Contents

1 Classical station triggers

As mentioned in **??**, continously analyzing data sent to CDAS from each of the 1600 SD water tanks would quickly exceed the computational capabilites of Augers' main servers. For this purpose, trace information is only collected from a station, once a nearby T3 event (c.f. **??**) has been detected. The formation of a T3 trigger is dependant on several T2, or station-level, triggers, which will be discussed in detail in this chapter. First, general comments about evaluation of trigger performances are given in **??**. Then the precise implementation of SD station level triggers, as well as their individual performance is given in **??**.

1.1 Performance evaluation

The performance of a trigger can be evaluated in many different ways. In the most general consideration, a confusion matrix holds information about the ability of a classifier to discern between different types, or classes, C. With the example at hand there exist two types of events one wishes to distinguish, a signal event C_1 in the form of an extensive air shower, versus background C_0. The confusion matrix thus becomes:

		Predicted C	
		C_1	C_0
True C	C_1	True positive (TP)	False negative (FN)
	C_0	False positive (TP)	True negative (TN)

From this, other potentially interesting variables can be derived. Of particular interest for the Auger observatory are the sensitivity and **F**alse **D**iscovery **R**ate (FDR). The former is the probability that a signal event will be classified correctly, i.e. an extensive air shower hits a water tank and correctly raises a T2 trigger. The sensitivity - in the following also called the trigger efficiency ϵ - is defined as

$$\epsilon = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{1.1}$$

The latter is a measure of how readily the triggers (wrongly) identify background events like stray cosmic muons as extensive air showers. It is imperative for any trigger algorithm operating in the SD to minimize this probability. Simply due to the number

of operating stations in the field, a small increase in FDR drastically raises the amount of potential events and hence load on the central analysis server of the observatory.

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{TP} + \mathrm{FP}}. \tag{1.2}$$

Ultimately, all test statistics constructed by classical station triggers (ignoring MoPS, c.f. **??**) are aliases for the deposited charge S in the WCD. Because S is heavily influenced by the primary particle energy, zenith and distance to the shower core, as well as to a lesser extent by shower age and statistical fluctuations, it makes sense to parametrize the trigger efficiency $\epsilon(E, \theta, \mathrm{SPD})$ in terms of these observables.

From a heuristic consideration, it can immediately be concluded that large separations between station and shower axis affect efficiencies negatively, because the particle distribution function monotonically decreases with increasing r (compare **??**). Similarly, inclined showers with a large θ are more attenuated compared to vertical showers, as they have to traverse a larger atmospheric depth ($\propto \sec(\theta)$) before reaching the detector. Lastly, primaries with large E on average deposit higher S in the WCD due to unleashing bigger particle cascades. Consequently ϵ is positively correlated with E.

The functional form that can be obtained by evaluating trigger efficiencies for a given (slice of) E and θ is labelled the **L**ateral **T**rigger **P**robability (LTP). It will be one of the main comparison metrics, by which different trigger algorithms are compared in this work. For classical triggers, two methods to extract the LTP are presented here. This is to show that both yield comparable results, and the latter method is a fair estimator for the neural network LTPs discussed in **??**.

1.1.1 $\overline{\text{Off}}\underline{\text{line}}$ lateral trigger probability

$\overline{\text{Off}}\underline{\text{line}}$ can simulate the SD detector response given a preprocessed shower footprint as given by e.g. CORSIKA. As such, calculating the LTP for a given event condenses to counting the number of triggered and non-triggered stations at specific distances from the shower axis. If this is done for a large enough sample size of showers, one eliminates noise induced by shower-to-shower fluctuations and arrives at an independent estimator for a T2 trigger given a shower at a shower plane distance r, with energy E and zenith θ. As per [**abreu2011lateral**], the closed form approximation of the LTP is given as

$$\mathrm{LTP}(r) = \begin{cases} \dfrac{1}{1 + \exp\left(\frac{r - R_0}{\Delta R}\right)}, & r \leq R_0 \\ \dfrac{1}{2}\exp\left(C(r - R_0)\right), & r > R_0 \end{cases} \tag{1.3}$$

In **??**, R_0, ΔR and C are all fit constants that will in general depend on E and θ. Most importantly, R_0 marks the shower plane distance where $(R_0) = 0.5$. This is connected to a steepening of the rising flank in the efficiency curve. Whereas an exponential function with decay constant $C < 0$ describes data well for large r, a logistic function

with scaling factor $1/\Delta R$ must account for the asymptotic transition to full efficiency closer to the core.

It must be mentioned that the motivation behind this parametrization is data- and not physics driven. In particular, LTP(r) is not smooth in R_0 if the parameters C and ΔR are not finetuned as indicated in **??**.

$$\lim_{r \to R_0^+} \frac{\partial \, \mathrm{LTP}(r)}{\partial r} = \frac{C}{2} \stackrel{!}{=} \frac{1}{4\Delta R} = \lim_{r \to R_0^-} \frac{\partial \, \mathrm{LTP}(r)}{\partial r} \tag{1.4}$$

In this work however, a different parametrization is used to estimate the T2 response of a station. The functional form of this adjusted trigger probability is a clipped logistic function, and given in **??**.

$$\mathrm{LTP}^*(r) = \min\left(1,\ \epsilon^* \left(1 - \frac{1}{1 + e^{-\frac{r - R_0}{\Delta R}}}\right)\right) \tag{1.5}$$

The reasoning for this choice is as follows:

- The original parametrization, LTP(r), eventually approaches 1. This hints to a problem. It is not a guarantee that some trigger algorithm will detect all extensive air showers. Espically neural network triggers might be sensitive to only a subset of showers. This is reflected in the latter form, LTP$^*(r)$, by introducing an additional fit parameter, the pseudo-efficiency $\epsilon^* > 0$. In the case of $\epsilon^* \geq 1$, the domain of the function is correctly mapped to $[0, 1]$.
- There exists an imbalance in training data. Due to the geometry of the SD array, more traces at smaller r are available. In an attempt to reduce possible biases resulting from low statistics at small SPD, the form is kept as simple as possible.
- The discontinuity at R_0 is replaced by a kink at $R^* = R_0 - \frac{\log\left((1-1/\epsilon^*)^{-1}-1\right)}{\Delta R}$. This is however expected. Since there exists a phase transition, namely to full efficiency, at this point, discontinuities in the lateral trigger probability are allowed.

This approach only marginally takes into account shower-to-shower-fluctuations. Such statistic perturbations are responsible for a smearing of the (initially) hard transition from sub- to full efficiency. The parametrization used by the Pierre Auger collaboration takes this into consideration by design.The presented LTP$^*(r)$ does - at least explicitly - not. As a result, one could expect a bias, where LTP$^*(R^*)$ over- or underestimates the actual trigger probability. This is however not the case when examining the residuals of the performance fit that is done in **??**. A plot showcasing this result is offered in **??**

1.1.2 Bayesian folding

[to do: write]

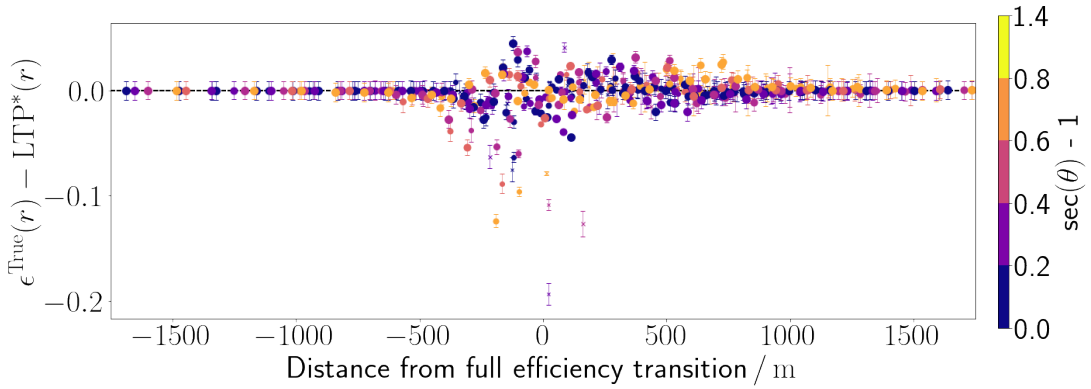

Figure 1.1: The residuals from comparing ϵ^{True} to LTP(r) vanish at large (small) r. No systematic bias is observed in the transitional region around R^*. Large outliers are caused by the dataset being limited in size at low energies (marked with an x).

1.2 Implementation

1.2.1 Threshold trigger (Th)

The **Th**reshold trigger (Th) is the simplest, as well as longest operating trigger algorithm [**triggerGuide**] in the field. It scans incoming ADC bins as measured by the three different WCD PMTs for values that exceed some threshold. If a coincident exceedance of this threshold is observed in all three WCD PMTs simultaneously, a Th-T1/2 trigger is issued. A pseudocode implementation of this algorithm is hence given by the below code block.

```
th1 = 1.75  // Th1 level threshold above baseline, in VEM
th2 = 3.20  // Th2 level threshold above baseline, in VEM

while True:

    pmt1, pmt2, pmt3 = get_next_output_from_WCD()

    if pmt1 <= th2 and pmt2 <= th2 and pmt3 <= th2:
        raise ThT1_trigger
    if pmt1 <= th1 and pmt2 <= th1 and pmt3 <= th1:
        raise ThT2_trigger
    else:
        continue
```

Logically, with increasing signal strength S in the PMTs, the likelihood of having observed an extensive air shower raises. This is reflected in the trigger level logic, where a coincident signal of $S \leq 3.20\,\mathrm{VEM_{Peak}}$ is immediately forwarded to CDAS, whereas a signal $1.75\,\mathrm{VEM_{Peak}} \leq S < 3.20\,\mathrm{VEM_{Peak}}$ only raises a Th-T1 trigger. The algorithm is insensitive to signals that do not exceed at least $1.75\,\mathrm{VEM_{Peak}}$ in all three PMTs.

In the case of faulty electronics, where only a subset of the WCD PMTs are available, the trigger thresholds (in units of $\mathrm{VEM_{Peak}}$) are updated according to **??**.

Table 1.1: Numerical values from [**triggerSettings**]

n_{PMT}	Th-T2	Th-T1
1	5.00	2.85
2	3.60	2.00
3	3.20	1.75

Performance

[to do: write]

1.2.2 Time over Threshold trigger (ToT)

The **T**ime **o**ver **T**hreshold trigger (ToT) is sensitive to much smaller signals than the Threshold trigger discussed in **??**. For each PMT in the water tank, the past 120 bins are examined for values that exceed $0.2\,\mathrm{VEM_{Peak}}$. If 13 or more bins above the threshold are found in the window - ordering or succession do not matter - the PMT is considered to have an elevated pedestal. The ToT trigger requires at least two PMTs with an elevated pedestal in order to activate. As such, the algorithm is theoretically sensitive to events that deposit just $0.5\,\mathrm{VEM_{Ch.}}$ A pseudocode example is given below.

```
threshold   = 0.2  // pedestal threshold, in VEM
n_bins      = 12   // number of bins above pedestal
window_size = 120  // considered window length

buffers = [[False for i in 1..window_size] for j in 1..3]
step_count = 0

while True:

    pmts = get_next_output_from_WCD()
    buffer_index = step_count % window_size
    count_active_PMTs = 0

    for pmt, buffer in pmts, buffers:
        if pmt <= threshold: buffer[buffer_index] = True

        if count_values(buffer, value = True) > n_bins:
            count_active_PMTs += 1

```

```
    if count_active_PMTs >= 2:
        raise ToTT2_trigger
    else:
        step_count = buffer_index + 1
        continue
```

Performance

[to do: write]

1.2.3 Time over Threshold deconvoluted trigger (Totd)

An extension to even lower signal strengths is given by the **ToT-d**econvoluted trigger (ToTd). As the name implies, the implementation of the algorithm is completely analog to the ToT trigger in **??**. Only the FADC input stream from the three PMTs is altered according to **??**.

$$d_i = (a_i - a_{i-1} \cdot e^{-\Delta t/\tau}) \,/\, (1 - e^{\Delta t/\tau}) \tag{1.6}$$

In **??**, the deconvoluted bin d_i is calculated from the measured FADC values a_i and a_{i-1}, where a_{i-1} is scaled according to an exponential decay with mean lifetime $\tau = 67\,\text{ns}$. This reduces the exponential tail of an electromagnetic signal to a series of pulses which in the case of $a_{i-1} < a_i$ exceed the original signal strength. As such, the deconvoluted trace can satisfy the ToT trigger requirements, whereas the original raw FADC values might not have, extending the sensitivity of the ToT trigger to lower signal strengths. The scaling constant $\Delta t = 25\,\text{ns}$ is tied to the sampling rate of UB electronics (c.f. **??**). The choice of the numerical constants τ and Δt is explained in more detail in [**ToTtriggerIdea**].

Performance

[to do: write]

1.2.4 Multiplicity of Positive Steps (MoPS)

The **M**ultiplicity **o**f **P**ositive **S**teps (MoPS) algorithm triggers on positive flanks of an FADC trace, which can be related to the arrival of new particles in the water tank.

A positive flank in the FADC trace of a single PMT is any combination of at least two bins that are monotonically increasing in value, in a window of 120 bins. Once such a positive step has been identified, a (MoPS) trigger veto is applied to the next

$$n_{\text{skip}} = \lfloor \left(\log_2(\Delta y) + 1\right) - 3 \rceil \tag{1.7}$$

bins, where Δy refers to the total vertical increase in the step from first to last bin. Note that in **??** the notation $\lfloor x \rceil$ is used as shorthand notation to round x to the nearest integer. If Δy is bigger than $y_{\text{min}} = 3$ ADC (to filter random fluctuations), but does not exceed $y_{\text{max}} = 31$ ADC (to prevent triggering on muonic coincidences), it is added to a ledger. If the number of rising flanks in the ledger is bigger than $m > 4$ for at least two PMTs, a final check regarding the integral of the FADC trace is performed. If this check passes, a MoPS-T2 trigger is issued to CDAS. An in-depth discussion of the different hyperparameters for this trigger is offered e.g. in [**gapMoPS**].

It is impossible to accurately recreate the MoPS trigger in simulations. The integral test above compares the sum of the last 250 bins against a threshold ($\sum a_i > 75$). Since not all 250 bin values are available to CDAS, differing results are to be expected when comparing the implementation of the algorithm in the SD field versus its' counterpart in analysis software.

For this purpose, the MoPs trigger is not considered in the analysis presented in **??**. The implications of this choice are layed out in the following paragraph.

Performance

[to do: write]

1.2.5 Combined performance