Visualization of Performance Anomalies with Kieker

Bachelor's Thesis

Sören Henning

September 8, 2016

Outline

Christian-Albrechts-Universität zu Kiel
Technische Fakultät

Motivation

- Performance is a significant quality characteristic
  - e.g., Amazon: 100 ms delay $\rightarrow$ 1% decrease in sales (Huang 2011)
  - e.g., Google: 500 ms delay $\rightarrow$ 20% drop in traffic (Huang 2011)

Motivation

- Performance is a significant quality characteristic
  - e.g., Amazon: 100 ms delay $\rightarrow$ 1% decrease in sales (Huang 2011)
  - e.g., Google: 500 ms delay $\rightarrow$ 20% drop in traffic (Huang 2011)
- Detect performance issues to react on them
  - As soon as possible
  - Use monitoring (e.g., measure execution times)
  - Investigate these measurements for anomalies

Motivation

- Performance is a significant quality characteristic
  - e.g., Amazon: 100 ms delay $\rightarrow$ 1% decrease in sales (Huang 2011)
  - e.g., Google: 500 ms delay $\rightarrow$ 20% drop in traffic (Huang 2011)
- Detect performance issues to react on them
  - As soon as possible
  - Use monitoring (e.g., measure execution times)
  - Investigate these measurements for anomalies
- Visualization can support interpretation of anomalies

ΘPAD's Anomaly Detection Approach

ΘPAD and ΘPADx

- Provide anomaly detection
- Part of Kieker
- Only R algorithms
- Problematic anomaly score
- No visualization
- More on this later

Development of an approach to detect performance anomalies with Kieker and visualize them

Goals

- G1: Migrate ΘPAD to a TeeTime configuration

Goals

- G1: Migrate ΘPAD to a TeeTime configuration
- G2: Implement multiple forecast algorithms

Goals

- G1: Migrate ΘPAD to a TeeTime configuration
- G2: Implement multiple forecast algorithms
- G3: Provide a visualization of measured time series and detected anomalies

Goals

- G1: Migrate ΘPAD to a TeeTime configuration
- G2: Implement multiple forecast algorithms
- G3: Provide a visualization of measured time series and detected anomalies
- G4: Evaluate the Implementation
  - G4.1. Feasibility evaluation
  - G4.2. Scalability evaluation

Foundations

- Performance Metrics (Koziolek 2008)
  - Time behavior and resource efficiency
- Time Series (Mitsa 2010)
  - Sequence of measurements at regular temporal intervals
- Anomaly Detection (Chandola, Banerjee, and Kumar 2009)
  - Anomaly: Abnormal data patterns
  - Detection: Compare measured values with reference model

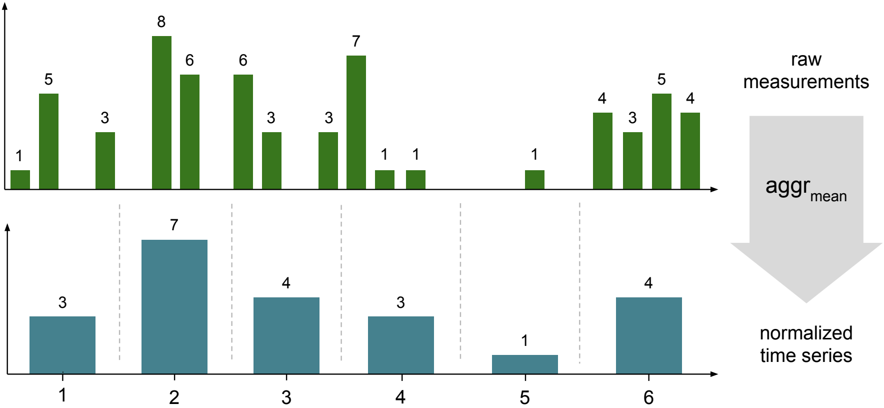


Figure: Based on Bielefeld (2012) and Frotscher (2013)

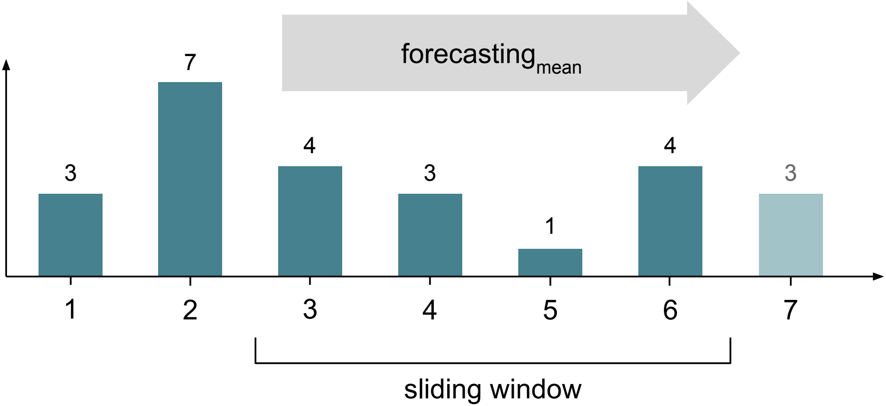
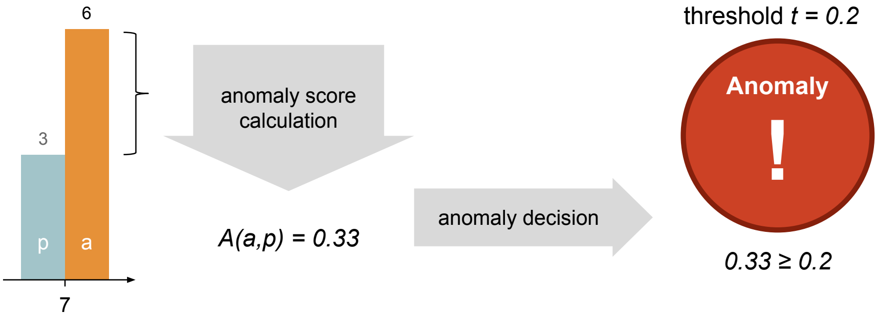


Figure: Based on Bielefeld (2012) and Frotscher (2013)

ΘPAD's Anomaly Detection Approach

Anomaly Score Calculation

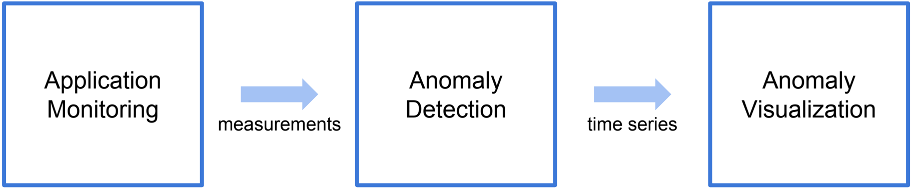


Technologies

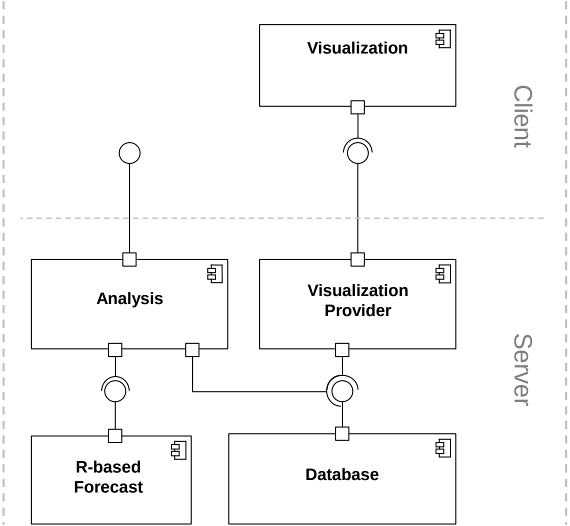
- Microservices Architectural Pattern (Wolff 2015)
- Kieker Monitoring Framework (Hoorn et al. 2009)
- TeeTime Pipe and Filter Framework (Wulf, Ehmke, and Hasselbring 2014)
- And further technologies (see next slides)

Outline

Graphical Overview of our Approach

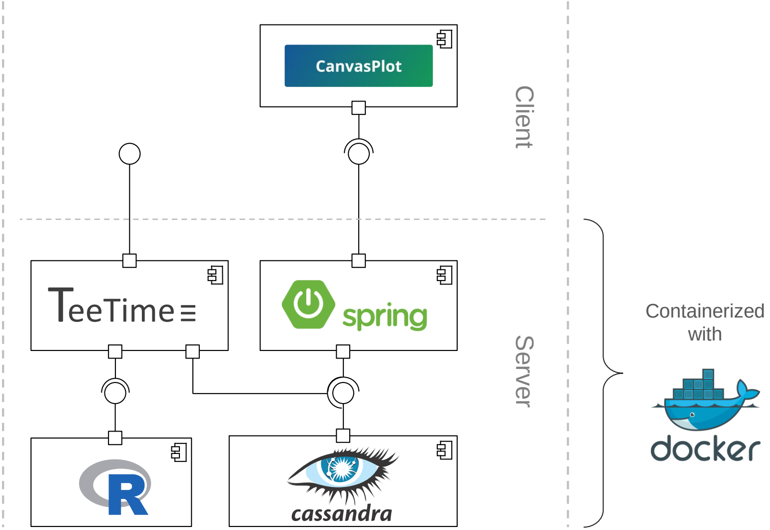


Architecture of our Approach

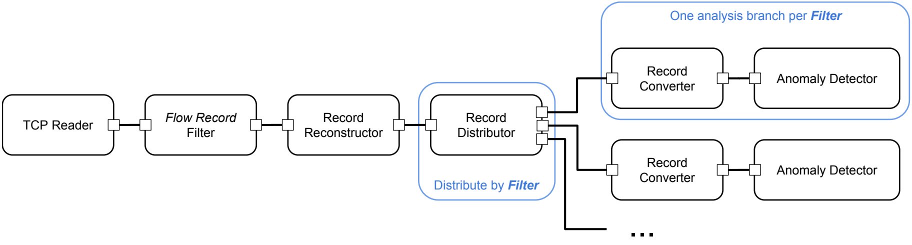


Architecture of our Implementation

Approach

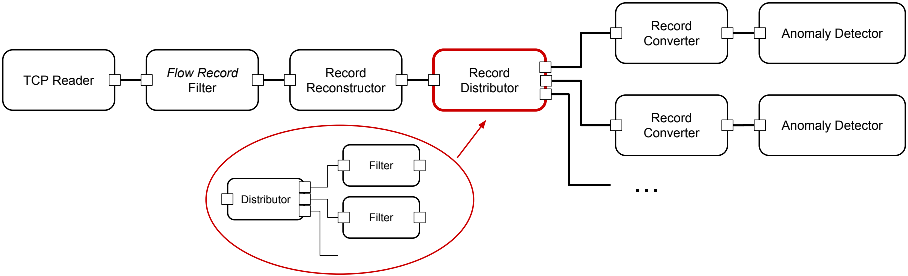


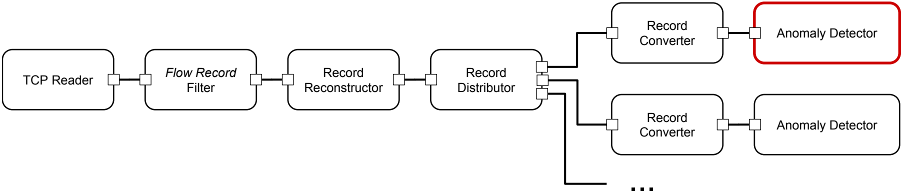
Christian-Albrechts-Universität zu Kiel
Technische Fakultät



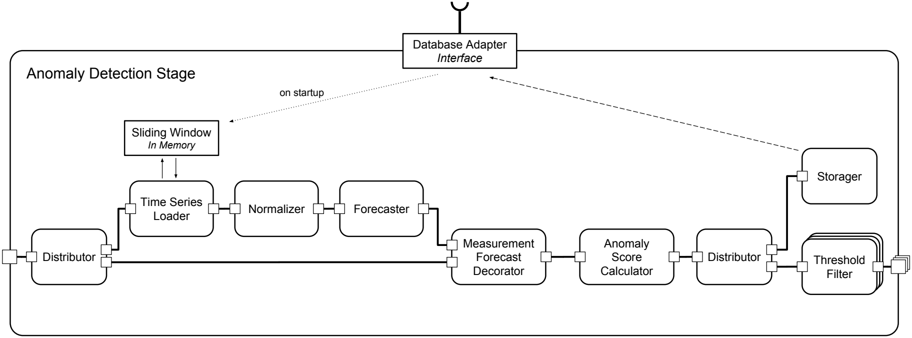
Filter:

- Operation signature
- Class signature
- Host name
- ...


TCP Reader
Flow Record Filter
Record Reconstructor
Record Distributor
Record Converter
Anomaly Detector
Record Converter
Anomaly Detector
...
Distributor
Filter
Filter



TCP Reader
Flow Record Filter
Record Reconstructor
Record Distributor
Record Converter
Anomaly Detector
Record Converter
Anomaly Detector
...

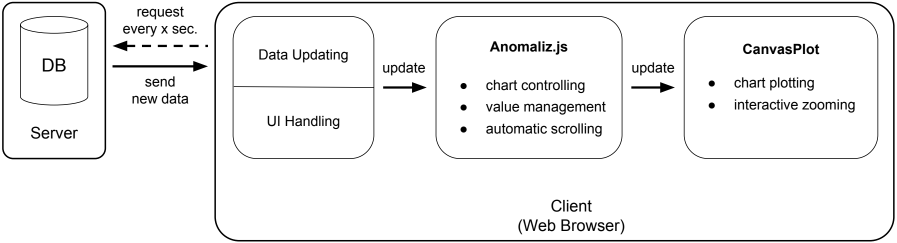


Database Adapter
Interface
Anomaly Detection Stage
on startup
Sliding Window
In Memory
Time Series Loader
Normalizer
Forecaster
Storager
Distributor
Measurement Forecast Decorator
Anomaly Score Calculator
Distributor
Threshold Filter

Demo of Visualization

Approach

Architecture of Visualization



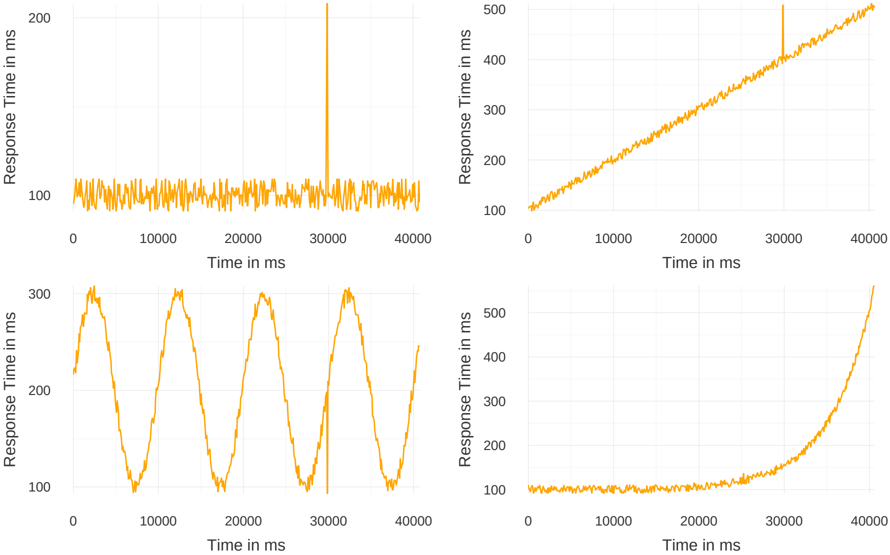
- Usage of Arne Johanson's CanvasPlot (Johanson 2016)

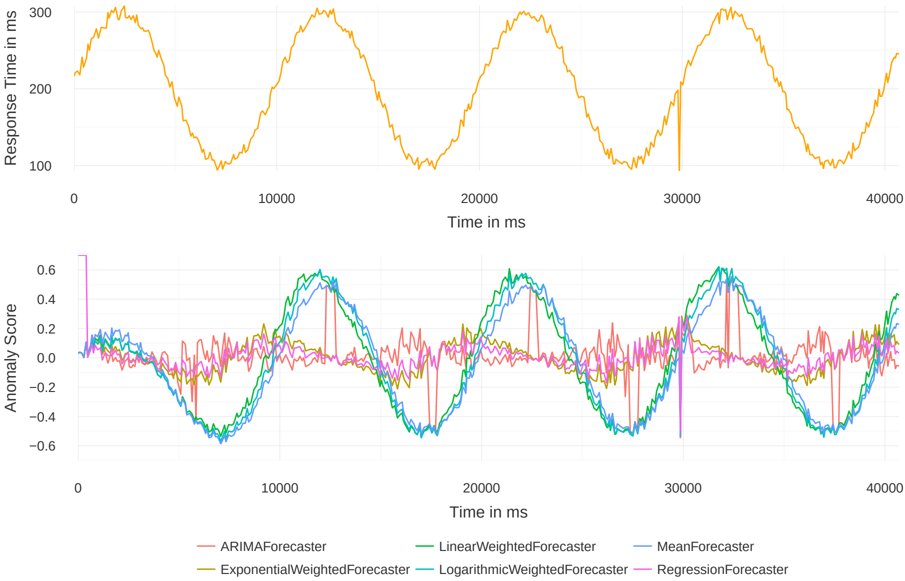
Outline

1. Introduction
2. Foundations
3. Approach
4. Evaluation
5. Conclusion and Future Work

Feasibility Evaluation

Scenarios




Response Time in ms
300
200
100
0
10000
20000
30000
40000
Time in ms
Anomaly Score
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
10000
20000
30000
40000
Time in ms
ARIMAForecaster
LinearWeightedForecaster
MeanForecaster
ExponentialWeightedForecaster
LogarithmicWeightedForecaster
RegressionForecaster

- Take time for record processing in analysis

- Take time for record processing in analysis
- Evaluate: Execution time $\leq$ measurement frequency ?

- Take time for record processing in analysis
- Evaluate: Execution time $\leq$ measurement frequency ?
- For all parameter combinations:

Measurement frequencies 2 ms, 5 ms, 10 ms, 50 ms, 100 ms, 150 ms, 200 ms

Sliding window 10,000 ms, 50,000 ms, 100,000 ms, 150,000 ms, 200,000 ms, 400,000 ms

Normalization interval 10 ms, 20 ms, 100 ms, 200 ms, 500 ms, 1000 ms, 2000 ms

Forecast algorithm ARIMAForecaster, RegressionForecaster

Normalization algorithm MeanAggregator

Some examples:

freq.	sld. window	norm. intvl.	forecaster	∅ exec. time
5	10,000	200	Regression	1.64
5	400,000	20	Regression	4.35
50	50,000	200	ARIMA	69.98
100	100,000	500	ARIMA	78.24
150	200,000	2,000	ARIMA	187.21

...

all values in ms

Outline

1. Introduction
2. Foundations
3. Approach
4. Evaluation
5. Conclusion and Future Work

- Further development of the ΘPAD Approach
  - Proving infrastructure via Docker containers
  - Immediately record processing

- Further development of the ΘPAD Approach
  - Proving infrastructure via Docker containers
  - Immediately record processing
- Native Java algorithms for anomaly detection

- Further development of the ΘPAD Approach
  - Proving infrastructure via Docker containers
  - Immediately record processing
- Native Java algorithms for anomaly detection
- Providing an interactive, real time visualization

- Further development of the ΘPAD Approach
  - Proving infrastructure via Docker containers
  - Immediately record processing
- Native Java algorithms for anomaly detection
- Providing an interactive, real time visualization
- All implementations available as open source: github.com/SoerenHenning

Future Work

- Handling of fast incoming measurements
  - Aggregate before analysis (ΘPAD)
  - Cache time series operations

Future Work

- Handling of fast incoming measurements
  - Aggregate before analysis (ΘPAD)
  - Cache time series operations
- Parallelized and distributed analysis
  - Is or will be supported by TeeTime

Future Work

Christian-Albrechts-Universität zu Kiel
Technische Fakultät

- Handling of fast incoming measurements
  - Aggregate before analysis (ΘPAD)
  - Cache time series operations
- Parallelized and distributed analysis
  - Is or will be supported by TeeTime
- Take advantage of Cassandra's features for data storage

Future Work

- Handling of fast incoming measurements
  - Aggregate before analysis (ΘPAD)
  - Cache time series operations
- Parallelized and distributed analysis
  - Is or will be supported by TeeTime
- Take advantage of Cassandra's features for data storage
- Configuration via Rest/GUI

References I

Bielefeld, Tillmann Carlos (2012). "Online Performance Anomaly Detection for Large-Scale Software Systems". Diploma Thesis. Germany: Kiel University.

Chandola, Varun, Arindam Banerjee, and Vipin Kumar (2009). "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* 41.3, p. 15.

Frotscher, Tom (2013). "Architecture-Based Multivariate Anomaly Detection for Software Systems". Master's Thesis. Germany: Kiel University.

Hoorn, André van et al. (2009). *Continuous Monitoring of Software Services: Design and Application of the Kieker Framework*. Tech. rep. TR-0921. Department of Computer Science, Kiel University, Germany.

References II

Huang, Cheng (2011). *Public DNS System and Global Traffic Management.* Accessed: 2016-08-25. URL: http://research.microsoft.com/en-us/um/people/chengh/slides/pubdns11.pptx.pdf.

Johanson, Arne (2016). *CanvasPlot.* Accessed: 2016-08-25. URL: https://a-johanson.github.io/canvas-plot.

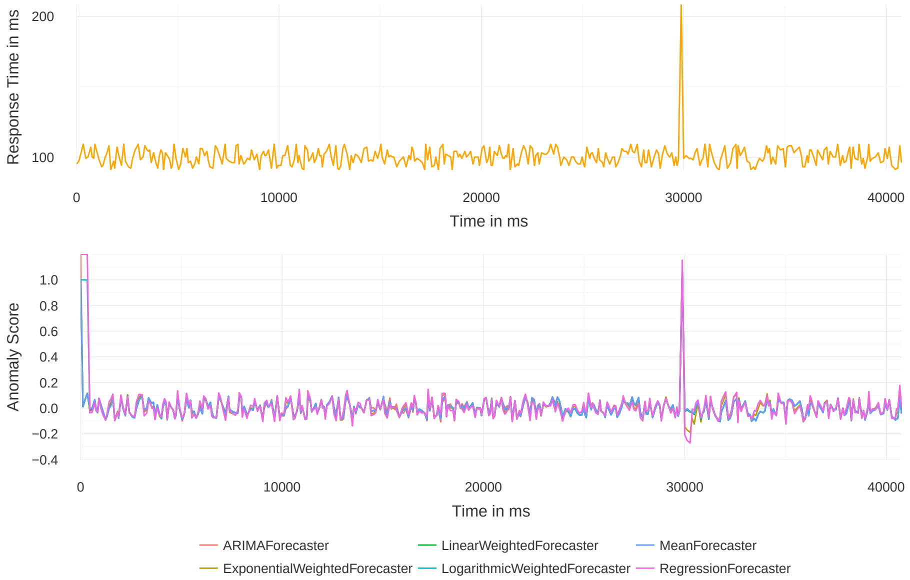
Koziolek, Heiko (2008). "Dependability Metrics". In: ed. by Irene Eusgeld, Felix C. Freiling, and Ralf Reussner. Berlin, Heidelberg: Springer-Verlag. Chap. Introduction to Performance Metrics, pp. 199–203. ISBN: 3-540-68946-X, 978-3-540-68946-1.

Mitsa, T. (2010). *Temporal Data Mining.* Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press. ISBN: 9781420089776.

- Wolff, E. (2015). *Microservices: Grundlagen flexibler Softwarearchitekturen*. Dpunkt.Verlag GmbH. ISBN: 9783864903137.
- Wulf, Christian, Nils Christian Ehmke, and Wilhelm Hasselbring (2014). "Toward a Generic and Concurrency-Aware Pipes & Filters Framework". In: *Symposium on Software Performance 2014: Joint Descartes/Kieker/Palladio Days*.

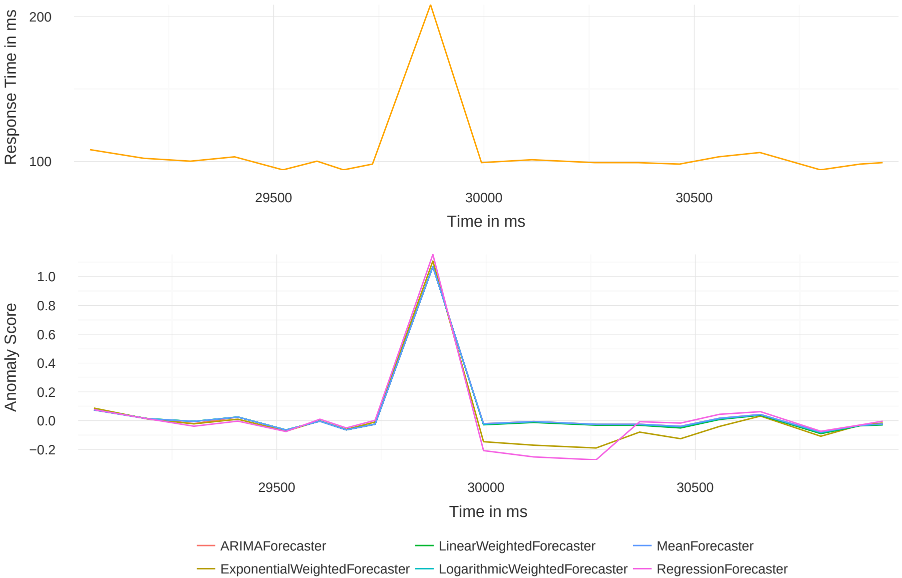
Feasibility Evaluation

Scenario: Constant with Anomaly



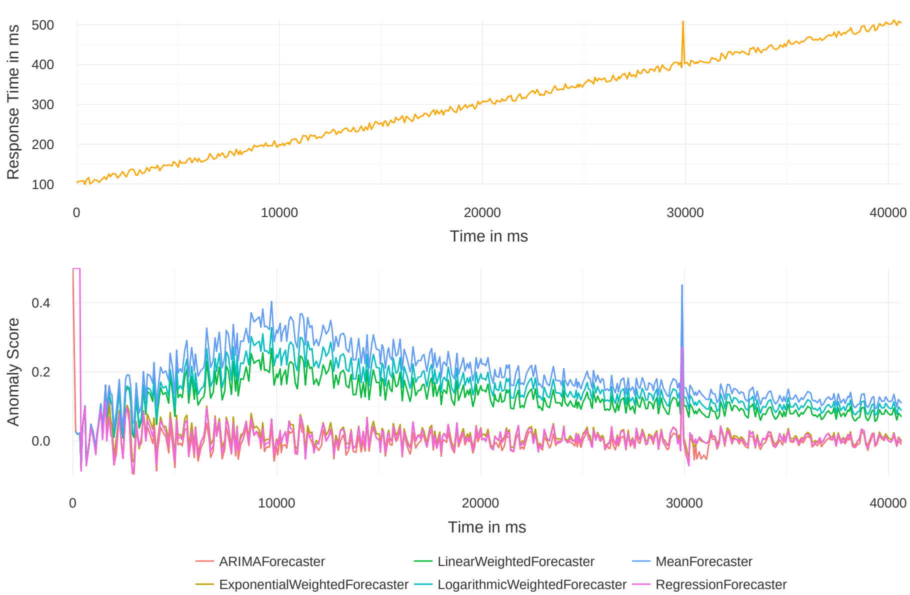
Feasibility Evaluation

Scenario: Constant with Anomaly - Detail



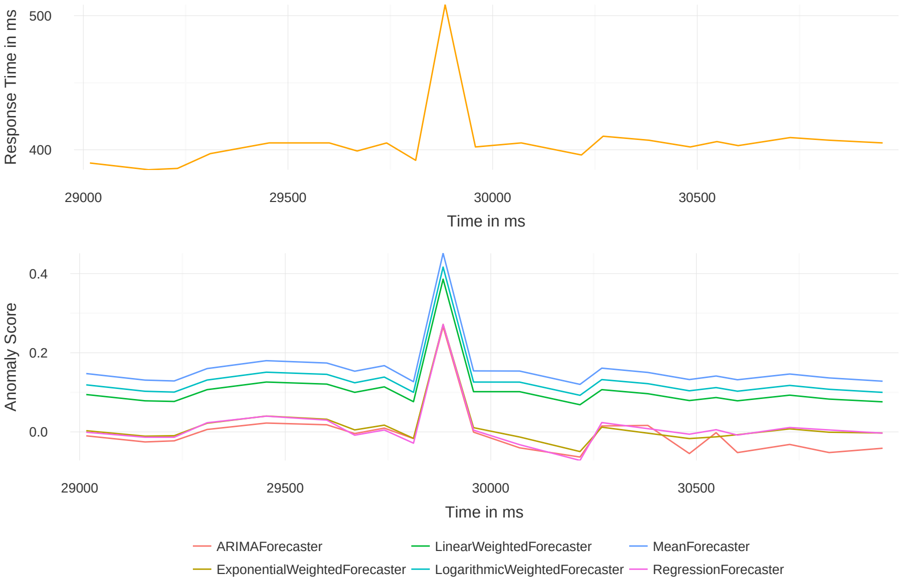
Feasibility Evaluation

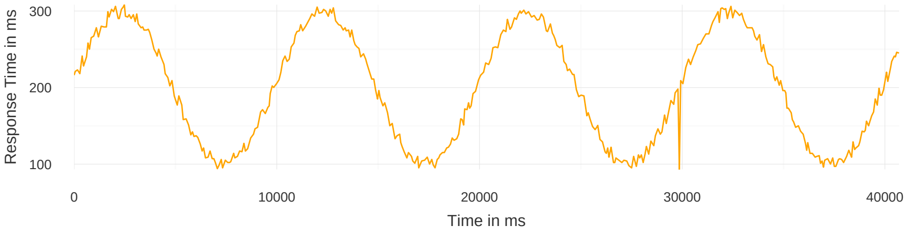
Scenario: Linearly Increasing with Anomaly

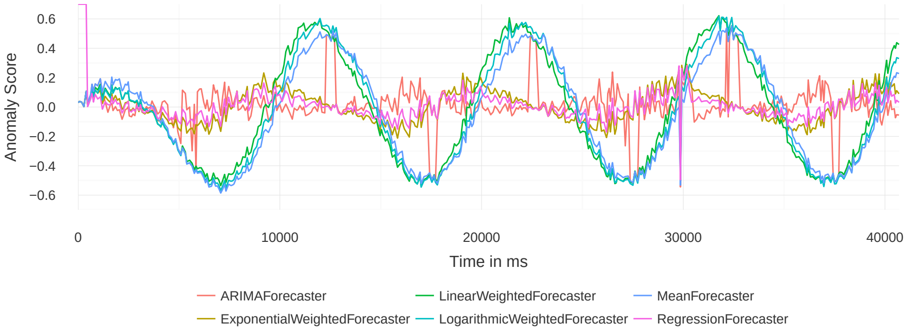


Feasibility Evaluation

Scenario: Linearly Increasing with Anomaly - Detail

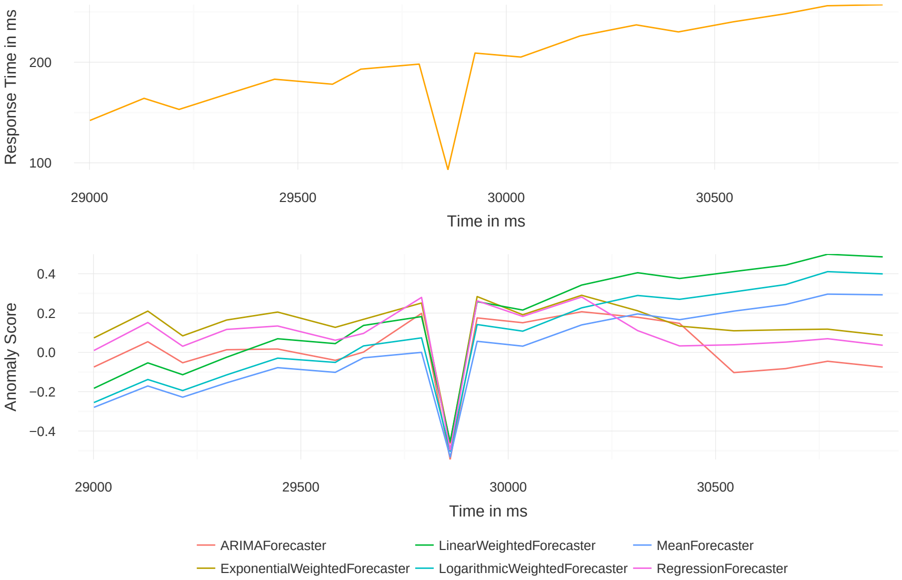



Response Time in ms
300
200
100
0
10000
20000
30000
40000
Time in ms


Anomaly Score
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
10000
20000
30000
40000
Time in ms
ARIMAForecaster
LinearWeightedForecaster
MeanForecaster
ExponentialWeightedForecaster
LogarithmicWeightedForecaster
RegressionForecaster

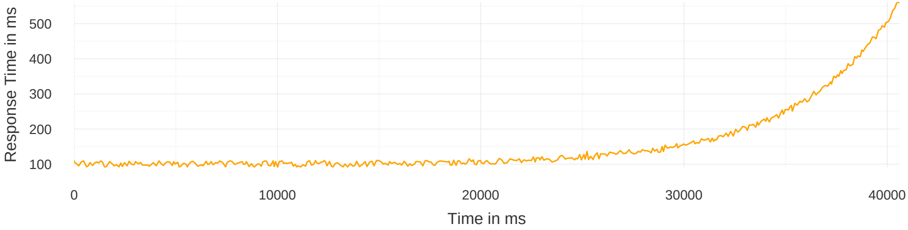
Feasibility Evaluation

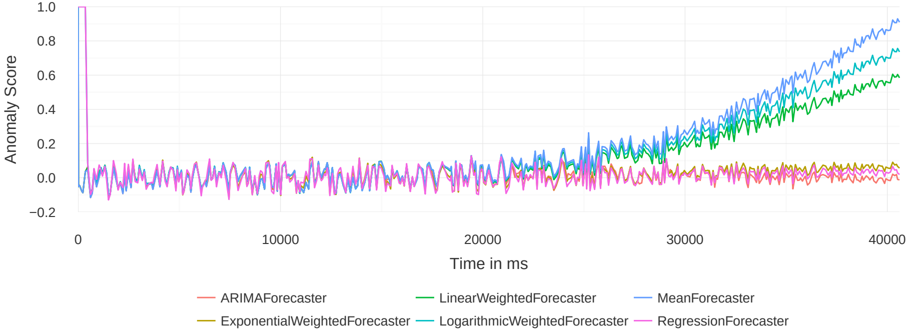
Scenario: Seasonal with Anomaly - Detail



Feasibility Evaluation

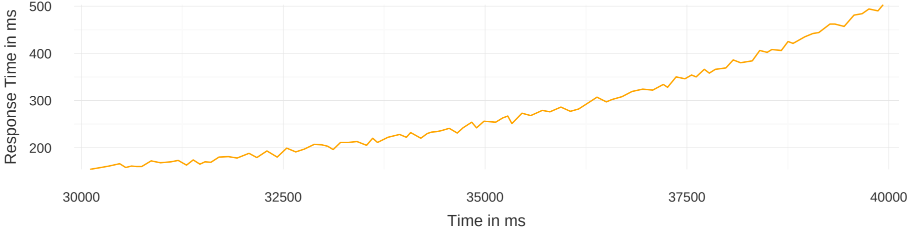
Scenario: Exponential Increasing

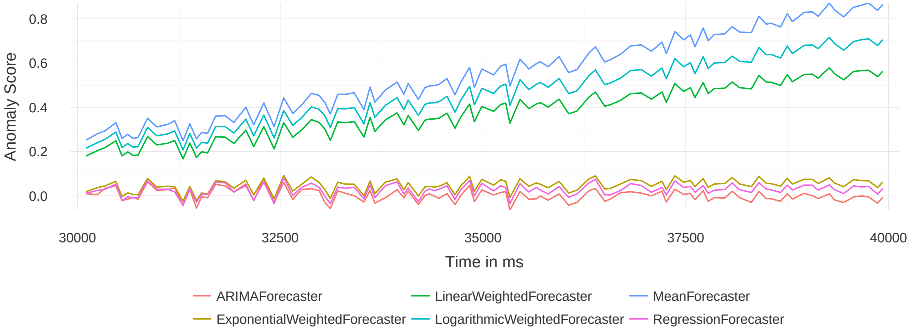




Feasibility Evaluation

Scenario: Exponential Increasing - Detail





KiekPAD
Anomaly Detection
demo-method
Response Time in ns
400,000,000
350,000,000
300,000,000
250,000,000
200,000,000
150,000,000
100,000,000
50,000,000
0
:55
13:28
:05
:10
:15
:20
:25
:30
:35
:40
:45
:50
:55
13:29
:05
:10

KiekPAD
Anomaly Detection
demo-method
Predictions
Anomaly Scores
Thresholds
-0.3
0.3
Refresh Interval
500
Response Time in ns
400,000,000
350,000,000
300,000,000
250,000,000
200,000,000
150,000,000
100,000,000
50,000,000
0
:55
13:28
:05
:10
:15
:20
:25
:30
:35
:40
:45
:50
:55
13:29
:05
:10
Anomaly Score
0.5
0.0