Five-sigma Network Events (and how to find them)

John O'Neil
Edgewise Networks
Halloween, 2018

Networks are Complex

- No one knows what's going on

Finding the Strange & Unusual

- Or the new & unexpected 🎃
- …and if it's different, it might be bad. ☠️
- Outlier — Improbable data point in the expected distribution 🦔
- Anomaly — Data point generated by a different distribution 👽

Mr. Splanky

Anomaly & Outlier Tools

"If you want something done right, do it yourself."
— Charles-Guillaume Étienne

Using Python

- Interpretable pseudocode
- Mature libraries.
- Easy to install
- Fast enough 😀

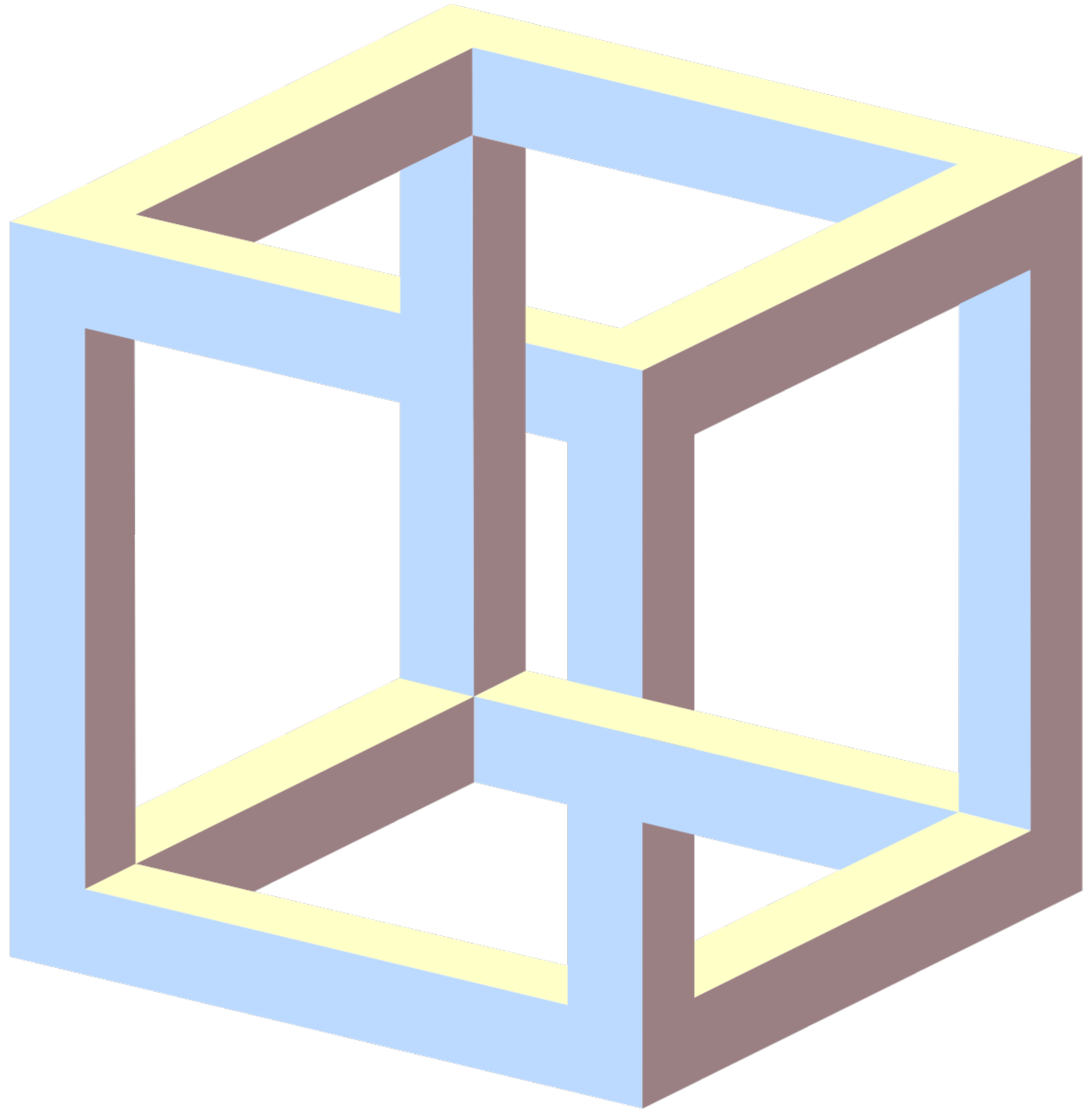
Creating Tools for Outlier Detection

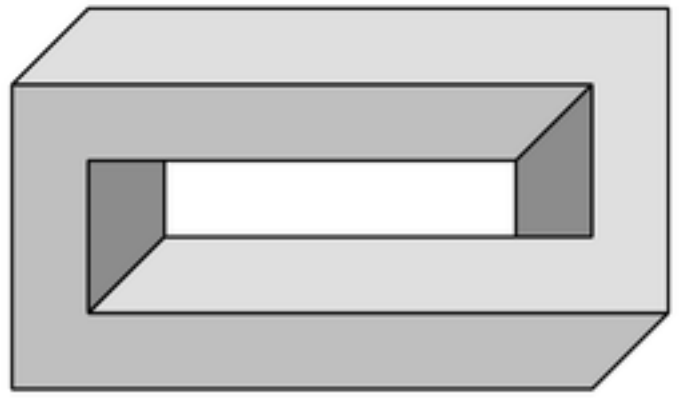
- Introducing a few tools written in Python
- Intended to answer interesting questions and scale well
- Easy to modify/improve to satisfy your curiosity
- A starting point for your own tools
- Code is available at:

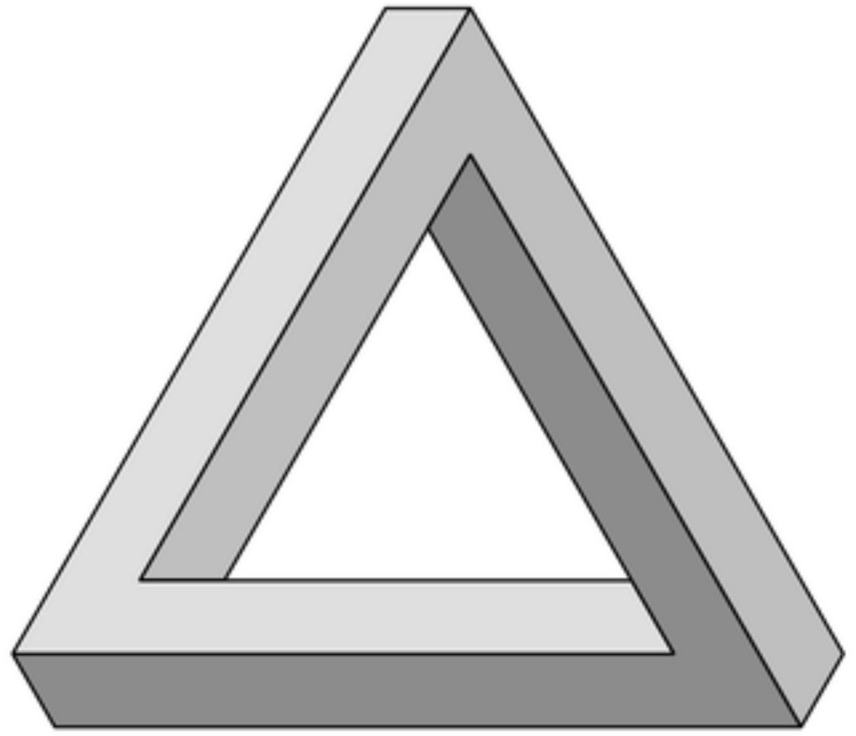
  `http://github.com/EdgewiseNetworks/five-sigma`

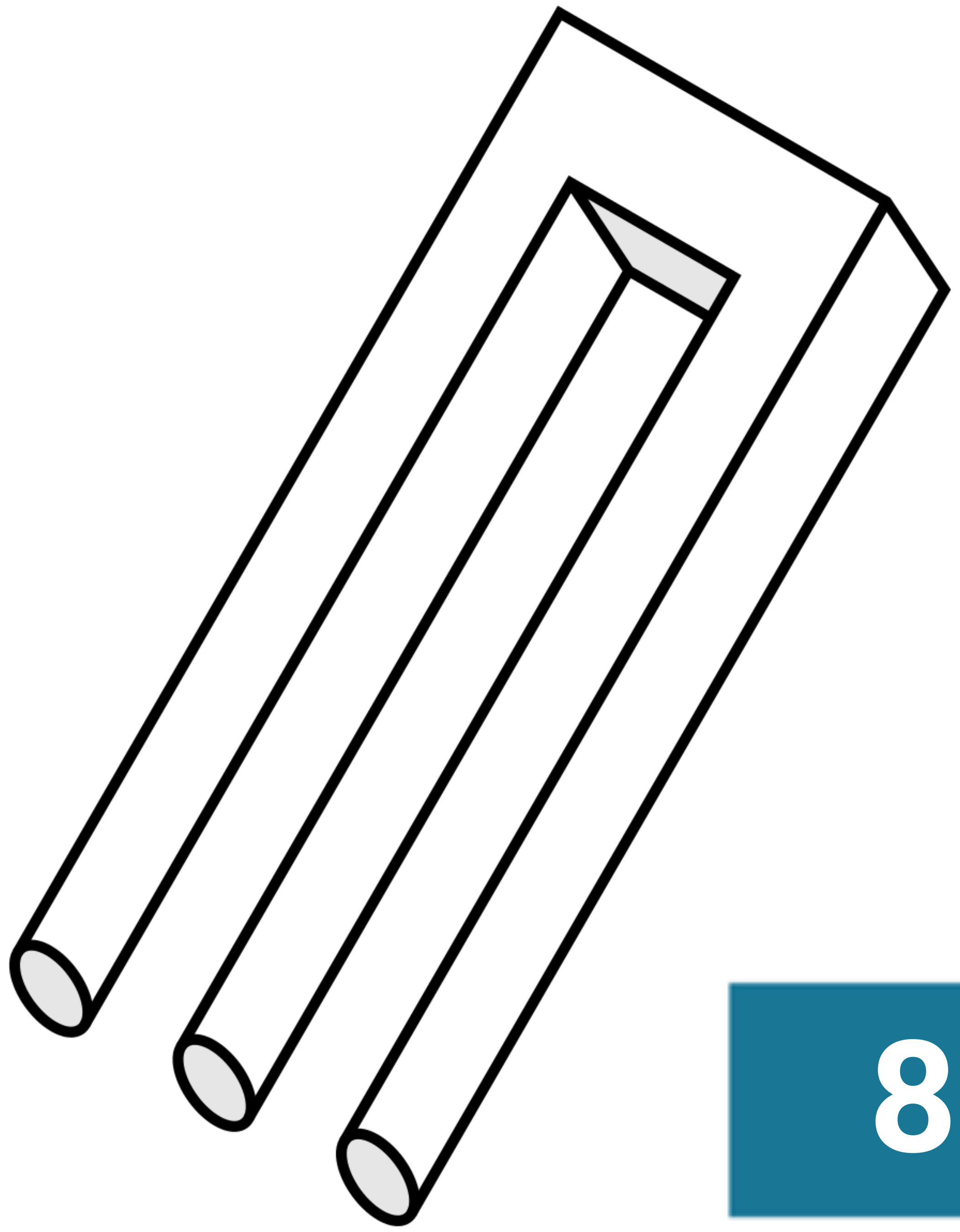
Discover Bad Things Before Big Problems

- Keep track of netflows across machines and across time
- Well enough to recognize unusual things
- But too much information
- And make it tunable







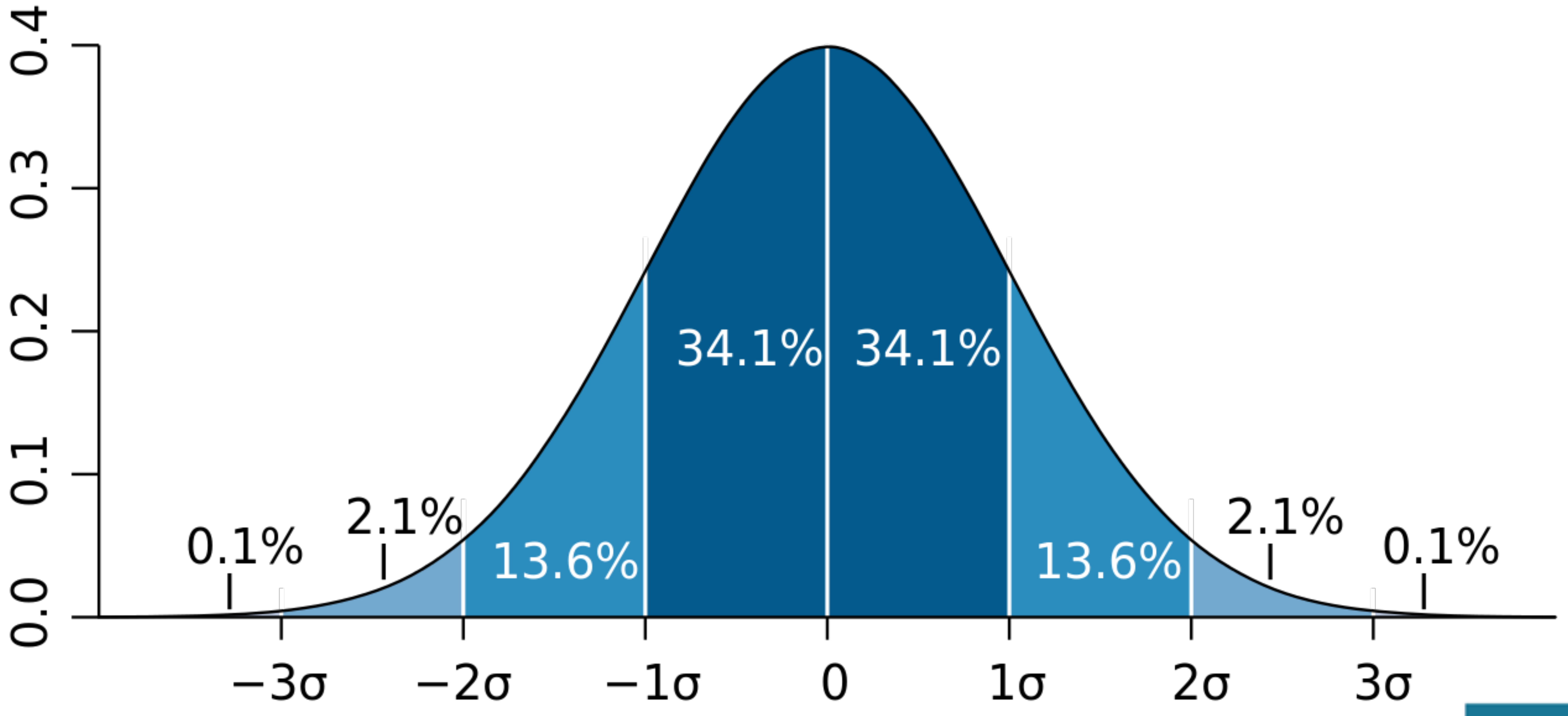


Standard Deviation

The amount of "spread" in a (usually Gaussian) distribution.

$$5\sigma \approx 10^{-6}$$

$$3.29\sigma \approx 10^{-3}$$



Project Overview

1. Create a feed of typical netflows
   - Based on real netflows but anonymized
   - Format: *timestamp*, *src_ip, src_port*, *dest_ip, dest_port*, *flow_count*
2. Create a consumer for these netflows.
3. Create a number of consumer tools to track interesting statistics.
   - Standard deviations
   - Update period

Examples Of Useful Information

1. Does an IP address keep scanning for new open ports?
2. Did an IP address suddenly get a lot busier than it's ever been in the past?
3. Did an IP address suddenly get a lot busier than any other IP address?
4. Shouldn't this IP address have stopped doing new things by now?

Tools To Use: Sketching & Streaming

- Lots of data to keep track of
- But we're only interested in certain aspects of it
  - Set cardinality — HyperLogLog
  - Incremental means & standard deviations
  - Online linear regression
- Make big data into small data

Other Examples of Approximate Probabilistic Sketches

- Bloom Filter (set membership)
- Count-Min Sketch (counting items)
- MinHash (set intersection)
- Locality-Sensitive Hashing (LSH: nearest neighbors)
- Q-digest/T-digest (quantile distribution — MORE ABOUT THIS LATER)

IpPortScanDetector

Q: Does an IP address keep scanning for new open ports?

Contains: {IP_address : HyperLogLog} map
Each HLL counts distinct IP:port destinations.

```
At each period:
 mean, sigma = Stdev(hll.cardinality() for every HLL)
 For each IP_address & HLL :
   if  HLL.cardinality() > N sigmas above the mean:
     report it.
```

GrowthDetector

Q: Did an IP address suddenly get a lot busier than it's ever been in the past?

Contains:
{IP_address : HyperLogLog} — periodCardinalityMap
  Each HLL counts distinct IP:port destinations over all time.
{IP_address : StdDev} — periodStatisticsMap
  Each StdDev incrementally calculates means and stdevs.

```
At each period:
  For each IP_address & HLL & StdDev:
    currCount = HLL.cardinality()
    mean, sigma = StdDev.getMeanAndStdev()
    if currCount > N sigmas above its mean:
      report it.
    HLL.clear()
    StdDev.add(currCount, current_period)
```

ExplosionDetector

Q: Did an IP address suddenly get a lot busier than any other IP address?

Contains:
{IP_address : HyperLogLog} — periodCardinalityMap
Each HLL counts distinct IP:port destinations in current period.

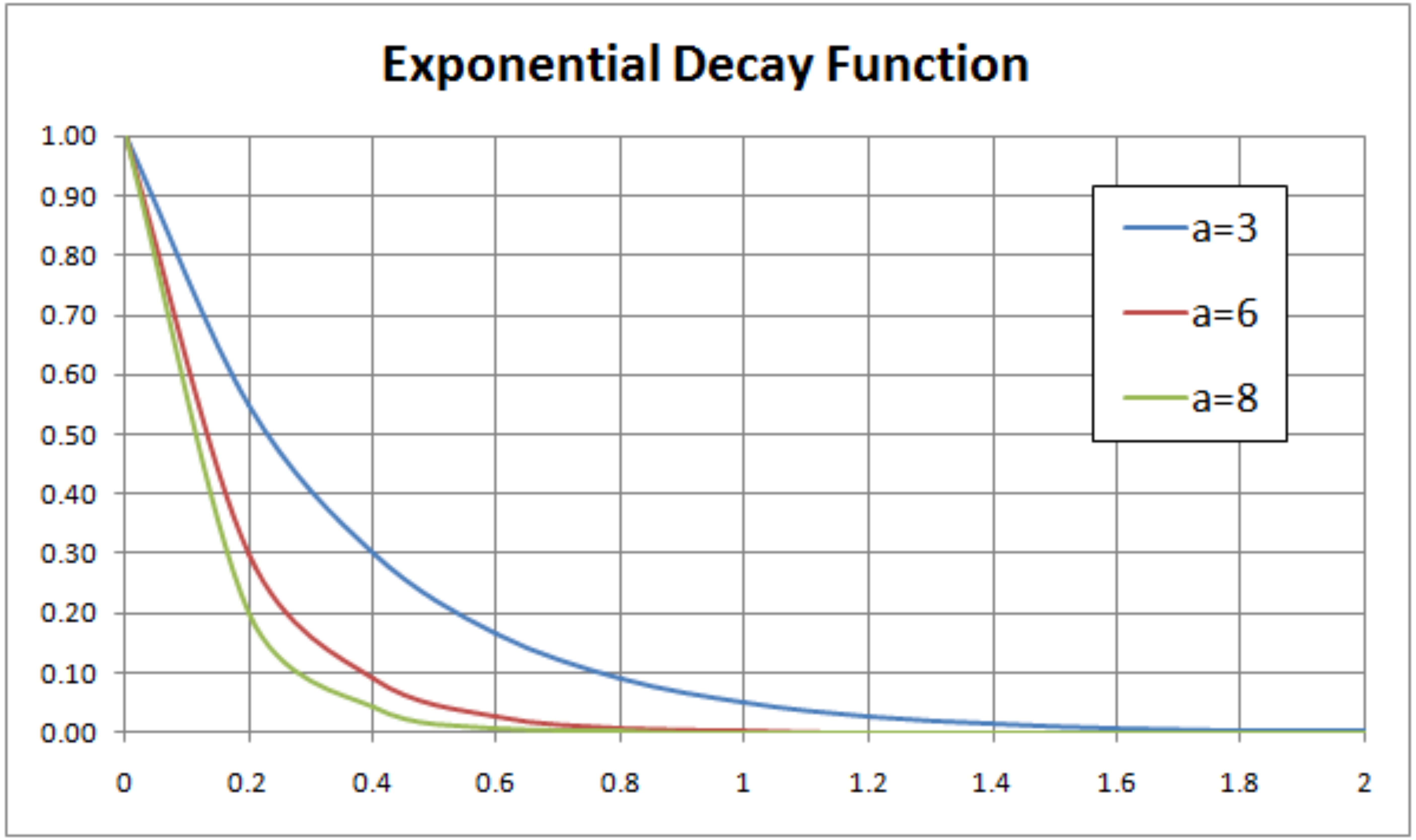
```
At each period:
  mean, sigma = Stdev(hll.cardinality() for every HLL)
  For each IP_address, hll:
    curr = hll.cardinality()
    if curr > N sigmas above the mean:
      report it.
    hll.clear()
```

Host Stabilization

- Assume an "exponential decay" of new IP:port contacts over time
- We know how many we've seen, but not how many are left.
- Can we estimate *N_rem* given *N_obs*? Some calculus later … why yes, we can.

$$N_{rem} \approx -\, slope(x_i) \times avg(x_i)$$

$$y \sim e^{-ax}$$



HostStabilizationDetector

Q: Shouldn't this IP address have stopped doing new things by now?

Contains:
{IP_address : HyperLogLog} — cardinalityMap
Each HLL counts distinct IP:port destinations over all time.
{IP_address : StdDev} — periodAverageMap
Each StdDev incrementally calculates means and stdevs.
{IP_address : IncrLinReg} — IncrementalLinearRegressionMap
Each StdDev incrementally calculates means and stdevs.

```
At each period:
  For each IP_address, HLL, StdDev, IncrLinReg:
    N_obs = HLL.cardinality()
    slope, intercept = IncrLinReg.estimate()
    avg = StdDev.getMean()
    N_rem = -slope * avg
    reportIfDisagree(N_rem < tol, IP_address.frozen)
    IP_address.setFrozen(N_rem < tol)
    {IncrLinReg, StdDev}.update(N_obs, current_period)
```

Demo Time!

But is it Gaussian?

- “Long-tail” or “fat-tail” distributions?
- Try power law or log-linear fitting
  - And many others?
  - But this can get complicated….
- Replace StdDev with tdigest.TDigest

Conclusions

- Without the agonizing pain
- Python data science tools FTW
- Cool sketching & streaming data structures
- *“A little learning is a dangerous thing”*
  … and a little statistics is even better!
- Only the beginning — lots of room for improvement

EDGEWISE NETWORKS

The End
Thanks for attending!

```
http://github.com/EdgewiseNetworks/five-sigma
```

Suggested questions

1. How do I install Python, again?
2. What can I do with *flow_counts* in my netflows?
3. Show me the calculus for estimating N_{rem}!
4. So, what is the real statistical distribution of that data?
5. How does HyperLogLog work?