# BAYESOPT IN THE WILD

# BAYESOPT IN THE WILD

OR

QUESTIONS PEOPLE KEPT ASKING ME THAT I DIDN'T HAVE A GOOD ANSWER TO

MATERIAL DEVELOPED WITH

KEVIN SWERSKY

JASPER SNOEK

RYAN ADAMS

HUGO LAROCHELLE

RICHARD WHITCOMB

RAMKI RAMAKRISHNA

JIANQIAO LIU

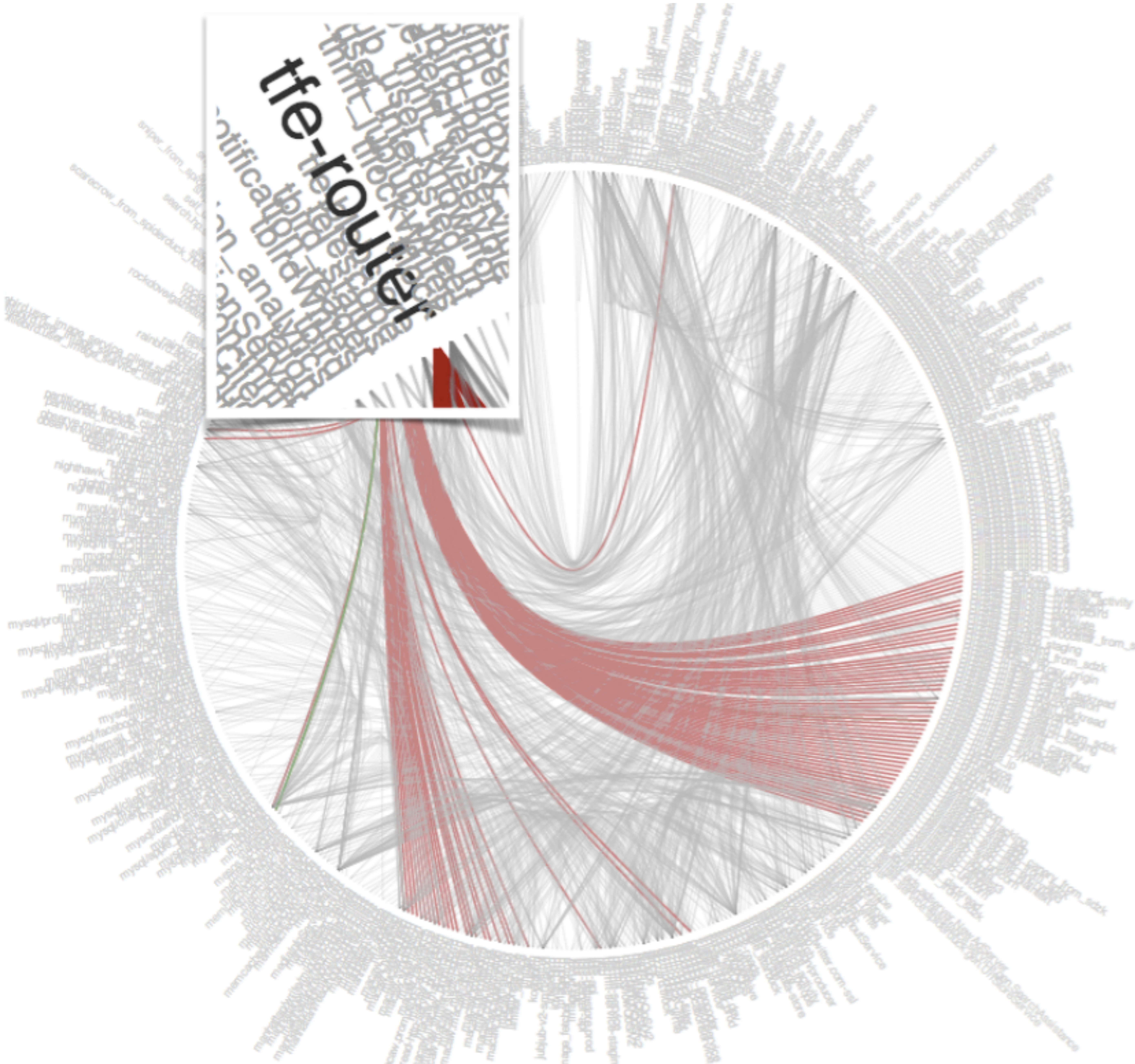
FIRST

## A BAYESOPT SUCCESS STORY

THEN

## CHALLENGES IN INDUSTRY

## TWITTER RUNS ON MICRO-SERVICES



***Source:*** "How we built a metering and chargeback system to incentivize higher resource utilization of Twitter infrastructure", Micheal Arul, Vinu Charanya, *LinuxCon 2016*, Toronto, August 22-24, 2016.

- $O(10^3)$ services
- $O(10^5)$ service instances
- Heterogeneous hardware
- Varying resources

## JAVA VIRTUAL MACHINE (JVM)

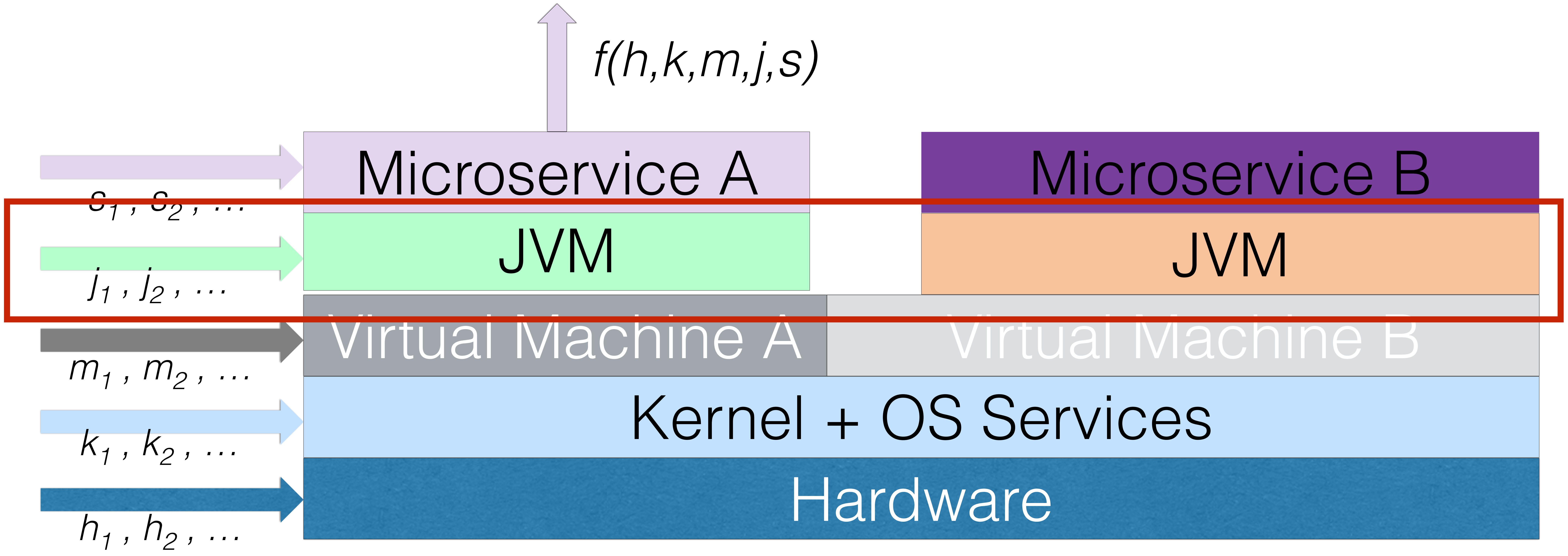
- Java is the language used for Android apps, many cloud applications
- JVM — a piece of software that runs Java programs
- The JVM is the platform of choice for almost all of Twitter

## JVM GARBAGE COLLECTION (GC)

- Garbage Collection (GC) — a strategy for automatic memory management.
- Instead of statically allocating memory, a piece of software reclaims “old” pieces of memory when it can.
- Second result for Google search “JVM garbage collection” are tips for *tuning* the garbage collector for performance.

# A PERFORMANCE STACK AT TWITTER

A simplified view



## TUNING AT THE JVM LAYER

- Hotspot JVM has hundreds of tunable knobs:

```
$ java -XX:+PrintFlagsFinal -version | grep "="
uintx AdaptiveSizePolicyWeight                 = 10                        {product}
uintx AdaptiveSizeThroughPutPolicy             = 0                         {product}
uintx AdaptiveTimeWeight                       = 25                        {product}
bool AdjustConcurrency                         = false                     {product}
bool AggressiveOpts                            = false                     {product}
intx AliasLevel                                = 3                      {C2 product}
bool AlignVector                               = false                  {C2 product}
…

$ java -XX:+PrintFlagsFinal -version | grep "=" | wc –l
757
```

A large variety of parameters:

- performance-sensitivity
- hardware-dependency
- mutual (in)dependency

## PERFORMANCE OPTIMIZATION

Needs to be continuous

Hand-tuning doesn’t scale:

- few parameters handled manually
- time-consuming, labor-intensive, error-prone

Cargo-culted configurations

Upgrades make optimality fleeting

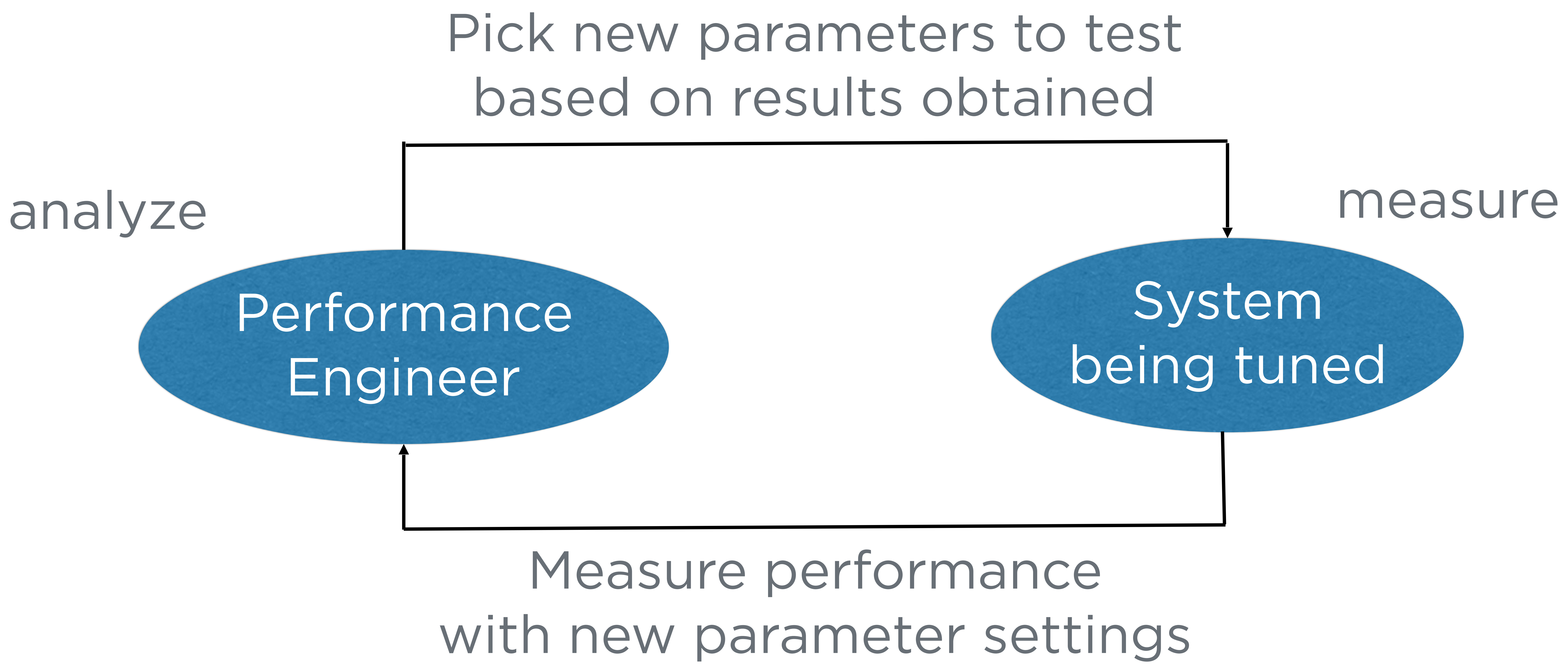
Human experts can only tune ~5 parameters at a time

*Hypothesis:* Most micro-services operate below optimality

*Preview:* **80% improvement on a large service**

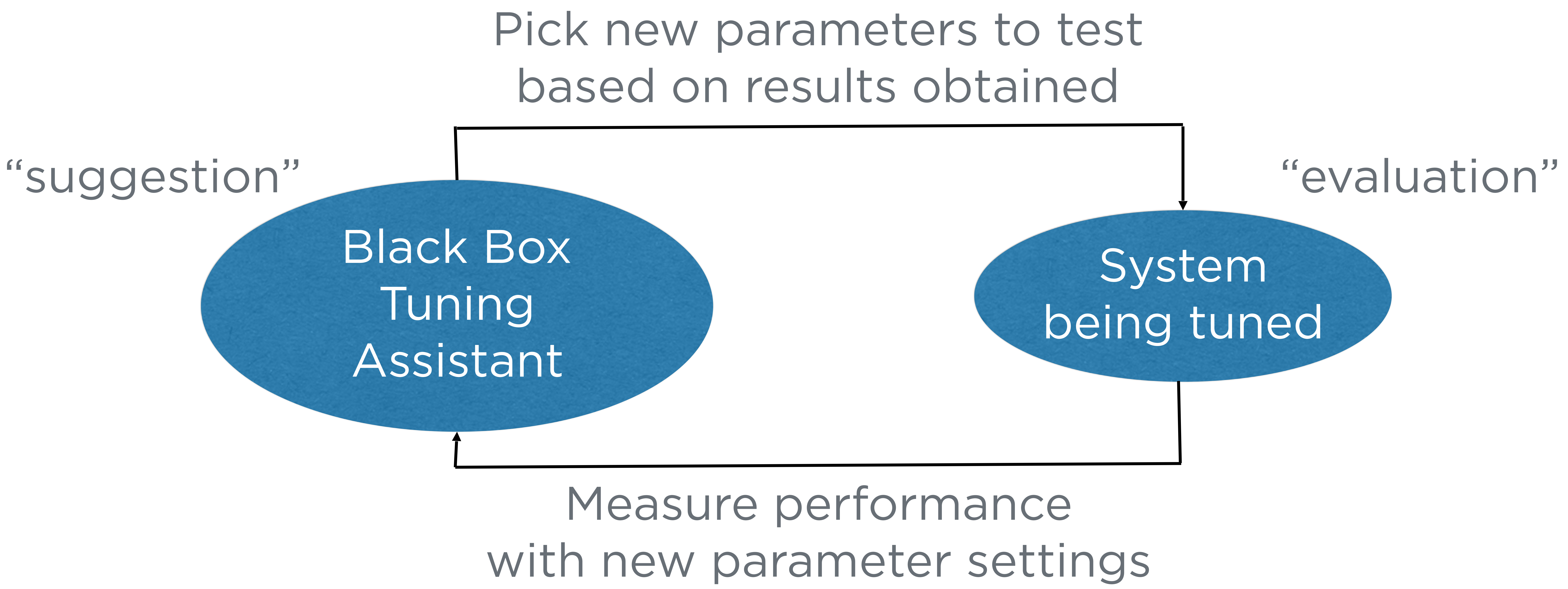
# PERFORMANCE TUNING

## The manual approach


Pick new parameters to test based on results obtained
analyze
Performance Engineer
measure
System being tuned
Measure performance with new parameter settings

## PERFORMANCE TUNING

Using an automation assistant


Pick new parameters to test
based on results obtained
“suggestion”
“evaluation”
Black Box Tuning Assistant
System being tuned
Measure performance
with new parameter settings

## BUILDING AN AUTOTUNER

What should an ideal autotuning system look like?

What do we want an implementation of BayesOpt to look like in practice?

- Easy-to-use
- Minimal coding required by the user
- Support multiple languages

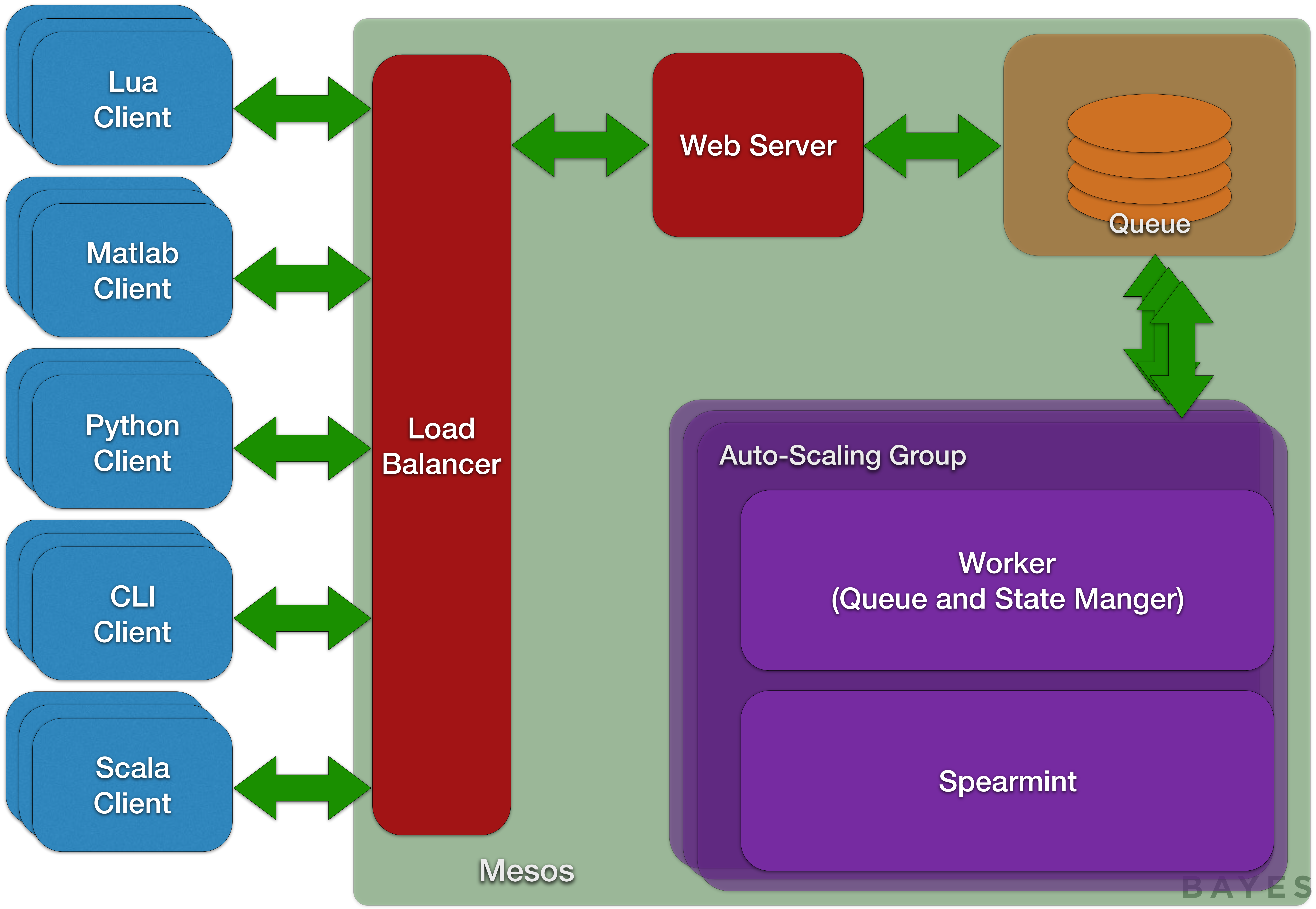
# BUILDING AN AUTOTUNER

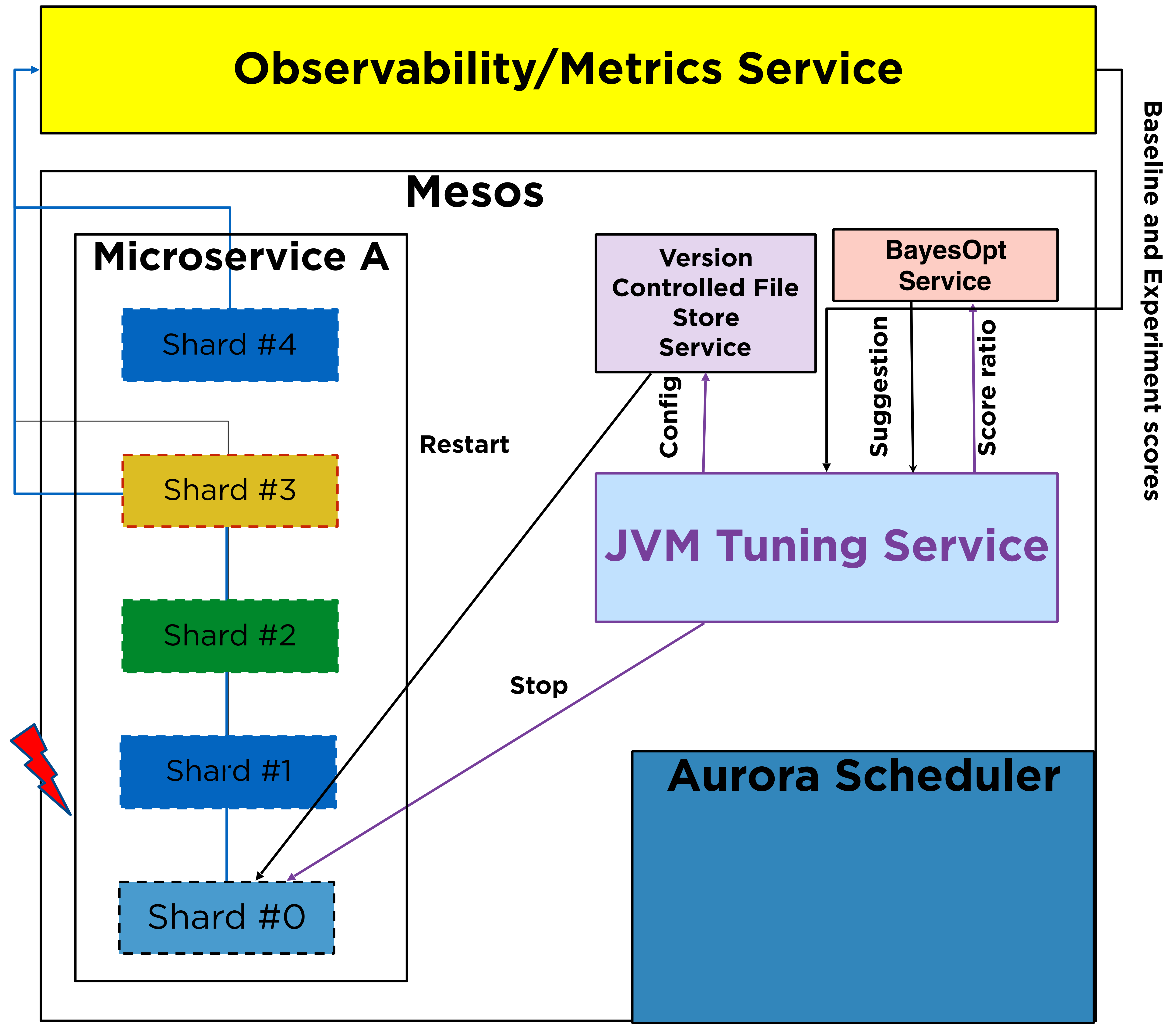
An example API call

```python
import whetlab
scientist = whetlab.ExperimentSet(name="My Experiment",
                    parameters={"learningRate": {"min": 0, "max": 1},
                                "proportionUsersInTest": {"min": 0, "max": 1},
                                "numberOfTweetsToShow": {"type": "int", "min": 1, "max": 25},
                                "extraTweetsType":, {"type": "enum", "options": ["moment", "ad", "anotherTweet"]}})
job = scientist.suggest()

dailyActiveMinutes = runMyExperiment(job)

scientist.update(job, dailyActiveMinutes)
```

# BUILDING AN AUTOTUNER

## The service layout of BayesOpt at Twitter





1. Get a new parm suggestion from BayesOpt
2. Generate JVM configuration
3. Upload new configuration to File Store Service
4. Get baseline platform information
5. Stop-and-restart test instance on specific hardware
6. Test for a fixed duration
7. Get valid baselines
8. Obtain baseline performance score
9. Obtain performance score from experiment
10. Compute the ratio and inform BayesOpt

## MEASURING PERFORMANCE OF JVM TUNING

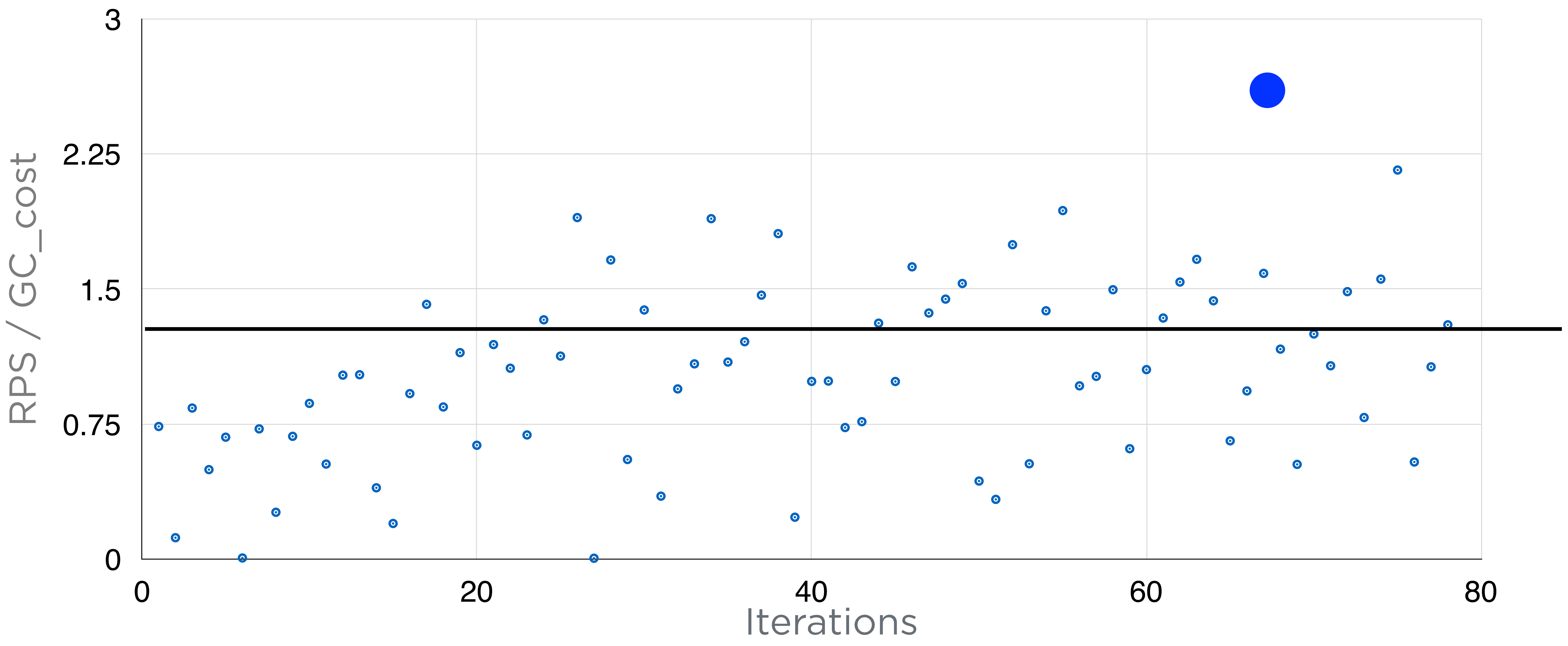
**RPS**: requests per second

**GC_cost**: wall-clock time spent doing GC

**Objective**: RPS / GC_cost

## EVALUATION

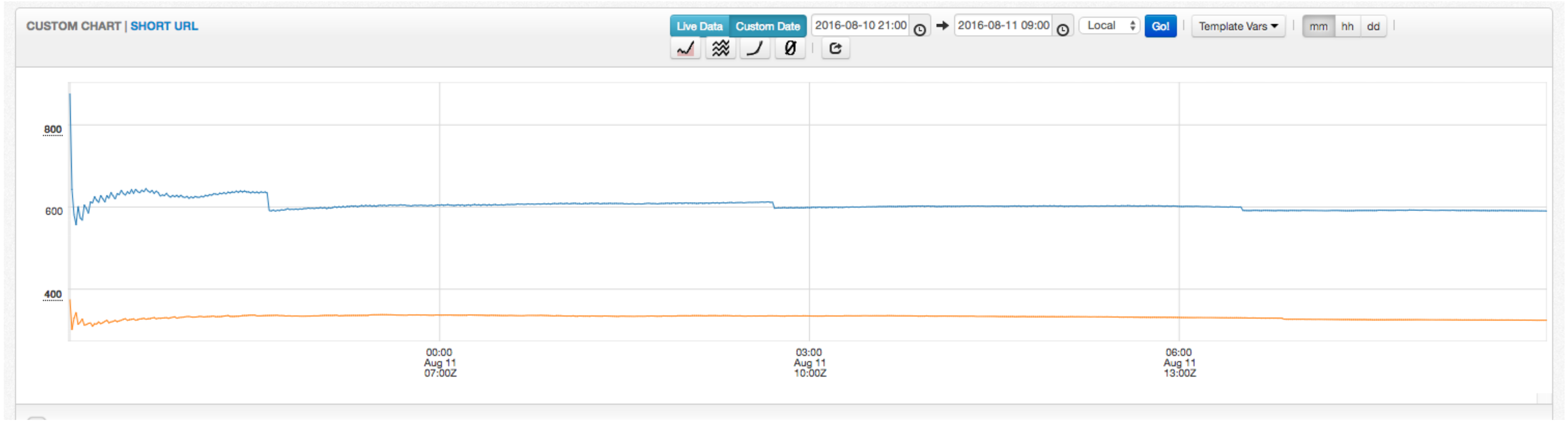
BayesOpt in action



# PERFORMANCE OF THE OPTIMUM RESULT

🟦 BayesOpt

🟨 Baseline

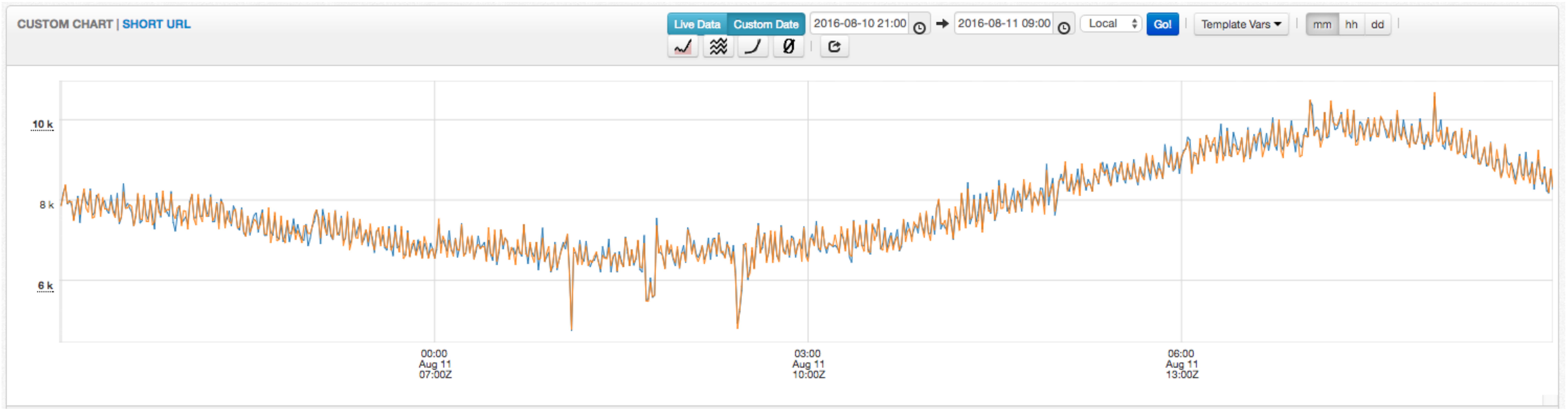


**Performance ratio: 80% performance improvement**

# REQUESTS PER SECOND

🟦 BayesOpt

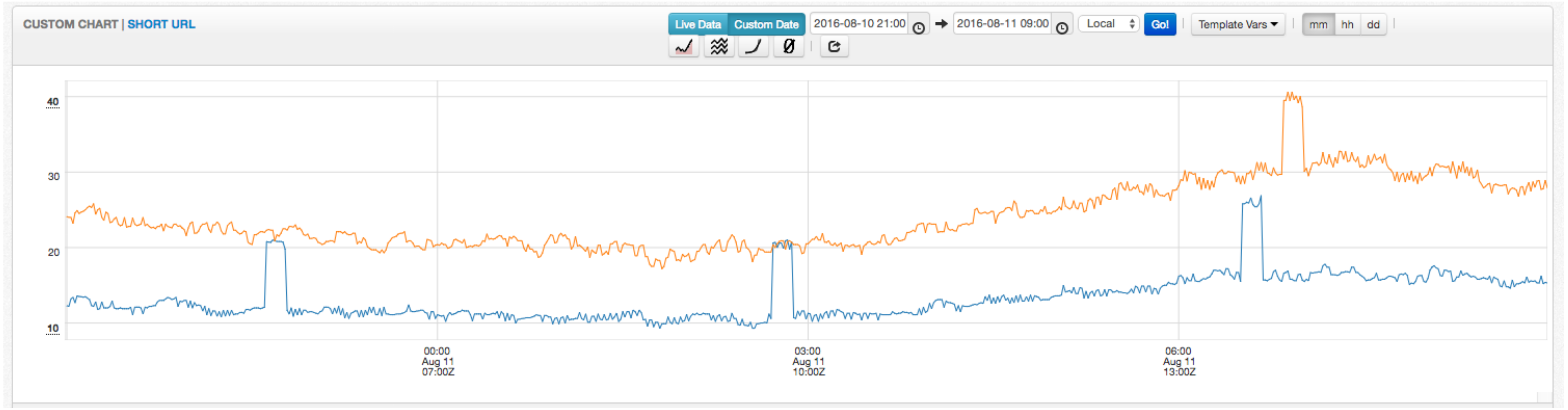
🟨 Baseline



# GC COST

🟦 BayesOpt

🟨 Baseline



**GC_cost ratio: 15.3/ 27.9 = 54.8%, reduce by 45%**

# BAYESOPT WINS IN THE DATACENTER

## Large Scalding Jobs

Auto-tuning ~30 Hadoop parameters

Hadoop jobs by default take too many resources

Drastic savings possible by slimming jobs

**80% cost reduction in largest jobs at Twitter**

Optimal parameter settings end up being very specific to individual job topologies.

Work by Richard Whitcomb

## BAYESOPT WINS IN THE DATACENTER

Deep Learning

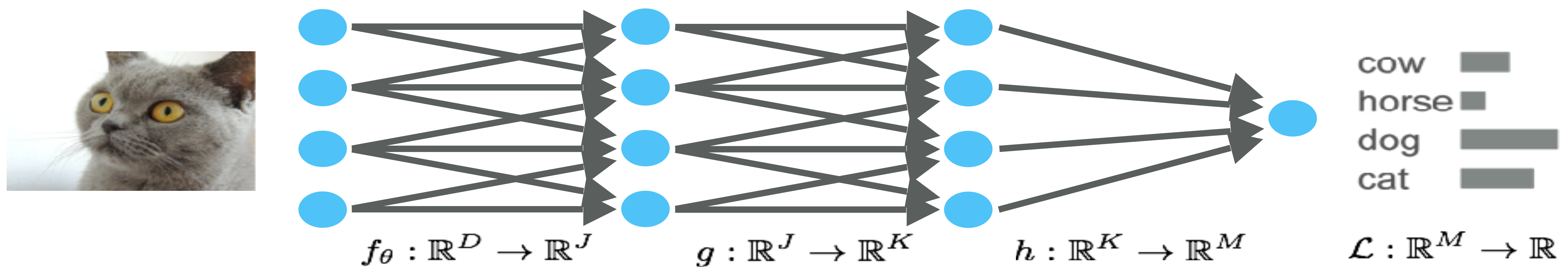
Periscope filtering, search & thumbnail generation

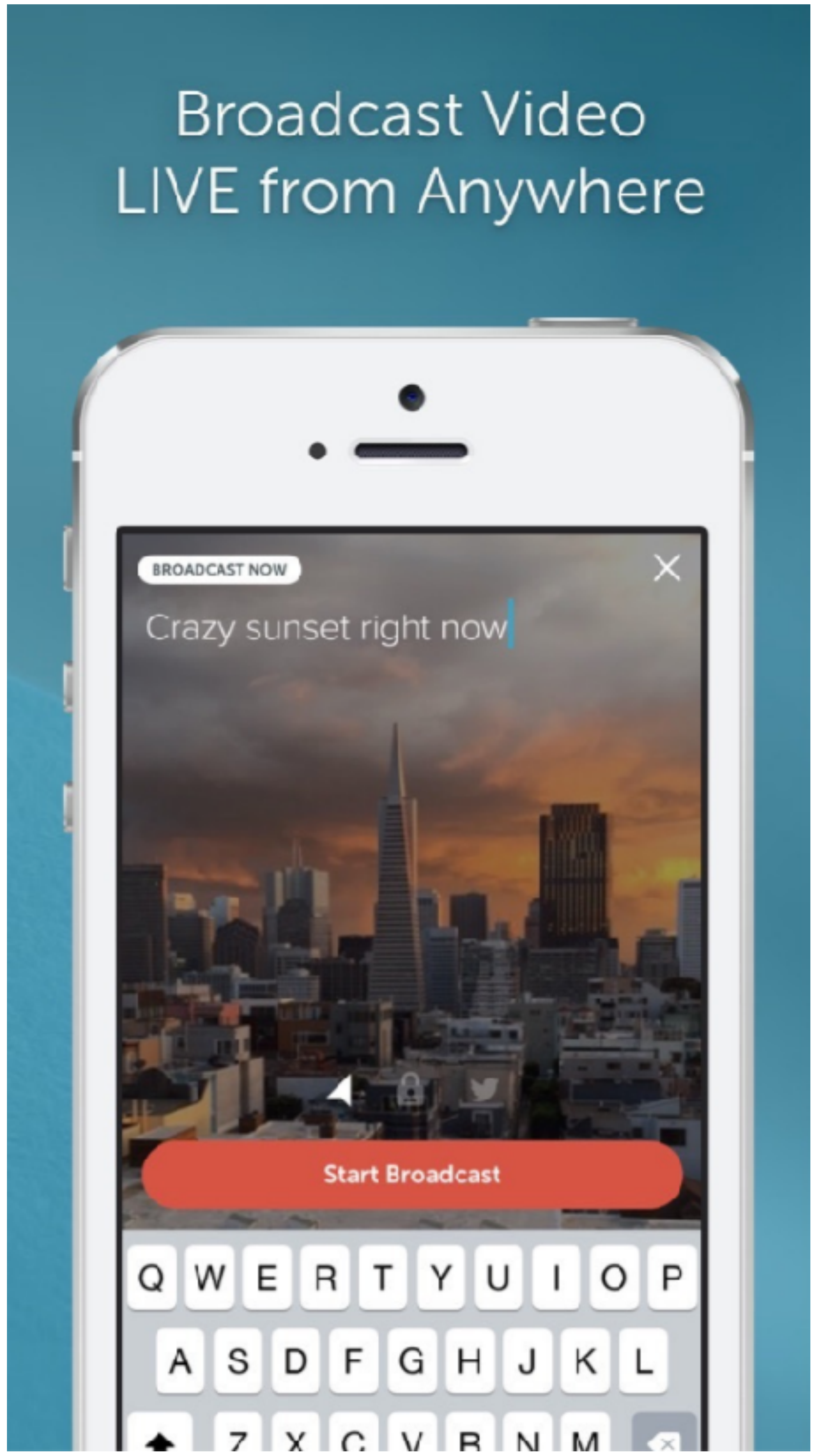
NSFW & NSFA image filtering

Abuse detection

**Autotuning all deep learning applications**

Dramatically speeds up development time


cow
horse
dog
cat
$f_\theta : \mathbb{R}^D \to \mathbb{R}^J$
$g : \mathbb{R}^J \to \mathbb{R}^K$
$h : \mathbb{R}^K \to \mathbb{R}^M$
$\mathcal{L} : \mathbb{R}^M \to \mathbb{R}$



## CHALLENGES IN INDUSTRY

## CHALLENGES IN APPLYING BAYESOPT

The teams that need BayesOpt are rarely expert in ML

Biggest deal: **problem definition & systematizing**

- Engineers find phrasing their problem as an optimization problem too constraining. This has to be a negotiation!
- Automating tuning of a costly system is costly

# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: How important is it to carefully pick my objective?**

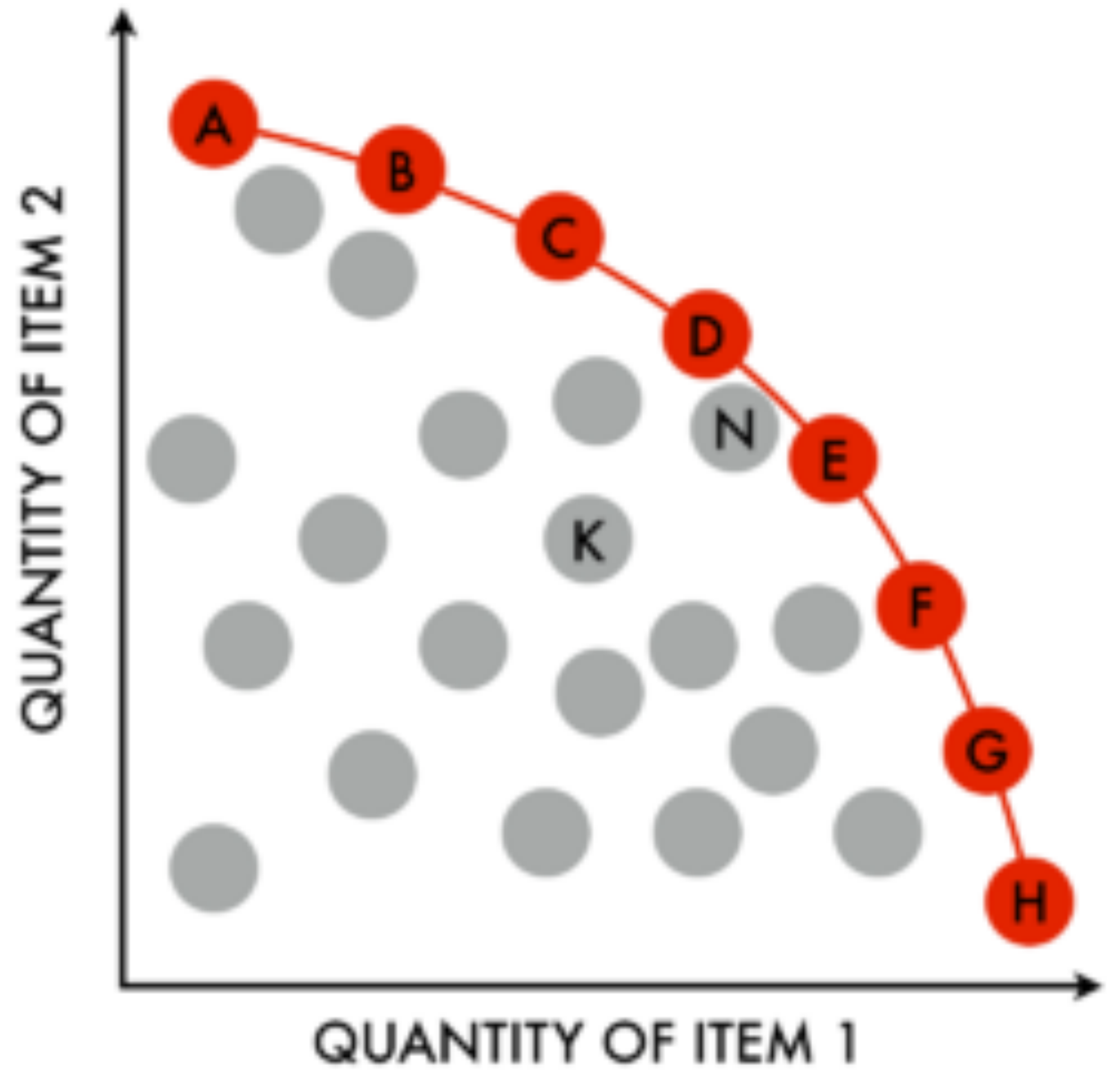
**A: The most important.**

BayesOpt is extremely good at making your objective go up, and *will* find any loopholes, weaknesses, or ways to cheat that exist in your experimental setup.

This is a blessing in disguise — systematizing your tuning and using BayesOpt will often expose unchallenged assumptions about your system.

Be prepared for surprises.

Tip: *Problem formulation is the "outer loop" of BayesOpt, and will be revisited multiple times. Don't hesitate to reformulate and restart.*



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: I want to make more than one number go up. Can BayesOpt handle more than one objective?**

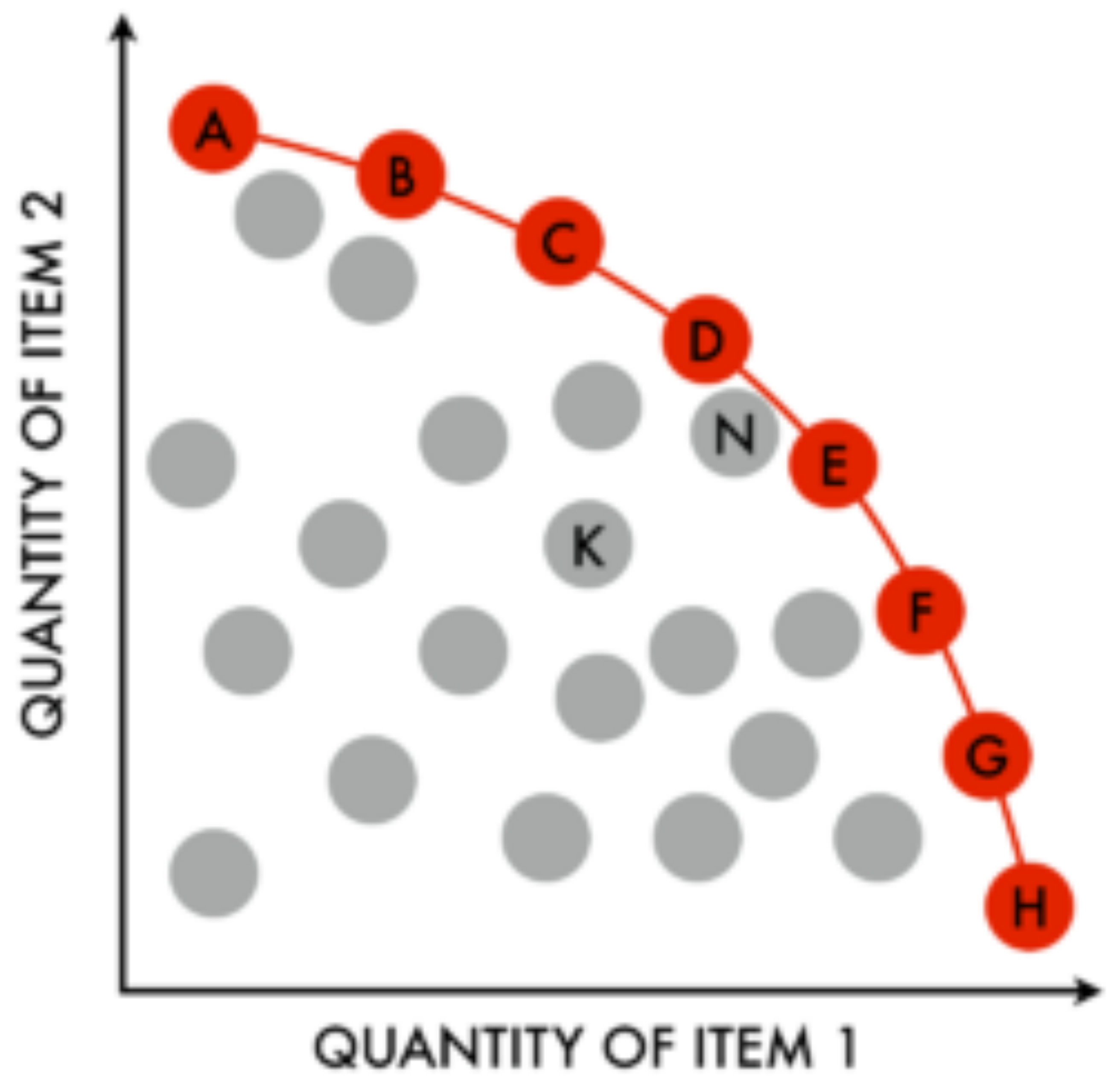
**A: Not directly, not easily.**

Many industrial problems don't naturally fit as the optimization of a scalar. Two possibilities:

1. **Weight your trade-offs** (I care 80% about making value X go higher, and 20% about making value Y go higher). Cognitively hard to do well. Often vulnerable to cheating.
2. **Rephrase the secondary objectives as constraints** (I want value X to be as high as possible, as long as value Y doesn't go below Z)

We have had the most success with the second option, and you can have as many constraints as you like.

*The choice of what and how to constrain is an art*



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: Tuning in production is hard, can I use BayesOpt in staging or a simulation?**

**A: Do it live!**

If your offline testing framework accurately reflects production, then go for it!

In our experience, approximations to the real problem always have led BayesOpt to "cheat"

It is worth the investment to train on the real problem.

Tip: *push collaborators to work on the real problem, not a possibly-related easy problem.* **Trust, but verify***.*

# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: How many iterations?**

**A: Empirical.**

Depends on how easy the problem is,

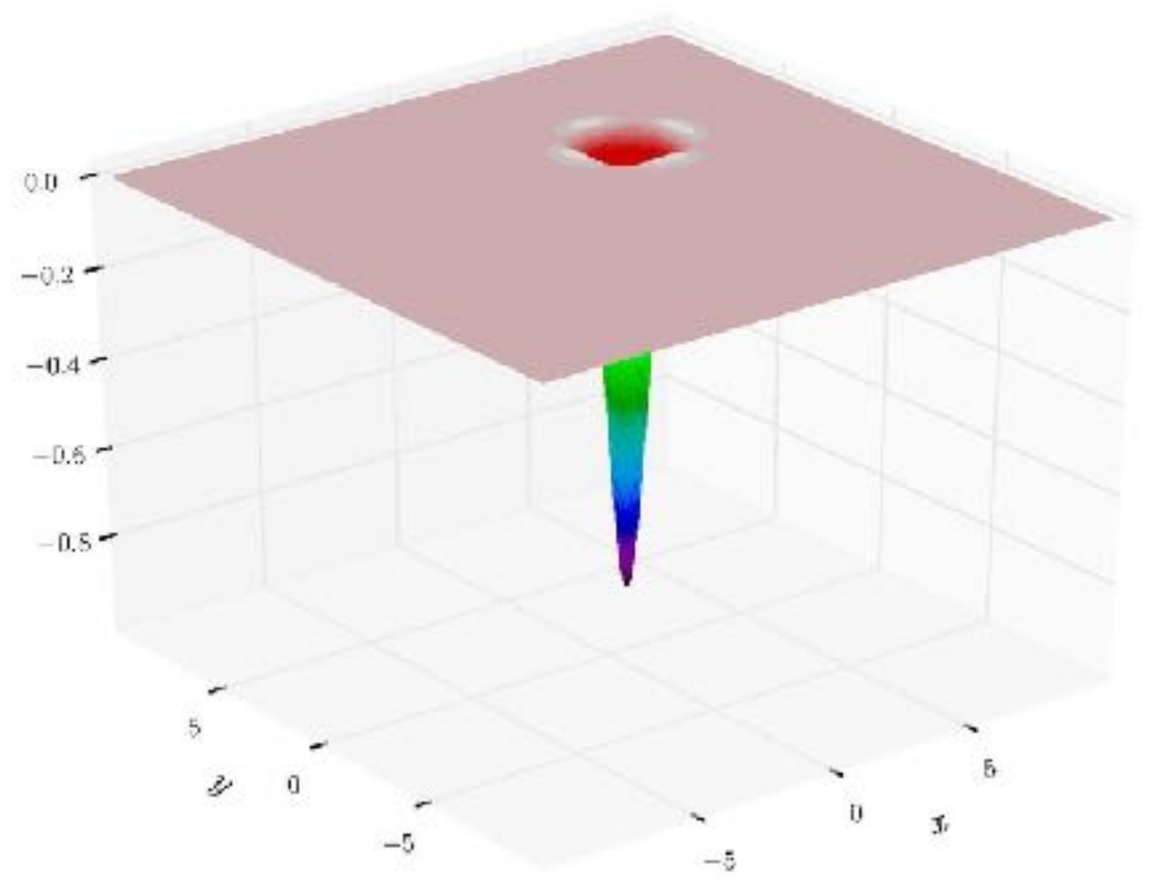
Depends on how noisy the problem is,

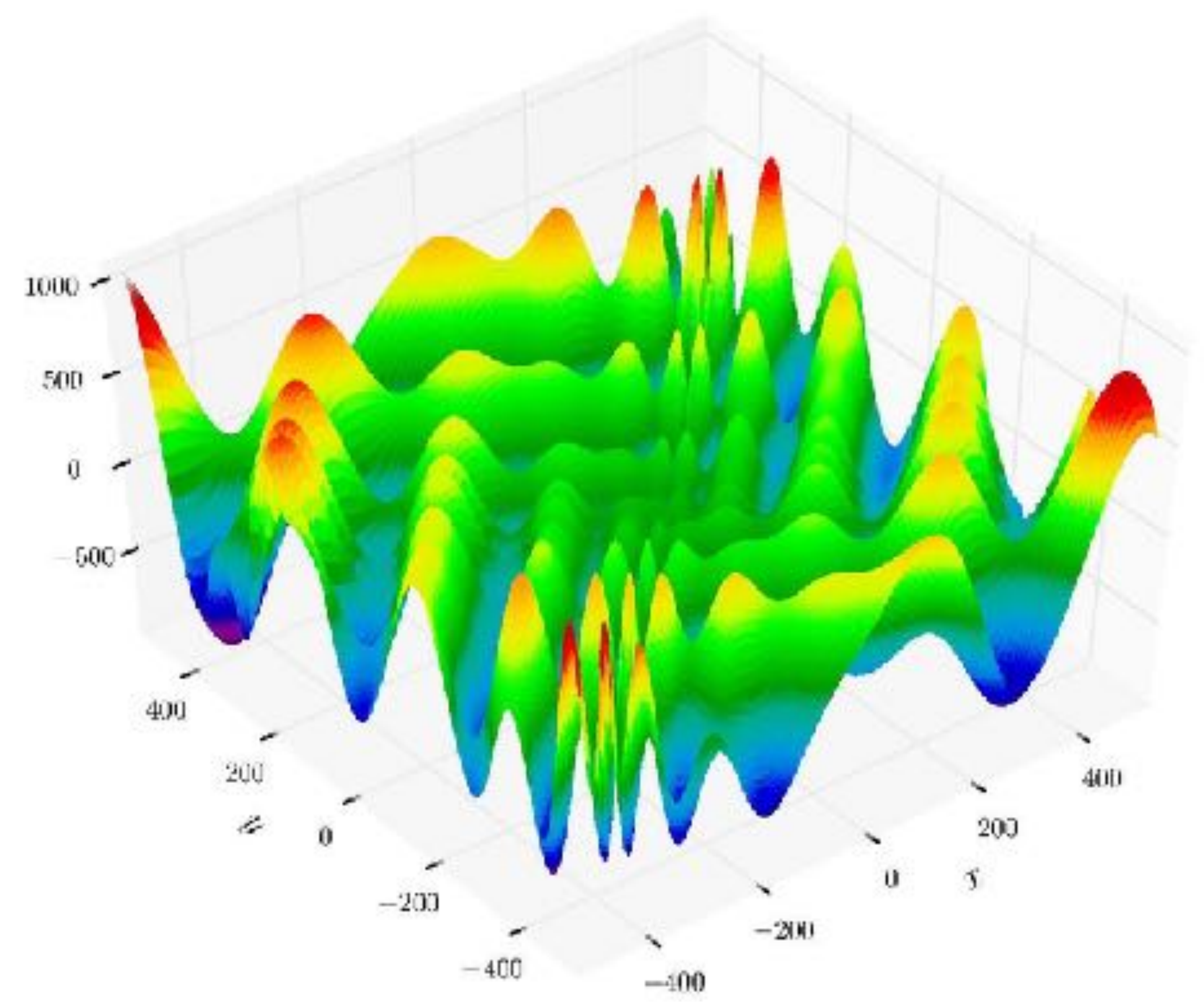
Depends on how good your first guesses are,

Our rules of thumb:

- **Don't do less than 20 suggestions.**
- **If your best result hasn't improved in 20-40 suggestions, you're probably done.**
- **You don't need to do more than 1000.**

Real need: *battle-tested stopping heuristics*

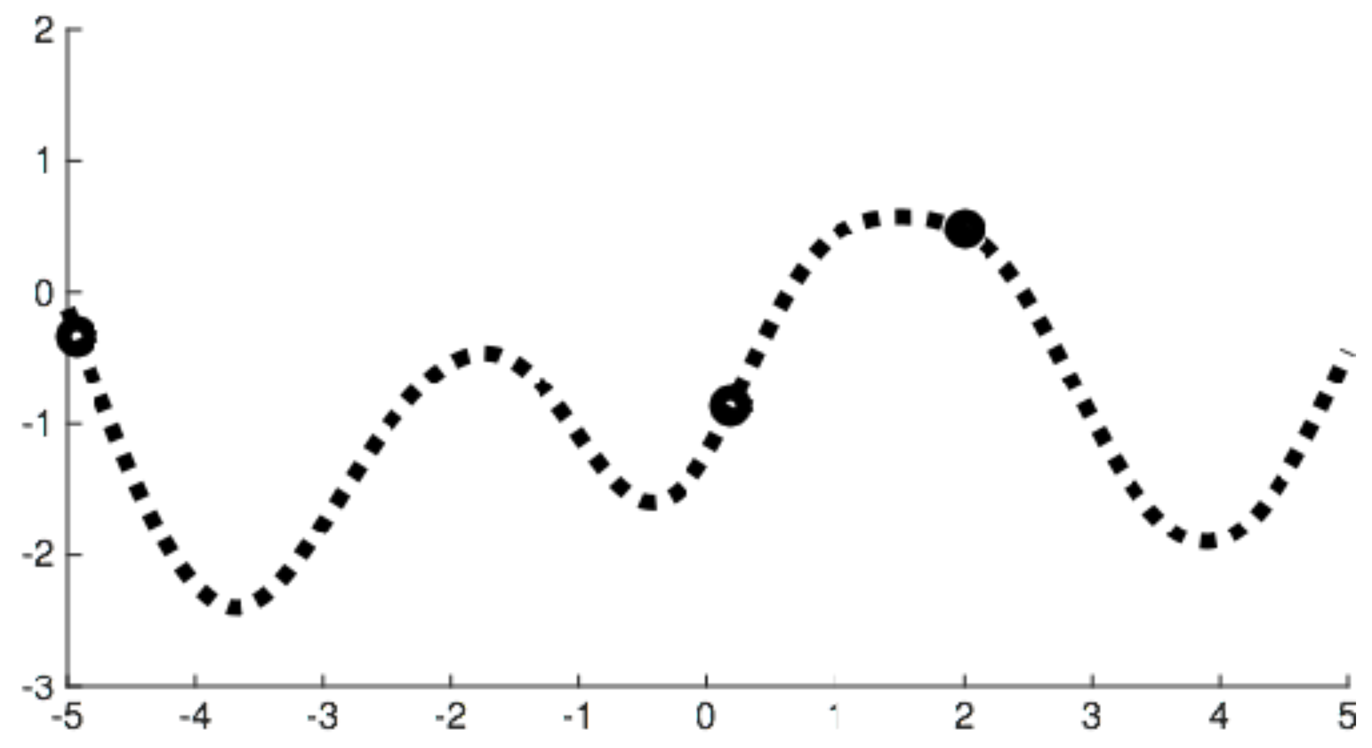




https://en.wikipedia.org/wiki/Test_functions_for_optimization

# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask



**Q: How many parameters should I use?**

**A: Empirical.**

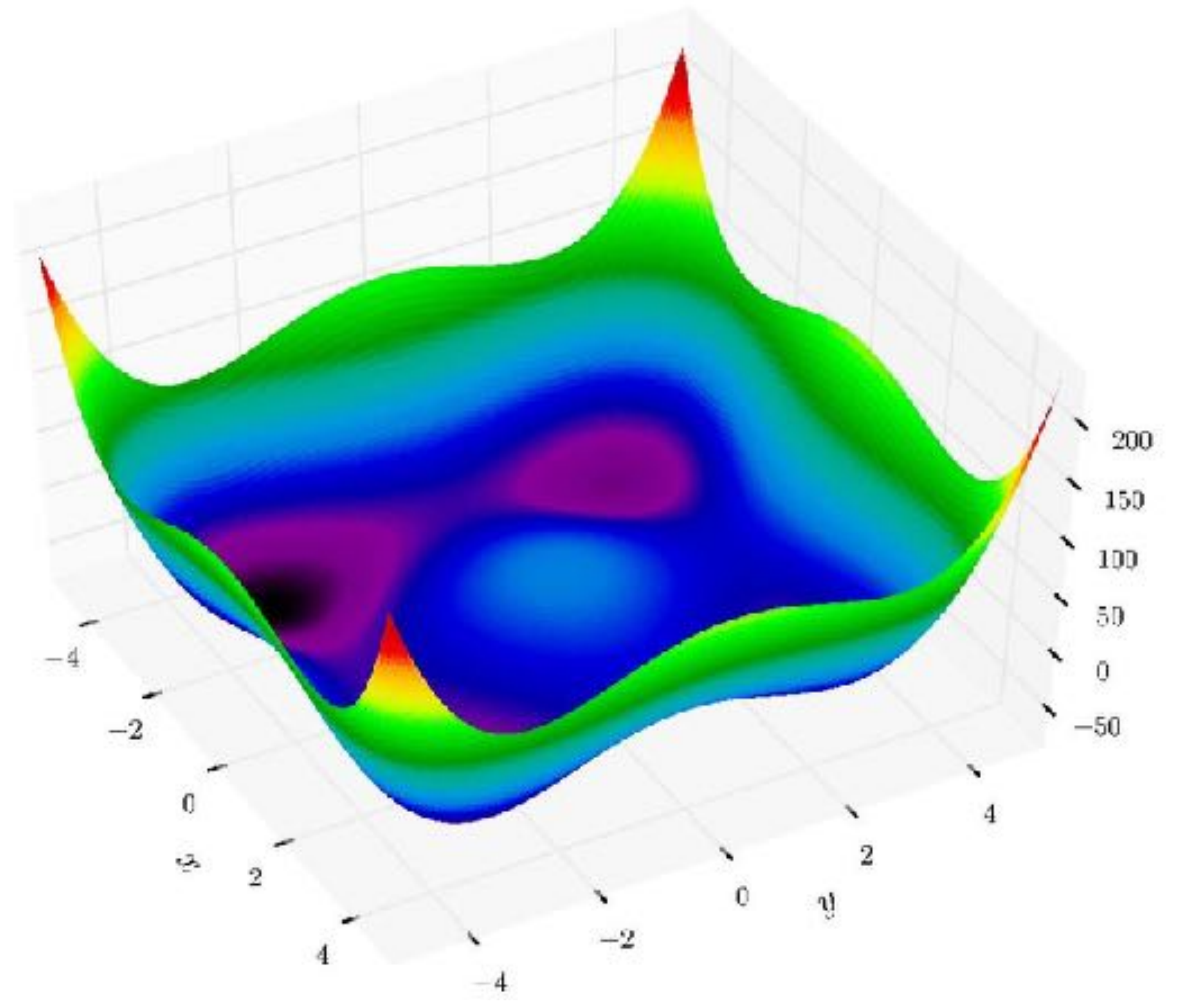
The curse of dimensionality lurks! Too many (> 100), and nothing is better than random search.

Ideally, everything you think would matter.

Practically, whatever you can afford.

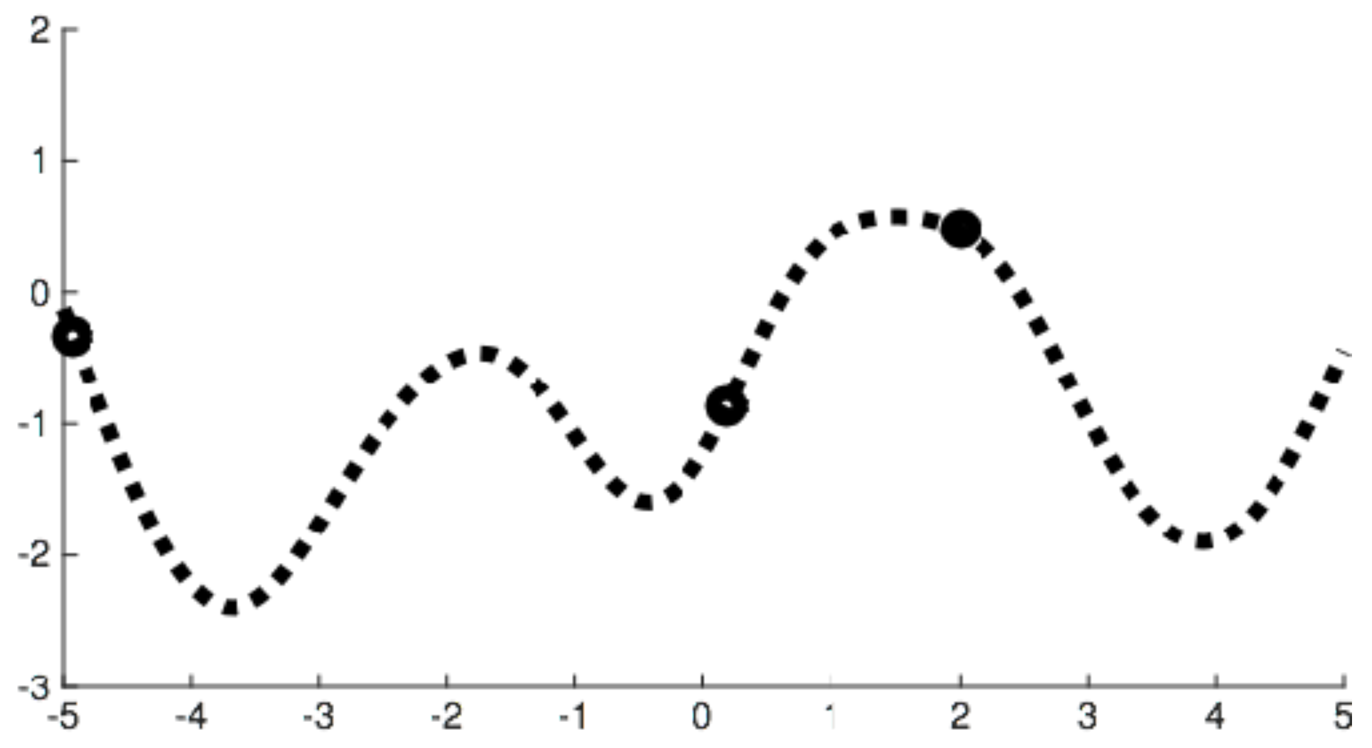
Good rule of thumb: **4 - 30 parameters**.

Real need: *automatically subsampling the kitchen sink.*



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

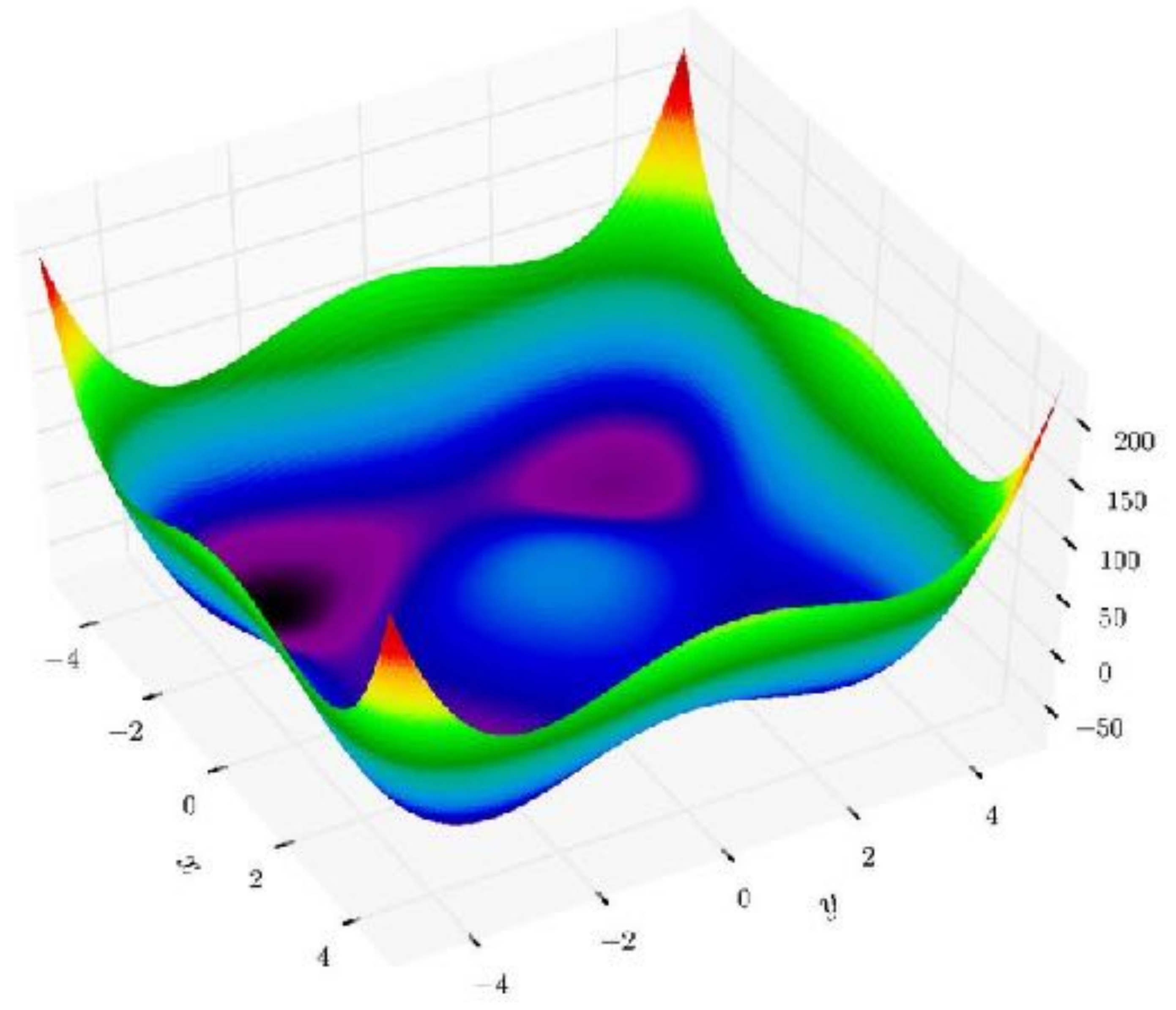


**Q: How many parameters should I use?**

**A: Empirical.**

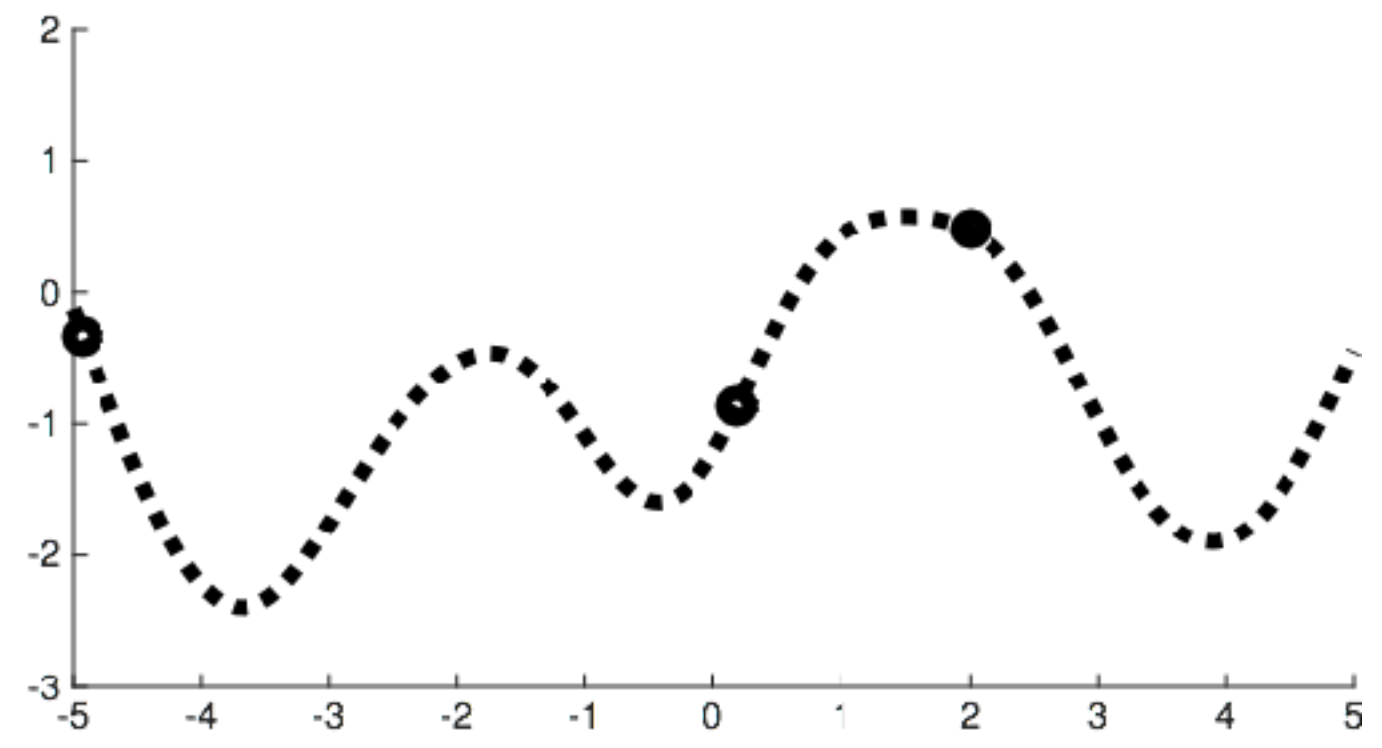
If you can **rewrite parameters to make some invalid settings impossible**, do so! For example, if you have two parameters a & b, and only valid settings are a < b, then reparameterize to min_ab, and offset_from_min_ab.

Tip: *help collaborators understand how to not waste model capacity.*



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

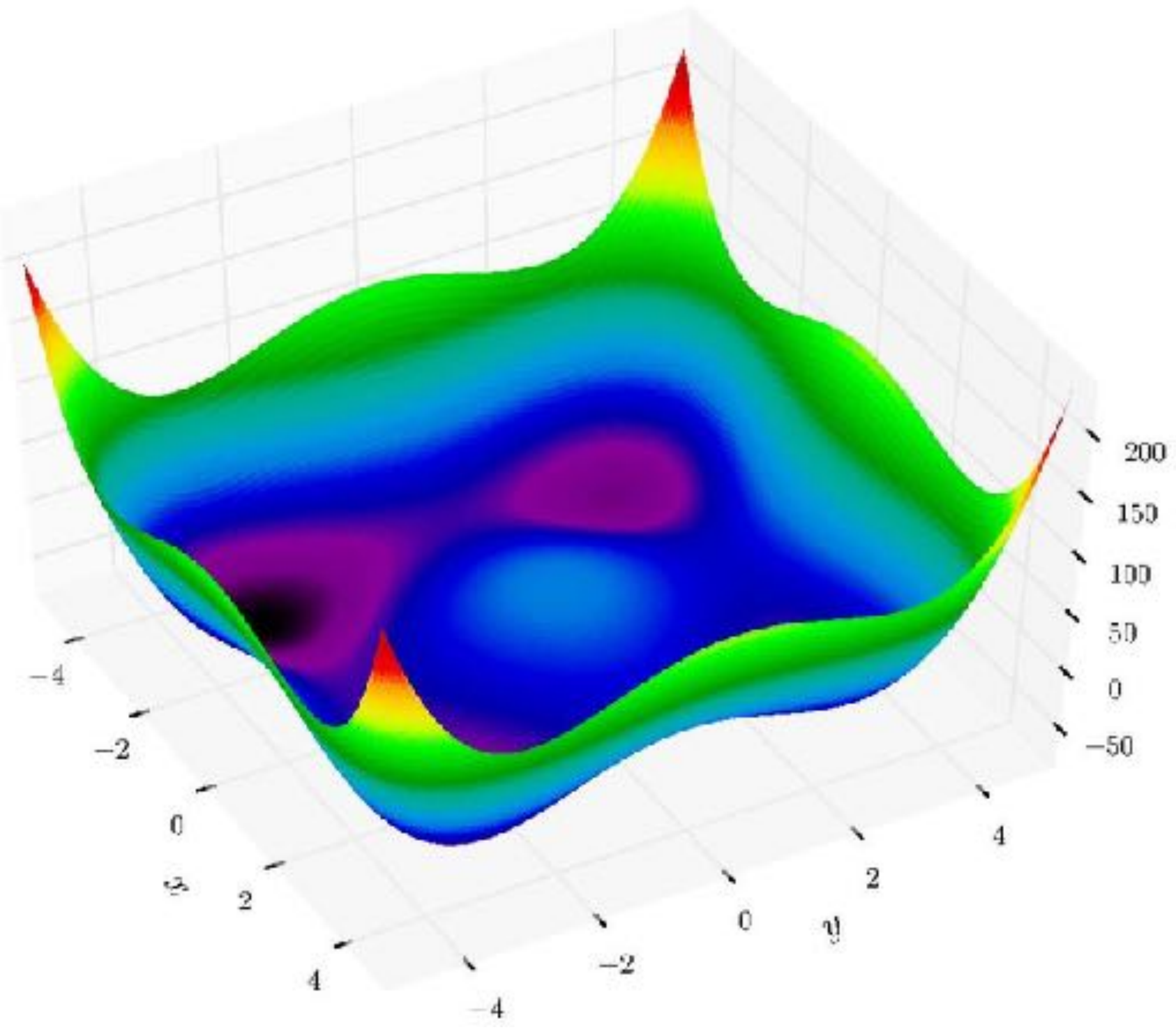


**Q: Why do the first suggestions look random?**

**A: Don't look at the first ten suggestions.**

Yes, they're off the grid, or they're effectively random.

You will have to deal with the *psychological* aspect of initial suggestions coming from a high-variance estimate of the objective function.



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

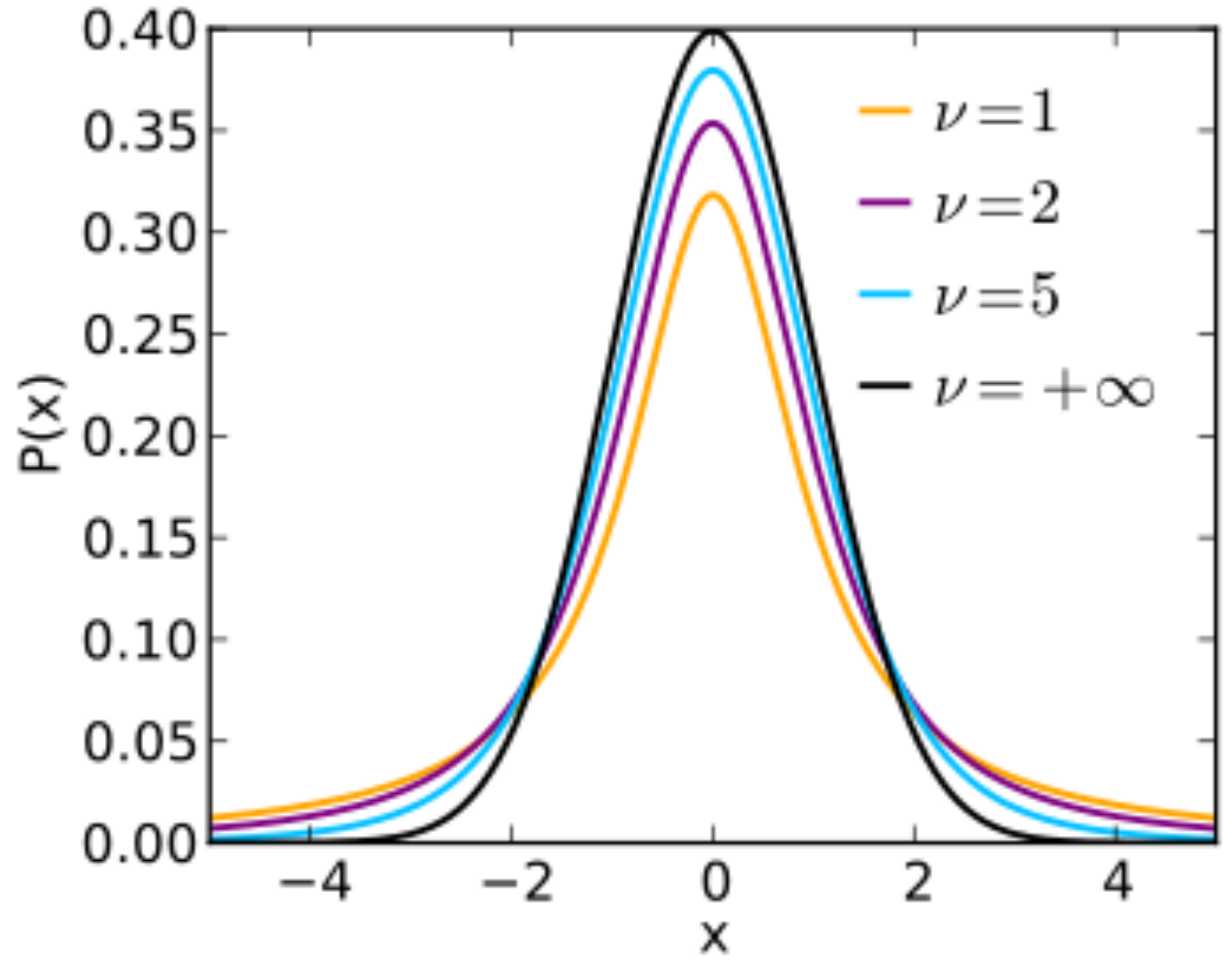
**Q: Which parameters actually matter?**

**A: This is up to the domain expert.**

This information isn't accessible from our BayesOpt service in an interpretable fashion.

Teams don’t just want an optimum: they want to know *why* some suggestion is an optimum.

Real need: *model-independent measures of sensitivity & good visualizations.*



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: What ranges should I use for my parameters?**

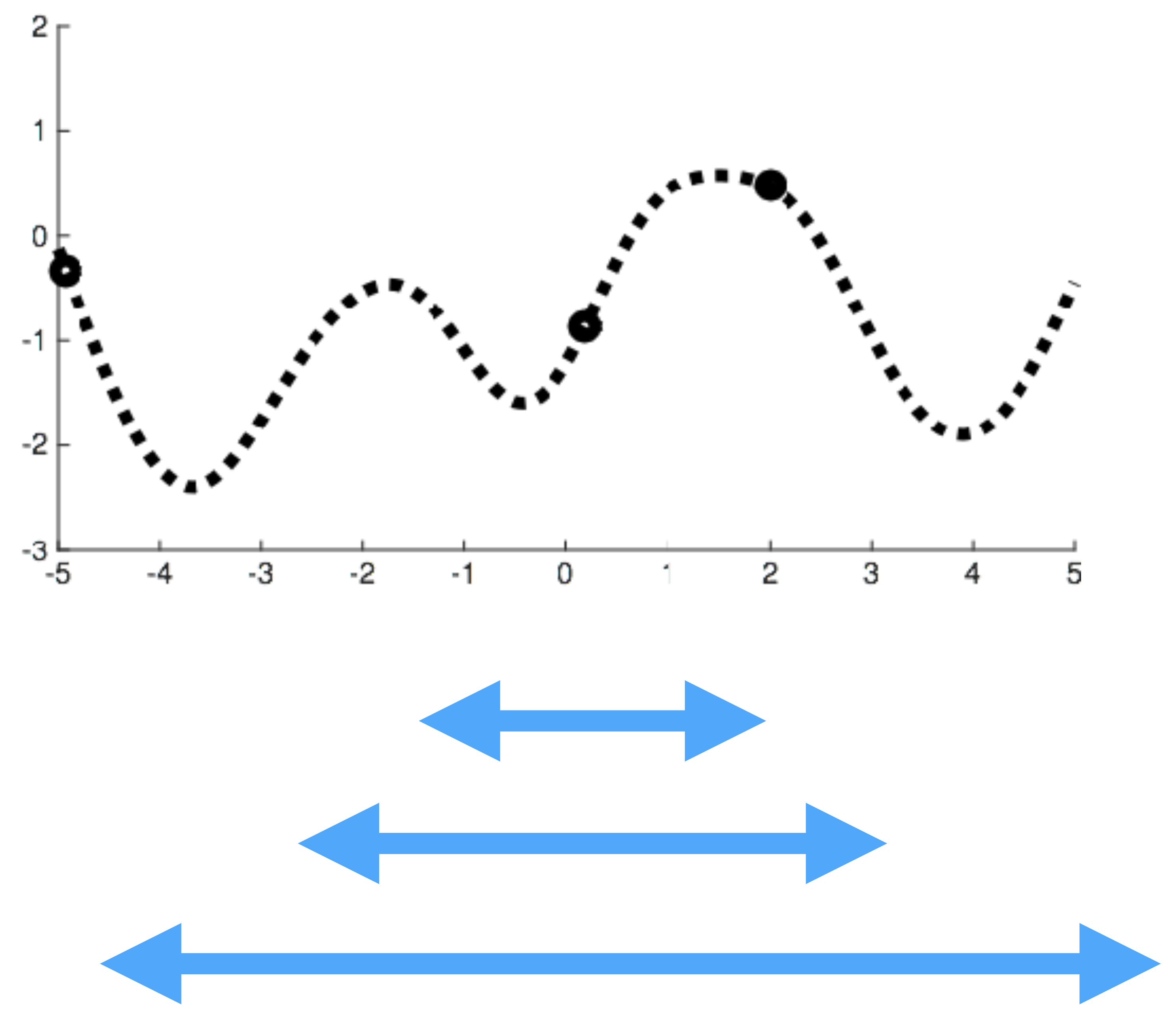
**A: As large as is safe, or as you can afford.**

Do your best not to let BayesOpt choose invalid or ridiculous settings.

This doesn't mean "strange" settings — it means ones that you can prove would lose money or cause problems. BayesOpt will often find weird solutions that humans didn't expect, and that's the point!

Whether or not to start with tight or wide ranges depends on your problem. We tell teams to start tight, and if the best BayesOpt results keep hitting the bounds, then open it up.

Note: *Automation here exists, but I don't know its reliability. This is an important feature that should be pervasive. Prior elicitation won't go away, though!*



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: How long should I run my experiments?**

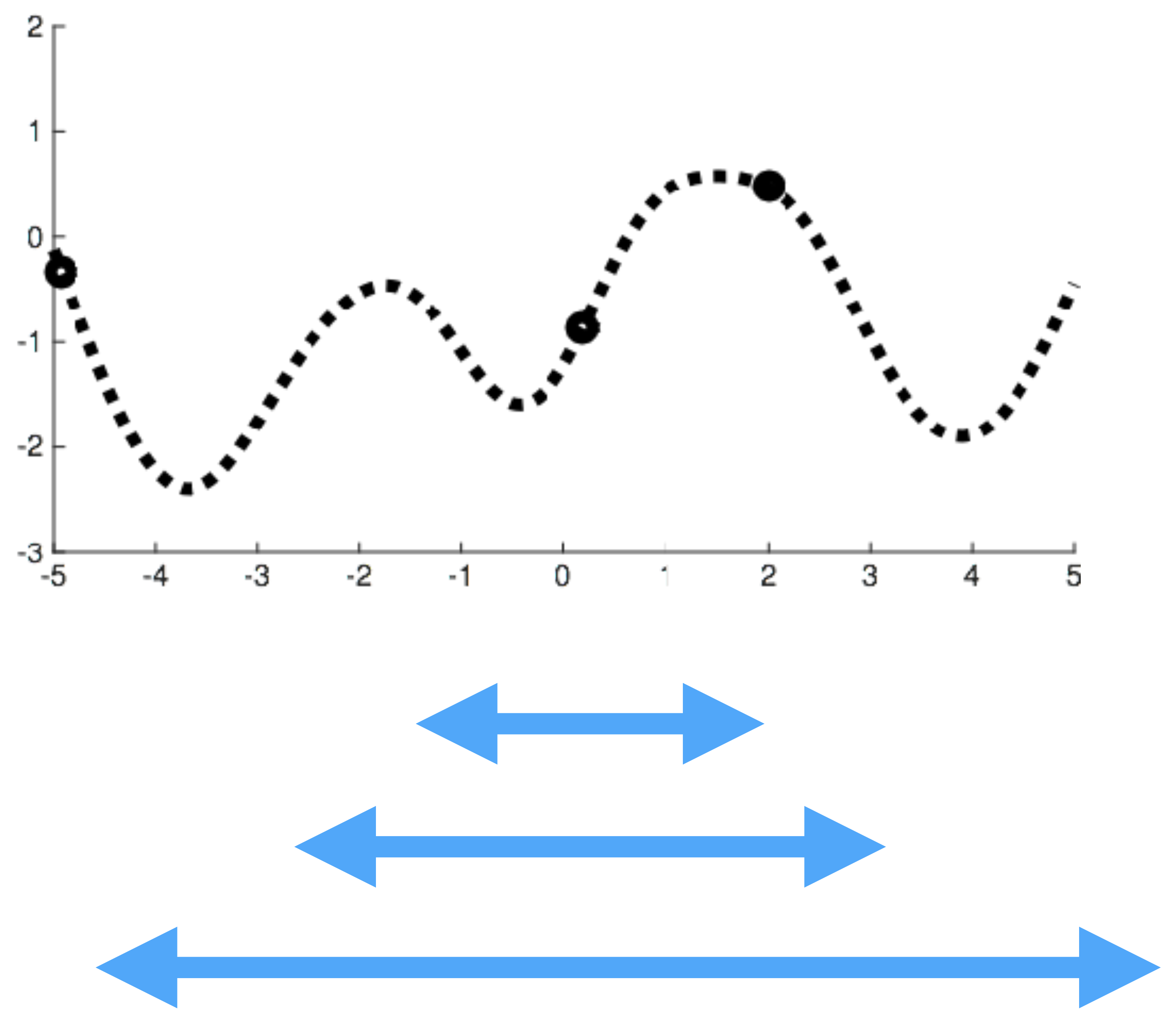
**A: As long as you can afford.**

BayesOpt can deal with noisy inputs. But we don't have an off-the-shelf answer for how to trade off noise in the observations versus noise in the model predictions.

In many problems, spending more time or money can reduce the noise of an observation.

Domain experts need an answer, in the form of "if experimentation costs \$X / minute, you will minimize overall cost of the experiment if you average over Y minutes".

Real need: *Metrics for trading off observation noise and total length of optimization*



# OPEN BAYESOPT PROBLEMS IN PRODUCTION

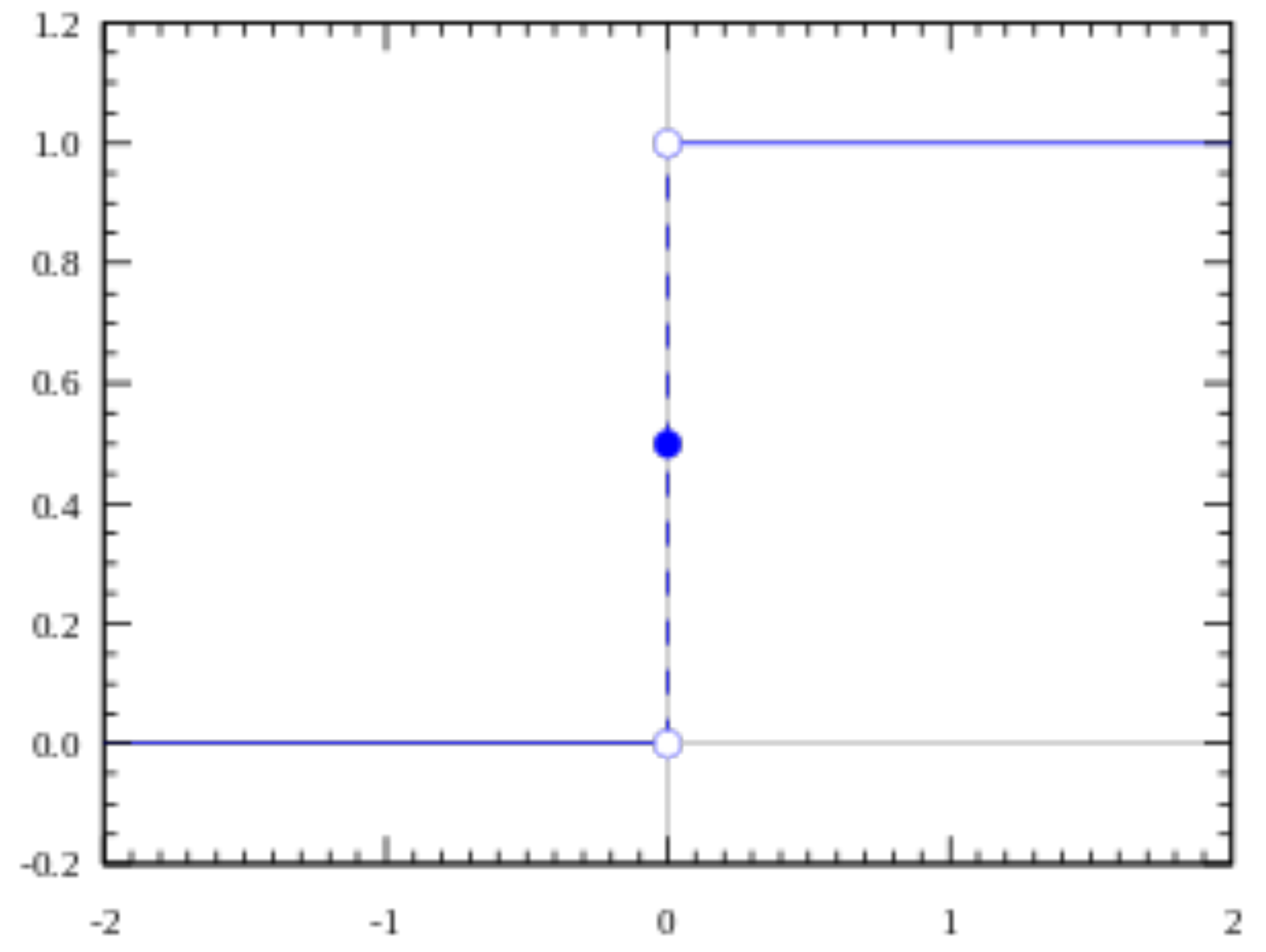
Or: questions that domain experts repeatedly ask

**Q: I know my optimum will change based on hardware/ day/season/something else. What do I do?**

**A: Control for these variables as best as you can.**

Generic BayesOpt assumes your problem is not a moving target.

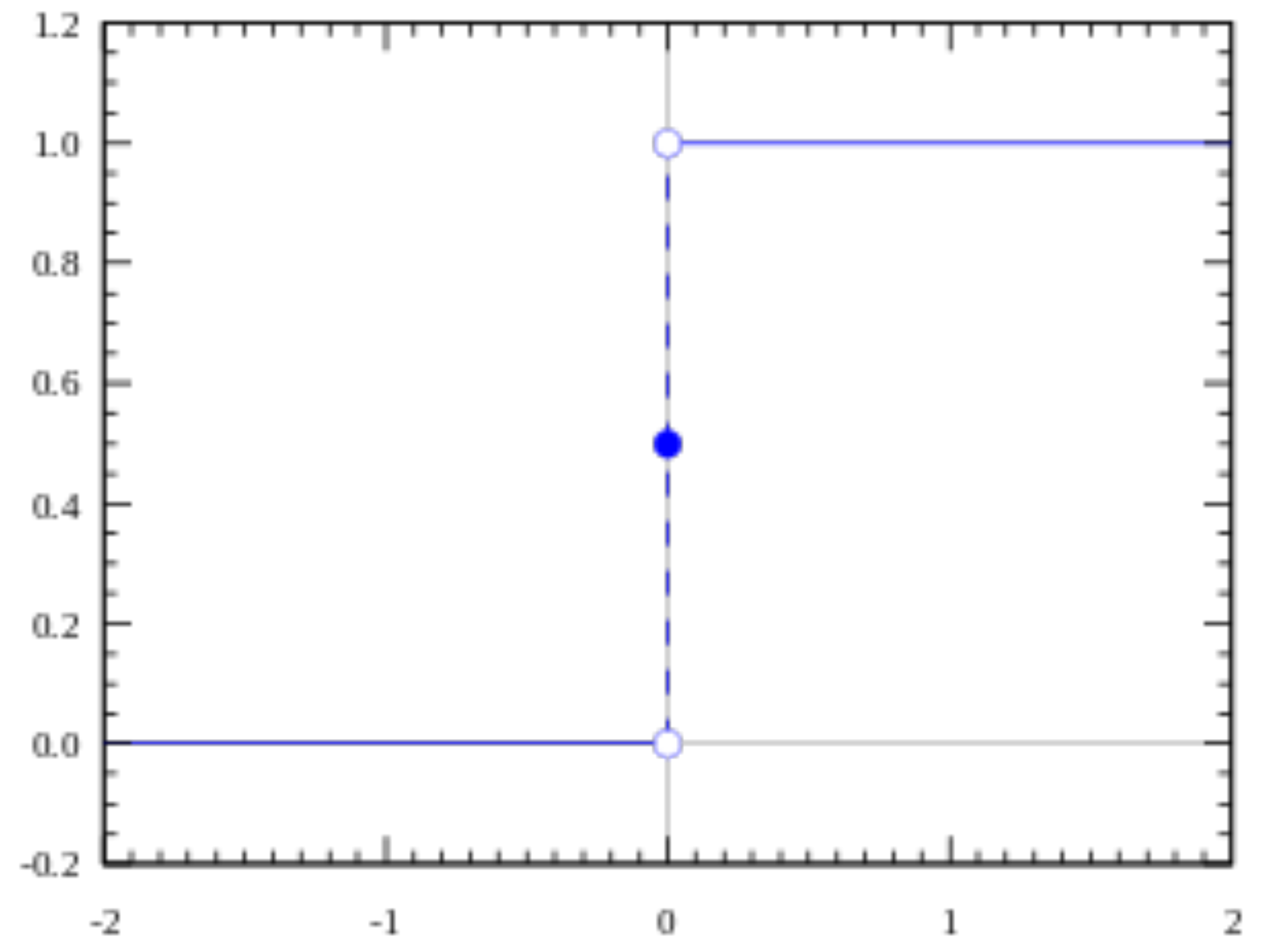
*Non-stationarity lurks in many industrial problems.*



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

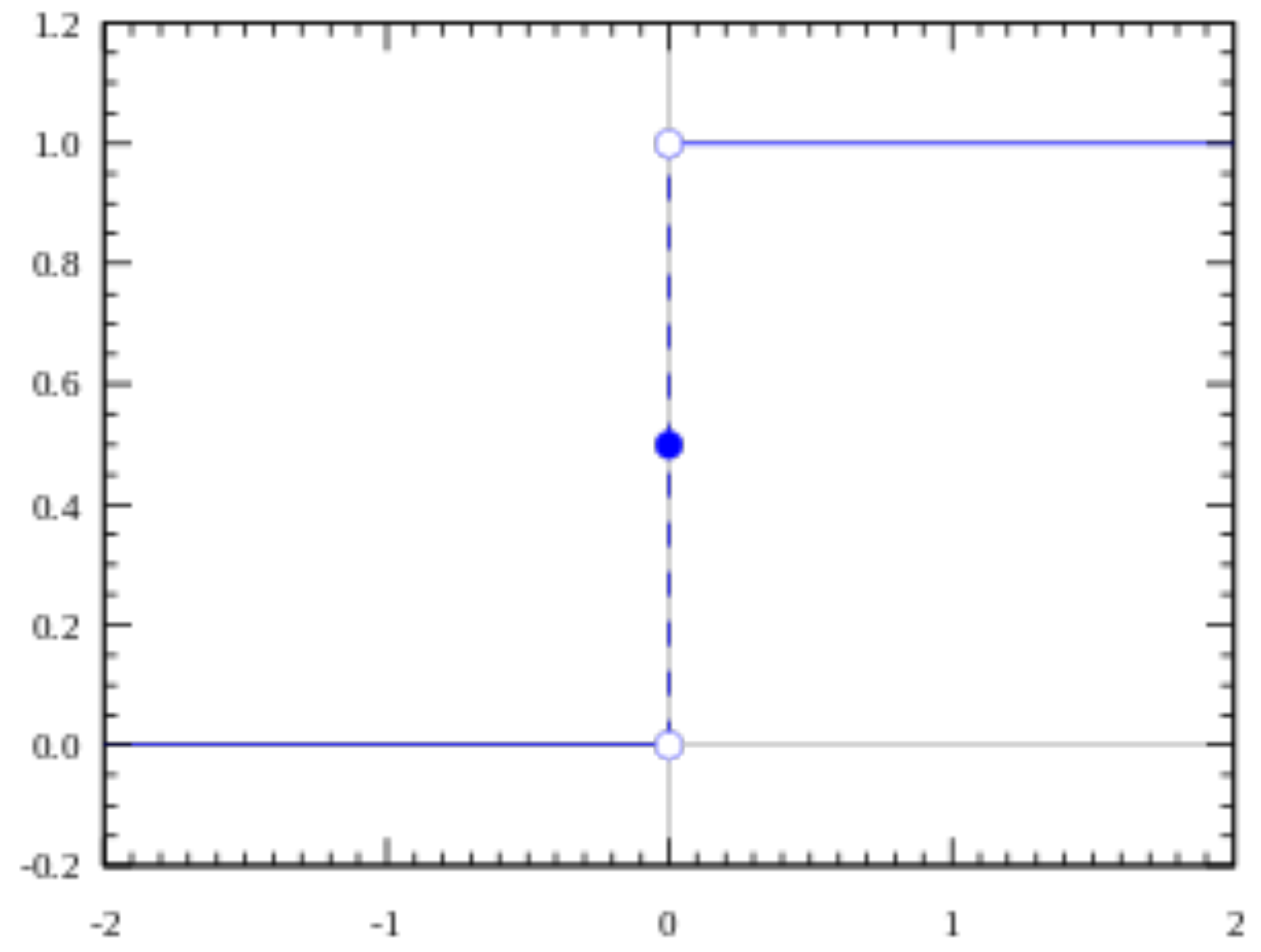
If you know the optimum depends on hardware: **set up separate experiments for each hardware platform** (e.g. this is necessary for tuning the JVM).



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

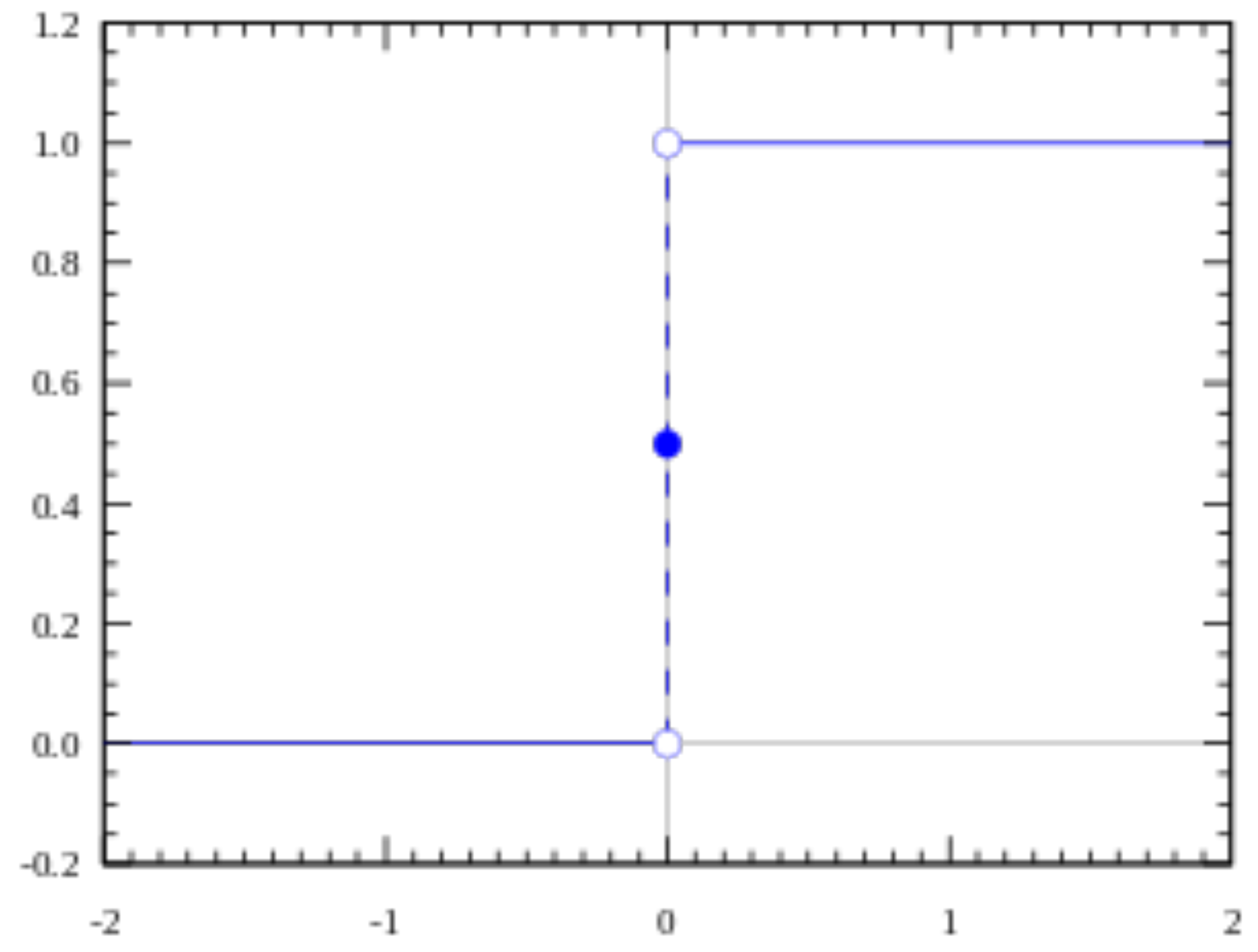
**If your problem drifts too fast, you have a control problem**, not a black-box optimization problem.



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

Heterogeneous load — **if you can't replicably produce conditions that will give you accurate measurements, then your experimental setup is inadequate**, and needs to be fixed.



## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Pay attention to the scale of your objective. Extremely bad results can throw BayesOpt off the scent.** Forecasting if this will happen requires aggressive interrogation of the domain expert. Otherwise, you will discover it as you work.

| | | |
|---|---|---|
| +1% | | +1% |
| -2% | → | -2% |
| -1000000% | | X |

We almost always had to build in auto-detection of constraints like these. Integrating out length scale has been inadequate for us.

Real need: *battle-tested outlier models.*

## OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**Q: Do I really need BayesOpt?**

**A: It depends.**

**Q: Do I need to understand my problem and automate my tuning?**

**A: Absolutely.**

The dirty secret: a huge portion of each successful industrial project came from *automation and systemization.* In many projects, we could afford to try many, many suggestions.

BayesOpt has a real edge over random search when experiments are expensive (usually time, not money), which is not always true.

BayesOpt, in our hands, has an unrealized advantage in potential interpretability of the learned objective function.

## QUESTIONS?

@awiltsch

alex.bw@gmail.com

# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

**If you make a mistake, or if you find your experiments aren't running long enough to give accurate results, correct them before moving on to the next suggestion.**

Not running load tests for long enough

Not including enough users

Rescaling vast differences in objective with sqrt() or log()

Error in objective entry (decimal places!)

# OPEN BAYESOPT PROBLEMS IN PRODUCTION

Or: questions that domain experts repeatedly ask

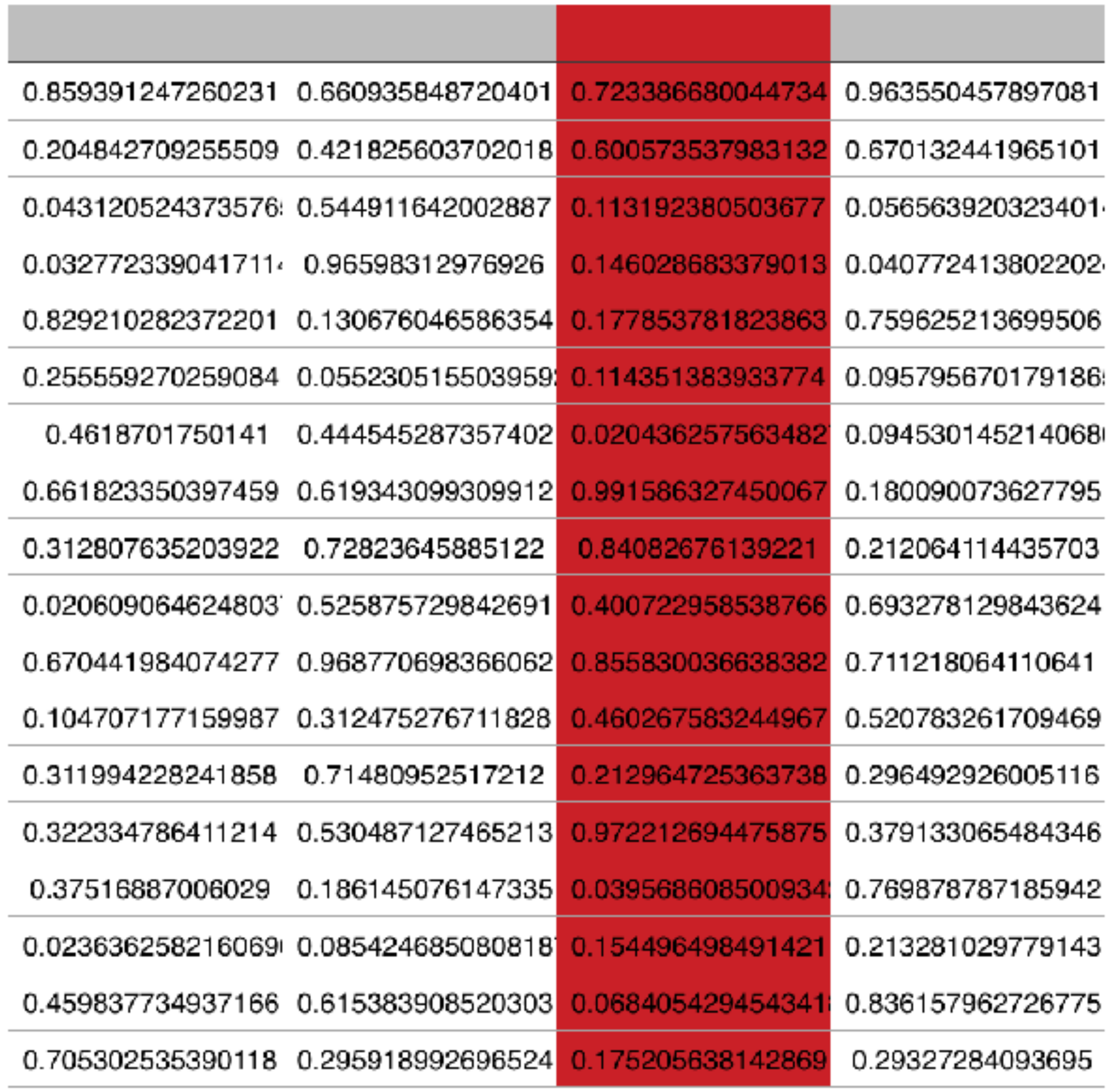
**Q: Can I ask BayesOpt to suggest values for only a subset of parameters?**

**A: This is possible.**

It is common that some parameters that affect performance cannot be set freely by a BayesOpt system, e.g. hardware platform. We either:

- If the parameters are categorical, we stratified the choices into separate experiments
- Otherwise, we simply overrode the suggested parameter value.

Real need: *allow users to fix the value of some parameters , and have BayesOpt fill in the rest.*



| | | | |
|---|---|---|---|
| 0.859391247260231 | 0.660935848720401 | 0.723386680044734 | 0.963550457897081 |
| 0.204842709255509 | 0.421825603702018 | 0.600573537983132 | 0.670132441965101 |
| 0.043120524373576 | 0.544911642002887 | 0.113192380503677 | 0.056563920323401 |
| 0.032772339041711 | 0.96598312976926 | 0.146028683379013 | 0.040772413802202 |
| 0.829210282372201 | 0.130676046586354 | 0.177853781823863 | 0.759625213699506 |
| 0.255559270259084 | 0.055230515503959 | 0.114351383933774 | 0.095795670179186 |
| 0.4618701750141 | 0.444545287357402 | 0.020436257563482 | 0.094530145214068 |
| 0.661823350397459 | 0.619343099309912 | 0.991586327450067 | 0.180090073627795 |
| 0.312807635203922 | 0.72823645885122 | 0.84082676139221 | 0.212064114435703 |
| 0.020609064624803 | 0.525875729842691 | 0.400722958538766 | 0.693278129843624 |
| 0.670441984074277 | 0.968770698366062 | 0.855830036638382 | 0.711218064110641 |
| 0.104707177159987 | 0.312475276711828 | 0.460267583244967 | 0.520783261709469 |
| 0.311994228241858 | 0.71480952517212 | 0.212964725363738 | 0.296492926005116 |
| 0.322334786411214 | 0.530487127465213 | 0.972212694475875 | 0.379133065484346 |
| 0.37516887006029 | 0.186145076147335 | 0.039568608500934 | 0.769878787185942 |
| 0.023636258216069 | 0.085424685080818 | 0.154496498491421 | 0.213281029779143 |
| 0.459837734937166 | 0.615383908520303 | 0.068405429454341 | 0.836157962726775 |
| 0.705302535390118 | 0.295918992696524 | 0.175205638142869 | 0.29327284093695 |