared libs

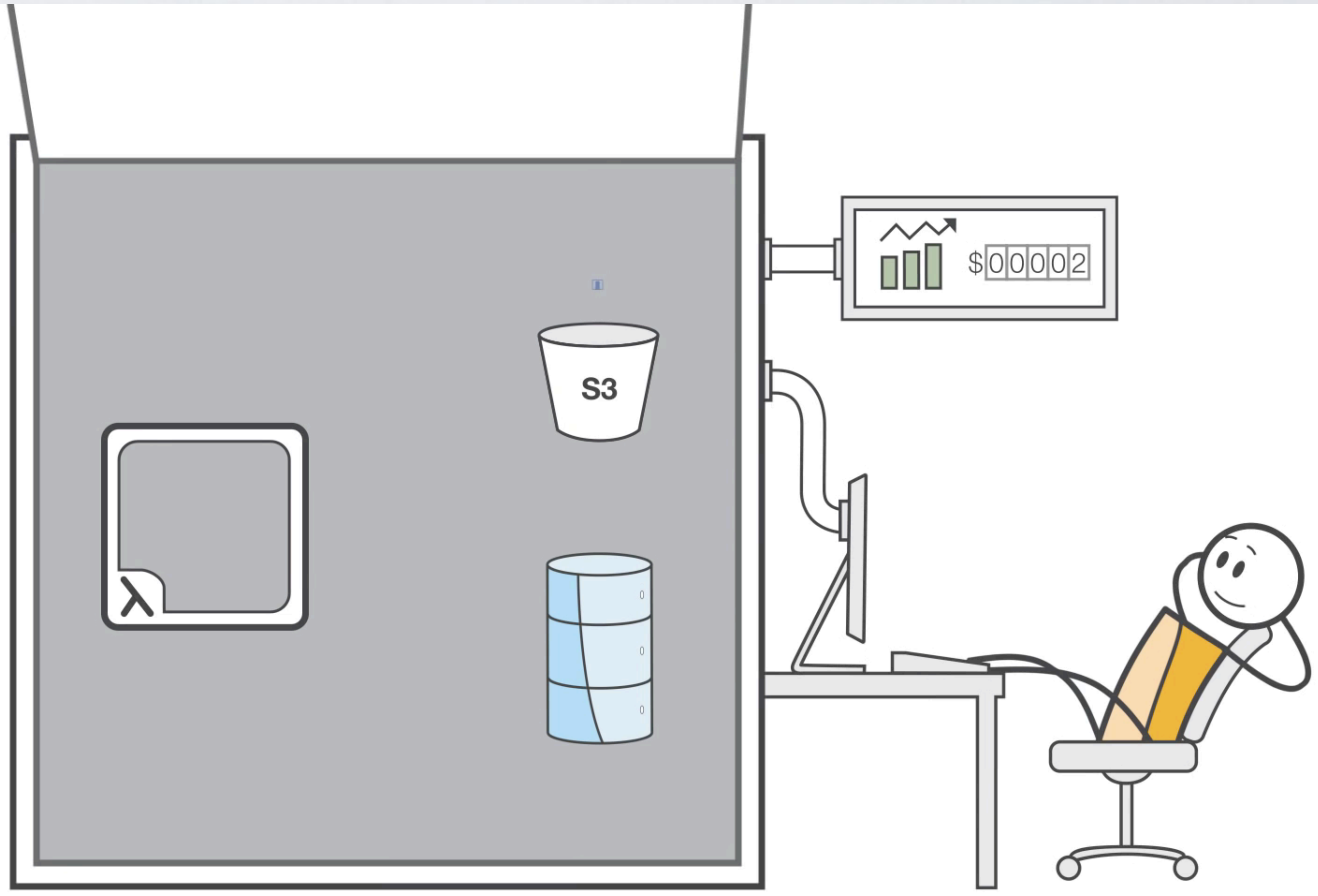
510MB

delete pyc

441MB

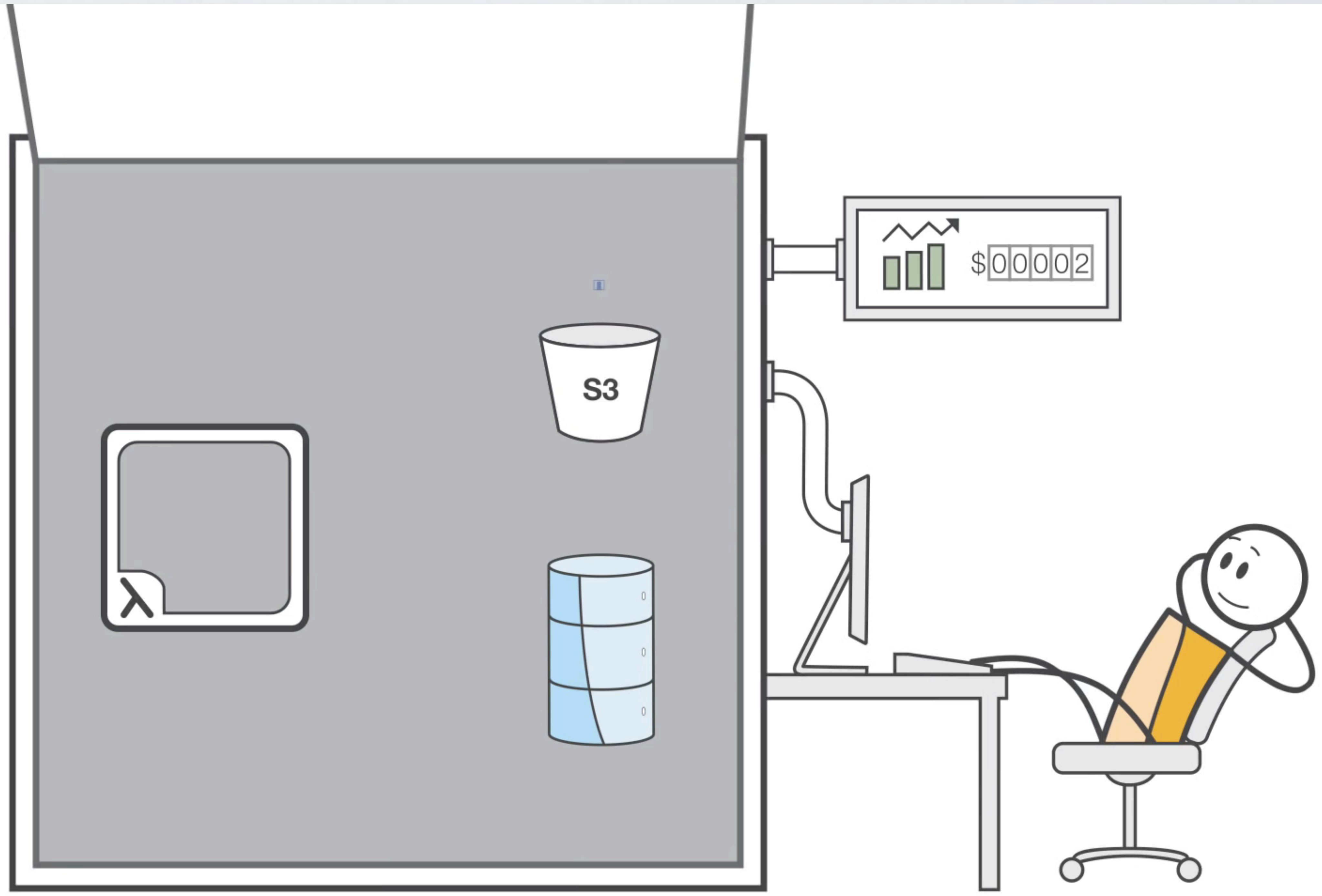
AWS LAMBDA

- 300 seconds
  single-core (AVX2)
- 512 MB in /tmp
- 1.5GB RAM
- Python, Java, Node

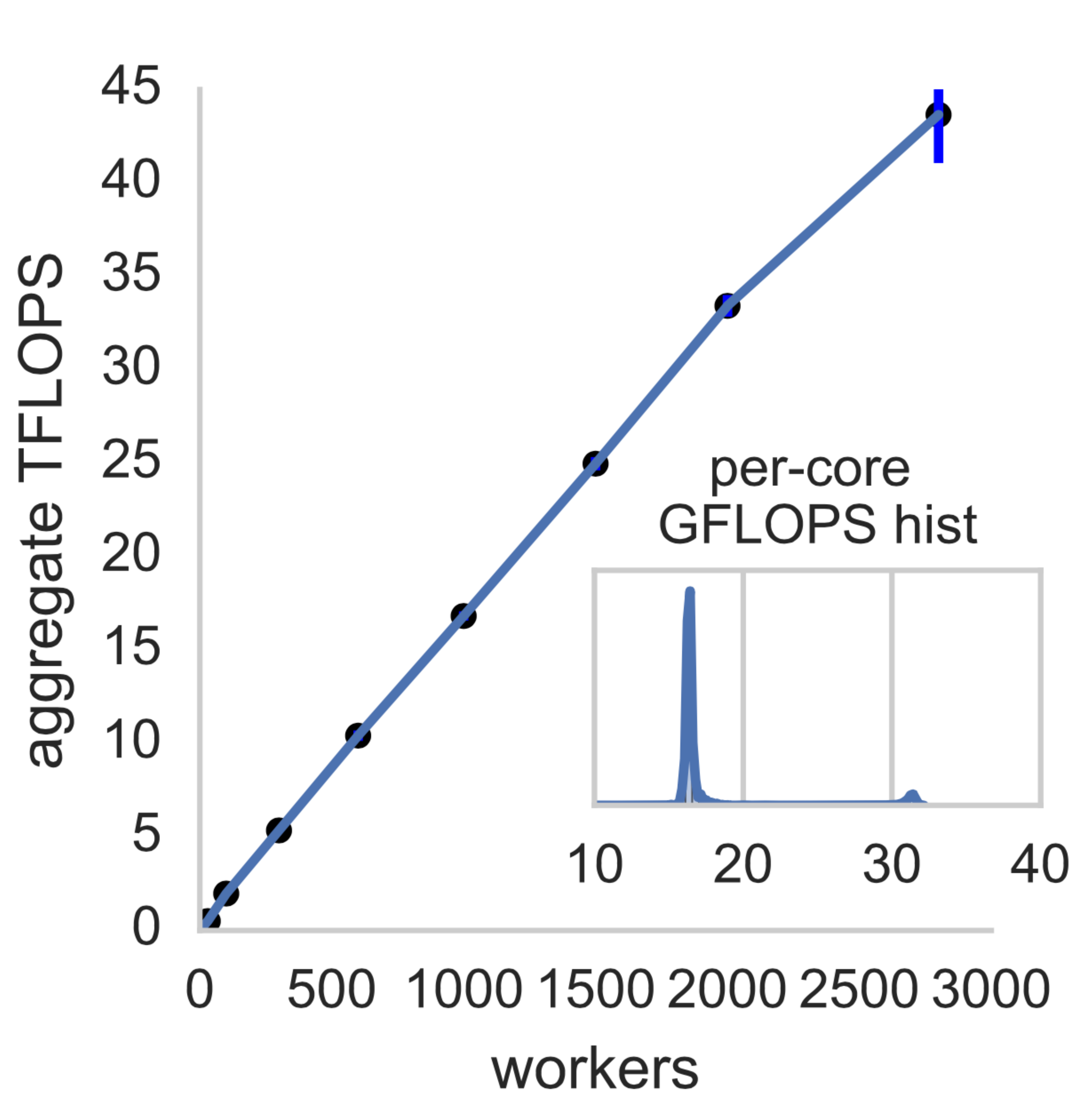


AWS LAMBDA

- 300 seconds single-core (AVX2)
- 512 MB in /tmp
- 1.5GB RAM
- Python, Java, Node

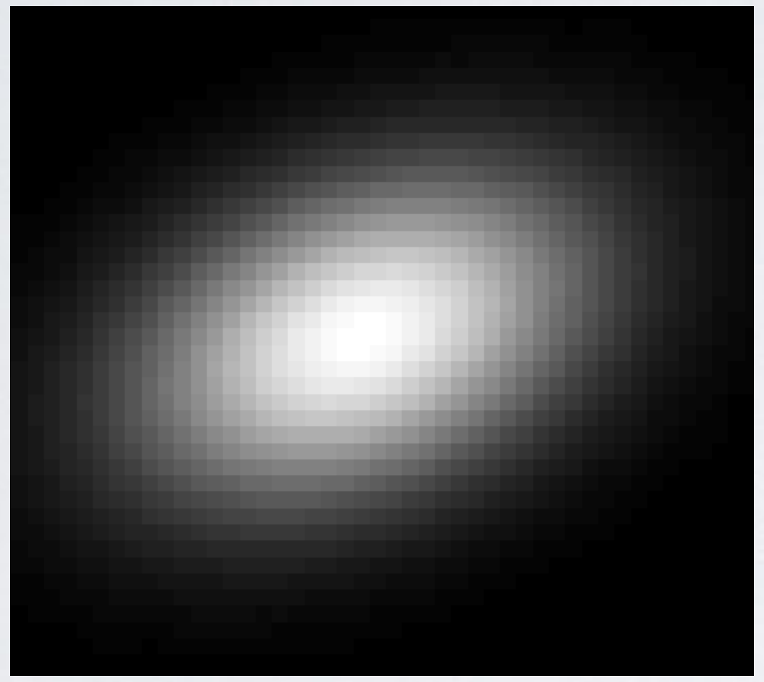


LAMBDA SCALABILITY



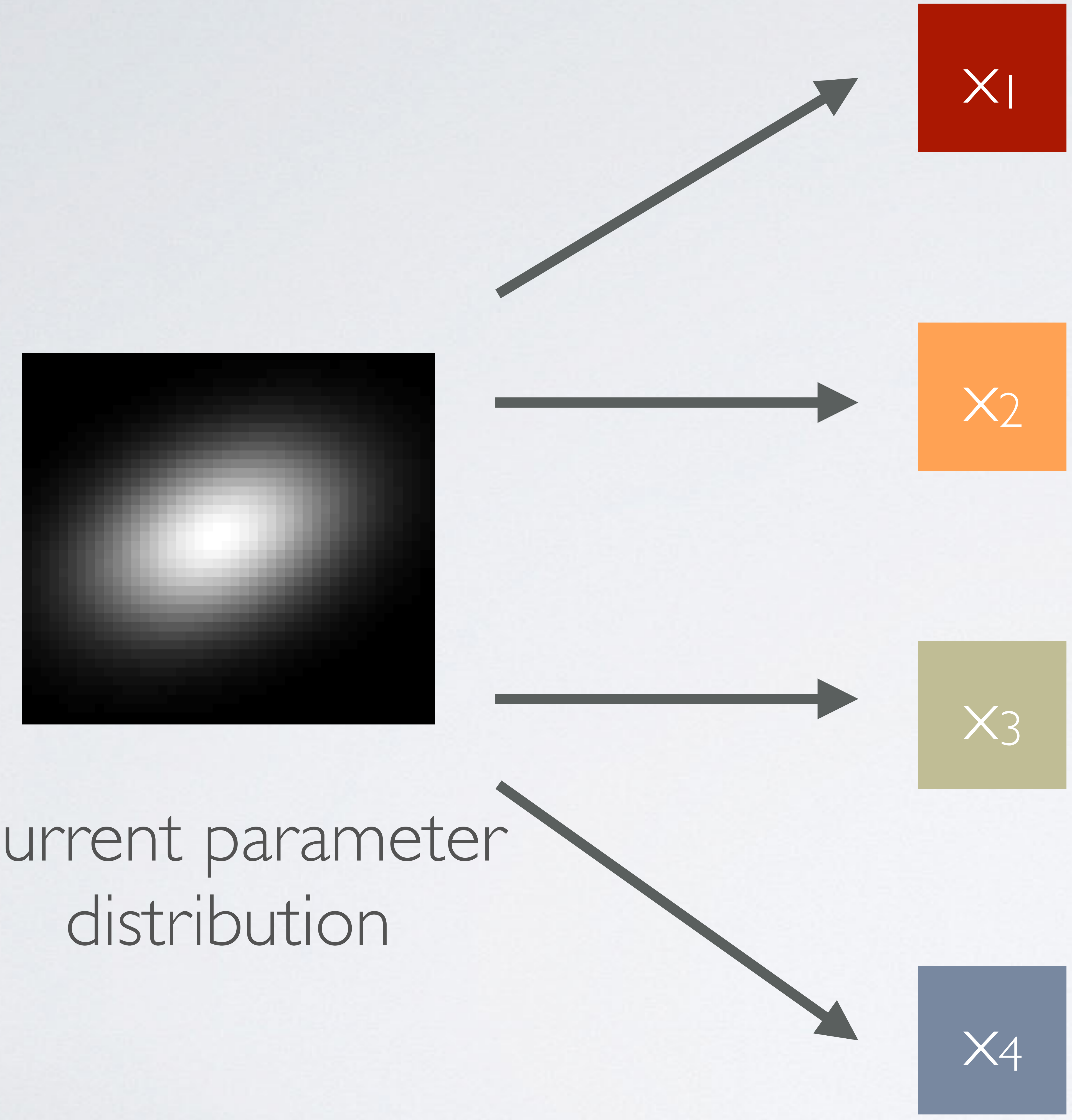
SOME THINGS TO TRY

MPC-INSPIRED SEARCH



Current parameter distribution

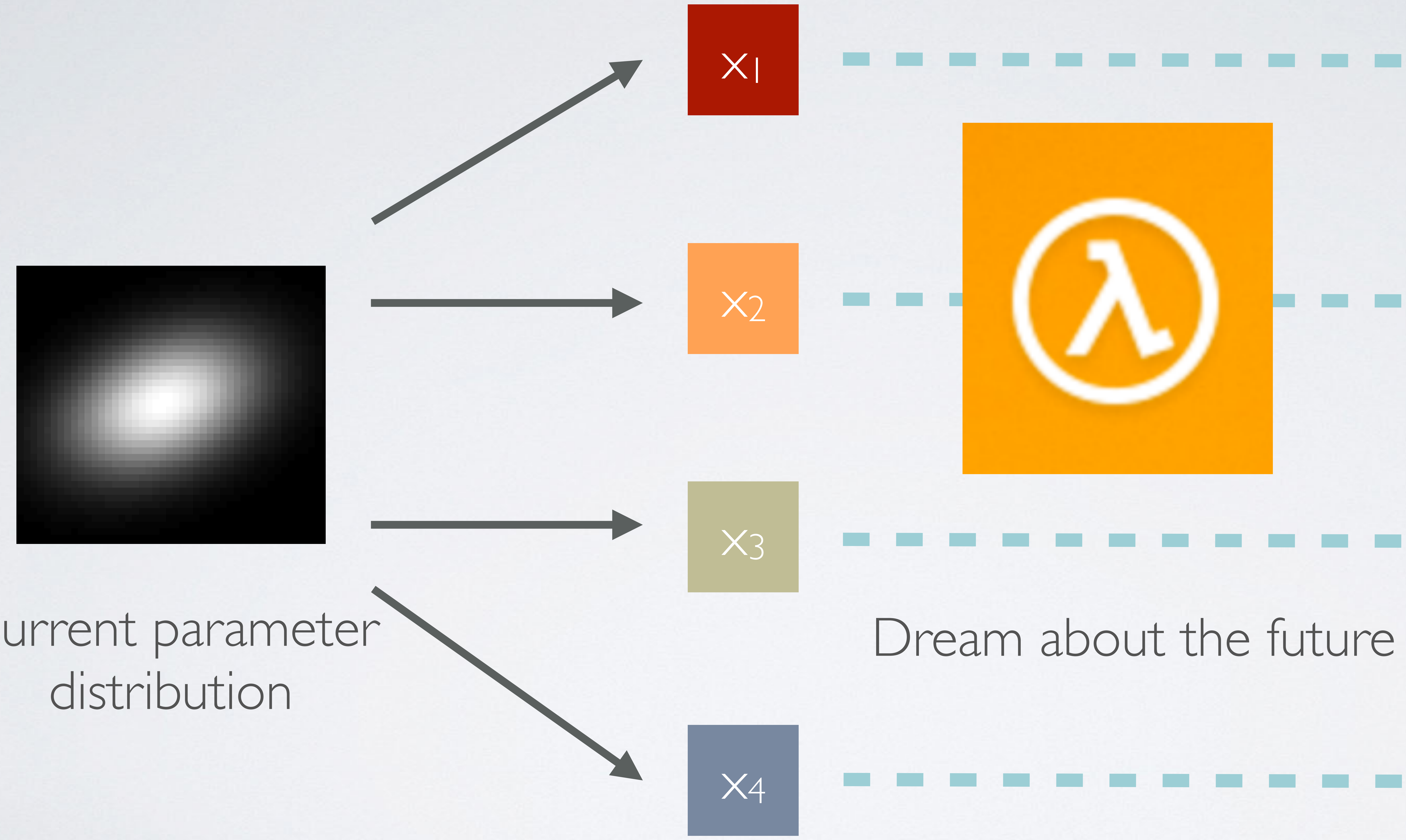
MPC-INSPIRED SEARCH



Current parameter distribution

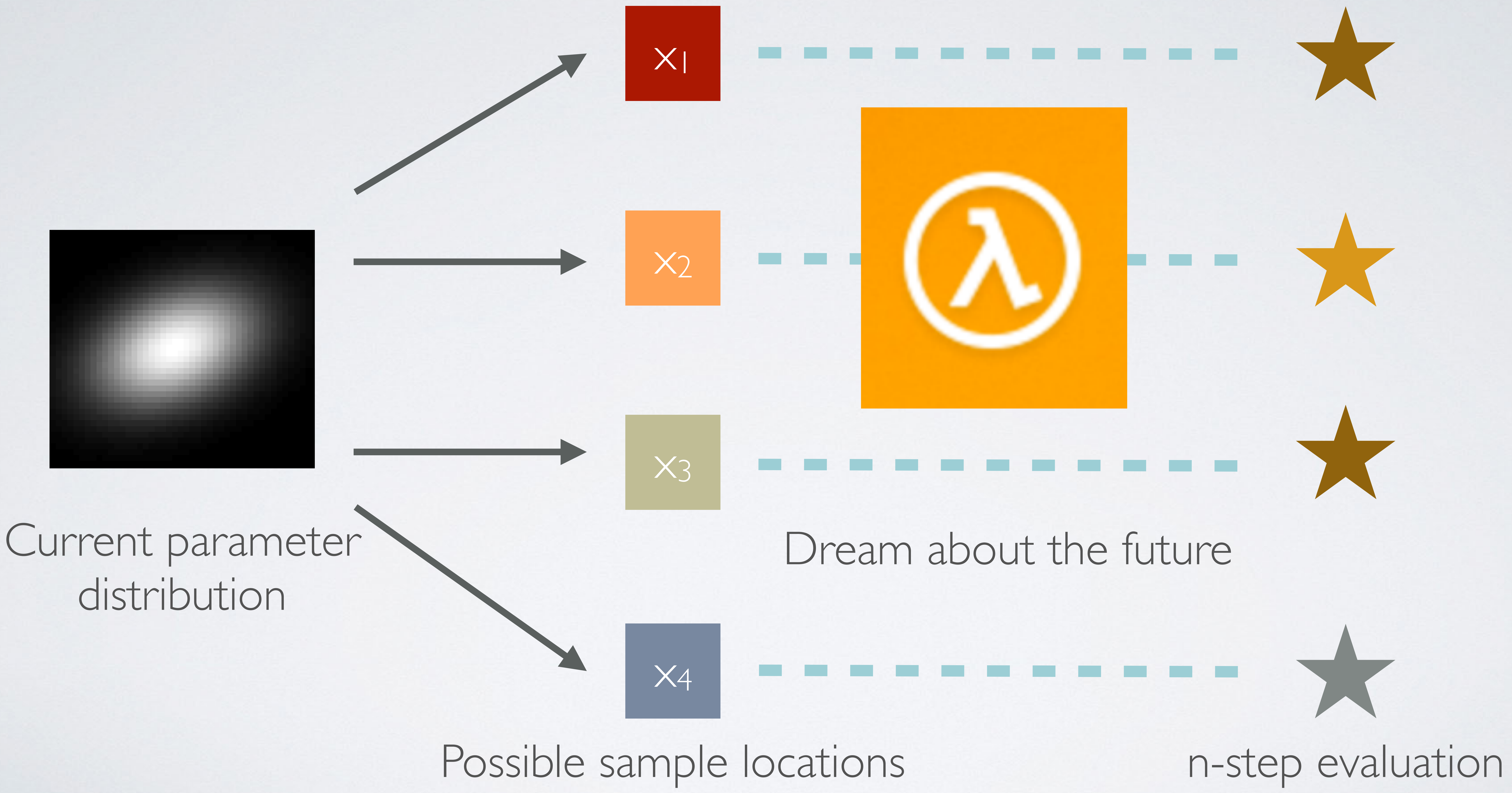
Possible sample locations

MPC-INSPIRED SEARCH



Possible sample locations

MPC-INSPIRED SEARCH



MPC-INSPIRED SEARCH

Current parameter distribution

FUNCTION APPROXIMATION

- Use Lambda services to generate rollouts to learn policies:

$$\pi_1(x_{1:t}, r_{1:t}\,;\,\hat{b}(\theta)_{t-1}) \rightarrow \hat{b}(\theta)_t$$

Belief update function

$$\pi_2(x_{1:t}, r_{1:t}\,;\,\hat{b}(\theta)_t) \rightarrow x_{t+1}$$

Adaptive measurement function

FUNCTION APPROXIMATION

- Fit with ML / adaptive control / reinforcement learning / deep learning technique based on problem

FUNCTION APPROXIMATION

- Fit with ML / adaptive control / reinforcement learning / deep learning technique based on problem
- An aside: DFO works again! http://www.argmin.net/2017/04/03/evolution/

THANKS!

- Here all month :)
- boczar@berkeley.edu
- pywren.io
- http://www.argmin.net/2017/04/03/evolution/