1 Neural network training data

Over their relatively brief existance, neural networks have been shown to perform increasingly impressive tasks (e.g. [**openai2019dota**], [**openai2023gpt4**], and many more). However, they learn by example. The performance of a neural network is directly linked to the input data it receives during training. If the training data is not an accurate example of real world information a network later operates on, insight gained from it is at best an approximation, and at worst completely randomly generated data.

As such, it is not a question *if* some neural network architecture can learn to identify an extensive air shower from WCD data, but rather which implementation, fed with which information, does. For this purpose, this chapter explains the procedure with which training data is generated. As stated above, this must occur with a focus on being representative of data actually measured in the SD array. The elected approach to create time traces is modularized. The structure of this chapter reflects this. First, general comments about the characteristics of background data (i.e. the WCD detector response in the absence of an extensive air shower) are made in **??**. Next, the process to extract signal originating from CRs is detailed in **??**. Lastly, building the time trace from the aforementioned modules and drawing samples from it for a neural network to train on is done in **??**.

1.1 Background dataset

While a flux of partices causes elevated ADC levels in both the HG and LG channels of a WCD PMT during a shower event, the lack of such a phenomenon does not imply the readout information is uniformly flat. Instead, it hovers around the channels' baseline (c.f. **??**) with occasional spikes upwards due to low-energy particles impinging on the detector. Coupled with electronic noise from the many digital components in the station electronics, the **U**pgraded **U**nified **B**oard (UUB), this constitutes the data that is collected inbetween air shower events.

1.1.1 Accidental muons

Most low-energy background particles present in the detector are muons. These are produced in the upper atmosphere during cascading processes analog to **??**. Due to the low primary energy the electromagnetic component of the shower is thermalized before it reaches surface level. The muonic component by itself does not contain enough information to enable an accurate reconstruction of primary energy and origin. This fact, coupled with the high flux of events at lower energies ($\Phi|_{E=100\,\mathrm{GeV}} \approx 1\,\mathrm{m}^{-1}\,\mathrm{s}^{-1}$

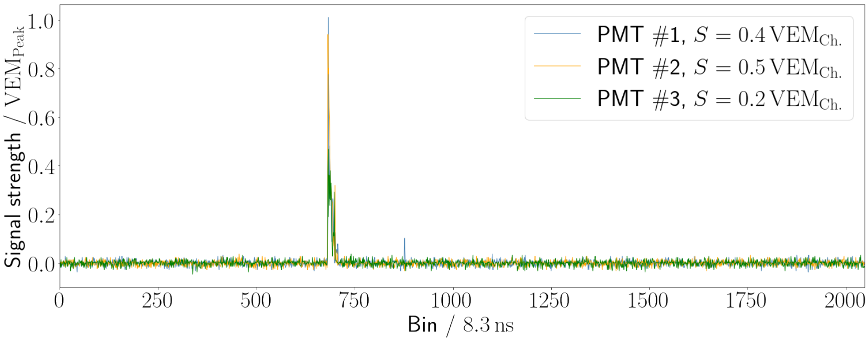


Figure 1.1: The simulated time trace from a single muon. The maximum peak of the time trace is equal to $1\,\mathrm{VEM_{Peak}}$. The integral over each PMT, S, sums to $\approx 1\,\mathrm{VEM_{Ch.}}$.

[**boezio2000measurement**]) make these events unsuitable for analysis. Stray muons, even though they originate from extensive air showers, must consequently be considered background events.

The rate at which such particles traverse a WCD tank is $f_{\mathrm{Acc.}} \approx 4.665\,\mathrm{kHz}$ [**DavidBackgroundSim**]. Their arrival time is Poisson-distributed. This implies that generally, one in 14 time traces contains signal from a low-energy background event. Coincidences of two accidental muons occur on a sub-percent level. Any higher order of coincidences is likely originating from a single air shower process. The typical signal recorded by the surface detector from a single muon is presented in **??**.

A library of background traces of this type was provided by David Schmidt [**DavidBackgroundSim**]. However, only the largest response of the three WCD PMTs is available for this library. Due to the lack of information one is either forced to assume the response to a low-energy background particle is the same across all PMTs, or neglect the response of the two remaining PMTs altogether upon injectiong a background muon into a signal trace (c.f. **??**). In both cases, neural networks are provided an easily detectable pattern to discern such particles from "real" shower signal. As a result, it should be refrained from training AI triggers on this dataset.

1.1.2 Electronic noise

Electronic noise is the umbrella term given to the distortions that some signal is subject to during digital readout. Such noise can have many different origins. An illustrative example is given by the **L**aser **I**nterferometer **G**ravitational wave **O**bservatory, which excludes the 60 Hz band and its' harmonics from analysis. This is owed to the fact that the DC frequency standard in the United States introduces systematic uncertainties in the detector [**martynov2016sensitivity**]. In the electronics of Pierre Augers' SD array, electronic noise is assumed to be Gaussian. That is to say that the ADC values of a time trace that was measured while no particle produced signal in the tank are normally distributed around the baseline. The standard deviation can be estimated from monitoring data, as is shown in **??**.

1.1.3 Random traces

Both above mentioned phenomena can be simulated, and the simulation results used as background training data for the neural networks discussed in the next chapter. A more accurate method however, and the approach elected for this work is to utilize directly measured data from the field. Thanks to the work of David Nitz, there exist collections of so called random-traces[1] that were gathered by forcing DAQ readout via a manually set trigger.

In particular, two datasets of UUB random-traces have been created until now. They were taken from 13th-18th March 2022, and 14th-18th November 2022 respectively. The first set contains data a total of sixteen million time traces distributed over four different SD stations. For reasons explained in **??**, only data from the first set is used in the analysis presented in this work.

Characteristics

Contrary to the naming of the random trigger, it occurs at a deterministic time. More accurately, the process of measuring random-traces is as follows; A single time trace ($2048 \cdot 8.333\,\mathrm{ns} = 17.07\,\mu\mathrm{s}$) is written to the local station buffer every 10 ms. Once the buffer has accumulated enough data, it is written to a USB storage device. Because of a bottleneck in the last step, the process takes about 22 h per station [**nitzCorrespondence**].

It is thus not the trigger that is unpredictable, but the data measured by each trigger. Due to the read/write process being indepentant of the measured data (as opposed to the algorithms in **??**) the latter must be considered to be essentially random. For the most part, random-traces are assumed to consist solely of electronic noise. However, signal of cosmic origin - be it accidental muons or extensive air showers - will appear in the data at a rate at least consistent with **??**.

A crude analysis of the type of noise in the random-traces can be made by examining the spectral density of the dataset, shown in **??**. Harmonic modulations visible in both spectra might originate from an offset between last and first bin of the random-traces. If this offset is nonzero, the periodic extension of $f(x)$ exerts a step-function-like behaviour. The Fourier transform consequently reflects this [**burrows1990fourier**]. Still, several features of $|\hat{f}(\xi)|^2$, espically present at 10 MHz, imply the presence of systematic noise in the UUB. Nevertheless, the large scale form of the spectral density is compatible with at least two noise models, that cannot be distinguished based on the data at hand:

- $|\mathbf{f}(\xi)|^2 \propto \exp\left(\frac{(\xi-\mu)^2}{2\sigma^2}\right)$. The spectral density is Gaussian. This implies the noise is Gaussian distributed as well, confirming the assumption in **??**.

[1] to avoid possible confusion between this dataset and a *random* trace in the statistical sense, the traces recorded by David Nitz are referred to as random-trace, with a hypen.

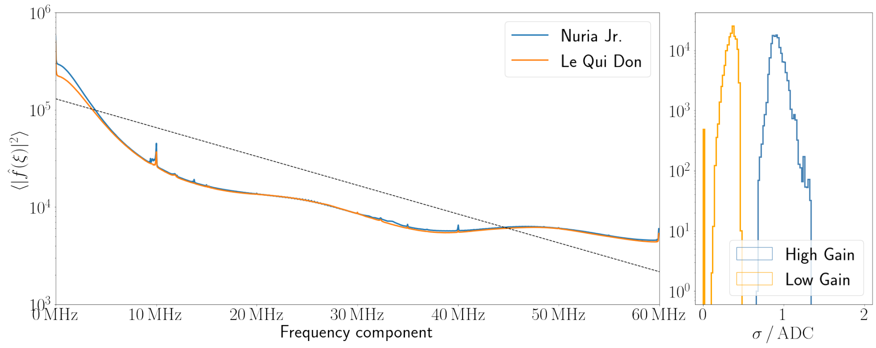


Figure 1.2: (*Left*) The random-trace spectral density for two stations. Plotted with a dashed black line is reference attenuation curve falling at −6 dB/Oct. The spike at 10 MHz is of unknown origin and represents systematic noise in the UUB electronics. (*Right*) Example variance of all UUB stations in the surface detector array. The data shown in this plot was recorded on November 15th 2022.

- $|\mathbf{f}(\xi)|^2 \propto \exp(-m\xi + b)$. The spectral density is proportional to ξ^{-n} for some power n. The case $n = 2$ (−6 dB/Oct attenuation) seems to describe the observations well, hinting that the generating function could be Brownian.

Calibration

The random-trace files contain raw measurement data in units of ADC for the HG and LG channel of the three WCD PMTs. In a first step to standardize this information, the baseline is substracted from each FADC bin. This is done via the baseline finding algorithm described in **??** and [**tobiasBaseline**, **tobiasBaselineUUB**]. Note that this approach differs from the baseline finding algorithm that runs on each station (c.f. **??**). However, the difference is negligible (<< 1 ADC) for traces that do not contain any signal, which is the case for the vast majority of the dataset.

Next, the baseline-substracted time traces are converted from units of ADC to $\mathrm{VEM}_{\mathrm{Peak}}$. This conversion is not straight forward, as it requires knowledge of I_{VEM} at the time of data taking. Each station estimates this value in periodic time intervals in the context of monitoring diagnostics.

For the second dataset of random-traces (taken from 14th-18th November 2022) a UNIX timestamp packaged with each time trace may be related to monitoring data. This reveals that no information regarding I_{VEM} was forwarded to CDAS for any station while it recorded random-traces. As a result, the entire dataset is unfortunately rendered useless for this work.

For the first collection of random-traces, monitoring data is available, but there exists no timing information for the individual traces. Only the date of the measurement is known. The elected procedure to evaluate data as accurately as possible is thus to calculate the day average of I_{VEM} and Q_{VEM} and take this as the best (first) estimate for each trace. As can be seen in **??**, this eliminates half of the remaining dataset, as two of the four stations show a large variance in I_{VEM} . The day average in these particular

Table 1.1: Calibration constant I_{VEM} for random-traces.

PMT #	Nuria	Le Qui Don
1	159.34 ADC	145.79 ADC
2	161.37 ADC	144.63 ADC
3	149.91 ADC	154.62 ADC

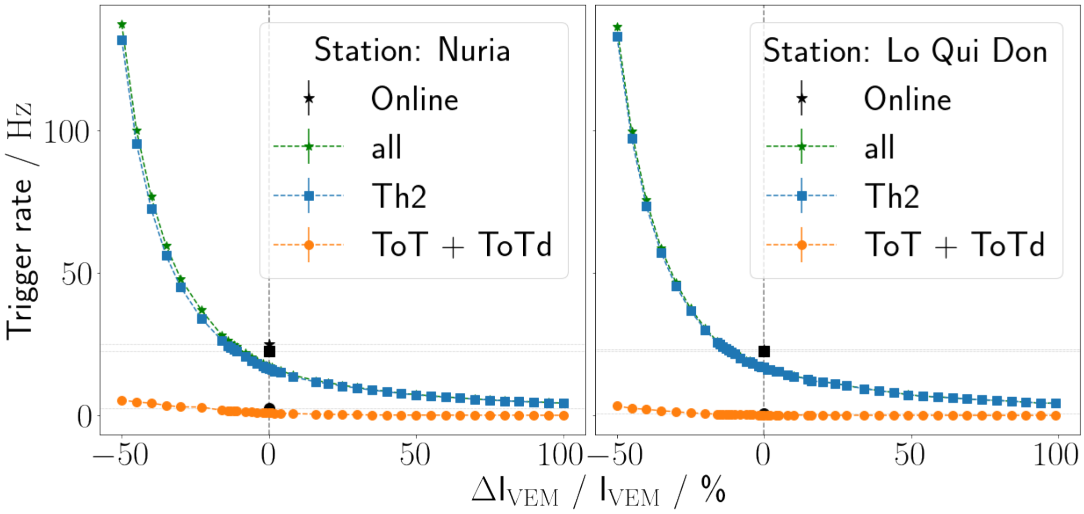


Figure 1.3: The online reported T2 trigger rate (black) does not match the calculated trigger rate. Only a decrease ΔI_{VEM} by 1/10th of the original I_{VEM} gives a close approximation of the observed rate when manually calculating trigger frequencies.

cases is not a good estimator for calibration purposes. For this reason, only random traces from the stations Nuria and Le Qui Don, measured in the March dataset are used for analysis purposes, as their $I_{\text{VEM, Rand.}}$ is stable.

As a crosscheck to verify the goodness of the approximation, the T2 trigger rate as reported by the calibration process is related to the trigger rate obtained by direct calculation over all (calibrated) random-traces. By extension, this also serves as a unit test for the classical triggers as they will be implemented in **??**. The results of this analysis are shown in **??**. As is clear from the plot, the rates calculated from the two different approaches are not in accordance. This indicates systematic errors in the calibration (a wrong implementation of trigger algorithms is disfavoured from the discussion in **??**). The errors are consistent across both considered stations. It is found that calculated trigger frequencies are ≈ 25% lower than what is taken from monitoring. It is unclear why this discrepancy occurs, as it implies that the stations do not use the same threshold values for triggering as they report.

In any case, $I_{\text{VEM, Rand.}}$ must be adjusted to reflect this. A 25% increase in trigger rate is relatable to a ≈ 10% decrease in I_{VEM} . The calibration scaling factors for random-traces thus become the values listed in **??**.

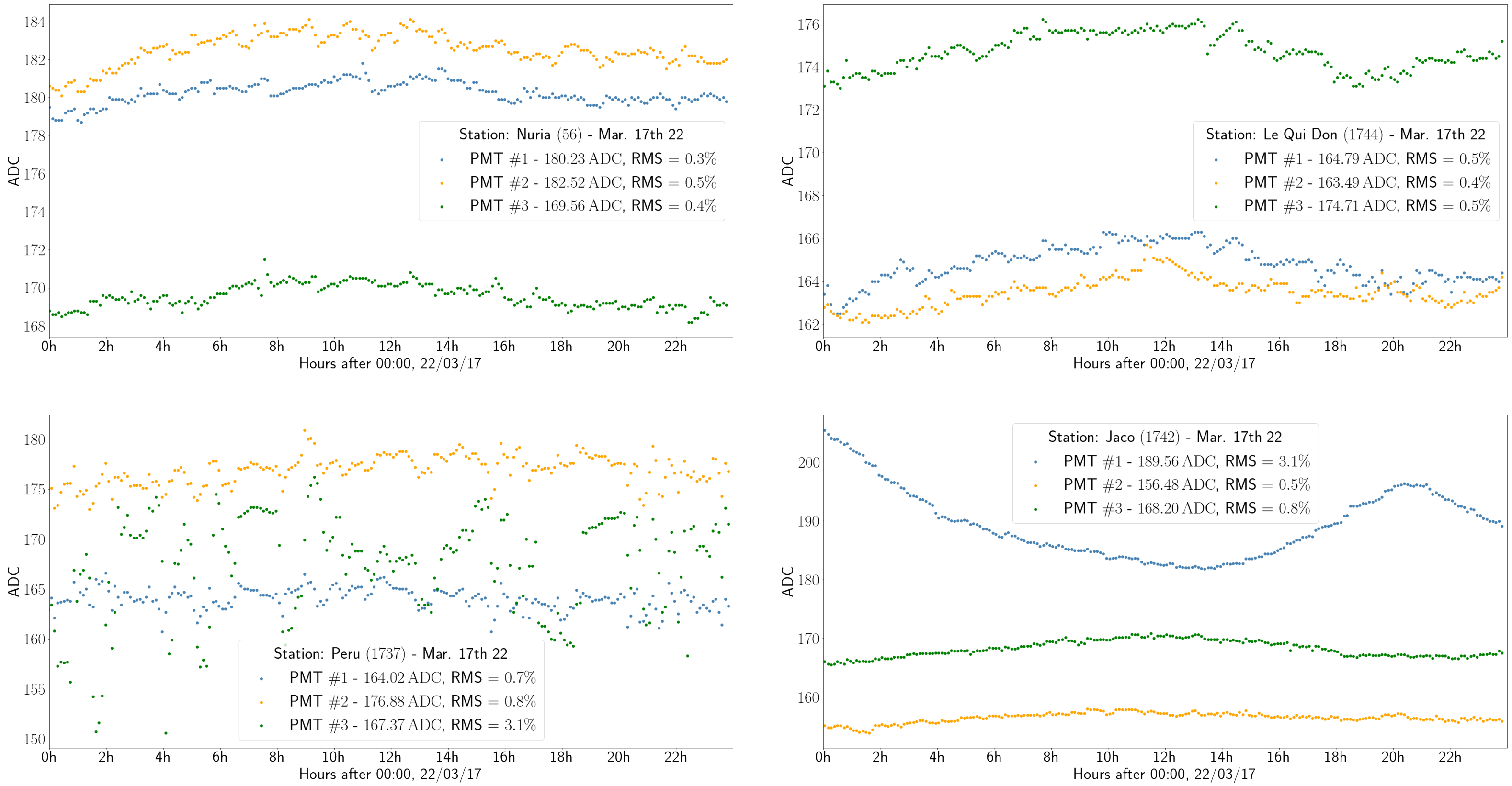


Figure 1.4: Monitoring values for the four stations available in the random-trace dataset measured in March ’22. The bottom two stations show a large variance in I_{VEM} for at least one PMT. The two stations in the first row (Nuria, Le Qui Don) are more stable (σ / μ < 1%).

1.2 Signal dataset

In the context of this work, a "signal" (as opposed to background) is *any* detector response caused by extensive air showers. Admittedly, this choice of classification is not ideal, as the particle density far from the shower axis grows sparse. Time traces recorded at those locations look very similar to ones raised by accidental muons. In any case, ramifications and possible solutions to this problem are further discussed in the following chapters.

In order to isolate the signal stemming from shower particles only, an $\overline{\text{Off}}\underline{\text{line}}$ simulation using Geant4 is executed on CORSIKA source files [**heck1998corsika**]. These are a total of 40557 simulated showers with a proton primary of energy $16 < \log(E/\text{eV}) < 19.5$. All showers are simulated using the hadronic interaction model QGSJET-II.04 [**ostapchenko2007status**].

In the process, the electronic feedback of the WCD PMTs is evaluated without any disturbance factor (see **??**). That is to say that the time trace obtained from such simulations is identically zero (in units of ADC) at any point in time where no ionizing particles are present in the WCD. An example trace that visualizes this is shown in **??**. Next, the trigger conditions both for individual stations and on the event-level are altered to trigger on everything. This step is needed in order to save all traces to the simulation output, an **A**dvanced **D**ata **S**ummary **T**ree (ADST). If this was not the case, only traces that already satisfy current trigger conditions would be written to disk. A neural network training on such data could therefore at best be as efficient as the current triggers.

The choice of this approach forces some detours in the ADST readout. Instead of extracting the VEM calibrated traces directly, individual component traces, i.e. the PMT signal caused by muons, electrons and photons individually are summed to yield a total ADC trace. Signal stemming from hadrons or other components in the cascade is neglected. This does not impose any errors in the analysis, as the hadronic component espically lays close to the shower core, where the EM- and muonic component of the shower alone should already enable easy detection. Finally, the total trace as calculated above is extracted to a more easily accessible data format alongside shower metadata like primary energy, zenith, but also SPD, and particle count in the station the trace was recorded from.

1.3 Trace building

With the componentwise traces at hand, the total trace as would be recorded in the WCD PMTs for a given event, can be constructed. First, a trace container with default UUB trace length ($2048\,\text{bins} \cdot 8.3\,\text{ns}\,/\,\text{bin} = 17.07\,\mu\text{s}$) and three components per bin (the three WCD PMTs) is initialized with all values equal to zero. Next, an arbitrary random-trace is selected as baseline. Since the FPGAs fundamentally count in the integer domain, the ADC data in the random-trace contains only whole numbers. As is, this wouldn't correctly model rollover when adding integer random-traces to

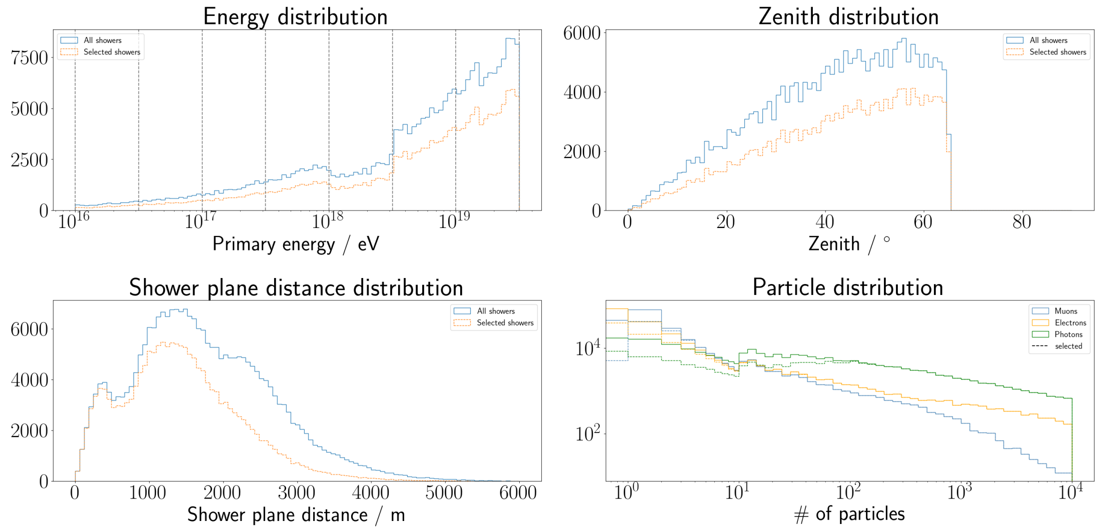


Figure 1.5: The distribution of primary energy (*top left*), shower inclination (*top right*), separation from shower axis (*bottom left*), and number of particles in a tank (*bottom right*) is histogrammed for each trace available in the dataset. Since showers with higher primary energy reach a larger number of stations, traces from showers with increasing energy are overrepresented in this dataset. This must be taken into account when generalizing results that are founded on this dataset. The dashed lines in each subplot represent the population of traces that deposit a signal $S > \mathrm{VEM}_{\mathrm{Ch.}}$ in the tank.

floating point simulation information. While two ADC signals by themself might not exceed the threshold to cross to the next higher value, their sum would might; $\lfloor 0.7\,\mathrm{ADC} \rfloor + \lfloor 0.4\,\mathrm{ADC} \rfloor = 0\,\mathrm{ADC} \neq 1\,\mathrm{ADC} = \lfloor 0.7\,\mathrm{ADC} + 0.4\,\mathrm{ADC} \rfloor$. To account for this, uniformly distributed random numbers from 0 (inclusive) to 1 (exclusive) are added to the random-trace.

Furthermore, the $I_{\mathrm{VEM,\,Rand.}}$ from random-traces (c.f. **??**) will in general be different from I_{VEM} simulated by Offline ($I_{\mathrm{VEM,\,Off.}} = 215.781\,\mathrm{ADC}$ compare [**offlineSource**]). Thus the random-trace must be scaled by a factor $\frac{I_{\mathrm{VEM,\,Off.}}}{I_{\mathrm{VEM,\,Rand.}}}$ before being added to the container.

If desired, accidental muons can be added to the trace container as well. This is done either by directly specifying a number of random injections, or throwing the dice according to the injection frequency specified in **??**. If the number of accidental muons is nonzero, a sample of random background traces from [**DavidBackgroundSim**] is drawn and each sample added to every PMT at a random uniform position somewhere in the trace. **??** shows an example where five muons are injected into the trace.

Last, the actual shower signal is added to the trace container. In principle, it can be added at any random position, similar to the random injections. However, for continuity reasons and ease of comparing to plots generated with other software, the latch bin for signal insertion is hardcoded to be the same as in Offline, bin 660. Since

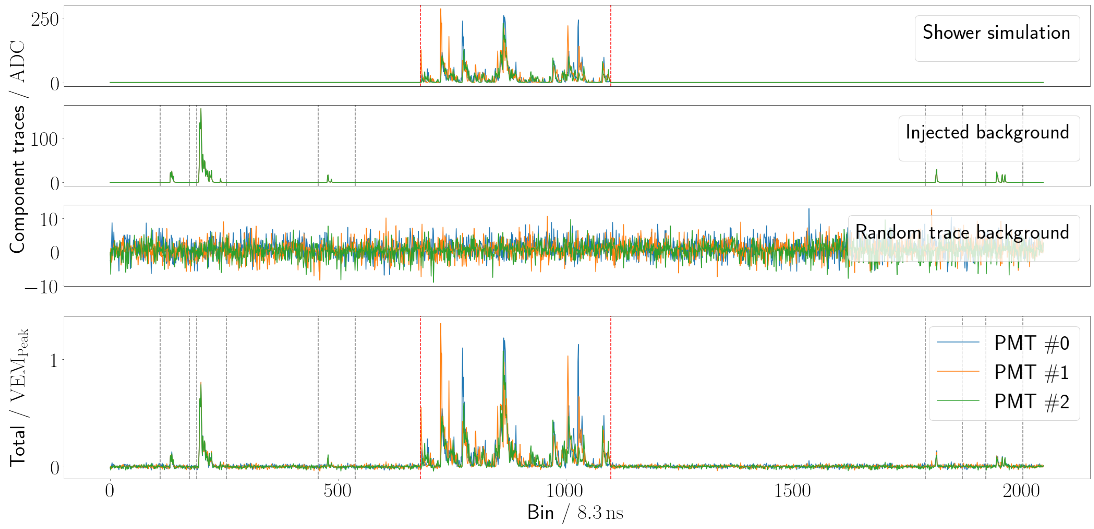


Figure 1.6: The individual component traces (in units of ADC, top three plots) make up the eventual VEM trace (in $\mathrm{VEM}_{\mathrm{Peak}}$, bottom plot). The dashed red (gray) lines signify where in the time trace the shower signal (injected muon signal) is located.

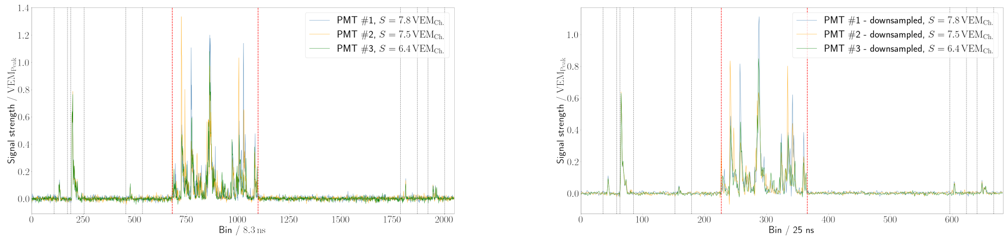
otherwise the data is in the correct ADC format, no further manipulation of the data is necessary.

The trace container now holds all necessary components together, but remains in units of ADC. To convert to $\mathrm{VEM}_{\mathrm{Peak}}$, each bin in each PMT trace is floored to mimic the FPGA digital counting, and divided by the appropriate scaling factor I_{VEM} times a correction factor that stems from a bias in the online peak estimation algorithm (see [**bertou2006calibration**]). If traces need to be UB compatible the FADC bins must be filtered and downsampled in an intermediate step. This influences the scaling factor $I_{\mathrm{VEM,\,compat.}}$ in a major way (compare **??**). If the so-called full-bandwidth, UUB time trace is analyzed, the appropriate factor becomes $I_{\mathrm{VEM,\,Off.}}$.

1.3.1 Filtering and downsampling

Altough the triggers discussed in the next chapter are meant to function completely autonomously in the SD field, their implementation requires some prior knowledge of the signal one desires to detect. For their use in the Auger observatory, several hyperparameters such as the thresholds of the Th-Trigger, or the window size of the ToT-trigger have been determined in studies ([**bertou2006calibration**], [**triggerSettings**], [**ToTtriggerSetting**]).

These studies were conducted using the predecessor, the **U**nified **B**oard (UB), of the hardware that is being installed during the AugerPrime upgrade of the observatory. Most importantly, the UUB has a sampling rate that is three times larger (120 MHz) than that of UB electronics (40 MHz). Not only does this raise the number of bins in a standard time trace from 682 to 2048, but also drastically reduces the efficiency (in particular for ToT-like triggers) of the above discussed algorithms. Whereas a new bin



(a) UUB time trace

(b) UB time trace

Figure 1.7: (a) A simulated signal as it would appear to UUB electronics. The ionizing particles originating in the extensive air shower hit the tank around bin 660 (≈ 5.5 µs). **(b)** The same signal but filtered and downsampled to emulate UB electronics.

is measured every 25 ns in a UB station, the triggers would receive a new input every ≈ 8.3 ns in a UUB setting. If the window size of e.g. the ToT trigger were to remain constant, only a third of the original signal becomes available for a given window frame.

The modus operandi elected by the Pierre Auger collaboration to circumvent this problem is to emulate UB electronics using the UUB electronics. This means that measured FADC bins are to be filtered and downsampled before any trigger runs over them. Software implementations by which this is achieved are listed in **??**. The effect the filtering and downsampling has on measured data is visualized in **??**.

While the features of the time trace largely remain intact, the absolute signal strength decreases due to a smearing effect imposed by filtering. Overall, this amounts to a 30% difference in amplitude between UUB full-bandwidth traces and their filtered and downsampled counterpart. Since both measurements are derived from the same signal in the WCD though, this implies that I_{VEM} must be adjusted by 30% as well if traces are to be filtered and downsampled. This results in a compatibility scaling factor $I_{\mathrm{VEM,\,compat.}} = 163.235$ ADC [**OfflineSource**].

Recent contributions within the Auger collaboration ([**nitzTriggers**, **quentinComparison**]), and to some extent also this work have shown that issues arise in the comparison of **L**ateral **T**rigger **P**robabilities (LTPs) that are run in this compatibility mode. Namely, the UUB trigger efficiency (where full-bandwidth traces are filtered and downsampled) is lower than that of UB stations. This implies that the filtering and downsampling algorithms in **??** either make imprecise assumptions about the station electronics, or $I_{\mathrm{VEM,\,compat.}}$ or the trigger thresholds themselves need to be adjusted further. This fact has to be kept in mind when discussing results and comparing lateral trigger probabilites from classical triggers and neural networks.

1.3.2 Sliding window analysis & Labelling

As will become apparent in **??**, three of the four trigger algorithms operate by examining only a window of measurement info, rather than evaluate the whole time trace as it is constructed in **??**. In a similar fashion, it is reasonable to assume neural networks do not need to receive the $3\,(\mathrm{PMT}) \cdot 2048\,(\mathrm{UUB\ bins}) = 6144$ input values from the entire trace to make an informed choice of whether or not a given signal stems from an extensive air shower.

For this reason, samples from the time trace are drawn via a sliding window analysis. A number of n_{bins} are extracted from the trace, to be analyzed by some classification algorithm. In order to not select the same information repeatedly, the window is moved by n_{skip} bins forward and the process can begin anew. Unless explicitly specified otherwise, the hyperparameters in this sliding window analysis are set as

$$n_{\mathrm{bins}} = 120, \qquad n_{\mathrm{skip}} = 10. \tag{1.1}$$

Whether or not a specific window contains signal from an extensive air shower - which is important for labelling data in the context of neural network training - is a simple exercise. The modular approach in **??** allows to simply check for nonzero bins in the shower signal component of the trace. In practice, upon creating a new combined trace, the first and last positive bin in the shower component are identified. This is e.g. visualized with dashed red lines in **??**. If any overlap - even just a single bin - exists between the sliding window and this signal region, the extracted window is consequently labelled as signal. If this is not the case, the window is labelled as background.

Of course, quality cuts can be applied, and a decision to count a given trace window can be made individually. This is further discussed in **??**.

2 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 **??**.

2.1 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

$$\epsilon = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{2.1}$$

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.

[to do: revisions regarding flux scaled efficiency]

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{TP} + \mathrm{FP}}. \tag{2.2}$$

In this work, the trigger rate on random-traces f_{Random} is used as an alias for the FDR. Of course, this is not completely accurate, as these to a small fraction contain signal from air showers. In any case, the distinction does not matter. Any trigger algorithm must have a sufficiently low trigger frequency on measured data - be it extensive air showers or background events - as to not overload CDAS readout capabilities.

Consequently a pseudo-score can be assigned to each classification algorithm in order to compare them. This score a is given by **??**.

$$a = \frac{\epsilon}{f_{\mathrm{Random}}\ [\mathrm{Hz}]} \tag{2.3}$$

The physical interpretation of this variable is not straight forward. If labels were known for random-trace datasets, a would be equivalent to the **S**ignal-to-**N**oise **R**atio (SNR) of the classifier. However, since f_{Random} and the T2 efficiency across all primary energies and zenith angles, must be determined using two different sets of data, only a (positive) correlation between the two exists.

In conclusion, all algorithms should maximize a, i.e. boost the T2 trigger efficiency, while keeping the random-trace trigger rate as low as possible.

2.1.1 Lateral Trigger Probability (LTP)

Ultimately, all test statistics constructed by classical station triggers (ignoring MoPS, c.f. **??**) are correlated to 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 **??**.

$\overline{\text{Off}}\underline{\text{line}}$ 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 the probability of 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

$$\text{LTP}\,(r) = \begin{cases} \dfrac{1}{1 + \exp\left(\frac{r-R_0}{\sigma_R}\right)}, & r \leq R_0 \\ \dfrac{1}{2}\exp\left(C(r - R_0)\right), & r > R_0 \end{cases} \tag{2.4}$$

In **??**, R_0, σ_R 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 $1/\sigma_R$ 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 σ_R are not finetuned as indicated in **??**.

$$\lim_{r\to R_0^+} \frac{\partial\,\text{LTP}(r)}{\partial r} = \frac{C}{2} \overset{!}{=} \frac{1}{4\sigma_R} = \lim_{r\to R_0^-} \frac{\partial\,\text{LTP}(r)}{\partial r} \tag{2.5}$$

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 **??**.

$$\text{LTP}^*(r) = \min\left(1,\, \epsilon^* \left(1 - \frac{1}{1 + e^{-\frac{r-R_0}{\sigma_R}}}\right)\right) \tag{2.6}$$

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

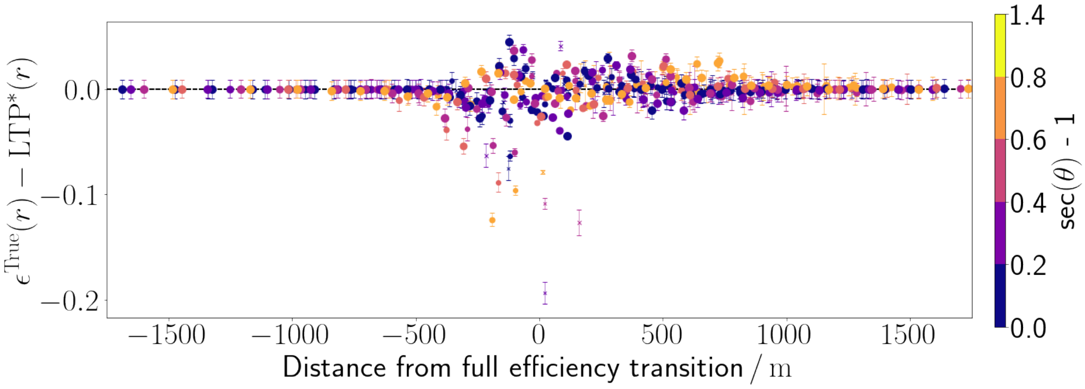


Figure 2.1: The residuals from comparing ϵ^{True} to $\mathrm{LTP}^*(r)$ vanish at large (small) r. No systematic bias is observed in the transitional region around R^*. Large outliers are caused by the dataset being limited in size at low energies (marked with an x).

extensive air showers. Espically neural network triggers might be sensitive to only a subset of showers. This is reflected in the latter form, $\mathrm{LTP}^*(r)$, by introducing an additional fit parameter, the pseudo-efficiency $0 \leq \epsilon^* < 2$. 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 function is guaranteed to be continously differentiable in R_0. For values $\epsilon^* > 1$ this is replaced with a kink at $R^* = R_0 - \frac{\log\left((1-1/\epsilon^*)^{-1}-1\right)}{\sigma_R}$, where $\mathrm{LTP}^*(r)$ would exceed 1 if not for clipping. There exists some physical motivations for this however. Due to the 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 $\mathrm{LTP}^*(r)$ does - at least explicitly - not. As a result, one could expect a bias, where $\mathrm{LTP}^*(R^*)$ 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 is offered in **??**. Still, results at low energies ($\log(E\,/\,\mathrm{eV}) < 17.5$) must be taken with a hint of skepticism, as only very little data is available for such showers.

Bayesian folding

Given a time trace of the form in **??**, it is easy to determine whether or not a given trigger algorithm analyzing this trace would raise a T2. The trigger probability obtained this way must however not be confused with the lateral trigger probability discussed above.

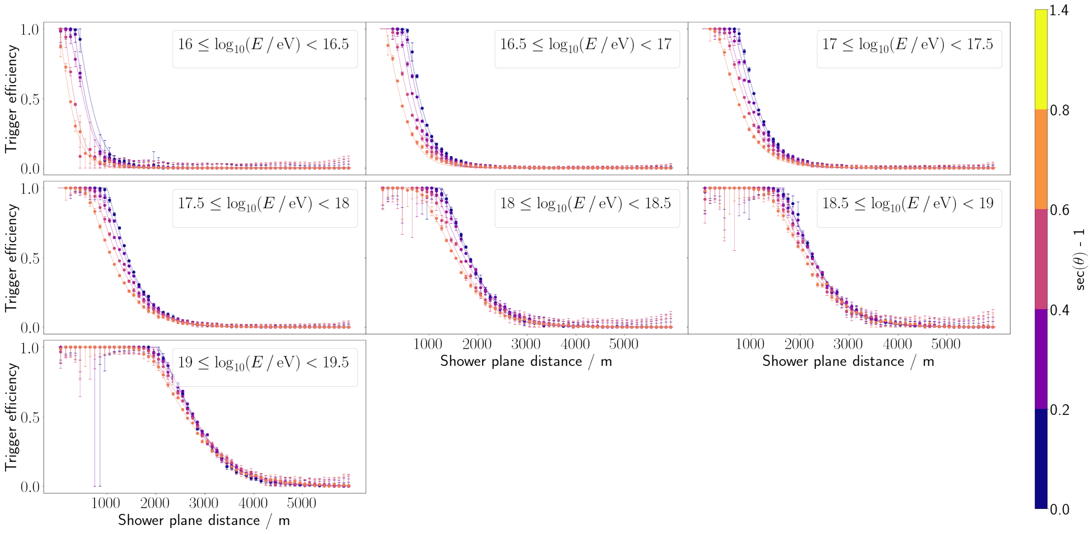


Figure 2.2: The probability of a station receiving a particle in relation to its' separation from the shower core axis. The probability at a constant SPD increases with increasing primary energy, but decreases with increasing zenith angle.

It is the conditional probability of a T2 given that a tank receives an air shower signal, $P(\mathrm{T2}\,|\,C_1)$. The complete LTP can be calculated in the following form:

$$\mathrm{LTP} := P(\mathrm{T2}) = P(\mathrm{T2}\,|\,C_1) * P(C_1). \tag{2.7}$$

$P(C_1)$ in **??** is the **L**ateral **P**article **P**robability (LPP) and quantifies the chance of a station receiving a signal from an extensive air shower. It is the probabilistic interpretation of the lateral distribution function (c.f. **??**). The LTP of an ideal classifier that is able to identify individual particles would be equal to the LPP.

For this work, the LPP is calculated by comparing the simulated stations in **??** to a catalog of known stations in the vicinity of the shower core. The ratio of simulated stations divided by all stations at a specific SPD is the LPP at a given (slice of) energy E and zenith θ. A function like in **??** is fitted to the data to extrapolate values at arbitrary SPD. The result of this analysis for all energies and zenith angles is shown in **??**. The best fit parameters are listed in **??**.

The comparison between the LTP gathered via Bayesian folding is compared to the $\overline{\text{Off}}\underline{\text{line}}$ station counting approach in **??**. This is done for every classical trigger individually. It is found that no remarkable difference between both results exists. As a consequence, this method of evaluating the LTP allows for an easy comparison of neural network triggers to the algorithms discussed in the following.

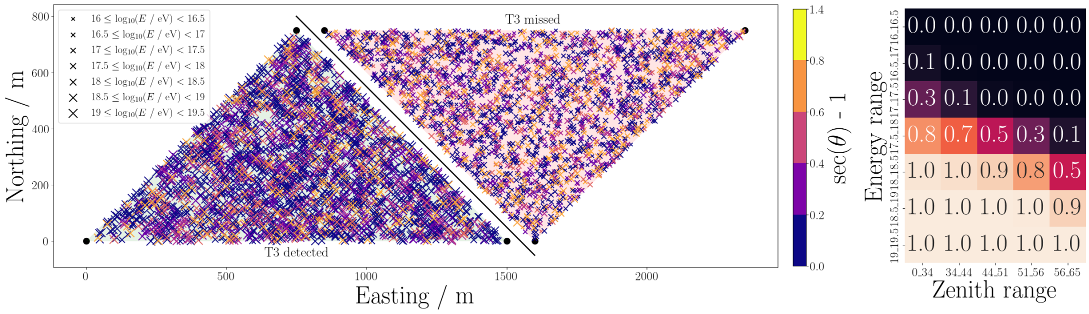


Figure 2.3: (*Left*) An example of randomized core positions color- and size-coded according to zenith and primary energy. Showers that raised a T3 are shown in the green unit triangle. The ones that do not are displayed in the red part. The stations at the vertices of the SD unit triangle are shown as black circles. (*Right*) The ratio of triggered vs. all showers, with respect to primary energy and shower inclination.

2.1.2 Calculation of T3 efficiency

When transitioning from station- to event-level, the important variable becomes the T3 efficiency. It states the probability of a shower not only being detected in individual stations, but also identified as such by the CDAS T3 triggers. In the end, the only measured data that is available comes from extensive air showers which passed this final hurdle.

With the lateral trigger probability at hand, a simple Monte-Carlo simulation can recover the T3 efficiency. Recall that the probability of a single station i detecting a shower at distance r_i, with energy E and arrival direction (θ, ϕ) is given by **??**.

$$\mathrm{LTP}^*(r_i) = \min\left(1,\, \epsilon^*(E,\theta)\left(1 - \frac{1}{1 + e^{-\frac{r_i - R_0(E,\theta)}{\sigma_R(E,\theta)}}}\right)\right). \tag{2.8}$$

In the simplest case, this results in a T3 trigger if the three closest stations raise a T2 within 11 µs of one another. By simulating a random core position and determining the detector response, one can calculate the event detection numerically. The approach presented here does not take into account timing differences resulting from the finite propagation speed of the shower front. Instead, it is assumed that all stations simultaneously receive a signal. This is only accurate if the primary particle initiated a perfectly vertical cascade infinitely far away from the SD. However, due to the permissivity of the T3 triggers such considerations need not be accounted for. Even in the most suboptimal case of a horizontal shower ($\theta = 90°$), the furthest station in the triangle receives a signal latest around $\frac{1.5\,\mathrm{km}}{c} = 5\,\mu\mathrm{s}$ after the closest one. Exemplary simulations and resulting T3 efficiencies for classical triggers are shown in **??** and further discussed in **??**.

2.2 Implementation

2.2.1 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 $S \leq 3.20\,\mathrm{VEM}_{\mathrm{Peak}}$ is immediately forwarded to CDAS, whereas a signal $1.75\,\mathrm{VEM}_{\mathrm{Peak}} \leq S < 3.20\,\mathrm{VEM}_{\mathrm{Peak}}$ only raises a Th-T1 trigger. The algorithm is insensitive to signals that do not exceed at least $1.75\,\mathrm{VEM}_{\mathrm{Peak}}$ 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 $\mathrm{VEM}_{\mathrm{Peak}}$) are updated according to **??**.

Table 2.1: 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

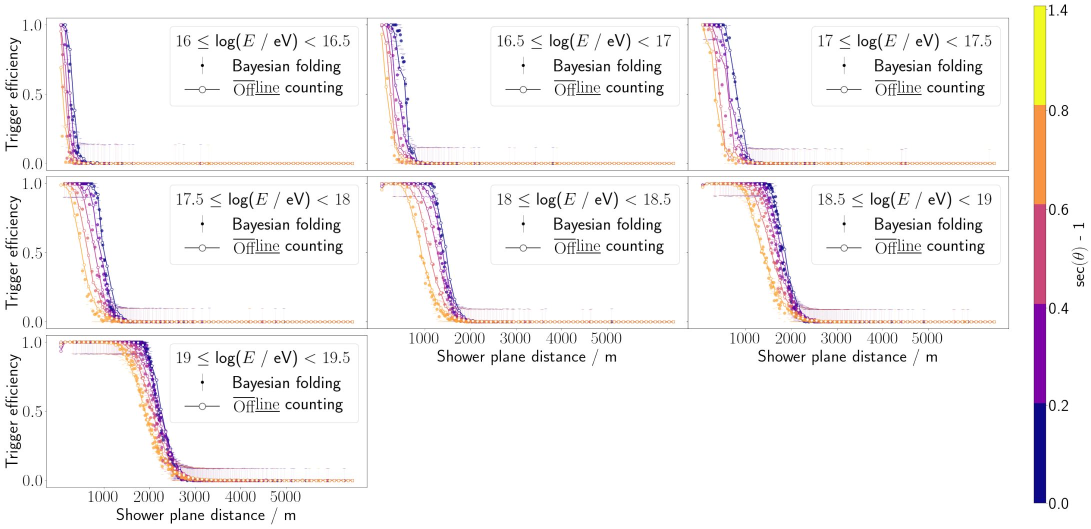


Figure 2.4: The lateral trigger probability for the threshold T2 trigger. Shown is the comparison between the different procedures discussed in **??**. Both results agree across all considered primary energies and zenith angles.

Performance

The average trigger rate for the Th-T2 trigger per station is *defined* to be ≈ 20 Hz (c.f. **??**). Comparing this to the nominal T3 trigger rate at CDAS level (≈ 0.03 Hz **[abraham2010trigger]**) over the entire array, it becomes obvious that a lot of background events pass this threshold. Consequently, the trigger has a very high false discovery rate on a station to station level FDR $\approx \frac{20\,\text{Hz}-0.03\,\text{Hz}/1600}{20\,\text{Hz}+0.03\,\text{Hz}/1600} = 0.999998$.

The efficiency of the threshold trigger is comparably poor. Only every fifth trace ($\epsilon = 0.2$) is detected as such. This number must however be taken with context. In **??**, a signal is considered to be any kind of detector response from an extensive air shower. This includes single muons injected into WCDs faraway from the shower core. As such, the dataset that triggers are being tested on contain a lot of information algorithms were designed to ignore. This drops the efficiency considerably. Nevertheless, it serves as a gauge to compare this trigger to the ones discussed on the following pages.

While this may seem like an indigent method of shower detection, the threshold trigger is invaluable in the search for neutrino cosmic rays. The EM component of such showers is heavily attenuated due to their inclination ($\theta \geq 65°$). Only the muonic component reaches the SD detector. The threshold trigger ensures the array is sensitive to such events, at the cost of a high background noise.

The lateral trigger probability for the Th-T2 type trigger is shown in **??**

2.2.2 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 $\mathrm{VEM}_{\mathrm{Peak}}$. 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 $\mathrm{VEM}_{\mathrm{Ch.}}$ 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

The nominal operation of the ToT algorithm sees a trigger rate of 2 Hz. While this still corresponds to a relatively large FDR, the signal to noise ratio is at least an order of magnitude better than Th-T2, simply by arguments of trigger frequency. Moreover, coincidences between neighbouring stations are likely to be extensive air showers. Events selected from ToT issued T3s have a purity of 90%, and are the main detection channel for showers with inclination $\theta < 60°$ [**abraham2010trigger**]. The ToT trigger itself has an efficiency of $\epsilon = 0.3969$ when evaluated over the neural network training dataset. The trigger probability w.r.t shower plane distance is shown for different primary energies and different arrival directions in **??**.

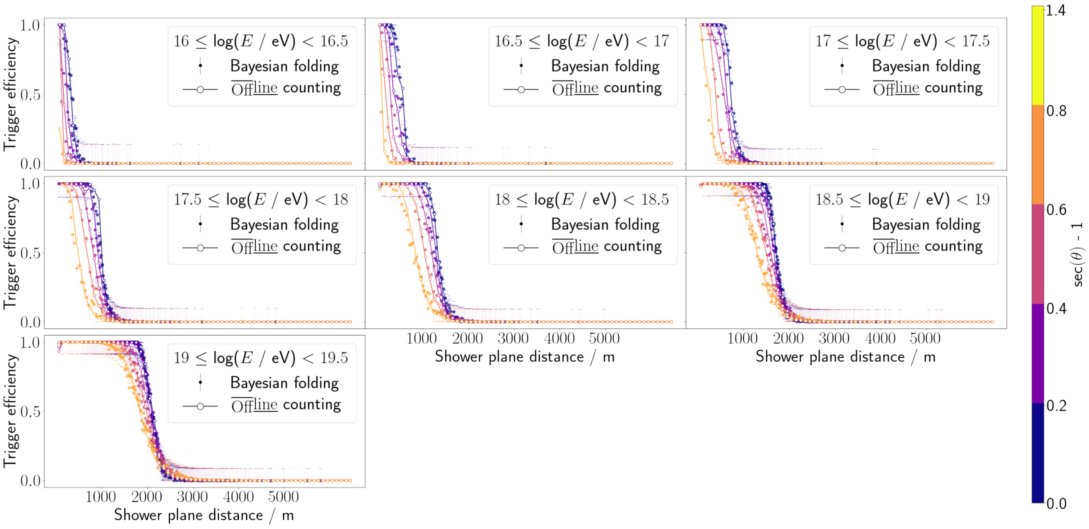


Figure 2.5: The lateral trigger probability for the time-over-threshold trigger for Bayesian folding and Offline station counting.

2.2.3 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 **??**.

$$d_i = (a_i - a_{i-1} \cdot e^{-\Delta t/\tau}) \,/\, (1 - e^{\Delta t/\tau}) \tag{2.9}$$

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

The performance of the ToTd trigger is very similar to that of the ToT discussed in **??**. It posesses a comparable efficiency $\epsilon = 0.4027$ to its' convoluted counterpart. The lateral trigger probability for this algorithm is shown in **??**.

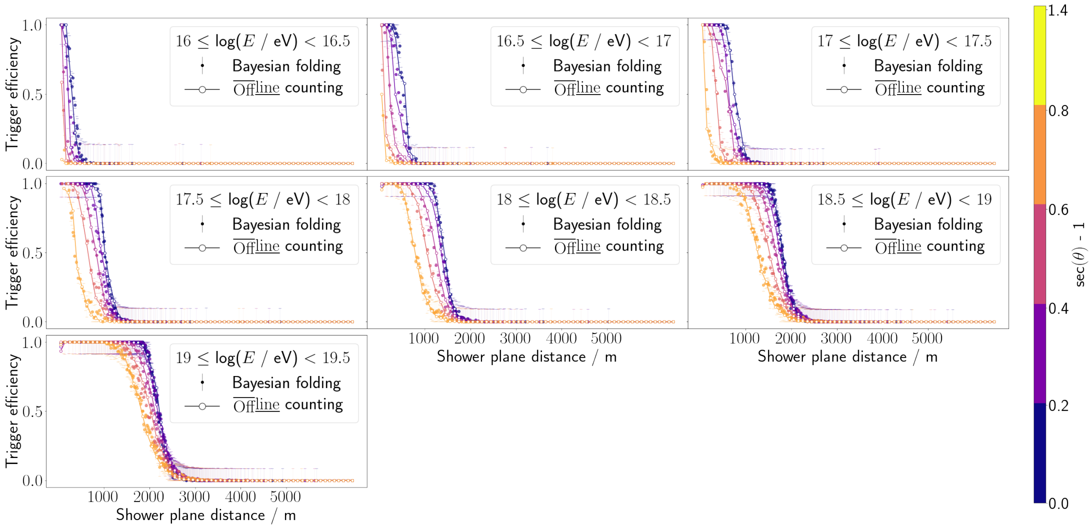


Figure 2.6: The lateral trigger probability for the time-over-threshold-deconvoluted trigger. Errorbars are plotted transparently in order to not overcrowd the figure.

2.2.4 Multiplicity of Positive Steps (MoPS)

The **M**ultiplicity **of** **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

$$n_{\text{skip}} = \lfloor \big(\log_2(\Delta y) + 1\big) - 3 \rceil \tag{2.10}$$

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 $y_{\text{min}} = 3\,\text{ADC}$ (to filter random fluctuations), but does not exceed $y_{\text{max}} = 31\,\text{ADC}$ (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 when comparing performances of classical triggers to those in **??**. The implications of this choice are layed out in the following paragraph.

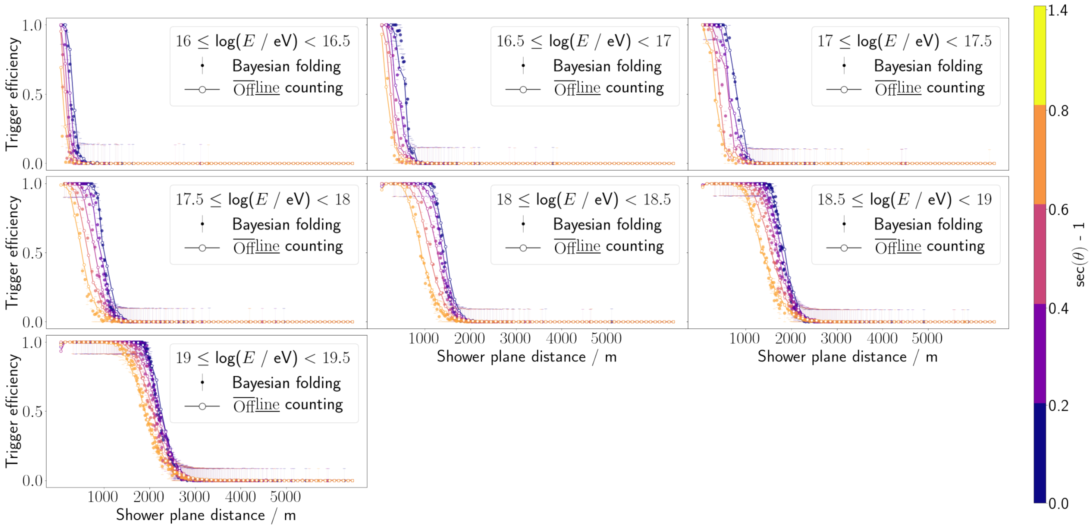


Figure 2.7: The lateral trigger probability for the combination of all previously discussed trigger algorithms.

Performance

By examining monitoring data, it follows that $1 - 2$ MoPS triggers are issued per station each minute. This corresponds to a trigger rate of 0.02 Hz to 0.03 Hz. This is orders of magnitude lower than other discussed trigger mechanisms. While the MoPS trigger consequently can be seen as a relatively noise free trigger, events in which a MoPS is critically required to form a T3 are extremely sparse. From a total of 20000 (simulated) showers at lower energies, none had a three-fold coincidence where at least one station only detected a MoPS T2 trigger. Considering this result, it is expected that T3 efficiencies are largely independant of the fact whether MoPS is considered or not.

2.2.5 Combined performance

In the field, all above discussed algorithms are run simultaneously. That is, a T2 trigger is issued whenever any of the Th-T2, ToT, ToTd, or MoPS trigger become active. This results in an overall trigger rate of roughly 22 Hz to 23 Hz, with a combined efficiency of $\epsilon = 0.4070$. The combined lateral trigger probability with respect to shower plane distance is shown in **??**. The overall T3 efficiency as calculated in **??** is shown in **??**. As can be seen in the plot, detection of air showers is guaranteed at primary energies of around 10^{18} eV and upwards.

3 Neural network triggers

[to do: write introductory shit here?]

3.1 Motivation

The station-level triggers in the previous chapter have been shown to perform well enough for the science case of the Pierre Auger Observatory. However, it has also been concluded that a lot of potential data, espically at low energies is ignored. This is by intention in order to keep DAQ readout at feasible levels.

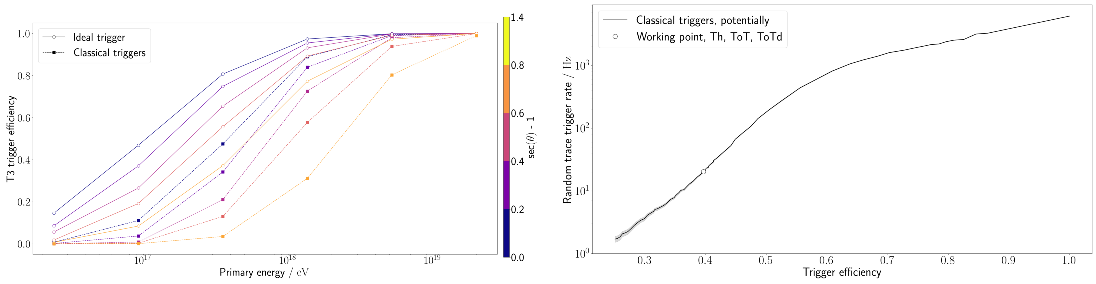
Attempts at improving the overal efficiency of the SD triggers can be made. This is only possible to a certain level. At lowest energies the particle cascade is not big enough to warrant coincident triggers in at least 3 WCD stations. As per **??**, the lateral trigger probability a given classification algorithm can maximally achieve is given by the LPP (c.f. **??**). The T3 detection probability of such an ideal trigger, and consequently the maximal efficiency for an array with 1.5 km spacing is compared to the efficiency of classical triggers in **??**.

Of course, efficiency can be improved simply by adjusting trigger thresholds of the algorithms in **??**. However, the more lenient these thresholds are, the more background events will be detected. This quickly results in trigger rates that are unmanagable for the infrastructure at the Pierre Auger observatory. The probability with which time traces correctly raise a T2 is shown alongside the resulting random-trace trigger rate for different thresholds of classical algorithms in **??**.

Ideally, neural network architectures developed in this chapter should undercut the random-trace trigger rate of classical triggers, while retaining an overall higher accuracy. That is, they lay below and right of the operating point in **??**. For any algorithm that achieves this, the corresponding LTP will be greater than that of classical triggers, resulting in higher event detection efficiency, while not exceeding the bandwidth limitations of the underlaying hardware.

3.2 Design considerations & Implementation

The hardware specifications at the FPGA level, where trigger conditions are currently checked, are limited. For this reason, NN architectures should be kept as simple as possible. Most importantly, the number of weights, biases and other trainable



(a) T3 efficiency

(b) Classical trigger potential

Figure 3.1: **(a)** Comparison of an ideal trigger sensitive to any shower signal from primary energies $E \geq 10\,\mathrm{PeV}$ to classical triggers. **(b)** The noise level over calculated efficiency for classical triggers. The tail ends of the potential curve are calculated by adjusting the trigger thresholds from +250% to −95% of the nominal values.

parameters will need to be hardcoded into station software. Because of minimal available storage space, this number needs to be kept low.

This immediately disqualifies powerful candidates like autoencoders or transformers (compare **??**) from consideration, due to their size. Only simple dense-, convolutional-, and recurrent neural networks are viable contenders that could theoretically be implemented in the SD electronics.

The python library TensorFlow [**tensorflow2015-whitepaper**] is used as a backend to implement the individual classifiers. All discussed architectures are built and trained with the release version 2.8.4 [**tensorflowversion**]. Adjustments to the trainable parameters are calculated according to a momentum-based stochastic gradient descent (Adam [**kingma2014adam**]) on a batch level. In this context, a single batch is made up of all traces that are recorded from a single air shower event. Since batch size grows quickly with increasing energy, a generative approach, where traces are created (c.f. **??**) at runtime upon requirement, is used in building training data in order to make the process as RAM-efficient as possible. This has important implications. As trace building relies heavily on randomization, the actual training data will not be the same if the random number generators are not seeded beforehand. This has been taken into account. All networks are - unless specifically stated otherwise - trained and validated using the same signal input data.

3.3 Choice of input data

3.3.1 Prior probability of Signal

The flux of cosmic rays with energies exceeding the proton knee is tiny ($\mathcal{O}(1\,\mathrm{m}^{-1}\,\mathrm{yr}^{-1})$ [**dembinski2017data**]). While the size of the SD guarantees decent exposure over the entire array, an individual station will mostly measure background. In fact, the prior probability p of encountering such events in a given random time trace is roughly 1 in 1 000 000. Of course, an accurate prior during training would thus result in poor network

Figure 3.2: A very faint positive slope ($m = 0.02 \pm 0.01$) is observed when relating prior probability to trigger sensitivity (blue dots). This could however be attributed to statistical fluctuations in the training fit. An ensemble of networks trained on the same data and with the same prior shows a comparable spread (orange dots).

performance. On the notion of a broken clock being correct twice a day, a naive classifier can have near perfect accuracy by labelling every input as background. Such behaviour is not desired. The prior probability must be artificially inflated. The influence of prior probability on subsequent trigger sensitivity is shown in **??**. No strong correlation between prior and network performance is found in the range $0.05 \leq p \leq 0.95$. As long as the fraction of signal over entire training set is statistically relevant, the network learns to discern between the signal and background. Nevertheless, a conservative prior of $p = 0.5$ is picked for the following analysis.

3.4 Performance

1. Have problem
2. Throw math at problem
3. ???
4. Profit

[to do: write this]