A tour through mathematical methods on systems telemetry

Math in Big Systems

If it was a
simple math problem,
we'd have
solved all this by now.

Please ask questions via the mobile app!

Engage

The many faces of

Theo Schlossnagle

@postwait

CEO Circonus

Picking

an Approach

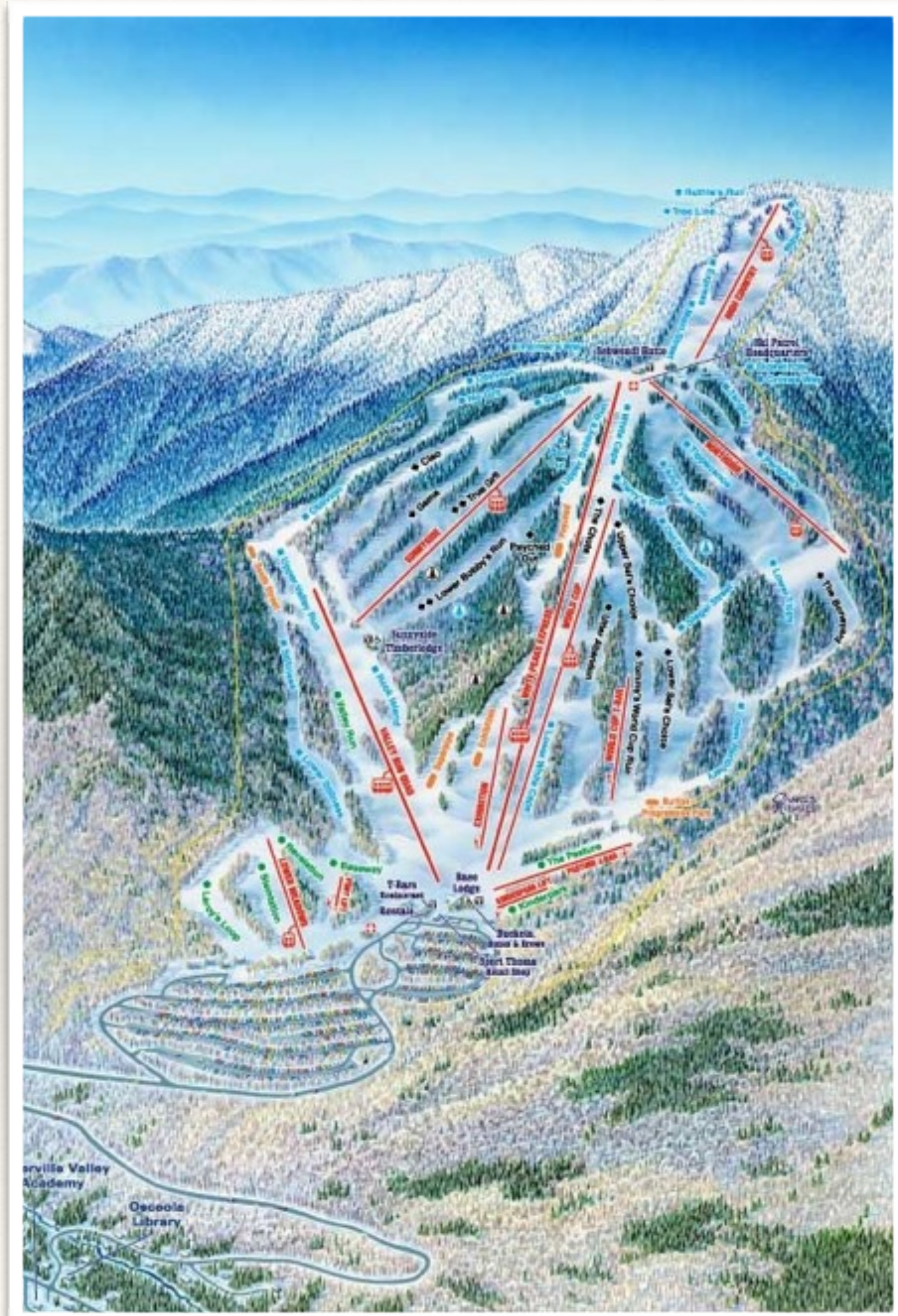
Statistical?

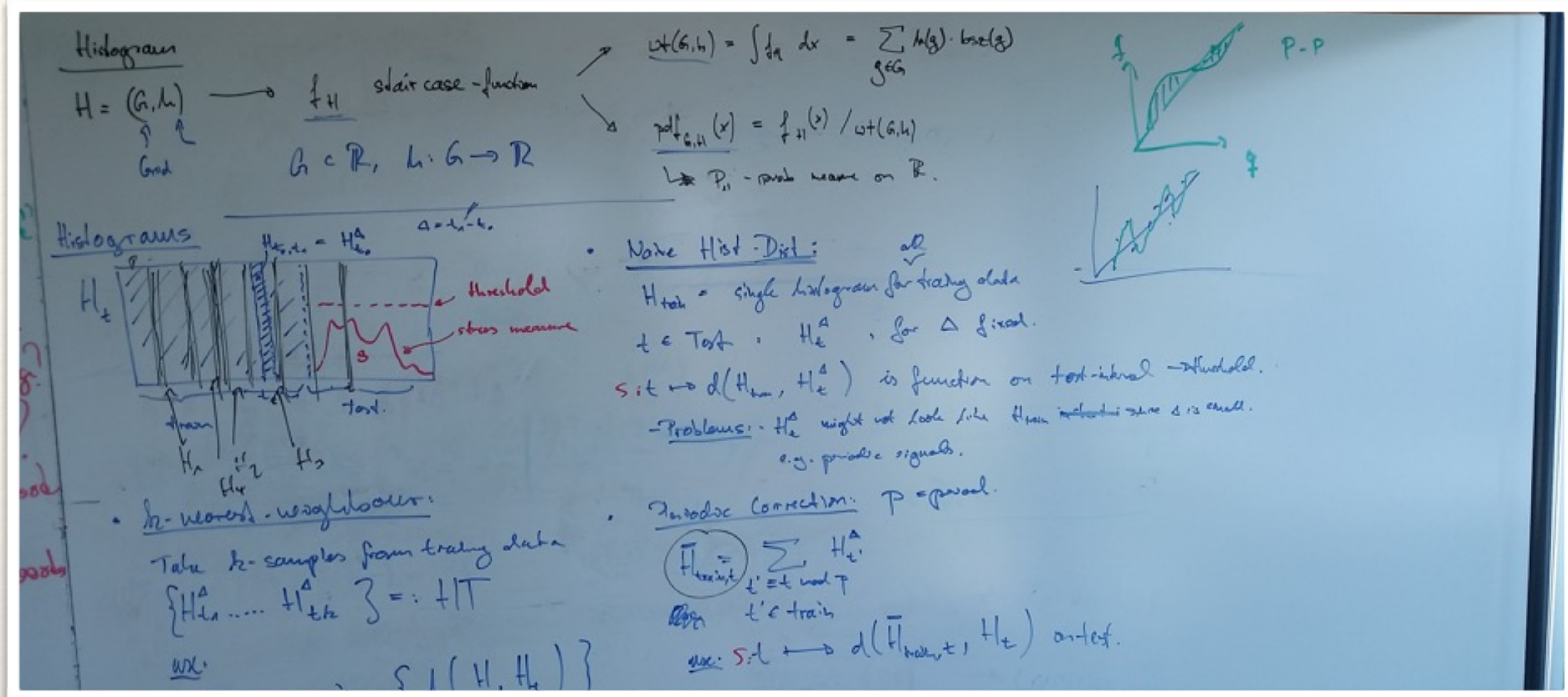
Machine learning?

Supervised?

Ad-hoc?

ontological? (why it is what it is)





tl;dr

Apply PhDs

- Apply PhDs
- Rinse
- Wash
- Repeat

Garbage in, category out.

Classification

Understanding a signal

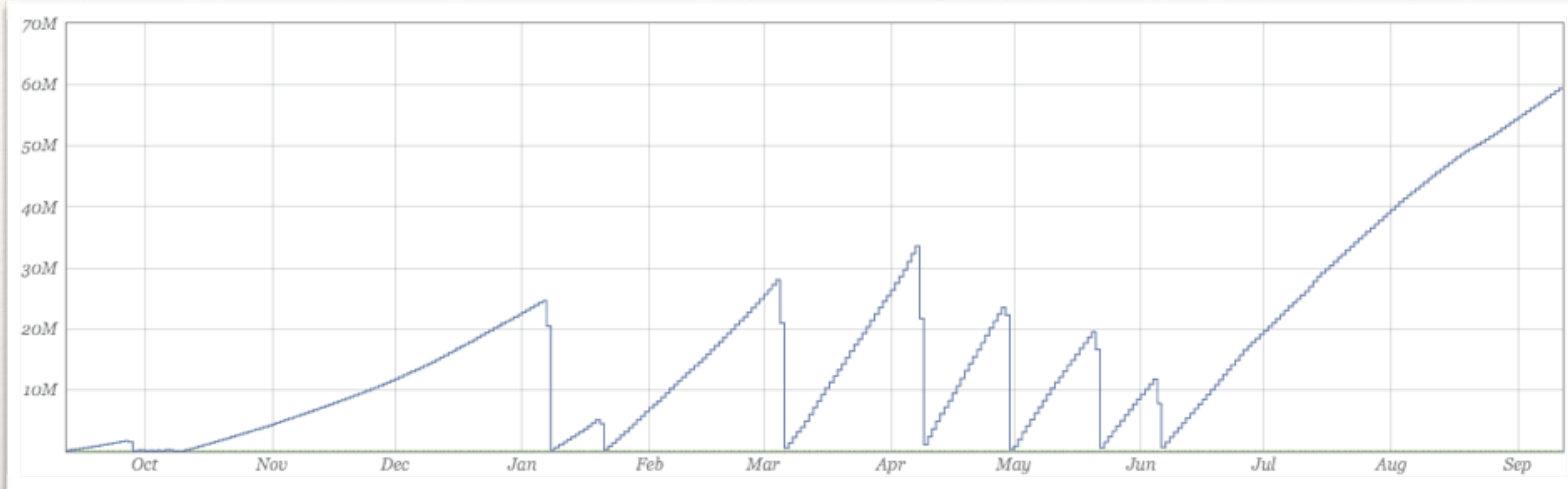
We found to be quite ad-hoc

At least the feature extraction

A year of service… I should be able to learn something.

API requests/second

1 year



A year of service… I should be able to learn something.

API requests

1 year

A year of service… I should be able to learn something.

API requests $\frac{\Delta v}{\Delta t}, \forall \Delta v \geq 0$

1 year

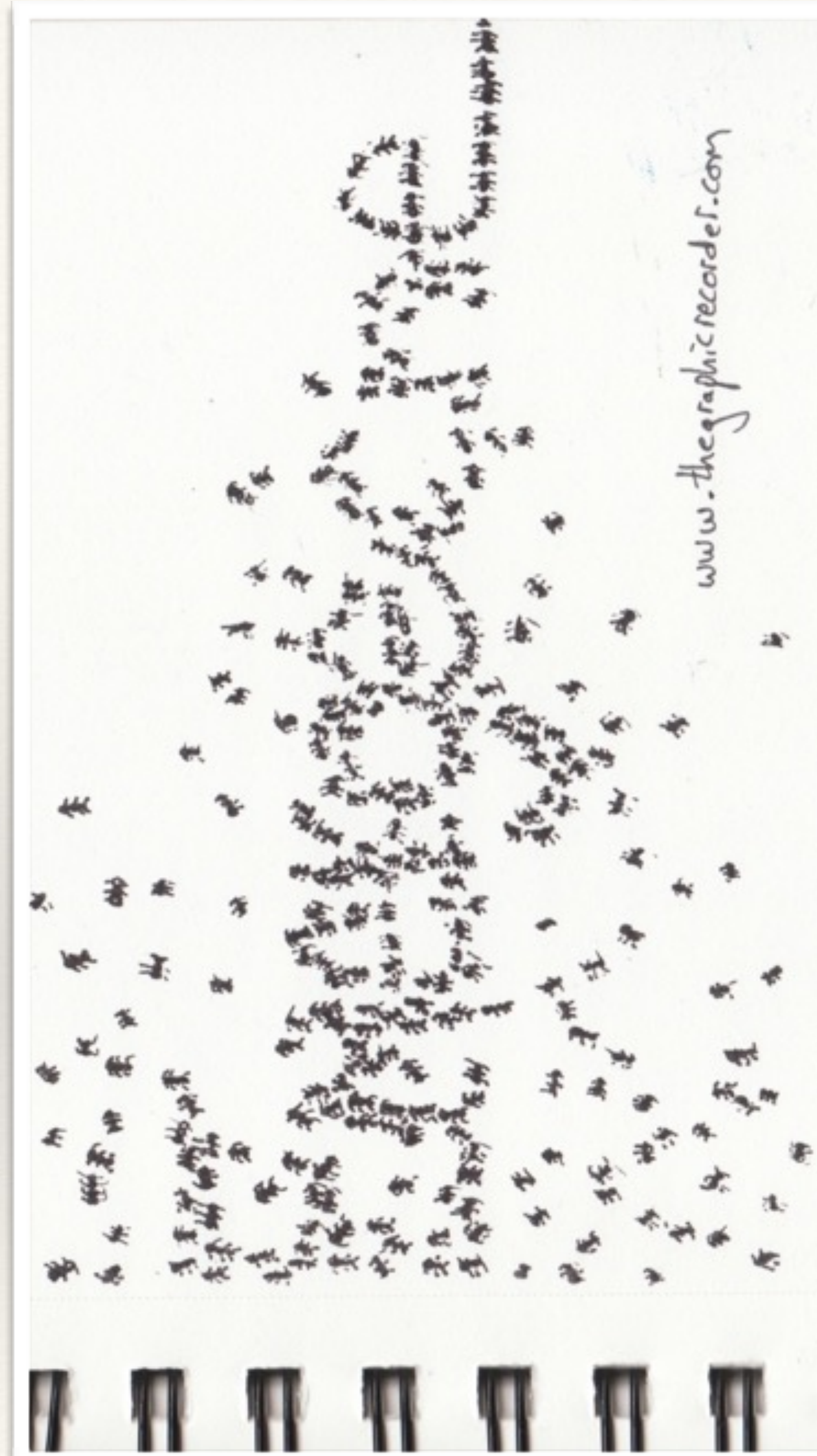
Some data goes both ways…

Complicating Things

Imagine disk space used…

it makes sense as a gauge (how full)

it makes sense as rate (fill rate)



Error + error + guessing = success

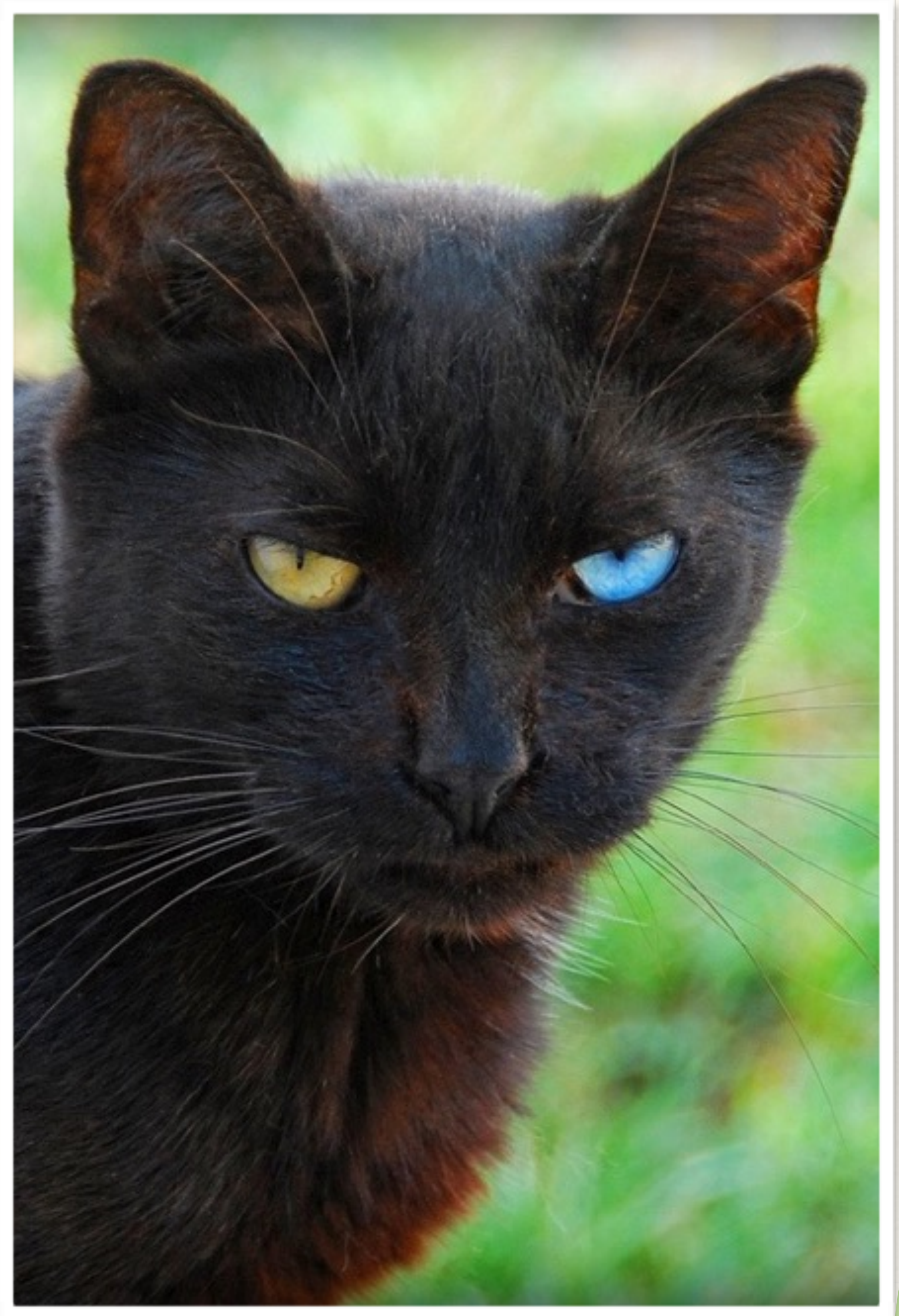
How we categorize

Human identify a variety of categories.

Devise a set of ad-hoc features.

Bayesian model of features to categories.

Human tests.



https://www.flickr.com/photos/chrisyarzab/5827332576

Many signals have significant noise around their averages

Signal Noise

A single "obviously wrong" measurement…

is often a reasonable outlier.

A year of service… I should be able to learn something.

API requests/second

1 year

At a resolution where we witness: "uh oh"

API requests/second

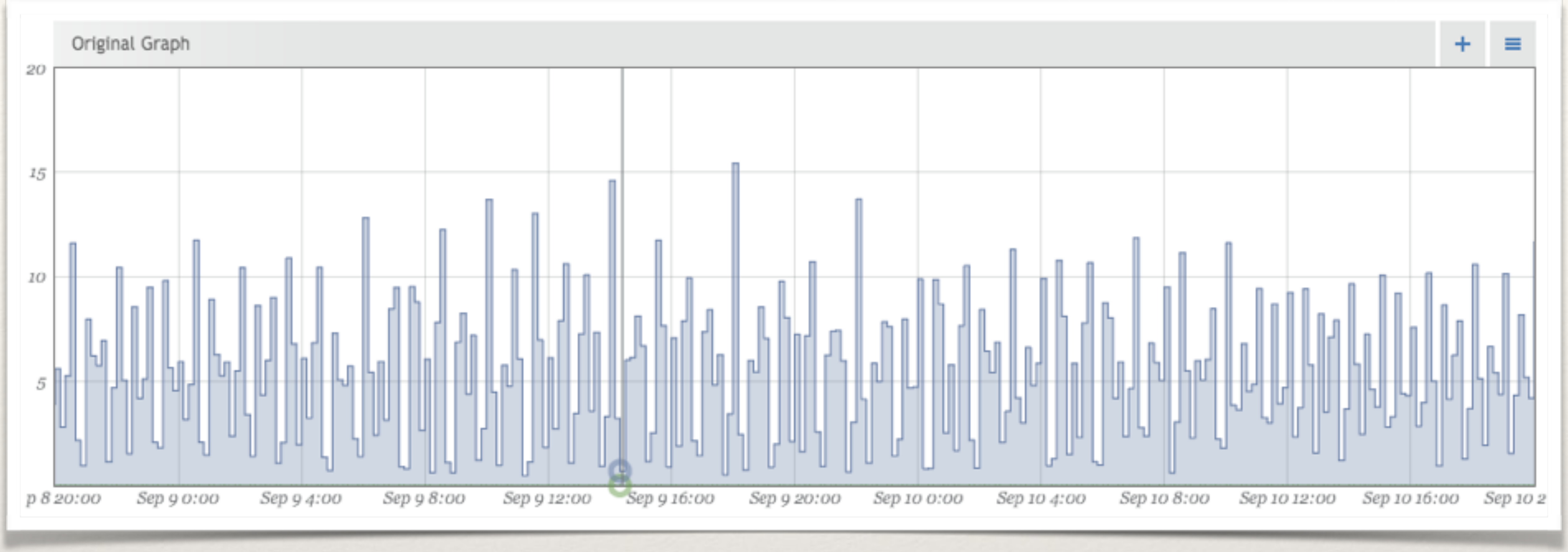
4 weeks

Is that super interesting?

But, are there two? three?

API requests/second

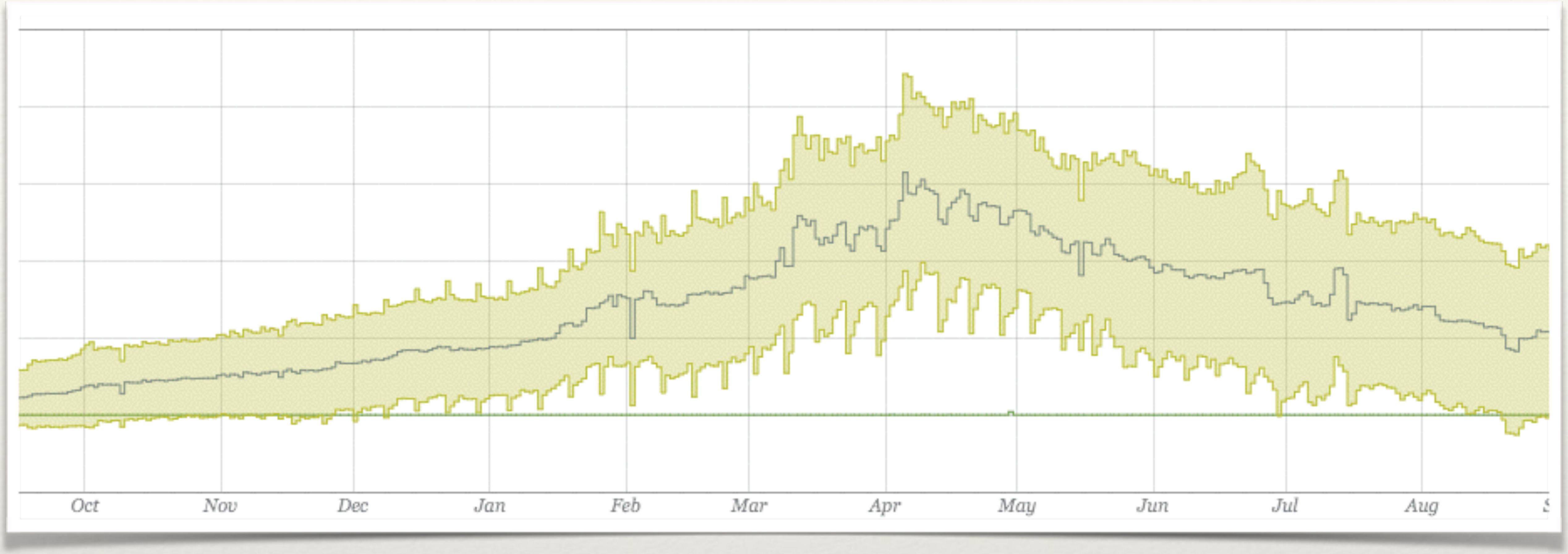
4 weeks



Bring the noise!

API requests/second

2 days



Think about what this means… statistically

API requests/second

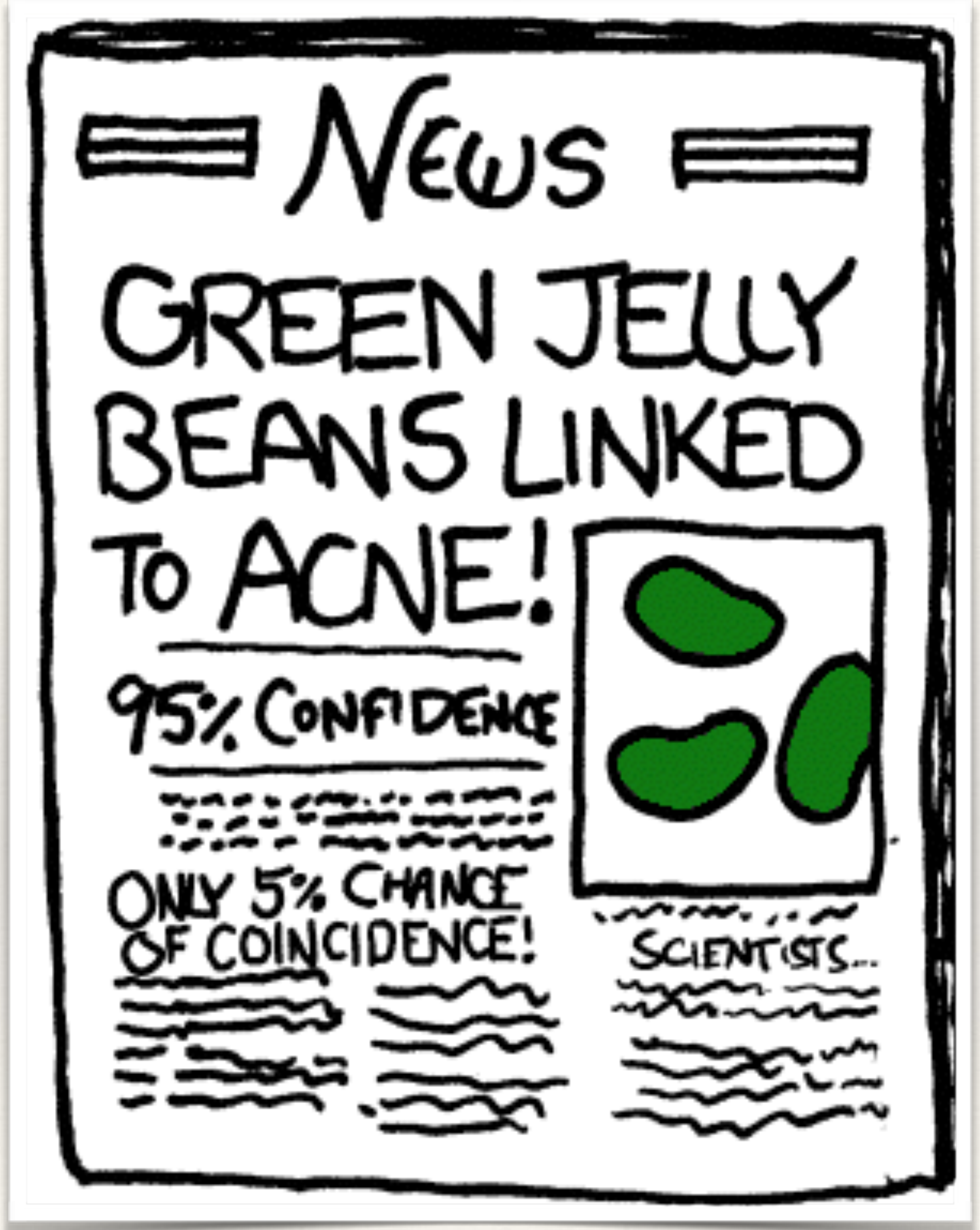
1 year
envelope of ±1 std dev

Lies, damned lies, and statistics

Simple Truths

Statistics are only really useful with p-values are low.

$p \leq 0.01$ very strong presumption against null hyp.
$0.01 < p \leq 0.05$ strong presumption against null hyp.
$0.05 < p \leq 0.1$ low presumption against null hyp.
$p > 0.1$ no presumption against the null hyp.



from xkcd #882 by Randall Munroe

What does a p-value have to do with applying stats?

The p-value problem

It turns out a lot of measurement data (passive) is very infrequent.

Our low frequencies lead us to

questions of doubt…

Given a certain statistical model:

How many few points need to be seen before we are sufficiently confident that it does not fit the model (presumption against the null hypothesis)?

With few, we simply have outliers or insignificant aberrations.

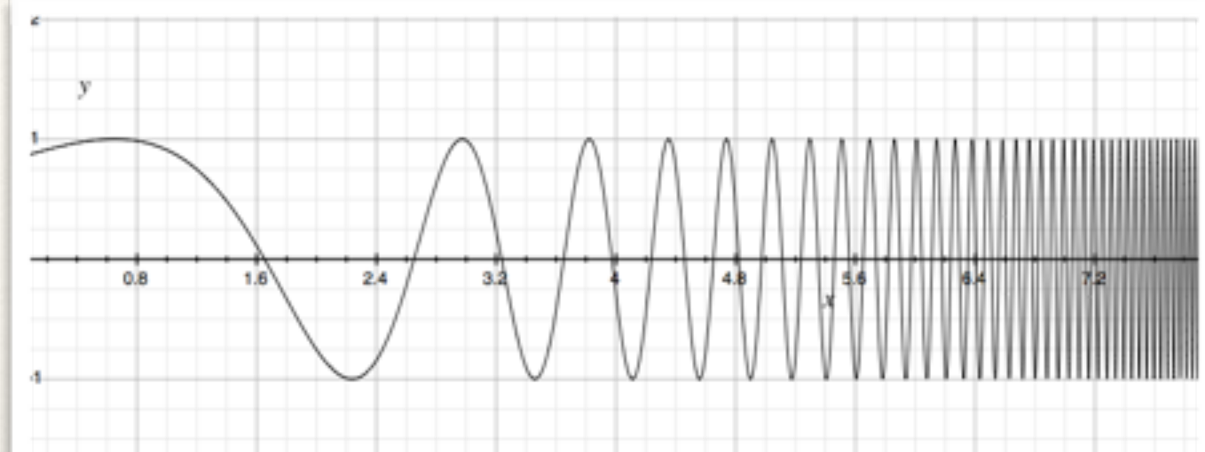
http://www.flickr.com/photos/rooreynolds/

Solving the Frequency Problem

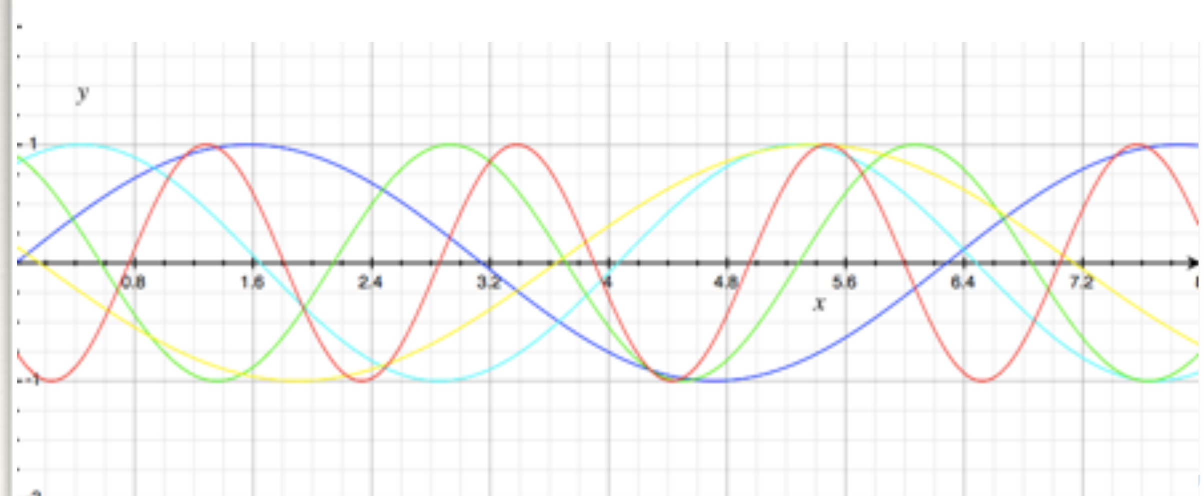
More data, more often…
(obviously)

1. sample faster
   (faster from the source)

2. analyze wider
   (more sources)



OR



Increasing frequency is the only option at times.

Signals of Importance

Without large-scale systems

We must increase frequency

Most algorithms require measuring residuals from a mean

Mean means

Calculating means is "easy"

There are some pitfalls

Newer data should influence our model.

Signals change

The model needs to adapt.

Exponentially decaying averages are quite common in online control systems and used as a basis for creating control charts.

Sliding windows are a bit more expensive.

Repeatable outcomes are needed

In our system…

We need our online algorithms to match our offline algorithms.

This is because human beings get pissed off when they can't repeat outcomes that woke them up in the middle of the night.

EWM: not repeatable
SWM: expensive in online application

Repeatable, low-cost sliding windows

Our solution: lurching windows

fixed rolling windows
of
fixed windows

actual math

Putting it all together

How to test if we don't match our model?

Hypothesis Testing

Take $b_0 < b_1 \in \mathbb{R}$, a time series y and $\theta = b_0, \theta' = b_1$. To test the hypothesis

- H_0: The instance y was drawn from $\mathbb{Y}_t = b_0 + \epsilon_t$
- H_1: The instance y was drawn from $\mathbb{Y}_t = b_1 + \epsilon_t$

we calculate the likelihood ratios as:

$$
\begin{aligned}
\lambda &= log(\frac{1/\sqrt{2\pi}exp(-\frac{1}{2}\sum_t(y_t - b_1)^2)}{1/\sqrt{2\pi}exp(-\frac{1}{2}\sum_t(y_t - b_0)^2)}) \\
&= \frac{1}{2}\sum_t (y_t - b_0)^2 - (y_t - b_1)^2 \\
&= \frac{1}{2}\sum_t (y_t - \bar{b} + \delta)^2 - (y_t - \bar{b} - \delta)^2 \\
&= 2\delta \sum_t (y_t - \bar{b}) \\
&= 2\delta T(\bar{y} - \bar{b})
\end{aligned}
$$

where $\bar{b} = \frac{1}{2}(b_0 + b_1)$ is the average of b_0, b_1, the variable $\delta = \frac{1}{2}(b_1 - b_0)$ is half the distance between b_1 and b_0 and $\bar{y} = \frac{1}{T}\sum_t y_t$ is the sample mean. For the second step we used the following simple identity:

$$(a+b)^2 - (a-b)^2 = 4ab$$

Note, that $\lambda = 0$ if $\bar{b} = \bar{y}$. And $\lambda > 0$ if $\bar{y} - \bar{b} > 0$, i.e. $\bar{y}$ is closer to b_1 than to b_0.

We accept the Hypothesis H_1 if $\lambda > log(\alpha)$, which is equivalent to:

$$\bar{y} - \bar{b} > \frac{log(\alpha)}{2\delta T}$$

The CUSUM Method

It turns out, that there is a simple recursion, which allows us to compute the likelyhood ratio $\lambda(T+1)$ for an instance $y_1, \ldots, y_T$ of length $T+1$ from the knowledge of $\lambda(T)$ for the instance $(y_1, \ldots, y_T)$ of length T.

Indeed, we have

$$\lambda(T+1) = S_1^{T+1} - \min_{k=1,\ldots,T+1} S_1^k.$$

Note, that $S_1^{T+1} = S_1^T + 2\delta(y_t - \bar{b})$.

The minimum-term we have

$$\min_{k=1,\ldots,T+1} S_1^k = \begin{cases} S_1^{T+1} & (A) \\ \min_{k=1,\ldots,T} S_1^k & (B) \end{cases}.$$

In case (A) we have $\lambda(T+1) = 0$ and in case (B) :

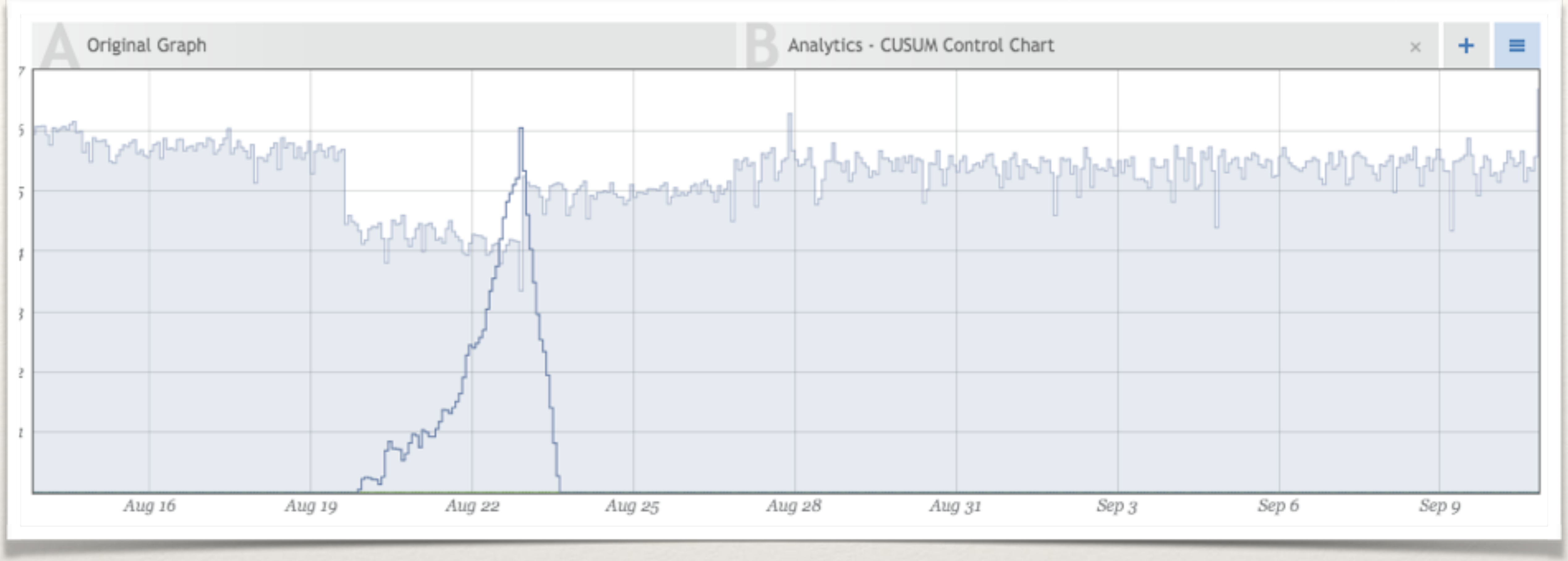
$$\lambda(T+1) = \lambda(T) + 2\delta(y_{T+1} - \bar{b}).$$

Since we always have $\lambda(T) \geq 0$, we get the total recursion:

$$\lambda(T+1) = \max\{0, \lambda(T) + 2\delta(y_{T+1} - \bar{b})\}$$

Set $g(T) := \lambda(T)/(2\delta)$ then we get the slightly simpler recursion:

$$g(T+1) = \max\{0, g(T) + (y_{T+1} - b_0 - \delta)\}$$



Applying CUSUM

API requests/second

4 weeks
CUSUM Control Chart

Can we do better?

Investigations

The CUSUM method has some issues. It's challenging when signals are noise or of variable rate.

We're looking into the Tukey test:

- compares all possible pairs of means
- test is conservative in light of uneven sample sizes

https://www.flickr.com/photos/st3f4n/4272645780

High volume data requires a different strategy

What happens when we get what we asked for?

10k measurements/second?
more? on each stream…

with millions of streams.

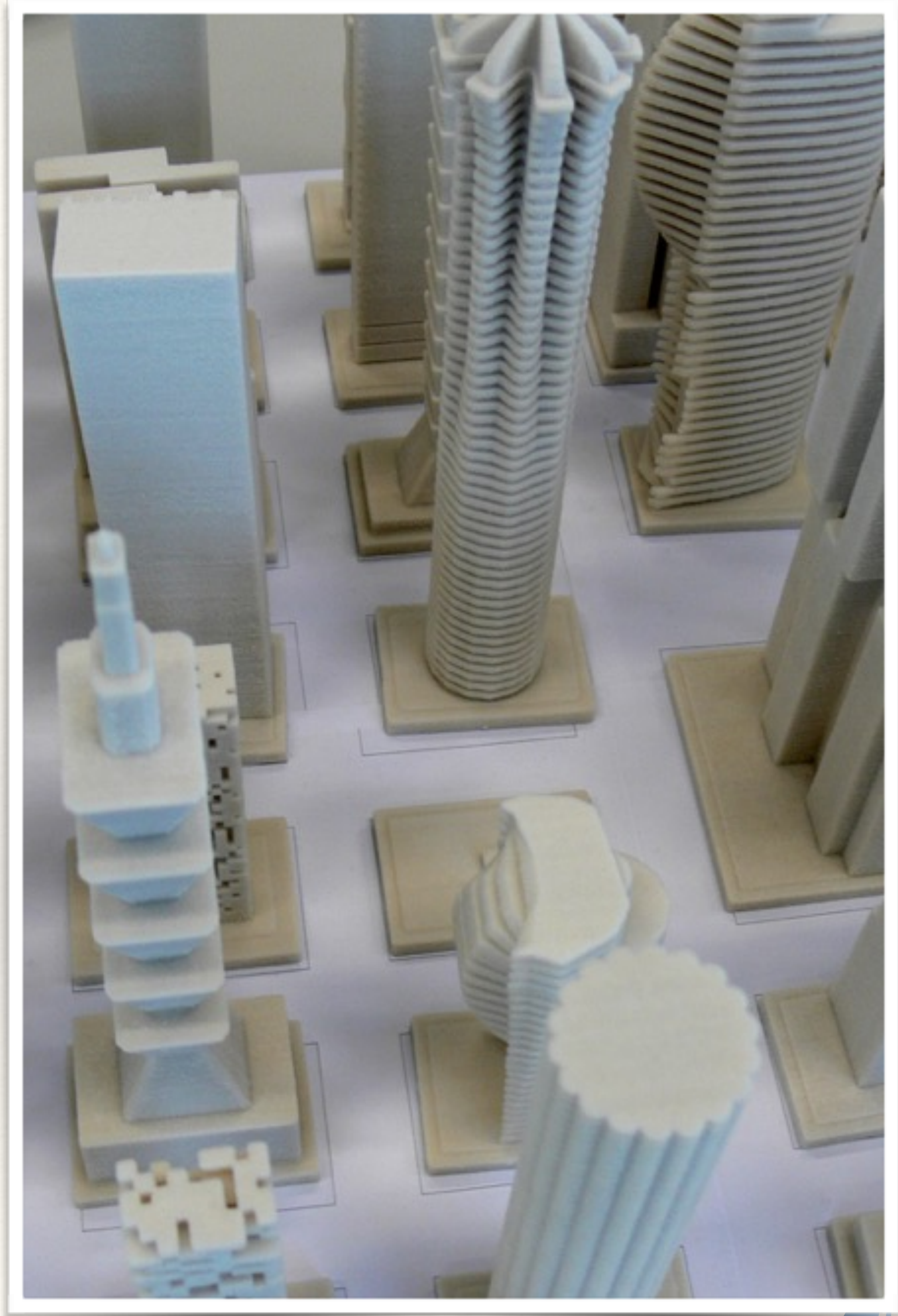
Let's understand the scope of the problem.

First some realities

This is 10 billion to 1 trillion measurements per second.

At least a million independent models.

We need to cheat.



https://www.flickr.com/photos/thost/319978448

When we have to much, simplify…

https://www.flickr.com/photos/meddygarnet/3085238543

Information compression

We need to look at a transformation of the data.

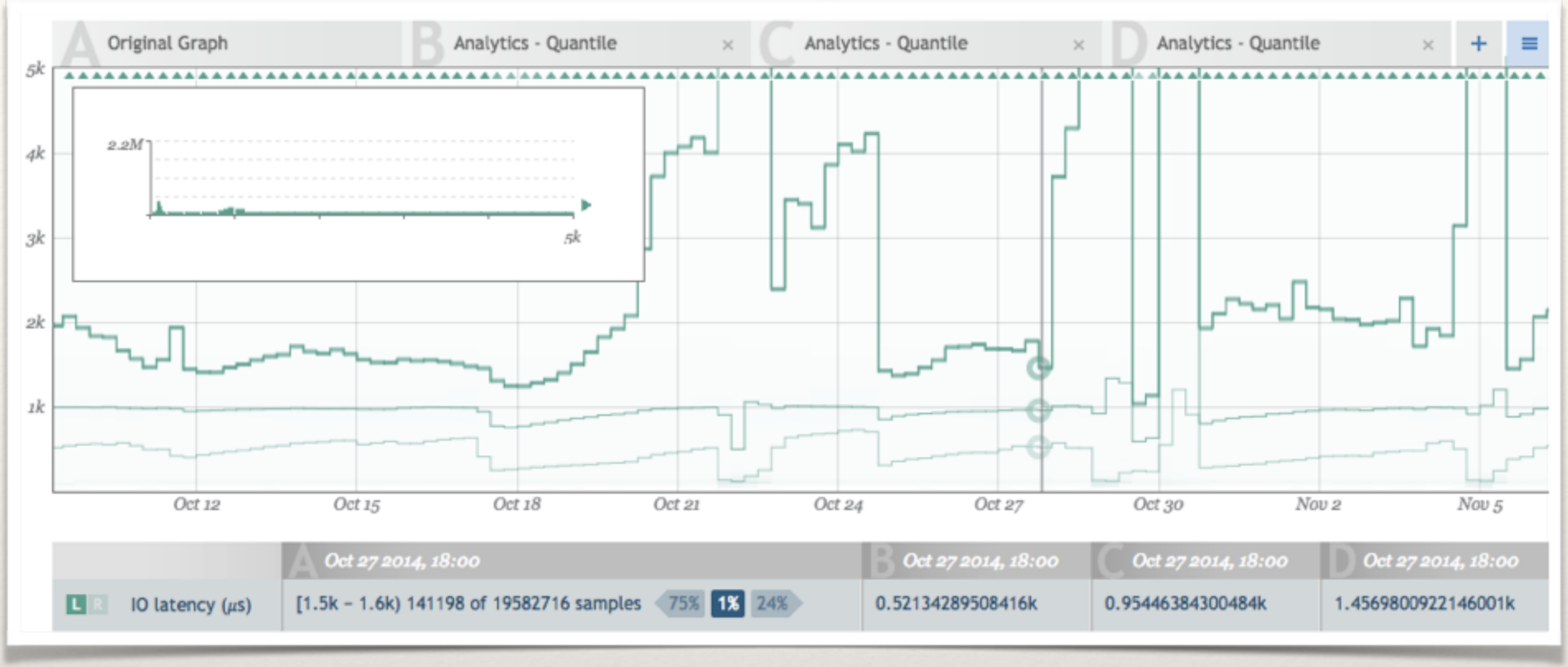
Add error in the value space.
Add error in the time space.

Summarization & Extraction

- Take our high-velocity stream
- Summarize as a histogram over 1 minute (error)
- Extract useful less-dimensional characteristics
- Apply CUSUM and Tukey tests on characteristics

Modes & moments.

Strong indicators of shifts in workload



Quantiles…

Useful if you understand the problem domain and the expected distribution.

Q: "What quantile is 5ms of latency?"

Inverse Quantiles…

Useful if you understand the problem domain and the expected distribution.

Please evaluate this talk via the mobile app!

Engage

BERLIN INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE