technische universität
dortmund

Bachelorarbeit

Strukturbrucherkennung zur frühzeitigen Erkennung von Inhomogenitäten im Verhalten der Pixel von FACT

Nils Killich
Oktober 2018

Gutachter:
Prof. Dr. Katharina Morik
M. Sc. Alexey Egorov

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl für Künstliche Intelligenz (LS VIII)
http://www-ai.cs.tu-dortmund.de

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Das *First G-APD Cherenkov Telescope (FACT)* wird zur Beobachtung von Gammastrahlungsquellen verwendet. Die Strahlung dieser Quellen kann auf der Erde nicht direkt beobachtet werden, da sie mit den Atomen in der Erdatmosphäre interagiert. Diese Interaktion löst jedoch eine Kaskade geladener Teilchen aus, die Teilchenschauer genannt wird. Da die Geschwindigkeit der Teilchen die Lichtgeschwindigkeit in der Luft übersteigt, wird Cherenkovstrahlung ausgelöst, die vom Teleskop beobachtet werden kann. Durch die beobachteten Eigenschaften der Cherenkovstrahlung können Rückschlüsse auf die ursprüngliche Strahlungsquelle gezogen werden. Das *FACT* ist das erste Cherenkovteleskop, das siliziumbasierte Photosensoren verwendet. Die Kamera des *FACT* besteht aus 1440 solcher Sensoren, die eintreffendes Licht in elektronische Signale umwandeln. Das Teleskop soll als Vorlage für weitere Teleskope dieser Art dienen [6, 9].

Da diese Teleskope automatisiert und ferngesteuert betrieben werden, ist eine schnelle, manuelle Überprüfung der einzelnen Pixel auf korrekte Funktion sehr aufwendig. So kann es vorkommen, dass defekte Bildpunkte erst spät entdeckt werden und die automatische Auswertung im Nachhinein korrigiert werden muss. Gerade beim parallelen Betrieb mehrerer Teleskope dieser Art, wie er beispielsweise beim *CTA* [1] Projekt erfolgen soll, ist eine automatische Detektion von Defekten und eine frühzeitige Warnung wünschenswert. Dazu sollen die täglichen Kalibriermessungen im Zeitverlauf untersucht werden. Mithilfe bekannter Methoden zur Strukturbrucherkennung sollen Fehlfunktionen erkannt werden.

1.2 FACT

Die Kamera des *FACT* besteht aus 1440 Photodetektoren auf Basis von Geigermode-Avalanche-Photodioden (G-APD). Werden diese oberhalb einer sogenannten Durchbruchspannung betrieben, löst ein eintreffendes Photon einen Strom in der Diode aus. Die Ladung, die von einer Diode freigesetzt wird, ist unabhängig von der Anzahl der eintreffenden Photonen. Um mehrere Photonen zu zählen wird die photoaktive Fläche der Sensoren in kleinere G-APD-Zellen aufgeteilt, um die Bildpunkte des Teleskops zu bilden. Die Anzahl der auslösenden Dioden im Sensor entspricht daher der Anzahl der eintreffenden Photonen. Somit ist es möglich die Anzahl der eintreffenden Photonen festzustellen, wenn die freigesetzte Ladung pro Photon bekannt ist [9].

Da die entstehenden Ströme abhängig von den Umgebungsbedingungen des Teleskops sind, werden regelmäßige Messungen zur Kalibrierung durchgeführt. Dazu werden die sogenannten *Dark counts* gemessen. Dabei werden Daten bei geschlossener Kameraabdeckung aufgezeichnet. Die Rate mit der die Dioden auslösen, ohne dass Licht in die Kamera fällt, hängt ebenfalls von diesen Umgebungsbedingungen ab und eignet sich somit gut zur Kalibration [9, 6].

Um aus den aufgezeichneten Daten auf die, durch ein eintreffendes Photon freigesetzte, Ladung zu schließen, werden die Daten zunächst nach steigenden Flanken durchsucht, um die Ankunftszeiten zu extrahieren. Nachdem diese bestimmt wurden, wird das Signal über einem festen Zeitintervall integriert, um auf einen Wert für die freigesetzte Ladung zu schließen. Diese Werte werden *extracted Charge* genannt. Um diese genauer zu untersuchen, werden sie in ein Histogramm übertragen. Dieses ist beispielhaft in Abbildung 1.1 dargestellt. Die aus diesem Histogramm extrahierten Daten können dann weiter untersucht werden, um inhomogenes Verhalten einzelner Pixel zu erkennen. Im einfachsten Fall bedeutet das, den vollständigen Ausfall eines Pixels festzustellen. Ferner sollen schon fehlerhafte Kalibration und ungewöhnliche Abweichungen im Messverhalten der Pixel erkannt werden. Im Rahmen dieser Arbeit soll vor allem der *Gain* untersucht werden. Außerdem sind *Baseline, Rate, Noise* und *Crosstalk* von Interesse, die im Rahmen dieser Arbeit jedoch nicht genauer betrachtet werden. Der *Gain* beschriebt die Ladung, die beim Eintreffen eines Photons freigesetzt wird. Um diesen Wert zu messen, wird der Abstand zweier benachbarter Maxima im Histogramm bestimmt [9].

1.3 Zielsetzung

Anhand der aufgezeichneten Kalibriermessungen soll die korrekte Funktion der Sensoren des *FACT* untersucht werden. Ausgefallene Sensoren sowie solche, die ungewöhnliches Verhalten zeigen, sollen automatisch erkannt werden. Dazu sollen die vorgestellten, etablierten Methoden zur Strukturbrucherkennung verwendet werden. Dabei wird untersucht,

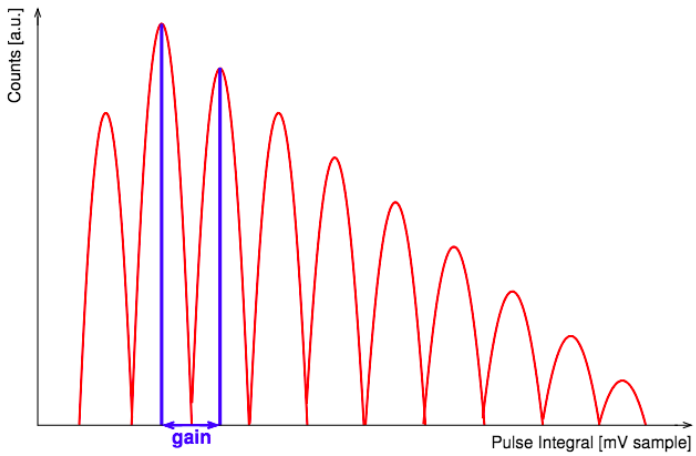


Abbildung 1.1: Beispielspektrum der *extracted charge* eines Pixels. In blau ist der bestimmte *Gain* markiert [9].

inwieweit sich diese Methoden für diese Untersuchung eignen und welchen Einfluss unterschiedliche Ansätze zur Vorverarbeitung der Daten haben. Durch die Verwendung der vorgestellten Frameworks *streams* und *MOA (Massive Online Analysis)* kann auf bereits vorhandene und getestete Implementierungen dieser Ansätze zurückgegriffen werden, um den Fokus auf den Vergleich der Leistungsfähigkeit in diesem Anwendungsfall zu legen.

1.4 Aufbau der Arbeit

Kapitel 2 beschreibt die verwendeten Methoden. Zunächst werden die Begriffe Zeitreihe und Strukturbruch beschrieben. Daraufhin werden die verwendeten Methoden zur Strukturbrucherkennung und Vorverarbeitung vorgestellt.

In Kapitel 3 werden die vorgestellten Methoden experimentell verglichen und die verwendeten Frameworks beschrieben. Danach werden die *Gain*-Daten des *FACT* genauer untersucht. Es wird diskutiert, wie die Ergebnisse verglichen werden können, da keine Informationen über tatsächliche Strukturbrüche verfügbar sind. Nachdem die Methoden an vier ausgewählten Pixeln verglichen wurden, werden die erfolgversprechendsten Methoden auf alle Pixel angewendet und die Ergebnisse beschrieben.

Kapitel 4 fasst die Ergebnisse zusammen und diskutiert die Leistungsfähigkeit der vorgestellten Methoden für das gegebene Problem.

Kapitel 2

Grundlagen

Um die *Gain*-Daten zu untersuchen, soll die Zielsetzung formalisiert werden. Dazu werden die Begriffe Zeitreihe und Strukturbruch eingeführt. Anschließend werden die Methoden zur Entdeckung von Strukturbrüchen, die zur Identifikation der Inhomogenitäten verwendet werden, vorgestellt.

2.1 Zeitreihen

Die betrachteten Teleskopdaten werden als Zeitreihen aufgefasst. Entsprechend der Definition aus [8] besteht eine solche Zeitreihe aus den Beobachtungen x_t, die zu den Zeitpunkten $t \in T$ betrachtet werden. Da T diskret ist spricht man auch von einer zeitdiskreten Zeitreihe. Zur Modellierung der Zeitreihe, werden diese x_t als Realisierungen einer Folge von Zufallsvariablen $\{X_t\}_{t \in T}$ mit gemeinsamer Verteilung interpretiert. Anstelle dieser T-dimensionalen Verteilungsfunktion werden häufig die Momente erster und zweiter Ordnung zur Beschreibung verwendet. In [8] definieren die Autoren zusätzlich zwei wichtige Eigenschaften von Zeitreihen.

Definition 2.1.1

Beschreibe $\{X_t\}$ eine Zeitreihe mit $\mathrm{Var}(X_t) < \infty$. Dann sei $\mu_X(t) = \mathrm{E}(X_t)$ die Erwartungswertfunktion von $\{X_t\}$. Die Kovarianzfunktion von $\{X_t\}$ sei $\gamma_X(r, s) = \mathrm{Cov}(X_r, X_s)$.

Definition 2.1.2

$\{X_t\}$ heißt stark stationär, falls die Verteilung von $\{X_{t+h}\}_{t\in T}$ nicht von der Verschiebung h abhängt. $\{X_t\}$ heißt schwach stationär, falls

(i) $\mu_X(t)$ nicht von t abhängt, das heißt für alle t gilt $\mathrm{E}(X_t) = \mu$ und

(ii) $\gamma_X(t+h,t)$ nicht von t abhängt $\forall h$.

Aus Bedingung (ii) folgt für schwach stationäre Zeitreihen insbesondere, dass $\mathrm{Var}(X_1) = \ldots = \mathrm{Var}(X_n) = \sigma^2$.

Um zuverlässige Aussagen über die zukünftige Entwicklung einer Zeitreihe zu treffen, muss Stationarität vorausgesetzt werden. Anderfalls kann von den vergangenen Werten nicht auf die zunkünftigen Daten geschlossen werden. Dazu kann eine Transformation der ursprünglichen Zeitreihe erforderlich sein, sodass die transformierte Zeitreihe stationär ist. Es kann beispielsweise erforderlich sein, Trends und saisonale Komponenten zu entfernen. Für eine (schwach) stationäre Zeitreihe ist es dann beispielsweise möglich, Schätzer für μ und σ^2 zu bestimmen, da diese Werte konstant sind.

2.2 Strukturbrucherkennung

Ein Strukturbruch liegt vor, falls sich die gemeinsame Verteilung der Zeitreihe verändert. Das heißt die beobachteten Werte nach dem Strukturbruch sind Realisierungen eines anderen Prozesses. Falls die Zeitreihe $X_1, \ldots, X_n$ in die stationären Zeitreihen $X_1, \ldots, X_\tau$ und $X_{\tau+1}, \ldots, X_n$ aufgeteilt werden kann und die Zeitreihe $X_1, \ldots, X_n$ nicht stationär ist, liegt ein Strukturbruch zum Zeitpunkt $t = \tau$ vor. Wenn in der betrachteten Zeitreihe mehrere Strukturbrüche auftreten, kann die Zeitreihe analog in mehrere Teilreihen unterteilt werden. Nachdem ein Strukturbruch festgestellt wurde, muss ein bereits gelerntes Modell verworfen, oder angepasst werden. In Abbildung 2.1 sind zwei Strukturbrüche skizziert. Im

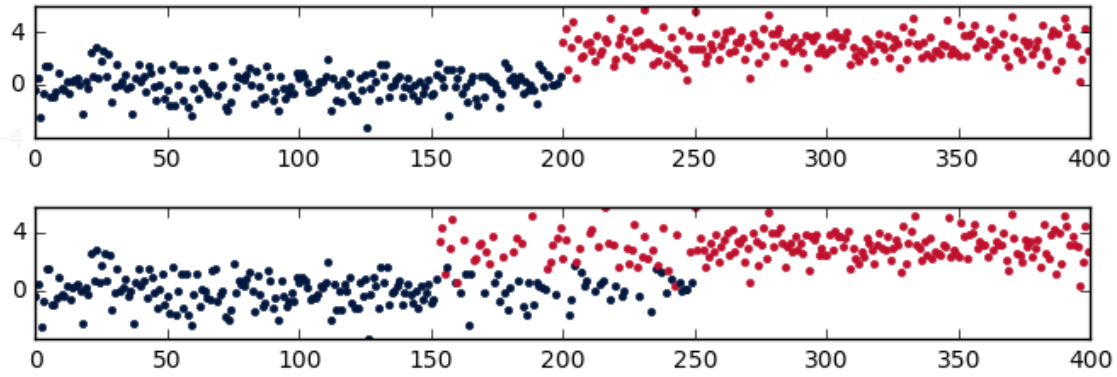


Abbildung 2.1: Skizzierte Darstellung von zwei Strukturbrüchen.

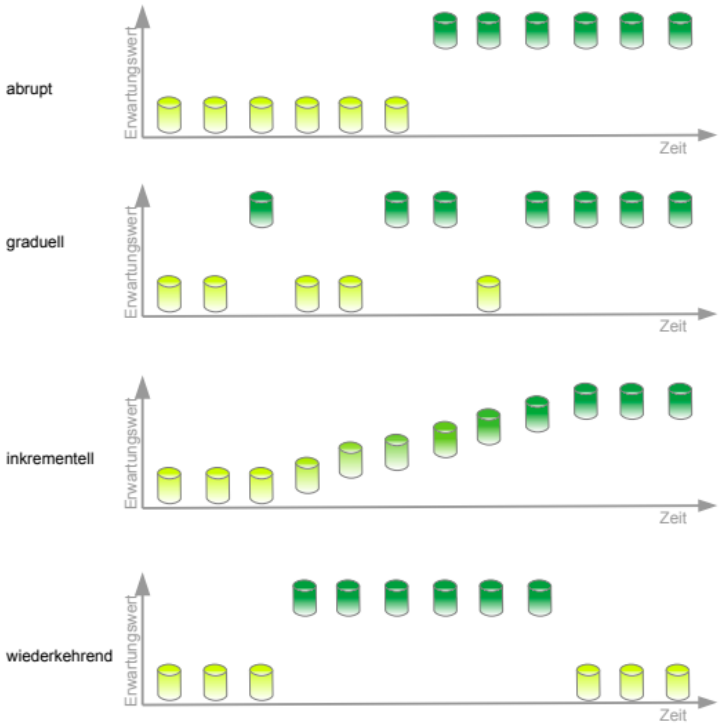


Abbildung 2.2: Strukturbrucharten [17]

ersten Beispiel ändert sich die Verteilung abrupt zum Zeitpunkt $t = 200$. Die dargestellten Zeitreihen bestehen aus unabhängig, gleich verteilten Zufallsvariablen $X_t \sim \mathcal{N}(0, 1)$, für $t < 200$ und $X_t \sim \mathcal{N}(4, 1)$, für $t \geq 200$. Dabei bezeichnet $\mathcal{N}$ die Normalverteilung. In diesem Fall ist der Strukturbruch deutlich zu erkennen. Im zweiten Beispiel wurden die Werte zwischen den Zeitpunkten $t = 150$ und $t = 250$ zusätzlich zufällig aus einer der beiden Verteilungen gezogen. Dadurch soll illustriert werden, dass ein Strukturbruch schwerer zu identifizieren ist, wenn die Änderung nicht abrupt erfolgt. In [17] werden die verschiedenen Arten von Strukturbrüchen noch weiter unterschieden. Die Arten sind in Abbildung 2.2 dargestellt. Neben der Unterscheidung zwischen abrupten und graduellen Strukturbrüchen, werden Strukturbrüche inkrementell genannt, falls die Änderungen zwischen den benachbarten Zeitpunkten so gering sind, dass die Änderung nur über einen längeren Zeitraum erkannt werden kann. Wenn die Verteilung nach einem ersten Strukturbruch wieder zur vorherigen Verteilung zurückkehrt, wird von einem wiederkehrenden Strukturbruch gesprochen.

Ein Algorithmus zur Strukturbrucherkennung soll feststellen, ob neue Beobachtungen signifikant für eine Veränderung des zu Grunde liegenden Prozesses sprechen. Die in [5] und [10] vorgestellten Methoden zur Strukturbrucherkennung werden im Folgenden weiter untersucht.

2.2.1 CUSUM und Page-Hinkley-Test

Der *Cumulative Sum* Algorithmus und der Page-Hinkley-Test werden in [15] und [5] beschrieben. Da verschiedene Varianten der Algorithmen existieren, orientiert sich die folgende Darstellung an der Implementierung des MOA-Frameworks, das in Kapitel 3 beschrieben wird und für die Experimente verwendet wurde. Seien $x_1, x_2, \ldots, x_n$ Realisationen der Zufallsvariablen $X_1, X_2, \ldots, X_n$, dann soll ein Test für H_0 gegen H_1 gefunden werden mit:

$$H_0 : \; X_1, X_2, \ldots, X_n \sim P(X)$$
$$H_1 : \begin{cases} X_1, \ldots, X_\tau & \sim P_1(X) \\ X_{\tau+1}, \ldots, X_n & \sim P_2(X) \end{cases}, \; P_1 \neq P_2$$

Das heißt es soll getestet werden, ob durch die Stichprobe $x_1, \ldots, x_n$ auf einen Strukturbruch im Zeitraum $T = \{1, \ldots, n\}$ geschlossen werden kann. Um einen solchen Test zu finden, werden zunächst Fehlerwerte z_i für jeden Zeitpunkt $t \in T$ gebildet. In der später verwendeten Implementierung gilt für diese:

$$z_i = x_i - \bar{x}^{(i)} - \delta.$$

Dabei ist $\bar{x}^{(i)}$ das arithmetische Mittel der Eingabewerte bis zum Zeitpunkt i:

$$\bar{x}^{(i)} = \frac{1}{i} \sum_{j=1}^{i} x_j.$$

δ wird als Eingabeparameter vom Benutzer gewählt, um die Empfindlichkeit des Tests anzupassen. Zur Durchführung des Tests werden die kumulativen Summen

$$S_n = \sum_{i=1}^{n} z_i \tag{2.1}$$

der Fehlerwerte gebildet. Die Nullhypothese wird abgelehnt, wenn

$$S_n - \min_{0 \leq j \leq n} S_j > \lambda. \tag{2.2}$$

Die Alternative wird also angenommen, sobald der aktuelle kumulative Fehler um mindestens λ größer als der minimale kumulative Fehler ist. λ wird ebenfalls als Parameter vom Benutzer gewählt. Falls kein Strukturbruch vorliegt, bleibt z_i konstant (oder sinkt). Sollte sich die Verteilung jedoch ändern, führt dies zu einer Veränderung von z_i, die erkannt wird. Es ist zu beachten, dass es sich bei CUSUM um einen einseitigen Test handelt. Es werden also nur positive Änderungen festgestellt. Um negative Änderungen festzustellen kann die angepasste Bedingung

$$S_n + \max_{0 \leq j \leq n} S_j < \lambda$$

verwendet werden.

Der oben beschriebene Test setzt voraus, dass $x_1, \ldots, x_n$ bereits bekannt sind. Um auch im Datenstrom verwendet werden zu können, wird der Test zu einer Onlinevariante erweitert. Dazu wird die Statistik g_i

$$\begin{aligned} g_0 &= 0 \\ g_t &= \max(0, g_{t-1} + z_t) \end{aligned}$$

gespeichert. Da $g_t = 0$ falls $S_t < \min_{0 \leq i \leq n} S_i$, ergibt sich äquivalent zu Bedingung (2.2):

$$\text{Sobald } g_t > \lambda: \text{ wird ein Strukturbruch erkannt.}$$

Neben g_t müssen zur Berechnung der z_i noch die Anzahl der Beobachtungen n, sowie $\bar{x}^{(i)}$ gespeichert werden. Nachdem ein Strukturbruch erkannt wird, werden alle gespeicherten Werte zurückgesetzt, um den Test erneut zu starten.

Der Page-Hinkley-Test verfolgt einen ähnlichen Ansatz wie CUSUM. Die Teststatistik unterscheidet sich allerdings:

$$\begin{aligned} g_0 &= 0 \\ g_t &= \alpha * g_{t-1} + z_t. \end{aligned}$$

Wie auch bei CUSUM wird ein Alarm ausgelöst, sobald $g_t > \lambda$. Der Parameter α $(0 < \alpha < 1)$ wird vom Benutzer gewählt.

2.2.2 Drift Detection Method (DDM)

Der von Gama et al. vorgestellte *DDM* Algorithmus [11] überwacht die Fehler des verwendeten Lernalgorithmuses. Dabei wird angenommen, dass die Voraussage des Lerners entweder richtig (d.h. $\hat{y}_t = y_t$) oder falsch (d.h. $\hat{y}_t \neq y_t$) sein kann. Die Wahrscheinlichkeit, ein Beispiel falsch zu klassifizieren, kann also als Ergebnis eines Bernoulliexperiments angesehen werden. Das Ergebnis des Experiments soll durch die Zufallsvariable B_t modelliert werden. Dabei sei

$$B_t = \begin{cases} 0 & ,\text{falls } \hat{y}_t = y_t \\ 1 & ,\text{falls } \hat{y}_t \neq y_t \end{cases}.$$

DDM ist in Algorithmus 1 skizziert. Es werden zwei Statistiken gespeichert: Die relative Häufigkeit ein Beispiel falsch zu klassifizieren p_t

$$p_t = \frac{1}{i} \sum_{i=0}^{t} b_i$$

und deren empirische Standardabweichung s_t

$$s_t = \sqrt{p_t \frac{1 - p_t}{t}}.$$

Weiterhin wird angenommen, dass die Wahrscheinlichkeit der Fehlklassifikation p_t mit steigender Beispielanzahl sinkt, da die Qualität des gelernten Modells mit zunehmender Beispielanzahl steigt, solange kein Strukturbruch vorliegt. Der Algorithmus soll dementsprechend einen Strukturbruch erkennen, sobald die Fehleranzahl signifikant größer als die minimale beobachtete Fehleranzahl ist. Daher werden p_{min} und s_{min} gespeichert. Das $1-\frac{\alpha}{2}$ Konfidenzintervall wird nach dem zentralen Grenzwertsatz durch $p_t \pm \alpha s_t$ approximiert, wenn genügend Beispiele ($n > 30$) vorliegen. Für einen Alarm wird das Signifikanzniveau $\alpha = 0,01$ festgelegt. Ein Strukturbruch wird daher erkannt sobald:

$$p_t + s_t \geq p_{min} + 3s_{min}.$$

Zusätzlich wird ein Warnungslevel mit dem Signifikanzniveau $\alpha = 0,05$ definiert. Eine Warnung wird also ausgegeben sobald:

$$p_t + s_t \geq p_{min} + 2s_{min}.$$

Durch dieses Warnungslevel soll eine frühzeitige Reaktion auf Strukturbrüche ermöglicht werden, ohne zu früh auf einen Strukturbruch zu schließen. Wenn solch eine Warnung ausgegeben wird, werden die folgenden Beispiele (x_t, y_t) gespeichert, das derzeitige Modell wird jedoch noch nicht angepasst. Erst wenn das Alarmlevel erreicht wird, muss das Modell angepasst oder verworfen werden. Dazu können die gespeicherten Werte verwendet werden. Falls das Warnungslevel unterschritten wird bevor ein Alarm ausgelöst wird, werden die gespeicherten Werte verworfen.

```
1  p_1 ← p_min ← b_1                                          // initialisieren
2  s_1 ← s_min ← √(p_0(1 − p_0))
3  foreach t>1 do
4  |  p_t ← (1/t)(p_{t−1}(t − 1) + b_t)
5  |  s_t ← √(p_t (1−p_t)/t)
6  |  if p_t + s_t < p_min + s_min then
7  |  |  p_min ← p_t                                          // Minimum anpassen
8  |  |  s_min ← s_t
9  |  end
10 |  if p_i + s_i ≥ p_min + 2s_min then
11 |  |  Warnung ausgeben                  // Die folgenden (x_t, y_t) werden gespeichert
12 |  end
13 |  if p_i + s_i ≥ p_min + 3s_min then
14 |  |  Alarm ausgeben                // gespeicherte (x_t, y_t) für neues Modell verwenden
15 |  |  p_min ← +∞
16 |  |  s_min ← +∞
17 |  end
18 end
```

Algorithmus 1 : DDM [11]

2.2.3 Early Drift Detection Method (EDDM)

Um die Erkennung inkrementeller Strukturbrüche zu verbessern, wird in [2] die *Early Drift Detection Method* vorgestellt. Diese Methode verfolgt eine vergleichbare Herangehensweise, allerdings wird nicht die Anzahl der Fehler verglichen. Stattdessen werden die Abstände zwischen den Fehlern untersucht. EDDM ist in Algorithmus 2 dargestellt. Der Abstand zwischen den Fehlern steigt während des Lernens, da das gelernte Modell mit steigender Beispielanzahl besser wird. Aus diesem Grund bestimmt der Algorithmus den durchschnittlichen Abstand zwischen falschen Vorhersagen p'_t und die entsprechende Standardabweichung s'_t. Diese Werte werden in p'_{max} und s'_{max} gespeichert, wenn $p'_t + 2s'_t$ das bisherige Maximum übersteigt. Eine Warnung wird ausgegeben, wenn gilt:

$$\frac{p'_t + 2s'_t}{p'_{max} + 2s'_{max}} < \alpha.$$

Auch hier werden die folgenden Beispiele gespeichert bis ein Strukturbruch erkannt wird, oder bis das Warnungslevel wieder unterschritten wird. Ein Strukturbruch wird festgestellt sobald:

$$\frac{p'_t + 2s'_t}{p'_{max} + 2s'_{max}} < \beta.$$

Daraufhin wird das gelernte Modell zurückgesetzt oder angepasst. Die gespeicherten Beispiele werden verwendet, um das neue Modell abzuleiten. Die Werte für p'_{max} und s'_{max} werden ebenfalls zurückgesetzt. Die Autoren empfehlen die Werte $\alpha = 0,95;\ \beta = 0,90$.

```
1  p'_t ← p'_max ← 0                                                    // initialisieren
2  s'_t ← s'_max ← √(p'_t(1 − p'_t))
3  n_err ← t_err ← 0
4  foreach t>1 do
5      if b_t = 1 then
6          n_err ← n_err + 1
7          Δt_err ← t − t_err
8          t_err ← t
9          p'_t ← 1/n_err (p'_t(n_err − 1) + Δt_err)
10         s'_t ← √(p'_t(1 − p'_t))
11         if p'_t + s'_t > p'_max + s'_max then
12             p'_max ← p'_t                                            // Maximum anpassen
13             s'_max ← s'_t
14         end
15         if (p'_t+2s'_t)/(p'_max+2s'_max) < α then
16             Warnung ausgeben                  // Die folgenden (x_t, y_t) werden gespeichert
17         end
18         if (p'_t+2s'_t)/(p'_max+2s'_max) < β then
19             Alarm ausgeben                // gespeicherte (x_t, y_t) für neues Modell verwenden
20             p'_max ← −∞
21             s'_max ← −∞
22             n_err ← p'_t ← s'_t ← 0
23             t_err ← t
24         end
25     end
26 end
```

Algorithmus 2 : EDDM [2]

2.2.4 EWMA Concept Drift Detection (ECDD)

Der ECCD-Algorithmus [16] erhält als Eingabe ebenfalls die Zufallsvariablen $B_1, \ldots, B_n$. Der gemeinsame Erwartungswert dieser Variablen vor dem Strukturbruch werde mit μ_0 bezeichnet. Der Erwartungswert nach dem Strukturbruch heiße μ_τ. Zudem bezeichne μ_t den aktuellen Erwartungswert zum Zeitpunkt t. Es soll zunächst angenommen werden, dass μ_0 und die Standardabweichung σ_B der Zufallsvariablen bekannt sind. Die Autoren benutzen den EWMA (exponentially weighted moving Average) Schätzer für μ_t:

$$\begin{aligned} Z_0 &= \mu_0 \\ Z_t &= (1-\lambda)Z_{t-1} + \lambda B_t \ , t > 0. \end{aligned} \tag{2.3}$$

Dieser Schätzer soll den aktuellen Wert von μ_t gut schätzen, da vergangene Werte, entsprechend ihres Alters, niedriger gewichtet werden. Es kann gezeigt werden, dass (unabhängig von der Verteilung der B_i) für den Erwartungswert und die Standardabweichung gilt:

$$\mathrm{E}(Z_t) = \mu_t$$

und

$$\sqrt{\mathrm{Var}(Z_t)} = \sigma_{Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})}\sigma_B.$$

Da die Zufallsvariablen B_i binomialverteilt sind, gilt außerdem $\mu_t = p_t$ und $\sigma_t = \sqrt{(1-p_t)p_t}$. p_t ist der Parameter der Biomailverteilung und beschreibt die relative Häufigkeit mit der ein Fehler auftritt. Daraus ergibt sich für die Standardabweichung des EWMA-Schätzers vor dem Strukturbruch:

$$\sigma_{0,Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})(1-p_0)p_0}. \tag{2.4}$$

Nach einem Strukturbruch ändert sich μ_t zu μ_τ. Der Schätzer Z_t wird sich daraufhin dem neuen Wert μ_τ annähern und sich damit vom alten Wert μ_0 entfernen. Ein Strukturbruch kann festgestellt werden, sobald dieser Abstand groß genug wird:

$$Z_t > \mu_0 + L\sigma_{0,Z_t}. \tag{2.5}$$

Durch den Parameter L wird bestimmt, wie weit Z_t von μ_0 abweichen muss, damit ein Strukturbruch erkannt wird. Dieser Parameter wird *control Limit* genannt. Er soll häufig so gewählt werden, dass die gewünschte erwartete Zeit zwischen Alarmen ARL (average run length) eingehalten wird. Das Verfahren, das von der gewünschten ARL auf den Parameter L schließt, ist nicht trivial und soll am Ende des Abschnitts nochmals thematisiert werden.

Da p_0 normalerweise nicht bekannt ist, wird ein weiterer Schätzer

$$\hat{p}_{0,t} = \frac{1}{t}\sum_{i=1}^{t} B_i = \frac{1}{t}\big[(t-1)\hat{p}_{0,t-1} + B_t\big]$$

für $p_0 = \mu_0$ definiert. Da dieser Schätzer neue Werte nicht stärker gewichtet, reagiert er langsamer auf die Änderungen von μ_t. Er kann deshalb als Schätzer für den Wert vor dem Strukturbruch (p_0) interpretiert werden. Um den Test zu erhalten, wird μ_0 in Gleichung (2.5) durch den Schätzer $\hat{p}_{0,t}$ ersetzt. Dadurch ergibt sich:

$$Z_t > \hat{p}_{0,t} + L\hat{\sigma}_{0,Z_t}.$$

Den Schätzer für die Standardabweichung des EWMA-Schätzer vor dem Strukturbruch erhält man durch Ersetzung in Gleichung (2.4):

$$\hat{\sigma}_{0,Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})(1-\hat{p}_{0,t})\hat{p}_{0,t}}.$$

Die Wahl von L ist nicht trivial, da es kein einfaches Verfahren gibt, um L für eine bestimmte ARL zu bestimmen. Es ist jedoch möglich von L auf die ARL zu schließen. Dazu wird eine Monte-Carlo-Simulation verwendet. Eine mögliche Lösung des Problems ist es daher so lange die ARL für verschiedene Parameter L zu bestimmen, bis eine gewünschte ARL gefunden wird. Dieses Verfahren ist im Regelfall sehr aufwendig und muss wiederholt werden wenn sich p_t ändert, um eine konstante ARL zu erhalten [16]. Die Autoren schlagen deshalb vor, die Funktion $f(p_0; ARL)$, die L für eine ARL und einen Wert p_0 bestimmt, durch ein Polynom anzunähern. So wird eine Lookuptabelle generiert, die Werte für L enthält, die gewählt werden müssen, um die gewünschte ARL zu erhalten. Da diese Tabelle im Vorhinein generiert werden kann, entsteht während der Strukturbrucherkennung wenig zusätzlicher Rechenaufwand.

2.2.5 Adaptive Sliding Window Algorithm

Einen anderen Ansatz liefert der *Adaptive Sliding Window Algorithm (ADWIN)*. ADWIN speichert ein Fenster W der neuesten x_i einer Zeitreihe. Die neuen x_i werden dazu an den Anfang des Fensters angefügt. Sobald zwei ausreichend große Teilfenster gebildet werden können, deren Mittelwerte signifikant unterschiedlich sind, werden x_i vom Ende des Fensters entfernt und ein Strukturbruchalarm wird ausgegeben [3].

Seien im Folgenden W_0 und W_1 Teilfenster von W, sodass $W = W_0 \cdot W_1$ und bezeichne $\hat{\mu}w$ den empirischen Mittelwert eines Fensters und n_i die Länge des Teilfensters W_i. Für jedes neue x_t werden alte x_i vom Ende des Fensters vergessen, bis für alle möglichen Teilfenster $|\hat{\mu}w_0 - \hat{\mu}w_1| \leq \epsilon_{cut}$ gilt. Dieser Ablauf ist in Algorithmus 3 skizziert. ϵ_{cut} soll nun so gewählt werden, dass der Fehler 1. Art durch $\delta \in (0,1)$ begrenzt ist. Das heißt, die Wahrscheinlichkeit mit der ADWIN das Fenster verkleinert, obwohl μ_t in diesem Fenster konstant bleibt, soll höchstens δ betragen.

Um ϵ_{cut} zu finden, wird zunächst die Nullhypothese $\mu_w = \mu_{w_0} = \mu_{w_1}$ angenommen. Das heißt, im betrachteten Fenster liegt kein Strukturbruch vor und $E(X_t)$ bleibt konstant. ϵ_{cut}

```
Initialisiere Fenster W
foreach t > 0 do
    // x_t zu Fenster hinzufügen
    W ← W ∪ {x_t}
    repeat
        ältestes Element aus Fenster W entfernen
    until |μ̂w_0 − μ̂w_1| ≤ ε_cut, ∀ Aufteilungen W = W_0 ∘ W_1
end
```

Algorithmus 3 : ADWIN [3]

muss nun so gewählt werden, dass ADWIN das Fenster höchstens mit Wahrscheinlichkeit δ verkleinert:

$$P(\bigcup_{w_0 \cdot w_1 = w} \{|\hat{\mu}_{w_1} - \hat{\mu}_{w_0}| \geq \epsilon_{cut}\}) \leq \delta.$$

Da höchstens n Teilfenster gebildet werden können, folgt daraus die Bedingung:

$$P(|\hat{\mu}_{w_1} - \hat{\mu}_{w_0}| \geq \epsilon_{cut}) \leq \frac{\delta}{n}. \tag{2.6}$$

Wenn die Ungleichung $|\hat{\mu}_{w1} - \hat{\mu}_{w2}| \geq \epsilon_{cut}$ gilt, gilt für jedes $k \in (0, 1)$ mindestens eine der beiden Bedingungen:

$$|\hat{\mu}_{w_1} - \mu_w| \geq k\epsilon_{cut}$$

$$|\mu_w - \hat{\mu}_{w_0}| \geq (1-k)\epsilon_{cut}.$$

Daher folgt:

$$P(|\hat{\mu}_{w_1} - \hat{\mu}_{w_0}| \geq \epsilon_{cut}) \leq P(|\hat{\mu}_{w_1} - \mu_w| \geq k\epsilon_{cut}) + P(|\mu_w - \hat{\mu}_{w_0}| \geq (1-k)\epsilon_{cut}).$$

Nach Anwendung der Hoeffding-Ungleichung erhält man:

$$P(|\hat{\mu}_{w_1} - \hat{\mu}_{w_0}| \geq \epsilon_{cut}) \leq 2\exp(-2((1-k)\epsilon_{cut})^2 n_0) + 2\exp(-2(k\epsilon_{cut})^2 n_1).$$

Um die Summe zu vereinfachen, wird k so gewählt, dass

$$((1-k)\epsilon_{cut})^2 n_0 = (k\epsilon_{cut})^2 n_1,$$

also

$$k = \frac{\sqrt{\frac{n_1}{n_0}}}{(1 + \sqrt{\frac{n_1}{n_0}})}.$$

Nach Einsetzen ergibt sich:

$$(k\epsilon_{cut})^2 n_0 = \frac{n_1 n_0}{(\sqrt{n_0} + \sqrt{n_1})}\epsilon_{cut}^2 \leq \frac{n_1 n_0}{(n_0 + n_1)}\epsilon_{cut}^2 = m\epsilon_{cut}^2.$$

m ist das harmonische Mittel der Fenstergrößen. Um Bedingung (2.6) zu erfüllen, reicht es aus, die Bedingung

$$4\exp(-2m\epsilon_{cut}^2) \leq \frac{\delta}{n}$$

zu erfüllen. Diese Bedingung wird erfüllt durch:

$$\epsilon_{cut} = \sqrt{\frac{1}{2m}\ln\frac{4n}{\delta}}.$$

Da sich die Verteilung des Mittelwerts für ausreichend große Fenster ($n > 30$) oft der Normalverteilung nähert, ist dieser Wert für ϵ_{cut} in der Praxis meistens zu konservativ. Es wird deshalb

$$\epsilon_{cut} = \sqrt{\frac{2}{m}\sigma_W^2\ln\frac{2}{\delta'}} + \frac{2}{3m}\ln\frac{2}{\delta'}$$

für die Implementierungen des Algorithmuses gewählt, wobei σ_W^2 die empirische Varianz der Elemente im Fenster bezeichnet. Der Wurzelterm ergibt sich aus der Überlegung $\epsilon_{cut} = k\sigma_W^2$ zu wählen und k so zu wählen, dass das Fenster höchstens mit Wahrscheinlichkeit δ verkleinert wird. Der zweite Summand soll die Fälle abdecken, in denen das Fenster zu klein für die Annahme einer Normalverteilung ist. Er wird aus der Bernstein-Ungleichung abgeleitet. Weiterhin argumentieren die Autoren, dass die Wahl von $\delta' = \frac{\delta}{\ln n}$ ausreichend ist [3].

2.3 Vorverarbeitung

Um die Entdeckung von Strukturbrüchen zu verbessern, können unterschiedliche Vorverarbeitungsschritte verwendet werden. Nachfolgend sollen einige dieser Methoden vorgestellt werden. In Kapitel 3 werden diese dann auf die Teleskopdaten angewendet und im Kontext dieser Daten verglichen.

2.3.1 Normalisierung

Damit verschiedene Zeitreihen miteinander verglichen werden können, ist eine Normalisierung erforderlich. Häufig werden die Zufallsvariablen deshalb standardisiert. Dazu sollen die Zeitreihen so transformiert werden, dass der Erwartungswert der resultierenden Zufallsvariablen null ist und die Standardabweichung eins ist. Für eine Zufallsvariable X mit Erwartungswert μ und Standardabweichung σ erhält man die standardisierte Zufallsvariable Z durch

$$Z = \frac{X - \mu}{\sigma}.$$

Da die Werte für μ und σ meist unbekannt sind, werden sie oftmals durch das arithmetische Mittel sowie die empirische Standardabweichung ersetzt. Dies stellt bei den hier behandelten Onlineverfahren ein Problem dar, da zu einem Zeitpunkt nur eine einzelne Realisierung der Zufallsvariable (bzw. ein begrenztes Fenster der letzten Realisierungen)

vorliegt. Daraus ergibt sich das Problem, dass zukünftige Messungen nicht berücksichtigt werden können, und es so zu einer Verfälschung der Standardisierung kommen kann.

Onlinenormalisierung

Es sollen nun drei Ansätze zur Onlinestandardisierung genauer betrachtet werden. Der erste Ansatz speichert zwei Statistiken: die Summe aller bereits gesehenen Werte und die Summe der Quadrate dieser Werte. Mithilfe dieser Werte werden Erwartungswert und Standardabweichung geschätzt, um die Werte auf oben genannte Weise zu normalisieren. Solange sich Erwartungswert und Standardabweichung der Beobachtungen nicht ändern, nähert sich diese Schätzung den optimalen Schätzwerten an, die erreicht werden, wenn alle Daten bekannt sind.

Da die beiden gespeicherten Statistiken mit zunehmender Beobachtungsanzahl steigen, lässt sich dieses Verfahren nicht für Datenströme beliebiger Länge verwenden. Wenn einer der Werte zu groß wird, kommt es zu Fehlern in der Berechnung. Aus diesem Grund wird im zweiten Verfahren ein Fenster der letzten k Werte genutzt, um den Erwartungswert und die Standardabweichung zu schätzen. Dadurch benötigt dieses Verfahren zunächst mehr Speicher, da das gesamte Fenster vorgehalten werden muss. Dieser Speicherbedarf bleibt im Gegensatz zum ersten Verfahren auch mit zunehmender Datenstromlänge konstant.

Um ein Ergebnis zu erhalten, dass der Verwendung aller vergangenen Werte zur Schätzung stärker ähnelt, verwendet der letzte Ansatz *Reservoir Sampling* [14] zur Bildung des Fensters, das zur Schätzung des Erwartungswerts und der Varianz benötigt wird. Das Ziel des *Reservoir Sampling* ist es eine Stichprobe so aus allen auftretenden Daten auszuwählen, dass jedes Datum mit gleicher Wahrscheinlichkeit ausgewählt werden kann. Dazu werden neue Werte in das Fenster aufgenommen, bis die maximale Fenstergröße erreicht ist. Sobald ein neuer Messwert eintrifft, wird dieser mit Wahrscheinlichkeit $\frac{k}{n}$ in das Fenster aufgenommen, n bezeichnet die Anzahl der bisher aufgetretenen Daten und k die Größe des Fensters. Wenn der Wert aufgenommen wird, wird ein zufälliger Wert aus dem Fenster entfernt [14]. Durch die Verwendung von *Reservoir Sampling* soll gewährleistet werden, dass auch Beobachtungen, die länger als k Zeitpunkte zurückliegen, Einfluss auf die Schätzwerte haben. Ist dies nicht der Fall, können allmähliche Änderungen der Zeitreihe durch die Normalisierung verdeckt werden. Vor allem inkrementelle Strukturbrüche können dann nicht explizit gefunden werden. Diese Eigenschaft wird in Kapitel 3 genauer untersucht.

2.3.2 Filterung

Um das Rauschen der Messdaten zu reduzieren, können verschiedene Filter und Glättungsverfahren angewendet werden. Zwei häufig verwendete Verfahren werden genauer betrachtet. Zusätzlich werden die Filterresiduen untersucht.

Exponentielle Glättung

Die *exponentielle Glättung* ist ein Verfahren, das zu jedem Zeitpunkt einen Schätzwert f_t errechnet, der sowohl den derzeitigen Messwert als auch den vergangenen Schätzwert berücksichtigt:

$$f_t = \lambda x_t + (1 - \lambda) f_{t-1}.$$

Der Faktor λ (mit $0 \leq \lambda \leq 1$) bestimmt dabei die Gewichtung des aktuellen Messwerts. Dieses Verfahren bietet sich für die Onlineverarbeitung an, da nur eine Statistik und der aktuelle Messwert benötigt werden.

Gleitender Mittelwert

Ein weiteres Verfahren ist der *gleitende Mittelwert*. Dabei wird ein Fenster über die Daten geschoben. Die Mittelwerte der überlappenden Fenster bilden die gefilterten Daten

$$f_t = \frac{1}{k} \sum_{i=t-(k-1)}^{t} x_i.$$

Da der Wert zum Zeitpunkt t durch den Mittelwert der letzten k Werte ersetzt wird, ergibt sich eine implizite Verzögerung des Datenstroms. Deshalb wird häufig ein symmetrisches Fenster gewählt. Dabei wird der Wert zum Zeitpunkt in der Mitte des Fensters durch den Mittelwert ersetzt. Das heißt bei einem Fenster der Größe 5 ergibt sich

$$f_t = \frac{1}{5}(x_{t-2} + x_{t-1} + x_t + x_{t+1} + x_{t+2}).$$

Bei der Online-Verarbeitung des Datenstroms ergibt sich für ein solches Fenster das Problem, dass der Wert für den Zeitpunkt t erst zum Zeitpunkt $t + \lfloor \frac{k}{2} \rfloor$ bestimmt werden kann, da die zukünfitgen Werte $x_{t+1}, \ldots, x_{t+\lfloor \frac{k}{2} \rfloor}$ für die Berechnung beötigt werden.

Filterresiduen

Neben der Betrachtung der gefilterten Daten bietet sich die Untersuchung der Filterresiduen an. Da die vorgestellten Filterverfahren kurzfristige Änderungen aus den Daten entfernen, stellen die gefilterten Daten die lang- und mittelfristige (abhängig von den gewählten Parametern) Entwicklung der Daten dar und modellieren die zu Grunde liegende Verteilung. Die Filterresiduen

$$r_t = |x_t - f_t|$$

sollten deshalb keine auffälligen Muster enthalten und annähernd normalverteilt sein. Diese Eigenschaft kann mithilfe der vorgestellten Verfahren zur Strukturbrucherkennung überprüft werden.

2.3.3 Repräsentation der Daten

Um bestimmte Eigenschaften einer Zeitreihe zu untersuchen und zu modellieren, kann es hilfreich sein die Repräsentation der Daten zu ändern. Durch solche Transformationen können neue Informationen, die das Lernergebnis verbessern, gewonnen werden, da Eigenschaften der Daten sichtbar werden, die in der vorherigen Repräsentation nicht (oder nur sehr schlecht) identifiziert werden konnten. Eine mögliche Transformation wird im Folgenden genauer untersucht.

Phase Space Embedding

In [12] verwenden die Autoren Phase Space Embedding zur Vorverarbeitung der Daten. Die Zeitreihe $x_1, x_2, \ldots, x_n$ wird zunächst in einen m-dimensionalen, euklidischen Raum (den sogenannten Phasenraum) transformiert. Dazu werden die folgenden Vektoren konstruiert:

$$\vec{x}_t = (x_t, x_{t+1}, \ldots, x_{t+(m-1)})^T \in \mathbb{R}^m,$$

mit $m \in \mathbb{N} \setminus 0$, $t = 1, 2, \ldots, n - (m-1)$ und $m \ll n$.

Die Autoren verweisen auf verschiedene Methoden, um m zu wählen. Im Folgenden wurde $m = 2$ gewählt, um das Verfahren im Kontext der Fragestellung zu testen und den Rechenaufwand dabei gering zu halten, damit verschiedene Kombinationen getestet werden können. Die Vektoren können in diesem Fall im zweidimensionalen Raum geplottet werden. Dies ist in Abbildung 2.3 dargestellt. Die auf diese Weise entstehende Vektorwolke wird *Phase Space Embedding* genannt [12]. Für einen stationären Gaußprozess ergibt sich eine elliptische Wolke, deren Lage und Größe vom Erwartungsvektor $\vec{\mu}$, sowie der Kovarianzmatrix Σ der Vektoren $(X_t, X_{t+1})^T$ abhängen. Um eine Abweichung von der stationären

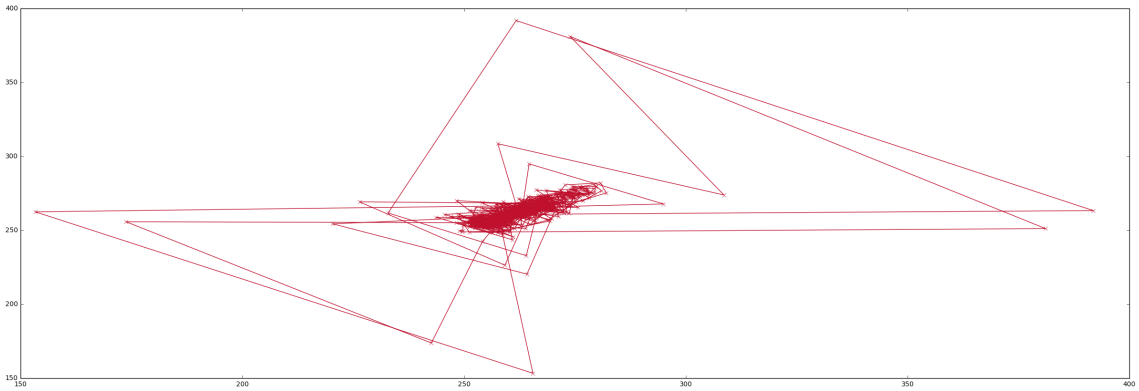


Abbildung 2.3: *Phase Space Embedding* für Gain von Pixel 0.

Verteilung festzustellen, kann der Abstand des Vektors $(x_t, x_{t+1})^T$ vom Erwartungsvektor $\vec{\mu}$ bestimmt werden. Dazu wird die Mahalanobis-Distanz bestimmt:

$$MD_t = \sqrt{(\vec{x_t} - \vec{\mu})^T \Sigma^{-1} (\vec{x_t} - \vec{\mu})}$$

In der Praxis werden die unbekannten Werte $\vec{\mu}$ und Σ durch geeignete Schätzer ersetzt. In den Experimenten wurden dazu die ersten 50 Werte des Datenstroms verwendet.Für weiterführende Experimente ist es ebenfalls denkbar, diese Schätzer zu bestimmten Zeitpunkten zu aktualisieren (z. B. nach einem Strukturbruch).

Kapitel 3

Experimenteller Vergleich der Methoden

Im Folgenden sollen die durchgeführten Experimente strukturiert beschrieben und ausgewertet werden. Zur Bewertung der unterschiedlichen Methoden zur Strukturbrucherkennung werden in [5] die Gütemaße *Mean Time between False Alarms (MTFA)*, *Mean Time to Detection (MTD)*, *Missed Detection Rate (MDR)* und *Average Run Length* beschrieben. Dabei beschreibt *MTFA* die durchschnittliche Zeit zwischen Fehlalarmen, *MTD* die durchschnittliche Zeit bis zur Detektion eines Strukturbruchs und *MDR* die Wahrscheinlichkeit einen Strukturbruch nicht zu erkennen. Mit der *ARL* wird die erwartete Zeit bis zu einem Strukturbruchalarm beschrieben. Sie kann als Generalisierung von *MTFA* und *MTD* interpretiert werden, da sowohl positive als auch falsch-positive Alarme gezählt werden. Die Verwendung dieser Werte bietet sich in diesem Fall jedoch nicht an, da keine genauen Informationen über das zeitliche Auftreten von Inhomogenitäten vorliegen. Deshalb erfolgt anstelle eines quantitaven Vergleichs, eine qualitative Auswerugn durch Visualisierung der Verfahren und Rücksprache mit Domänenexperten. Dazu ist zunächst eine genaue Betrachtung der Daten hilfreich. Hierbei sollen auffällige Stellen in den Daten identifiziert werden, die für die Identifikation von Defekten und fehlerhafter Kalibrierung interessant sind. Zur Identifikation dieser Stellen sollen auch vorhandene Notizen über Ausfälle und ungewöhnliches Verhalten der Pixel verwendet werden, die während des Betriebs des FACT erstellt wurden. Durch diese Untersuchung der Daten kann zudem die Auswahl an Vorverarbeitungsverfahren und Modellen eingegrenzt werden.

3.1 Daten

Bei den verwendeten Daten handelt es sich um den Gain der Pixel des Teleskops. Diese Daten werden etwa zwei bis drei mal am Tag erfasst. Diese Messungen erfolgen meist zu Beginn und am Ende der Nacht. Insgesamt besteht der Datensatz aus 2389 Einträgen für

jedes der 1440 Pixel. Die vorliegenden Daten umfassen den Zeitraum vom 02.01.2013 bis zum 01.11.2017. Aus dem Datensatz wurden bereits einige bekannte, fehlerhafte Messwerte entfernt. Hier ist besonders eine 2014 durchgeführte Kalibrierung des Teleskops zu erwähnen, die zu fehlenden Messwerten in diesem Zeitraum führt. Daneben gibt es weitere Lücken in den Messungen, die bei der Betrachtung berücksichtigt werden müssen.

Die Pixel 0, 171, 775 und 1165 werden nachfolgend betrachtet. Die Pixel 171, 775 und 1165 wurden ausgewählt, da sie auffälliges Verhalten zeigen. Pixel 0 wurde gewählt, da es keine offensichtlichen Auffälligkeiten aufweist. In Abbildung 3.1 sind die Daten dargestellt

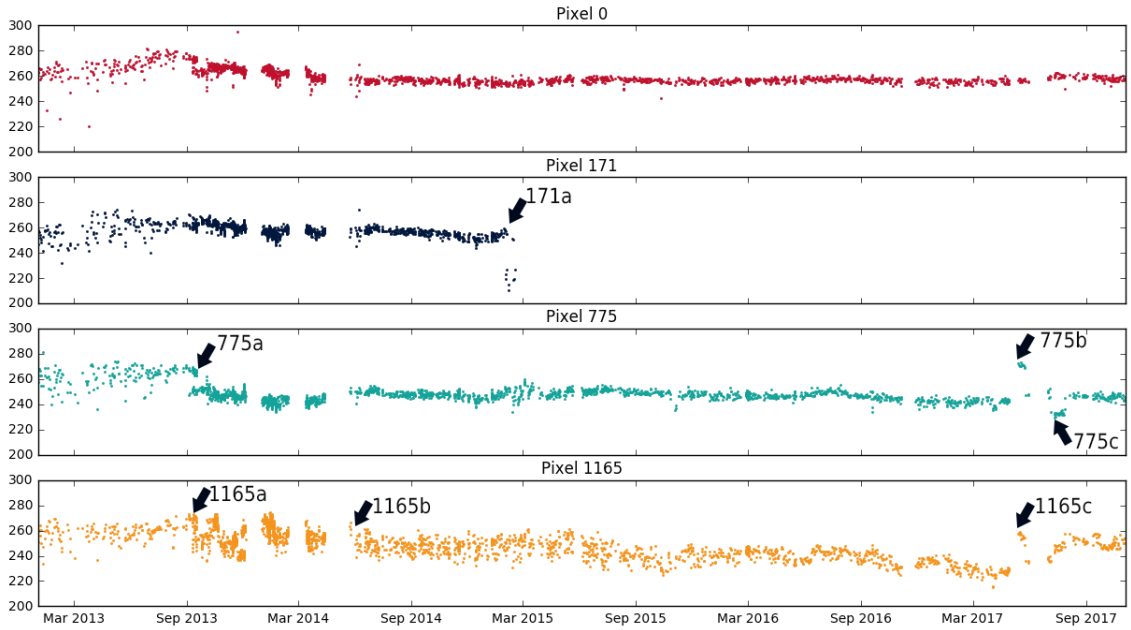


Abbildung 3.1: Zeitlicher Verlauf der Pixel 0, 171, 775, 1165 vom 02.01.2013 bis zum 03.11.2017.

und auffällige Stellen sind markiert. Deutlich zu erkennen ist der Ausfall von Pixel 171 im Februar 2015 (171a). Nach dem Ausfall liefert der Sensor konstant den Wert 0. Ebenfalls auffällig ist der Niveauabfall im September 2013 von Pixel 775 (775a), sowie der Anstieg (775b, 1165c) und Abfall (775c) zwischen Mai und August 2017 bei Pixeln 775 und 1165. Auch die fehlenden Messwerte in den Jahren 2014, 2017 und zum Ende des Jahres 2016 sind zu erkennen. Zuletzt fällt auf, dass das Signal von Pixel 1165 deutlich stärker streut als das Signal der anderen Pixel. Dies lässt sich besonders nach den Stellen 1165a und 1165b erkennen. Die markierten Stellen sollten von den Verfahren mindestens erkannt werden. Wenn weitere Alarme ausgegeben werden, müssen diese genauer untersucht und bewertet werden. Dabei soll darauf geachtet werden, dass nicht zu viele Warnungen ausgegeben werden. Dies gilt besonders für Pixel 0, da hier keine Inhomogenitäten vermutet werden.

3.2 Frameworks

Die zur Durchführung der Vergleiche verwendeten Frameworks werden nachfolgend genauer beschrieben. Diese Frameworks erleichtern die Onlineverarbeitung von Datenströmen. Dabei liegen im Gegensatz zum Batchlernen nicht alle Daten gleichzeitig vor. Stattdessen steht zu jedem Zeitpunkt nur der aktuelle Datenpunkt (bzw. ein Fenster der letzten Datenpunkte) zur Verfügung. Es wird also ein kontinuierlicher Datenstrom aus Datenobjekten verarbeitet. Durch dieses Vorgehen ist das Datamining von großen Datensätzen mit begrenzten Ressourcen möglich. Onlineverfahren bieten sich im hier betrachteten Fall an, da täglich neue Messungen durchgeführt werden.

3.2.1 Streams

Durch die Verwendung des *streams*-Framework [7] wird eine Abstraktion der Onlineprozesse erreicht. Ein solcher Prozess wird im Framework als Datenflussgraph beschrieben, bei dem jeder Knoten eine Funktion darstellt, die auf die Daten angewendet wird. Die Kanten beschreiben den Datenfluss zwischen diesen Funktionen. Dieser Datenfluss wird in einem XML-Dokument beschrieben und ermöglicht die schnelle und flexible Kombination unterschiedlicher Funktionen.

Der auf diese Weise beschriebene Graph besteht hauptsächlich aus Datenstrom- (*stream*), Warteschlangen- (*queues*) und Prozesselementen (*process*). Daneben stehen zusätzliche Kontrollflusselemente zur Verfügung. Ein beispielhafter Graph ist in Abbildung 3.2 dargestellt.

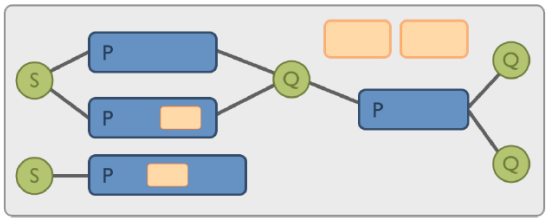


Abbildung 3.2: Beispielhafter Datenflussgraph mit Datenstromelementen (S), Prozesselementen (P) und Warteschlangen (Q) [7].

Datenelemente bestehen aus Schlüssel-Wert-Paaren. Ein Datenstromobjekt ermöglicht den Zugriff auf diese Objekte und dient somit als Quelle für Datenelemente. Dies ist in Abbildung 3.3 dargestellt. Die in den Experimenten verwendete Instanz eines Datenstroms liest eine csv-Datei und stellt die entsprechenden Daten im Framework zur Verfügung.

Die auf diese Weise zur Verfügung gestellten Daten können von einem Prozess verbraucht werden. Die Prozesse sind die funktionalen Elemente. Sie beinhalten Prozessoren (*processor*), die eine Funktion darstellen, die auf die Daten angewendet werden kann und ein Datenelement zurückgeben. Der Prozess wendet die Prozessoren nacheinander auf die

Abbildung 3.3: Darstellung von Datenelementen und einem Datenstromobjekt [7].

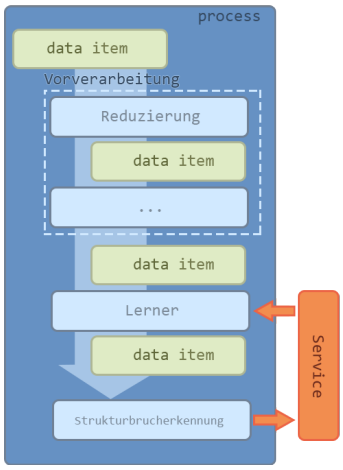


Abbildung 3.4: Darstellung eines beispielhaften Prozesses zur Entdeckung der Inhomogenitäten in den Teleskopdaten und der Kommunikation über Services. Darstellung nach [7].

erhaltenen Daten an. Dabei dient die Ausgabe des vorangegangenen Prozessors als Eingabe für den nächsten.

Zur Kommunikation zwischen Prozessen können Warteschlangen verwendet werden, die eine begrenzte Anzahl von Elementen speichern können. Sie stellen außerdem die Funktionalität eines Datenstroms bereit und können so als Eingabe eines anderen Prozesses dienen.

Um Informationen asynchron zum Datenfluss zur Verfügung zu stellen, können diese von Prozessoren als Service bereitgestellt werden. Strukturbruchdetektoren können beispielsweise ihren Zustand (Alarm oder Warnung liegt vor) bereitstellen.

In Abbildung 3.4 ist ein beispielhafter Prozess für die durchgeführten Versuche dargestellt. Im Prozess werden verschiedene Vorverarbeitungsschritte als Prozessoren aufgerufen. Anschließend werden die Datenelemente an einen Lerner weitergereicht und schlussendlich an den Strukturbrucherkenner übergeben. Neben dem Prozess ist die Kommunikation zwischen Strukturbrucherkenner und Lerner als Service dargestellt. So kann ein gelerntes Modell verworfen werden, sobald ein Strukturbruch erkannt wurde.

Zusätzlich zu den vorhandenen Prozessoren stellt das Framework eine API zur Verfügung, um eigene Prozessoren zu erstellen. Dazu muss das *Processor*-Interface implementiert werden, das eine Methode definiert, die vom Prozess für jedes Datenelement aufgerufen wird. Auf dieser Grundlage wurden im Rahmen dieser Arbeit Prozessoren für exponentielle Glättung, gleitenden Mittelwert, Onlinenormalisierung, Phase Space Embedding und weitere Vorverarbeitungsschritte erstellt. Daneben wurden einige Prozessoren erstellt, die unterschiedliche (technische) Hilfsfunktionen, wie die Erzeugung eines Schlüssel-Wert-Paares für erkannte Strukturbrüche, umsetzen. Diese Prozessoren stehen im Git-Repository [13] zur Verfügung. Außerdem stehen, durch die Schnittstelle *streams-moa* [17], Klassifikatoren aus dem MOA-Framework zur Verfügung, das im folgenden Abschnitt ausführlicher beschrieben werden soll.

In Abbildung 3.5 ist die XML-Beschreibung eines Prozesses dargestellt. Neben den vorgestellten Elementen (*process, stream, processor*) ist die Definition von *properties* zu erkennen. Diese können als Parameter für Prozessoren verwendet werden. Zusätzlich können diese beim Aufruf des Programms als Kommandozeilenparameter übergeben werden.

```
<property name="pixel" value="775"/>
<property name="lambda" value="50"/>
<property name="delta" value="0.05"/>
<stream header="true" id="input" class="stream.io.CsvStream" url="file:///c:/Users/nilsk/Documents/Bachelorarbeit/data_csv/Gain_check.csv"/>
<process id="process" input="input">
    <SelectKeys keys="${pixel}, fRunStart"/>
    <de.killich.TimebasedReduction valueKey="${pixel}" dateKey="fRunStart" aggregationPeriod="P1D"/>
    <de.killich.ReservoirNorm windowLength="200" inputKey="${pixel}"/>
    <de.killich.filter.ExponentialSmoothing lambda="0.3" inputKey="@norm" outputKey="filtered"/>
    <de.killich.PhaseSpaceEmbedding inputKey="filtered" windowLength="50"/>

    <moa.classifiers.core.driftdetection.CusumDM id="driftDetection" delta="${delta}" lambda="${lambda}"/>
    <de.killich.ChangeKeyGenerator driftDetectionService="driftDetection"/>

    <stream.io.CsvWriter url="file:///c:/Users/nilsk/Documents/Bachelorarbeit/experimente/b_cusum${pixel}.csv"/>
</process>
```

```
\ba>java -Dlambda=50 -Ddelta=0.05 -Dpixel=775 -jar target\nils-killich-ba.jar examples\read02.xml
```

Abbildung 3.5: Datenflussgraph als XML-Datei und Aufruf für die Kombination: Reduzierung → Normalisierung → exponentielle Glättung → Phase Space Embedding → CUSUM

3.2.2 Massive Online Analysis (MOA)

Massive Online Analysis [4] ist eine Softwareumgebung, die Werkzeuge zur Onlineuntersuchung von Datenströmen zur Verfügung stellt. Sie enthält Klassifikatoren für Datenströme, Clustering, Verfahren zur Strukturbrucherkennung und Methoden zum Vergleich verschiedener Klassifikatoren. Für diese Arbeit ist vor allem das Modul zur Strukturbrucherkennung (MOA-CD) von Interesse. Es ermöglicht die Verwendung von ADWIN, Page-Hinkley-Test, CUSUM, DDM, EDDM und ECCD. Diese können dann durch die Schnittstelle *streams-moa* im Streams-Framework als Prozessoren, Services und Datenströme verwendet werden. Dies erlaubt die schnelle und einfache Kombination der verschiedenen Verfahren zur Vorverarbeitung der Daten im Framework. Dadurch können viele Kombinationen effizient im Kontext der Teleskopdaten getestet und verglichen werden.

3.3 Vorverarbeitung

Zunächst werden verschiedene Ansätze zur Vorverarbeitung genauer betrachtet und verglichen. Daraufhin werden diese Verfahren mit Strukturbrucherkennern kombiniert um zu untersuchen, welchen Einfluss die Darstellung und Vorverarbeitung der Daten auf die Leistungsfähigkeit der Verfahren hat.

3.3.1 Äquidistanz und fehlende Daten

Für viele der Verarbeitungsschritte ist es erforderlich (oder zumindest hilfreich), dass die betrachteten Daten einen konstanten zeitlichen Abstand zueinander haben. Da die verwendeten Verfahren äquidistante Daten annehmen, würden andernfalls Bereiche mit einer hohen Messwertdichte stärker gewichtet als solche mit wenigen Messwerten. Bei den Teleskopdaten bietet sich der Abstand von einem Tag an. Dazu müssen Messwerte eines Tages zu einem Wert kombiniert werden. Fehlende Messwerte müssen ergänzt werden.

Um die Messwerte auf einen Wert pro Tag zu reduzieren, werden die Daten eines Tages durch ihr arithmetisches Mittel ersetzt. Alternativ ist die Ersetzung durch den Median denkbar, um eine stärkere Robustheit gegenüber Ausreißern zu erreichen. Die Ergänzung fehlender Werte ist weniger trivial und muss noch genauer betrachtet werden. In den weiteren Experimenten werden die tatsächlichen zeitlichen Abstände nach der beschriebenen Reduzierung ignoriert. Der Abstand der Daten wird dann als äquidistant interpretiert. Trotz dieser Vorgehensweise liefern die folgenden Untersuchungen gute Ergebnisse, sodass in diesem Rahmen zunächst auf anspruchsvollere Methoden zum Umgang mit fehlenden Daten verzichtet wurde. In Abbildung 3.6 ist dieser Vorverarbeitungsschritt dargestellt.

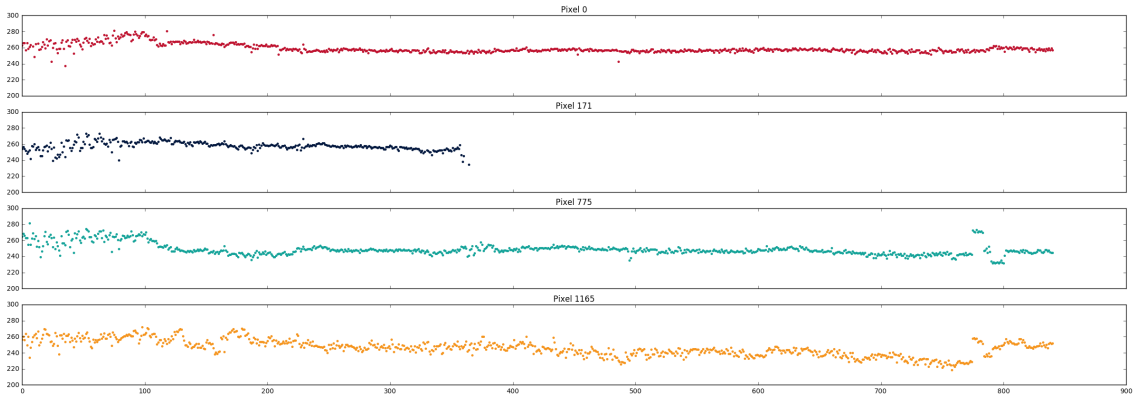


Abbildung 3.6: Messwerte der Pixel 0, 171, 775 und 1165 nach Reduzierung auf einen Messwert pro Tag.

Die Messwerte der Pixel wurden auf einen Datenpunkt pro Tag reduziert und chronologisch nummeriert. Die Daten wurden anhand dieser Nummerierung und unabhängig von den tatsächlichen zeitlichen Abständen geplottet.

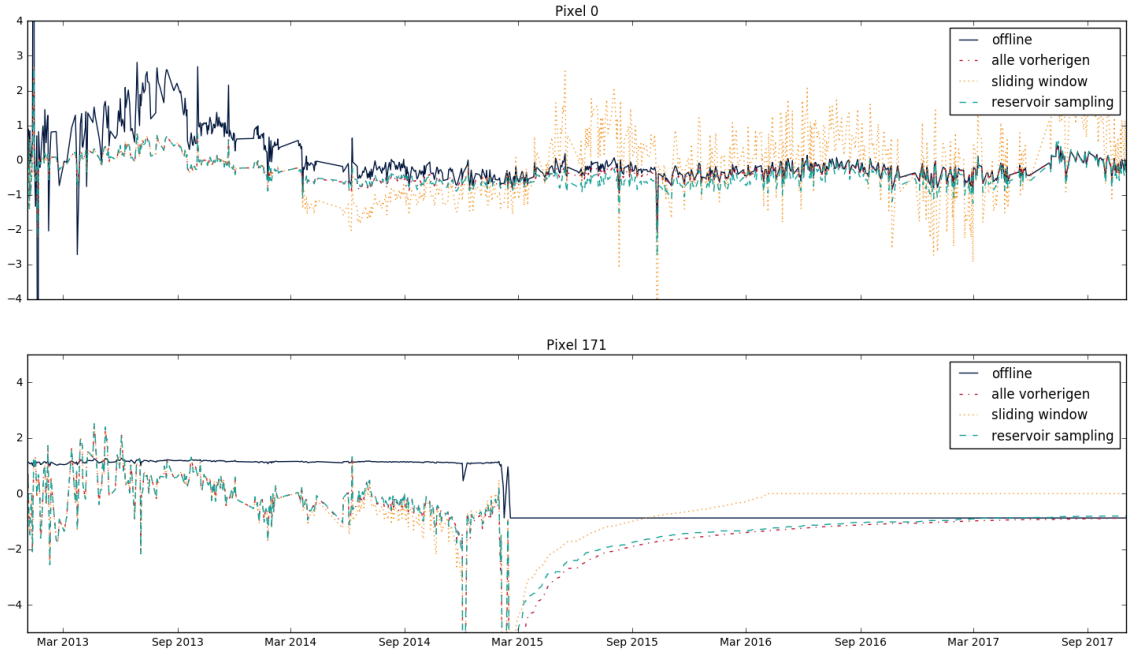


Abbildung 3.7: Vergleich der Normalisierungsmethoden. Als Fenstergröße wurde für Sliding-Window-Normalisierung und Reservoir-Sampling-Normalisierung $k = 200$ gewählt.

3.3.2 Normalisierung

Die in Kapitel 2 vorgestellten Normalisierungsverfahren sollen nun auf die Daten angewendet und verglichen werden. In Abbildung 3.7 sind die Daten nach der Normalisierung mit den beschriebenen Verfahren dargestellt. Wenn alle vorherigen Beobachtungen verwendet werden um den aktuellen Wert zu normalisieren, nähern sich die normalisierten Werte den Werten an, die bei Verwendung des Offlineverfahrens erzeugt werden. Dies ist nicht weiter verwunderlich, da sich die Schätzer beider Verfahren in diesem Fall annähern. Für den letzten Datenpunkt sind die Schätzer identisch, da auch im Offlineverfahren keine weiteren zukünftigen Daten bekannt sind.

Außerdem sieht man, dass die Werte, die durch Sliding-Window-Normalisierung entstehen, bei der gewählten Fenstergröße stark von der optimalen Normalisierung abweichen. Dieser Fehler lässt sich allerdings durch ausreichend große Fenstergrößen beliebig gering halten. Im Gegensatz dazu werden mit Reservoir-Sampling-Normalisierung Werte erreicht, die deutlich weniger von der optimalen Normalisierung abweichen.

Ein weiterer Unterschied zwischen den Verfahren lässt sich nach dem Ausfall von Pixel 171 (171a) erkennen. Bei Verwendung der Sliding-Window-Normalisierung ergibt sich für den normalisierten Wert etwa 0, während sich bei den anderen Verfahren etwa -1 ergibt. Da im Fenster nur die letzten k Werte betrachtet werden, passt sich dieses Verfahren besser an Veränderungen der zu Grunde liegenden Verteilung der Daten an. Diese Eigenschaft könnte sich in diesem Kontext als nachteilig herausstellen, da die explizite Entdeckung solcher

Veränderungen gewünscht ist, um Warnungen auszugeben. Daher wird vor allem für Pixel 171 Reservoir-Sampling-Normalisierung vorgezogen. Bei der Verwendung von Reservoir-Sampling-Normalisierung ist jedoch zu beachten, dass es vor allem bei kleinen Fenstergrößen zu unterschiedlichen Ausgabewerten (bei gleicher Eingabe) kommen kann, da zufällig ausgewählte Werte zur Normalisierung genutzt werden. Trotz dieser Einschränkung wird Reservoir-Sampling-Normalisierung (k=200) für die nachfolgenden Vergleiche verwendet.

3.3.3 Filterung

In Kapitel 2 wurden die exponentielle Glättung und der gleitende Mittelwert beschrieben. Zunächst soll die exponentielle Glättung betrachtet werden.

Exponentielle Glättung

In Abbildung 3.8 sind die geglätteten Messungen für die betrachteten Bildpunkte dargestellt. In der Abbildung sieht man, dass der Ausfall von Pixel 171 (171a) nicht mehr zu erkennen ist, wenn λ zu klein gewählt wird. Auch die Niveauänderungen von Pixel 775 (775b, 775c) und 1165 (1165c) lassen sich bei Wahl von $\lambda = 0.01$ nicht mehr erkennen. Bei Wahl von $\lambda = 0.1$ scheinen viele mittelfristige Veränderungen erhalten zu bleiben, während Rauschen und kurzfristige Änderungen der Messdaten verschwinden.

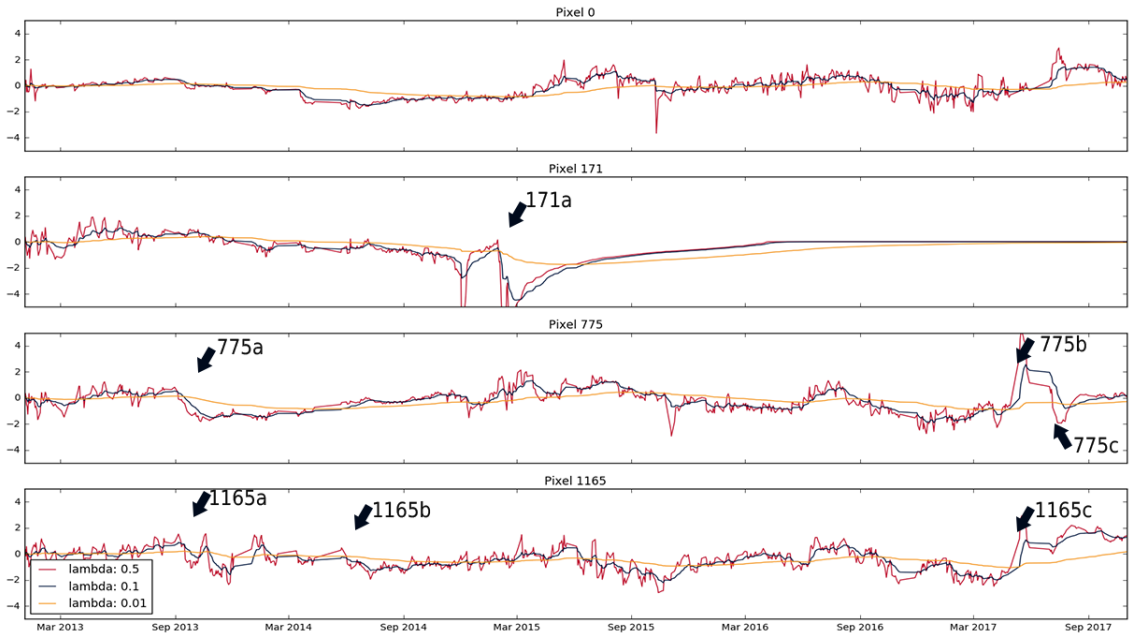


Abbildung 3.8: Exponentielle Glättung für die Pixel 0, 171, 775, 1165 mit verschiedenen Parametern. Zur Vorverarbeitung wurden die Daten reduziert und mit Sliding-Window-Norm (Fenstergröße: 200) normalisiert.

Gleitender Mittelwert

Die Filterung mit dem gleitenden Mittelwert ist in Abbildung 3.9 dargestellt. Zur Bildung des Fensters wurden nur vergangene Werte verwendet, da das Erstellen eines symmetrischen Fensters für die Teleskopdaten bedeuten würde, dass (nach der Vorverarbeitung in Abschnitt 3.3.1) eine mögliche Inhomogenität frühestens nach $\frac{k}{2}$ Tagen erkannt werden kann. An der Abbildung lässt sich erkennen, dass für die Fenstergröße 51 viele interessante

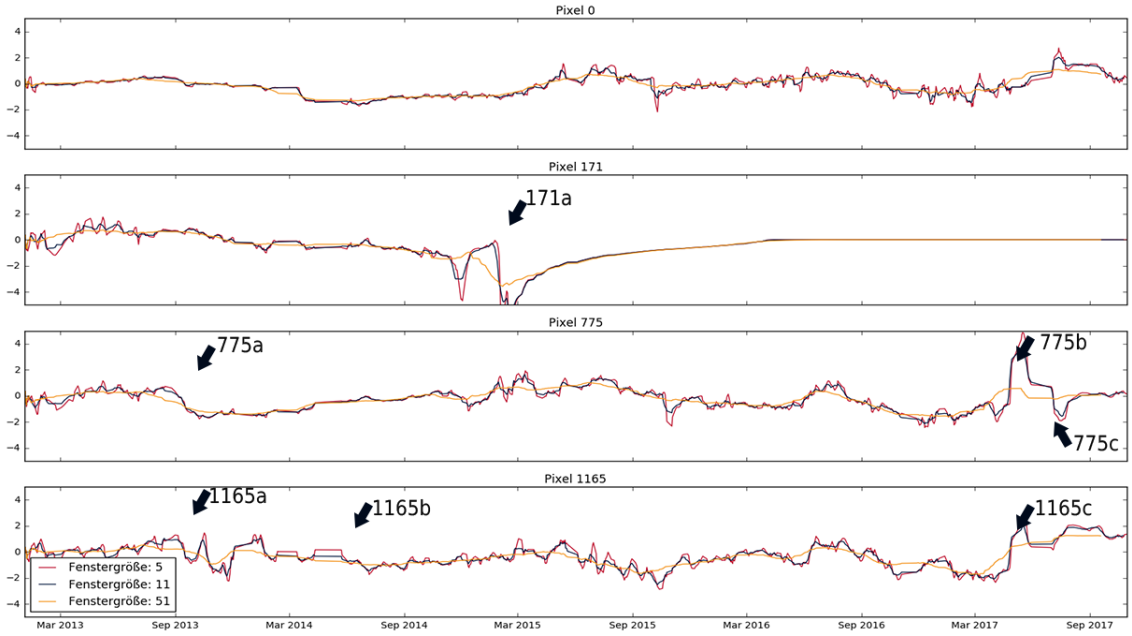


Abbildung 3.9: Moving Average für die Pixel 0, 171, 775 und 1165 mit unterschiedlichen Fenstergrößen und symmetrischen Fenster. Zur Vorverarbeitung wurden die Daten reduziert und mit Sliding-Window-Norm (Fenstergröße: 200) normalisiert.

Stellen verschwinden. Die gefilterten Daten zeigen nur noch sehr langfristige Entwicklungen der Messdaten. Bei Fenstergröße 11 sind hingegen viele der interessanten Stellen noch erhalten, während sehr kurzfristige Änderungen entfernt wurden. Im Vergleich zur exponentiellen Glättung lassen sich keine großen Unterschiede feststellen. Im Folgenden wird deshalb nur eine der Methoden angewendet. Da sich die exponentielle Glättung für Onlineprozesse anbietet, wird sie bevorzugt.

Filterresiduen

Abbildung 3.10 zeigt beispielhafte Residuen bei Verwendung von exponentieller Glättung. Viele der interessanten Stellen lassen sich auch in diesen Abbildungen erkennen. Der Ausfall des Pixels 171 (171a) lässt sich an der starken Abweichung der Werte zum Zeitpunkt des Ausfalls erkennen. Die Niveauänderungen (775b, 775c, 1165c) lassen sich bei der dargestellten Parameterwahl erkennen, während die Niveauänderung 775a nicht zu erkennen ist.

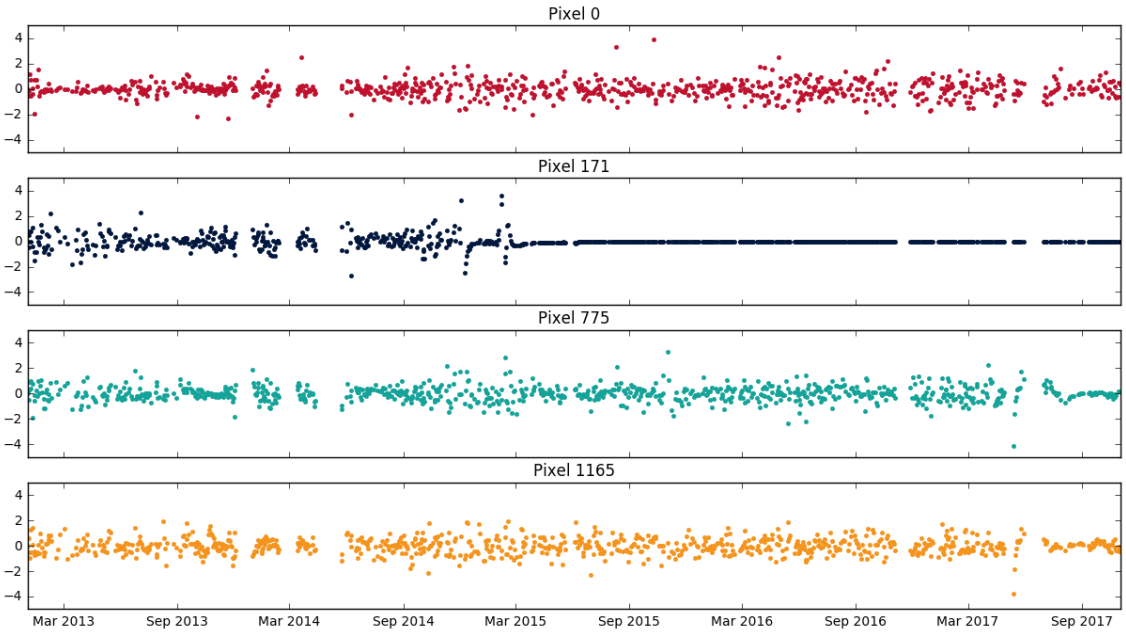


Abbildung 3.10: Filterresiduen bei exponentieller Glättung mit $\lambda = 0.3$. Die Daten wurden auf einen Wert pro Tag reduziert und mit Sliding-Window-Normalisierung normalisiert (Fenstergröße: 200).

3.4 Vergleich der Methoden

Zum Vergleich der Methoden wurden verschiedene Kombinationen der vorgestellten Methoden an den Pixeln getestet. Dazu wurden entsprechende Prozesse im Streams-Framework erstellt und auf die Daten angewendet. Die untersuchten Kombinationen werden der besseren Übersicht halber in Anlehnung an den Datenflussgraphen beschrieben. Eine Auswahl der verwendeten Kombinationen soll im Folgenden genauer beschrieben werden. Diese Kombinationen sind:

1. Reduzierung → Normalisierung → ADWIN / CUSUM / Page-Hinkley-Test
2. Reduzierung → Normalisierung → Phase Space Embedding → ADWIN / CUSUM / Page-Hinkley-Test
3. Reduzierung → Normalisierung → exponentielle Glättung → Phase Space Embedding → CUSUM
4. Reduzierung → Normalisierung → exponentielle Glättung → CUSUM
5. Reduzierung → Normalisierung → exponentielle Glättung (Filterresiduen) → CUSUM
6. Reduzierung → Normalisierung → exponentielle Glättung (Filterresiduen) → DDM / EDDM / ECCD
7. Reduzierung → Normalisierung → Phase Space Embedding → DDM / EDDM / ECCD

Bei Kombination 1 werden die Concept-Drift-Detektoren auf die reduzierten und normalisierten Daten angewendet. Die Verfahren CUSUM und Page-Hinkley-Test werden ein zweites Mal auf die invertierten Daten angewendet, da es sich bei diesen Verfahren um

einseitige Test handelt, die nur positive Abweichungen erkennen. Kombination 2 verwendet zusätzlich die Transformation mittels Phase Space Embedding. Als Eingabe für den Concept-Drift-Detektor dient die Mahalanobis-Distanz vom Erwartungswertvektor. Kombination 3 erweitert die zweite Kombination durch die Verwendung von exponentieller Glättung vor der Transformation mittels Phase Space Embedding. Kombination 4 verwendet ebenfalls exponentielle Glättung, allerdings werden die gefilterten Werte direkt als Eingabe für CUSUM verwendet. Eine Transformation findet nicht statt. Diese Kombination entspricht Kombination 1 mit zusätzlicher Filterung. Bei der fünften und sechsten Kombination werden an Stelle der gefilterten Werte die Filterresiduen als Eingabe für die Concept Drift Detektoren verwendet. Die sechste Kombination unterscheidet sich insofern, als die verwendeten Strukturbrucherkenner einen diskreten Fehlerwert (0, falls Vorhersage korrekt und 1 andernfalls) erwarten. Es wird deshalb geprüft, ob die Residuen zwischen den vorgegebenen Quantilen der Standardnormalverteilung liegen. Dabei wird angenommen, dass die Residuen annähernd normalverteilt sind und viele Werte außerhalb dieser Quantile für einen Strukturbruch sprechen. Der Fehlerwert b_t, der als Eingabe für die Concept-Drift-Detektoren verwendet wird, ist also:

$$b_t = \begin{cases} 0 & ,\text{falls } z_{1-q} < r_t < z_q \\ 1 & ,\text{sonst} \end{cases} .$$

Dabei ist z_q das q-Quantil der Standardnormalverteilung mit $q \geq 0,5$. Kombination 7 verfolgt einen ähnlichen Ansatz. Da die Abstände der Phasenraumvektoren vom Erwartungswertvektor annähernd χ^2_2-verteilt sind [12], wird geprüft ob die Werte zwischen den vorgegebenen Quantilen dieser Verteilung liegen. Der resultierende Fehlerwert wird als Eingabe für die Concept-Drift-Detektoren verwendet. Um den Umfang der dargestellten Kombinationen zu reduzieren, wird für die Kombinationen 3, 4 und 5 nur CUSUM verwendet, da mit ADWIN und PHT schlechtere Ergebnisse erzielt wurden.

3.4.1 Ausgewählte Pixel

Die erkannten Strukturbrüche dieser Kombinationen sind für die untersuchten Pixel im Folgenden dargestellt. Es wurde durch geeignete Parameterwahl versucht ein Verfahren zu finden, das für alle beschriebenen Inhomogenitäten Strukturbrüche feststellt. Daneben sollen die Verfahren nur wenige Alarme ausgeben.

Pixel 775

In Abbildung 3.11 sind die von Kombination 1 erkannten Strukturbrüche zu sehen. In rot ist der Verlauf des Gains von 2013 bis 2017 dargestellt. Die Stelle 775a wird von ADWIN erkannt. An Stelle 775c wird ebenfalls ein Strukturbruch erkannt, der allerdings auch Stelle 775b zugeordnet werden kann, da an dieser Stelle noch kein Alarm ausgegeben

wird. Daneben werden an vielen weiteren Stellen Alarme ausgegeben, die im Vorhinein nicht als interessant identifiziert worden sind. Trotz unterschiedlicher Wahl des Parameters können die Alarme nicht auf offensichtlich auffällige Stellen begrenzt werden. Der Page-Hinkley-Test erkennt keine der identifizierten Stellen und gibt einen unerwarteten Alarm aus. CUSUM erkennt alle interessanten Stellen (775a, 775b und 775c). Wie bei ADWIN werden viele weitere Alarme ausgeben. Bei allen drei verwendeten Concept Drift Detektoren werden zu viele Alarme ausgegeben. Im direkten Vergleich scheint CUSUM die besten Ergebnisse zu liefern, da alle interessanten Stellen erkannt werden. Um gute Ergebnisse zu erreichen, scheinen weitere Vorverarbeitungsschritte notwendig zu sein. Aus diesem Grund wird diese Kombination nicht für die anderen Pixel getestet.

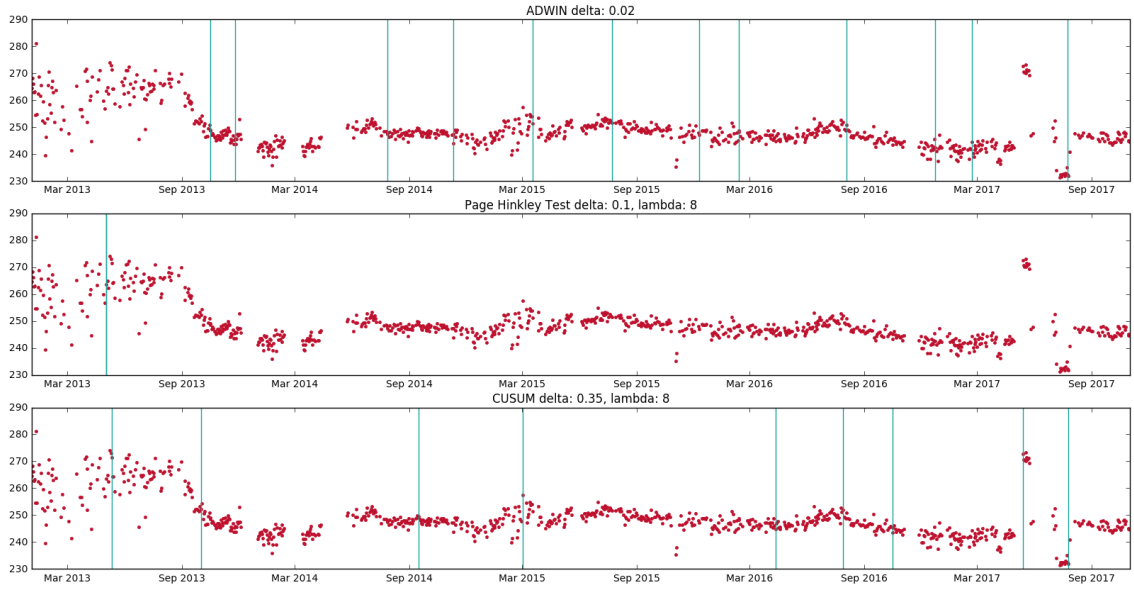


Abbildung 3.11: Kombination 1 für Pixel 775.

Abbildung 3.12 zeigt Kombination 2 für Pixel 775. Neben dem Verlauf des Gains sind die Abstände vom Erwartungswertvektor dargestellt, die als Eingabe für die Concept Drift Detektoren verwendet werden. Die Niveauänderung 775a wird von allen drei Verfahren erkannt. Vom Page-Hinkley-Test werden die Stellen 775b und 775c nicht erkannt. Allerdings werden keine unerwarteten Alarme ausgegeben. CUSUM und ADWIN erkennen im Bereich der Stellen 775b und 775c jeweils nur einen Strukturbruch. ADWIN liefert als einziges Verfahren viele weitere Alarme. Im direkten Vergleich ist CUSUM vorzuziehen, da bei den wenigsten falschen Alarmen die meisten Stellen identifiziert werden. In Abbildung 3.13 ist Kombination 3 dargestellt, bei der die Daten zusätzlich gefiltert werden. Die Ergebnisse sind mit Kombination 2 vergleichbar, allerdings ist die Wahl der Parameter leichter und intuitiver.

Abbildung 3.14 zeigt Kombination 4. Die gefilterten Daten dienen direkt als Eingabe für CUSUM. Im Vergleich zu Kombination 1, in der kein Filter verwendet wurde, lässt

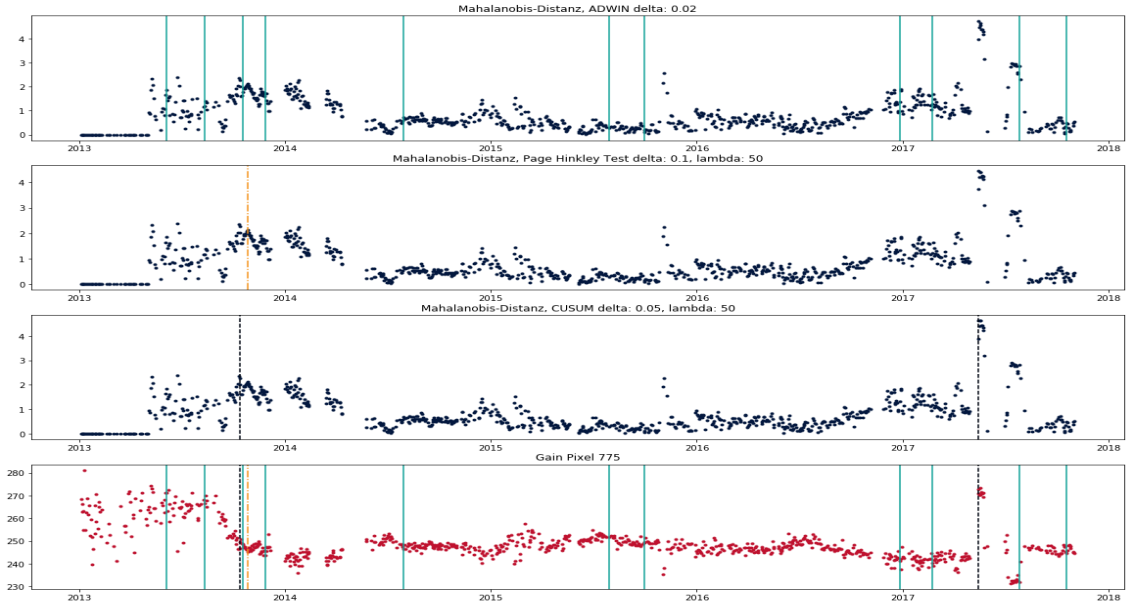


Abbildung 3.12: Kombination 2 für Pixel 775.

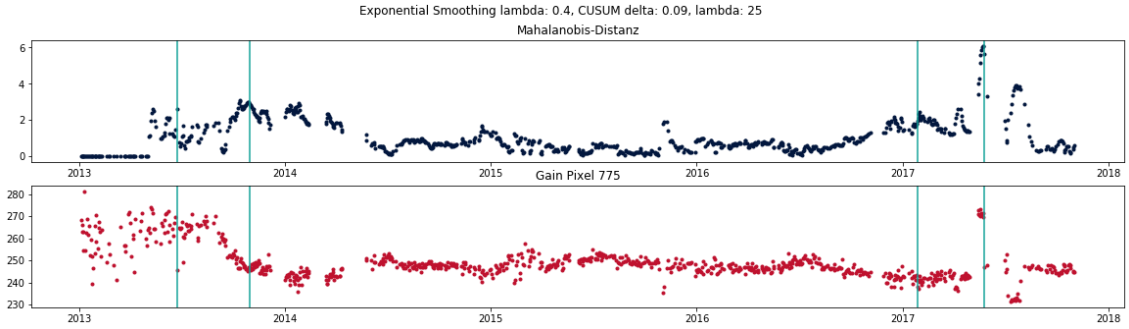


Abbildung 3.13: Kombination 3 für Pixel 775.

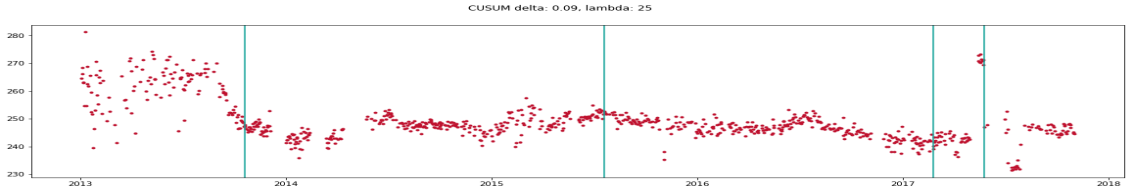


Abbildung 3.14: Kombination 4 für Pixel 775.

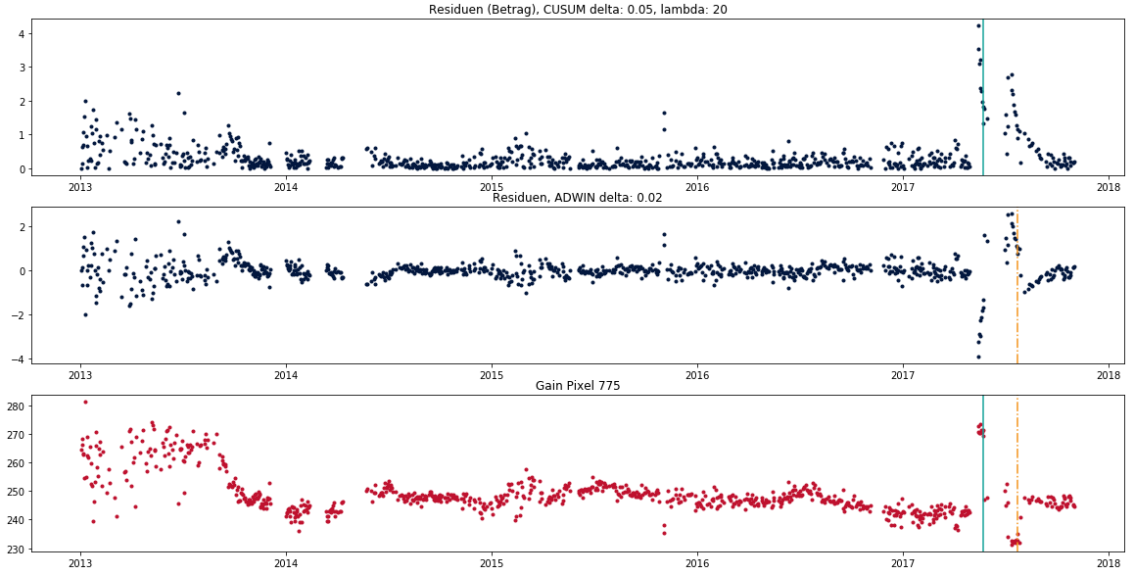


Abbildung 3.15: Kombination 5 für Pixel 775.

sich die Anzahl der falschen Alarme deutlich besser eingrenzen. Die Stellen 775a und 775b werden erkannt, während 775c nicht erkannt wird.

In beiden Kombinationen, die Filterung mit exponentieller Glättung verwenden, ergeben sich vergleichbare Ergebnisse zu den Kombinationen ohne Filterung, wobei sich die Parameter besser einstellen lassen. Durch die Wahl des Parameters des Filters kann entschieden werden, in wie weit kurz- bzw. langfristige Änderungen berücksichtigt werden sollen. Bei Änderung der Parameter von CUSUM lassen sich die Veränderungen der Alarme besser nachvollziehen als bei den vorhergegangenen Ansätzen. Dadurch ist eine genauere Anpassung der Empfindlichkeit möglich.

Kombination 5 ist in Abbildung 3.15 dargestellt. Hier dienen die Filterresiduen als Eingabe für die Concept- Drift-Detektoren. Die Stelle 775a wird von beiden Verfahren nicht erkannt. Die Stelle 775b wird nur von CUSUM erkannt, während 775c nur von ADWIN erkannt wird. In Abbildung 3.16 ist Kombination 6 zu sehen. Zusätzlich zu den Filterresiduen sind die Quantile der Normalverteilung eingezeichnet, die verwendet werden um zu bestimmen ob ein Fehler vorliegt. Bei Verwendung von DDM und EDDM wird ein Alarm im Bereich der Stellen 775b und 775c erkannt. Bei EDDM und ECCD wird ebenfalls die Stelle 775a erkannt. ECCD erkennt zusätzlich Stelle 775b sehr frühzeitig, gibt allerdings auch drei weitere, unerwartete Alarme aus. Im direkten Vergleich liefert ECCD die frühzeitigsten Alarme. Im Gegenzug erkennt das Verfahren auch als einziges unerwartete Strukturbrüche, die nicht von Interesse sind. EDDM erkennt die Stelle 775b bzw. 775c später, liefert allerdings keine unerwarteten Alarme.

In Abbildung 3.17 ist Kombination 7 dargestellt. Die Niveauänderung im Jahr 2013 (775a) wird von keinem Verfahren erkannt. Ebenso wird der Abfall im Jahr 2017 (775c) nicht erkannt. Stelle 775b wird nur von EDDM und ECCD erkannt, die weitere, falsche Alarme liefern. Dabei gibt ECCD weniger falsche Alarme aus und ist daher zu bevorzugen.

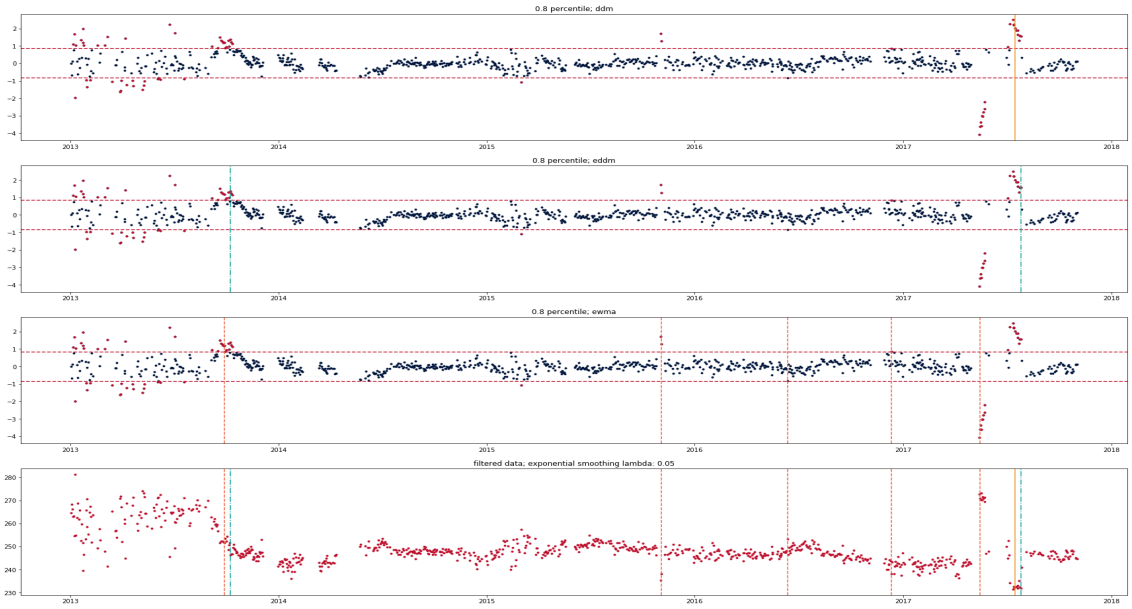


Abbildung 3.16: Kombination 6 für Pixel 775. Das 0.8-Quantil der Standardnormalverteilung ist eingezeichnet.

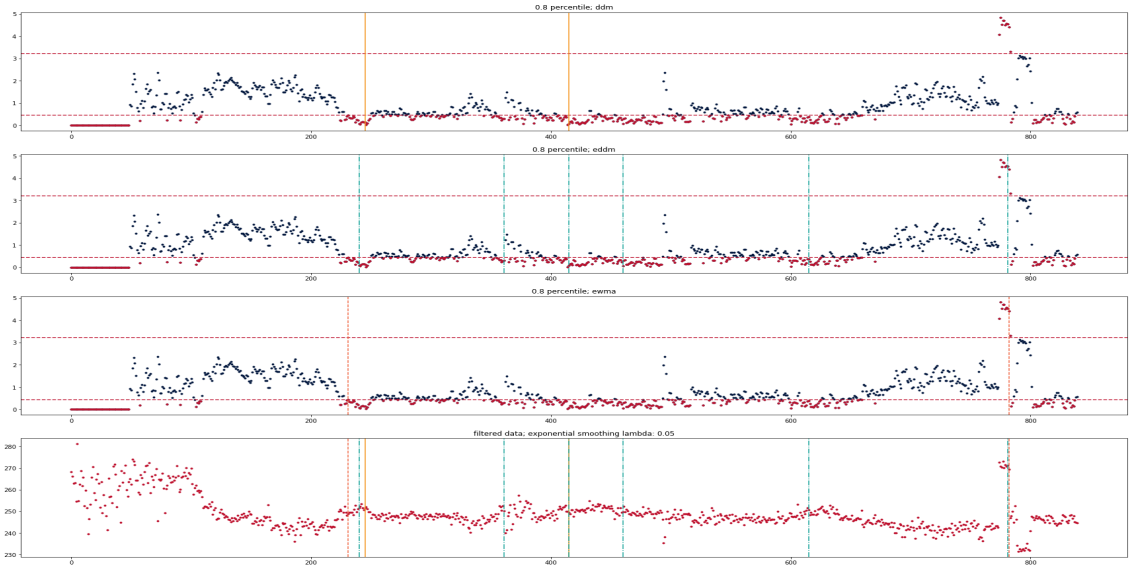


Abbildung 3.17: Kombination 7 für Pixel 775. Das 0,8-Quantil und das 0,2-Quantil der χ^2_2-Verteilung sind eingezeichnet.

Pixel 171

Als nächstes wird Pixel 171 betrachtet. Hier soll der Ausfall des Pixels (171a) erkannt werden. Ansonsten werden keine Alarme erwartet. In Abbildung 3.18 ist Kombination 2 für Pixel 171 dargestellt. Bei Verwendung der Parameter aus Abschnitt 3.4.1 wird der Ausfall des Pixels (171a) erkannt. Ein weiterer Strukturbruch wird zum Ende des Jahres 2014 erkannt. Diser Alarm lässt sich jedoch gut nachvollziehen, da die Messwerte einiger Tage in diesem Zeitraum stark abweichen. Da mit dem Page-Hinkley-Test und ADWIN in Abschitt 3.4.1 keine besseren Ergebnisse erzielt wurden, wurde an dieser Stelle zunächst auf die Anwendung verzichtet, um die Komplexität der Kombinationen zu begrenzen.

Abbildung 3.19 zeigt Kombination 3. Die Daten wurden zusätzlich gefiltert. Reservoir-Sampling-Normalization wurde zur Normalisierung verwendet. Der Ausfall des Pixels (171a) wird erkannt. Zusätzlich wird ein unerwarteter Alarm zum Ende des Jahres 2014 ausgegeben. Die Ergebnisse sind daher mit denen aus Abbildung 3.18 vergleichbar. Wie bei Pixel 775 ist die Wahl der Parameter intuitiver, wenn eine Filterung erfolgt.

Die von Kombination 4 ausgegebenen Strukturbrüche sind in Abbildung 3.20 zu sehen. Bei Verwendung der für Pixel 775 gewählten Parameter wird der Ausfall (171a) erkannt. Allerdings werden weitere Alarme ausgegeben. Diese sind teilweise nicht gut nachzuvollziehen. Durch die Wahl strengerer Parameter ($\lambda = 50, \delta = 0.05$) lässt sich die Anzahl der Alarme weiter reduzieren, ohne die Stelle 171a zu übersehen.

In Abbildung 3.21 ist Kombination 5 mit CUSUM und ADWIN dargestellt. Bei Wahl der Parameter aus dem vorherigen Abschnitt ($\lambda = 20, \delta = 0.05$) werden gute Ergebnisse erreicht und der Ausfall (171a) wird erkannt. Auch bei der Versendung von ADWIN wird der Ausfall erkannt, allerdings erfolgt der Alarm deutlich später als bei CUSUM. In Abbildung 3.22 werden DDM, EDDM und ECCD in Kombination 6 verwendet. Die interessante Stelle 171a wird von allen Verfahren erkannt, ohne dass weiter Fehlalarme ausgegeben werden. In Abbildung 3.23 sind die, in Kombination 7 erkannten, Strukturbrüche dargestellt. ECCD und EDDM erkennen den Ausfall 171a gut. Auch DDM erkennt den Ausfall, der Alarm wird jedoch später ausgegeben. Nur EDDM liefert unerwünschte Alarme. Im direkten Vergleich ist ECCD vorzuziehen.

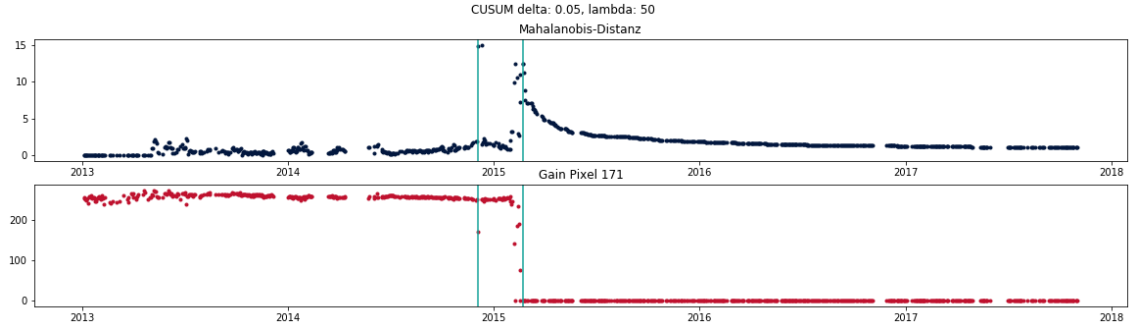


Abbildung 3.18: Kombination 2 für Pixel 171.

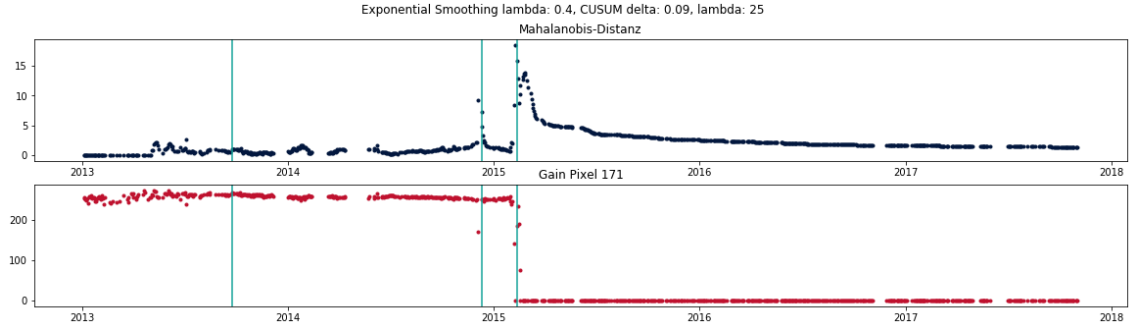


Abbildung 3.19: Kombination 3 für Pixel 171.

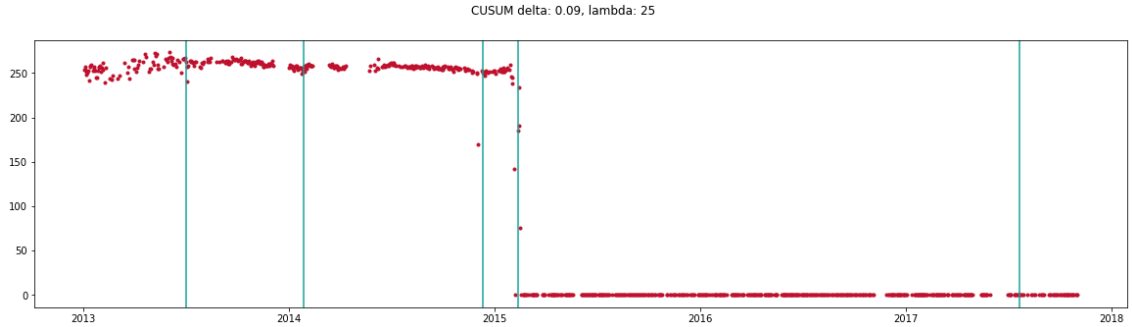


Abbildung 3.20: Kombination 4 für Pixel 171.

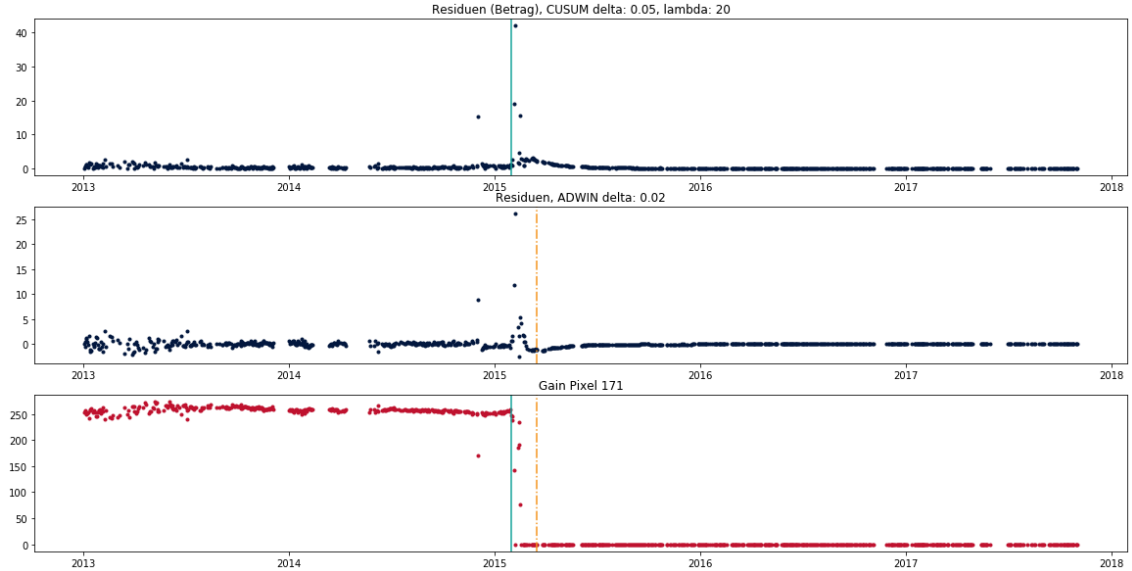


Abbildung 3.21: Kombination 5 für Pixel 171.

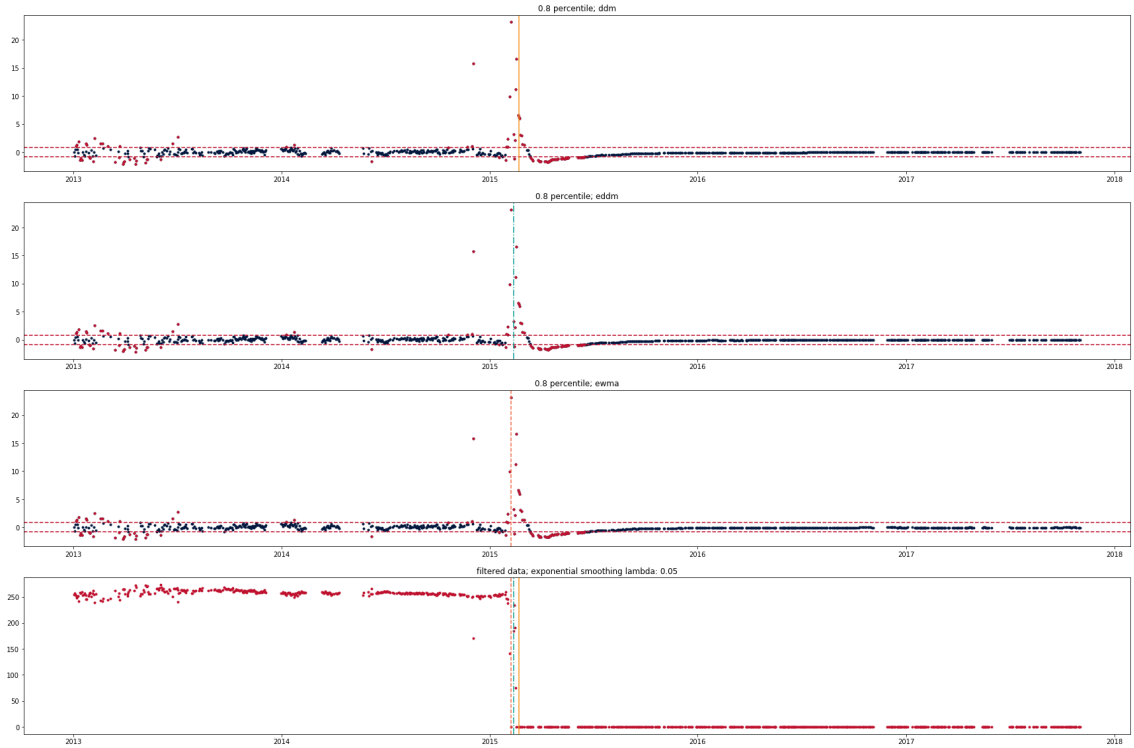


Abbildung 3.22: Kombination 6 für Pixel 171.

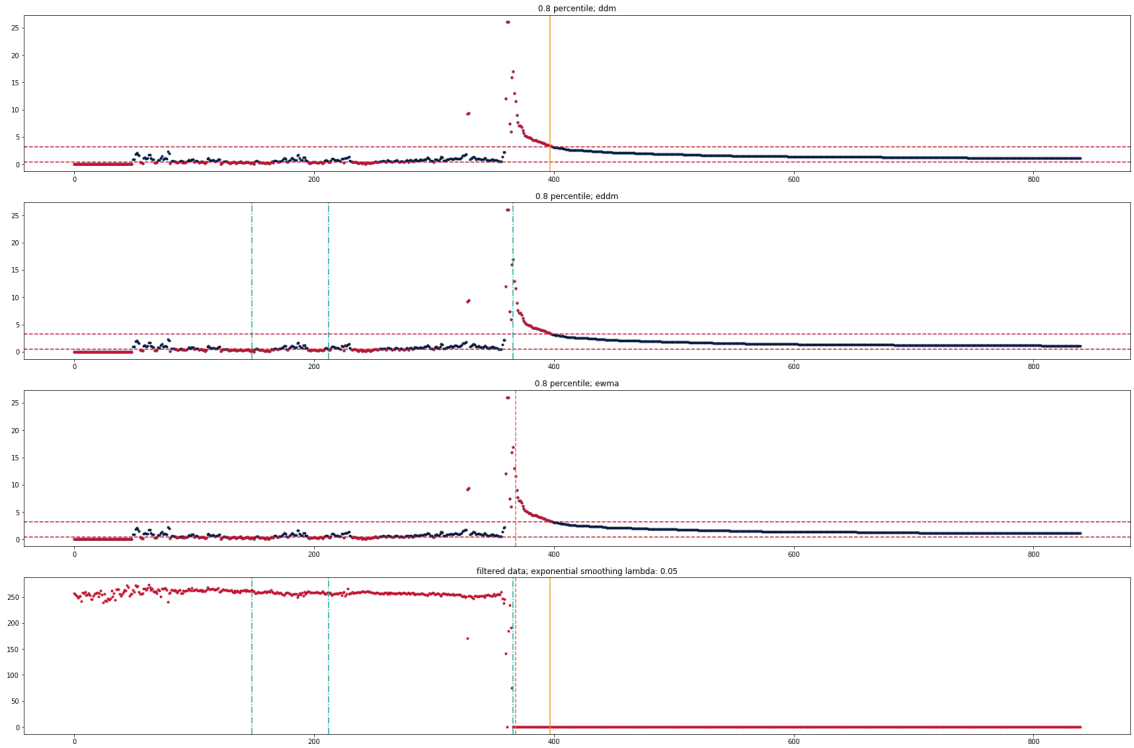


Abbildung 3.23: Kombination 7 für Pixel 171.

Pixel 1165

Bei der Betrachtung von Pixel 1165 fällt vor allem auf, dass die Werte stärker streuen als bei den anderen Pixeln. Die interessanten Stellen lassen sich mit allen Kombinationen nur sehr schlecht finden. Die besten Ergebnisse liefert Kombination 6. Diese Kombination ist in Abbildung 3.24 dargestellt. DDM, EDDM und ECCD geben Alarme im Bereich 1165a aus. Auch im Bereich 1165c erkennen EDDM und ECCD Strukturbrüche. Die Stelle 1165b wird nicht erkannt. EDDM gibt zum Ende des Jahres 2015 eine zusätzliche Warnung aus. Im direkten Vergleich ist ECCD am besten zu bewerten.

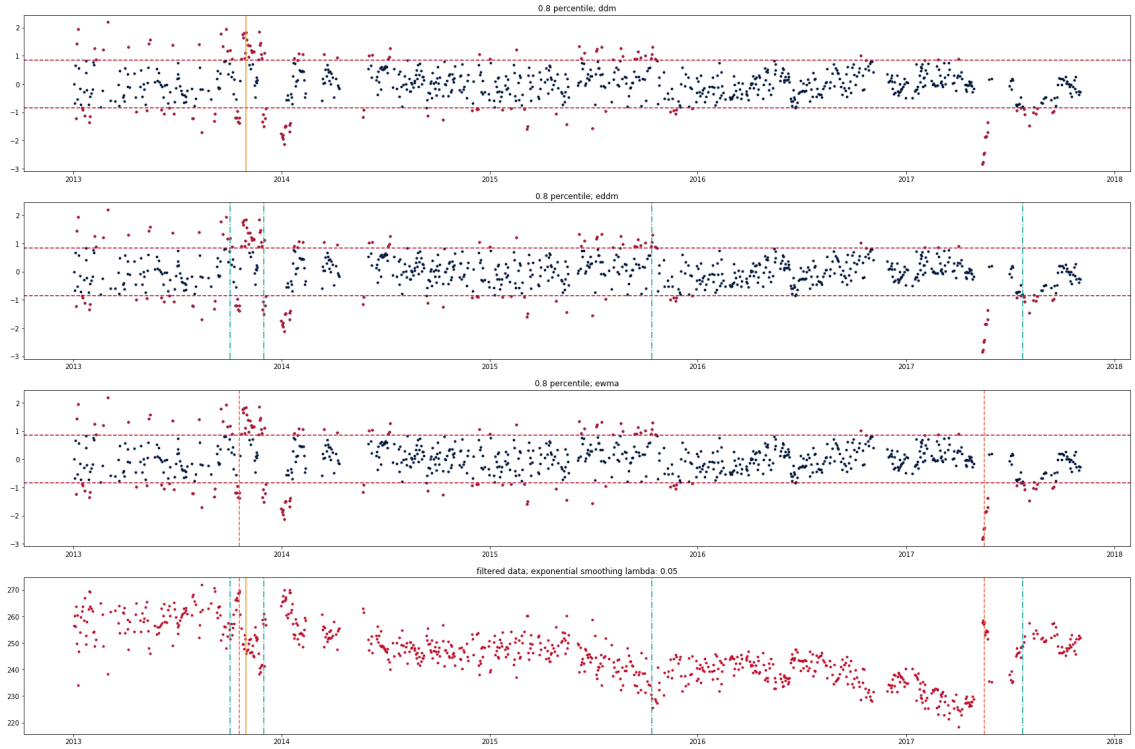


Abbildung 3.24: Kombination 6 für Pixel 1165.

Pixel 0

Da das Verhalten dieses Pixels unauffällig ist, sollen wenig Alarme ausgegeben werden. Wie im vorherigen Abschnitt werden die Kombinationen und Parameter gewählt, die in den vorherigen Experimenten die besten Ergebnisse geliefert haben, um ein Verfahren zu finden, das sich zur Entdeckung aller Inhomogenitäten eignet, und wenig unerwünschte Alarme ausgibt.

Abbildung 3.25 zeigt die Kombination 2 mit CUSUM. Bei Wahl von $\lambda = 50, \delta = 0.05$ wird nur ein falscher Alarm ausgegeben. In Abbildung 3.26 ist Kombination 3 dargestellt, bei der die Daten zusätzlich gefiltert werden. Bei der dargestellten Parameterwahl führt dieser zusätzliche Vorverarbeitungsschritt zu einer Verschlechterung der Ergebnisse, da

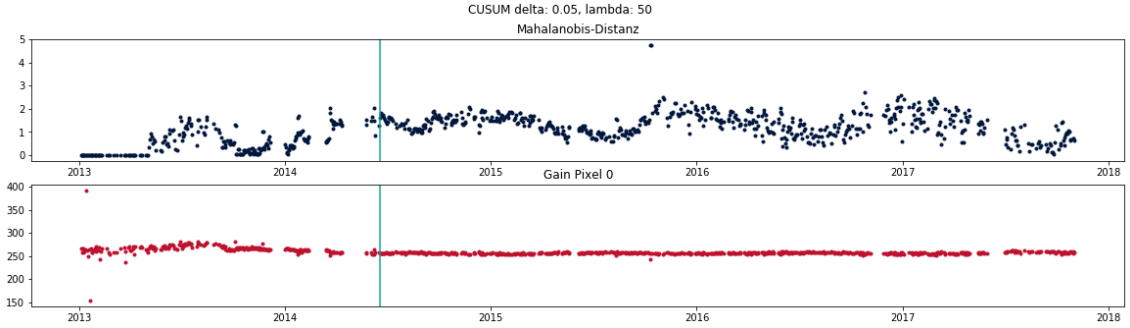


Abbildung 3.25: Kombination 2 für Pixel 0.

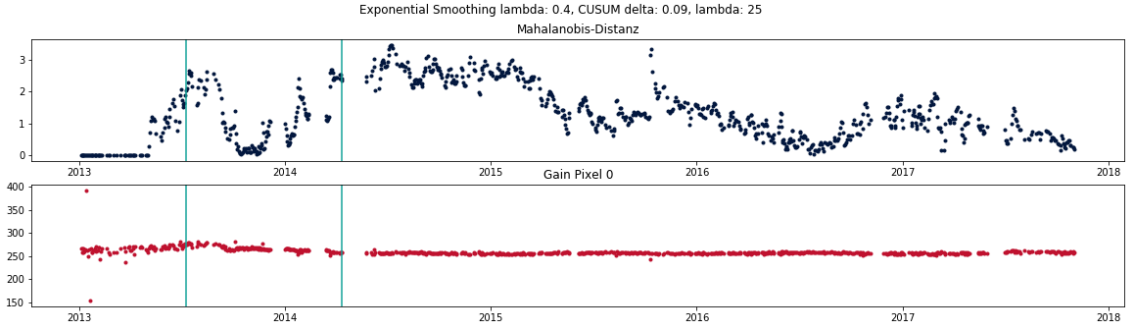


Abbildung 3.26: Kombination 3 für Pixel 0.

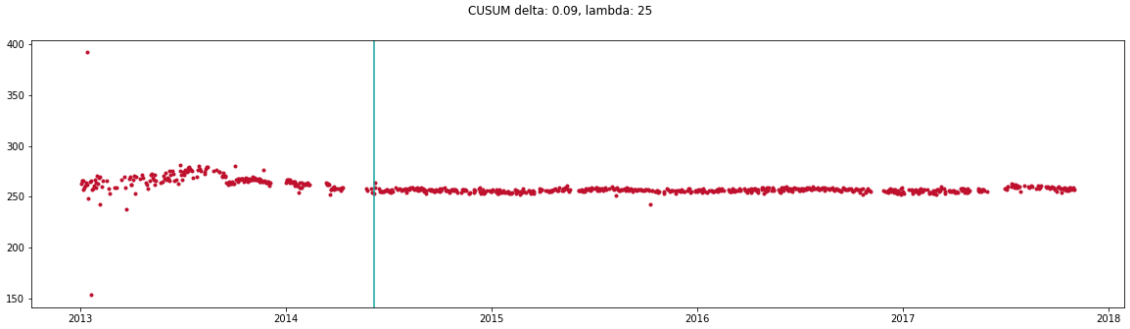


Abbildung 3.27: Kombination 4 für Pixel 0.

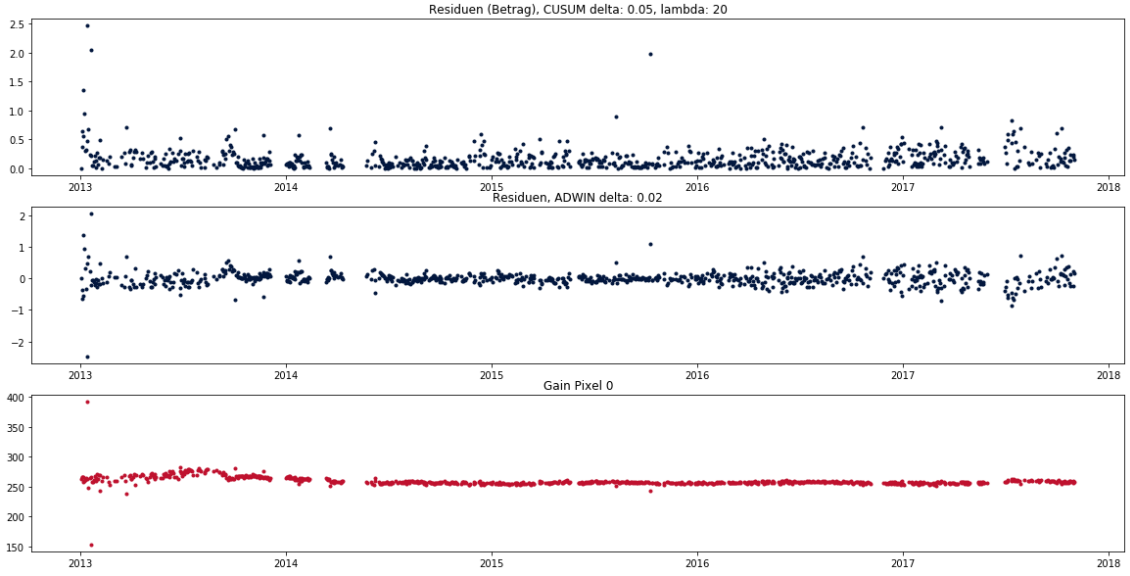


Abbildung 3.28: Kombination 5 für Pixel 0.

mehr Alarme ausgelöst werden. Trotzdem ist die Anzahl der Alarme weiterhin gering. In Abbildung 3.27 ist Kombination 4 für Pixel 0 dargestellt. Es wird nur ein Alarm ausgelöst. Abbildung 3.28 zeigt die Anwendung von Kombination 5 mit CUSUM und ADWIN. Es werden keine Alarme ausgelöst. Abbildung 3.29 zeigt die von Kombination 6 erkannten Strukturbrüche. Bei Verwendung von DDM und EDDM werden keine Alarme ausgegeben. ECCD gibt für diese Kombination nur einen Alarm aus. In Abbildung 3.30 sind die Alarme dargestellt, die bei Verwendung von Kombination 7 festgestellt werden. Bei Verwendung von ECCD werden mit zwei Alarmen die wenigsten Strukturbrüche erkannt. Die anderen Concept-Drift-Detektoren geben 3 Alarme aus.

Bei allen Kombinationen, die für die betrachteten Inhomogenitäten gute Ergebnisse geliefert haben, werden bei dem untersuchten unauffälligen Pixel wenige Alarme ausgelöst. Während einige Verfahrens- und Parameterkombinationen fehlerhafte bzw. unerwartete Alarme liefern, ist die Anzahl auch im schlechtesten betrachteten Fall gering (4 Alarme im Zeitraum vom 02.01.2013 bis zum 01.11.2017).

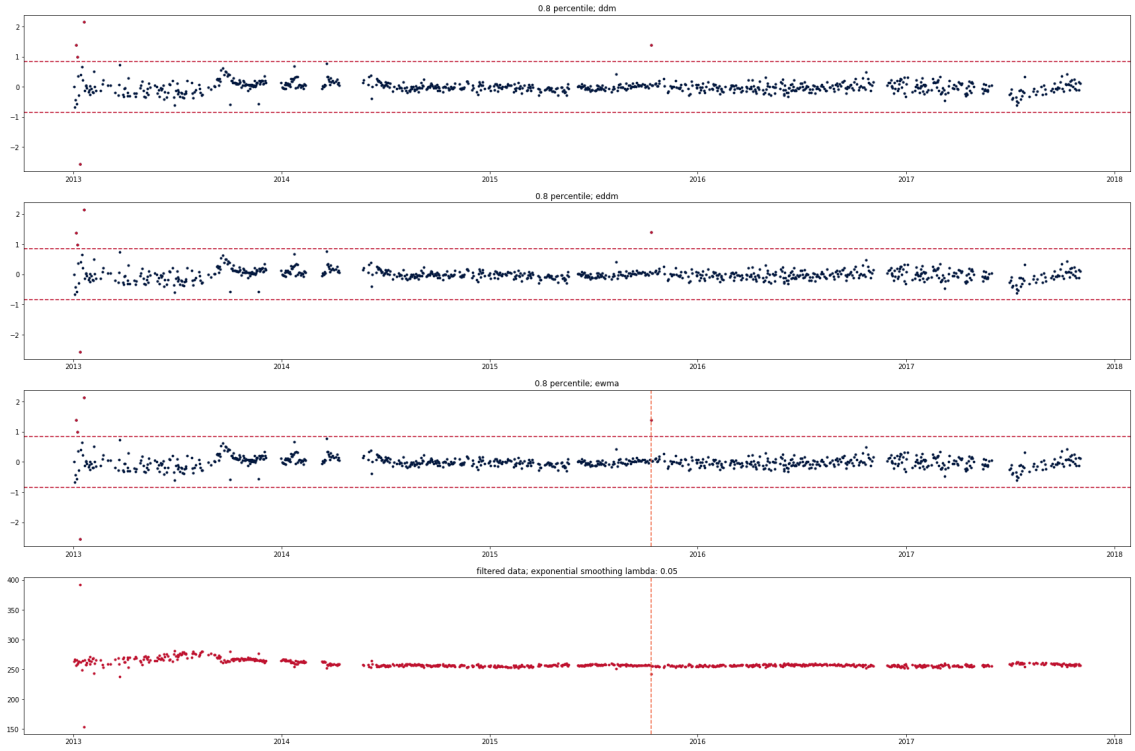


Abbildung 3.29: Kombination 6 für Pixel 0.

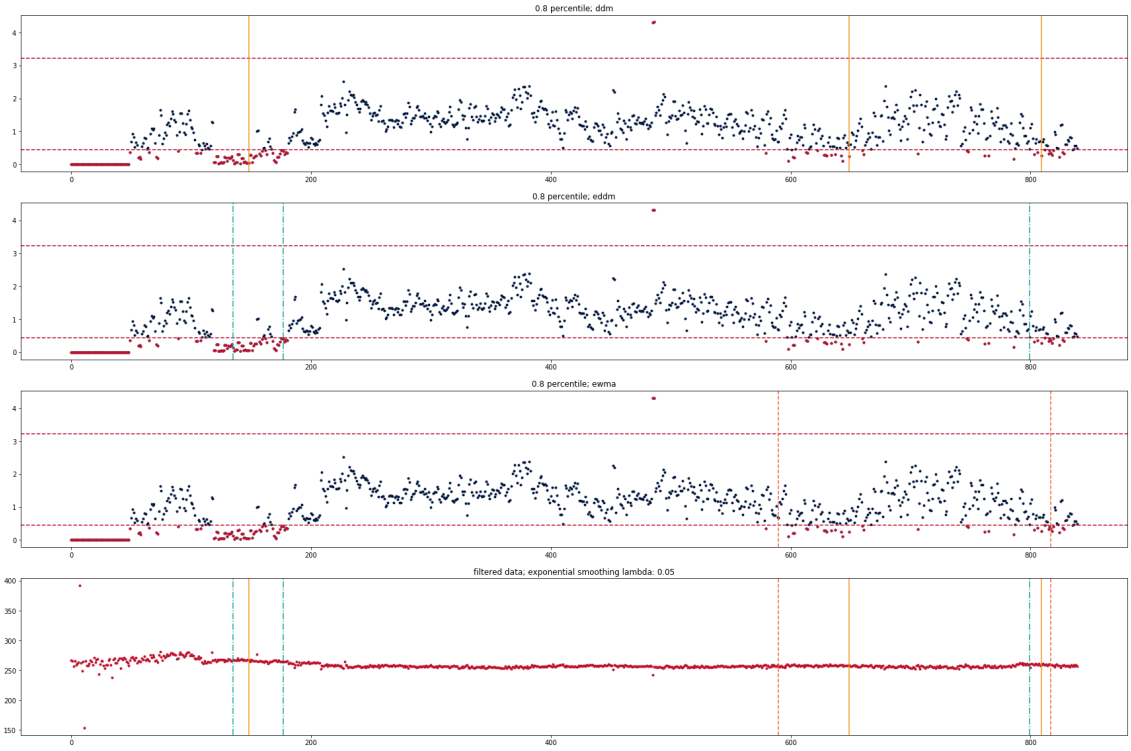


Abbildung 3.30: Kombination 7 für Pixel 0.

3.4.2 Auswertung des Vergleichs

Die Ergebnisse des Vergleichs der Kombinationen sind in Tabelle 3.1 zusammengefasst. Für jede Kombination ist in der Tabelle dargestellt welche Stellen entdeckt wurden. Neben den erkannten Stellen ist die Anzahl der unerwarteten Alarme dargestellt. Dabei wurde zwischen der Anzahl der Alarme für den unauffälligen Bildpunkt 0 und der Anzahl der Alarme für alle vier betrachteten Pixel unterschieden. Da die anderen drei Pixel (171, 775, 1165) inhomogenes Verhalten aufzeigen, werden die unerwarteten Alarme bei diesen Pixeln als weniger schlimm bewertet. Die Tabelle enthält in der Zeile von Kombination 1 keine Einträge für die Pixel 0, 171 und 1165, weil diese Kombination nur für das Pixel 775 angewendet wurde. Aus dem gleichen Grund bezieht sich die Anzahl der unerwarteten Alarme nur auf dieses Pixel. Kombination 1 liefert in diesem Fall bereits so viele unerwünschte Alarme, dass sie im Folgenden nicht weiter betrachtet wird.

Die Stellen 171a und 775b werden von den Kombinationen 2 bis 7 erkannt. Sie scheinen mit den Verfahren am einfachsten zu finden zu sein. Die Stelle 775a scheint ähnlich gut zu erkennen zu sein und wird nur von Kombination 5 nicht erkannt. Anders sieht es bei Stelle 775c aus. Diese Stelle wird nur von einer Kombination gefunden. Dies könnte an dem geringen Abstand zwischen den Stellen 775b und 775c liegen. Wie bereits im vorherigen Abschnitt erwähnt, sind die Stellen an denen die Messwerte von Pixel 1165

	Erkannte Stellen							Unerwartete Alarme	
Kombination	171a	775a	775b	775c	1165a	1165b	1165c	Pixel 0	gesamt
1 (CUSUM)	-	j	j	j	-	-	-	-	6+
2 (CUSUM)	j	j	j	n	n	n	n	1	4
3	j	j	j	n	j	n	n	2	10
4	j	j	j	n	j	n	j	1	10
5	j	n	j	n	n	n	n	0	0
6 (ECCD)	j	j	j	n	j	n	j	1	4
7 (ECCD)	j	j	j	n	n	n	j	2	5

Tabelle 3.1: Zusammenfassung des Vergleichs der Kombinationen 2-7.

stark streuen nur schwer mit den beschriebenen Methoden zu finden. Es lässt sich in der Auswertung nicht abschließend beurteilen, ob die Kombinationen, die an den Stellen einen Strukturbruch feststellen, tatsächlich aufgrund der Streuung der Daten Alarm schlagen. Im Zeifelsfall wurden die Alarme als korrekt bewertet. Während alle Kombinationen für das unauffällige Pixel 0 ähnlich wenige Alarme auslösen, unterscheidet sich die Gesamtzahl der Alarme stärker. Auch bei der Betrachtung dieser Alarme lassen sich Unterschiede in der Nachvollziehbarkeit feststellen. Die unerwarteten Alarme sind deshalb nicht gleich zu bewerten. Inwiefern sich die Anzahl dieser unerwarteten Alarme auf die Leistungsfähigkeit dieses Ansatzes auswirkt, soll im folgenden Abschnitt genauer untersucht werden. Insgesamt liefern die Kombinationen 2 mit CUSUM, sowie 6 und 7 mit ECCD die besten Ergebnisse für die betrachteten Pixel. Diese Kombinationen erkennen viele der im Vorhinein identifizierten Stellen und geben wenig falsche Alarme aus. Wie im vorherigen Abschnitt beschrieben wurde, sind viele dieser Alarme außerdem gut nachvollziehbar. Obwohl Kombination 5 weniger Stellen erkennt, soll auch diese Kombination weiter betrachtet werden, da keine falschen Alarme ausgelöst wurden.

3.4.3 Übertragung auf alle Pixel

Zur abschließenden Bewertung der Verfahren und der Anwendbarkeit in der Praxis sollen die Verfahren nun auf alle Pixel des Teleskops angewendet werden. Dadurch soll vor allem die Anzahl der unerwünschten Alarme besser bewertet werden. Wenn zu viele Alarme ausgegeben werden, ergibt sich im Vergleich zur manuellen Überprüfung aller Daten auf Inhomogenitäten kaum eine Verbesserung, da ein großer Teil der Daten weiterhin manuell überprüft werden muss. Ein gutes Verfahren soll daher neben den erkennbaren Inhomogenitäten wenige unerwartete (und schlecht nachvollziehbare) Alarme ausgeben.

Die Kombinationen 2, 6 und 7 wurden auf alle Pixel angewendet. Daraufhin wurde die Anzahl der Tage, an denen für mindestens ein Pixel ein Strukturbruch festgestellt wurde, bestimmt. Die relative Anzahl der Tage mit Alarmen ist in Tabelle 3.2 dargestellt. Wenn die Parameter verwendet werden, die in den vorherigen Experimenten ermittelt wurden,

Kombination	Parameter	Relative Anzahl der Tage mit Alarm
2 (Red. → Norm.→ PSE → CUSUM)	δ: 0.05 λ: 50	0,6
6 (Red. → Norm.→ exp. Gl. Residuen→ ECCD)	0,8-Quantil Φ	0,2
7 (Red. → Norm.→ PSE → ECCD)	0,8-Qunatil χ^2_2	0,7
2 (Red. → Norm.→ PSE → CUSUM)	δ: 0.5 λ: 50	0,32
5 (Red. → Norm.→ exp. Gl. Residuen→ CUSUM)	δ: 0.05 λ: 20	0,03
6 (Red. → Norm.→ exp. Gl. Residuen→ ECCD)	0,95-Quantil Φ	0,06
7 (Red. → Norm.→ PSE → ECCD)	0,95-Qunatil χ^2_2	0,5

Tabelle 3.2: Vergleich der Kombinationen 2, 5, 6 und 7 bei Anwendung auf alle Pixel.

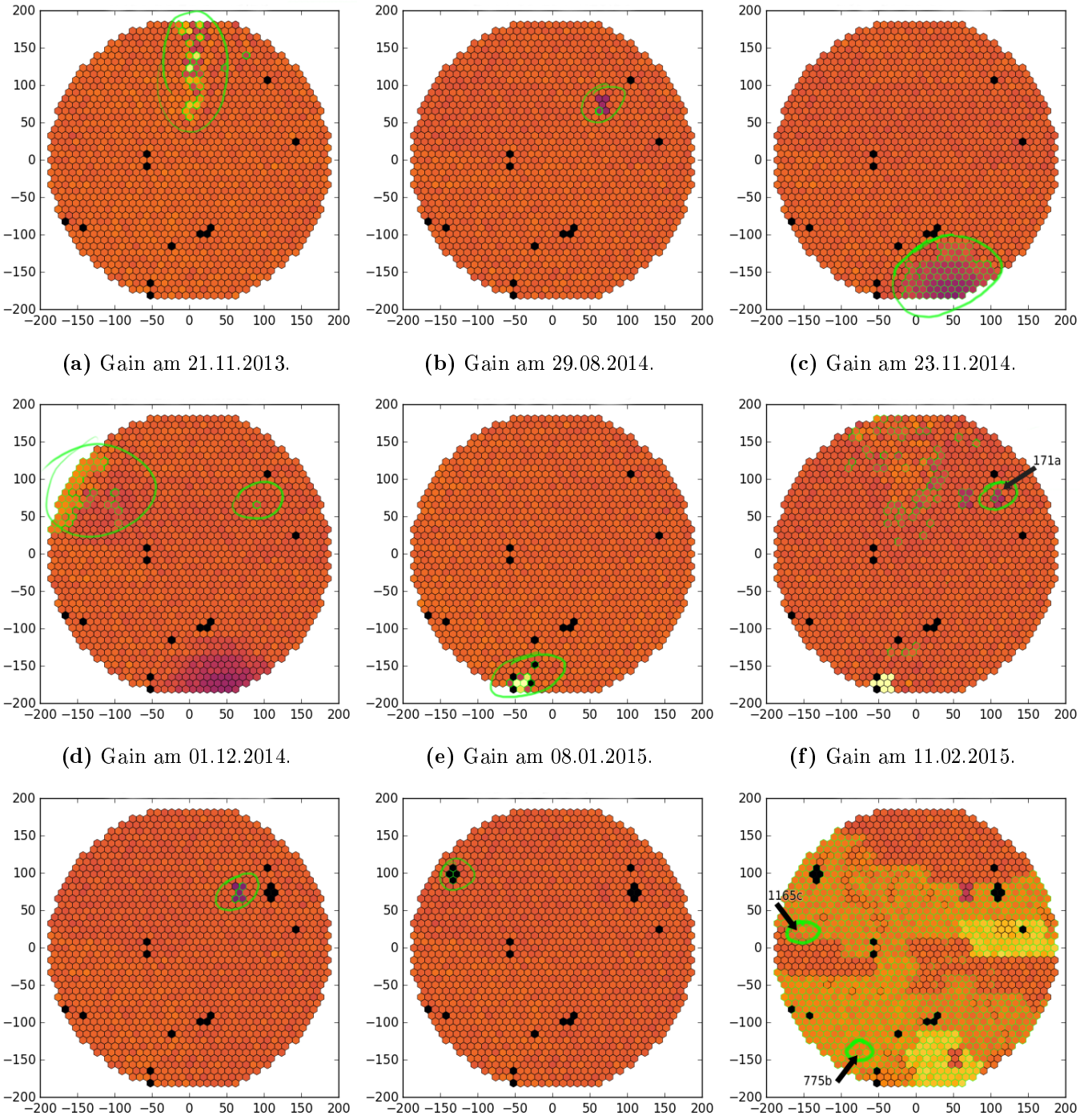
werden an vielen Tagen Alarme ausgegeben. Vor allem die Alarmraten von Kombination 2 und 7 sind zu hoch.

Eine Möglichkeit, um die Anzahl der Alarme zu reduzieren, ist die Parameter strenger zu wählen. Die Ergebnisse für diese Parameter sind ebenfalls in Tabelle 3.2 dargestellt. Auch die Verwendung von Kombination 5 reduziert die Anzahl deutlich. Die vorherigen Experimente mit einzelnen Pixeln haben jedoch gezeigt, dass eine strengere Wahl der Parameter (so dass weniger Alarme ausgelöst werden) dazu führt, dass einige der Inhomogenitäten nicht mehr zuverlässig identifiziert werden können. Ebenso werden mit Kombination 5 nicht alle Stellen erkannt. Welche Stellen mit den neuen Parametern gefunden werden und wie viele Alarme ausgegeben werden ist in Tabelle 3.3 dargestellt. Es ist zu erkennen, dass sich die Anzahl unerwünschter Alarme deutlich reduziert. Außerdem sind diese Alarme sehr gut nachvollziehbar und daher nicht zwingend als Fehler zu bewerten.

In Abbildung 3.31 ist für ausgewählte Alarme der Gain für alle Pixel zum Zeitpunkt des Alarms dargestellt. Zur Strukturbrucherkennung wurde Kombination 6 mit den beschriebenen, strengeren Parametern verwendet. Die gewählten Zeitpunkte zeigen, dass trotz der oben genannten Einschränkungen Inhomogenitäten im Verhalten der Pixel gefunden werden. Obwohl einige Stellen mit dieser Methode nicht erkannt werden, lassen sich Inhomogenitäten erkennen. Insbesondere können die Stellen 171a, 775b und 1165c in der Abbildung erkannt werden. Der Ausfall von Pixel 171 wird am 11.02.2015 festgestellt (Abb. 3.31f). Es lässt sich erkennen, dass auch benachbarte Pixel ausfallen. In Abbildung 3.31i sind die

	Erkannte Stellen							Unerwartete Alarme	
Kombination	171a	775a	775b	775c	1165a	1165b	1165c	Pixel 0	gesamt
2 (CUSUM)	j	**n**	**n**	n	n	n	n	0	2
6 (ECCD)	j	j	j	n	**n**	n	j	0	1
7 (ECCD)	j	**n**	**n**	n	n	n	n	0	0

Tabelle 3.3: Von Kombinationen 2, 6 und 7 mit strengeren Parametern erkannte Stellen.



(a) Gain am 21.11.2013. **(b)** Gain am 29.08.2014. **(c)** Gain am 23.11.2014.

(d) Gain am 01.12.2014. **(e)** Gain am 08.01.2015. **(f)** Gain am 11.02.2015.

(g) Gain am 29.02.2016. **(h)** Gain am 14.03.2016. **(i)** Gain am 13.05.2017.

Abbildung 3.31: Gain aller Pixel zum Zeitpunkt ausgewählter Strukturbruchalarme von Kombination 6. Pixel, für die ein Strukturbruch erkannt wurde, sind grün umrandet.

Stellen 775b und 1165c zu erkennen. Zu diesem Zeitpunkt werden für viele andere Pixel ebenfalls Alarme ausgegeben. Neben diesen Stellen, die schon im vorherigen Abschnitt gezielt gesucht wurden, werden auch auffällige Inhomogenitäten für Pixel erkannt, die nicht einzeln betrachtet wurden. In Abbildung 3.31a sind Auffälligkeiten zu beobachten, die Pixel in der Mitte der oberen Hälfte des Teleskops betreffen. Dies kann darauf hindeuten, dass die Abdeckung der Kamera zum Zeitpunkt der Messung nicht vollständig geschlossen gewesen ist und Licht einfallen konnte. In den Abbildungen 3.31b, 3.31e, 3.31g und 3.31h sind Inhomogenitäten zu erkennen, die kleine Gruppen von benachbarten Pixeln (< 10) betreffen. Besonders interessant ist die letzte Gruppe, für die am 14.03.2016 ein Strukturbruchalarm ausgegeben wird. Ähnlich wie Pixel 171 ist die Gruppe zu diesem Zeitpunkt ausgefallen und lieferte für die folgenden Messungen keine gültigen Messwerte. Die Methode eignet sich also auch zur Identifikation des Ausfalls dieser Pixelgruppe. In den Abbildungen lässt sich erkennen, dass viele Alarme nicht nur für einzelne Pixel, sondern für Gruppen von benachbarten Pixeln ausgelöst werden. Aus dieser Beobachtung ergibt sich eine weitere Möglichkeit, um die Alarmrate zu verringern, ohne zu viele wichtige Stellen zu übersehen. Diese Möglichkeit folgt aus dem Aufbau des Teleskops.

Viele der betrachteten Inhomogenitäten (die z. B. auf einen Defekt hindeuten) betreffen bauartbedingt nicht nur einzelne Pixel. Oftmals lässt sich inhomogenes Verhalten in Gruppen von benachbarten Pixeln beobachten. Es bietet sich daher an, nur dann einen Alarm auszugeben, wenn ein Strukturbruch für mehr als ein Pixel festgestellt wird. Die Alarmraten dieses Ansatzes sind in Tabelle 3.4 für verschiedene Grenzwerte dargestellt. Auch durch diesen Ansatz, können nicht alle Stellen aus dem vorherigen Abschnitt gefunden werden, wenn die Anzahl der Alarme reduziert werden soll. Kombination 6 liefert im Vergleich die besten Ergebnisse. Die Stellen, die von dieser Kombination bei steigenden Grenzwerten zuerst nicht mehr gefunden werden, sind die Stellen, die schwieriger zu erkennen sind. Der Ausfall 171a wird in diesem Fall auch bei den dargestellten Grenzwerten festgestellt, während er bei den Kombinationen 2 und 7 schon früh nicht mehr erkannt wird, obwohl noch viele Alarme ausgegeben werden.

Die beiden vorgestellten Kompromisse liefern, trotz der Einschränkungen, immer noch gute Ergebnisse. Obwohl sich einige, der im Vorhinein festgelegten Stellen, nicht finden lassen, können viele auffällige Zeitpunkte identifiziert werden. Die Anzahl der Alarme lässt sich auf eine akzeptable und überprüfbare Anzahl reduzieren. Gerade die Stellen, die schon im Vorhinein einfacher zu finden schienen, werden von den Verfahren erkannt.

Kombination	Parameter	Grenzwert	Relative Anzahl der Tage mit Alarm	nicht mehr gefunden
2 (CUSUM)	δ: 0.05 λ: 50	2	0,6	-
		6	0,29	171a, 775a
		15	0,04	171a,775a
6 (ECCD)	0,8-Quantil Φ	2	0,11	775a, 1165a
		6	0,05	775a, 1165a
7 (ECCD)	0,8-Qunatil χ^2_2	2	0,6	-
		6	0,29	171a,775a
		20	0,04	171a,775a,1165c

Tabelle 3.4: Vergleich der Methoden 2,6 und 7 bei Anwendung auf alle Pixel. Alarme werden nur gezählt, wenn die Anzahl der Pixel mit Strukturbruch mindestens dem Grenzwert entspricht.

Kapitel 4

Fazit und Ausblick

In dieser Arbeit wurden verschiedene Methoden untersucht, um Inhomogenitäten im Verhalten der Pixel des *FACT* zu finden. Das *First A-GPD Cherenkov Telescope* wird verwendet, um Gammastrahlungsquellen zu beobachten und soll als Vorlage für weitere Teleskope dieser Art dienen. In dieser Arbeit wurde untersucht, in wie weit sich verschiedene Ansätze zur Strukturbrucherkennung eignen, um fehlerhaftes Verhalten der Pixels des Teleskops automatisiert zu identifizieren. Dazu wurden die Verfahren CUSUM, Page-Hinkley-Test, DDM, EDDM, ECCD und ADWIN beschrieben und experimentell verglichen. Zusätzlich wurden diese Verfahren mit verschiedenen Vorverarbeitungsschritten kombiniert.

Zum Vergleich der Kombinationen wurden zunächst drei Pixel ausgewählt, die auffälliges, unerwünschtes Verhalten aufweisen. Zusätzlich wurde ein Pixel ohne offensichtliche Auffälligkeiten für den Vergleich ausgewählt. Da keine Informationen über tatsächliche Strukturbrüche in diesen Daten vorhanden waren, wurden (vor den Experimenten) auffällige Inhomogenitäten bestimmt. Daraufhin wurden die Kombinationen auf die Daten der ausgewählten Pixel angewendet, um eine Methode zu finden, die die markierten Stellen identifizieren kann, ohne weitere Alarme auszugeben. Durch diesen experimentellen Vergleich konnten Kombinationen (und Parameterkombinationen) ausgewählt werden, die viele der gewünschten Stellen identifizieren konnten, ohne dass viele zusätzliche Alarme ausgegeben wurden. Es konnte allerdings keine Kombination gefunden werden, die alle (zuvor ausgewählten) Stellen findet.

Um zu bewerten welche Auswirkungen die unerwünschten Alarme haben, wurden die besten Kombinationen auf alle Pixel des Teleskops angewendet. Dabei stellte sich heraus, dass zunächst an zu vielen Tagen Alarme ausgegeben werden. Um die unerwünschten Alarme zu reduzieren, wurden die Parameter angepasst. Dadurch ließ sich die Anzahl der Alarme auf eine manuell überprüfbare Anzahl reduzieren, wobei mit diesen Parametern einige interessante Stellen nicht mehr gefunden werden können. Alternativ dazu wurden zur Reduzierung der Alarmrate nur Zeitpunkte betrachtet, an denen für mehrere Pixel

Strukturbrüche erkannt wurden. Dieser Ansatz basiert auf der Beobachtung, dass Unregelmäßigkeiten im Verhalten der Pixel häufig Gruppen von benachbarten Pixeln betreffen. Auch bei diesem Ansatz musste ein Kompromiss zwischen erkannten Stellen und unerwarteten Alarmen gefunden werden.

Zusammenfassend lassen sich mit den vorgestellten Verfahren Inhomogenitäten im Verhalten der Pixel feststellen. Die Anzahl der ausgegebenen Alarme lässt sich allerdings nur auf eine sinnvolle Anzahl begrenzen, wenn in Kauf genommen wird, dass nicht alle interessanten Stellen identifiziert werden können. Es wurde jedoch eine Methode gefunden, die nur die schwer zu identifizierenden Strukturbrüche nicht findet. Zur Verbesserung dieser Ergebnisse können die Kombinationen durch verschiedene Lernverfahren ergänzt werden. Im Rahmen dieser Arbeit wurden diese nicht betrachtet, da der Fokus auf die vorgestellten Vorverarbeitungsschritte und Strukturbrucherkenner gelegt wurde. Weil die Laufzeit der Methoden bei den durchgeführten Experimenten vernachlässigbar war und die Rate mit der neue Datenpunkte eintreffen gering ist (etwa 2/24h), ist auch die Verwendung aufwendiger Lernverfahren denkbar. Zusätzlich zu den betrachteten Gain-Daten können Crosstalk, Baseline, Rate und Noise der Pixel untersucht werden, die aus den gleichen Kalibrierungsmessugen extrahiert werden.

Abbildungsverzeichnis

Literaturverzeichnis

[1] *Cherenkov Telescope Array*. `https://www.cta-observatory.org`. Abgerufen am 23.05.2018.

[2] Baena-García, Manuel, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, Ricard Gavaldà und Rafael Morales-Bueno: *Early drift detection method*. 2006.

[3] Bifet, Albert und Ricard Gavalda: *Learning from time-changing data with adaptive windowing*. In: *Proceedings of the 2007 SIAM International Conference on Data Mining*, Seiten 443–448. SIAM, 2007.

[4] Bifet, Albert, Geoff Holmes, Richard Kirkby und Bernhard Pfahringer: *MOA: Massive Online Analysis*. Journal of Machine Learning Research, 11:1601–1604, 2010.

[5] Bifet, Albert, Jesse Read, Bernhard Pfahringer, Geoff Holmes und Indrė Žliobaitė: *CD-MOA: Change Detection Framework for Massive Online Analysis*, Seiten 92–103. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[6] Biland, A, T Bretz, J Buss, V Commichau, L Djambazov, D Dorner, S Einecke, D Eisenacher, J Freiwald, O Grimm, H von Gunten, C Haller, C Hempfling, D Hildebrand, G Hughes, U Horisberger, M L Knoetig, T Krähenbühl, W Lustermann, E Lyard, K Mannheim, K Meier, S Mueller, D Neise, A K Overkemping, A Paravac, F Pauss, W Rhode, U Röser, J P Stucki, T Steinbring, F Temme, J Thaele, P Vogler, R Walter und Q Weitzel: *Calibration and performance of the photon sensor response of FACT — the first G-APD Cherenkov telescope*. Journal of Instrumentation, 9(10):P10012, 2014.

[7] Bockermann, Christian und Hendrik Blom: *The streams Framework*. Technischer Bericht 5, TU Dortmund University, 12 2012.

[8] Brockwell, Peter J. und Richard A. Davis: *Introduction to Time Series and Forecasting*. Springer Text in Statistics. Springer, 3 Auflage, 2016.

[9] Buss, J.: *FACT - Signal Calibration: Gain Calibration and Development of a Single Photon Pulse Template for the FACT Camera.* Diplomarbeit, TU Dortmund, 2013.

[10] Gama, João, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy und Abdelhamid Bouchachia: *A Survey on Concept Drift Adaptation.* ACM Comput. Surv., 46(4):44:1–44:37, März 2014.

[11] Gama, João, Pedro Medas, Gladys Castillo und Pedro Rodrigues: *Learning with Drift Detection*, Seiten 286–295. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[12] Gather, Ursula, Roland Fried und Michael Imhoff: *Online Classification of States in Intensive Care*, Seiten 413–428. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[13] Killich, Nils: *Im Rahmen der Arbeit implementierte Prozessoren.* `https://github.com/KhakiSalad/ba_conceptdrift_fact`.

[14] Leskovec, Jure, Anand Rajaraman und Jeffrey David Ullman: *Mining of massive datasets.* Cambridge university press, 2014.

[15] Page, E. S.: *Continuous Inspection Schemes.* Biometrika, 41(1/2):100–115, 1954.

[16] Ross, Gordon J, Niall M Adams, Dimitris K Tasoulis und David J Hand: *Exponentially weighted moving average charts for detecting concept drift.* Pattern recognition letters, 33(2):191–198, 2012.

[17] Rötner, Stefan: *Behandlung von Concept Drift in zyklischen Prozessen*, 2014.

Eidesstattliche Versicherung

_____ _____

Name, Vorname Matr.-Nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit/Masterarbeit* mit dem Titel

_____

_____

_____

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

_____ _____

Ort, Datum Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird gfls. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

_____ _____

Ort, Datum Unterschrift