Statistics for Engineers

SRECon EMEA 2023

Heinrich Hartmann | zalando.de | @HeinrichHartmann

Hi, I'm Heinrich

Bio

Head of SRE @ zalando

Data Scientist @ CIRCONUS

Mathematician @ The University of Bonn
Doctor of Philosophy (PhD), Matchematik
2008 - 2011

Talks & Publications

- Statistics for Engineers (SRECon 2015..2022)
- How to measure Latency (P99 Conf 21)
- State of the Histogram (SLOConf 2021)
- Latency SLOs Done Right (FOSDEM 2019)
- Circllhist - A Histogram Data Structure (arxiv)
- blog: heinrichhartmann.com
- twitter: @HenrichHartmann

Statistics for Engineers

Visualisations

Summary Statistics

SLOs

Sampling

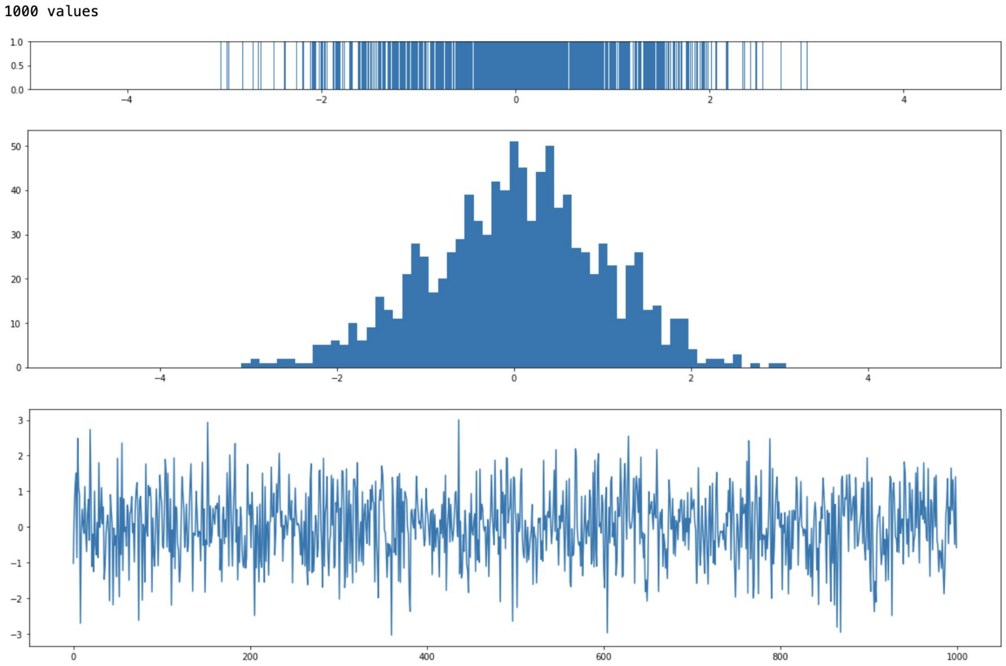
Visualizations

Example Dataset - Normal Noise

Rugplot

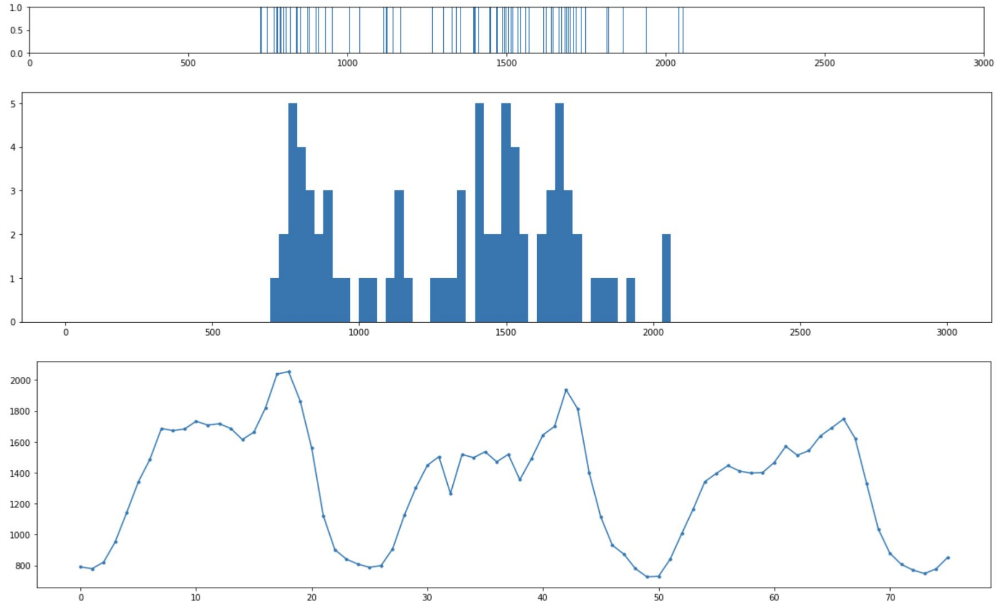
Histogram

Linechart

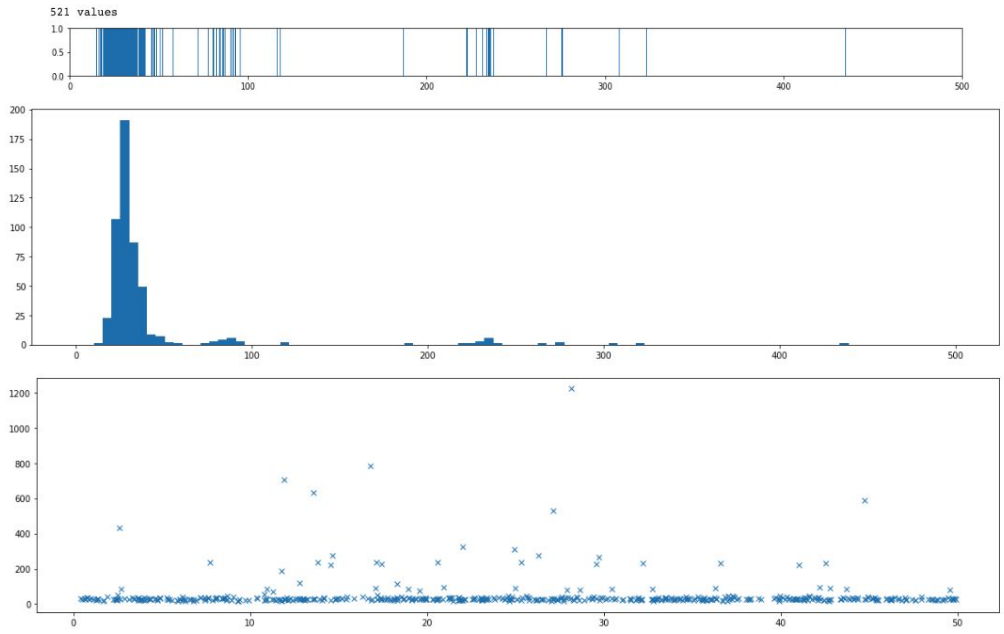


Example Dataset - Request Rates

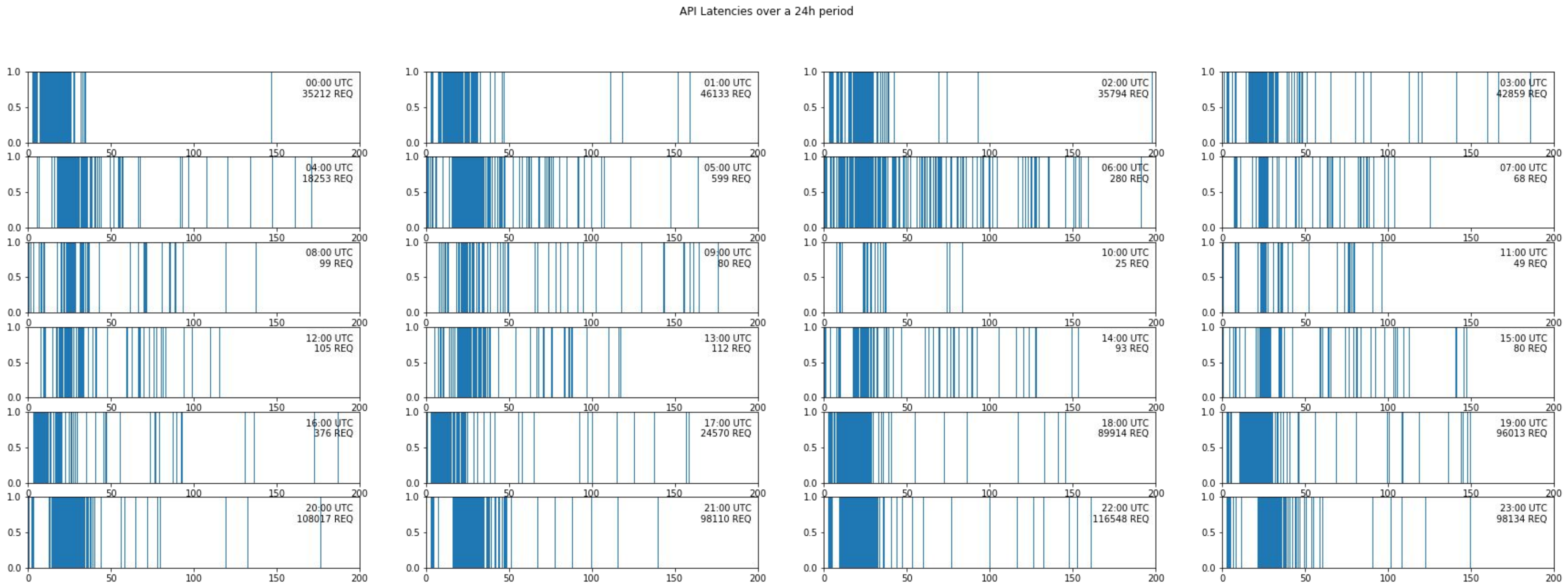
76 values



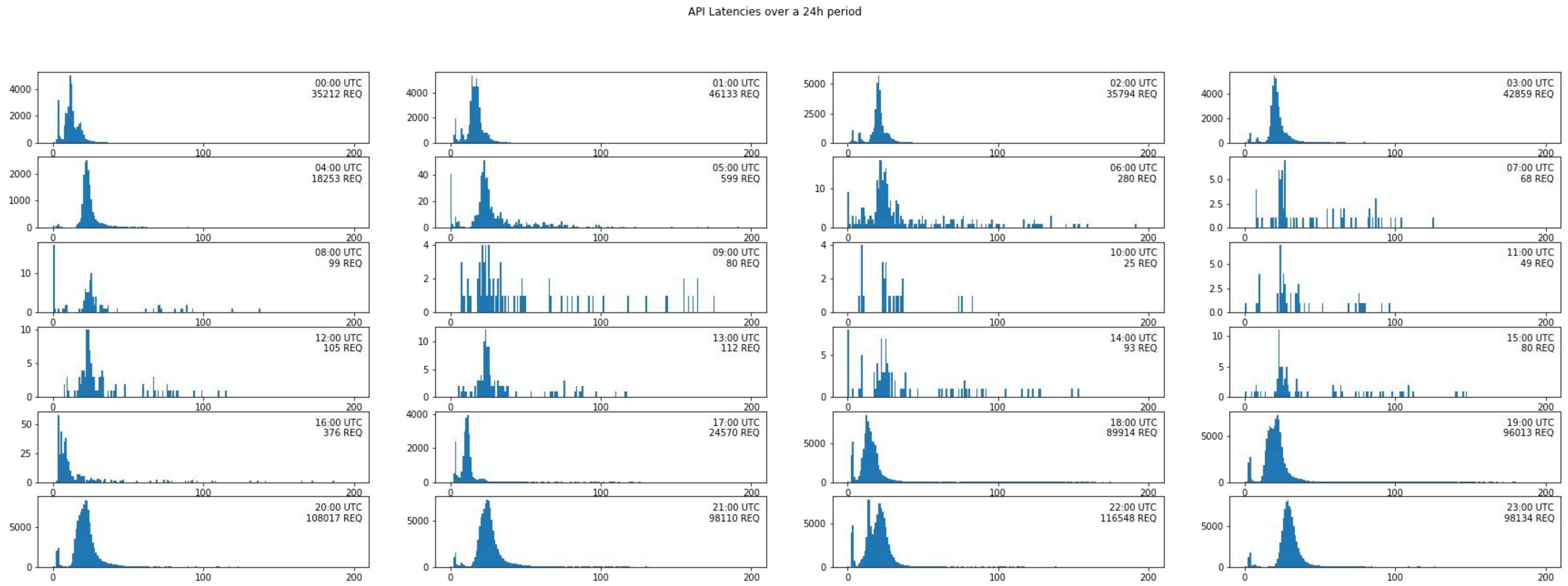
Example Dataset - Request Latencies



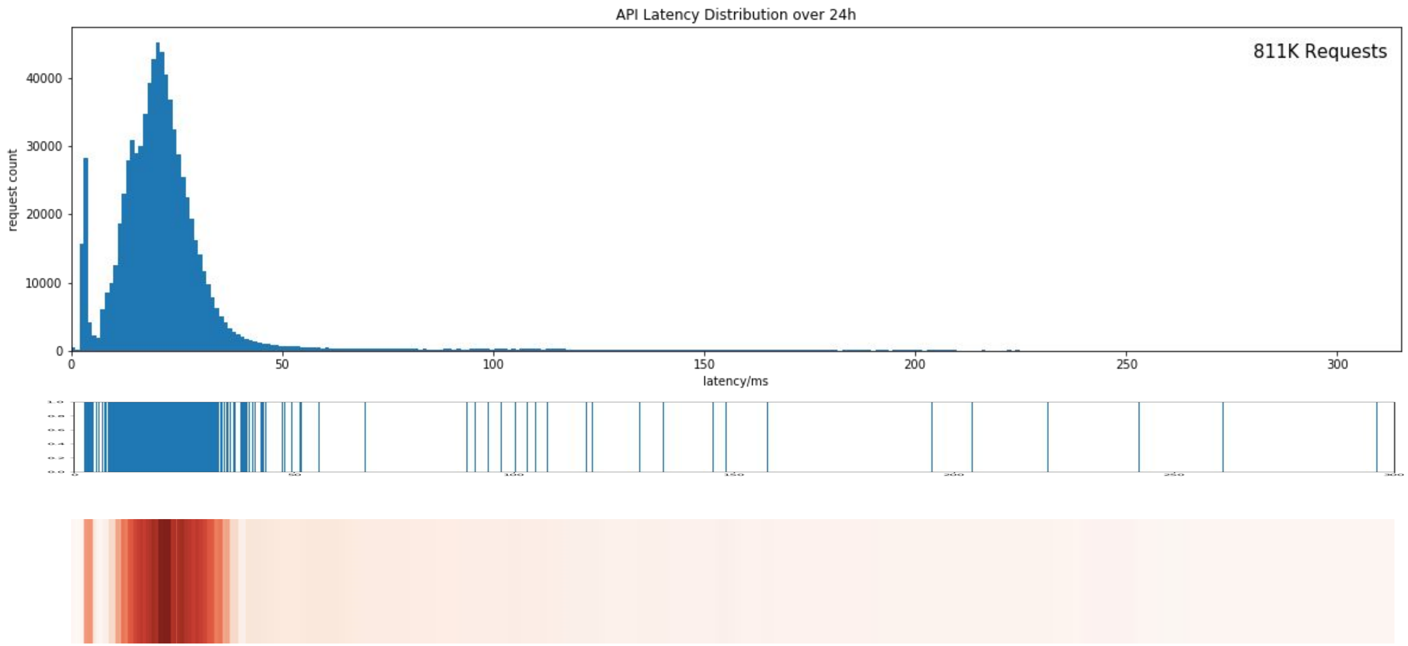
Example Dataset - Request Latencies over Time



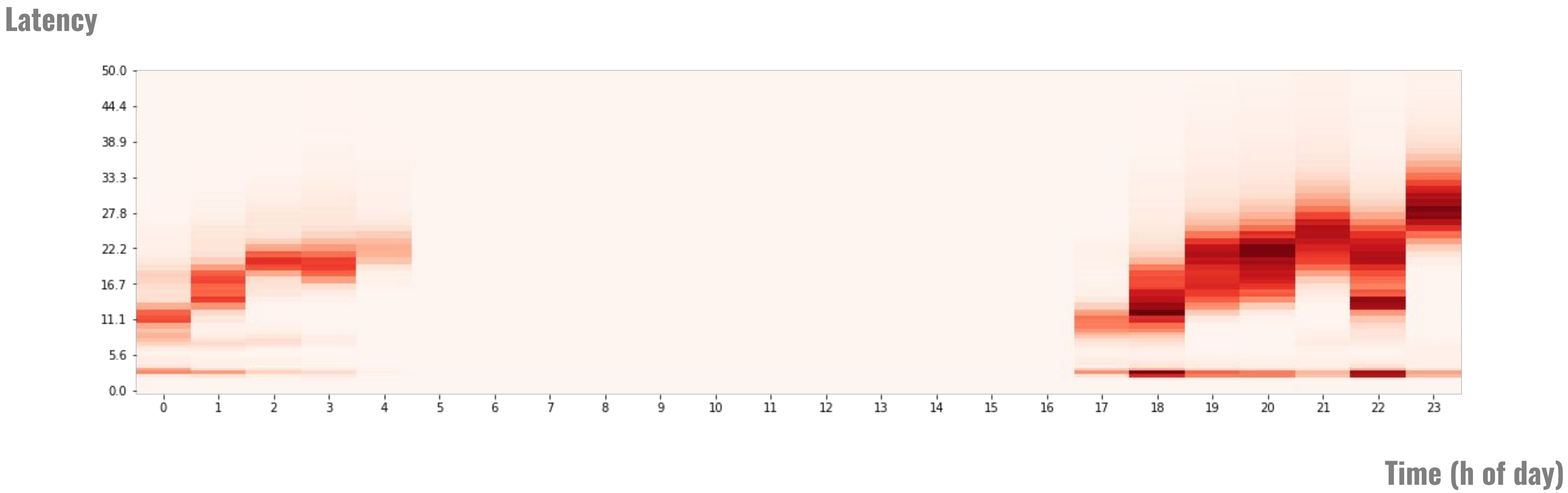
Example Dataset - Request Latencies over Time



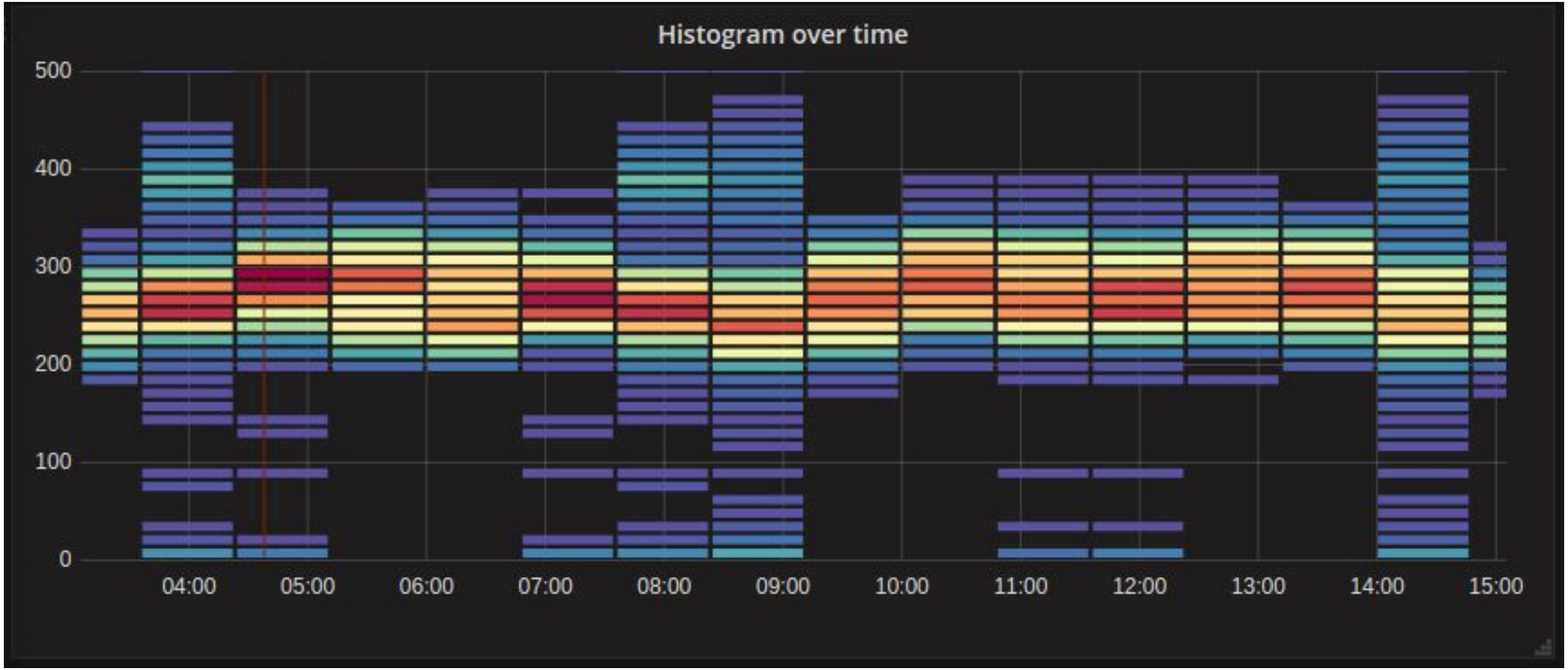
Density Heatmap



Example Dataset - Request Latencies over Time



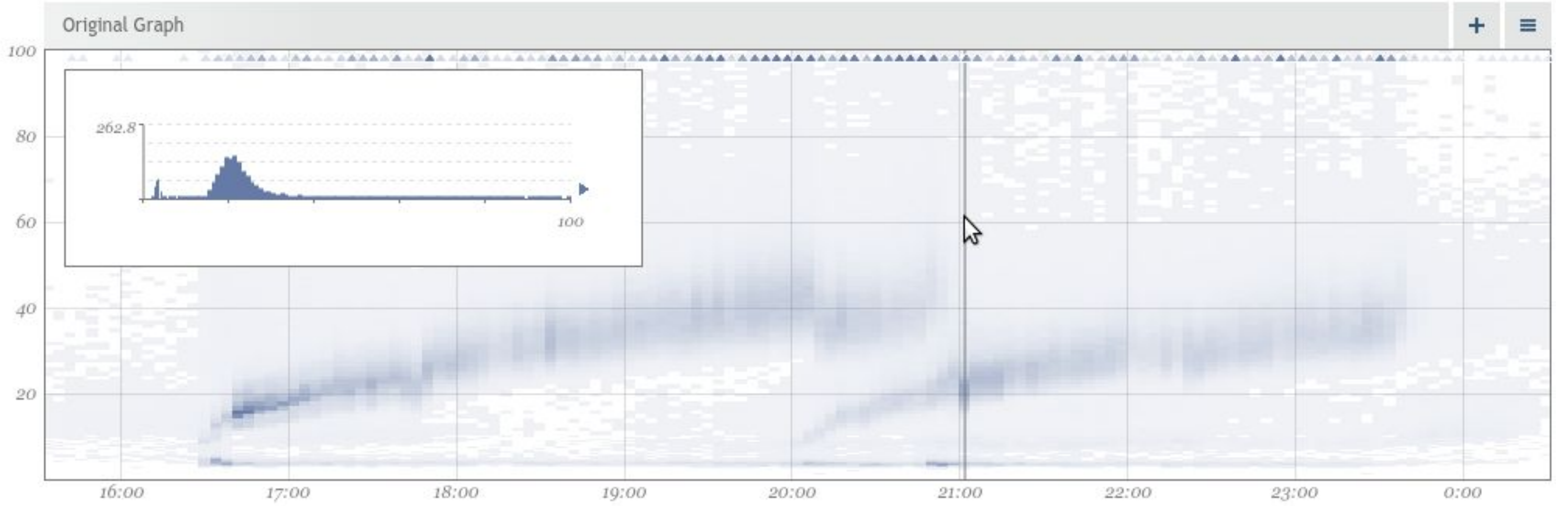
Grafana Histograms



source: https://grafana.com/docs/grafana/latest/fundamentals/intro-histograms/

Grafana

Production Example - Request Latencies over Time



Summary Statistics

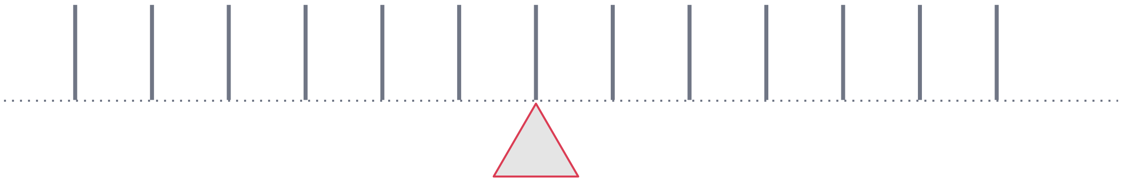
Aggregating Telemetry Data

- Across Time (Graphs, SLOs)
- Across Hosts (www-*)
- Across Endpoints (/, /posts, /archive)

AVERAGES

Summary Statistic - Averages

$$\mu = mean(X) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$



Summary Statistic - Averages

$$\mu = mean(X) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Summary Statistic - Averages

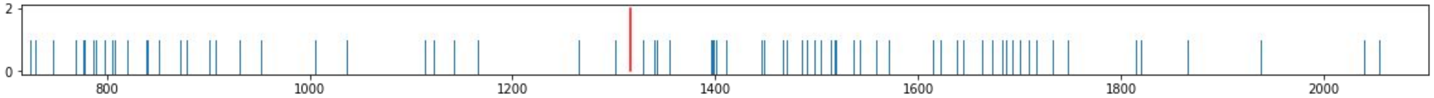
$$\mu = mean(X) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Summary Statistic - Averages

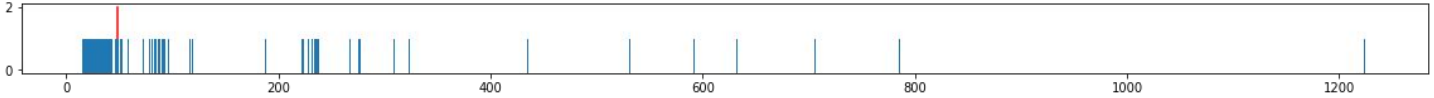
$$\mu = mean(X) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Normal Noise

Request Rates



Request Latency



Can you spot the average?

Spike Erosion - Full Picture

more on blog.circonus.com - Show me the Data (2016)

max throughput = 14 rps

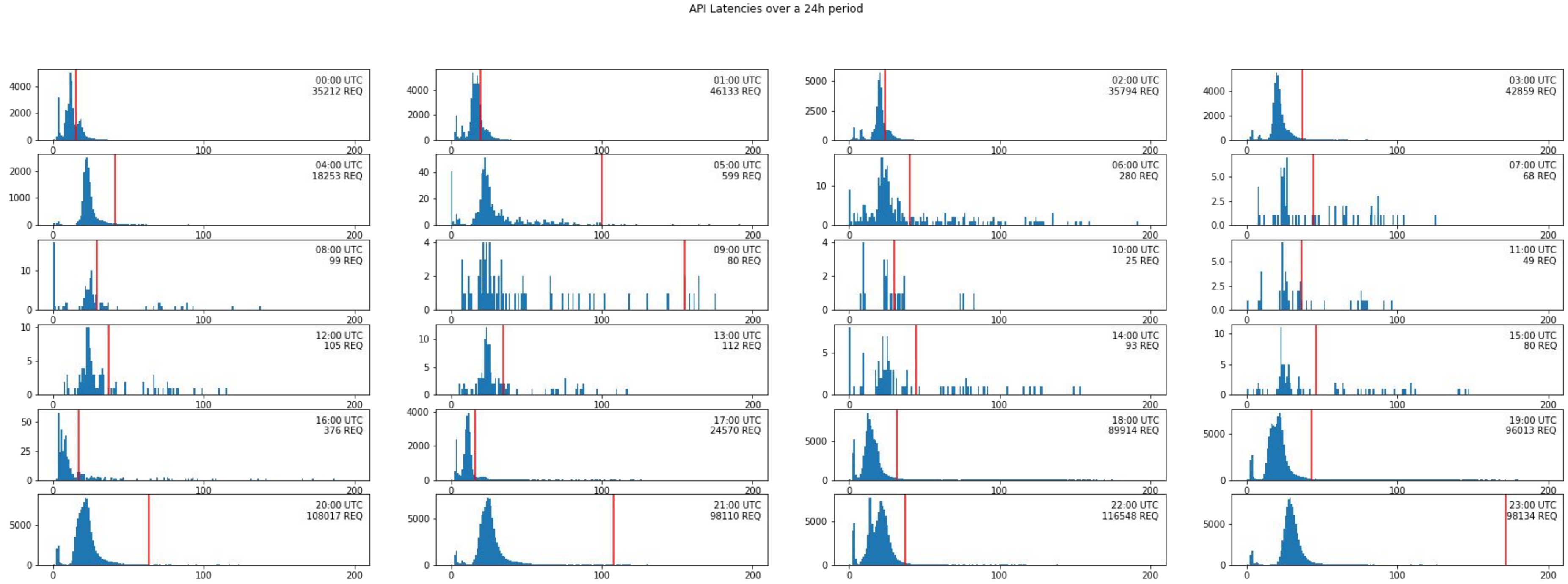
max

average

min

Example Dataset - Request Latencies over Time

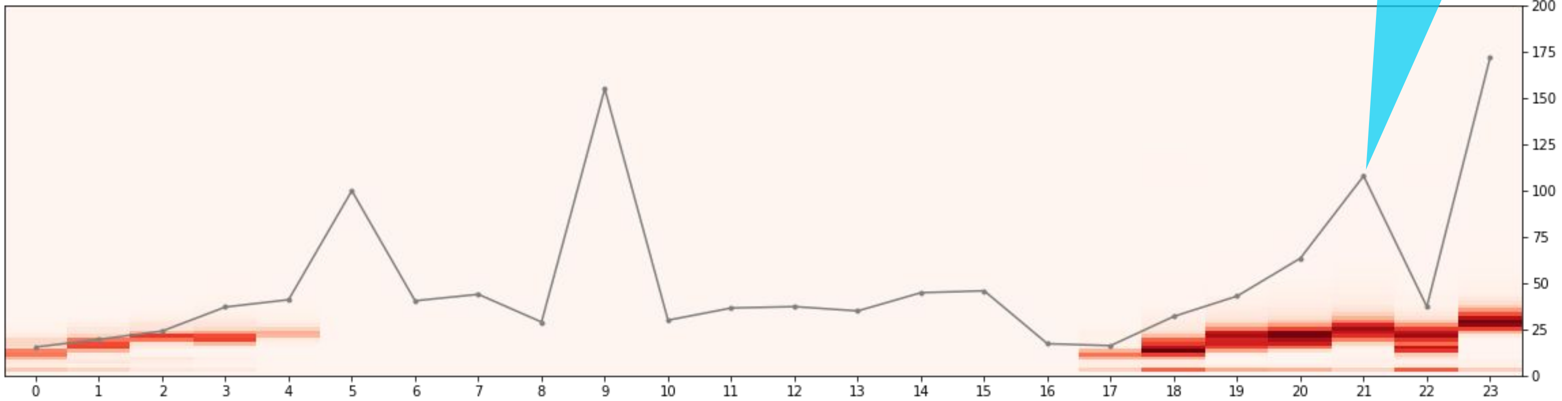
$$\mu = mean(X) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$



Example Dataset - Request Latencies Mean Values

Latency

Averaged Latency



"Looking at your average latency is like measuring the average temperature in a hospital."

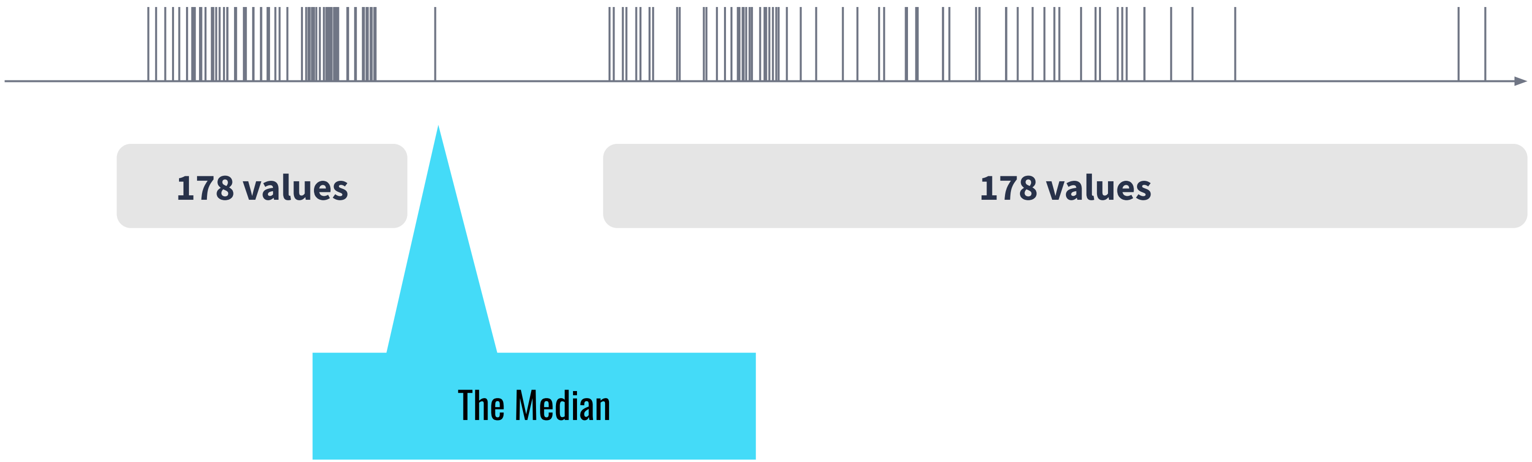
Source: Dogan Ugurlu @ Optimizely

Averages

- Most data we see on graphs is averaged: Be aware of Spike Erosion.
- Averages are easy to compute and mergeable
- Looking at average latency has little value

MEDIAN

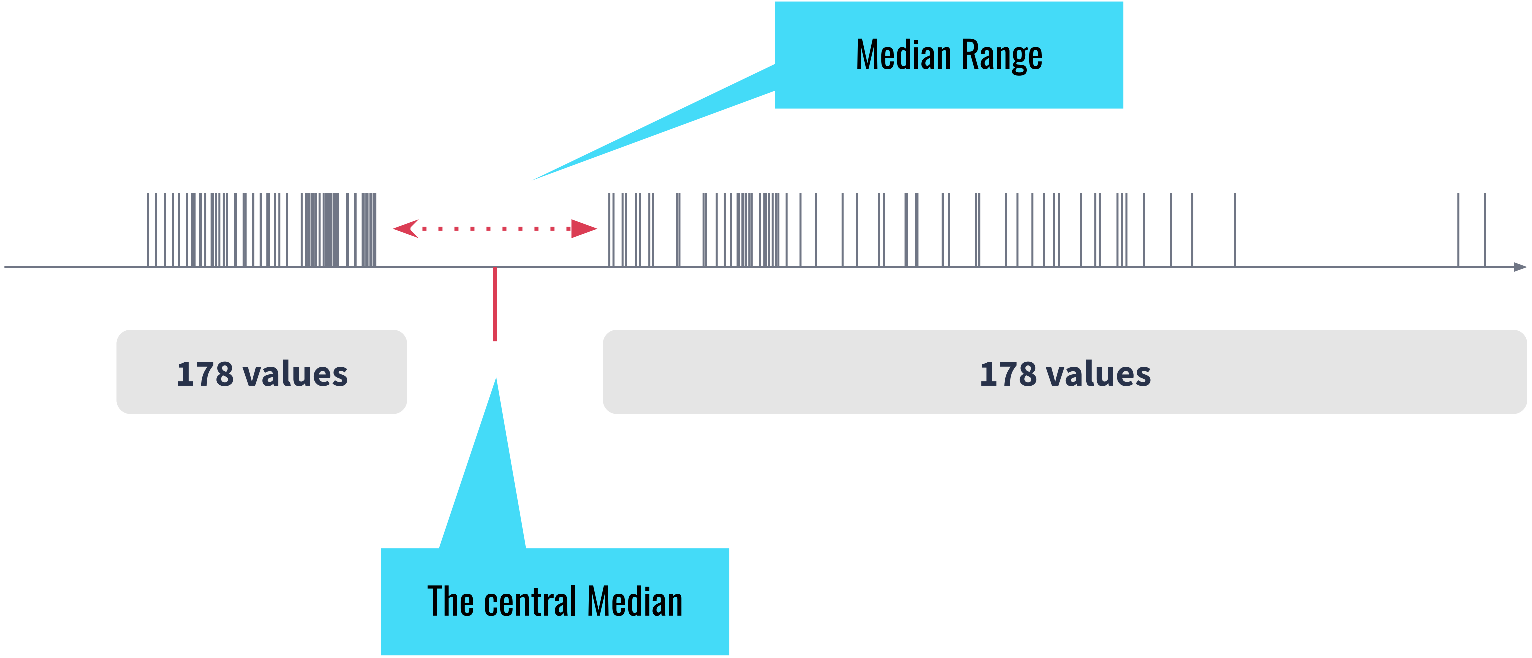
Median Definition



Median Definition

Where goes the Median?

Median Definition

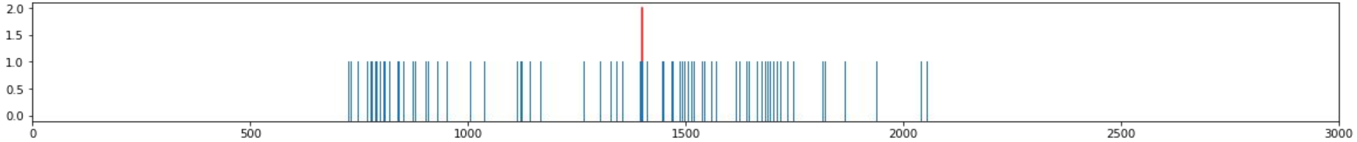


Summary Statistic - Medians

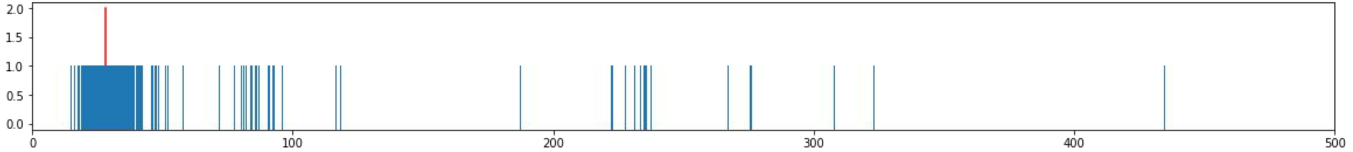
Median

Normal Noise

Request Rates

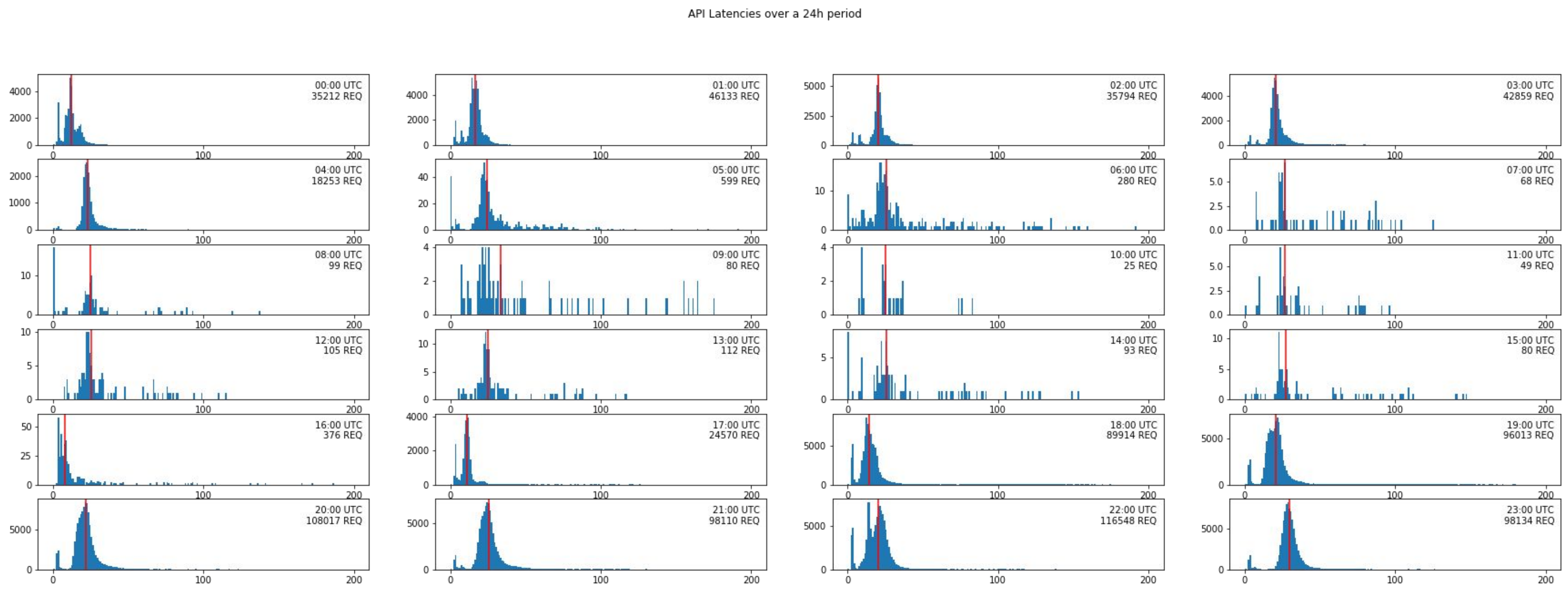


Request Latency



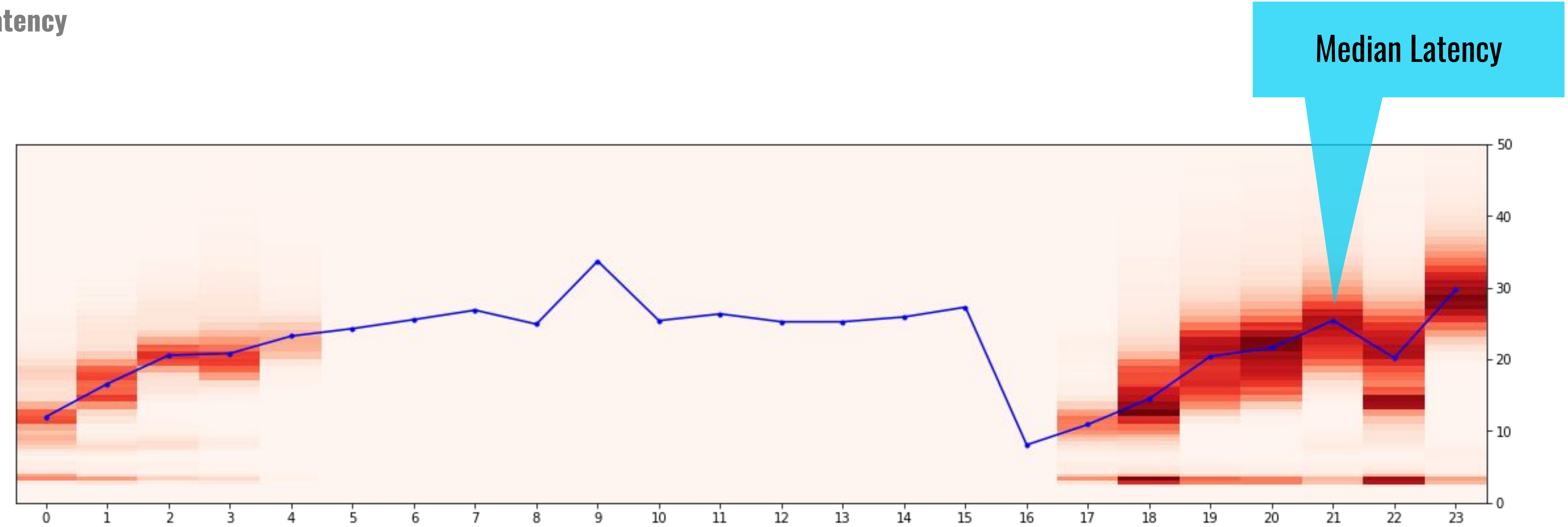
Example Dataset - Request Latencies over Time

Median



Example Dataset - Request Latencies Mean Values

Latency

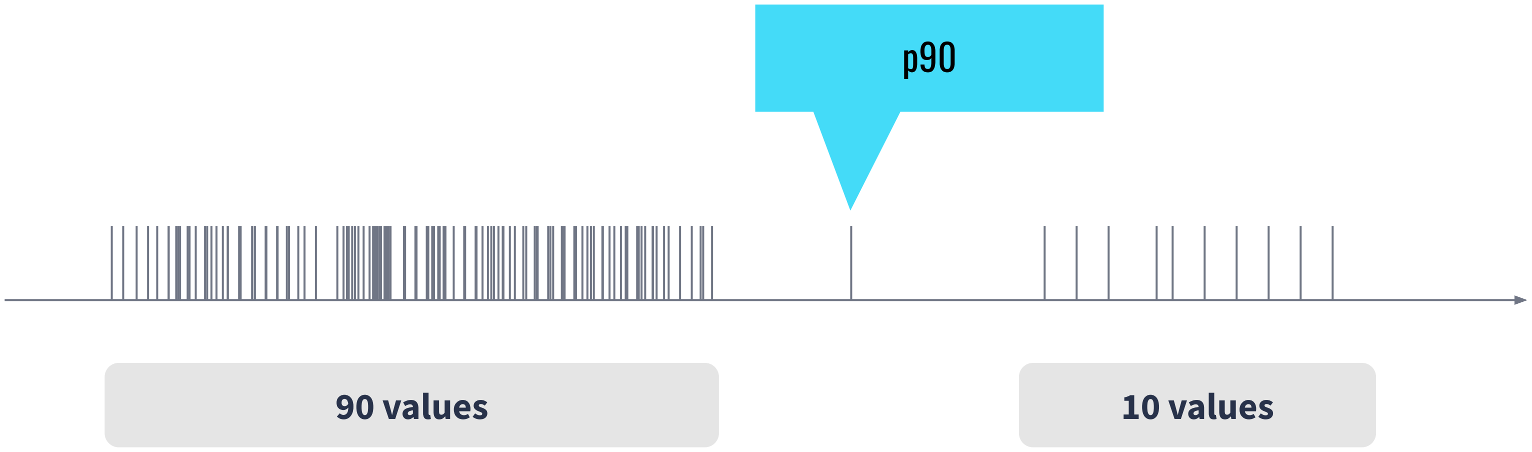


Median - Take Aways

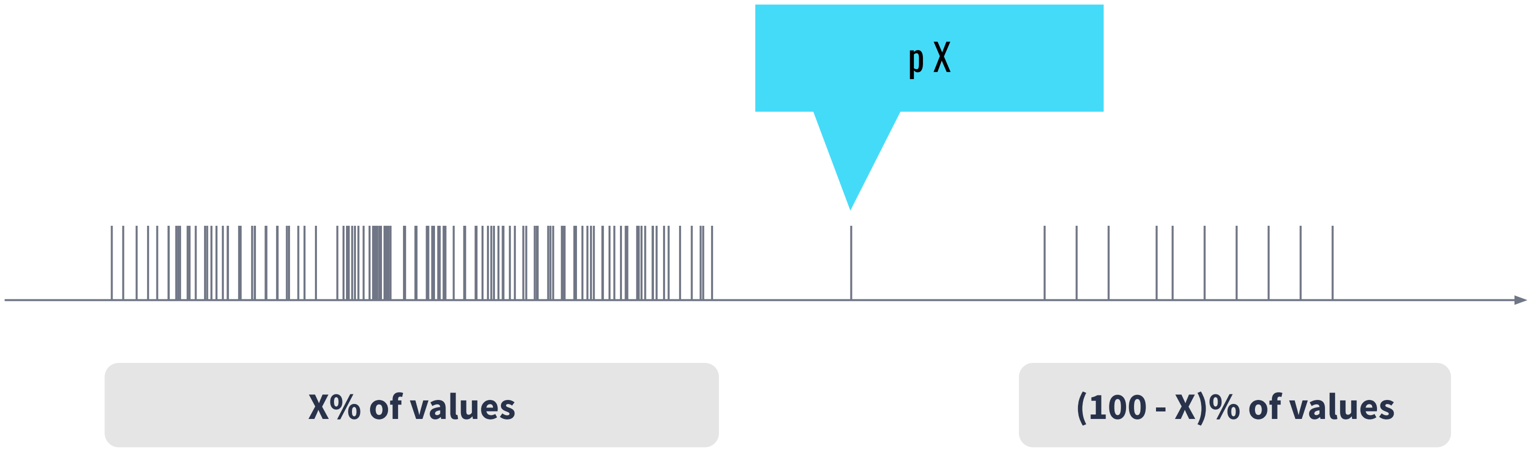
- Median values give "central representatives" of a data-set
- Medians are robust to outliers
- Fact: Medians are not easily mergeable

PERCENTILES

p90 Example



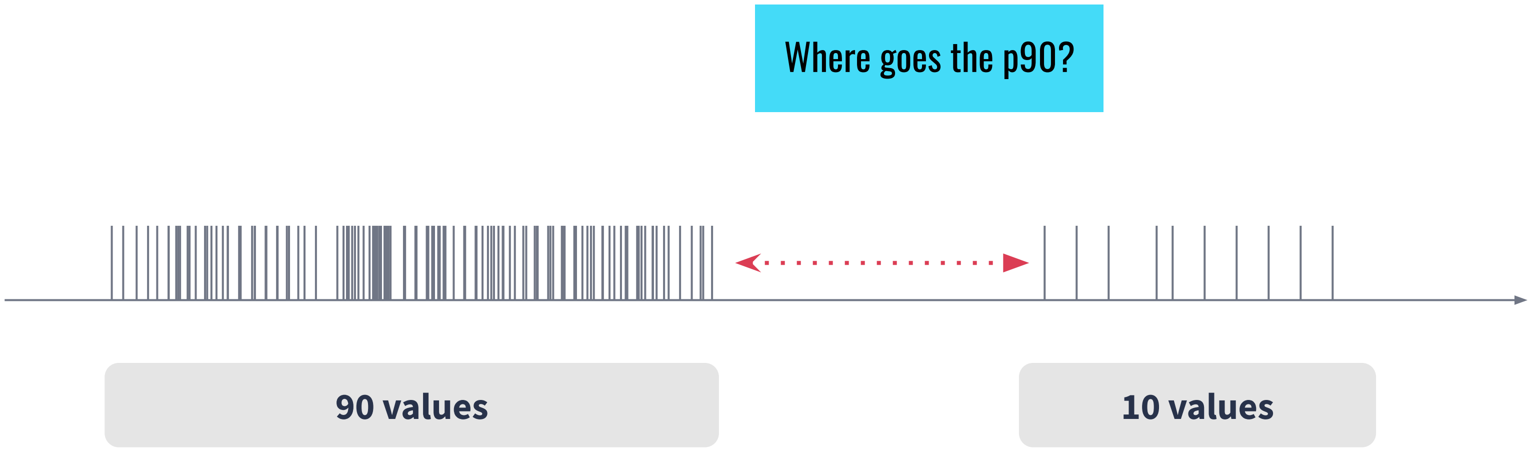
Percentile Definition



Percentile Special Cases

- ➔ p0 = Minimum
- ➔ p50 = Median
- ➔ p100 = Maximum

Percentile Definition



Lots of choices found in the wild

more on heinrichhartmann.com - Quantiles

The estimate types and interpolation schemes used include:

Type	h	Q_p
R-1, SAS-3, Maple-1	Np	$x_{\lceil h \rceil}$
R-2, SAS-5, Maple-2, Stata	$Np + 1/2$	$(x_{\lceil h - 1/2 \rceil} + x_{\lfloor h + 1/2 \rfloor}) / 2$
R-3, SAS-2	$Np - 1/2$	$x_{\lfloor h \rceil}$
R-4, SAS-1, SciPy-(0,1), Maple-3	Np	$x_{\lfloor h \rfloor} + (h - \lfloor h \rfloor)(x_{\lceil h \rceil} - x_{\lfloor h \rfloor})$
R-5, SciPy-(1/2,1/2), Maple-4	$Np + 1/2$	
R-6, Excel, Python, SAS-4, SciPy-(0,0), Maple-5, Stata-altdef	$(N+1)p$	
R-7, Excel, Python, SciPy-(1,1), Maple-6, NumPy, Julia	$(N-1)p + 1$	
R-8, SciPy-(1/3,1/3), Maple-7	$(N + 1/3)p + 1/3$	
R-9, SciPy-(3/8,3/8), Maple-8	$(N + 1/4)p + 3/8$	

Wikipedia - Percentiles

Sample Quantiles in Statistical Packages

Rob J. HYNDMAN and Yanan FAN

There are a large number of different definitions used for sample quantiles in statistical computer packages. Often within the same package one definition will be used to compute a quantile explicitly, while other definitions may be used when producing a boxplot, a probability plot, or a QQ plot. We compare the most commonly implemented sample quantile definitions by writing them in a common notation and investigating their motivation and some of their properties. We argue that there is a need to adopt a standard definition for sample quantiles so that the same answers are produced by different packages and within each package. We conclude by recommending that the median-unbiased estimator be used because it has most of the desirable properties of a quantile estimator and can be defined independently of the underlying distribution.

KEY WORDS: Percentiles; Quartiles; Sample quantiles; Statistical computer packages.

1. INTRODUCTION

The quantile of a distribution is defined as

$$Q(p) = F^{-1}(p) = \inf\{x\colon F(x) \geq p\}, \qquad 0 < p < 1,$$

where $F(x)$ is the distribution function. Sample quantiles provide nonparametric estimators of their population counterparts based on a set of independent observations $\{X_1, \ldots, X_n\}$ from the distribution F. Let $\{X_{(1)}, \ldots, X_{(n)}\}$ denote the order statistics of $\{X_1, \ldots, X_n\}$, and let $\hat{Q}_i(p)$ denote the ith sample quantile definition.

One difficulty in comparing quantile definitions is that there is a number of equivalent ways of defining them. However, the sample quantiles that are used in statistical packages are all based on one or two order statistics, and can be written as

$$\hat{Q}_i(p) = (1-\gamma)X_{(j)} + \gamma X_{(j+1)}$$

$$\text{where} \quad \frac{j-m}{n} \leq p < \frac{j-m+1}{n} \qquad (1)$$

for some $m \in \mathbb{R}$ and $0 \leq \gamma \leq 1$. The value of γ is a function of $j = \lfloor pn + m \rfloor$ and $g = pn + m - j$. Here, $\lfloor u \rfloor$ denotes the largest integer not greater than u; later we shall use $\lceil u \rceil$ to denote the smallest integer not less than u.

We consider estimators of the form (1), including some that are not found in statistical packages. There have been several other nonparametric quantile estimators proposed that are not of the form (1) (e.g., Harrell and Davis 1982; Sheather and Marron 1990), but these are not implemented in widely available packages and so are not considered here. We also exclude sample quantiles that are not defined for all p including hinges and other letter values (Hoaglin 1983) and related methods (Freund and Perles 1987).

A closely related problem is the selection of plotting position in a quantile plot in which $X_{(k)}$ is plotted against p_k or in a quantile–quantile plot in which $X_{(k)}$ is plotted against $G^{-1}(p_k)$ where G is a distribution function. Various rules for p_k have been suggested (see Cunnane 1978; Harter 1984; Kimball 1960; Mage 1982). Each plotting rule corresponds to a sample quantile definition by defining $\hat{Q}_i(p_k) = X_{(k)}$ and using linear interpolation for $p \neq p_k$. However, the criteria by which a plotting position is chosen (e.g., the five postulates of Gumbel 1958, pp. 32–34 or the three purposes of Kimball 1960) may be quite different from the criteria for choosing a good sample quantile definition.

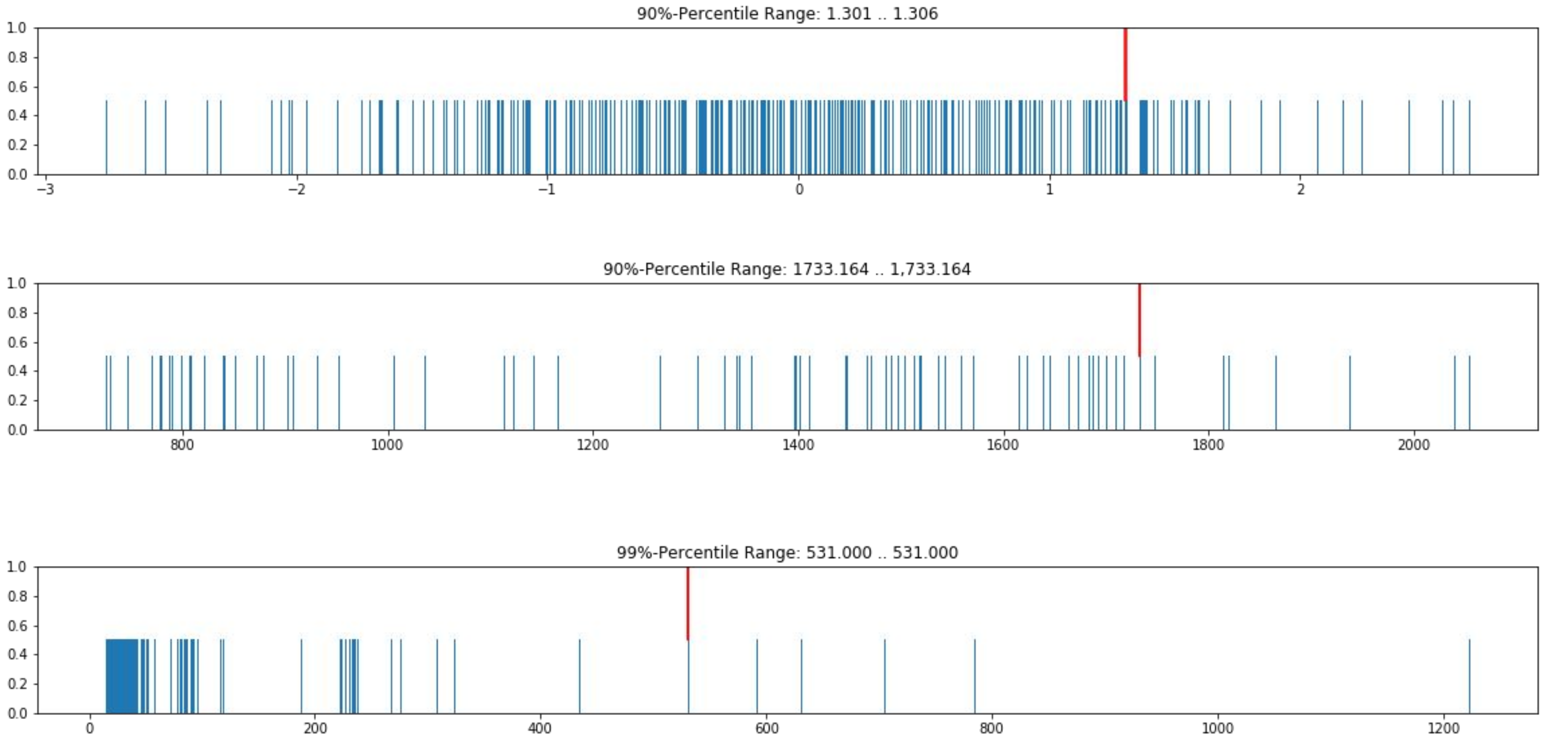
We compare sample quantile definitions of the form (1) by describing their motivation and whether or not they pos-

Table 1. Six Desirable Properties for a Sample Quantile

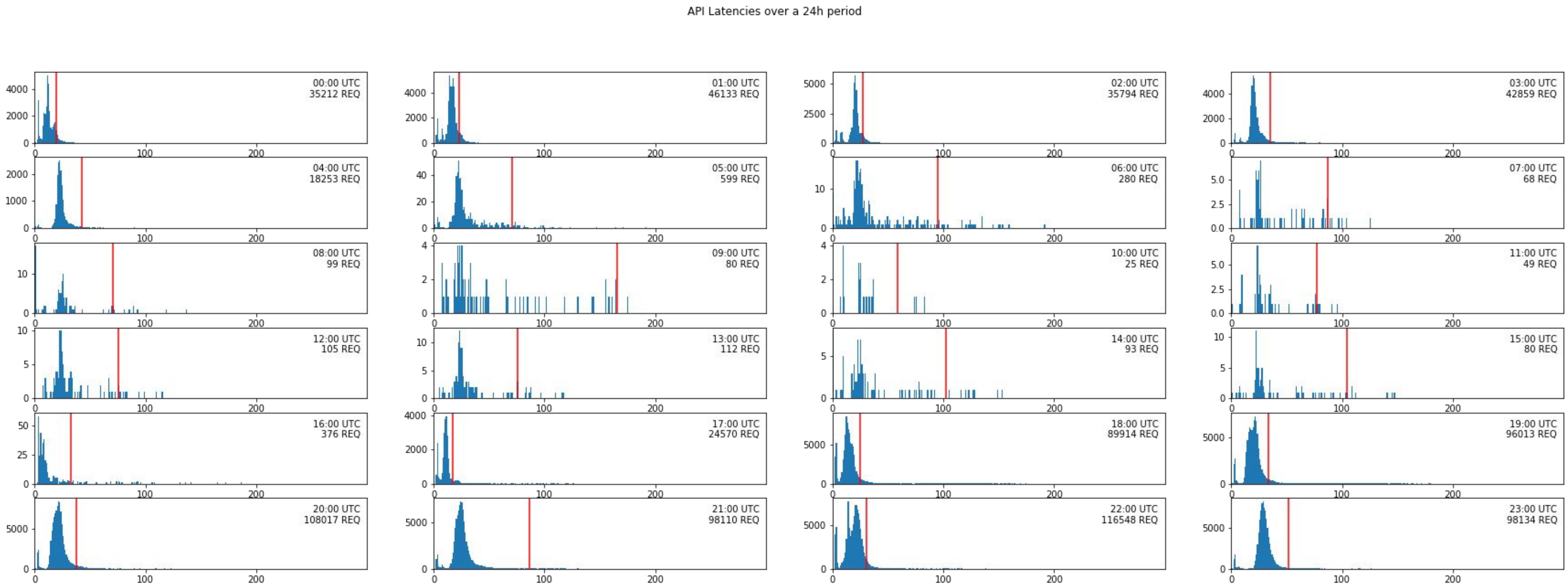
P1: $\hat{Q}_i(p)$ is continuous.
P2: $\mathrm{Freq}(X_k \leq \hat{Q}_i(p)) \geq pn$.
P3: $\mathrm{Freq}(X_k \leq \hat{Q}_i(p)) = \mathrm{Freq}(X_k \geq \hat{Q}_i(1-p))$.
P4: Where $\hat{Q}_i^{-1}(x)$ is uniquely defined,

Hyndman, Fan (1996)

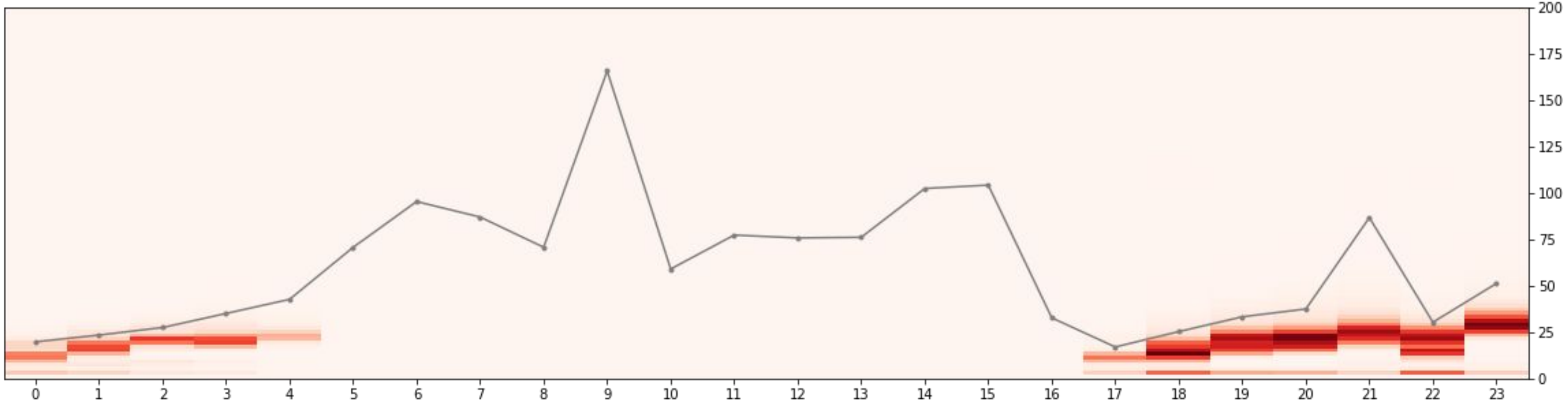
Summary Statistic - Percentile



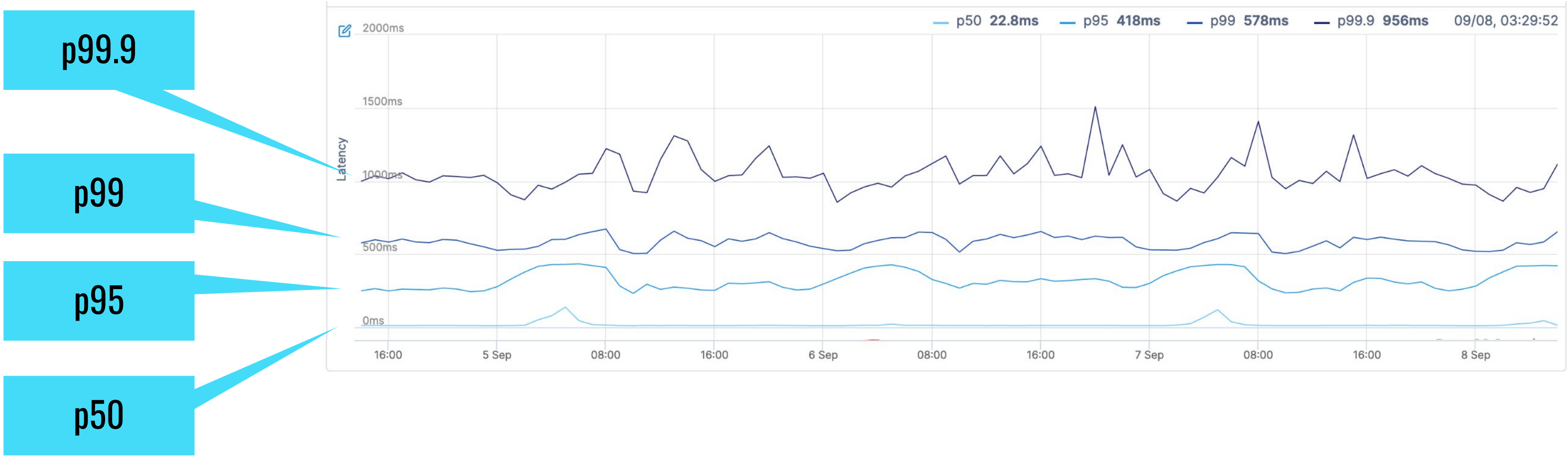
Latency Percentiles



Latency Percentiles - over time



Example - Production View - Request Latency over Time

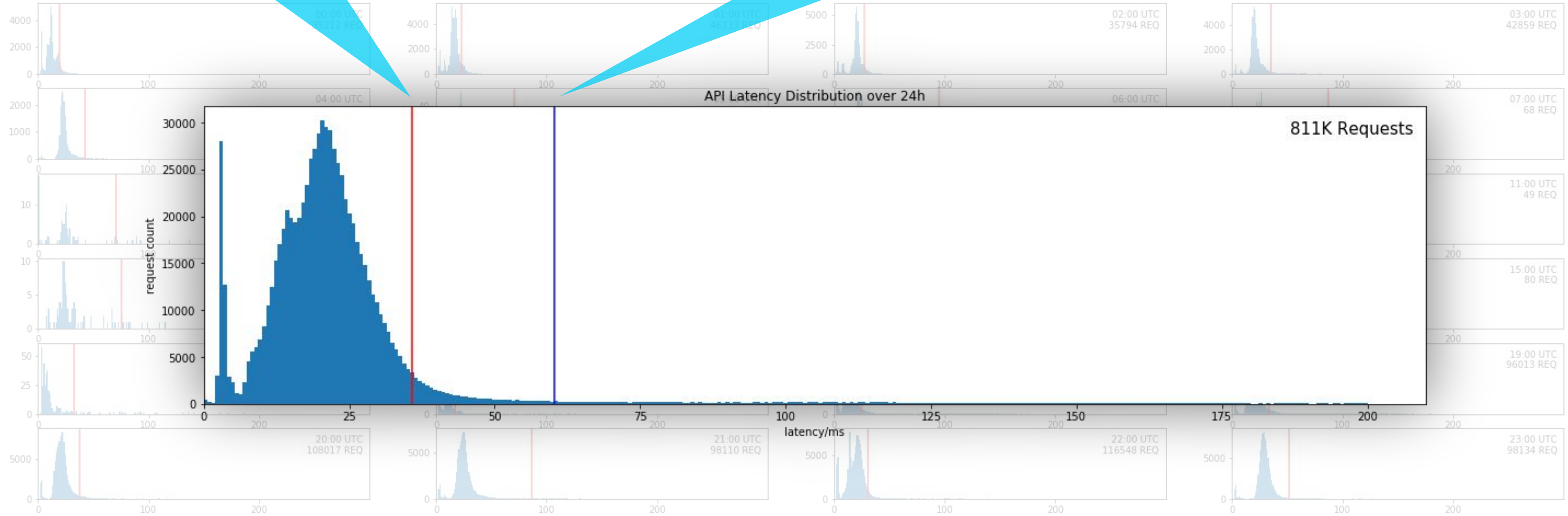


Lightstep from ServiceNow

Aggregated Percentiles

True p90

Averaged p90 (68% off)

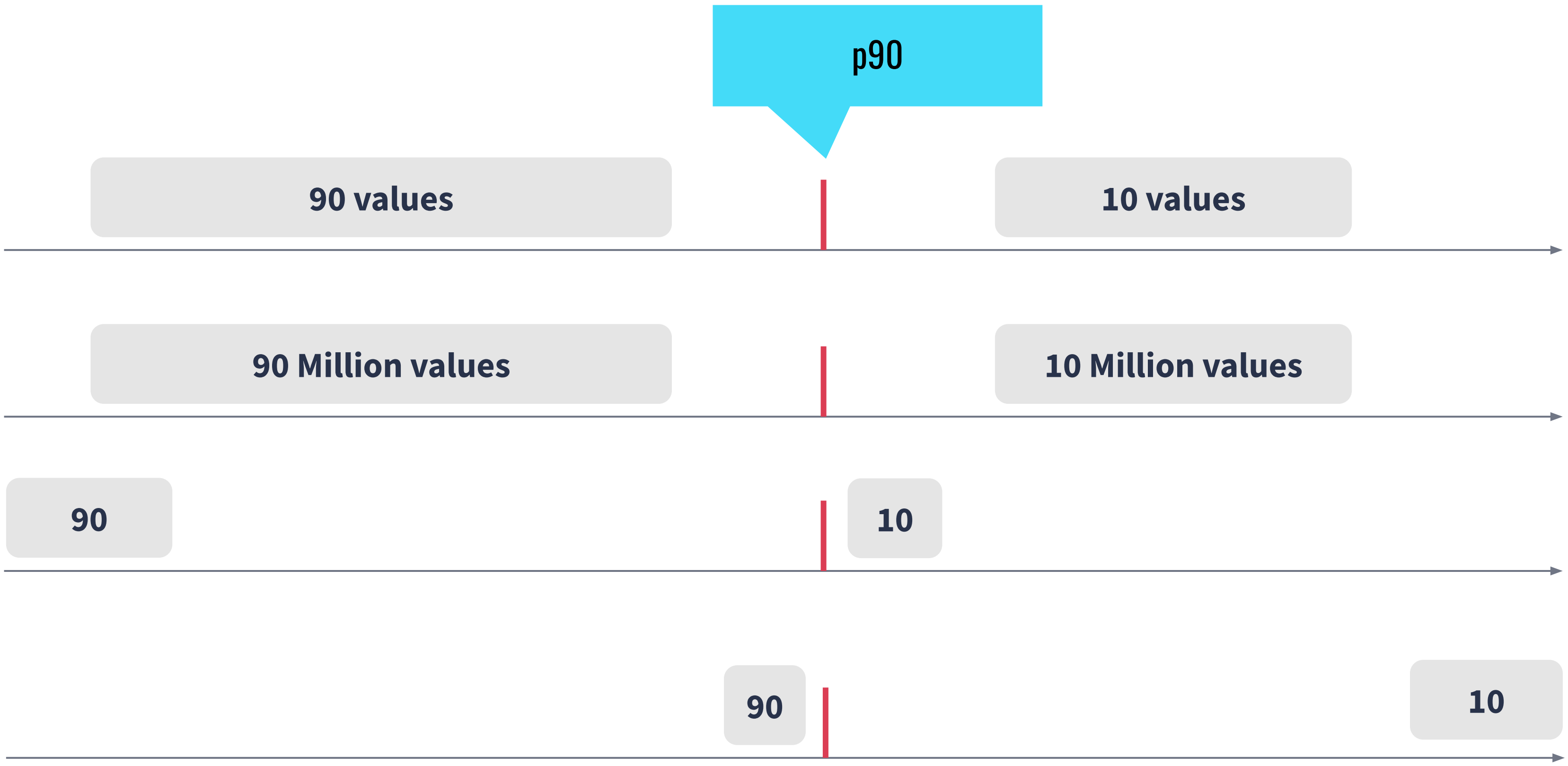


“

Don't average percentiles!

(… and expect the results to be something meaningful)

Distributions with the same p90



How to do better? Use Histogram data-structures!

- Prometheus Sparse Histograms (see Björn Rabenstein's Talk!)
- OTelemetry Exponential Histograms (new!)
- HDR Histogram (Tene @ Aszul)
- T-Digest (Dunning @ Dynatrace)
- OpenHistogram (Schlossnagle, - @ Circonus)
- DD/UUD-Sketch (Masson, Rim, Lee @ Data Dog)

Percentiles - Take Aways

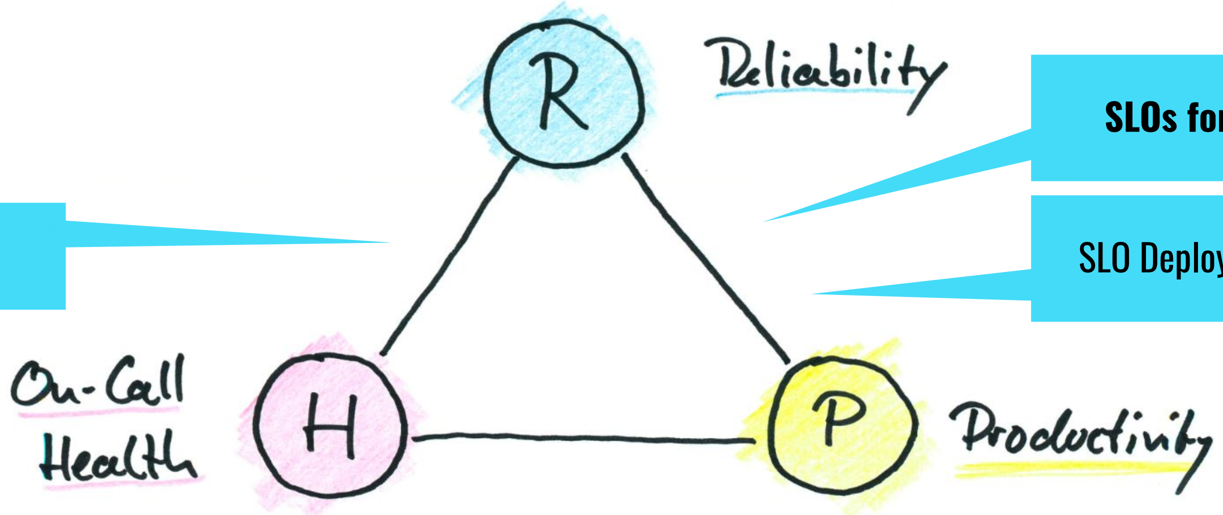
- Percentiles generalize min/max/median
- Percentiles (p90, p99, p99.9) are used to describe latency
- Percentiles can't be aggregated (need histograms for this!)

SLOs

SLO Goals

Steer engineering investments into reliability using data.

SRE Triangle



SLO Alerting

SLOs for Steering

SLO Deployment Gating

SLO Concepts

- SLI = Reliability KPI
  = Number between 0..1 that measures reliability of a given service over a managerial time horizon (i.e. 4 weeks).

- SLO = Reliability Objective
  = Number between 0..1 that quantifies the degradation of service that is acceptable for business and customers, as threshold on an SLI.

Event Based SLIs

$$\text{SLI} = \frac{\text{\# good events over past 4 weeks}}{\text{\# total events over past 4 weeks}}$$

Availability SLIs

1. Synthetic Probe SLI = # successful probe runs in 4 weeks / # total probe runs
2. Error Rate SLI = # good responses over 4 past weeks / # total requests over 4 past weeks
3. Error Rate SLI measured on client (mobile, web)

"Latency" Type SLI Types

Better with Histograms

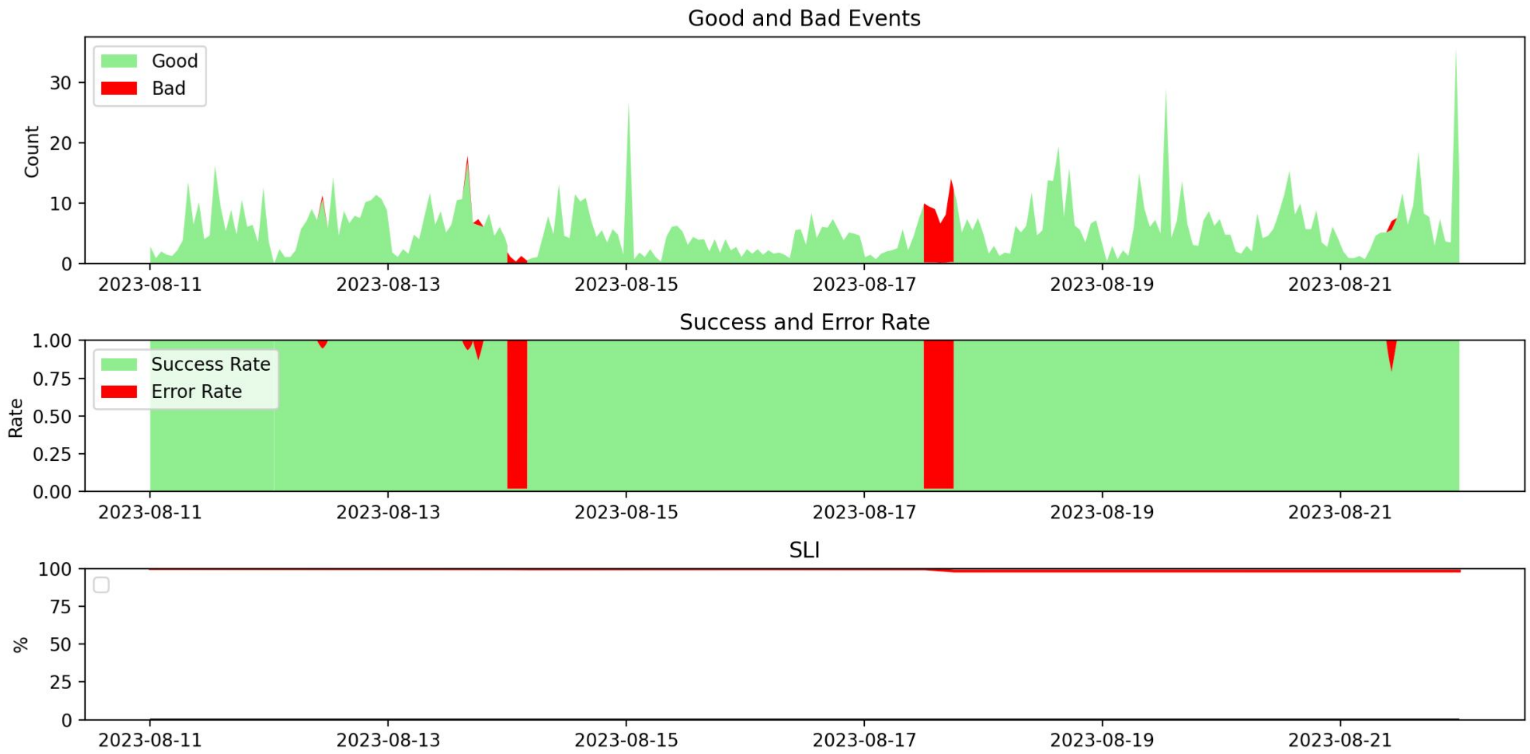
Latency SLI = # successful requests served within 200ms in 4w /
successful requests in 4w

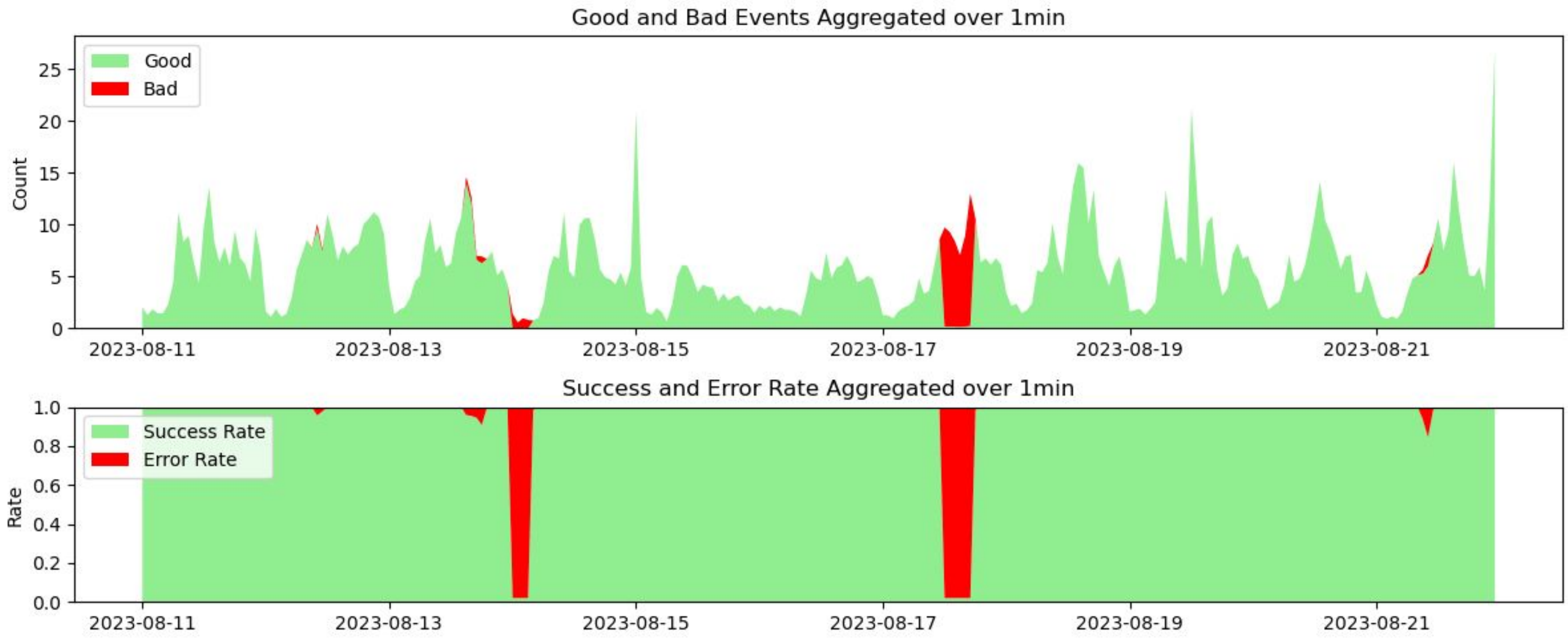
Freshness SLI = # events ingested within 120 seconds in 4w /
events invested in 4w

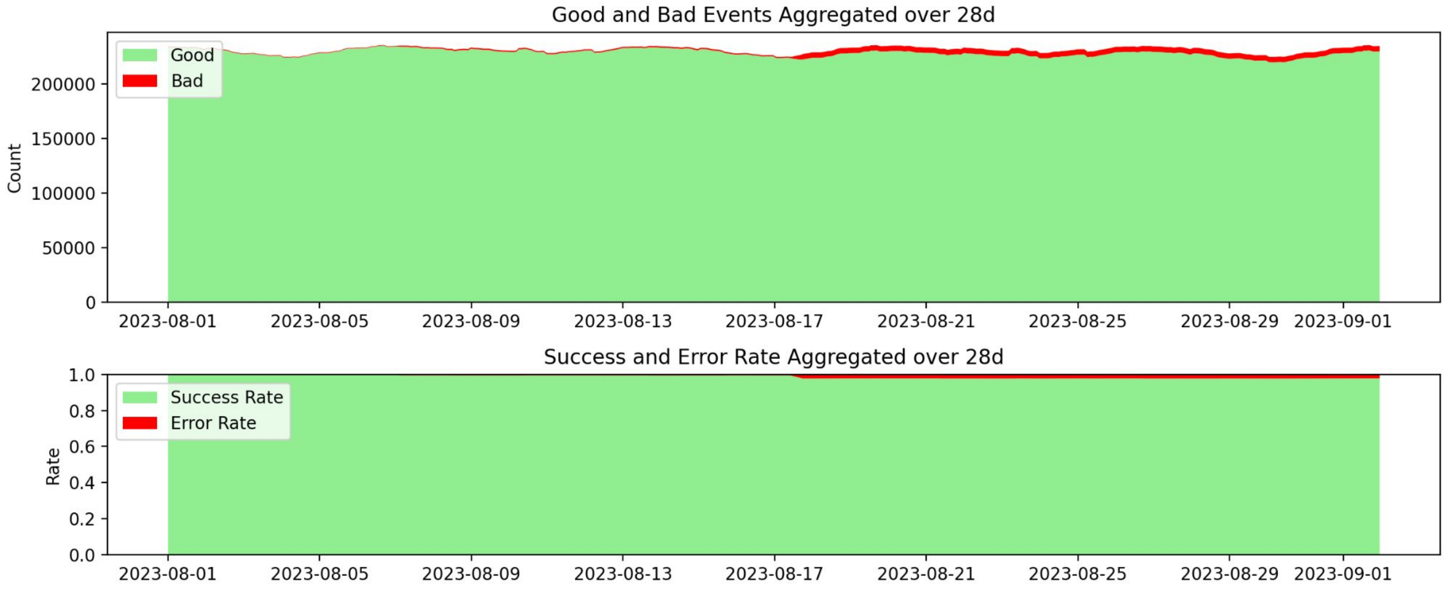
Execution SLI = # job executions completed within 5 seconds in 4w /
job executions in 4w

Event Based SLIs

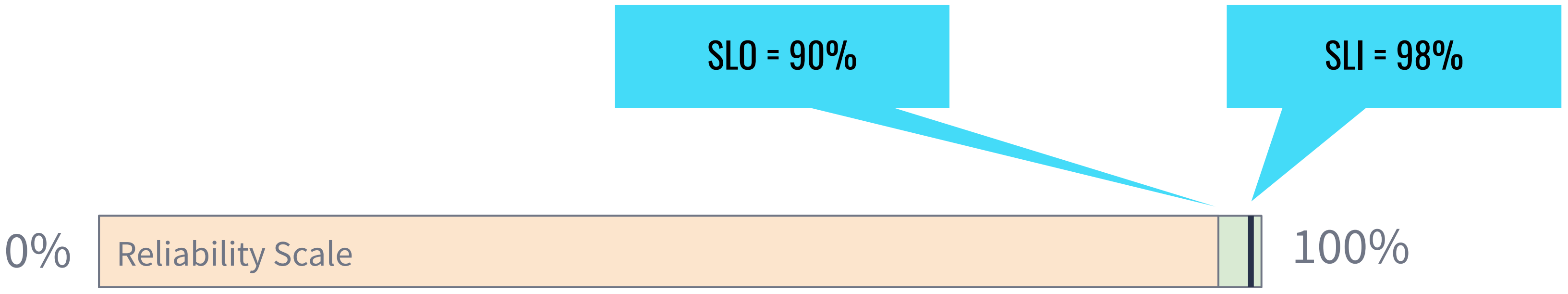
$$\text{SLI} = \frac{\text{\# good events over past 4 weeks}}{\text{\# total events over past 4 weeks}}$$


Good and Bad Events
Good
Bad
Count
0
10
20
30
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21
Success and Error Rate
Success Rate
Error Rate
Rate
0.00
0.25
0.50
0.75
1.00
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21
SLI
%
0
25
50
75
100
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21

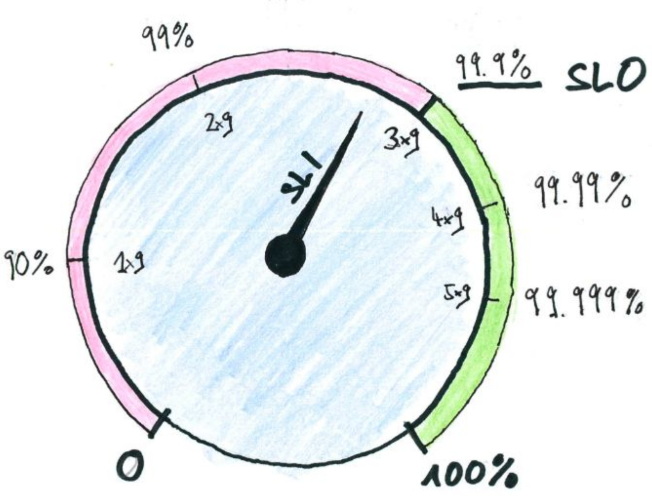

Good and Bad Events Aggregated over 1min
Good
Bad
Count
0
5
10
15
20
25
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21
Success and Error Rate Aggregated over 1min
Success Rate
Error Rate
Rate
0.0
0.2
0.4
0.6
0.8
1.0
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21


Good and Bad Events Aggregated over 28d
Good
Bad
Count
0
50000
100000
150000
200000
2023-08-01
2023-08-05
2023-08-09
2023-08-13
2023-08-17
2023-08-21
2023-08-25
2023-08-29
2023-09-01
Success and Error Rate Aggregated over 28d
Success Rate
Error Rate
Rate
0.0
0.2
0.4
0.6
0.8
1.0

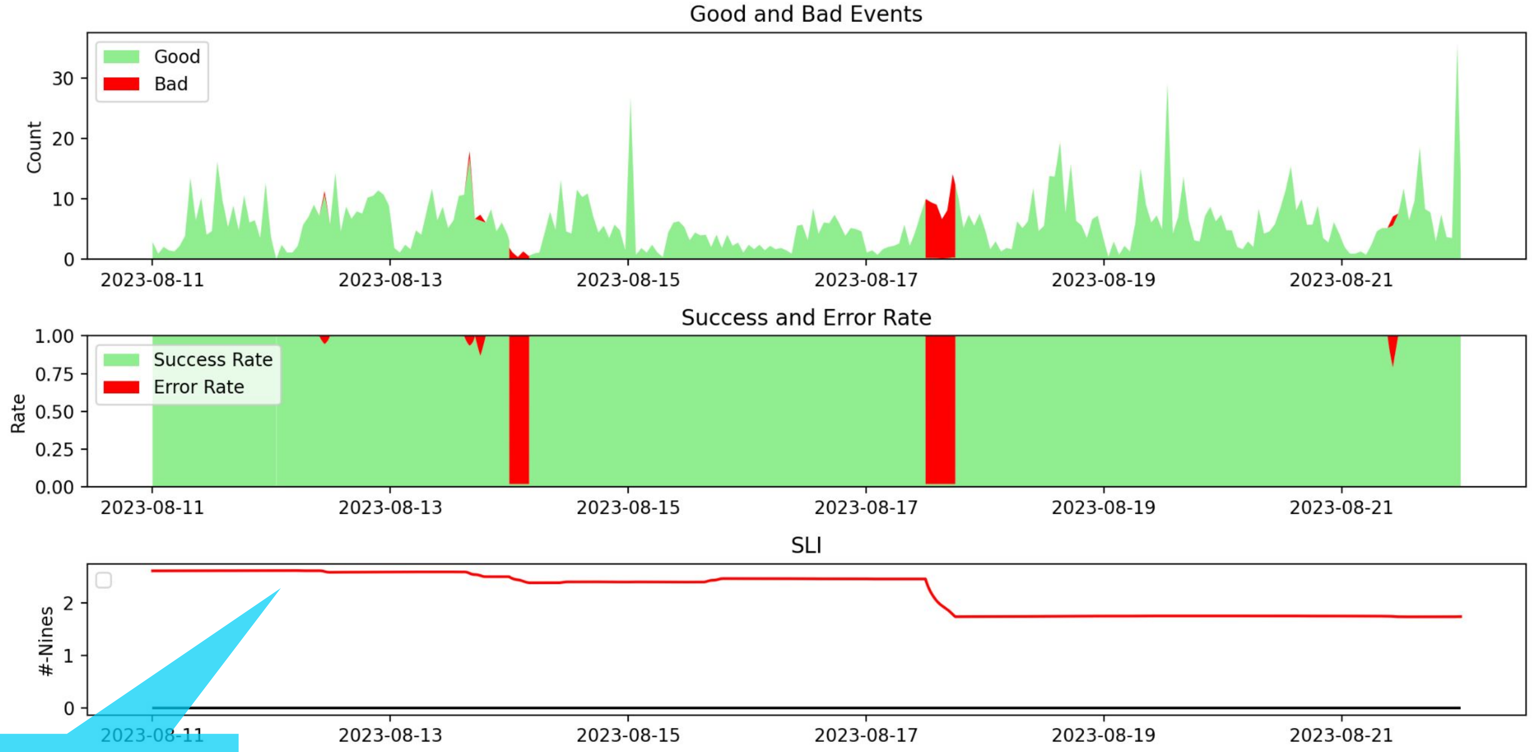
SLO on the Linear Reliability Scale



The 9's Scale for SLI/SLO Values



#-Nines = - Log10(1 - SLI)



SLI on #-9 Scale

The Error Budget Scale

- Error Budget = 100% *Perfect Reliability (SLI = 100%)*
- Error Budget > 0% *Reliability within SLO*
- Error Budget = 0% *Reliability at SLO (SLI = SLO)*
- Error Budget < 0% *SLO violated*

Error Budget Formula

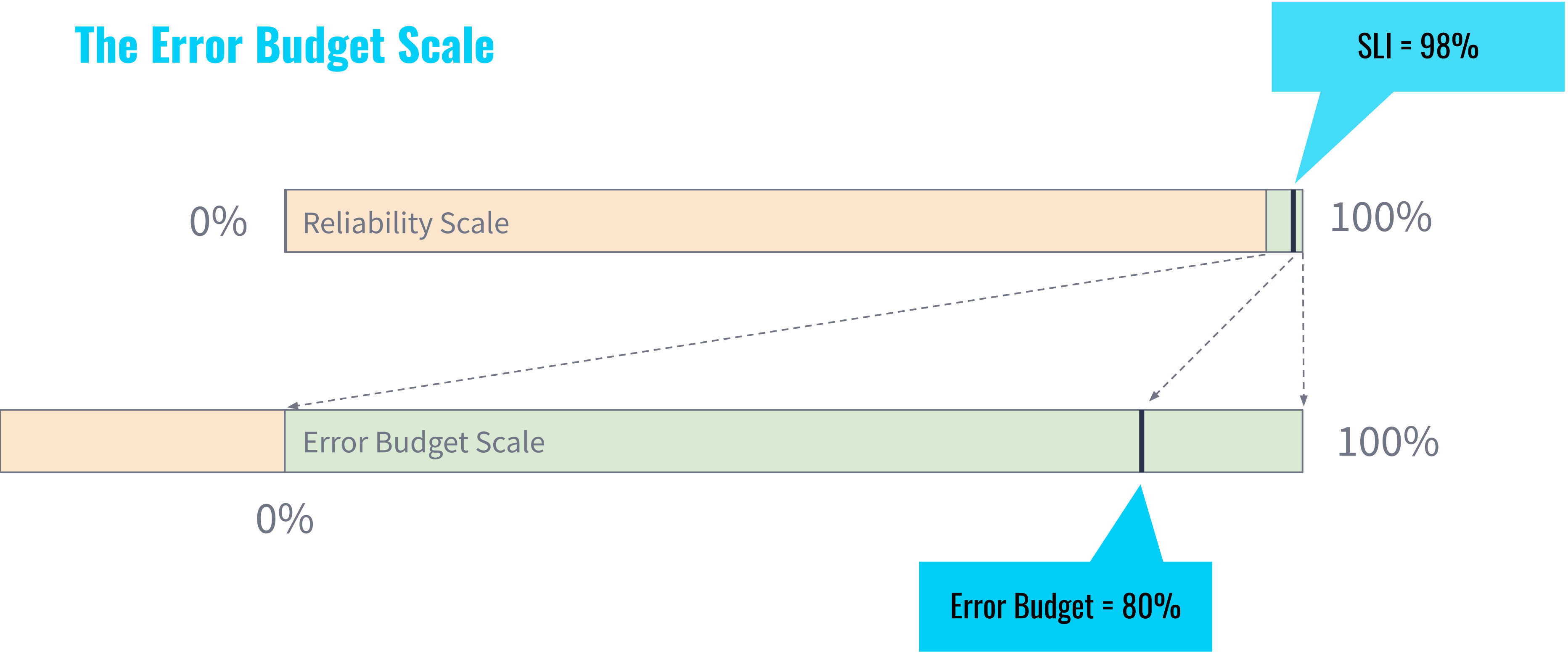
$$\text{Error Budget} = \frac{\text{SLI} - \text{SLO}}{100\% - \text{SLO}}$$

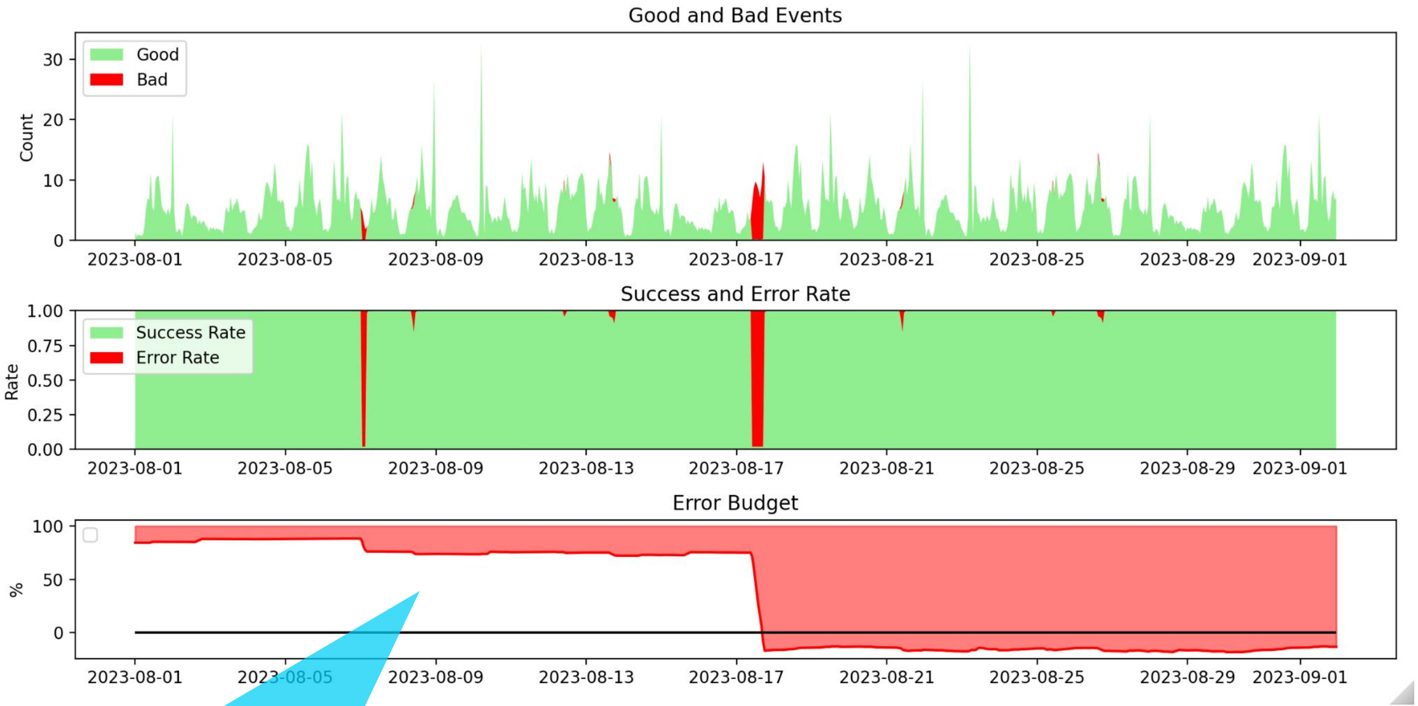
Error Budget Formula - for event based SLOs

$$\text{Error Budget} = \frac{(1\text{-SLO}) * (\#\ \text{total events}) - (\#\ \text{bad events})}{(1\text{-SLO}) * (\#\ \text{total events})}$$

$$= \frac{(\#\ \text{acceptable bad events}) - (\#\ \text{bad events})}{(\#\ \text{acceptable bad events})}$$

The Error Budget Scale





Error Budget = Rescaled SLI

Steering with SLOs - Example

- GREEN Error Budget > 20%

  No investment in reliability needed.

- AMBER Error Budget in 0% .. 20%

  Reliability needs attention. Increase testing efforts.

- RED Error Budget < 0%

  Investments in reliability needed. Stop Deployments.

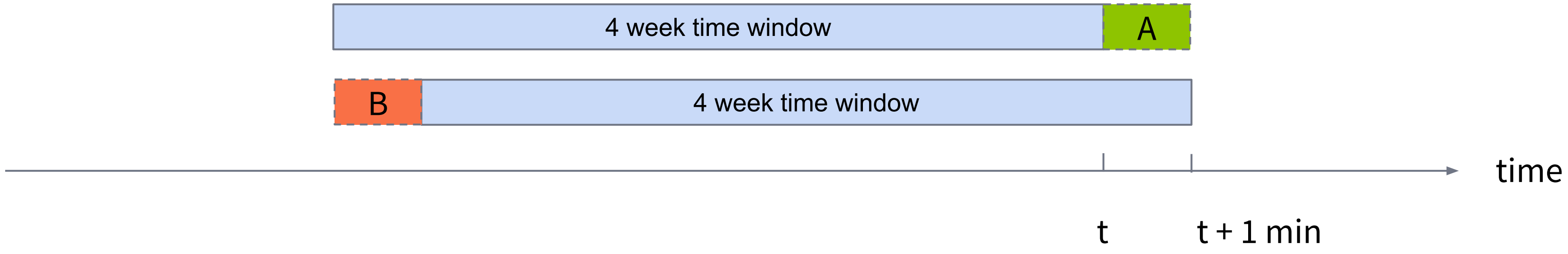
Alerting on SLOs

- SLI measure **Symptoms** Experienced by user
- SLI can be used as a high-quality alerting signal
- + Few false positives
- - Alerts only fire once user experience is already degraded

Changes to Error Budget come from two sides

There is a constant C depending on the SLO and the average request rate, so that

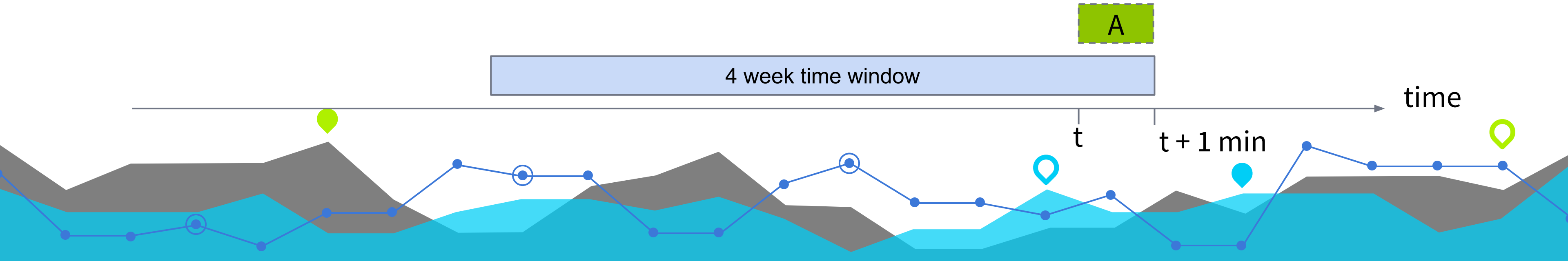
$$\text{ErrorBudget}(t + 1\text{ min}) - \text{ErrorBudget}(t) \approx C\,(\#\text{ good events in A}) - C\,(\#\text{ good events in B})$$



Burn Rates

Burn rates quantify impact of events in last minute (A) to error budget.

- Burn Rate = 0 — *No bad events in last minute*
- Burn Rate < 1 — *Error budget is not in danger*
- Burn Rate = 1 — *Error budget will be exactly depleted when sustained*
- Burn Rate > 1 — *Error budget will be exceeded when sustained*



Naive Burn Rate

Naive Burn Rate = "Compare current error rate to SLO"

= (error rate over last minute) / (allowable error rate)

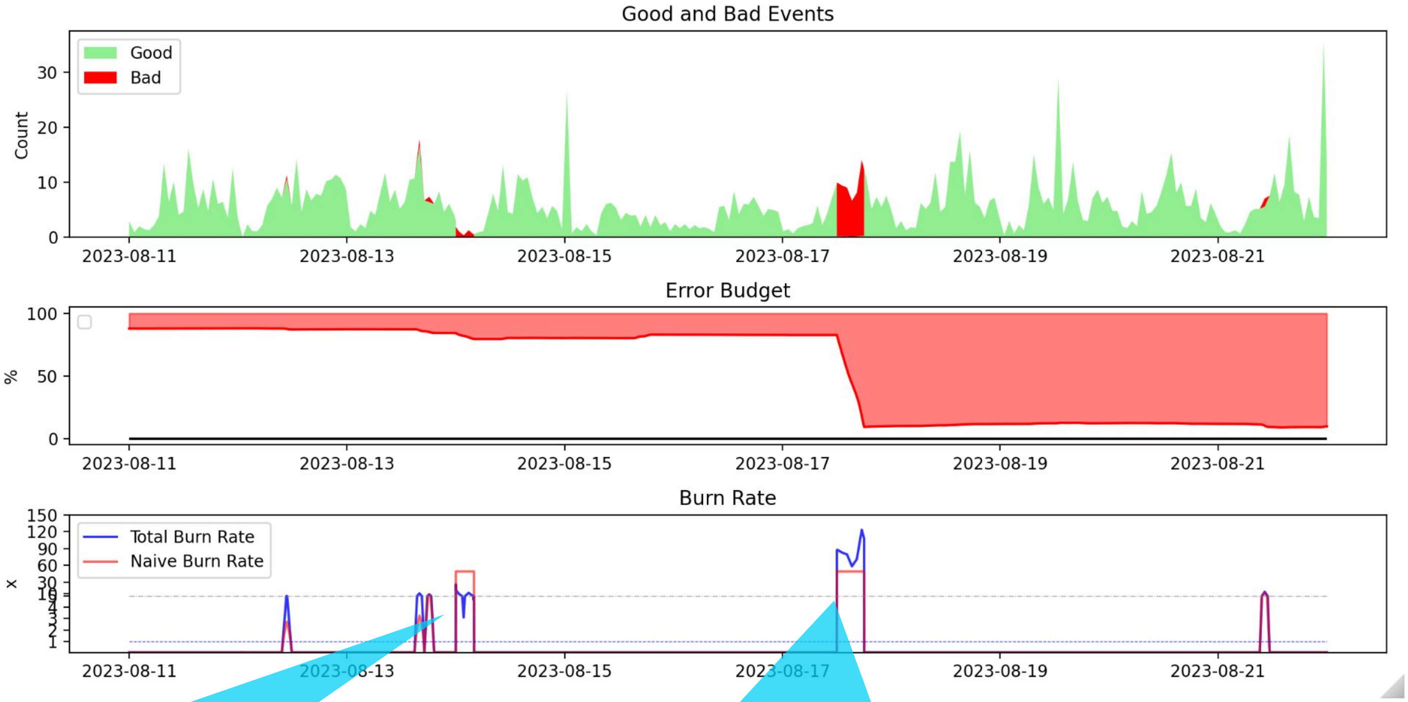
= (# good events) / (# total events) / (1 - SLO)

Total Burn Rate

Total Burn Rate = "How fast are we burning error budget?"

= (bad events over last min) / (error budget for each min)

$$= \frac{\text{(\#bad events over last min)}}{\text{((\# events over 4 weeks) / (\#min in 4 weeks)) *(1-SLO)}}$$



Total Burn Rate = 10
Naive Burn Rate = 50

Total Burn Rate = 120
Total Burn Rate = 50

Multi “Burn Rate” Alerts from the Book

- Paging Alert if 2% of error budget consumed in 1h
- Paging Alert if 5% of error budget consumed in 5h
- Notification if 10% of error budget consumed in 3d

Translate into conditions on Burn Rate

SLO based Alerting Rule

Alert if 2% error budget consumed in 1h

⇔ Alert if (1h total burn rate) > 2% * (#h in 4 weeks) = 13.4

⇔ Alert if (errors over past hour) > 13.4 * (1-SLO) * (4w average events per h)

Metric

Constant

"Almost" Constant

SLO based Alerting Rule

Assume that **request rate is constant**, then:

Alert if 2% error budget consumed in 1h

⇔ Alert if (1h naive burn rate) > 2% * (#h in 4 weeks) = 13.4

⇔ Alert if (error rate over past hour) > 13.4 * (1-SLO)

Metric

Constant


Good and Bad Events
Good
Bad
Count
35
30
25
20
15
10
5
0
2023-08-11
2023-08-13
2023-08-15
2023-08-17
2023-08-19
2023-08-21
Burn Rate
Total 1h-Burn Rate
x
150
120
90
60
30
10
5
4
3
2
1
2023-08-11
2023-08-13
2023-08-15
2023-08-19
2023-08-21
Alert!

Take Aways

- SLOs are good if they are used for Steering Engineering Investments
- Error Budgets are re-scaled SLIs
- Burn rate quantify changes in SLI caused by current events
- SLOs give effective alerting rules via Burn Rates

Sampling

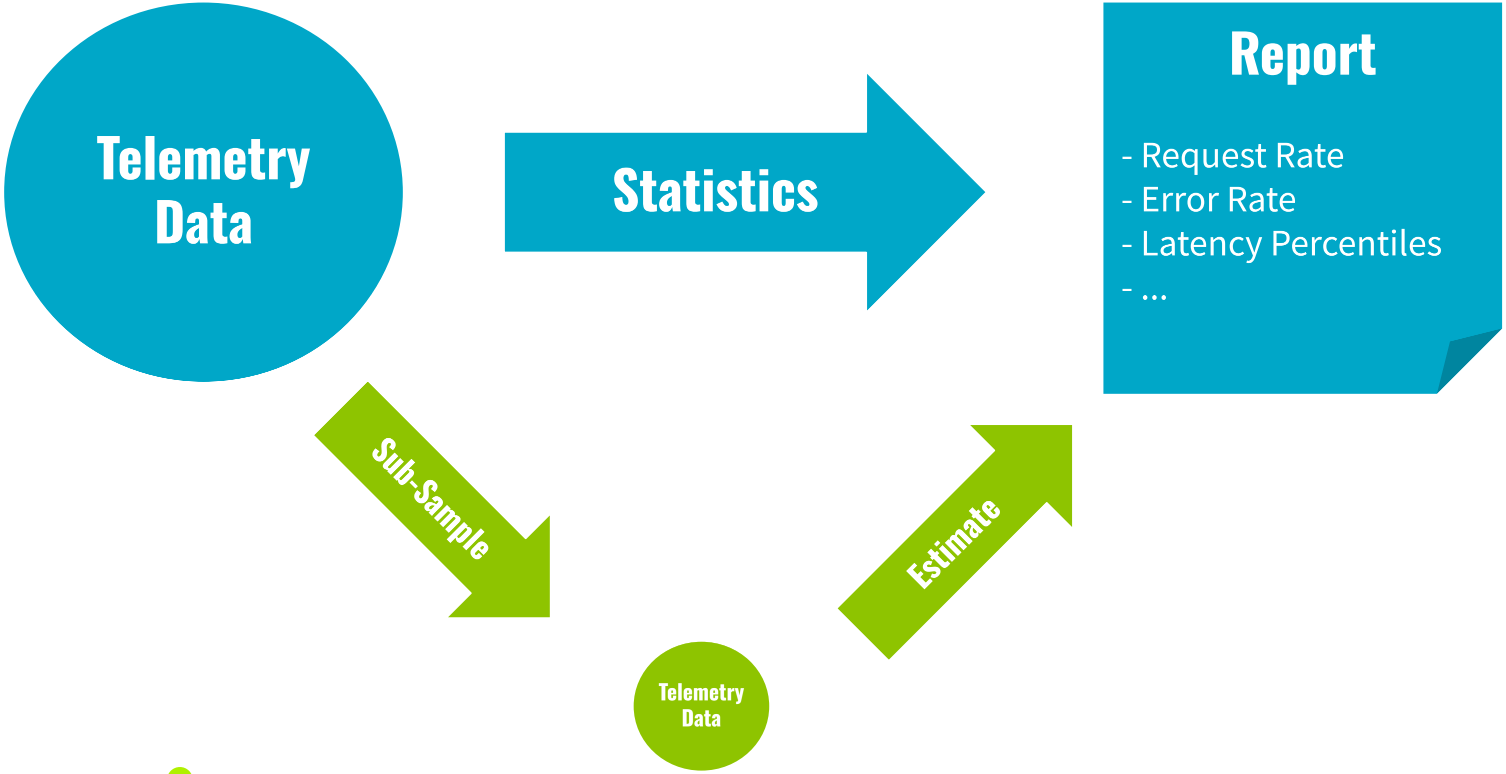
Accuracy

Graphs, Dashboards, KPIs, SLOs

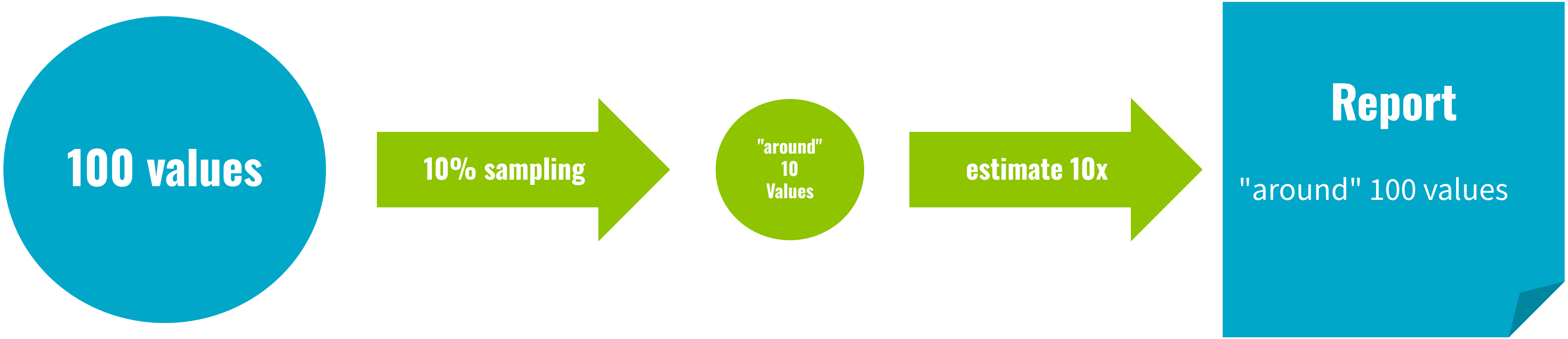
Sampling

Costs

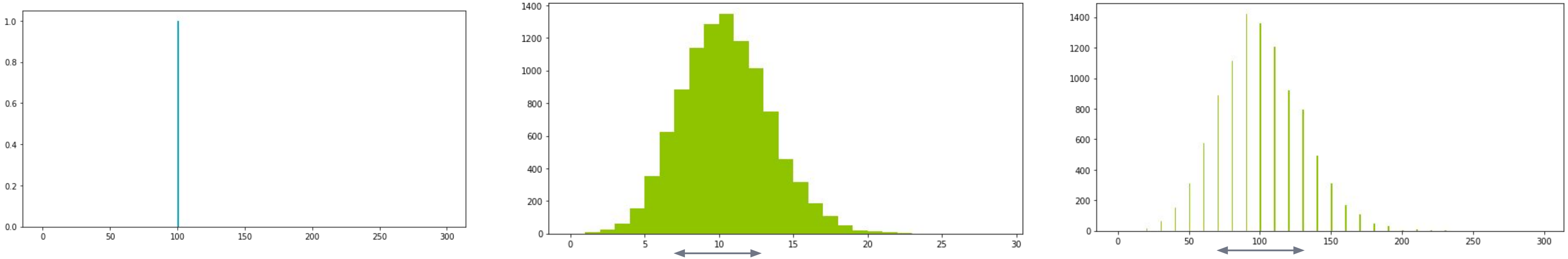
Logs/Metrics/Traces


Telemetry Data
Statistics
Report
- Request Rate
- Error Rate
- Latency Percentiles
- ...
Sub-Sample
Telemetry Data
Estimate

Sampling


100 values
10% sampling
"around"
10
Values
estimate 10x
Report
"around" 100 values

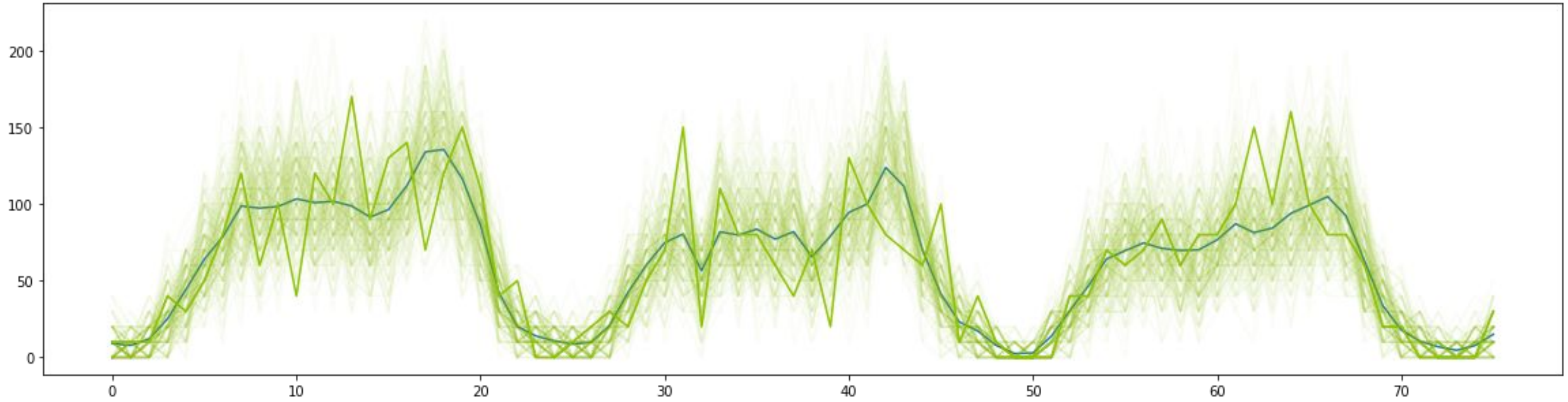
Sampling - Simulation (10K iterations)



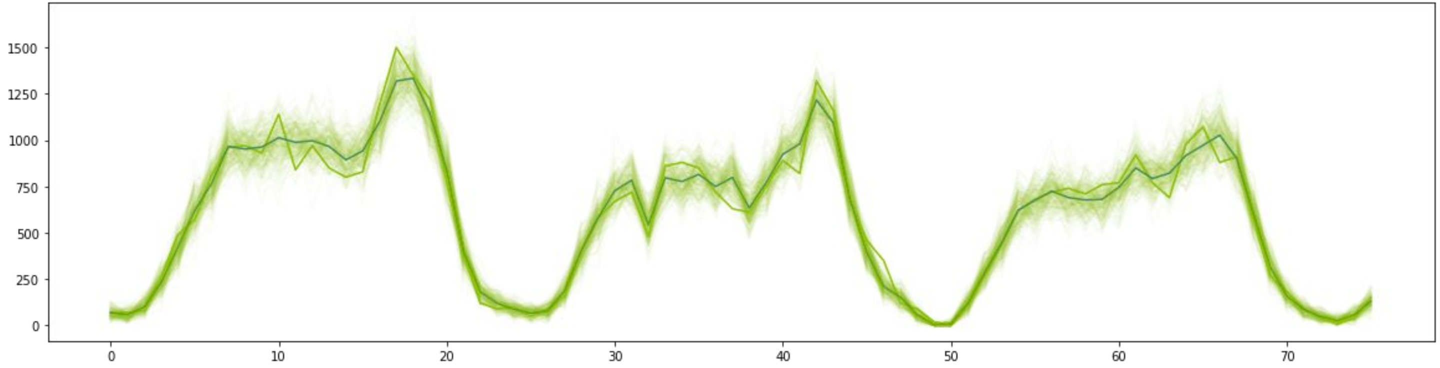
Measured stddev: 2.98

Measured stddev: 29.8

Sampling - Request Rates



Sampling - Request Rates



Sampling - Request Rates

10 rpm

10% sampling

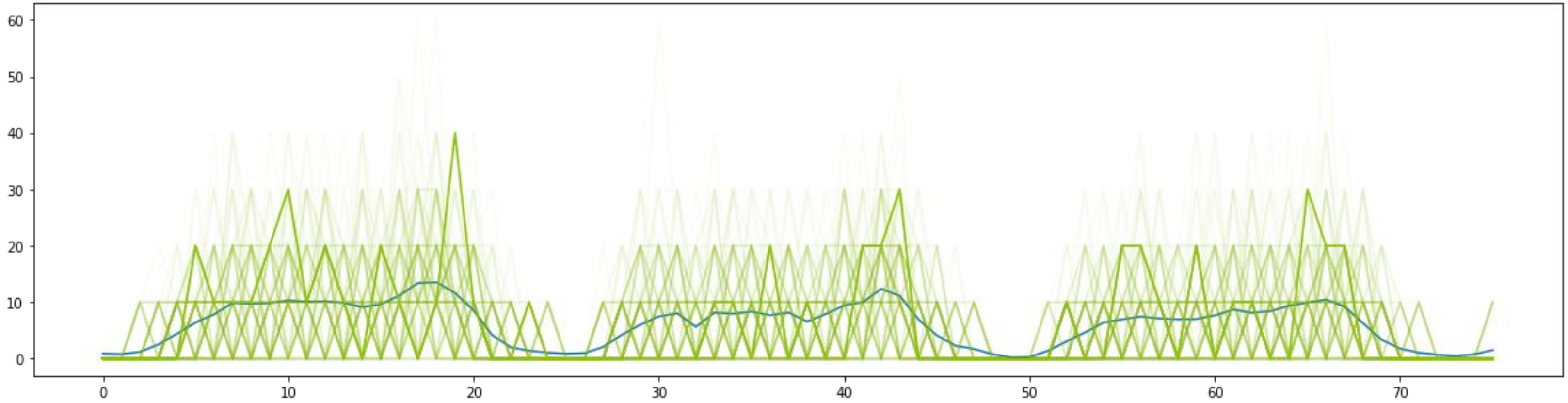
~ 1
rpm

estimate 10x

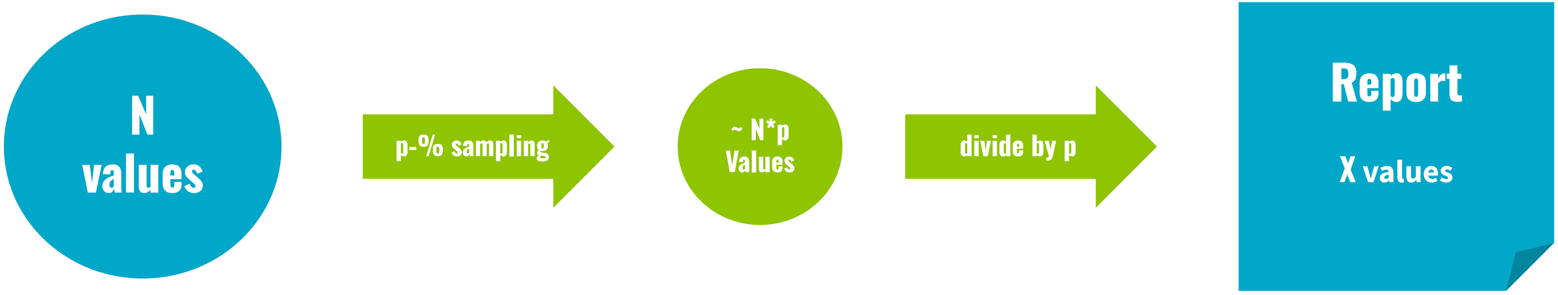
94%
Error

Chart

10 rpm (+/-94%)



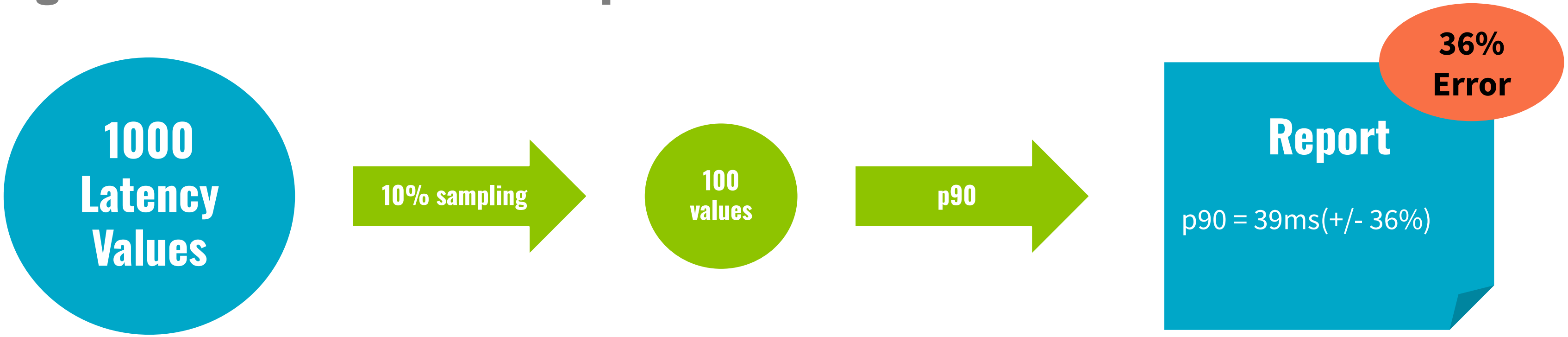
Bernoulli Sampling Theory

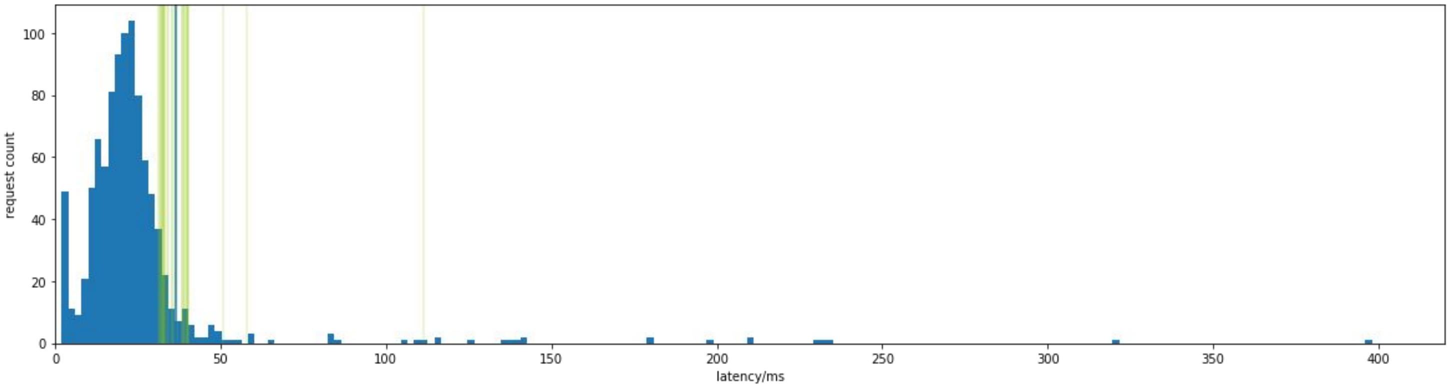


<u>Proposition</u>

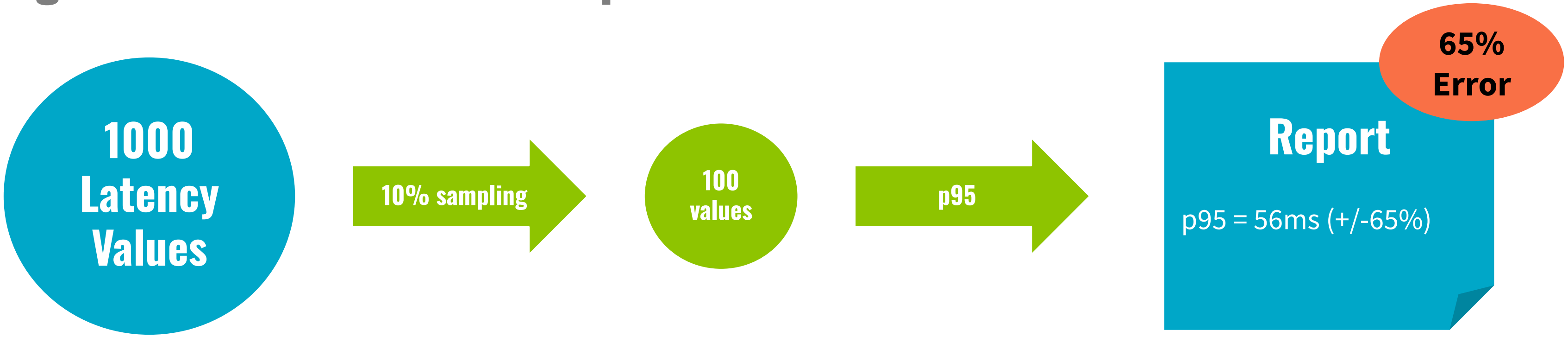
- X is a random variable following a scaled binomial distribution
- The expected value of X is N
- The standard deviation of X is $\mathrm{std}(X) = \sqrt{\frac{1-p}{N \cdot p}}$

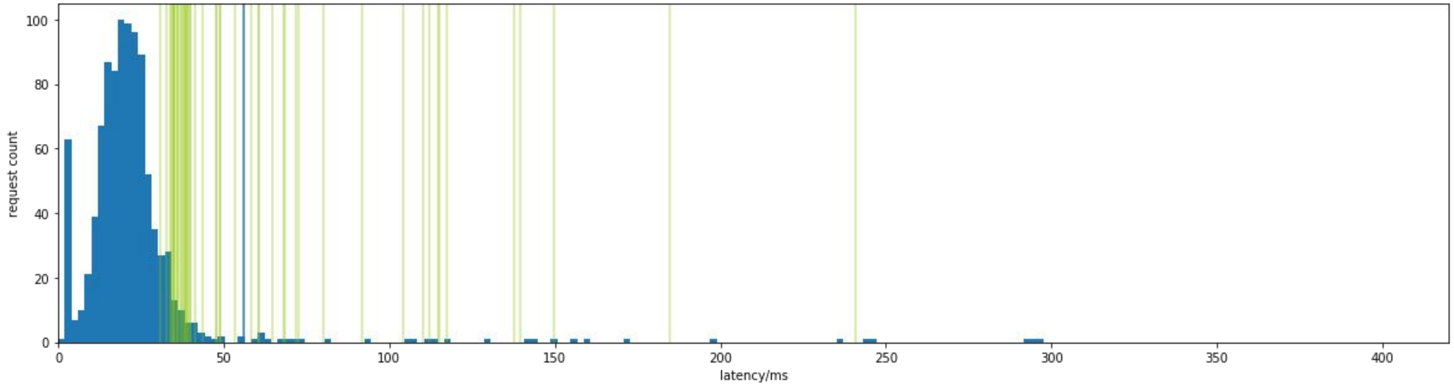
Sampling - Simulation - Percentiles p90





Sampling - Simulation - Percentiles p95





Sampling - Simulation - Percentiles

800K Latency Values over 24h

10% sampling

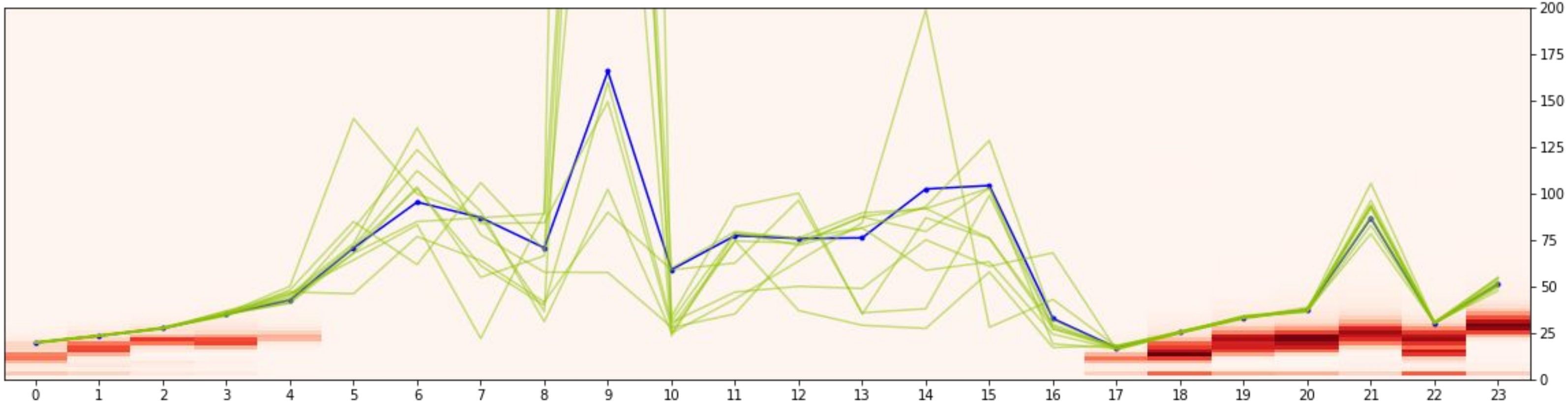
80K Values /24h

p90

? % Error

Report

p90 = ?



The Sampling Error Calculator

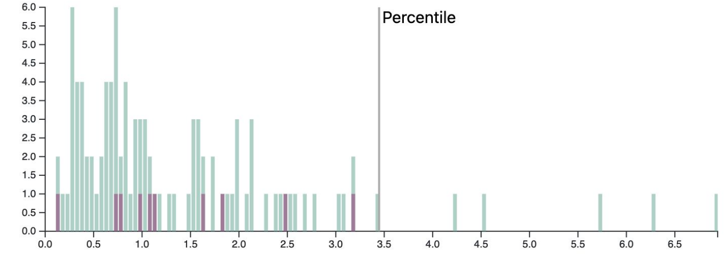
heinrichhartmann.com/sampling

# Sampling			
Sampling Rate	10 %		

# Request Rates			
Request Rate	100 rpm		
Time window	1 min (60 sec)		
Population	Total 100 requests contained in 60 sec time-window.		
## Samping Effects on Request Rate Estimates			1000 iterations
Sample	We expect to retrain 10 requests, after sampling with 10% probability.		
	Value	Standard Error	Realtive Error
Estimate Req. Count	100.0 req	± 30.00 req	30.00%
Simulate Req. Count	98.3 req	± 30.31 req	30.82%
Estimate Req. Rate	1.7 rps	± 0.50 rps	30.00%
Simulate Req. Rate	1.6 rps	± 0.51 rps	30.82%

# Error Rates			
Error Rate	5 %		
Population	From the 100 requests 5 are marked as error.		
## Samping Effects on Error Rate Estimates			1000 iterations
Sample	We expect to retrain 0.5 errors in the sample of size 10.		
	The probability that no error will be retained is 59.049000000%		
Estimate Err. Rate	5.0 %	± 7.02 ppt	140.36%
Simulate Err. Rate	4.9 %	± 7.33 ppt	150.90%

# Latency			
Latency Distribution	LogNormal		
Percentile p 95.0	3.4 ms		
Population	Total 100 requests following LogNormal distribution.		
	The true p95.0 is at 3.4ms.		
## Samping Effects on Latency			1000 iterations
Sample	We expect to retain 10.0 requests.		
Estimate Percentile p95.0	2.78 ms	± 1.68 ms	60.45%



Sampling - Take Aways

- Effective way to trade costs vs. accuracy
- Accuracy loss depends on sample size and other factors
- Simulation gives effective tool to study sampling impact

Thank You!

Further Reading

- twitter: @HenrichHartmann
- blog: heinrichhartmann.com
- source: github.com/HeinrichHartmann/Statistics-for-Engineers