Anomaly Detection in Time Series from Scratch

Using statistical analysis

by Ivan Shubin

Booking.com

Ivan Shubin

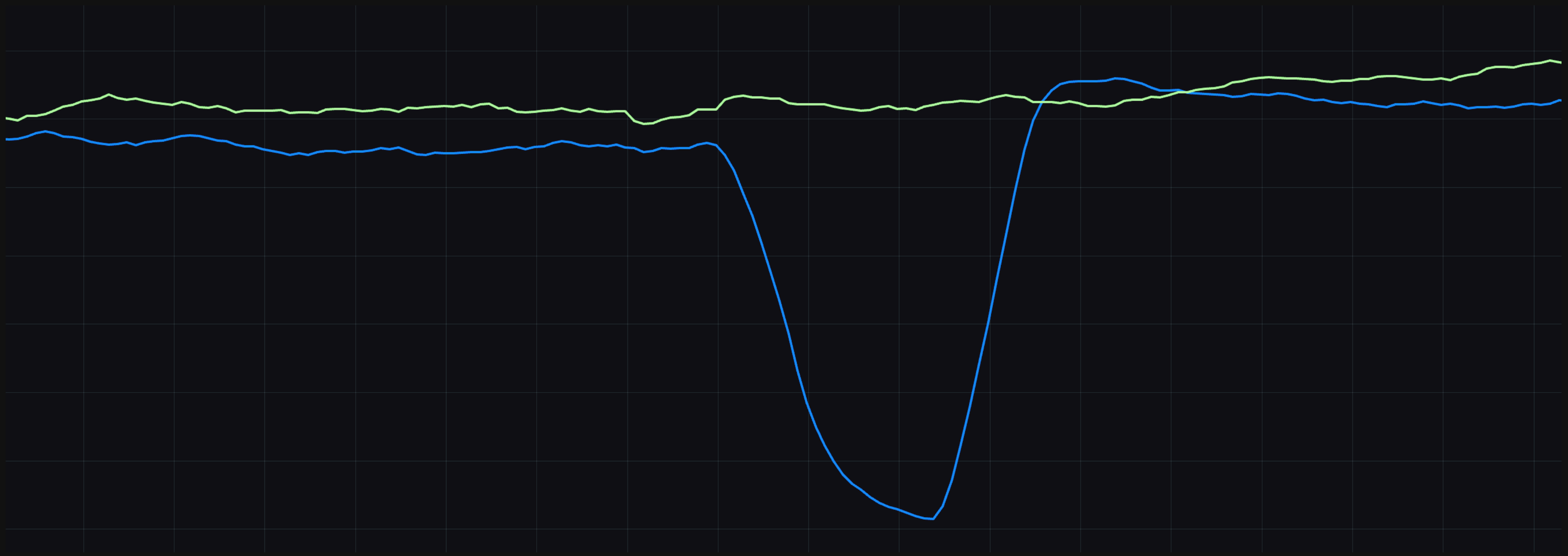
Bio

- Booking.com (Senior SRE)
- TomTom (SRE)
- eBay Classifieds Group (SRE, Dev, QA)
- Author of Schemio and Galen Framework

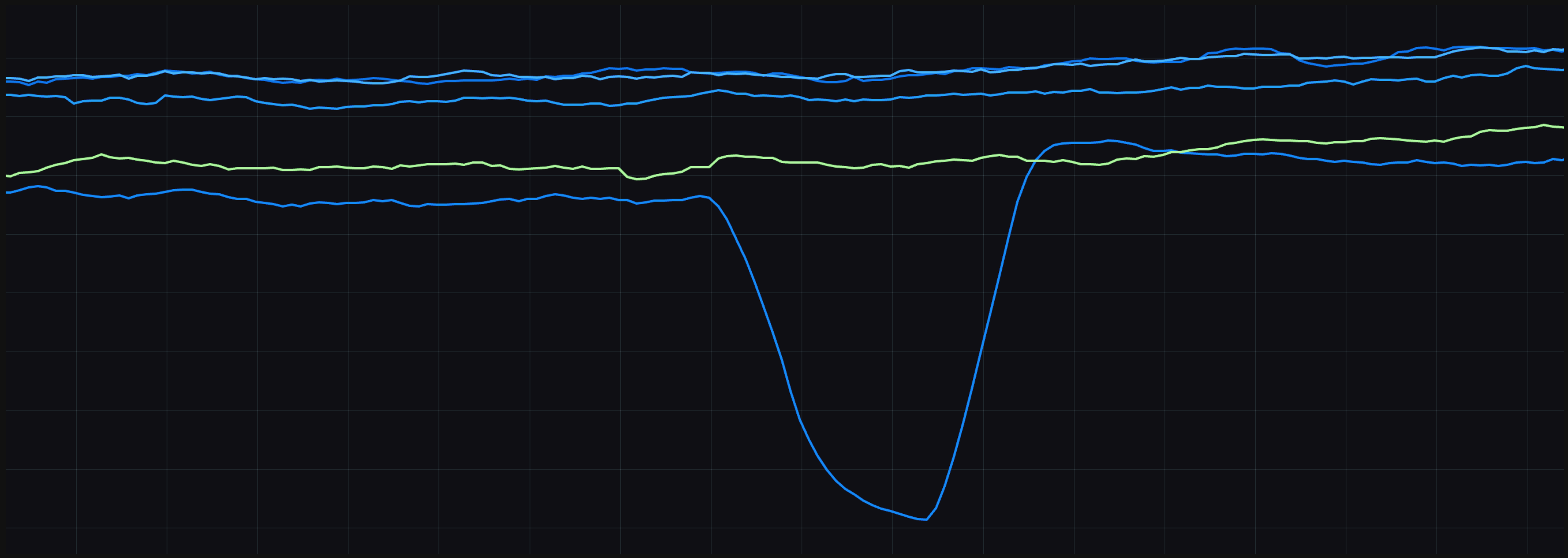
Links

https://github.com/ishubin
https://www.linkedin.com/in/ivan-shubin

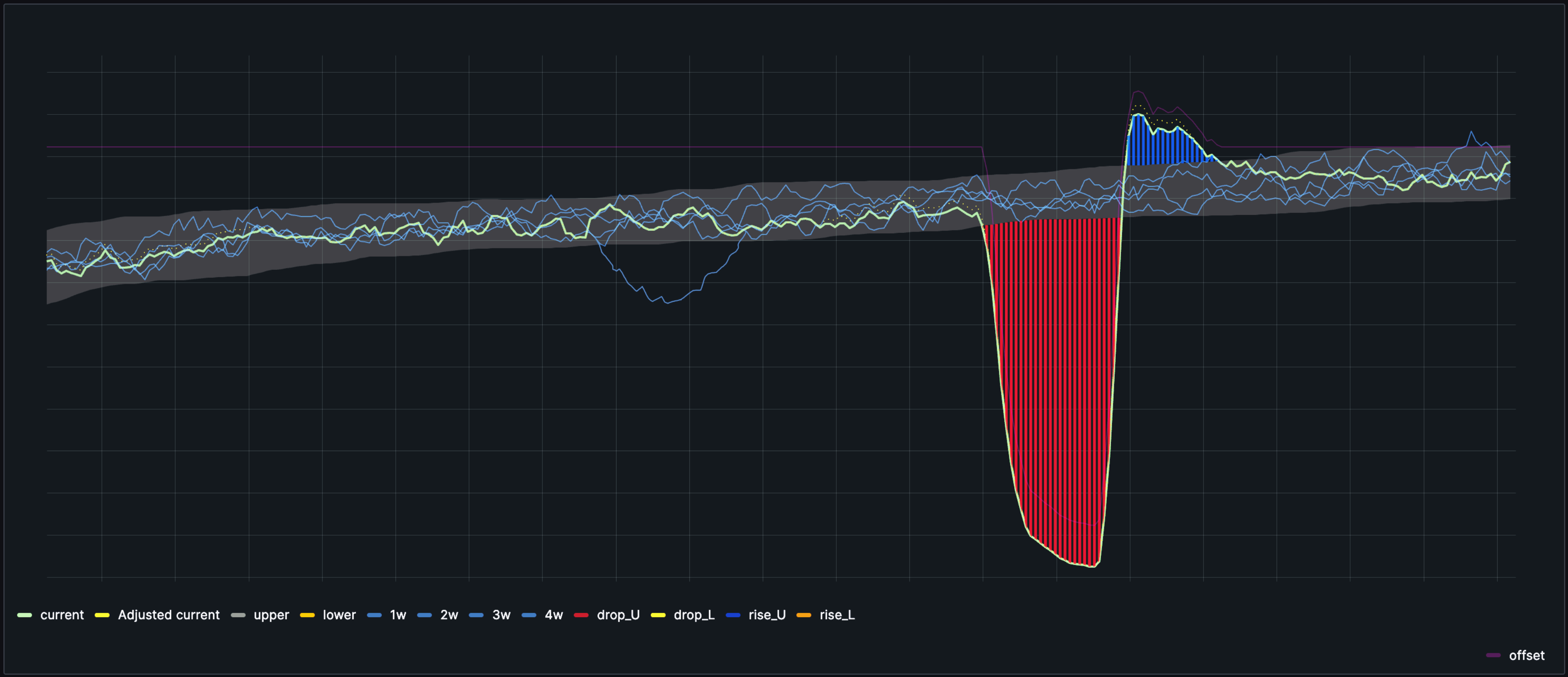
Simple approach: Current vs Previous week



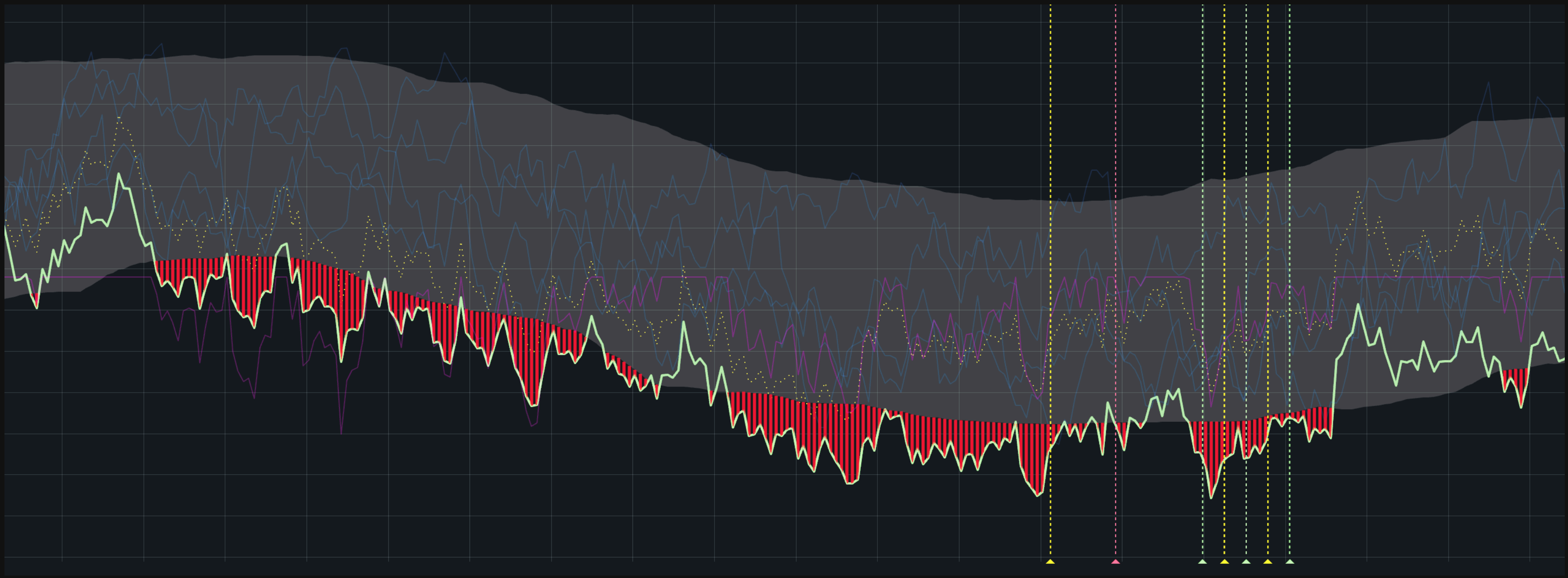
Simple approach: Current vs Previous week



How it should look like



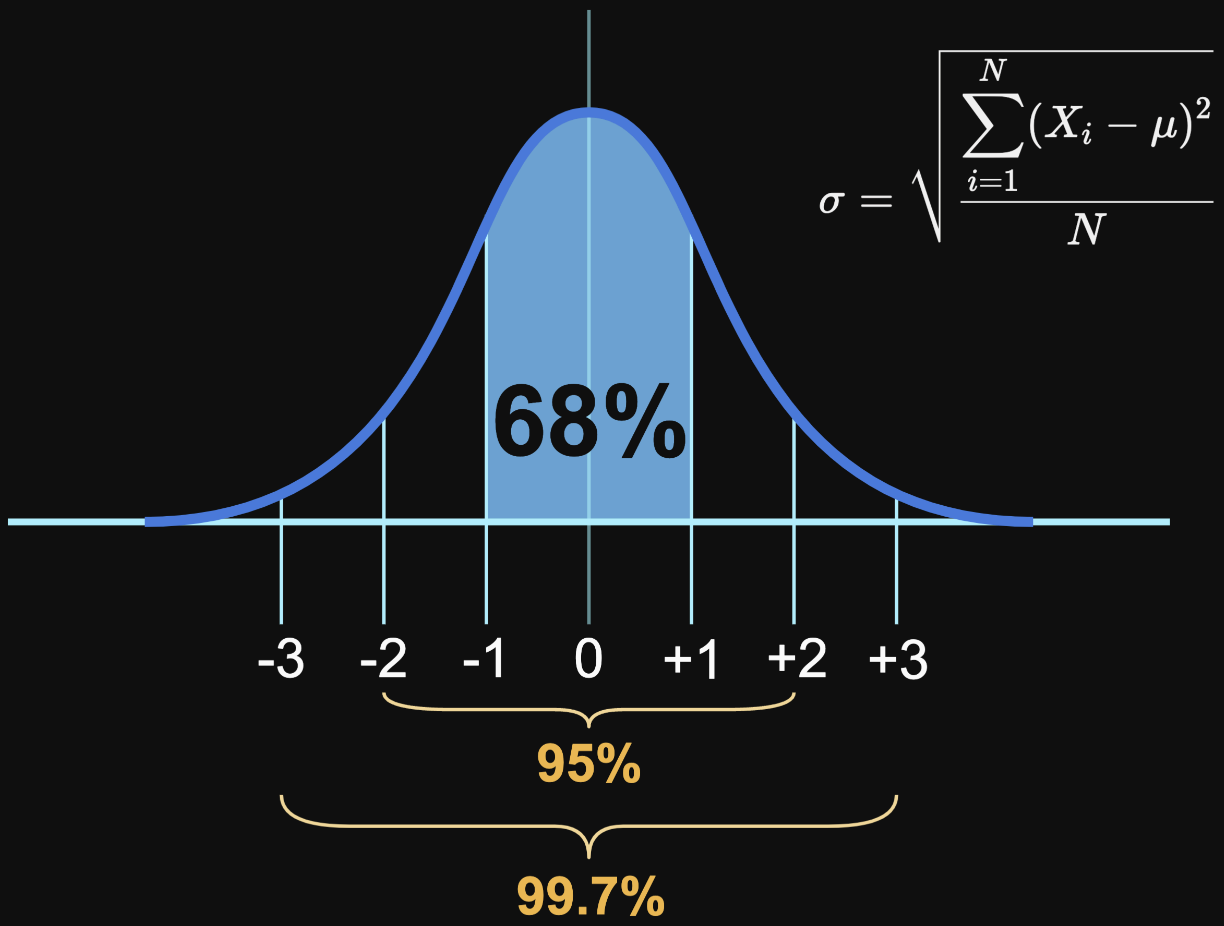
We wanted to detect "slow burns"



Statistics


34.1%
34.1%
13.6%
13.6%
2.15%
2.15%
Z - Distribution
T - Distribution
99.7%
95%
(n)
-2 -1 μ=0 1 2
-1.96
1.96
-2.201
2.201
50
40
30
20
10
0
100 120 140 160 180 200
μ-3σ μ-2σ μ-σ μ μ+σ μ+2σ μ+3σ

Standard deviation



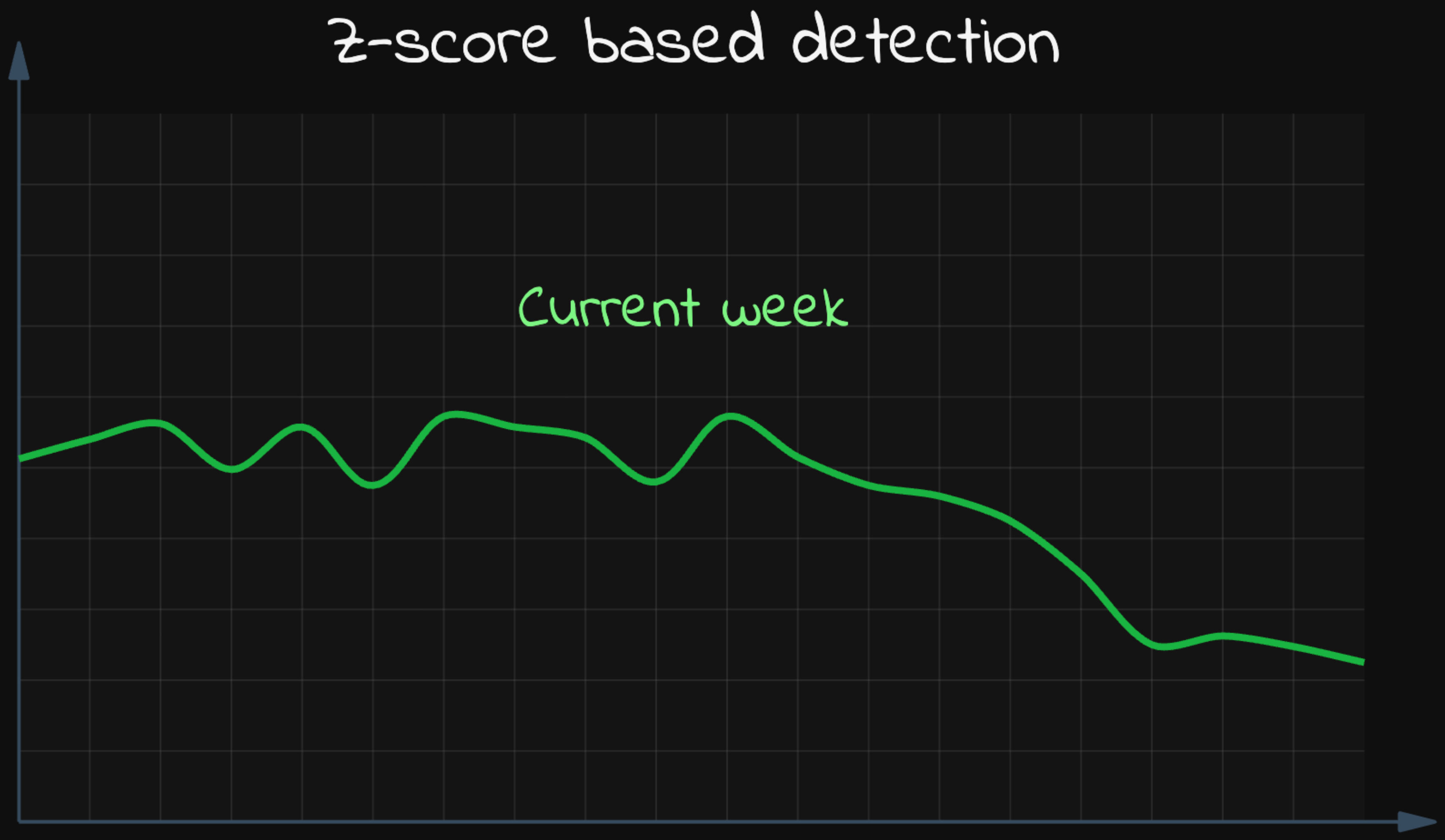
$$Zscore = \frac{x - \mu}{\sigma}$$

Point value

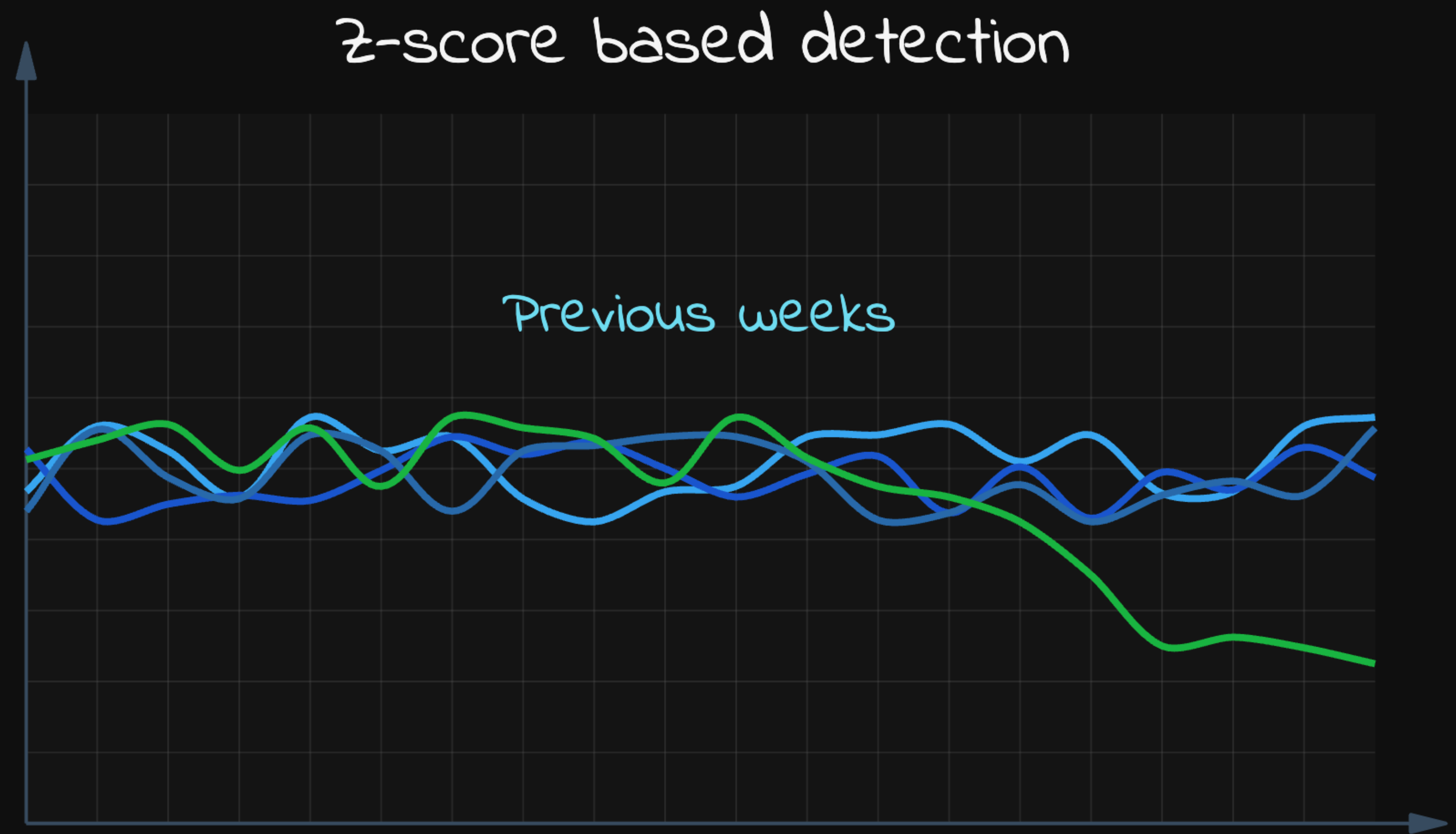
Mean

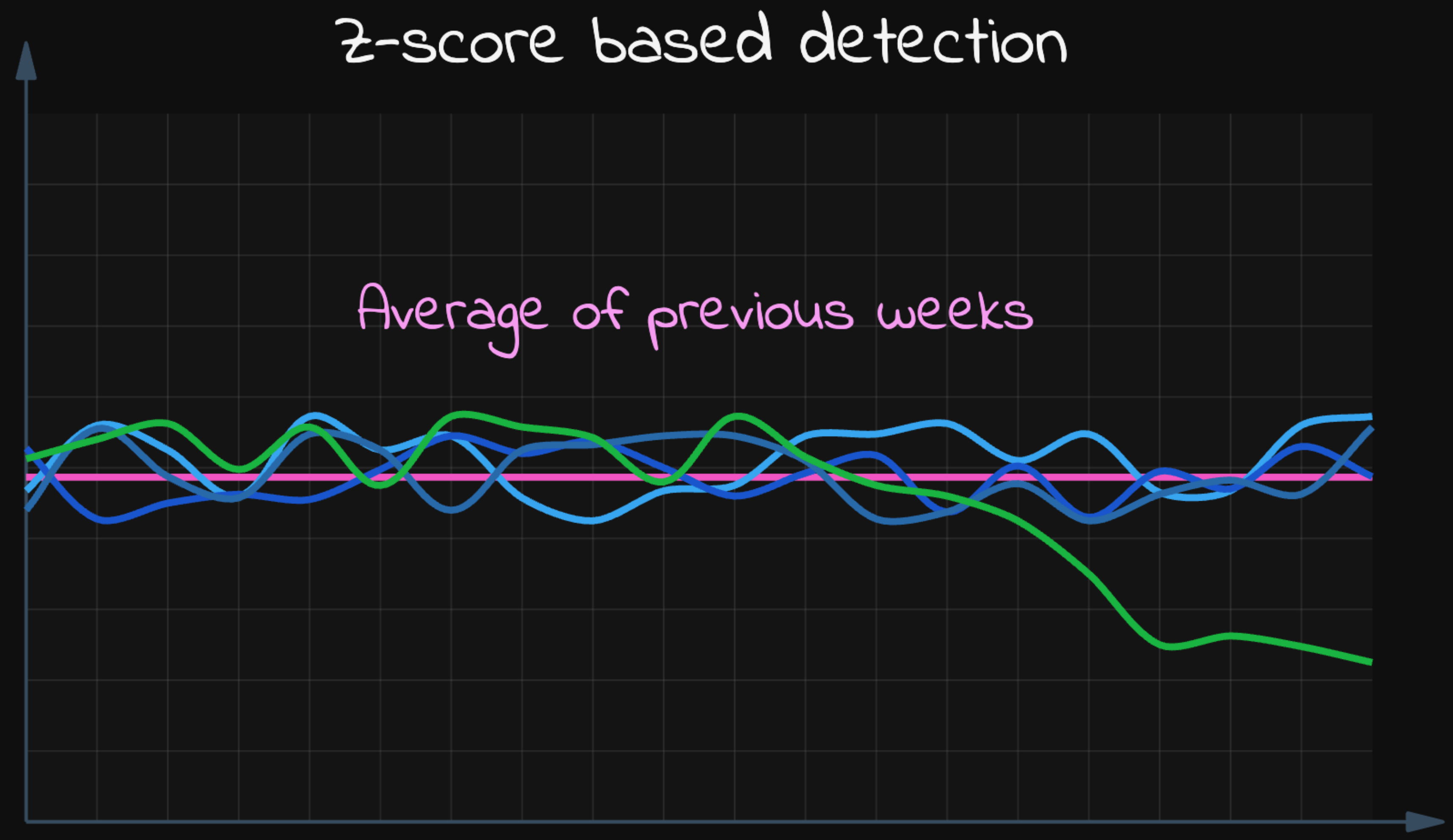
Standard Deviation

z-score	Confidence
0.	50%
1.	84%
1.5	93%
2.	97.7%
3.	99.87%

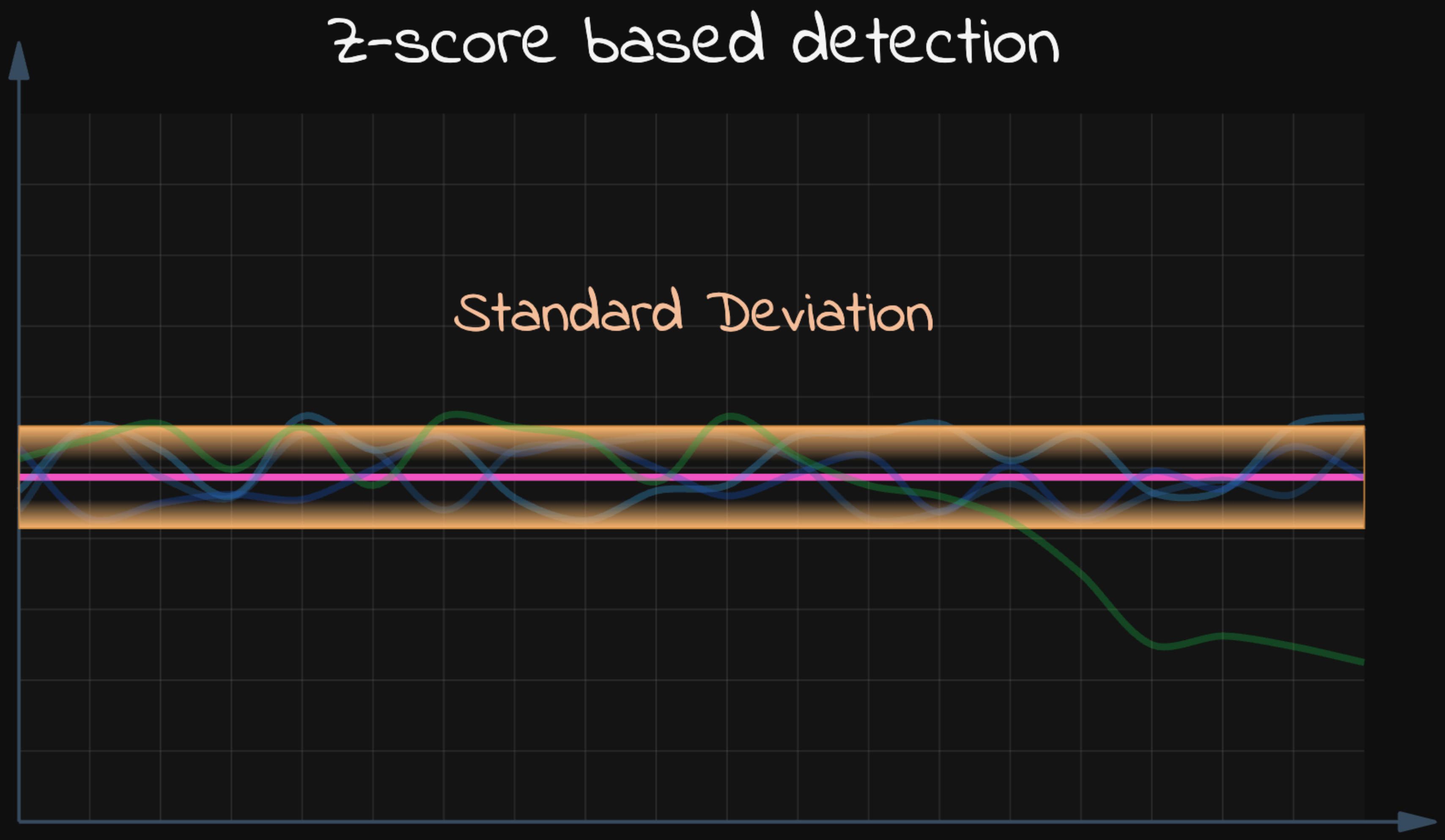


Z-score based detection
Current week

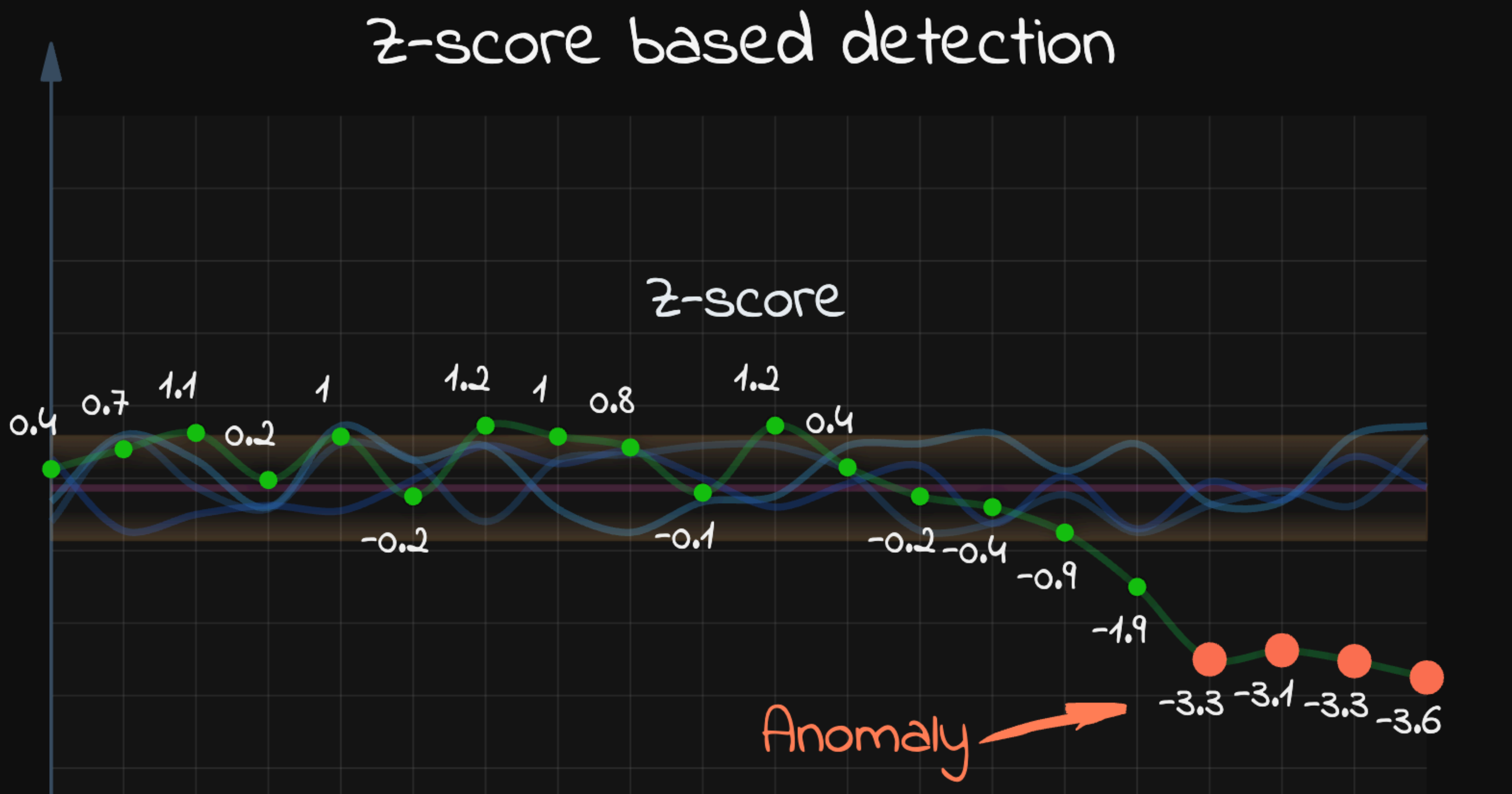

Z-score based detection
Previous weeks



Z-score based detection
Average of previous weeks


Z-score based detection
Standard Deviation

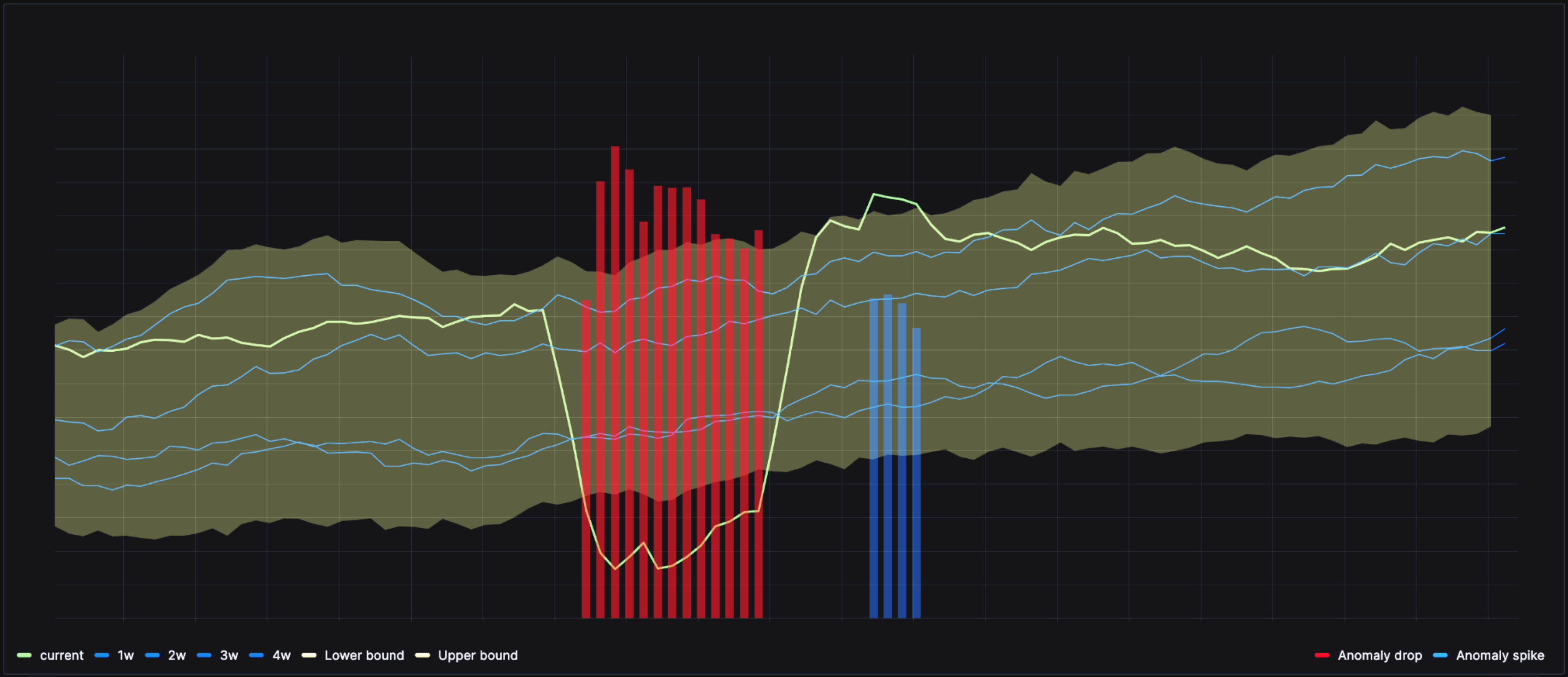
z-score based detection
z-score
0.4
0.7
1.1
0.2
1
-0.2
1.2
1
0.8
-0.1
1.2
0.4
-0.2
-0.4
-0.9
-1.9
-3.3
-3.1
-3.3
-3.6



Z-score based detection
Z-score
0.4
0.7
1.1
0.2
1
-0.2
1.2
1
0.8
-0.1
1.2
0.4
-0.2
-0.4
-0.9
-1.9
-3.3
-3.1
-3.3
-3.6
Anomaly

Can we use Graphite itself?

Graphite based anomaly detection

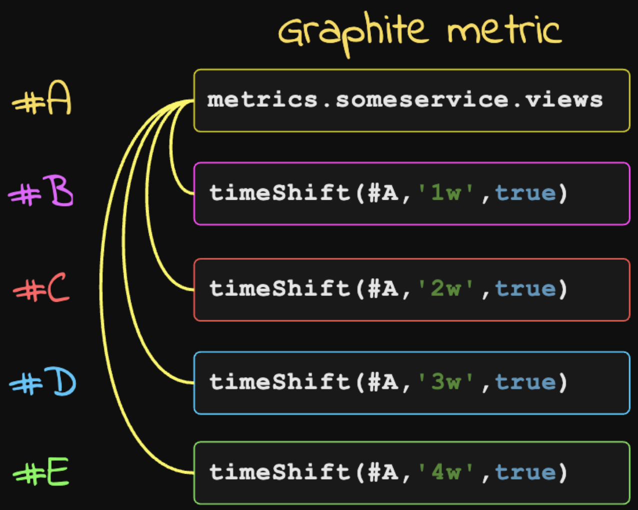


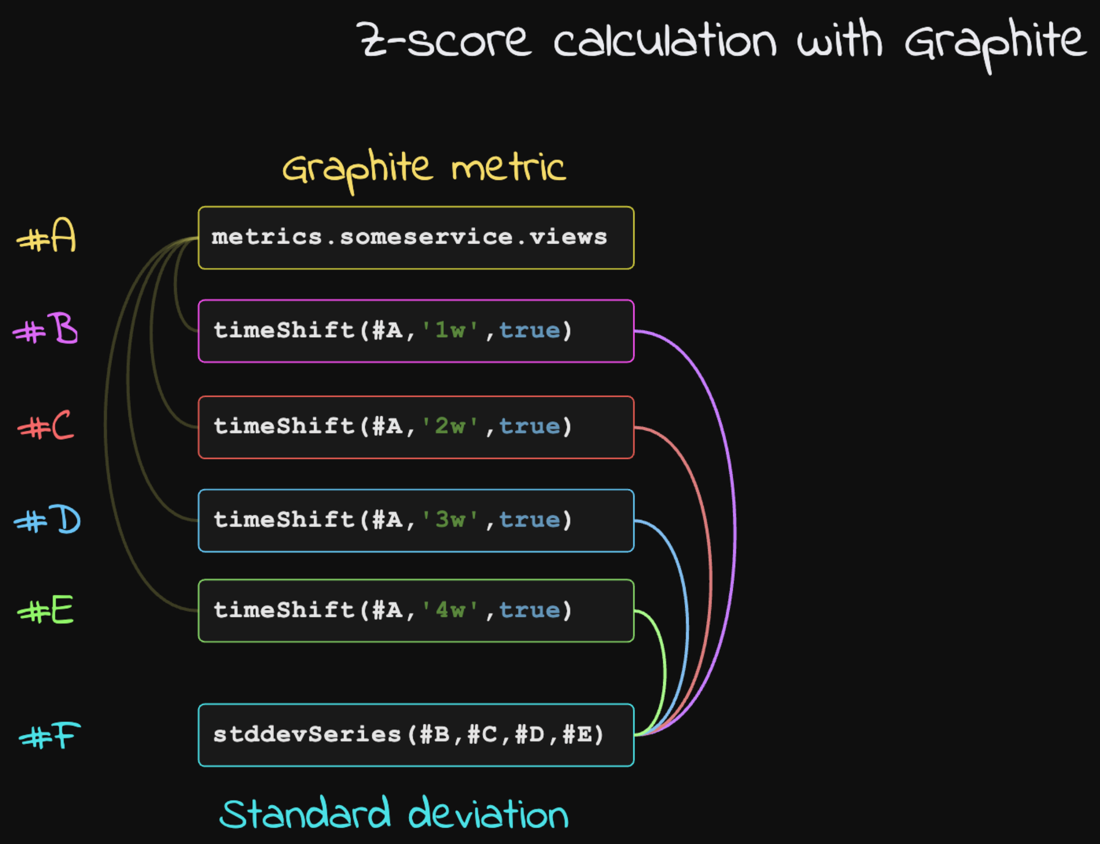
Z-score calculation with Graphite

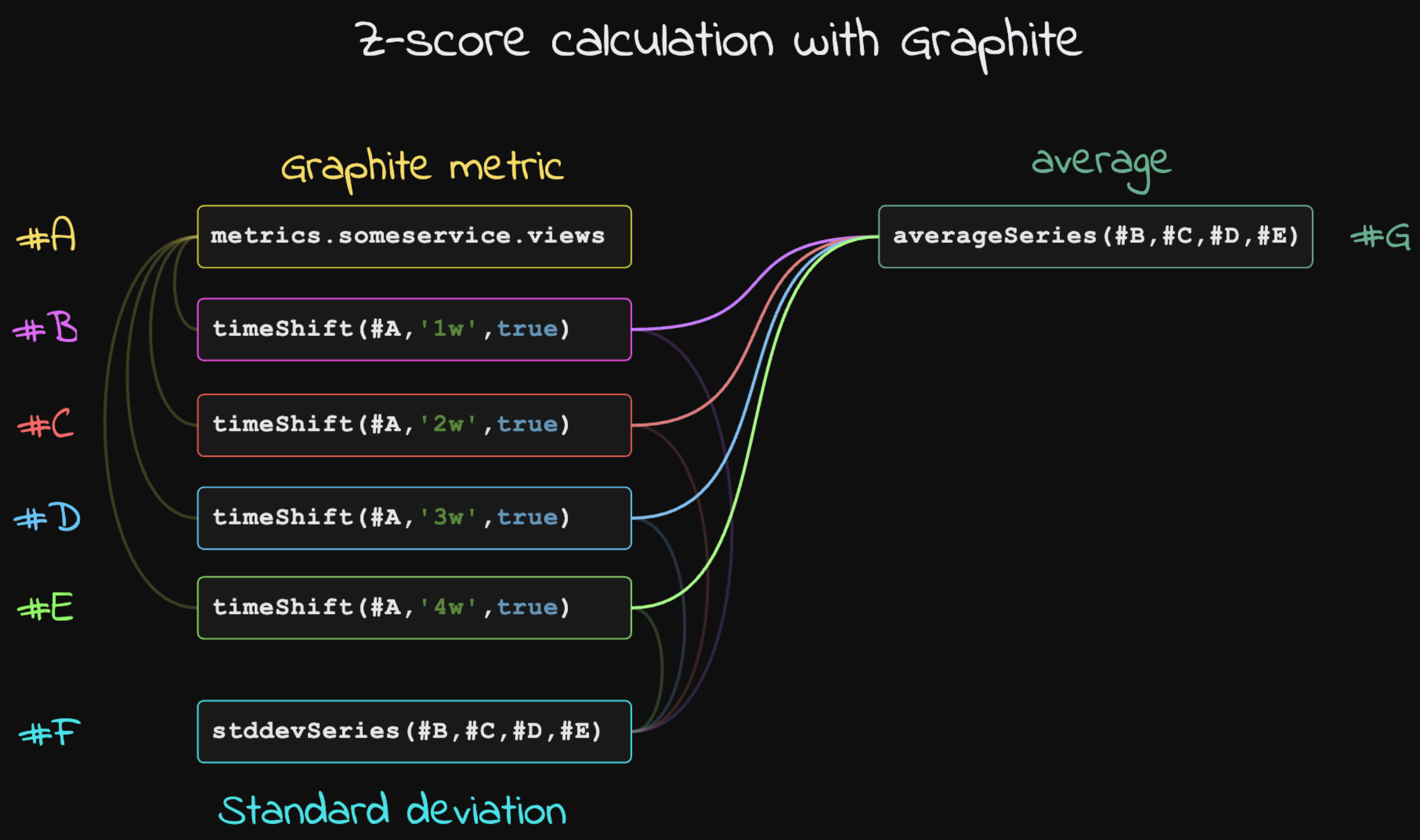
Graphite metric

#A `metrics.someservice.views`

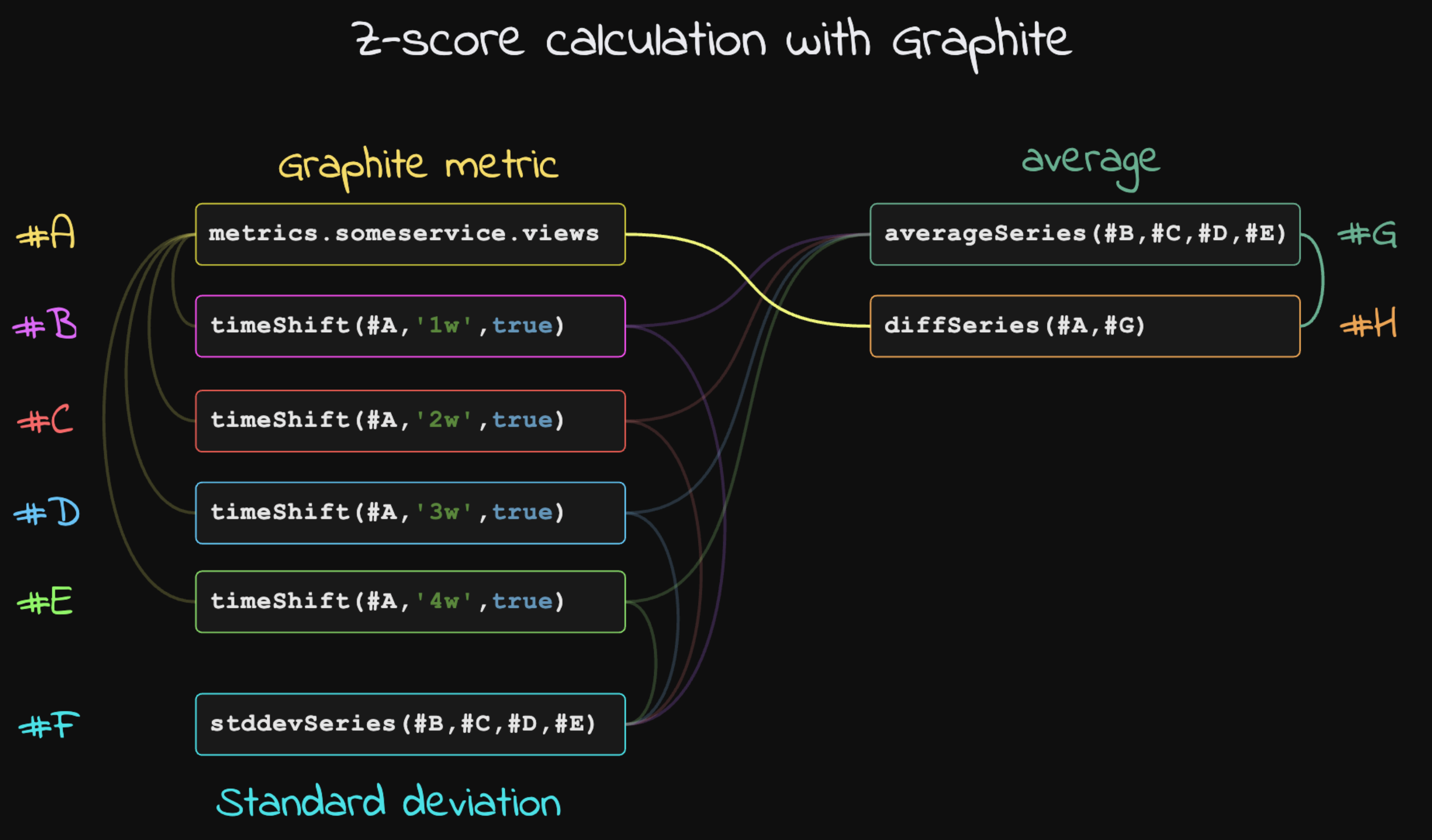
Z-score calculation with Graphite

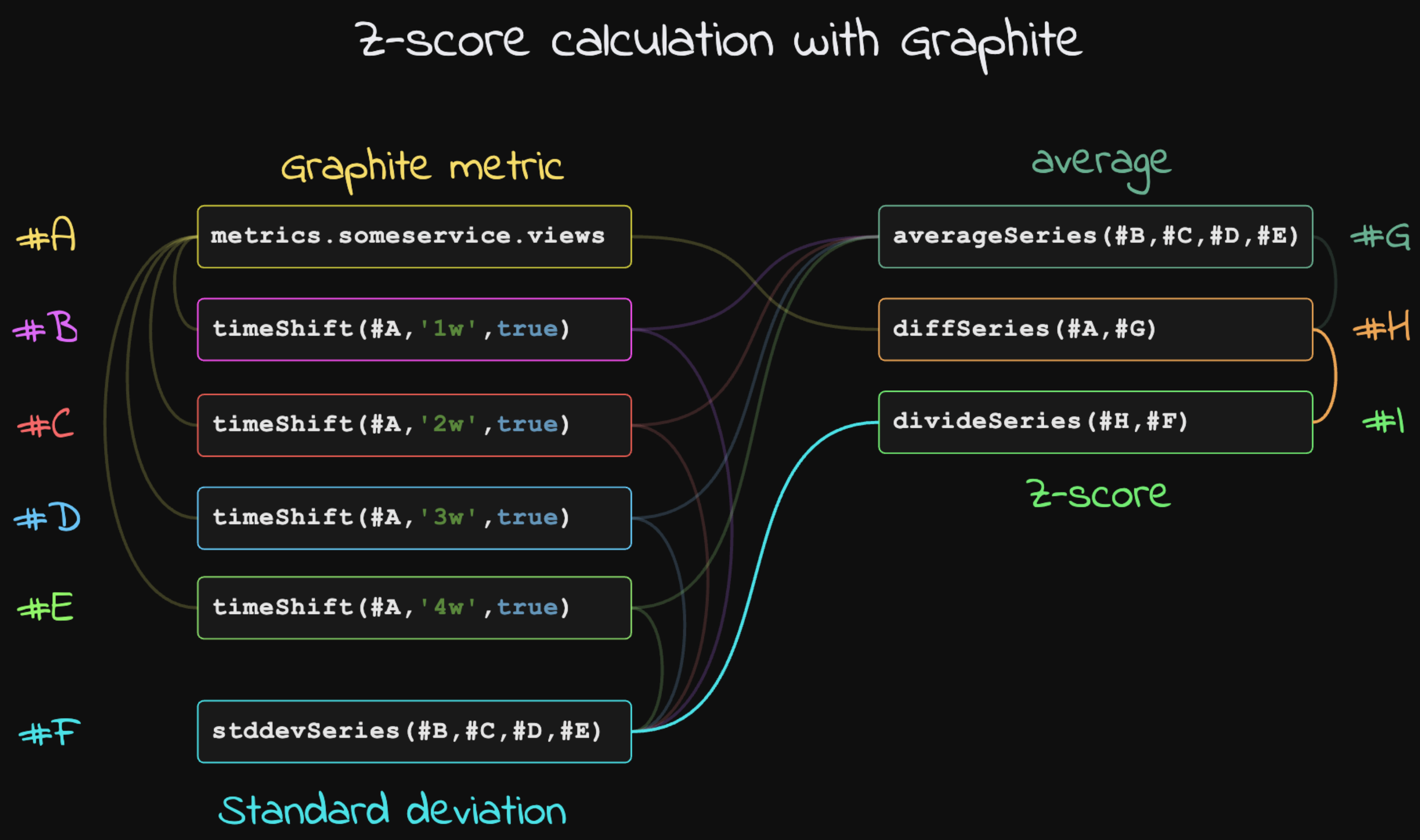



Z-score calculation with Graphite
Graphite metric
#A
metrics.someservice.views
#B
timeShift(#A,'1w',true)
#C
timeShift(#A,'2w',true)
#D
timeShift(#A,'3w',true)
#E
timeShift(#A,'4w',true)
#F
stddevSeries(#B,#C,#D,#E)
Standard deviation



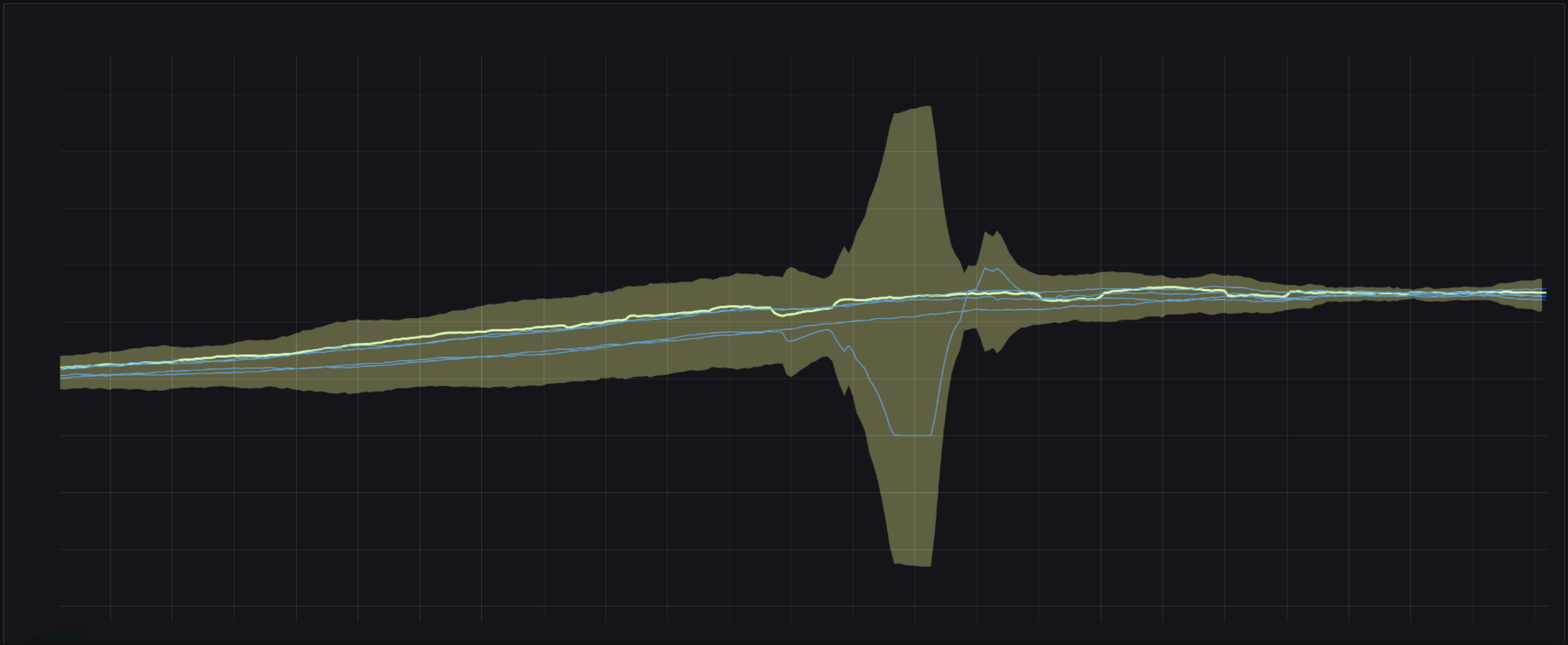
Z-score calculation with Graphite
Graphite metric
average
#A
metrics.someservice.views
averageSeries(#B,#C,#D,#E)
#G
#B
timeShift(#A,'1w',true)
#C
timeShift(#A,'2w',true)
#D
timeShift(#A,'3w',true)
#E
timeShift(#A,'4w',true)
#F
stddevSeries(#B,#C,#D,#E)
Standard deviation

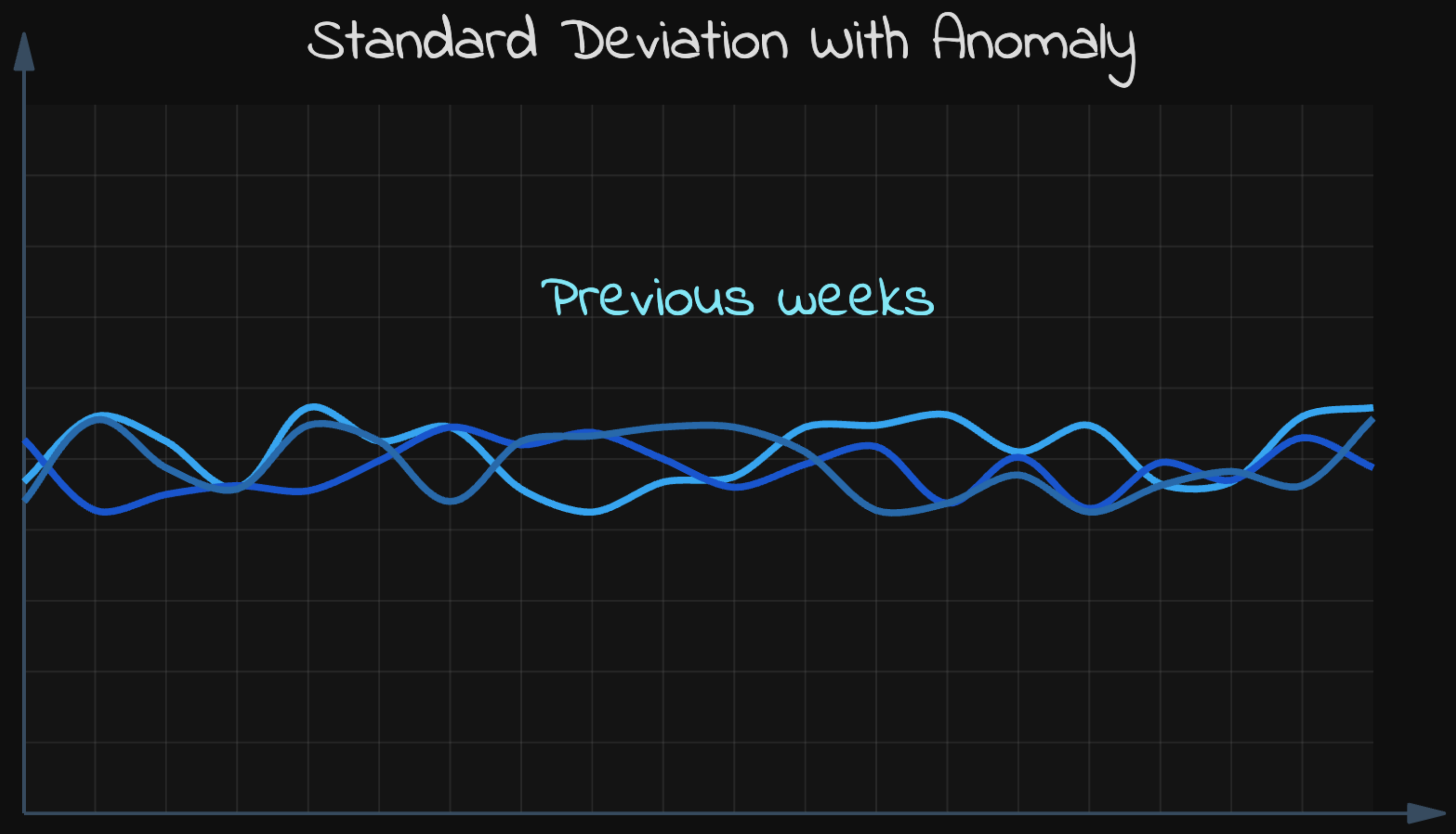

Z-score calculation with Graphite
Graphite metric
average
#A
metrics.someservice.views
#B
timeShift(#A,'1w',true)
#C
timeShift(#A,'2w',true)
#D
timeShift(#A,'3w',true)
#E
timeShift(#A,'4w',true)
#F
stddevSeries(#B,#C,#D,#E)
Standard deviation
averageSeries(#B,#C,#D,#E)
#G
diffSeries(#A,#G)
#H

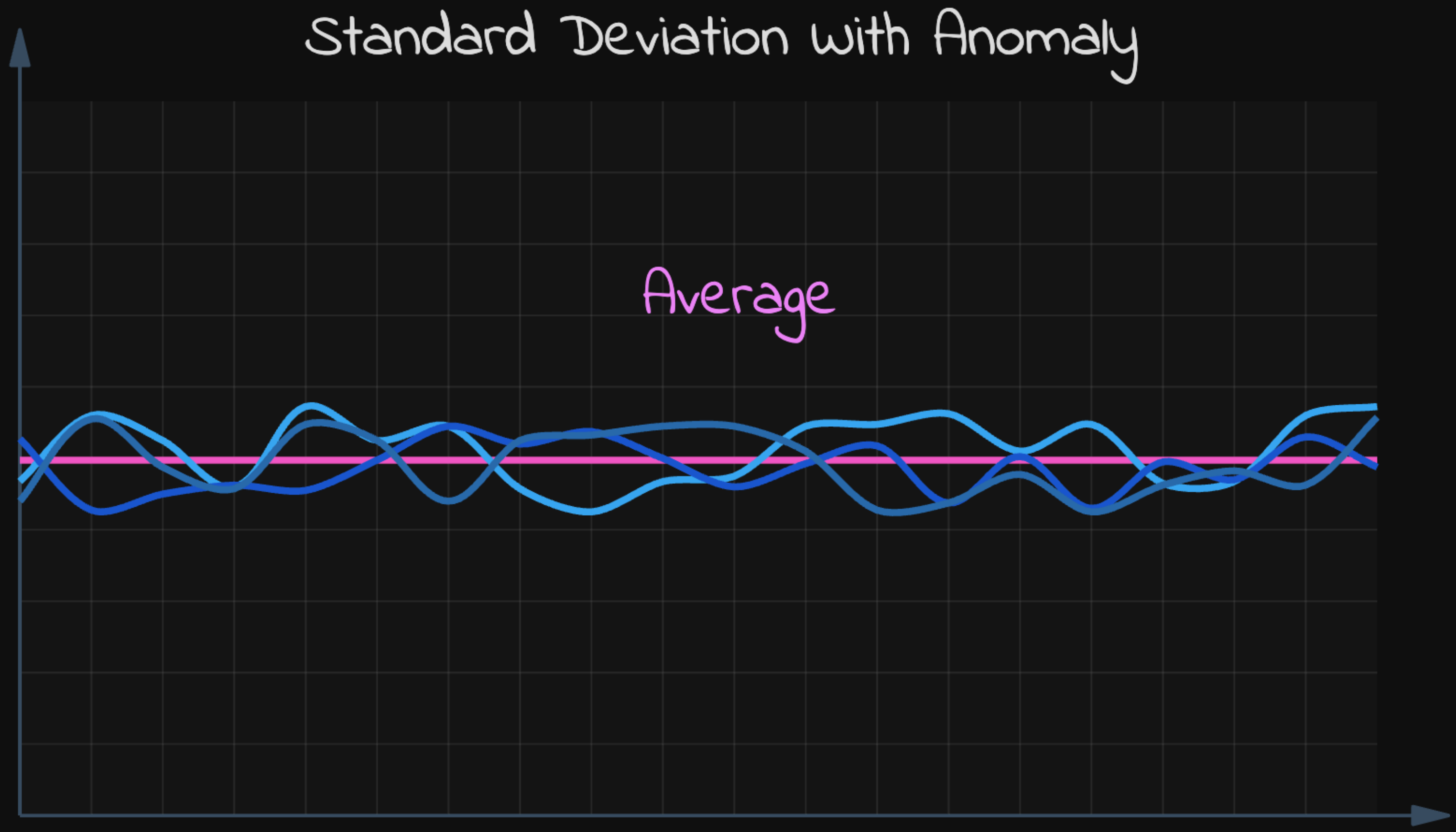


Z-score calculation with Graphite
Graphite metric
average
#A
metrics.someservice.views
averageSeries(#B,#C,#D,#E)
#G
#B
timeShift(#A,'1w',true)
diffSeries(#A,#G)
#H
#C
timeShift(#A,'2w',true)
divideSeries(#H,#F)
#I
Z-score
#D
timeShift(#A,'3w',true)
#E
timeShift(#A,'4w',true)
#F
stddevSeries(#B,#C,#D,#E)
Standard deviation

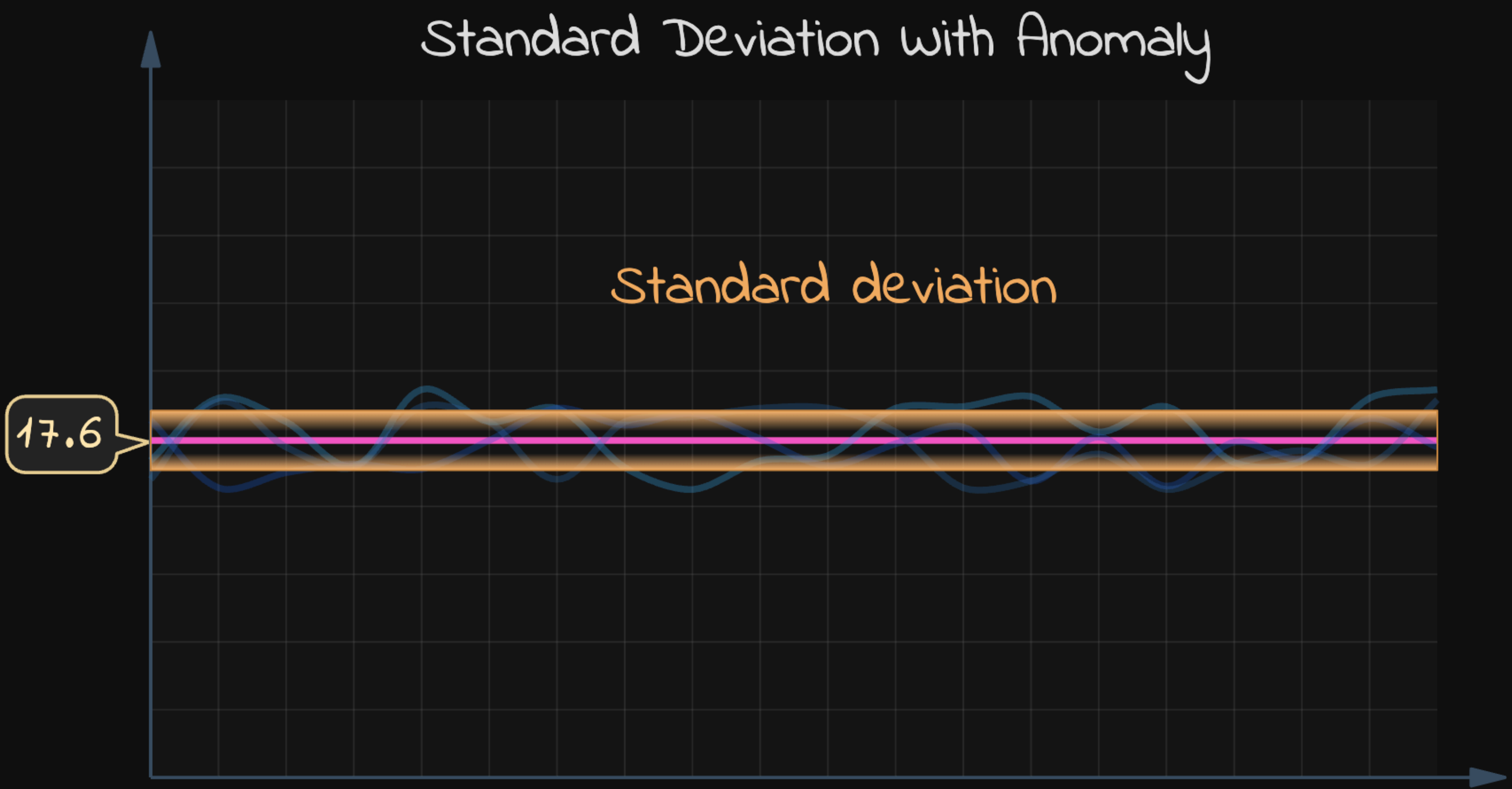
Past incidents distorts the prediction

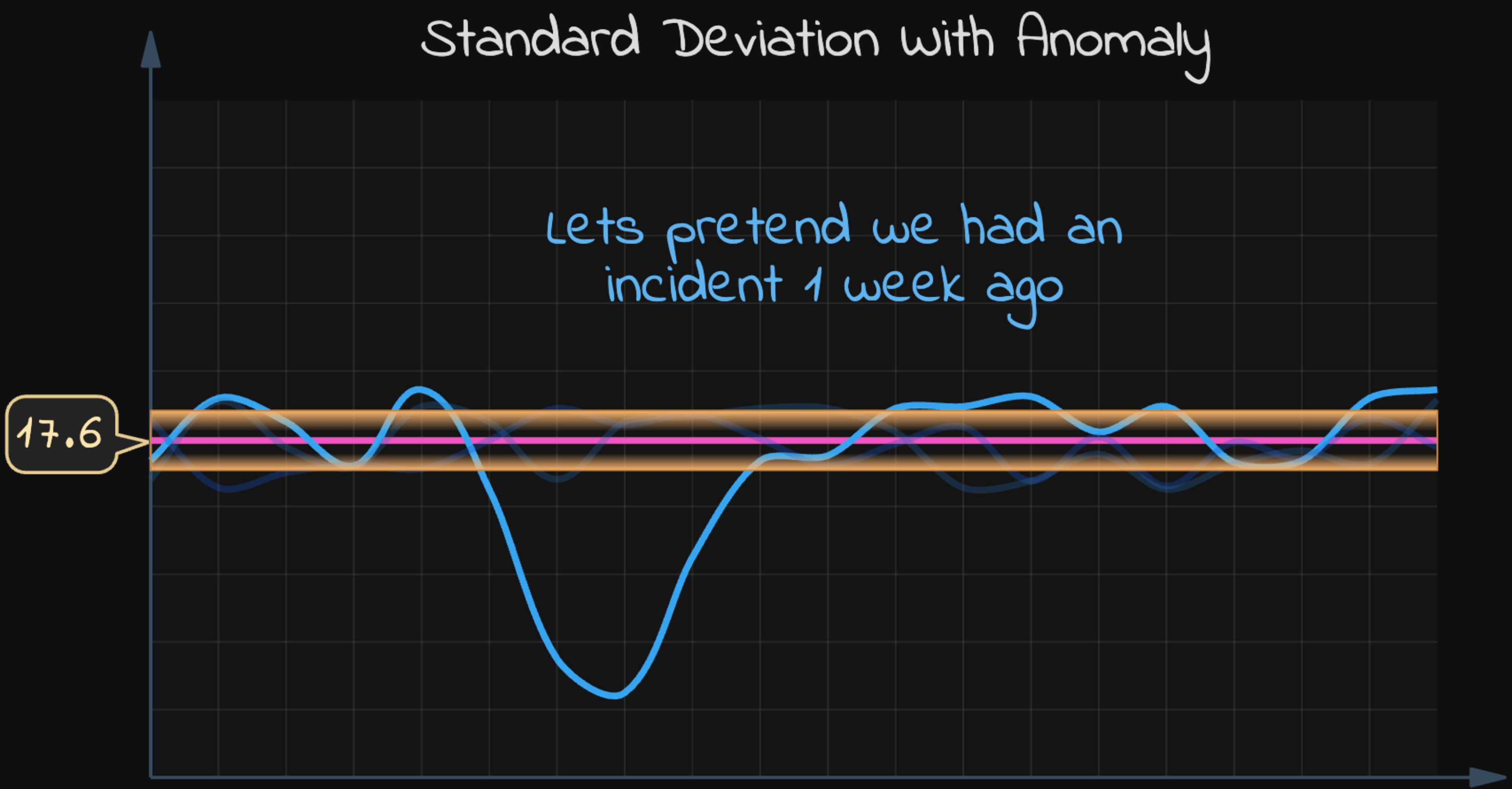


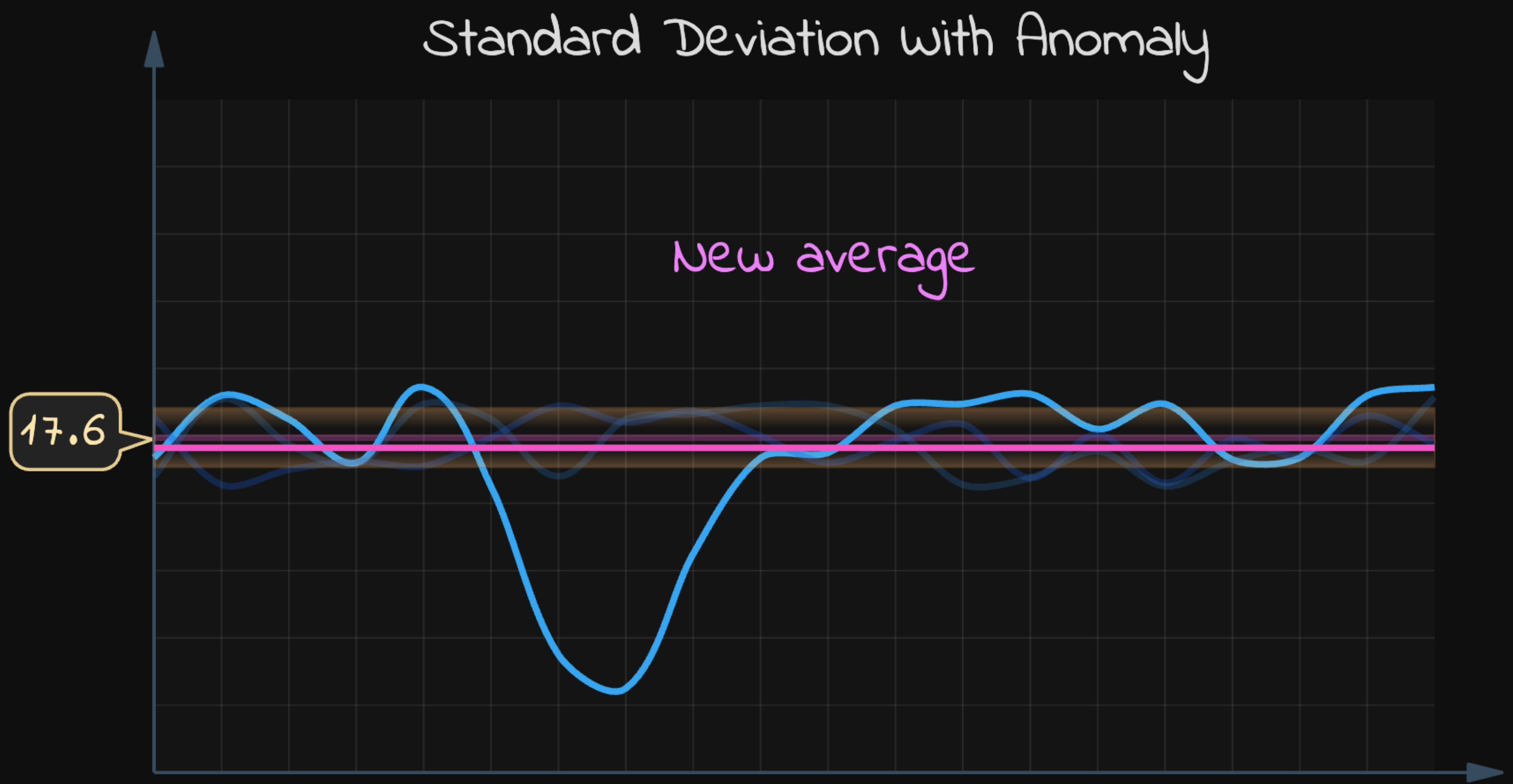

Standard Deviation with Anomaly
Previous weeks



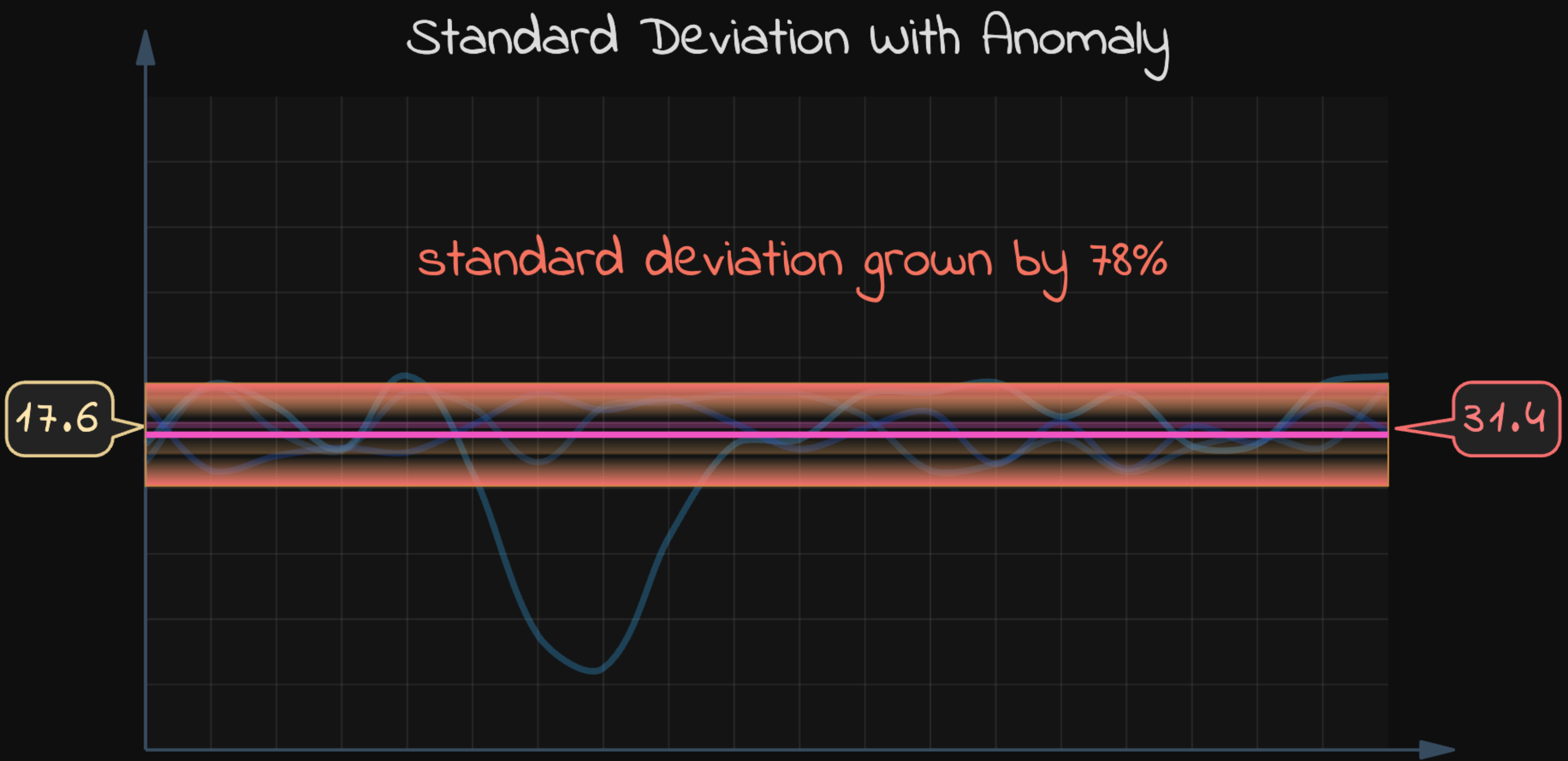
Standard Deviation with Anomaly
Average


Standard Deviation with Anomaly
Standard deviation
17.6


Standard Deviation with Anomaly
Lets pretend we had an
incident 1 week ago
17.6



Standard Deviation with Anomaly
New average
17.6


Standard Deviation with Anomaly
standard deviation grown by 78%
17.6
31.4

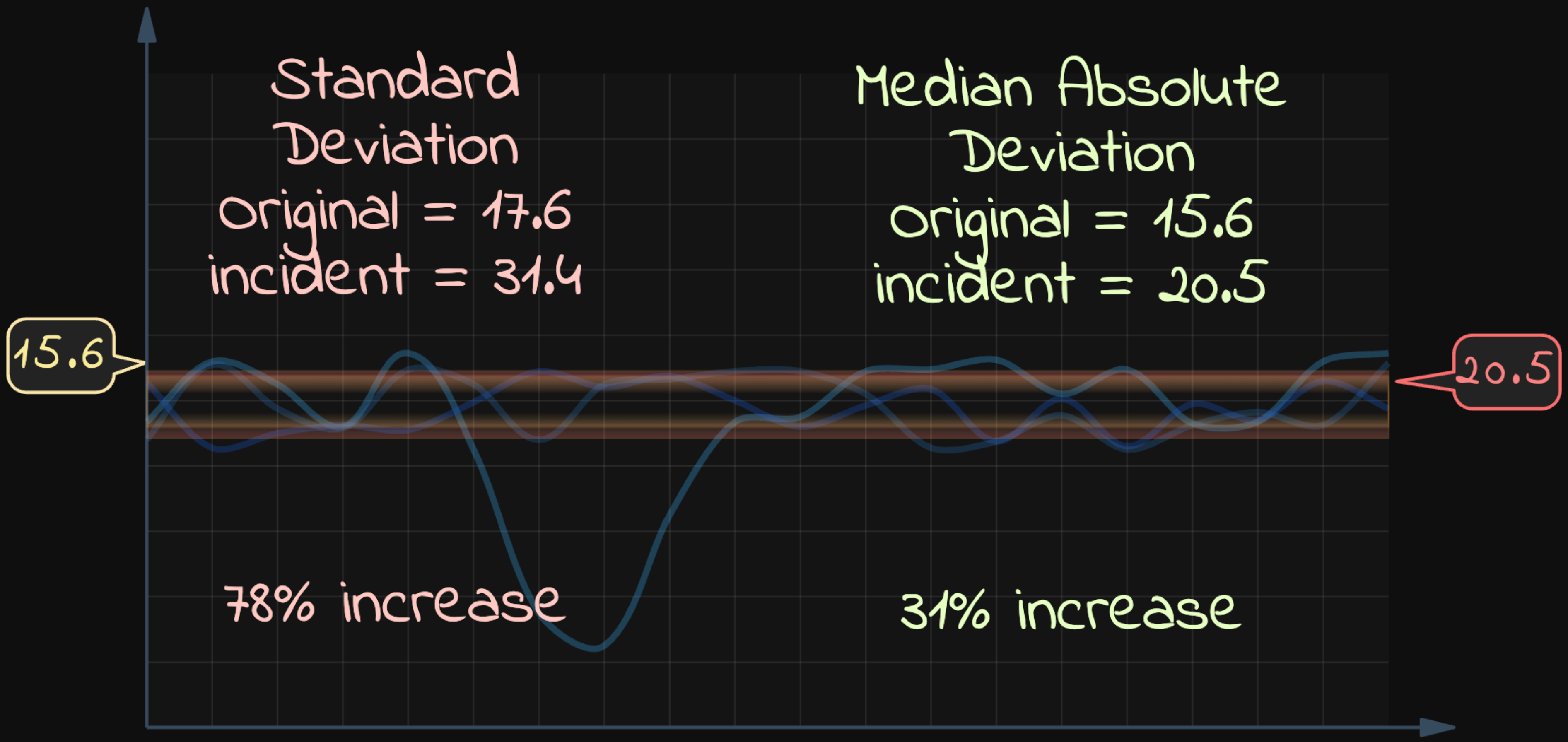
Median absolute deviation

Point value

Median

$$MAD = \frac{\sum_{i=1}^{N} |x_i - \mu|}{N}$$

Number of observations


Standard Deviation
original = 17.6
incident = 31.4
Median Absolute Deviation
original = 15.6
incident = 20.5
15.6
20.5
78% increase
31% increase

Z-score drawbacks

Too sensitive in smaller deviations

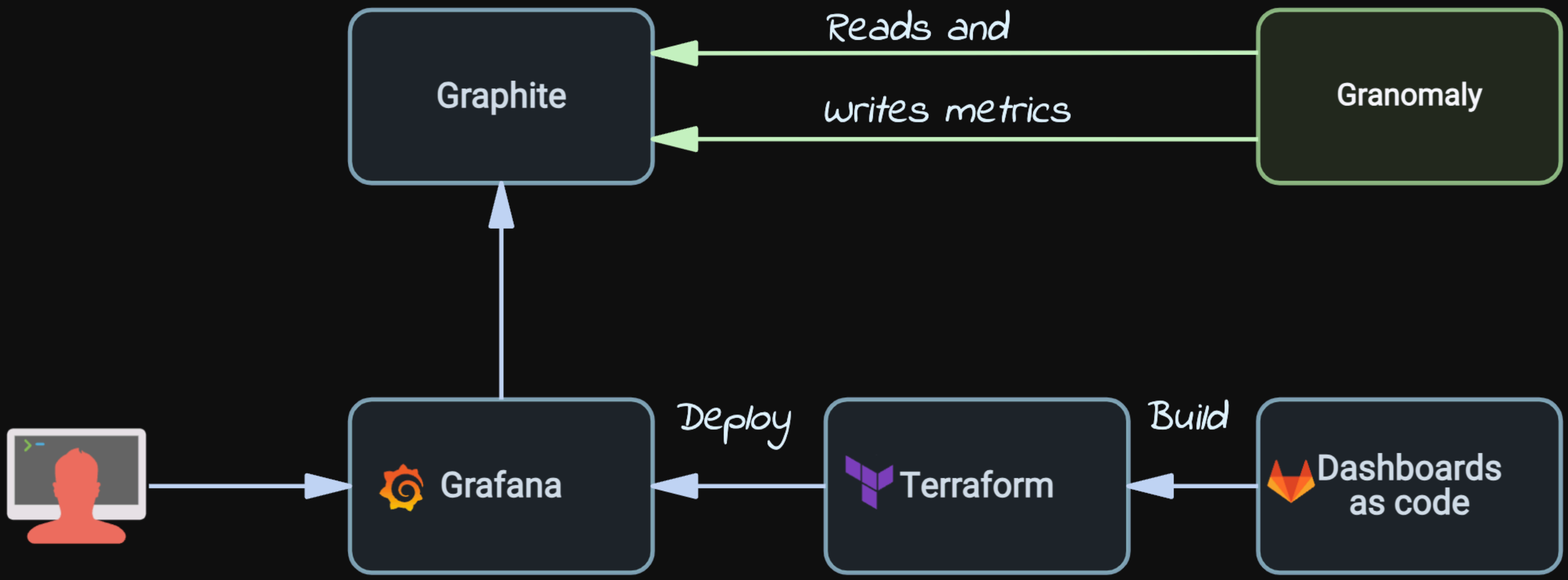
Too sensitive in smaller deviations

Granomaly

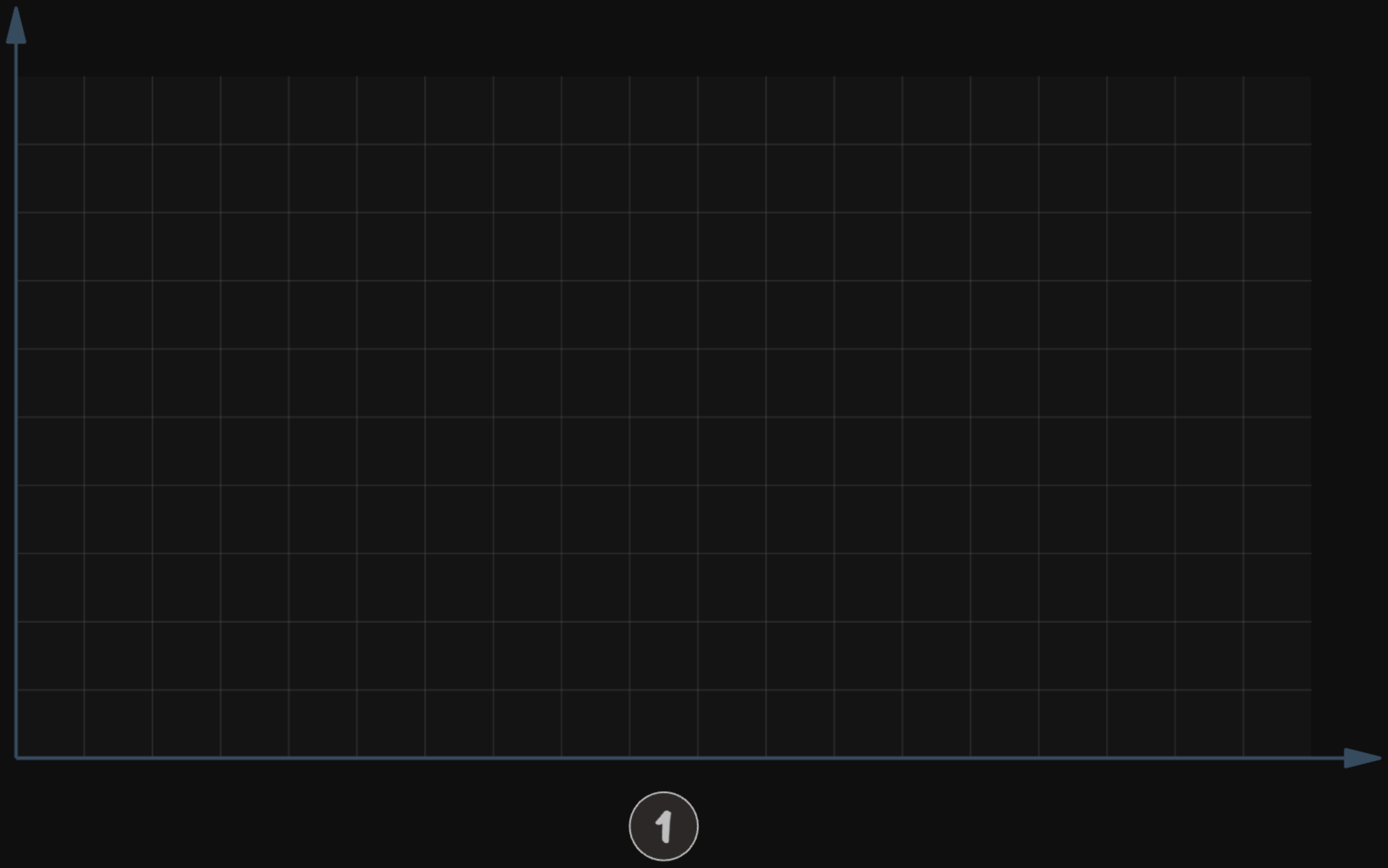
Building our own service

- Use Graphite as a source
- Detect “slow burning” events and short outages
- Exclude past anomalies from the data samples
- Flexible tuning in Grafana
- Configure holidays, events and time of a day corrections

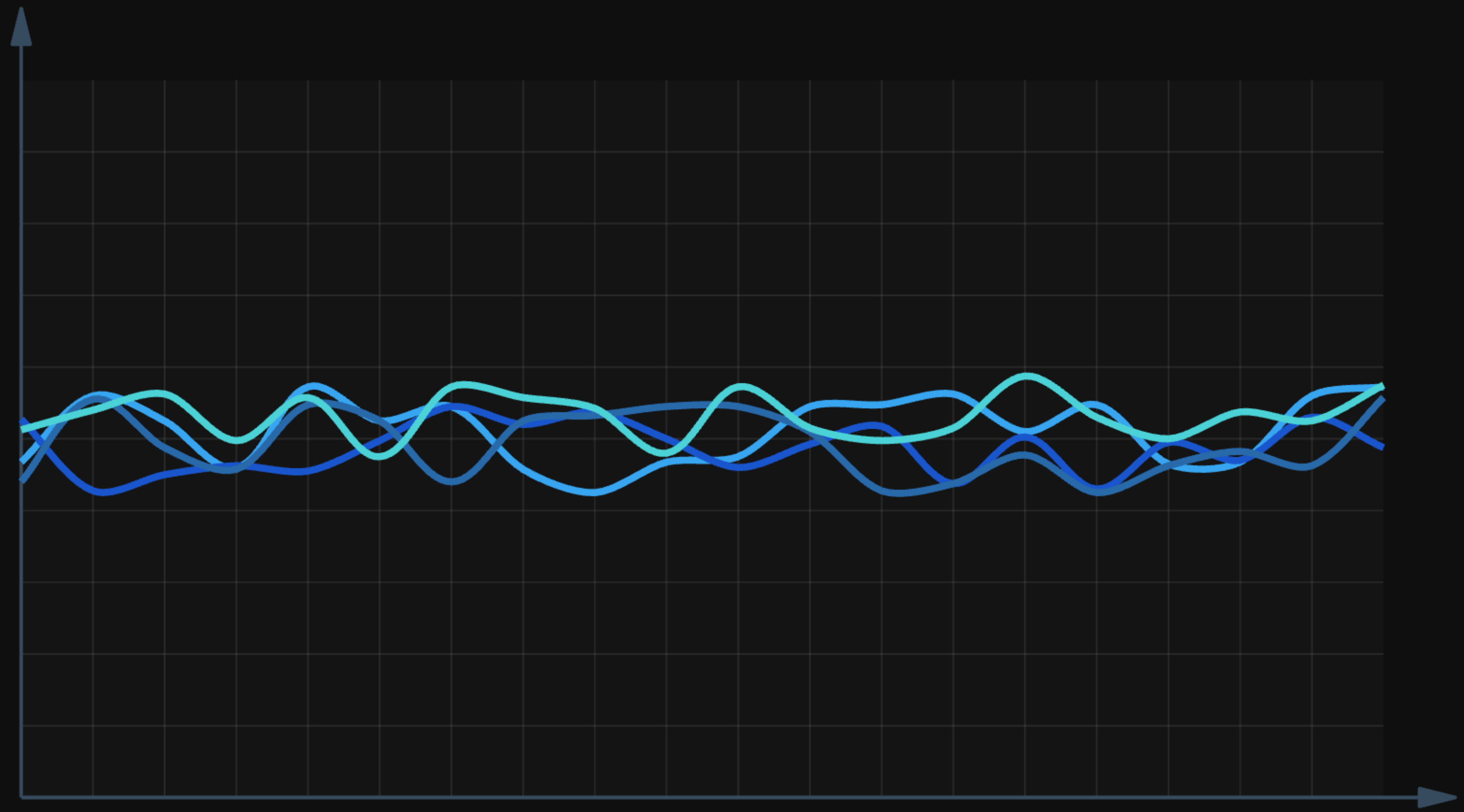
Granomaly


Graphite
Reads and
writes metrics
Granomaly
Grafana
Deploy
Terraform
Build
Dashboards as code

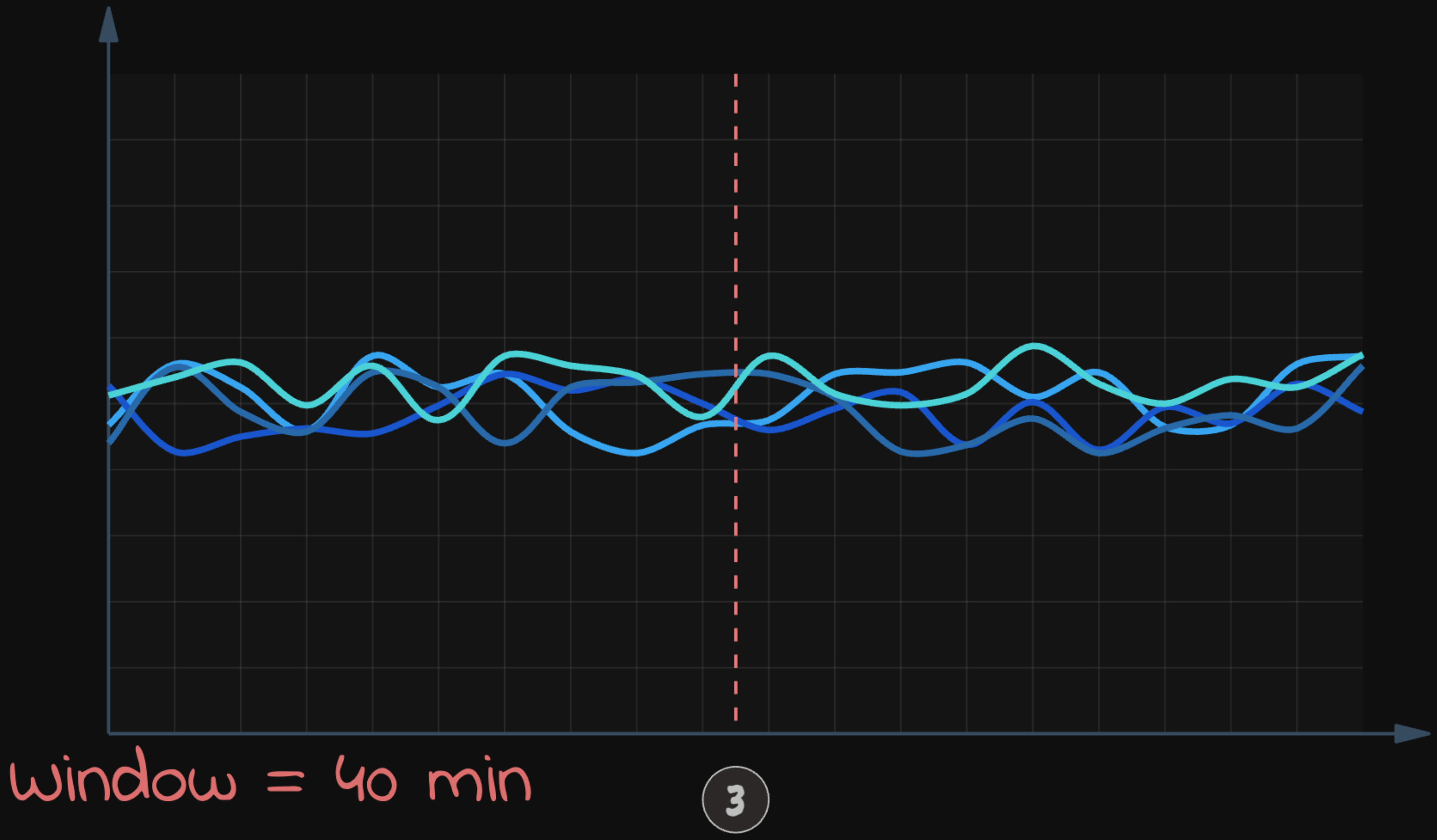
How Granomaly works



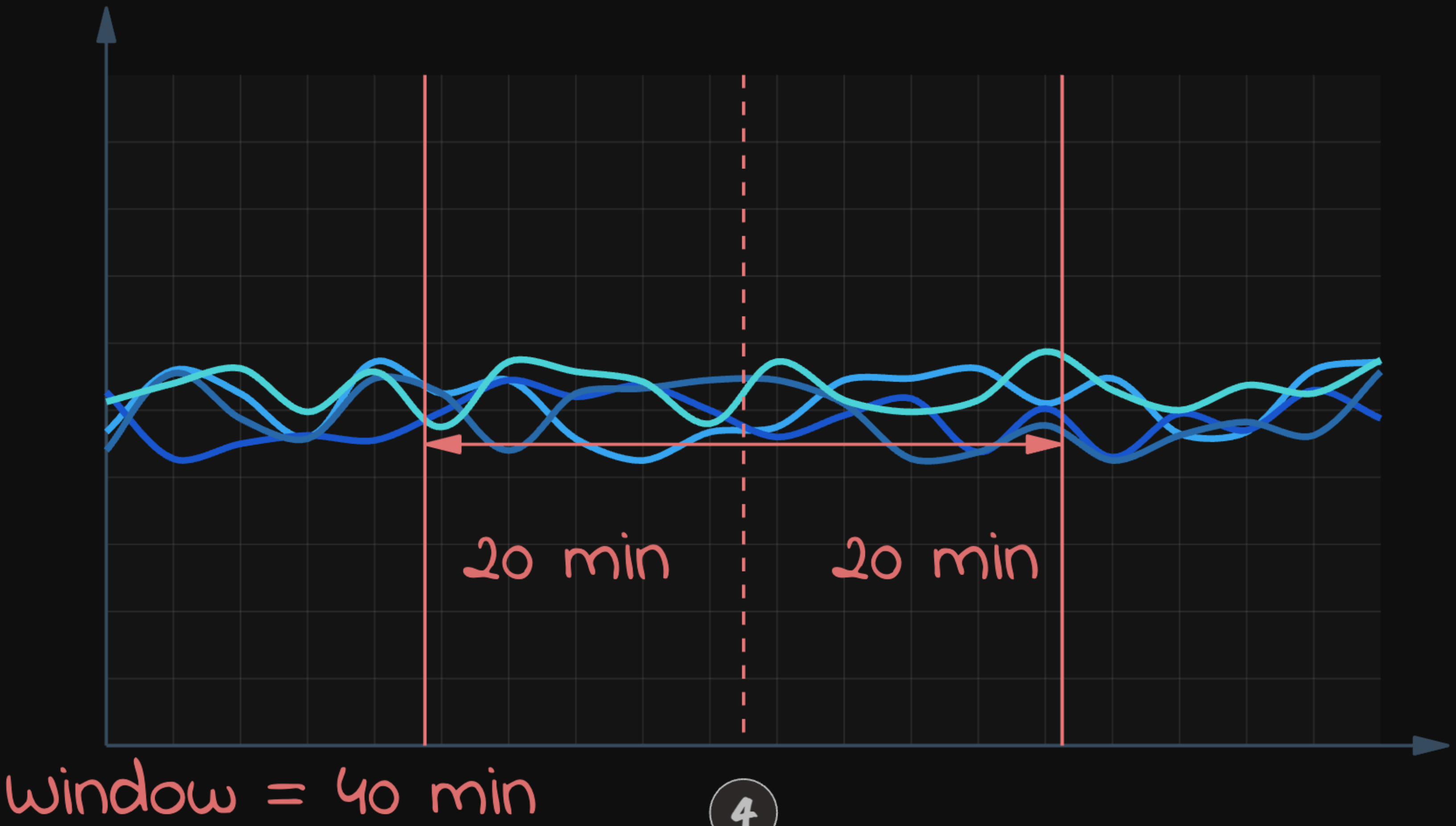
How Granomaly works



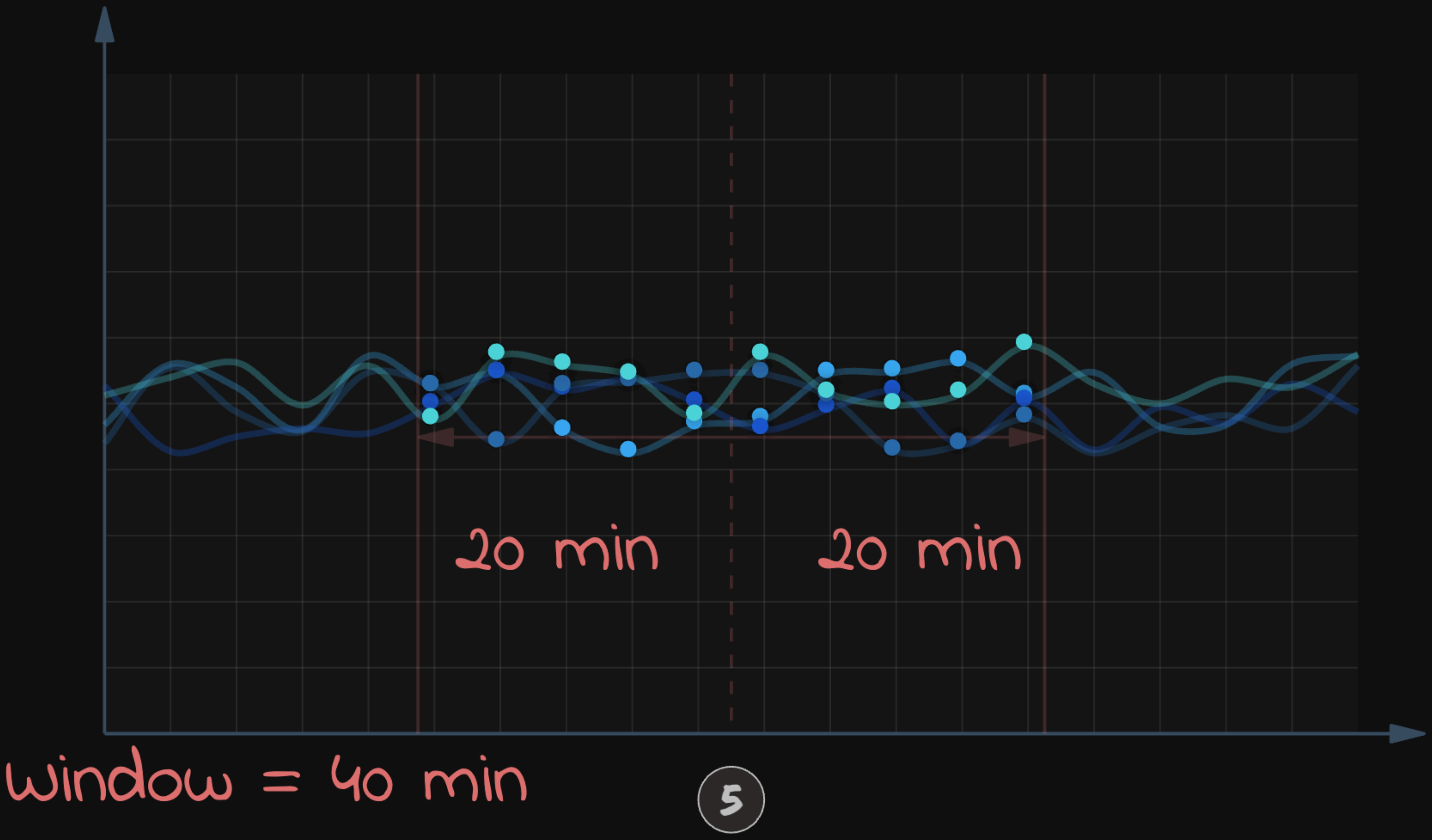
How Granomaly works



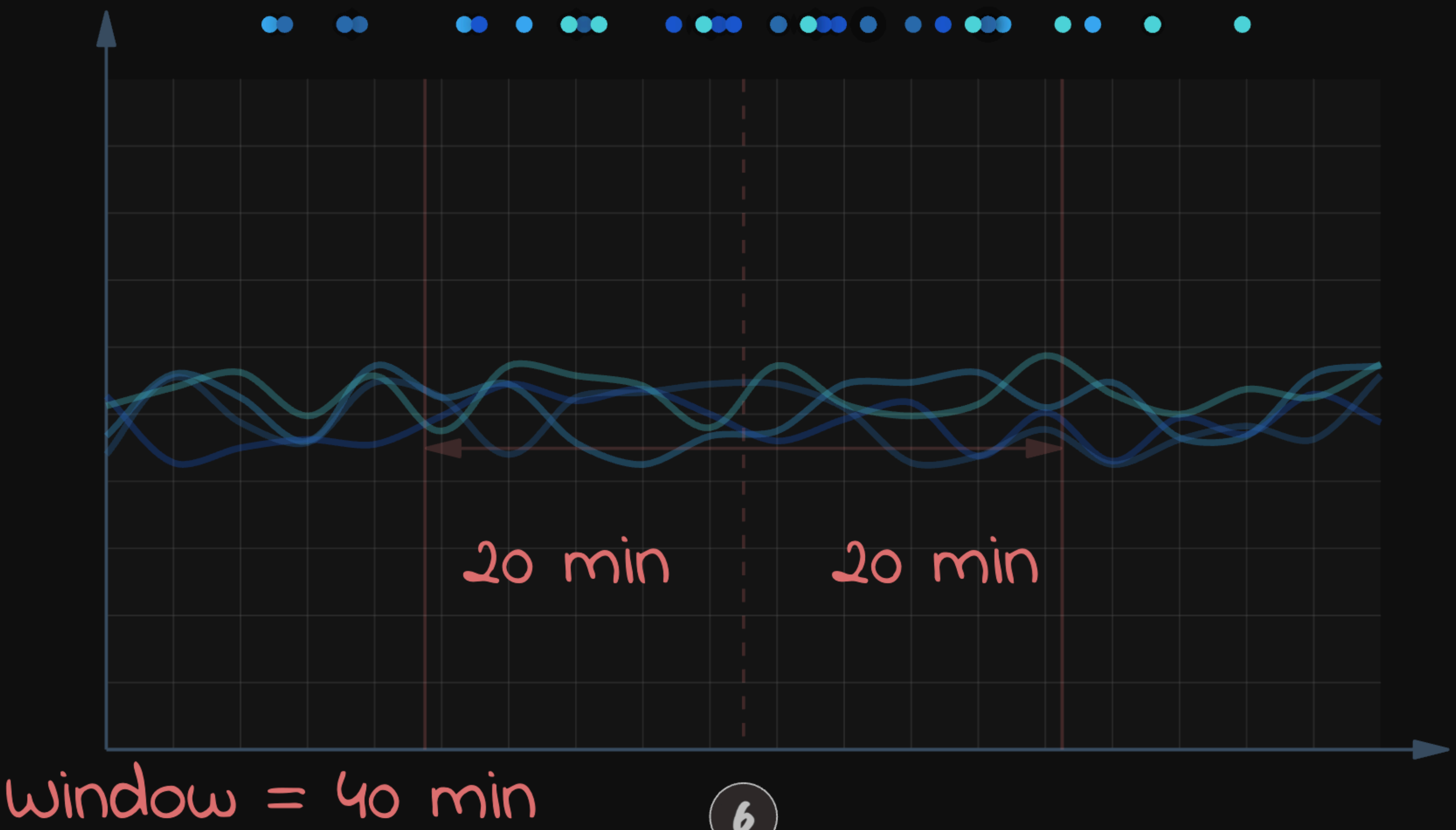
How Granomaly works



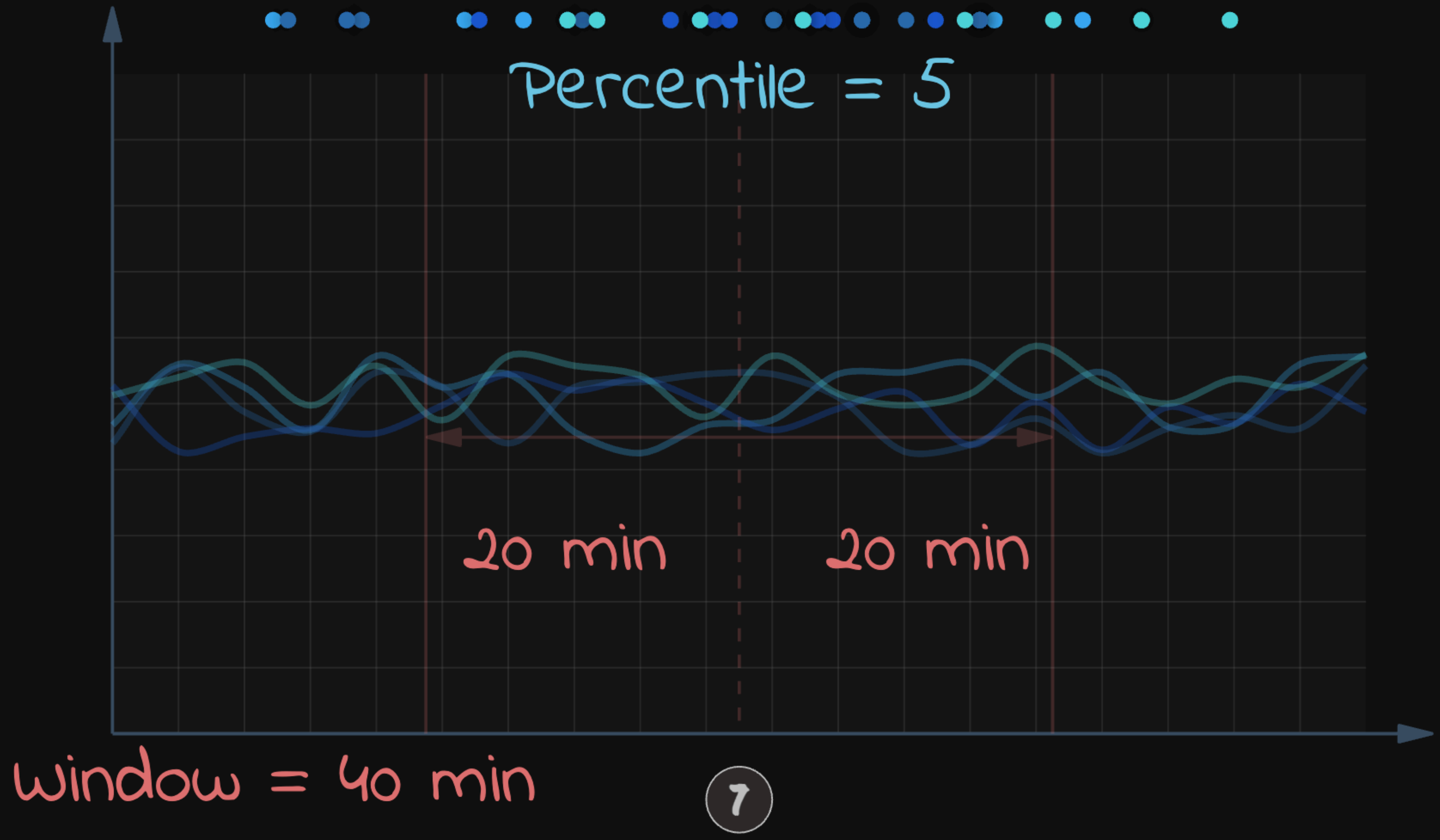
How Granomaly works



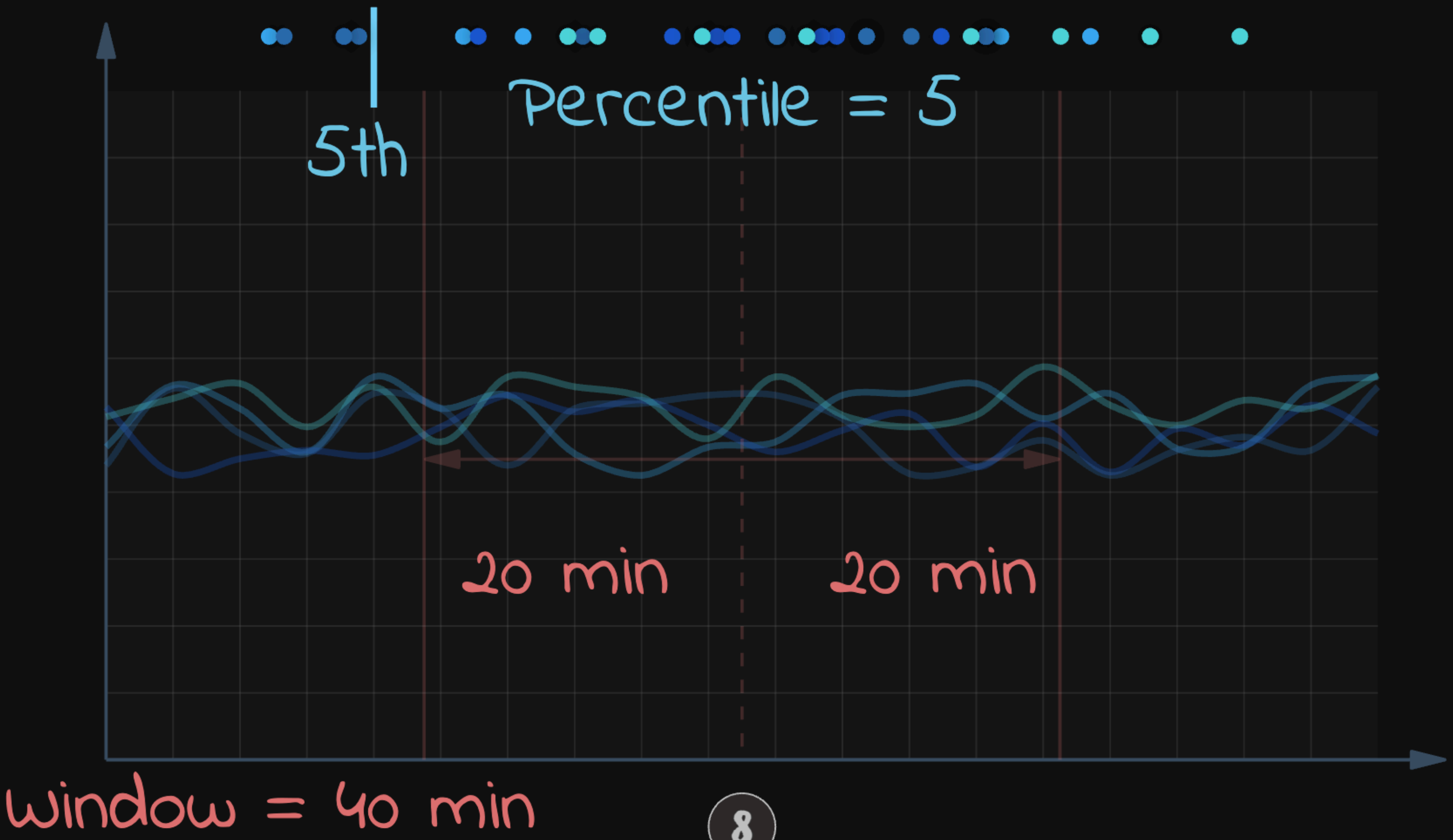
How Granomaly works

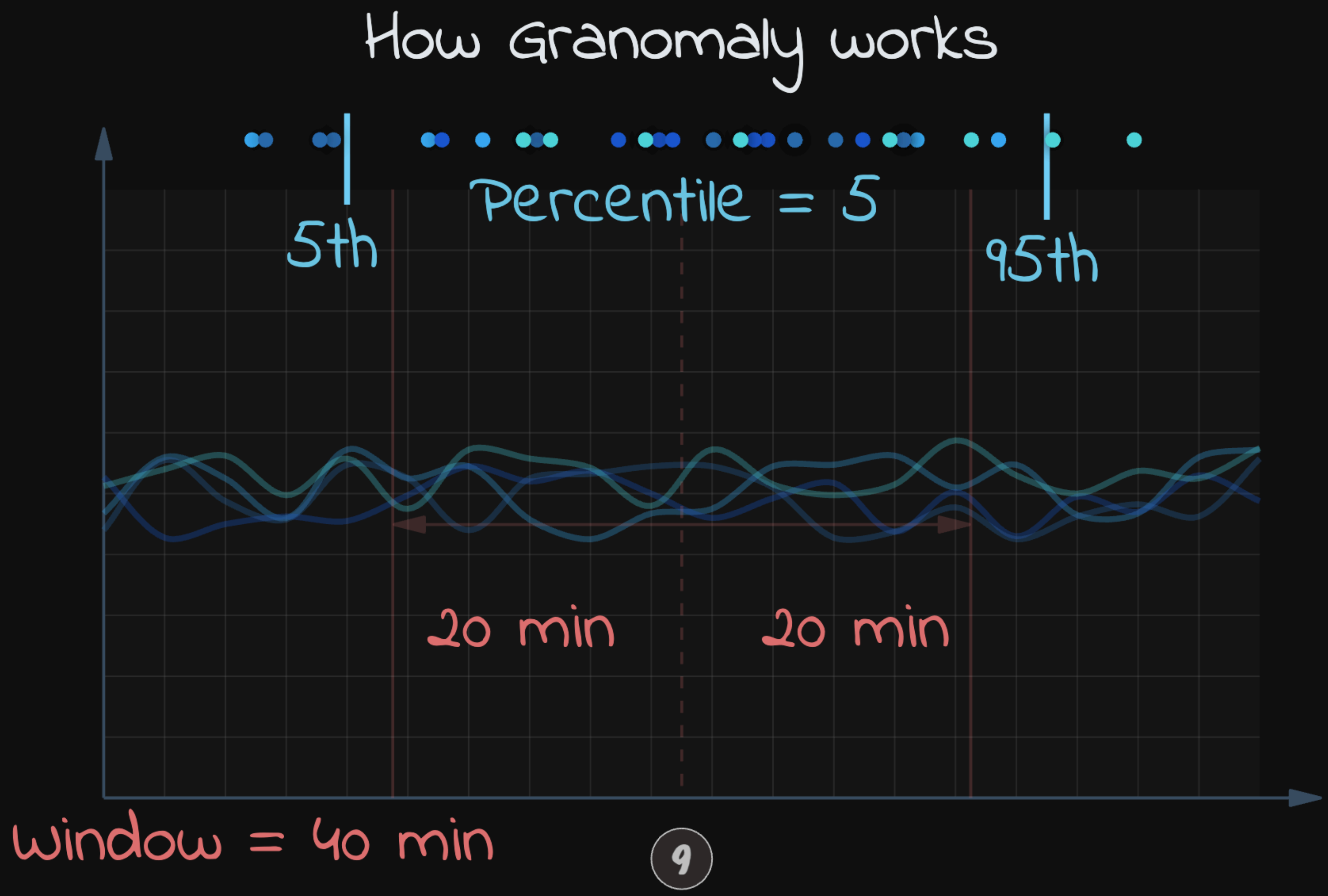


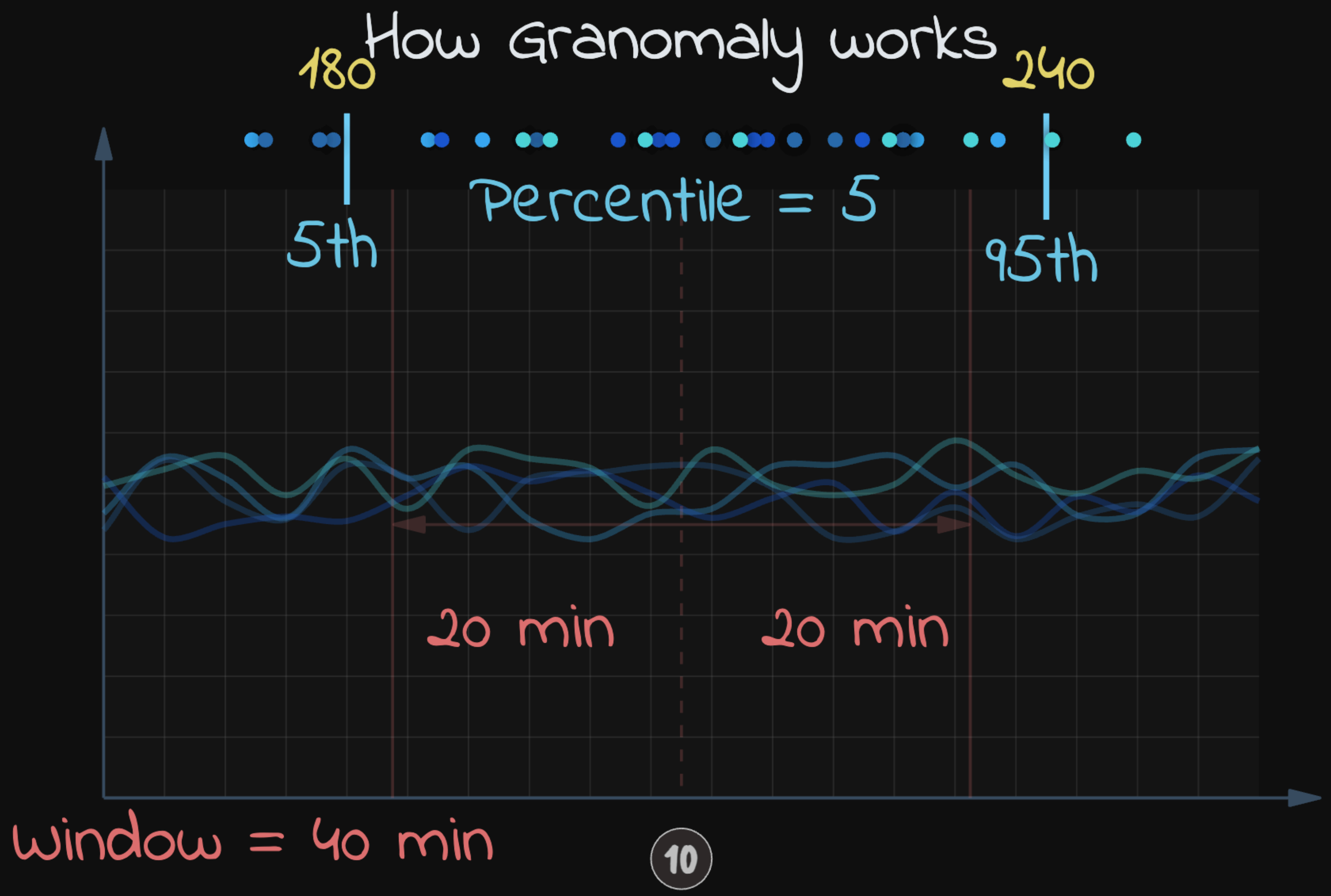
How Granomaly works

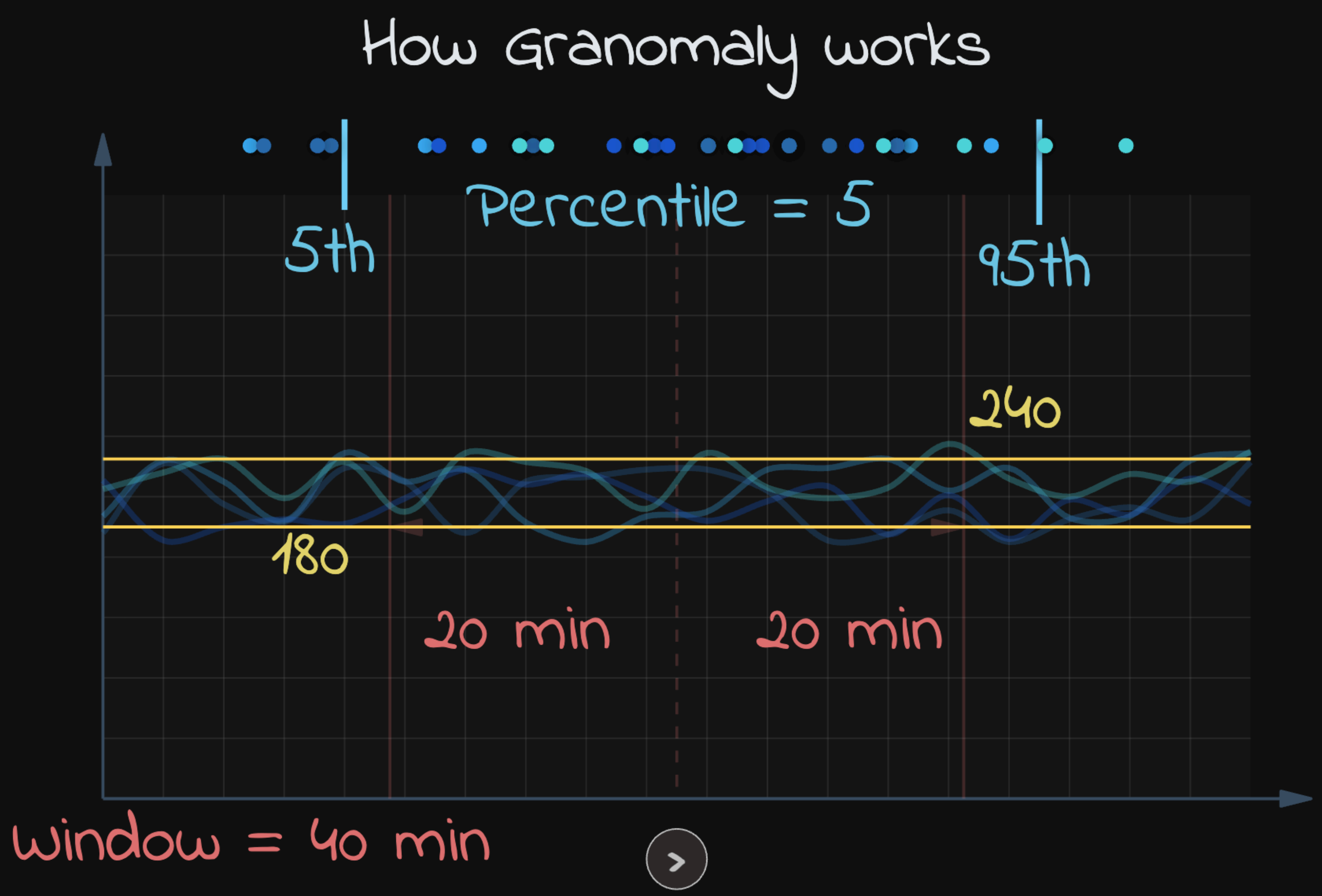


How Granomaly works

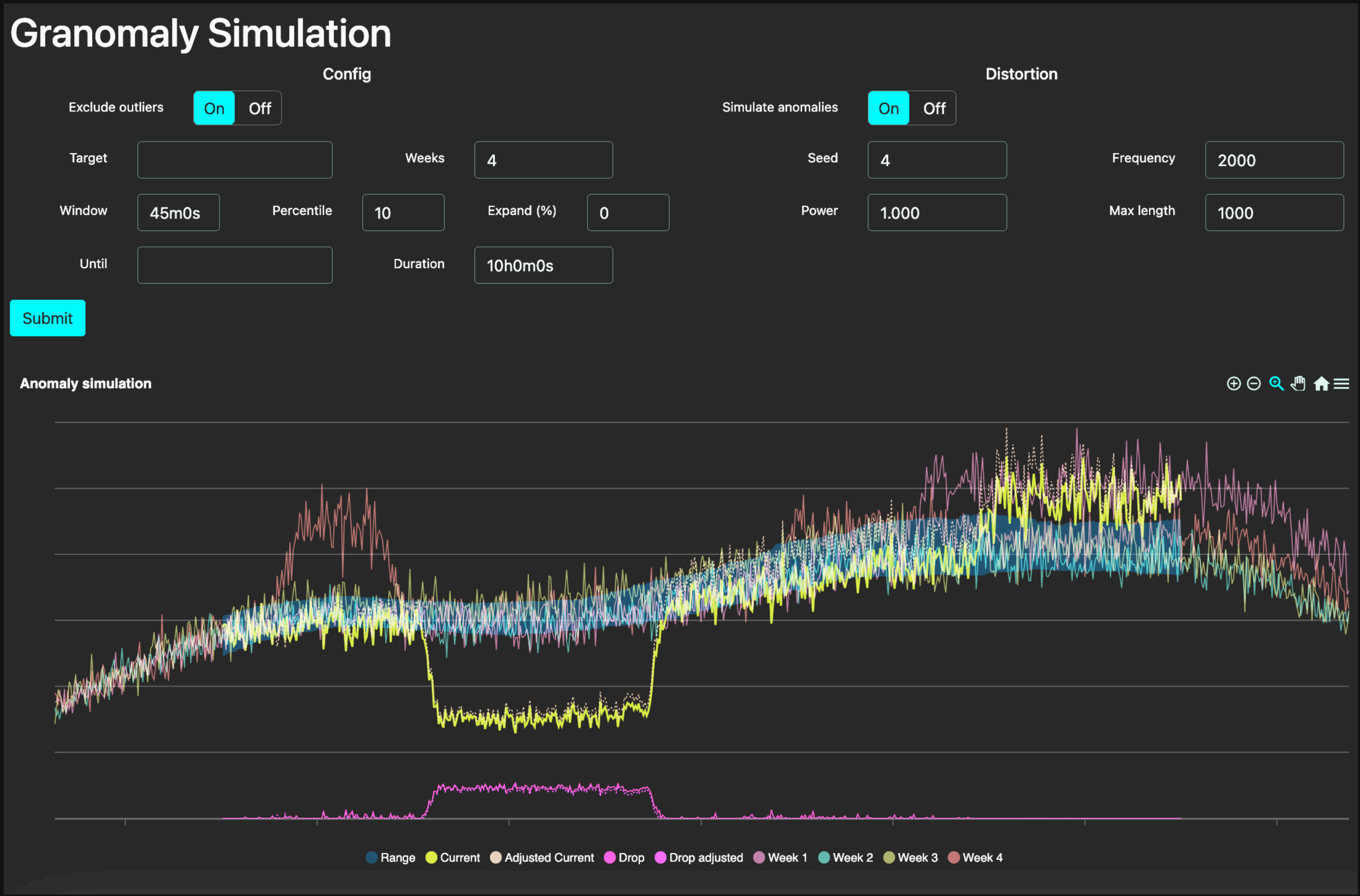



How Granomaly works
Percentile = 5
5th
95th
20 min
20 min
window = 40 min
9


How Granomaly works
180
240
5th
Percentile = 5
95th
20 min
20 min
window = 40 min
10


How Granomaly works
5th
Percentile = 5
95th
240
180
20 min
20 min
window = 40 min

Boundaries for last 24 hours

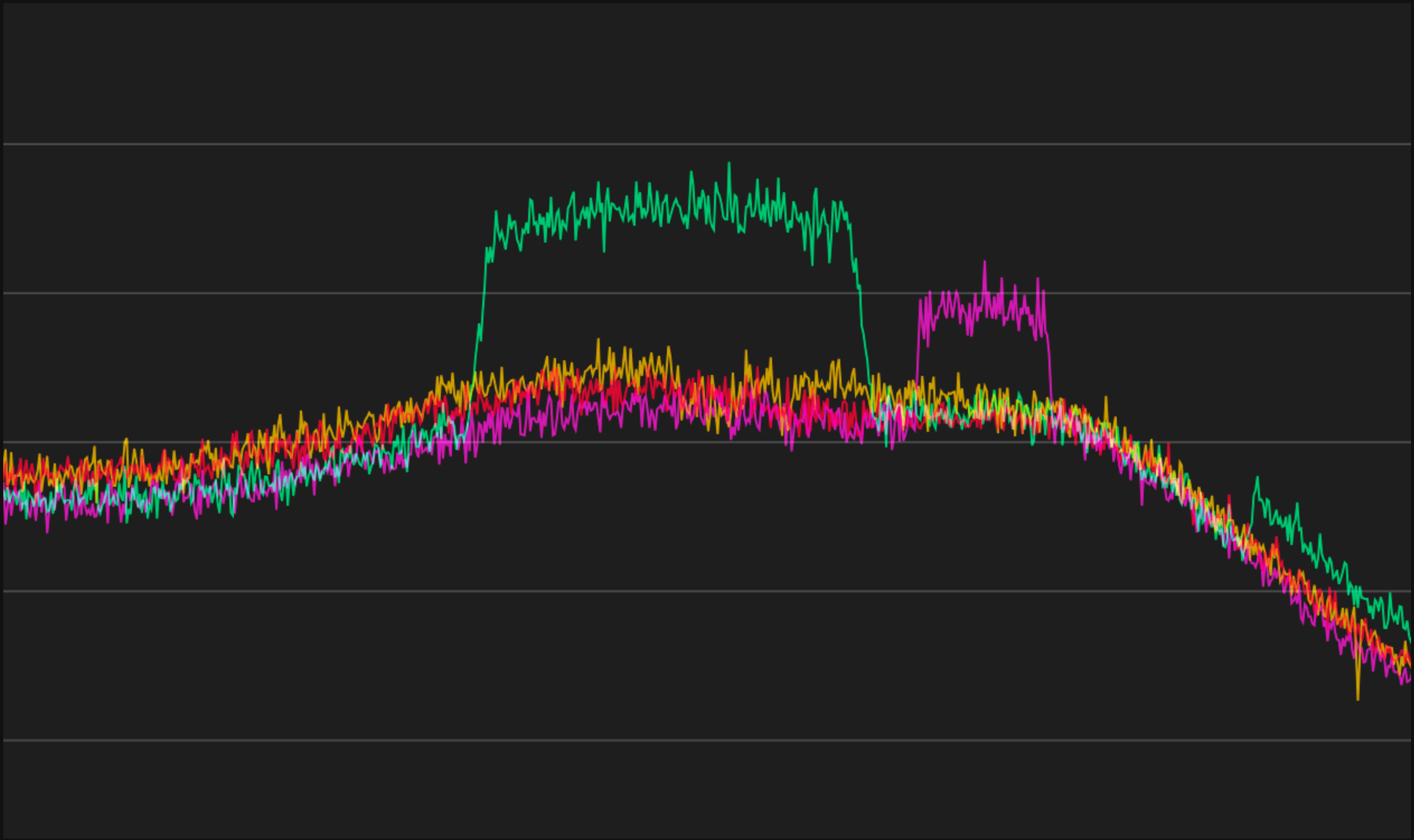


Granomaly Simulation
Config
Distortion
Exclude outliers
On
Off
Simulate anomalies
On
Off
Target
Weeks
4
Seed
4
Frequency
2000
Window
45m0s
Percentile
10
Expand (%)
0
Power
1.000
Max length
1000
Until
Duration
10h0m0s
Submit
Anomaly simulation
Range
Current
Adjusted Current
Drop
Drop adjusted
Week 1
Week 2
Week 3
Week 4

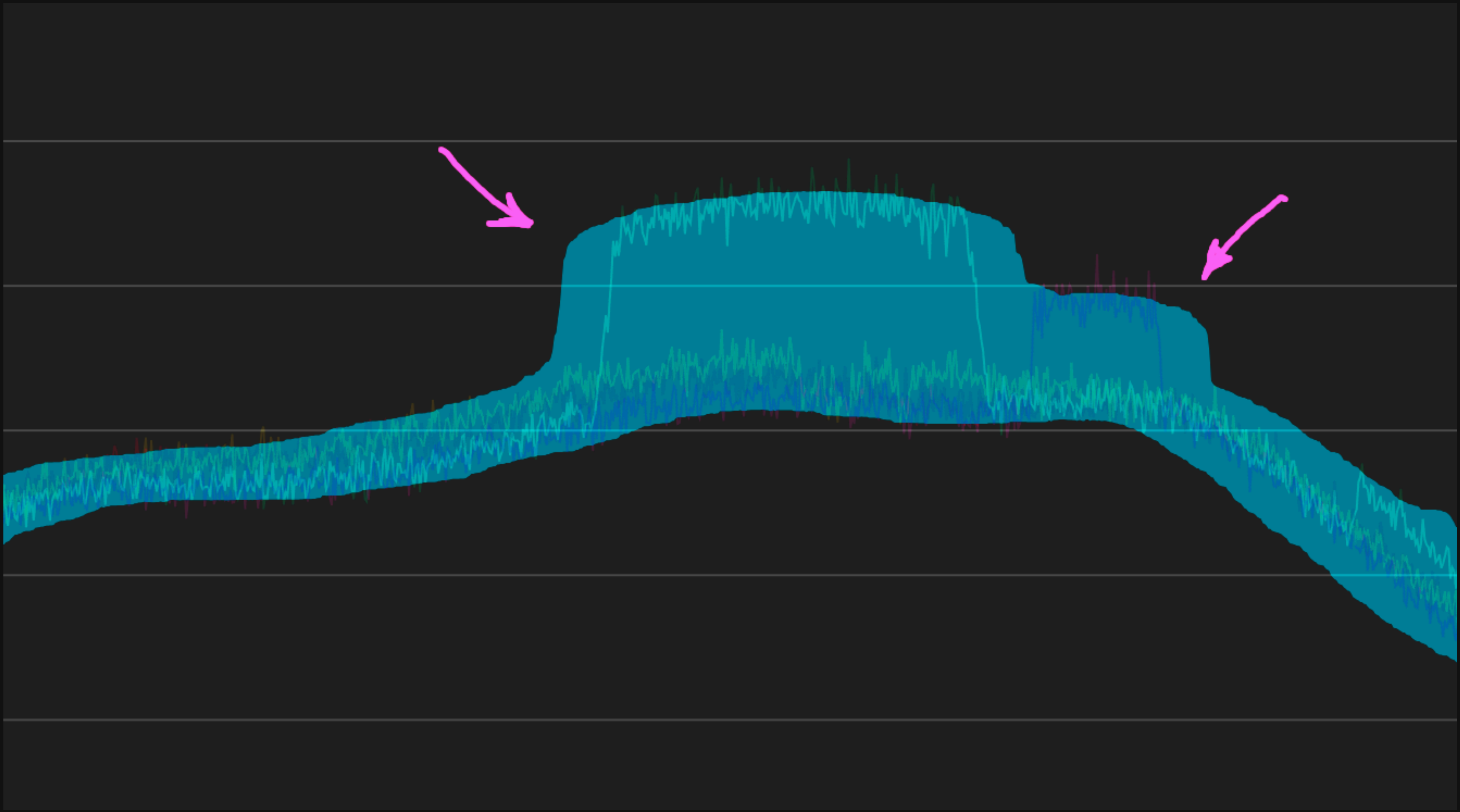
Running into some problems on the way

- Past incidents
- Daylight Savings Time
- Incidents while overperforming
- Correcting for known events
- Query complexity

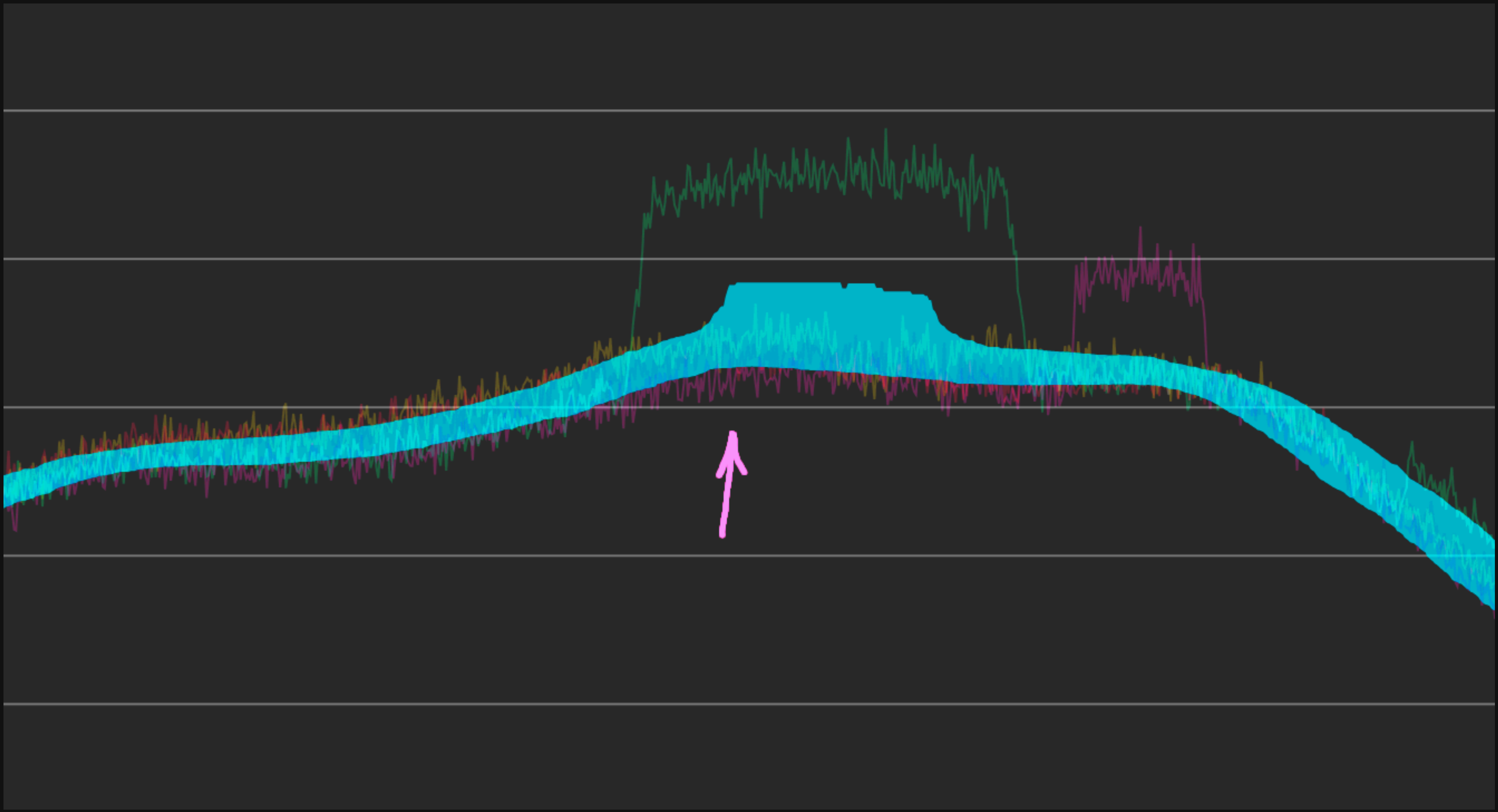
Problem #1: Past incidents



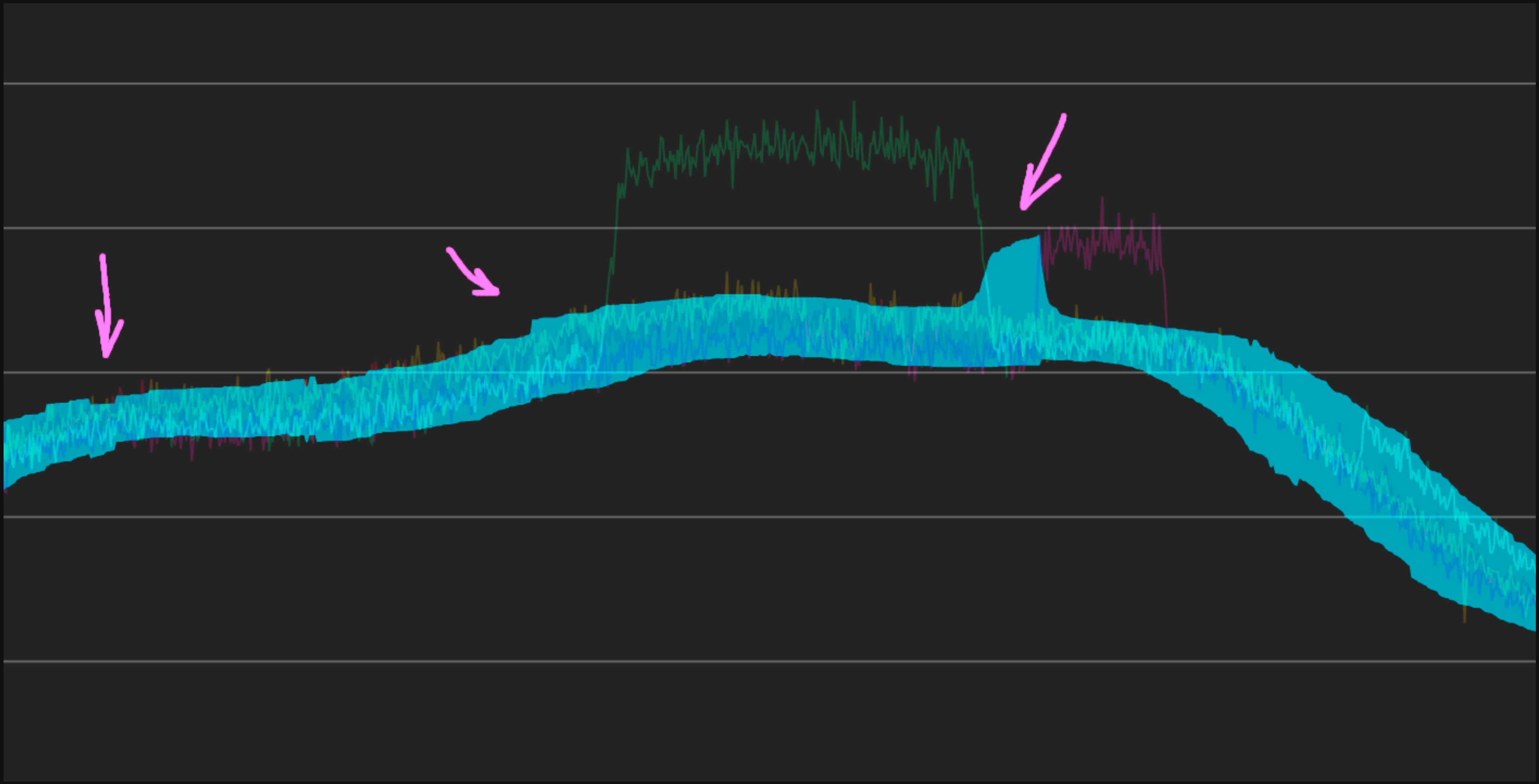
Using 5th percentile



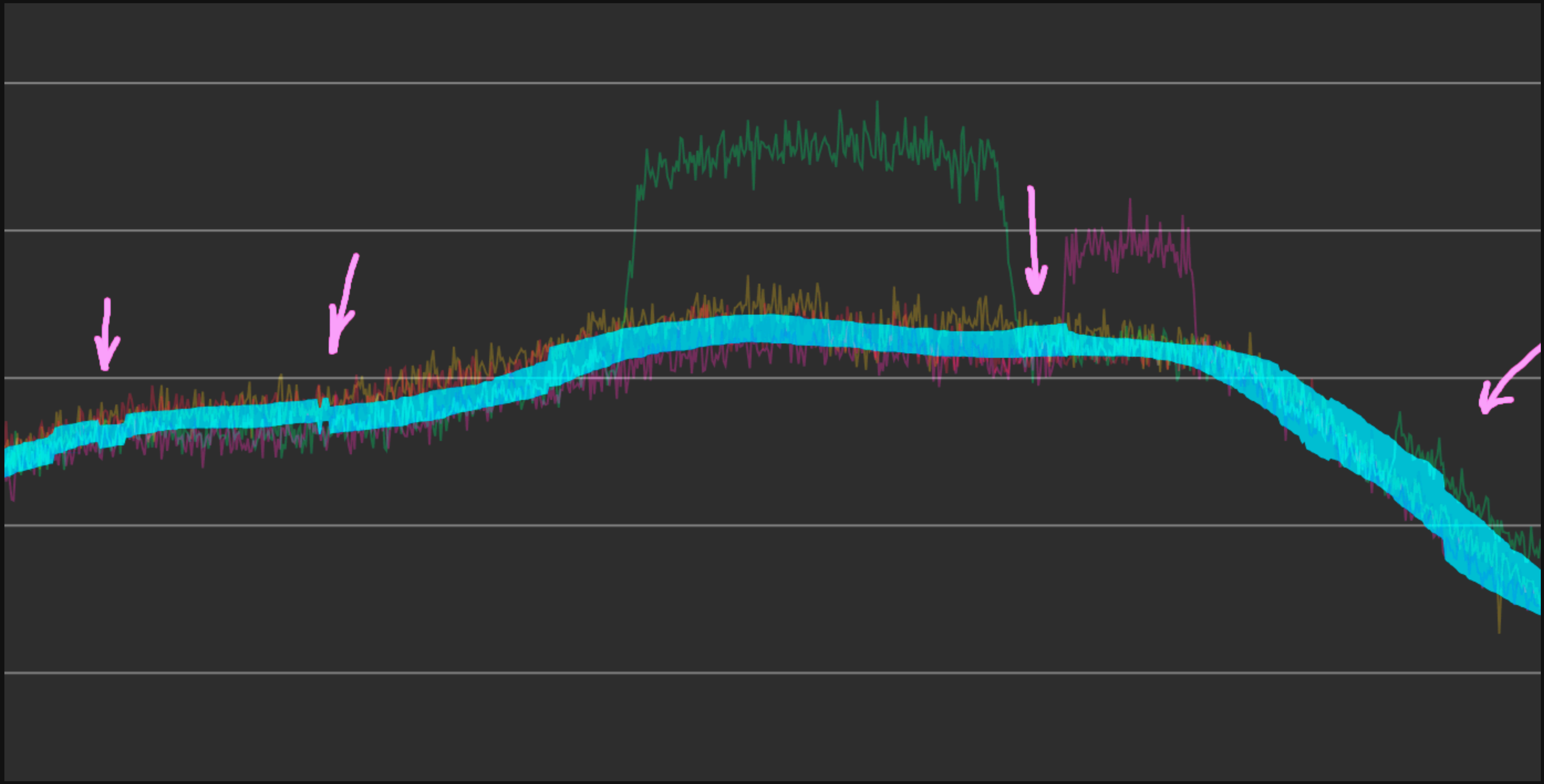
25th percentile

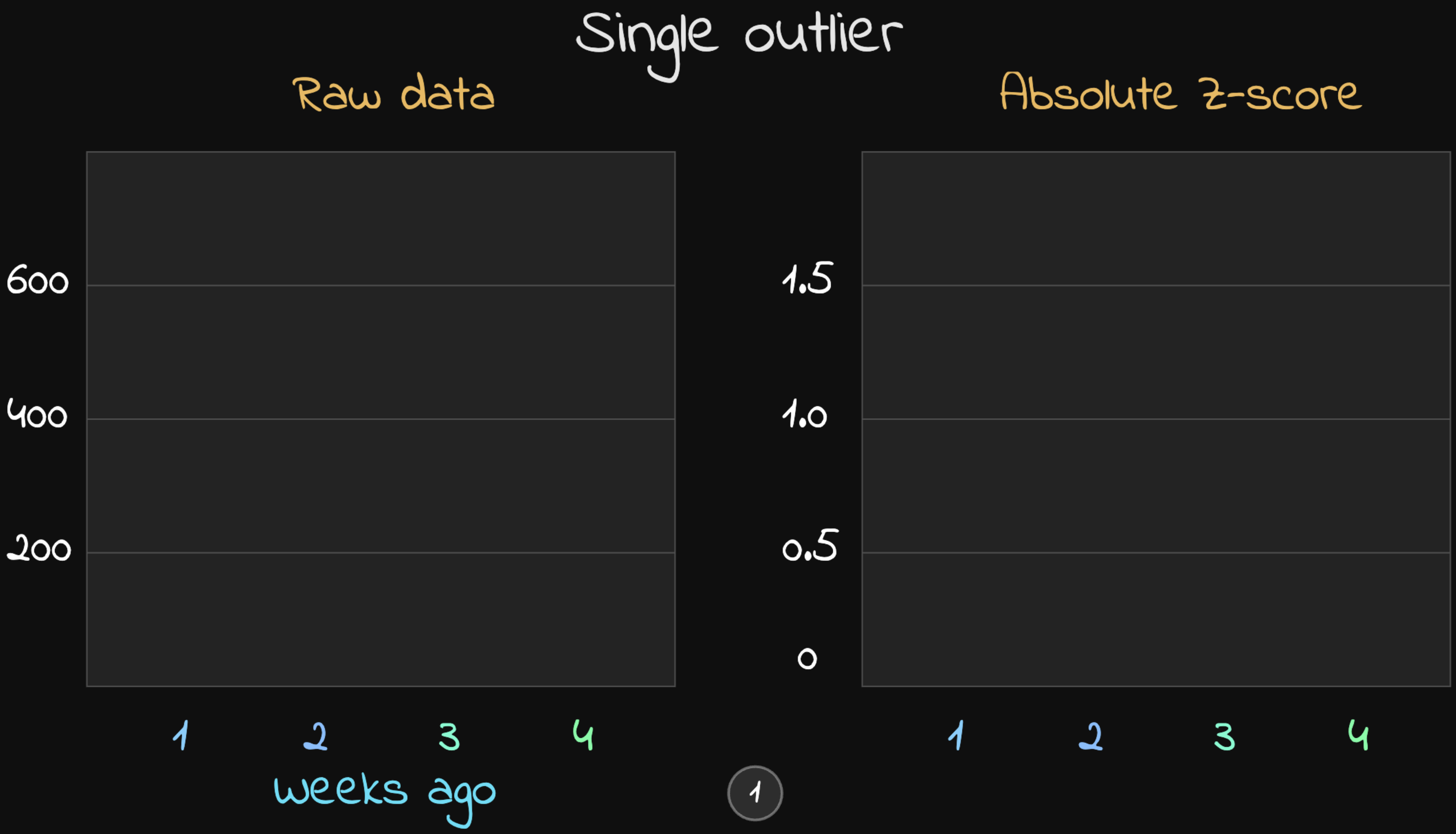


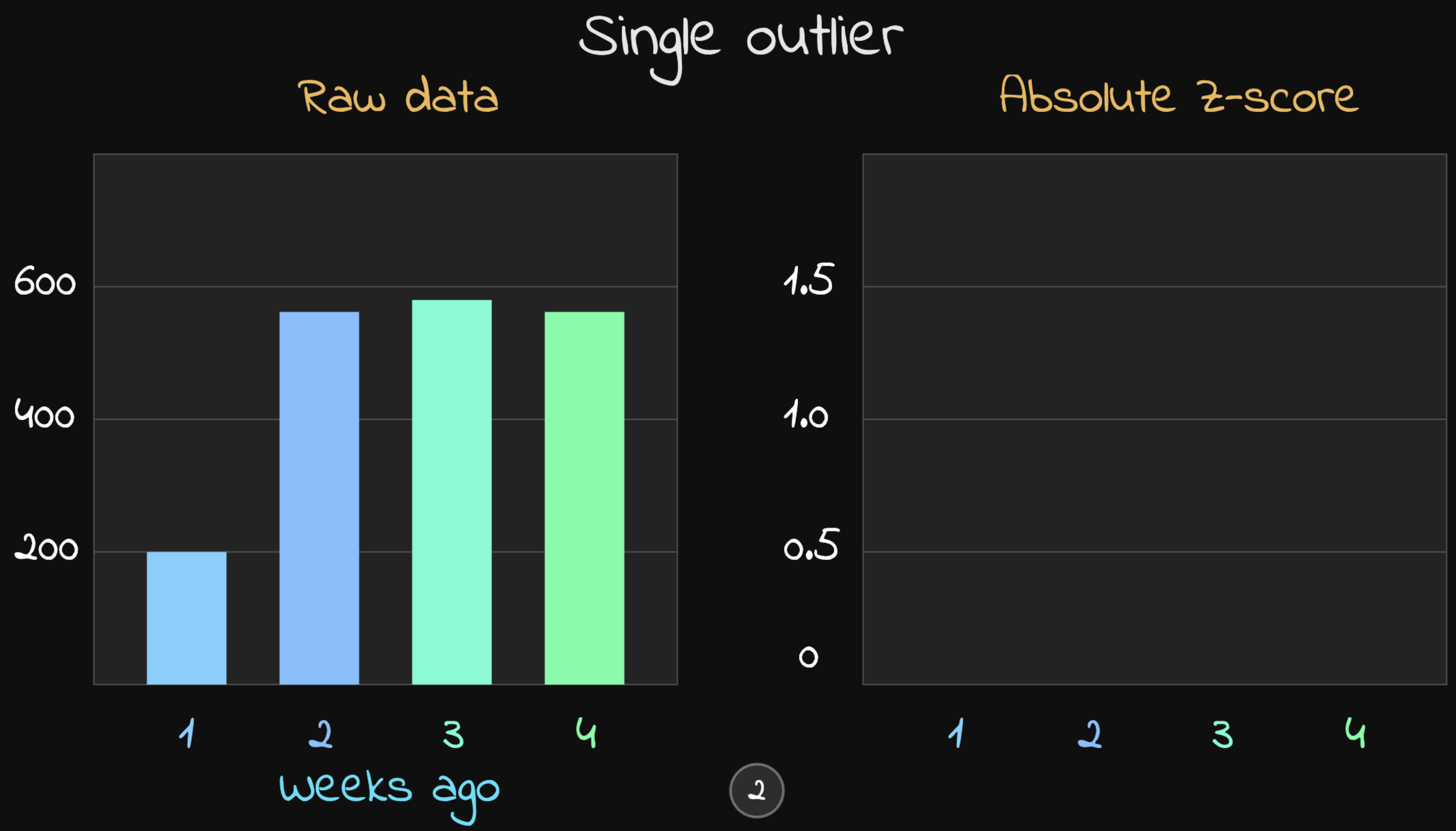
Least stddev based exclusion (5th percentile)



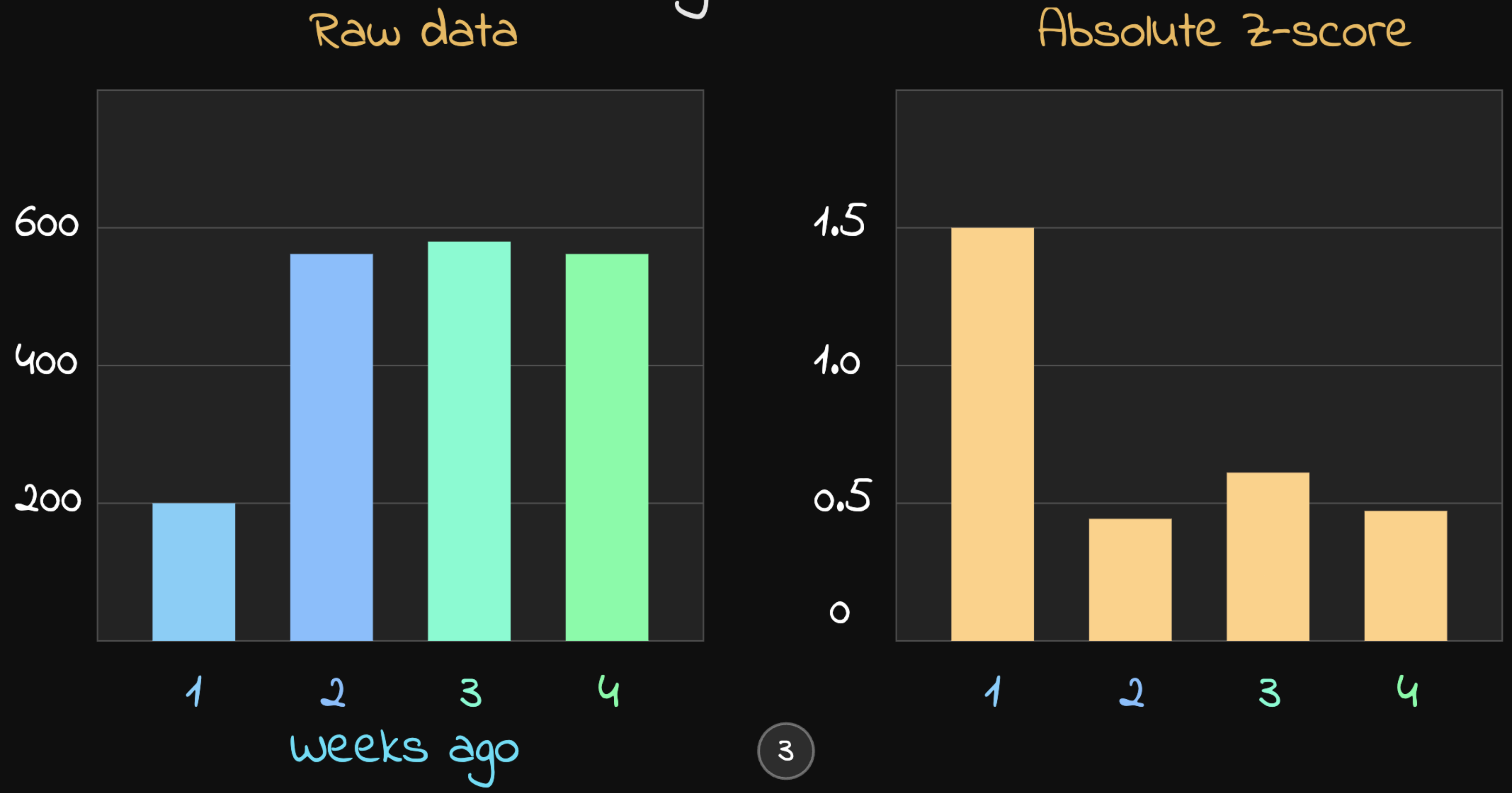
Least stddev based exclusion (25th percentile)

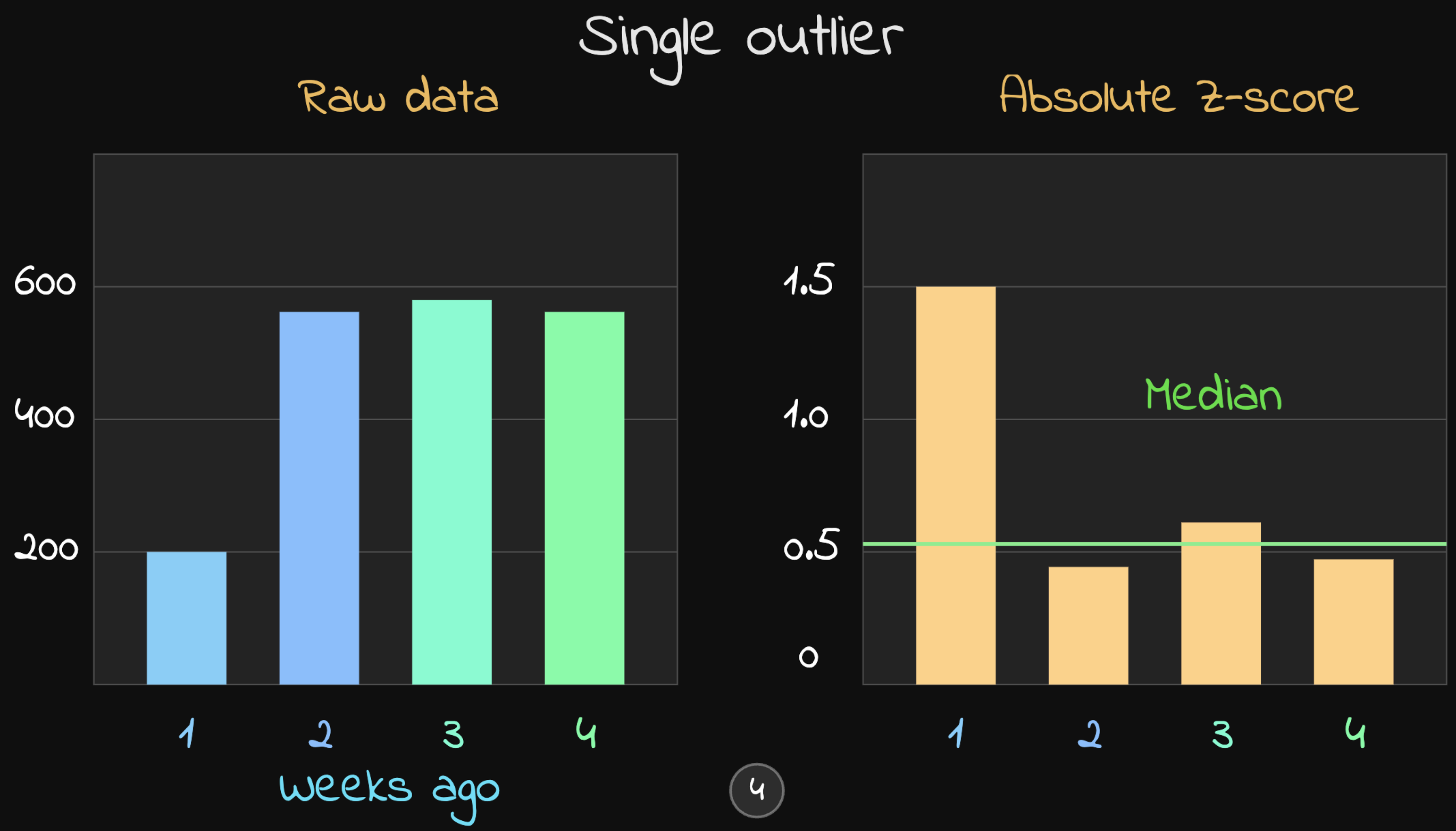



Single outlier
Raw data
Absolute Z-score
600
400
200
1.5
1.0
0.5
0
1
2
3
4
1
2
3
4
weeks ago
1


Single outlier
Raw data
Absolute Z-score
600
400
200
1.5
1.0
0.5
0
1
2
3
4
1
2
3
4
weeks ago
2

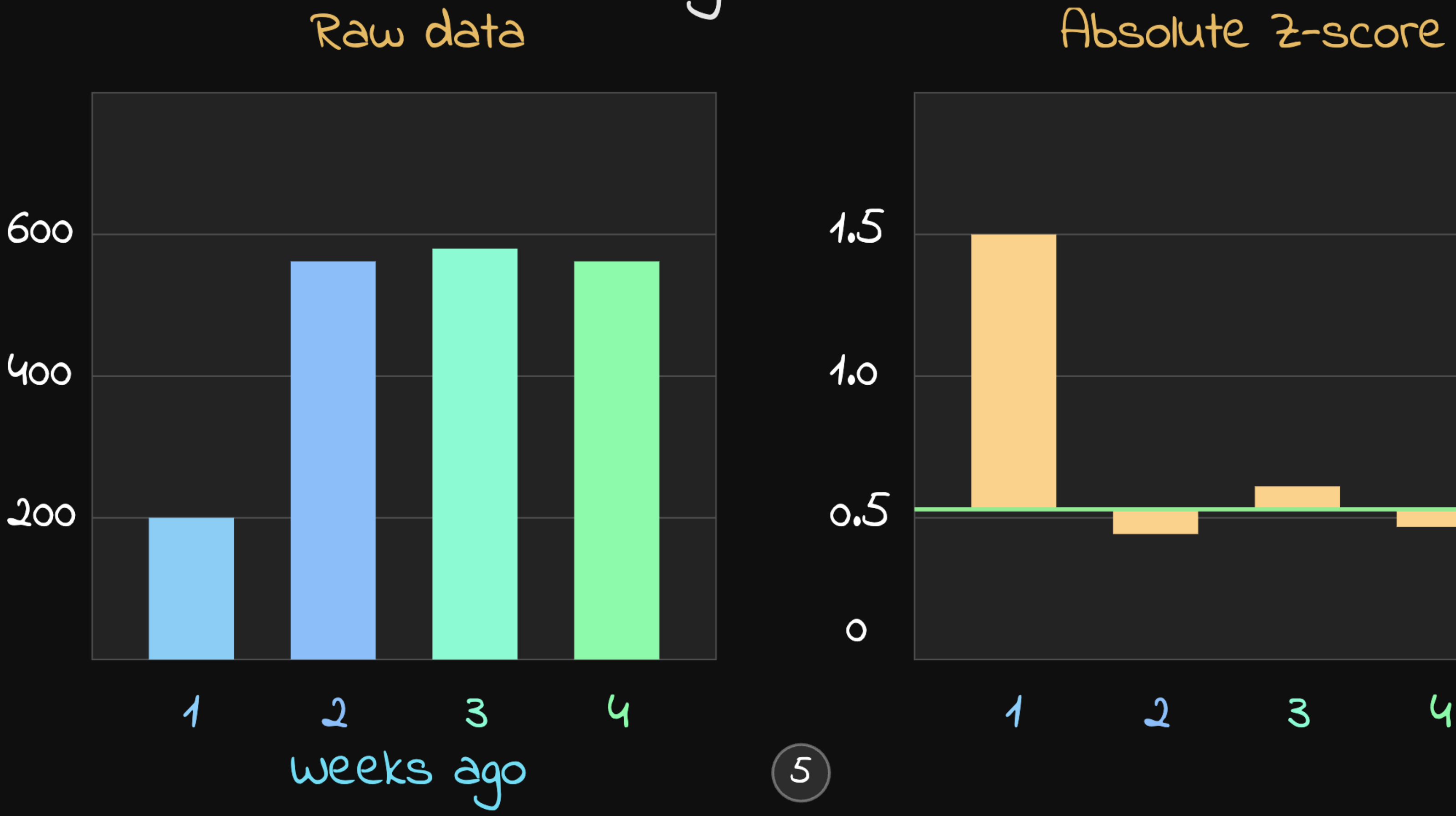
Single outlier

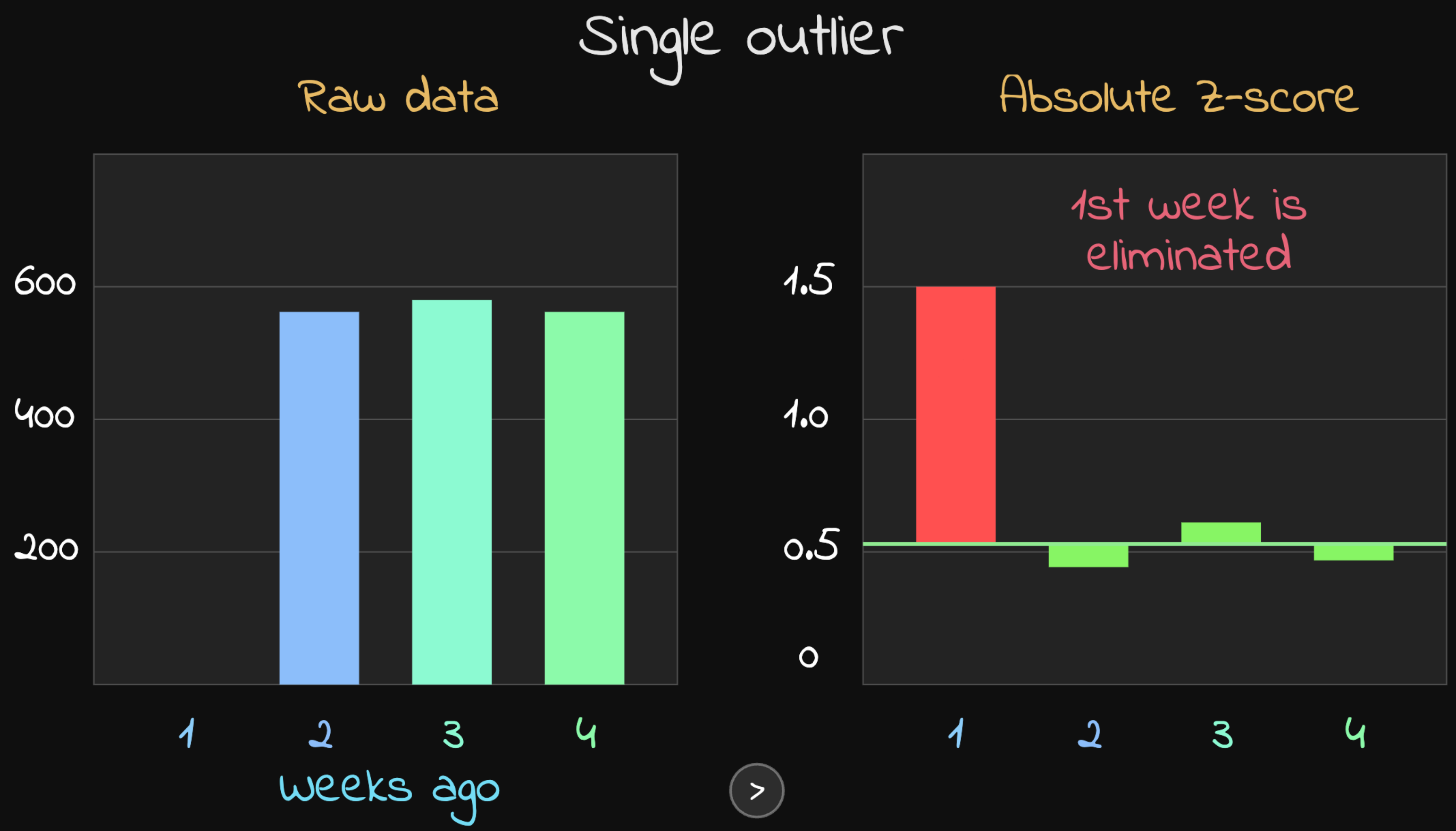




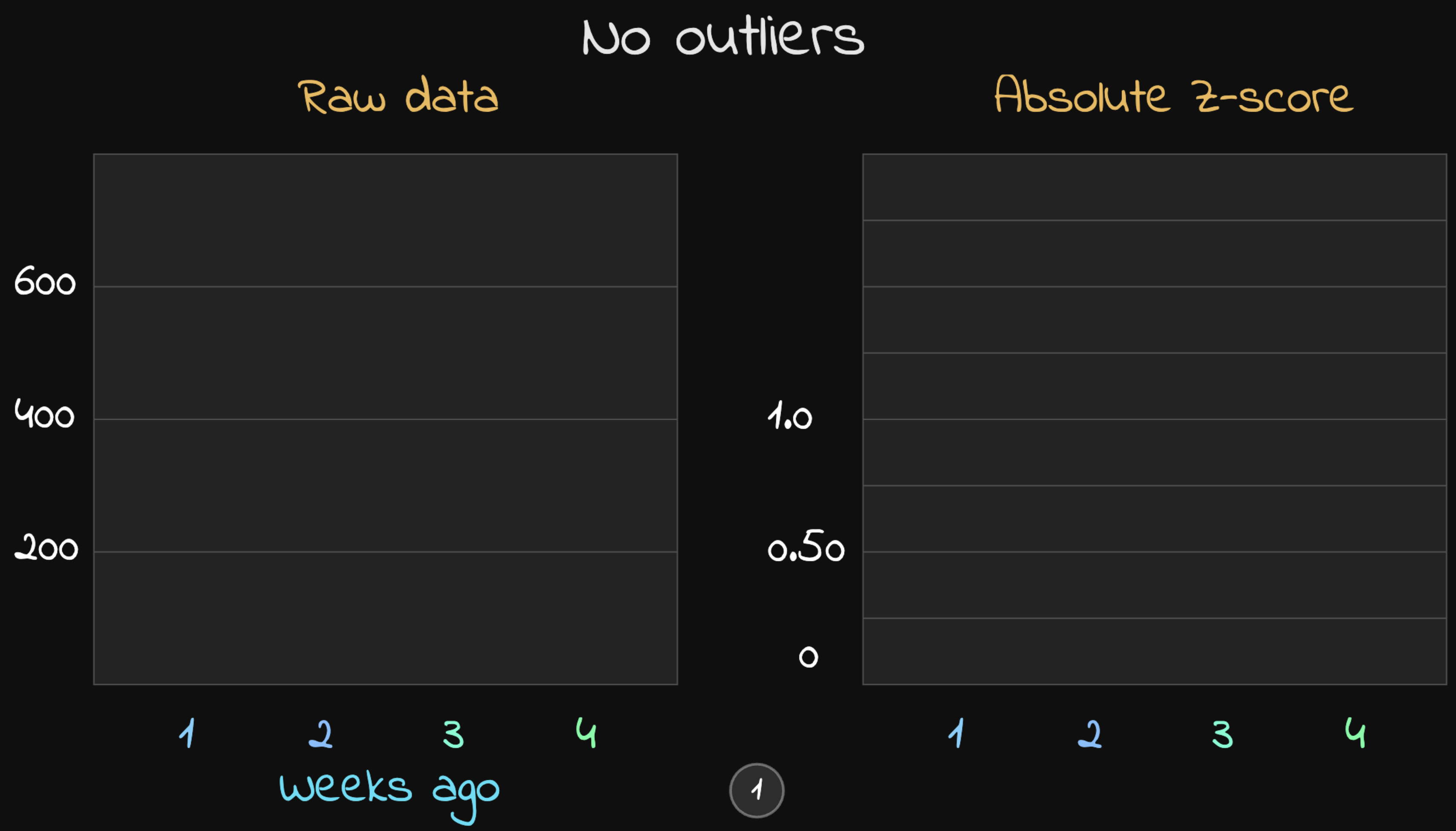
Single outlier
Raw data
Absolute Z-score
600
400
200
1
2
3
4
weeks ago
1.5
1.0
0.5
0
Median
1
2
3
4
4

Single outlier

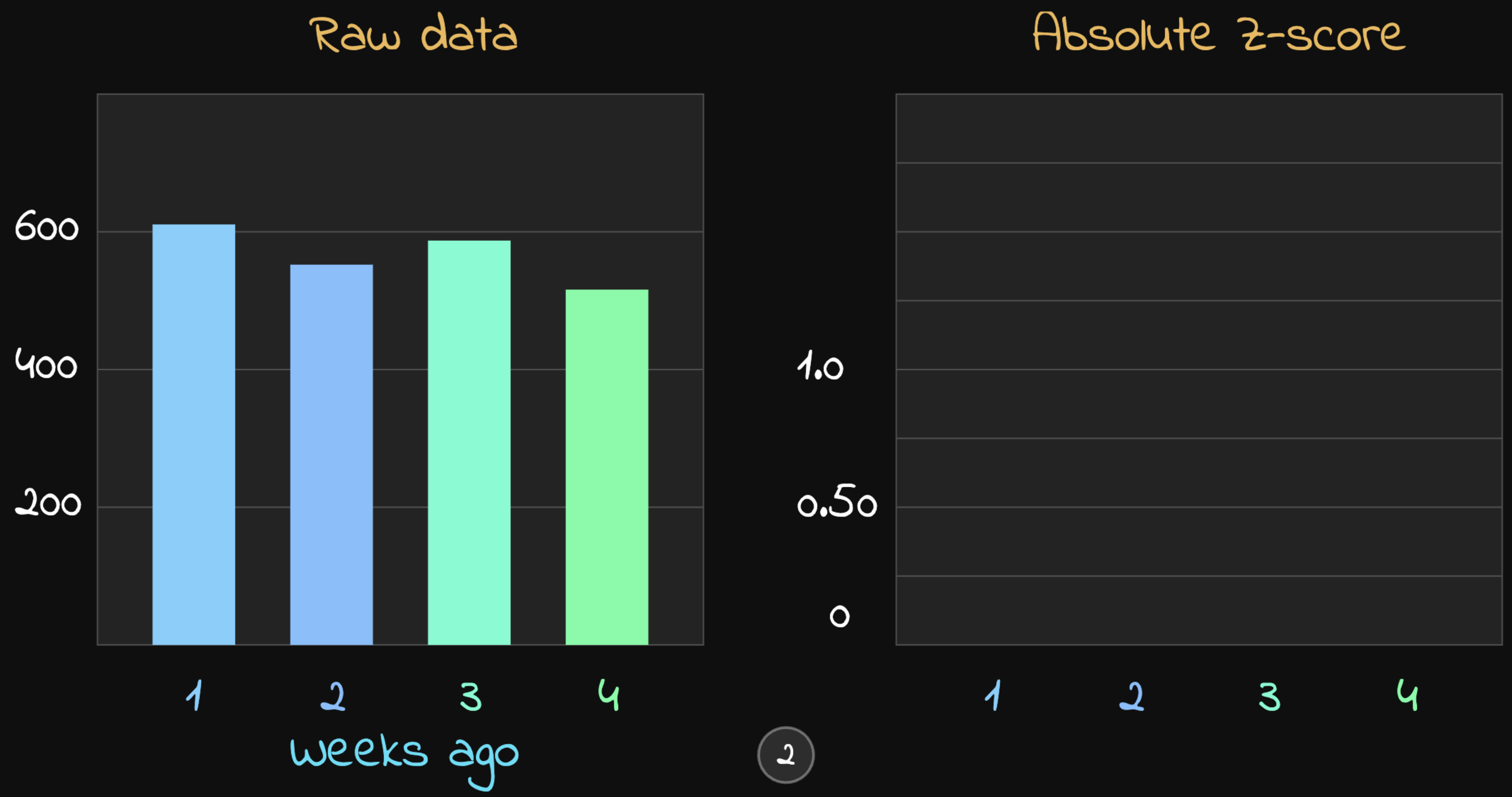




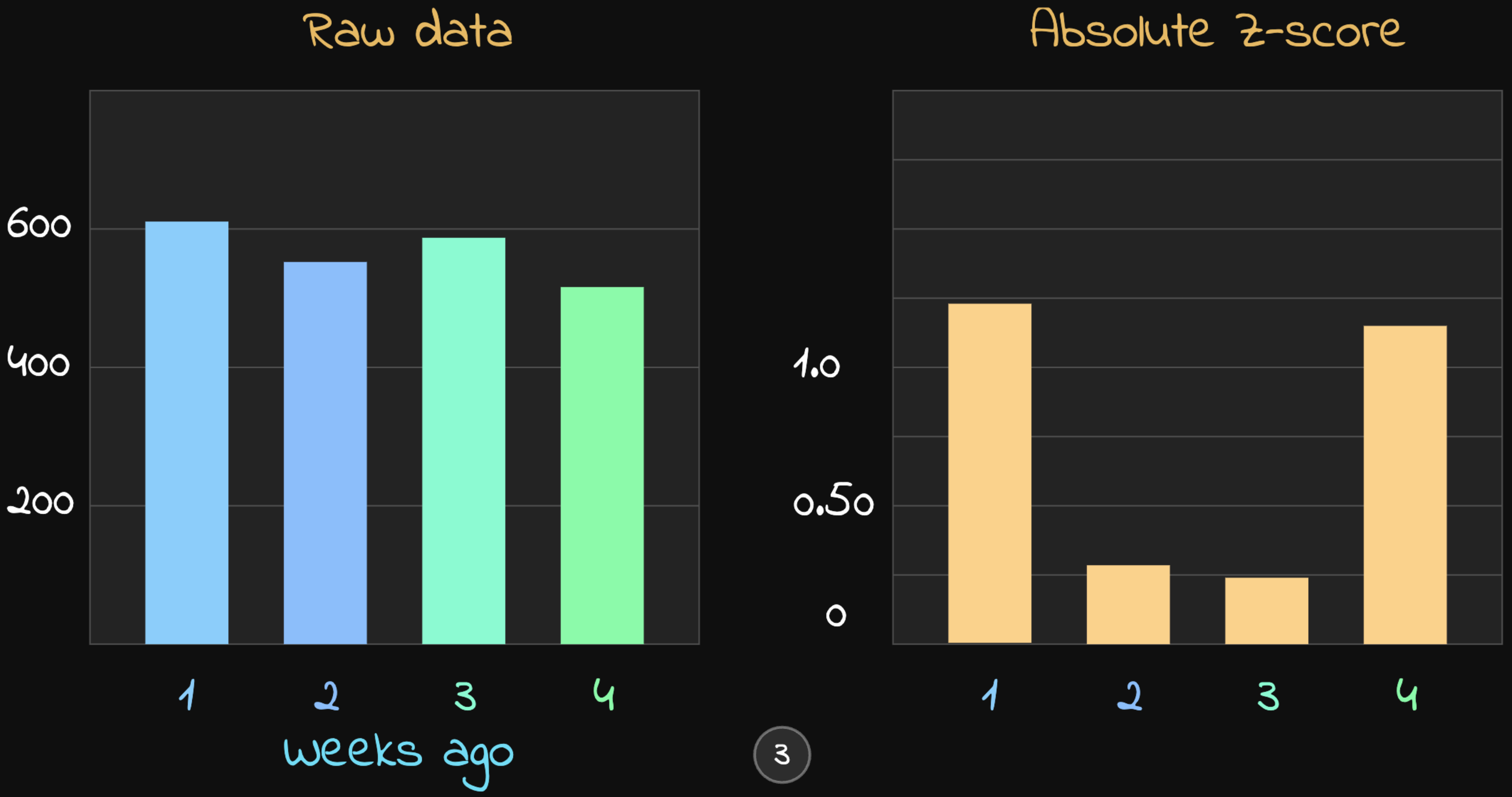
Single outlier
Raw data
Absolute Z-score
600
400
200
1.5
1.0
0.5
0
1st week is eliminated
1
2
3
4
1
2
3
4
weeks ago
>


No outliers
Raw data
Absolute z-score
600
400
200
1.0
0.50
0
1
2
3
4
1
2
3
4
weeks ago
1

No outliers

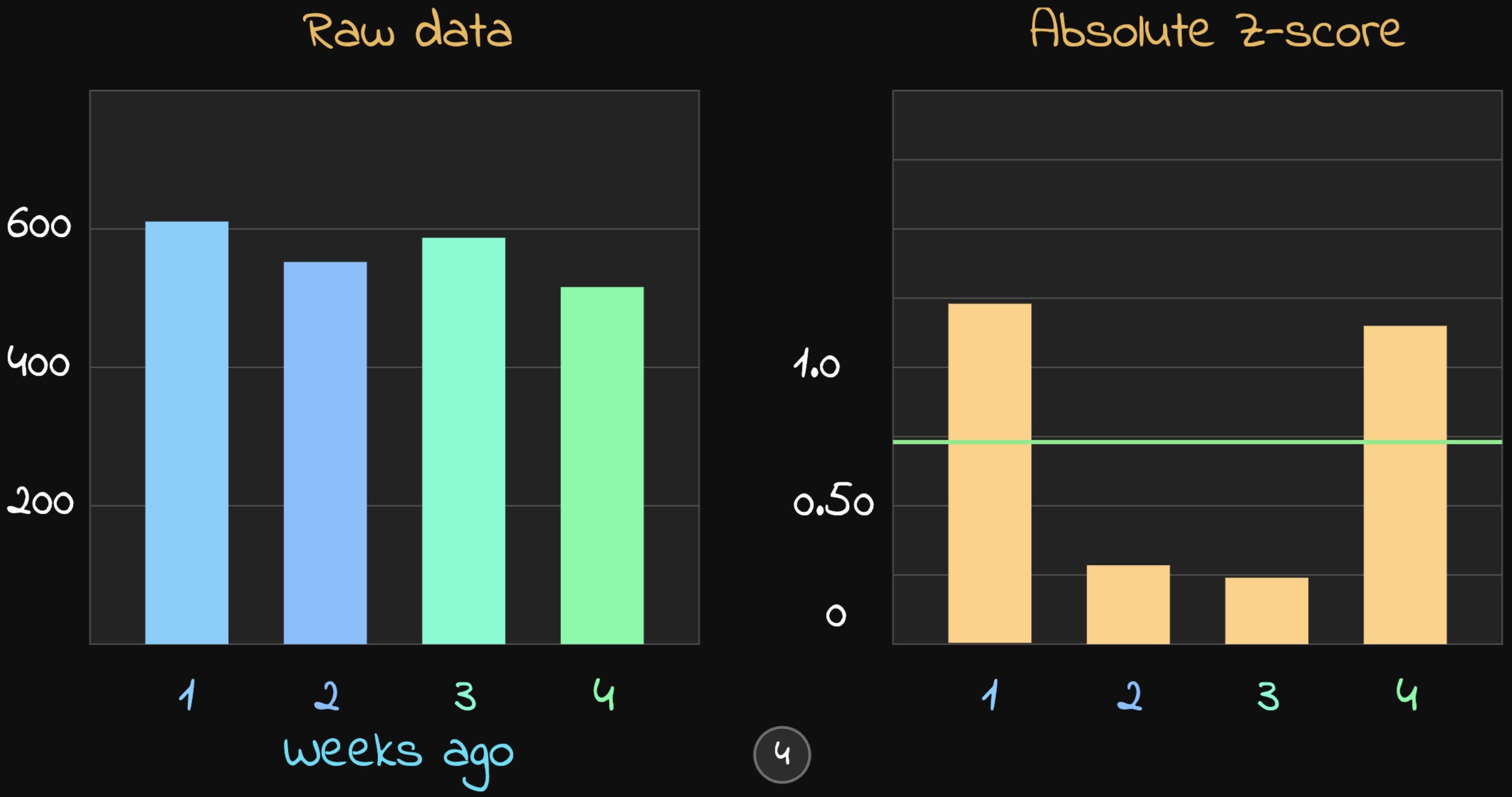

Raw data
Absolute z-score
600
400
200
1.0
0.50
0
1
2
3
4
1
2
3
4
weeks ago

No outliers

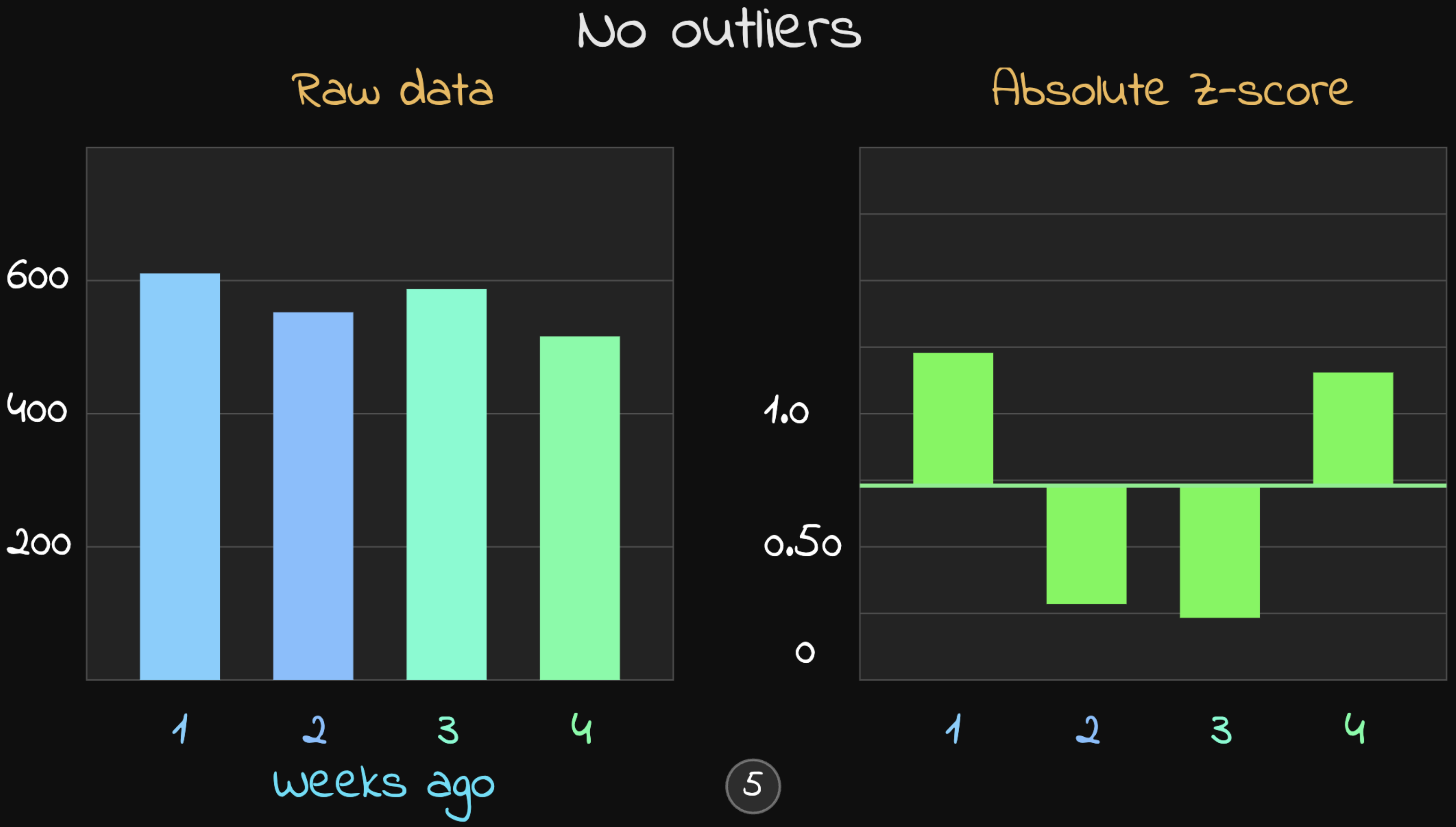


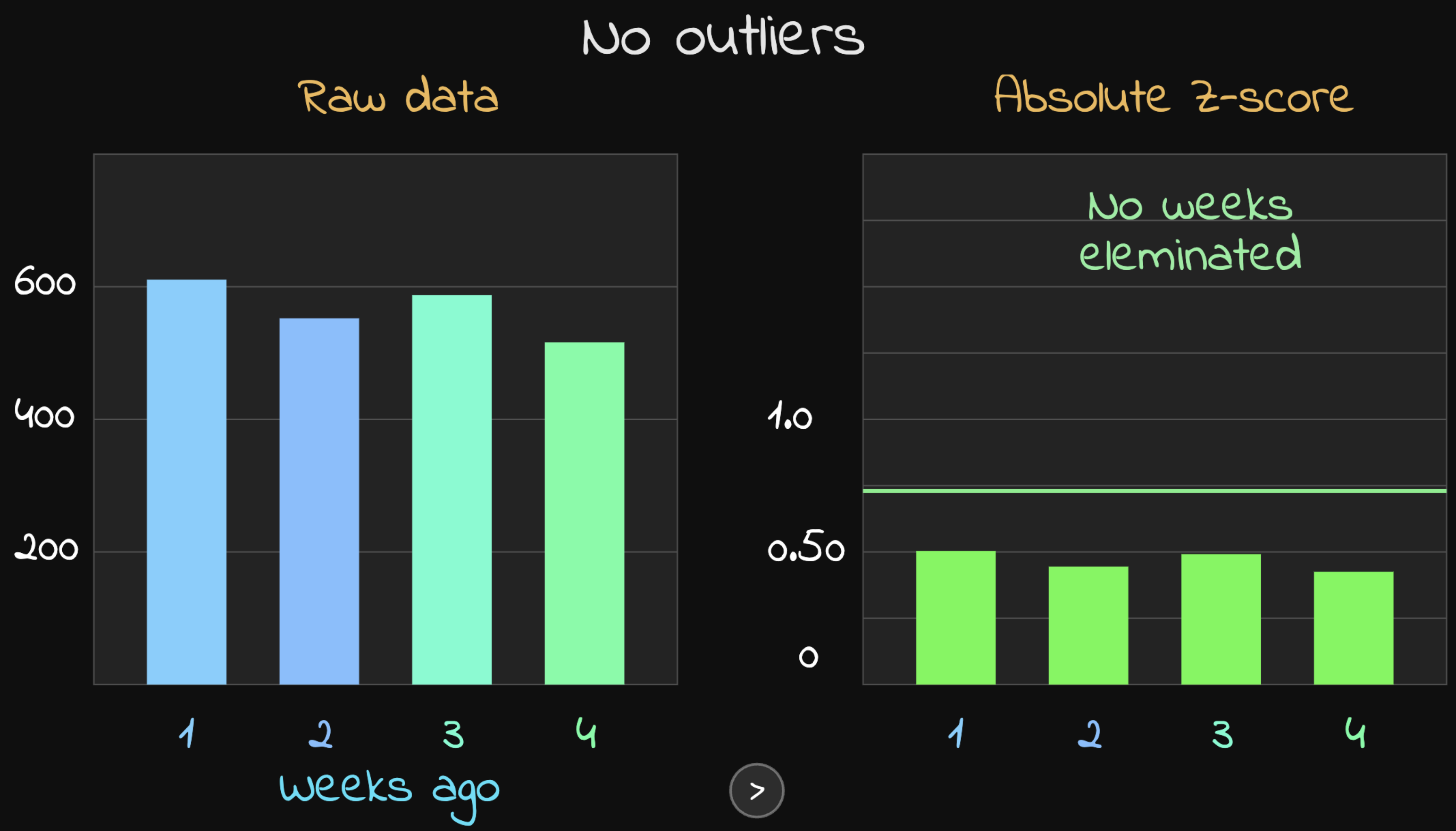
3

No outliers

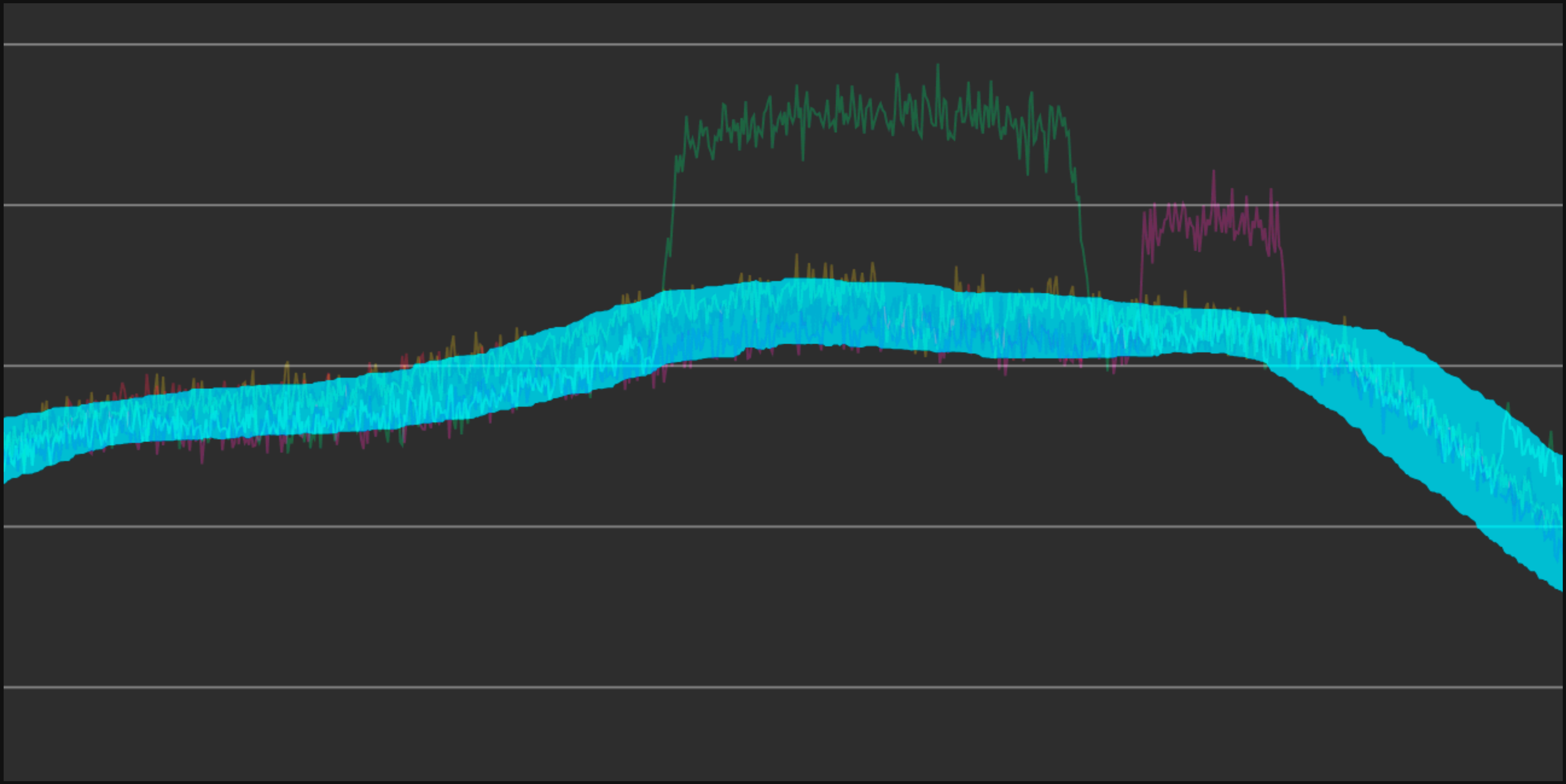


Raw data
600
400
200
1
2
3
4
weeks ago
Absolute z-score
1.0
0.50
0
1
2
3
4

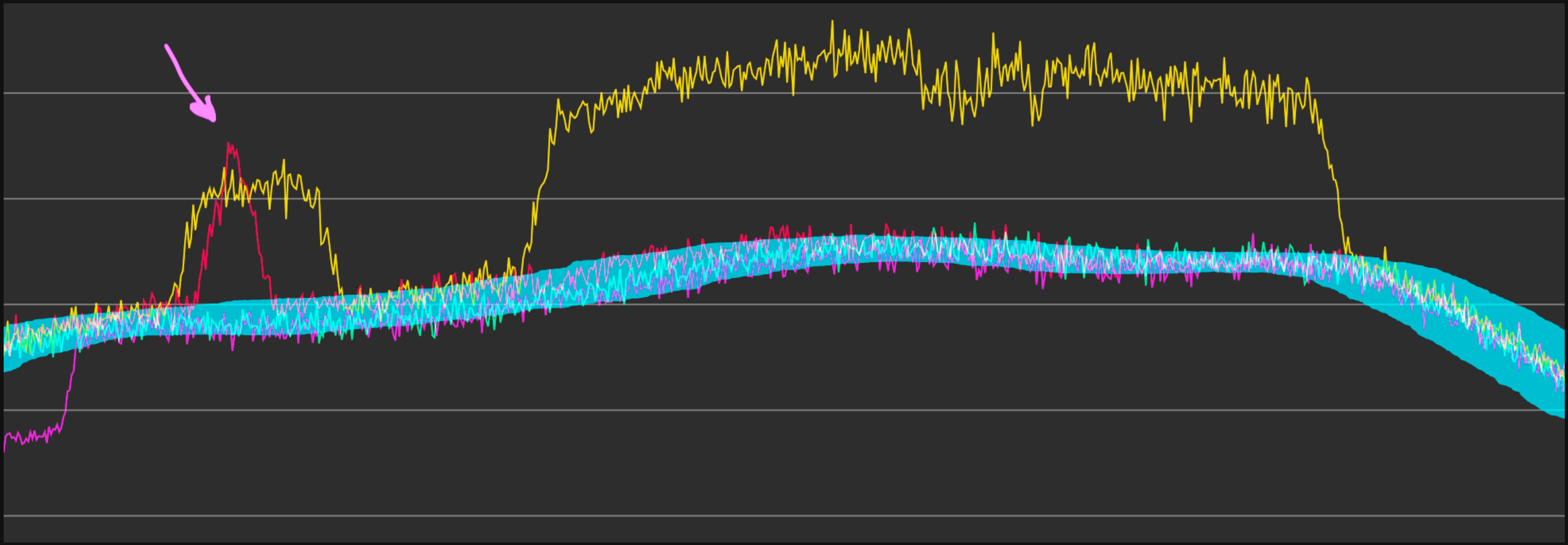

No outliers
Raw data
Absolute Z-score
600
400
200
1.0
0.50
0
1
2
3
4
weeks ago
1
2
3
4
5


No outliers
Raw data
Absolute Z-score
No weeks
eleminated
600
400
200
1.0
0.50
0
1
2
3
4
1
2
3
4
weeks ago
>

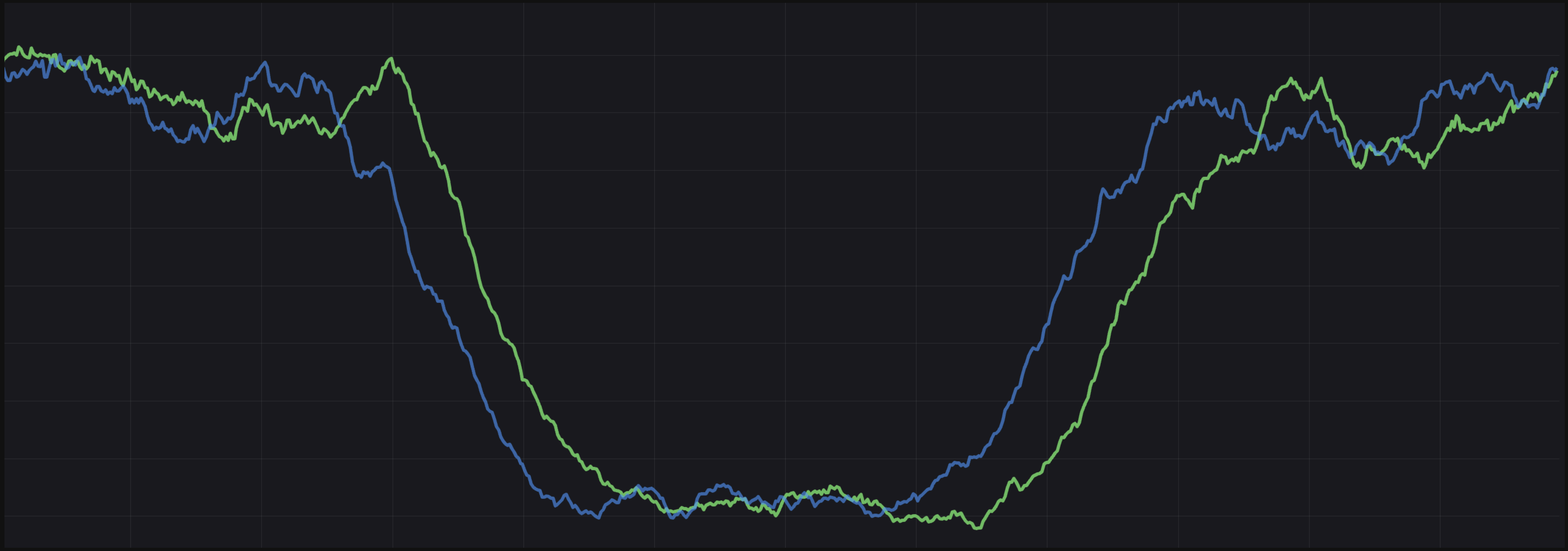
Result with 5th percentile



Overlapping incidents



Problem #2: DST



Problem #2: DST

```
func sameTimeWeeksAgo(currentTime time.Time, weeksAgo int) time.Time {
    t := currentTime.Add(-time.Duration(weeksAgo*minutesInWeek) * time.Minute)
    _, cOffset := currentTime.Zone()
    _, tOffset := t.Zone()
    diff := cOffset - tOffset
    return t.Add(time.Duration(diff) * time.Second)
}
```

Problem #2: DST

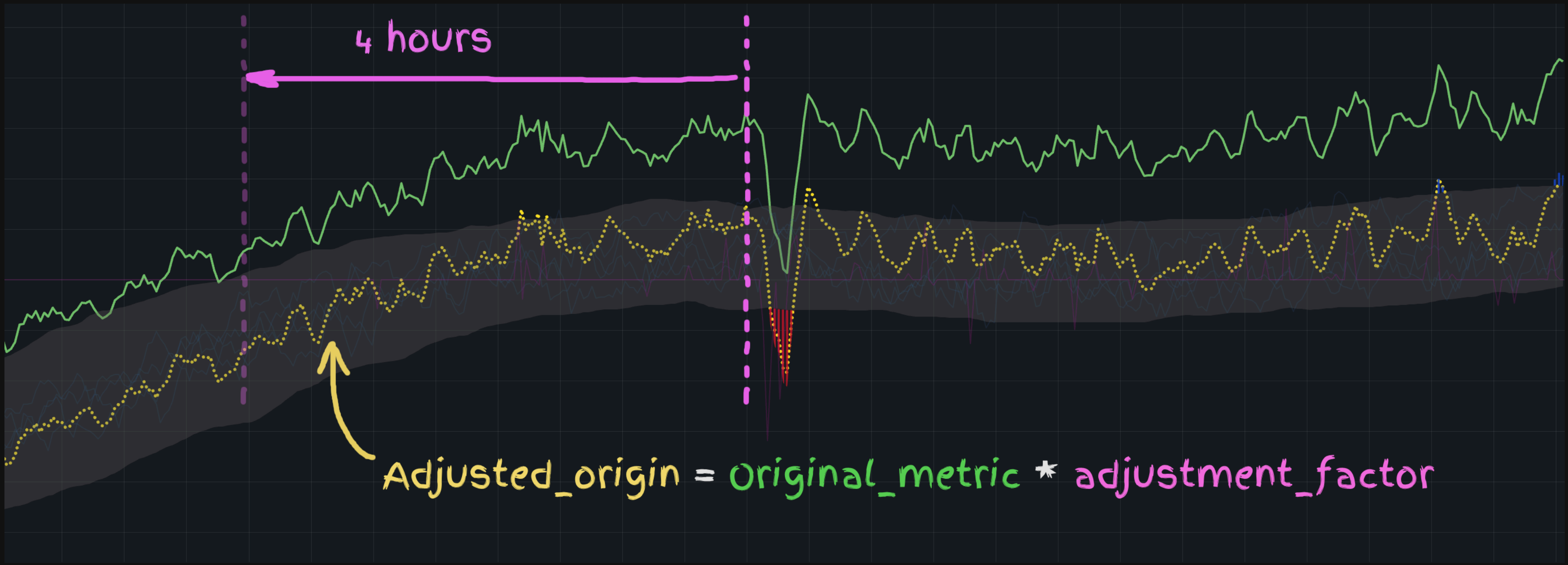
- Not all countries have DST
- Some users do not adapt instantly to new time

Problem #3: Overperforming

Problem #3: Overperforming

Problem #3: Overperforming

Problem #3: Overperforming



Problem #4: Known events

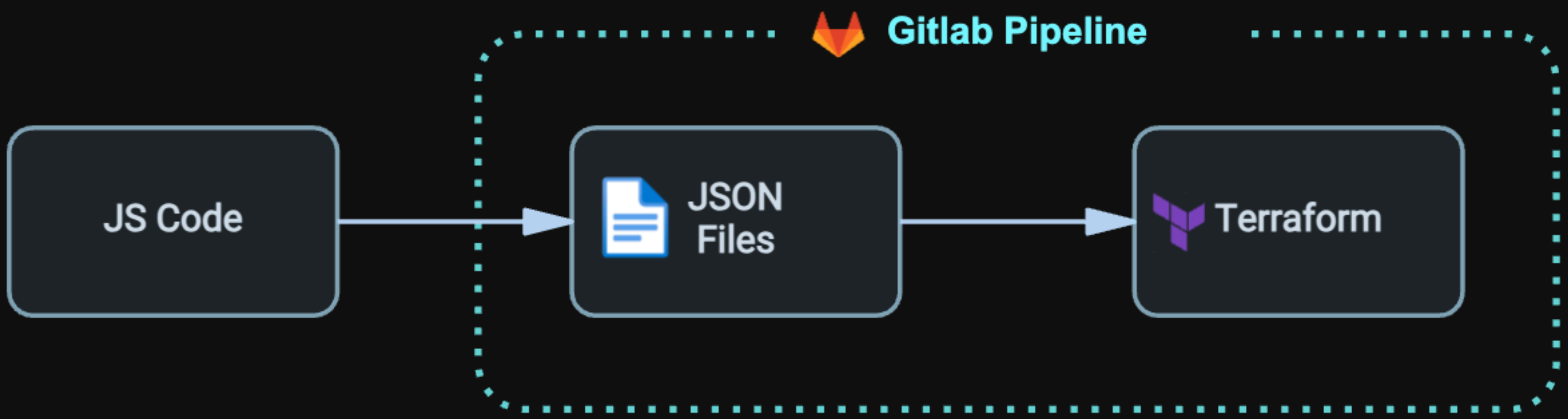
```yaml
corrections:
  nightly:
    baseline: 0
    daily:
      - constant: 6
        ranges:
          - ['00:00', '09:00']

  weekends:
    baseline: 0
    weekly:
      - constant: 10
        days: ['Saturday', 'Sunday']

  holidays:
    baseline: 0
    calendar:
      - name: 'Ascension Day'
        constant: 10
        ranges:
          - ['2023-05-18 00:00', '2023-05-18 23:59']
```

Problem #5: Complexity

(Graphite Prod)

```
alias(sumSeries(minSeries(transformNull(removeAboveValue(diffSeries(movingAverage
(sum(some.example.metric.count), '$smoothing'),
multiplySeries(movingAverage(some.example.metric.percentiles.$percentile.lower,
'$smoothing'), offset(scale(movingAverage(some.example.metric.correction,
'$smoothing'), -0.01), 1))), 0), 0),
transformNull(removeAboveValue(diffSeries(multiplySeries(movingAverage(sum(some.e
xample.metric.count), '$smoothing'),
movingAverage(some.example.metric.origin_adjustment.scale, '$smoothing')),
multiplySeries(movingAverage(some.example.metric.percentiles.$percentile.lower,
'$smoothing'), offset(scale(movingAverage(some.example.metric.correction,
'$smoothing'), -0.01), 1))), 0), 0)),
transformNull(removeBelowValue(diffSeries(movingAverage(sum(some.example.metric.c
ount), '$smoothing'),
multiplySeries(movingAverage(some.example.metric.percentiles.$percentile.upper,
'$smoothing'), offset(scale(movingAverage(some.example.metric.correction,
'$smoothing'), 0.01), 1))), 0), 0)), 'offset')
```

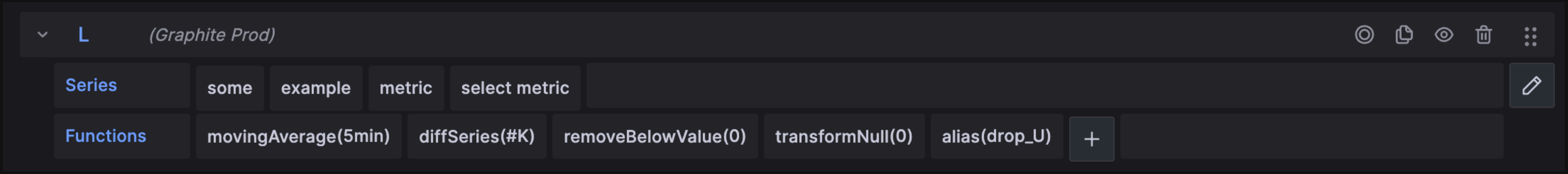
Dashboards as code



```javascript
class QueryBuilder {
    constructor(target) {
        this.target = target;
    }
    groupByNodes(operator, ...indices) {
        return new QueryBuilder(`groupByNodes(${this.target}, '${operator}', ${indices.join(',')})`);
    }

    sum() {
        return new QueryBuilder(`sum(${this.target})`);
    }

    movingAverage(period) {
        return new QueryBuilder(`movingAverage(${this.target}, '${period}')`);
    }

    divideSeries(otherSeries) {
        return new QueryBuilder(`divideSeries(${this.target}, ${otherSeries})`);
    }

    multiplySeries(otherSeries) {
        return new QueryBuilder(`multiplySeries(${this.target}, ${otherSeries})`);
    }

    maxSeries(...series) {
```

```
const rise = query(current.build())
    .diffSeries(upper.build())
    .removeBelowValue(0)
    .transformNull(0);

const drop = query(current.build())
    .diffSeries(lower.build())
    .removeAboveValue(0)
    .transformNull(0);

const dropAdjusted = query(adjustedCurrent.build())
    .diffSeries(lower.build())
    .removeAboveValue(0)
    .transformNull(0);

const finalDrop       = drop.minSeries(dropAdjusted.build());
const offset          = finalDrop.sumSeries(rise.build());
const slowBurnOffset  = drop.sumSeries(rise.build());
```

```
const rise = query(current.build())
    .diffSeries(upper.build())
    .removeBelowValue(0)
    .transformNull(0);

const drop = query(current.build())
    .diffSeries(lower.build())
    .removeAboveValue(0)
    .transformNull(0);

const dropAdjusted = query(adjustedCurrent.build())
    .diffSeries(lower.build())
    .removeAboveValue(0)
    .transformNull(0);

const finalDrop      = drop.minSeries(dropAdjusted.build());
const offset         = finalDrop.sumSeries(rise.build());
const slowBurnOffset  = drop.sumSeries(rise.build());
```



Understanding the anomaly

Breakdown by Regions, Devices, Order/Users types, Marketing channels etc.

Recap

- Basic statistics works for detecting anomalies
- User driven metric is better for anomaly detection
- Median absolute deviation vs Standard Deviation
- Grafana can be used for complex calculations
- Understanding anomaly is harder than detecting it.

Thank you!