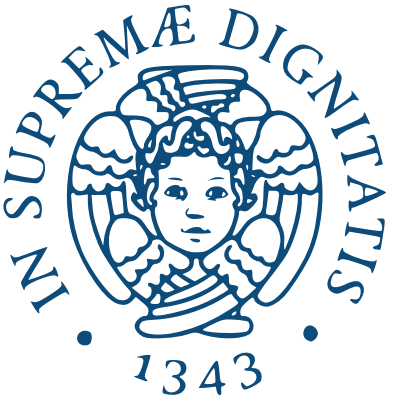
Master's degree thesis: Online processing of the large area SiPM detector signals for the DarkSide20k experiment

Giacomo Petrillo
Supervisors: Eugenio Paoloni, Simone Stracka
University of Pisa

May 31, 2021



Contents

Introduction

DarkSide20k is a planned dual-phase liquid argon (LAr) time projection chamber (TPC) designed to detect dark matter, the successor to DarkSide-50. It will be the largest detector of its kind, with 20 metric tons of argon in the fiducial volume. The predicted resulting upper bound on the spin-independent WIMP-nucleon scattering cross-section, in case of no discovery, is $\approx 1 \times 10^{-47}\,\mathrm{cm}^2$ at $1\,\mathrm{TeV/c}^2$ WIMP mass, to be compared with the current best limit $\approx 1 \times 10^{-45}\,\mathrm{cm}^2$ by XENON1T.

DarkSide20k is part of a larger effort to improve the sensitivity to elastic nuclear recoils down to the ultimate goalpost of coherent neutrino scattering. The expected number of neutrino recoils in the 100 t yr exposure of DarkSide20k is approximately 1. It is thus worthwhile to pursue this effort since we are indeed close to the conclusion. If a WIMP-like signal was not found before then, new detection techniques would be needed to continue the WIMP search, in particular, sensitivity to the recoil direction would permit to push down further the limits.

In the predecessor DarkSide-50, 90 % of the neutron background was traced to originate in the photomultiplier tubes (PMTs). To scale from the 50 kg of argon in DarkSide-50 to the 20 tons of DarkSide20k, it is then necessary to replace the PMT with a more radio-pure photodetector. A key enabling technology is thus the silicon photomultiplier (SiPM), which has better radio-purity. Of the various advantages over the PMT, another important one is single-photon resolution, needed because the isotropic scintillation signals will be very dispersed in the large TPC of DarkSide20k, with often only one photon hitting a photodetector.

The readout of SiPMs, however, requires more complicated processing than PMTs. First, to equip its uniquely large photodetection area, DarkSide20k will employ large $25\,\mathrm{cm}^2$ photodetectors. A SiPM of this size produces an high amount of floor electrical noise. Furthermore, the SiPM suffers from correlated noise, i.e., secondary output pulses induced by other pulses instead of by incident light or dark count. The generation of secondaries is recursive, causing loss of dynamic range through pile-up and leading eventually to saturation if not kept under control.

In this thesis we present reconstruction and characterization studies on the photodetector modules (PDMs) that will be used in the TPC. These studies quantify the effects and the amount of noise in the SiPMs and are primarily meant as a support to the definition of the first stages of the online processing chain.

Each PDM consists of a $25\,\mathrm{cm}^2$ 6×4 matrix of individual SiPMs, and a front end board (FEB) that pre-amplifies and sums analogically the output of the SiPMs. Thus qualitatively its behavior is akin to a single large SiPM. The SiPM has Geiger-mode single photon response, i.e., each detected photon produces one fixed amplitude pulse. The pulse looks like a sharp peak, $\approx 20\,\mathrm{ns}$, followed by a rather long exponential tail, $\approx 1\,\mu\mathrm{s}$. SiPMs have three kinds of noise: 1) stationary electric noise, which scales with the square root

of the area; 2) a dark count rate (DCR) of pulses independent of incident light that scales with the area; 3) the correlated noise produced by primary pulses, which contributes a factor proportional to the DCR and photon pulses.

The first two stages in the readout chain will be the digitizers and the front end processors (FEP). The digitizers find candidate pulses, and for each one send a slice of waveform to the FEP, where the final identification of pulses is decided. The performance of these stages is mainly determined by the electric noise, characterized with the signal to noise ratio (SNR), which is the ratio of the amplitude of pulses over the noise standard deviation. It influences the fake rate, i.e., the rate of random oscillations high enough to be mistakenly identified as pulses, and the temporal resolution of pulse detection.

By applying linear filters to digitized waveforms acquired from the PDMs illuminated by a pulsed laser, both in a testing setup at Laboratori Nazionali del Gran Sasso (LNGS) and in the small prototype TPC "Proto0", we study the noise parameters of single pulse detection: SNR, temporal resolution, fake rate.

We consider 1) an autoregressive filter, which uses the least possible computational resources, 2) a matched filter without spectrum correction, which gives almost optimal performance, 3) a moving average, which is a compromise between simplicity and performance. Simple filters are needed on the digitizers, which must process all the incoming data, while the FEP will probably use the optimal filter. We also study the baseline computation and the filter length.

Then using a custom peak finder algorithm we measure the DCR and study the correlated noise, which consists of additional pulses produced recursively by each pulse, divided in two main categories: afterpulses (AP), which arrive with some delay from the parent pulse, and have smaller amplitude as the delay goes to zero, and direct cross talk (DiCT), which manifests as an integer multiplication of the amplitude of pulses because the children pulses are overlapped with the parent.

The thesis is divided in eight chapters. The first two chapters provide a short pedagogical introduction to dark matter and to the DarkSide experiment. Chapter 3 is a reference for the sources of the datasets used in the analyses. Chapters 4, 5 and 6 deal with the performance of single pulse detection in the PDM output: signal to noise ratio, temporal resolution, and fake rate. Chapter 7 analyzes correlated noise. Finally, chapter 8 summarizes the key results from each chapter. The computer code implemented for this thesis is released as open-source, see Appendix C.

Chapter 1

Dark matter

When introducing the subject of fluid dynamics, one starts by considering an ideal fluid with indefinite extension, that sits alone in the universe, has no particular properties, does not undergo chemical reactions, has no friction, and is subject only to forces with an easy to write expression, such that the kinetic equation can be exemplified without too much fuss. This spherical cow fluid does not know the beauty of the foam riding on sea waves, of violent explosions, of sinking treasures, of stars shining with the power of a million suns, nor anything that makes fluids interesting in real life. I consider it a great victory of Physics that most of the matter in the universe is indeed such a fluid.

While preparing the work for this thesis, I had the occasion to talk to a friend I had not seen in a while. He is endowed with a rather curious mind, so he gladly listened to me talking about dark matter. After I explained that the visible part of the Milky Way is overlapped with an invisible halo of an intangible substance that is probably passing through us at every instant, he immediately asked if this "dark galaxy" has solar systems, planets, life, a complete world parallel to ours.

«Of course *you* would ask such a question!—I replied—And, of course, the answer is *no*.» I continued: «Dark matter has no friction, it can not agglomerate to form objects. We know this because from the gravitational attraction exerted by the dark matter halo we can infer its shape, and it is spherical, while the galaxy is a disc with a central bulge. Now if you think about it, all the structure of the galaxy is recursively a ball surrounded by a disc: the solar systems are like that, and each planet in turn can have satellites and rings, then you can even have stuff orbiting around satellites, and then your weird hat.»

«Yes. . . »

«If the planets are not sufficiently discous for you, consider the asteroid belt, and that the planets originated from a more homogeneous disk of matter swirling around the sun. This is not a coincidence, because nothing is ever a coincidence. This fractal pattern originates from the effect of friction, and the conservation of angular momentum.»

«You have to breathe, Giacomo.»

«*gasp*—You know what happens when a ballerina closes her arms, right? She spins faster. Now imagine a primordial, immense, uniform sphere of matter standing still. It will start collapsing under the effect of the gravitational force. This is a *really* vast sphere, so as it contracts the radius can reduce by a large factor before it arrives at the scale of the galaxy. Now, if there is even just a slight perturbation to the initial stillness of the sphere, at some point in the contraction this turns into a significant rotational speed, which grows continuously. Do you know what happens to something that

rotates too fast?»

«Sigh. It feels sick?»

«It *breaks down*. The external shell of the sphere gets thrown around in shards, while the core, freed from the faster rotating parts, can continue shrinking. This process can repeat until the inner core is small enough that the pressure of the compressed matter starts balancing the gravitational force. So now you have a core surrounded by a large spherical cloud of matter streaming around it. This is still spherical, because things that break are chaotic, but the only possible final outcome is that it forms a disk. This is because the way of preserving the angular momentum that minimizes the kinetic energy is by just performing the rotation implied by the momentum, without additional directions of movement. Since friction dissipates energy, with time the sphere will flatten into a disc.»

«. . . »

«A more direct way to picture this is the following: consider each lump of matter orbiting around chaotically in the cloud. They go in different directions, so the pieces will regularly cross and knock each other. This is a sort of battle between different orbital planes. What plane will prevail in the end? The only one that, even if alone, would be able to preserve the angular momentum. In principle friction can collapse the disc too, but it is much slower because there are no hard hits, the only friction is a smooth one between the concentric layers of the disc that rotate at different speeds. And, when the disk is steady, each part of this thing that now looks more like a galaxy will start collapsing on its own, since it is not perturbed any more. Once the pieces have collapsed they do not touch each other, and so the global shape is frozen. The local collapses produce stars, and so on until satellites. The process stops because small things are too solid—friction wins over gravity. Matter piles up and can not shrink any more.

Now, what happens if there is no friction? Let me answer for you. The answer is not that we get sub-planets and sub-satellites and tiny sub-hats. Go back to the beginning. The primordial sphere is collapsing. If it was perfectly still, and intangible, by symmetry everything would pass through the center, and continue straight to the other side, and the sphere would continue to oscillate radially. Given an initial perturbation, things will start to rotate, but not in the sense of an object that rotates, everything orbits randomly in all directions. The sphere shrinks until the gravitational energy is balanced with the kinetic energy, then lingers in the state of spherical cloud. Thus, by Landau's arrow, dark matter must be frictionless.»

Where the "Landau's arrow" is Bayes' theorem. If the reader knows a bit about astronomy, she may have noticed that my explanation was, to put it charitably, qualitative. Elliptical galaxies do exist, are found in sizes smaller to larger than the Milky Way, and possibly with low eccentricity. Furthermore, simulations of the formation of the dark matter halo show that it actually ends up more spherical if it *does* interact, see Figure 1.1, and that it has a rich substructure. Dark matter is expected to form aggregations of all sizes down to the mass of the Earth [VZL12, p. 1].

To rule out the possibility of dark matter solar systems and planets etc., I have to invoke actual measurements. In 2006 an analysis of a pair of colliding clusters of galaxies [Clo+06], where the smaller one is named "Bullet cluster", showed with high confidence that the barycenters of the two clusters have passed through, following the galaxies, while the intergalactic gas clouds, visible in X-rays, bumped into each other. The clouds are about ten times more massive than the galaxies, so the observed motion of the barycenters is a strong evidence of the presence of heavy, invisible and collisionless

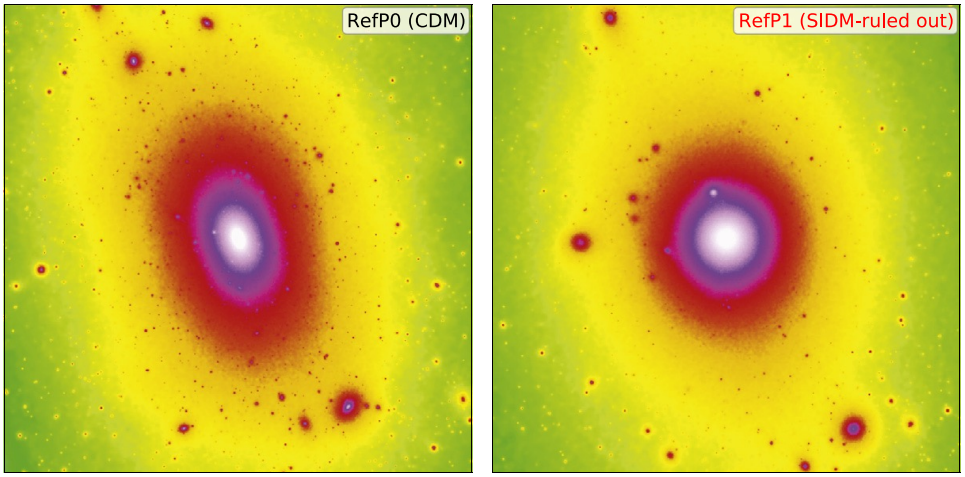


Figure 1.1: Projected density on a 270 kpc cube of the simulation of a Milky Way-like non-relativistic dark matter halo ("cold" dark matter, CDM). Left panel: non-interacting dark matter. Right panel: self-interacting dark matter (SIDM) with cross section/particle mass ratio $10\,\mathrm{cm}^2/\mathrm{g}$. From [TY18, p. 21], originally from [VZL12, p. 6].

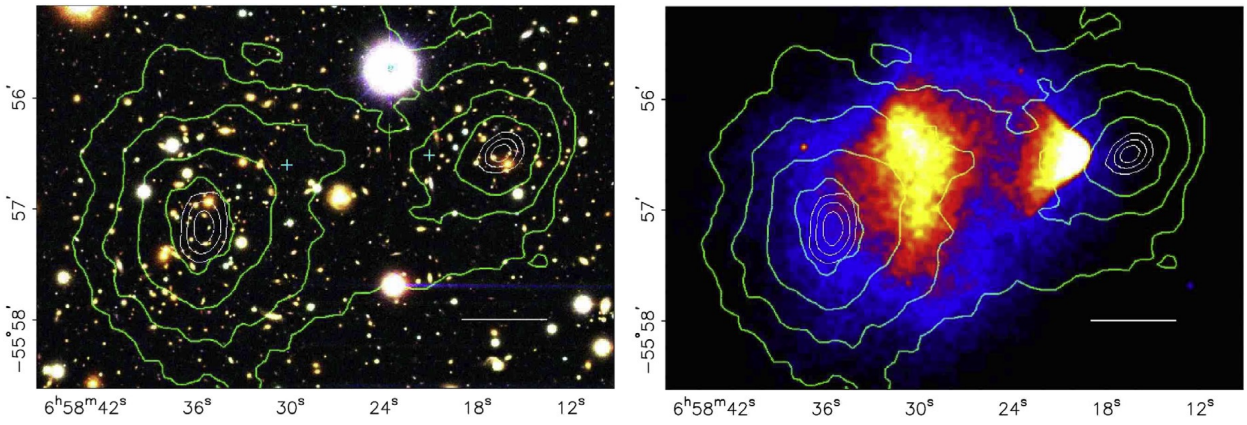


Figure 1.2: Left panel: photo of the Bullet cluster and its companion in the optical band. Right panel: the same patch of sky in X-rays, where the intergalactic gas is visible. The contours represent the mass density obtained from gravitational lensing. From [TY18, p. 29], originally from [Clo+06, p. 3].

haloes associated to the clusters. See Figure 1.2.

Looking at thirty such collisions, [Har+15] obtains a 95 % confidence level upper bound on the cross section/particle mass ratio of $0.47\,\mathrm{cm}^2/\mathrm{g} = 0.84\,\mathrm{barn/GeV}$. The limit is on the ratio, instead of just the cross section, because the total mass is fixed by the lensing measurement, and the mass of individual dark matter particles is unknown, and could vary by many orders of magnitude [Zyl+20, p. 474]. This is about the same ratio of a nucleon: cross section $\sim 1\,\mathrm{barn}$, mass $1\,\mathrm{GeV}$. An atom has the same mass, but cross section 10^{10} times higher, so definitely dark matter is not forming anything similar to us unless we seek very far-fetched speculations.

The density of dark matter in the neighborhood of the solar system can be inferred from its effect on the motion of nearby stars, which is measured precisely. The most recent estimate, done with the Gaia satellite data, gives $\sim 1\,\mathrm{GeV/cm}^3$ [BLF19]. For comparison, the solar wind density at the Earth orbit is some particles per cm^3. So to have a concrete picture in our mind we can imagine dark matter as a cloud of stable neutrons which does not interact with ordinary matter, and with density similar to the solar wind. The same article informs us that whether self-interacting dark matter would

form a disk is still under debate, so my improvised speculations were not too far off the mark. Assuming that it would, they put a limit on the fraction of self-interacting dark matter in the halo to less than 1 %.

Are there any other cool things that dark matter could do, beyond parallel worlds? I once wondered whether it would form black holes. Black holes are thought to originate from the cores of stars and galaxies, so definitely they involve a lot of friction to accumulate stuff in one single place until it collapses. But a totally non-interacting matter could form a black hole through another venue: if by coincidence a sufficient amount of fluid passes through one point, the density can get above the critical level necessary to form a black hole. There is no classical lower limit to the size of a black hole, since the Schwarzschild radius is proportional to the mass.

It turns out that, given the dark matter density, this is unlikely enough to be considered impossible. I know this because a guy at the Kavli Institute told me so. I could not find the paper searching on the net because, when I query "dark matter black hole", I am swamped by a swarm of results on the primordial black holes (PBH). They are hypothesized small black holes formed during the big bang that could make up all or part of the dark matter. Their mass is constrained to be less than about five solar masses, or the effect of their individual attraction would be visible. They are a viable hypothesis for dark matter, although under debate [Zyl+20, p. 485].

1.1 Experimental evidence

The Bullet cluster is currently one of the most important evidences of dark matter, since it excludes that the observed gravitational effects could be due to a simple modification of the gravitational laws instead of the presence of additional unseen matter. Indeed, also the very first evidence was obtained from galaxy clusters, by [Zwi33]. He observed that in the Coma cluster the velocity of galaxies is much higher that what would be predicted by applying the virial theorem with the masses obtained from the expected mass to light ratio.

As in the case of the Bullet cluster, mass excesses are also observed with gravitational lensing. Another clear effect is the orbital speed of stars in the galaxies. Applying Gauss' theorem, since most of the light-emitting mass of a galaxy is concentrated in the bulge, the orbital speed should fall off like $1/\sqrt{r}$. Instead, $v(r)$ rises in the bulge and then stabilizes to an almost constant value, indicating the presence of a halo.

The other line of evidence is cosmological. The density of matter is clearly very inhomogeneous, with aggregation at all scales. The variations of density at the last scattering epoch correspond to the temperature anisotropies of the cosmic microwave background (CMB), shown in Figure 1.3, which are $\sim 10^{-5}$. It is predicted that the variations would increase linearly with the scale factor of the expansion of the universe, which from the origin of the CMB is ≈ 1100, giving a still very small factor of 10^{-2} nowadays. The additional aggregation can be explained by a pre-existing matter which was already decoupled from radiation. Moreover, the presence of this matter directly influences the angular spectrum of the CMB, which is given by stochastic collapse-compression-expansion cycles in the primordial plasma, and from which it is determined that dark matter amounts to 85 % of the total matter density of the universe. This value agrees with the gravitational observations.

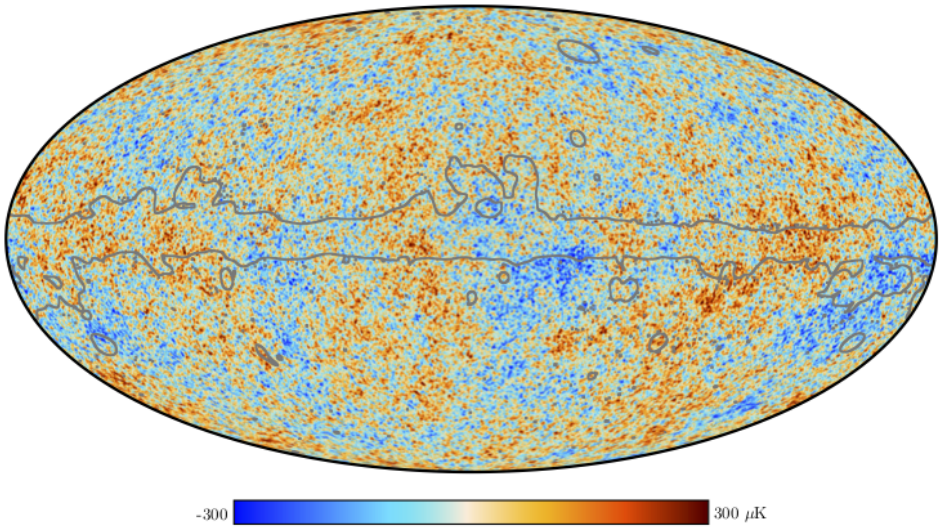


Figure 1.3: Map of the variation of temperature of the cosmic microwave background (CMB) relative to the mean 2.73 K and to the Doppler shift caused by our motion relative to the CMB. From the 2018 Planck release [Pla+20, fig. 6].

1.2 Direct detection

All models of dark matter, apart from modified gravitational theories and primordial black holes, postulate that dark matter is constituted by unclassified particles. Neutrinos do behave like dark matter and account in some part for it, but they are estimated to contribute only 1 %. The limit comes from the fact that, being relativistic ("hot" dark matter), they would not aggregate and solve the problem of the density variations. I have never heard of models where dark matter is a continuous fluid.

Most of the experimental efforts are directed towards the detection of WIMPs, "weakly interacting massive particles". WIMP stand for a generic additional particle in the Standard Model, which to predict the correct relic density of dark matter after the cool down, would need to have a self-annihilation cross section with an order of magnitude similar to the one expected for a particle which interacts through the weak force, and a mass with an order of magnitude around 100 GeV.

Anyway, the point is hoping that dark matter scatters on electrons and nucleons in the same way of a neutrino. Then, to leading order, the angular distribution can be calculated independently of the details of the interaction. Assuming a thermal distribution for the velocity of dark matter particles in the halo, using the measured local mass density, and assuming a specific particle mass, then the rate of scatterings in a target can be predicted, proportional to the cross section. The lack of signals in a given time frame then puts an upper limit on the cross section given the assumed mass.

Figure 1.4 shows the limits obtained in this way by recent experiments. Currently the most stringent constraints are set by experiments hosted at Laboratori Nazionali del Gran Sasso (LNGS), DarkSide-50 and Xenon1T. Those limits are for spin-independent (SI) scattering. There are also measurements that consider an interaction term with the spin (spin-dependent, SD), but they are less sensible because the whole nucleus of the target atoms counts for its overall spin, while with SI each nucleon contributes coherently.

From the curves it is evident that there is only a finite range of masses probed by these searches.

The lower limit comes from the mass of the target particle. The scattering would be detected by the recoil of the nucleus, which releases energy in

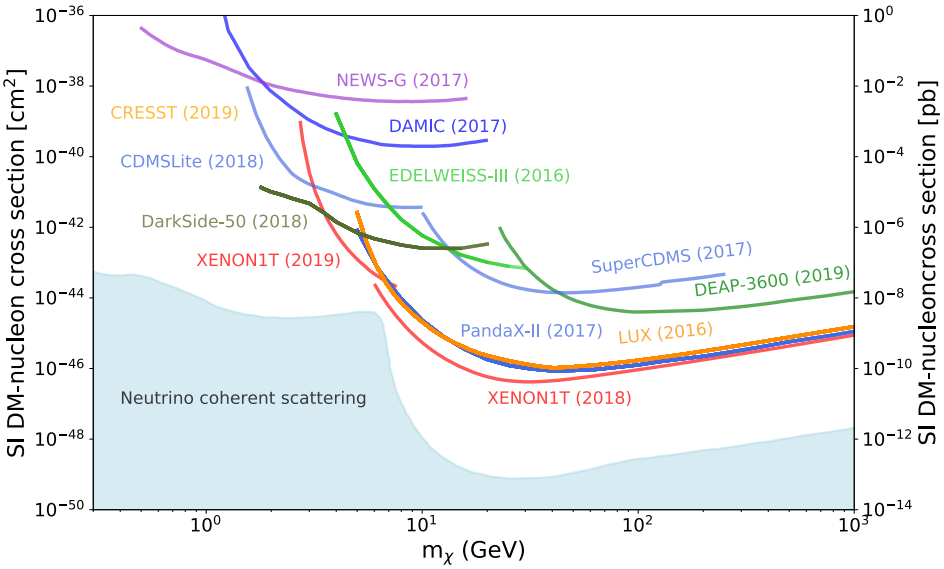


Figure 1.4: 90 % C.L. upper limits on the WIMP-nucleon SI scattering cross section, conditional on the WIMP mass. From [Zyl+20, fig. 27.1 p. 481]. These limits do not keep into account the uncertainty on the local mass and velocity distributions of dark matter [Bax+21, sec. 3.1].

the detector. If the recoil energy is too small it can not be detected, with different thresholds depending on the detection technique, and of course the recoil momentum goes to zero as the incoming particle mass decreases. Considering electrons instead of nuclei, lower masses can be probed, but the measurements are less sensitive because the radiation background of neutrons, impacting nuclei, is kept better under control than gamma and beta rays which impact electrons.

The upper limit is softer and comes from the fact that in this problem the mass density of dark matter is fixed by gravitational measurements. Assuming an higher particle mass means a proportionally lower number density, which is what determines the rate together with the cross section, so the limits go up linearly with the mass. In other words, the limitation is the size of the target times the exposure time.

The two most popular choices for the target material are liquid argon and xenon. Being fluid, they can be purified from radioactivity to an high degree, more than a solid scintillator. Being noble elements, ionized electrons can drift without recombining, allowing to use a time projection chamber (TPC) to reconstruct the position, which is needed to exclude signals near the surface, where there is additional background from the surrounding components of the detector. Of the stable noble elements, they have the best scintillation yield, about 40 photons per keV.

There are other dark matter models and other kinds of experimental techniques. At the large hadron collider (LHC), dark matter is searched through missing energy or looking for new resonances. An interesting method is discriminating the dark matter signal from the background using the motion of the Earth relative to the halo, which is expected to induce a seasonal variation in the rate and directionality of the recoils. A curious proposal of this kind considered using RNA strands to achieve high spatial and directional resolution [Dru+15]. For a complete overview on the matter, see the particle data group [Zyl+20, sec. 27]. [Sav18, ch. 1] gives a more detailed introduction to some topics.

Chapter 2

The DarkSide experiment

The DarkSide20k experiment is a planned dual phase liquid argon (LAr) time projection chamber (TPC) designed to detect nuclear recoils (NR) with energy in the range 30 keV to 200 keV, with a 20 metric ton fiducial target mass. With 5 years of exposure DarkSide20k should reach a sensitivity of $1.2 \times 10^{-48}\,\mathrm{cm}^2$ on the spin-independent (SI) WIMP-nucleon cross section at WIMP mass 1 TeV, to be compared with the current best limit $1 \times 10^{-46}\,\mathrm{cm}^2$ by XENON1T. DarkSide20k is optimized for the search at high WIMP masses.

The project consolidates the efforts of four LAr dark matter search experiments: ArDM, DarkSide-50, DEAP-3600 and MiniCLEAN. It will be hosted by Laboratori Nazionali del Gran Sasso (LNGS). We give here a brief description of the experiment. For details, see the published "Yellow Book" [Aal+18]. An unpublished but publicly available preliminary design report [Aal+19] provides more up to date information, in particular the veto system was completely redesigned.

2.1 Interaction of WIMPs in argon and background

The experiment is designed to detect elastic WIMP-nucleus scatterings. Assuming that the dark matter halo is thermalized, the WIMP speed is of the same order of the orbital speed of the solar system in the galaxy, 230 km/s $\sim 0.001c$, and bounded by the escape velocity 550 km/s. So the WIMPs are not relativistic and we can model the collision with classical dynamics. Thus the nucleus recoil speed is suppressed for WIMP masses below the mass of the nucleus, and is asymptotically bounded by twice the WIMP speed for large WIMP masses, giving a maximum recoil energy of $1/2(1\,\mathrm{GeV})(2 \cdot 0.001)^2 = 2\,\mathrm{keV/nucleon}$ so $\sim 80\,\mathrm{keV}$ for the A=40 argon nucleus. A precise calculation gives 270 keV. The result with the electron as target is 3.6 eV, less than the argon ionization energy 13 eV and than the average photon energy of the argon scintillation light, 10 eV [Apr+06, p. 16, 79], so elastic collisions on electrons are not detectable.

When radiation or our hypothetical WIMP hits an argon atom, the release of energy produces ion-electron pairs, and excited states which decay emitting "vacuum ultraviolet" (VUV) photons with a distribution peaking at 130 nm. See [Apr+06, ch. 2, sec. 3.4] for an explanation of the chain of reactions. After the primary ionization and scintillation, recombination converts part of the ion pairs into photons. The overall process is different depending on whether the incoming particle scatters on an electron ("electron recoil", ER) or a nucleus ("nuclear recoil", NR). In particular:

1. the relative amount of ionization and scintillation produced is different for ER and NR;

2. the photons are produced by two possible excited states, one with a fast decay constant, 7 ns, and a slow one with 1.6 µs;

3. the relative amount of fast and slow photons is different for ER (roughly 1 to 3) and NR (roughly 3 to 1).

Property (3) allows to distinguish ER from NR using the fast/slow ratio. This is the most effective NR/ER discrimination in argon and is called pulse shape discrimination (PSD). For comparison, in xenon detectors the PSD is not effective, and the most important discriminant is the light/ionization ratio.

To reach the required sensitivity, it is necessary for the experiment to have zero background, in the sense that all background radiations must be identified and removed from the data instead of accounted for with a model. To get an idea why, consider that if a detector does not observe any signal for a time t on n targets, the upper bound on the interaction probability goes down like $1/(nt)$. Instead, if there is a background rate r, the uncertainty is dominated by the Poisson error and goes like $1/\sqrt{rnt}$.

The principal way of reducing the background is limiting the radiation in the first place: building the detector with as radio-pure as possible materials, employing shielding, and operating the detector underground to reduce the cosmic ray flux. These measures must bring the background rate down to a level which is manageable by the detector and which leaves enough "clean time" for WIMP detection. After that, the residual background events must be actively excluded with some criteria. An important background is given by gamma and beta rays. These interact with electrons, so they can be excluded with the PSD. A more difficult background are neutrons. Individually, a neutron-induced NR is indistinguishable from the signal. To exclude neutrons DarkSide20k will use a time projection chamber (TPC) and a veto system, described in the next section. In the planned exposure of 100 t yr, less than 0.1 background events surviving the cuts are expected.

An irreducible background is given by the coherent elastic neutrino-nucleus scatterings, which are indistinguishable from WIMPs without directional information. When the "neutrino floor" will be reached (see Figure 1.4), the experiments will at least do some new measurements on neutrinos, if dark matter is not found. DarkSide20k expects approximately one neutrino-induced NR in the planned exposure [Aal+18, p. 119].

2.2 Dual phase TPC

To implement the PSD, it would be sufficient to surround the mass of argon with photodetectors which count the photons produced by the recoils. This is the approach followed by the DEAP experiment.

However, this would not allow to reconstruct precisely and reliably the position of the signal. The position information can improve the rejection of two kinds of backgrounds: 1) backgrounds producing multiple hits, typically attributed to neutrons, considering that a WIMP would scatter just once, and 2) backgrounds in the outer part of target volume, produced by, e.g., the radioactive contamination of the materials of the detector.

In DarkSide20k, to measure accurately the position, a time projection chamber (TPC) design is employed. The argon is put into a cylindrical vessel, about 2.5 m tall and 3.5 m wide, where an uniform 200 V/cm electric field pointing downward is maintained. The free electrons produced by ionization drift upward, taking at most 4 ms from the bottom to the top of the vessel,

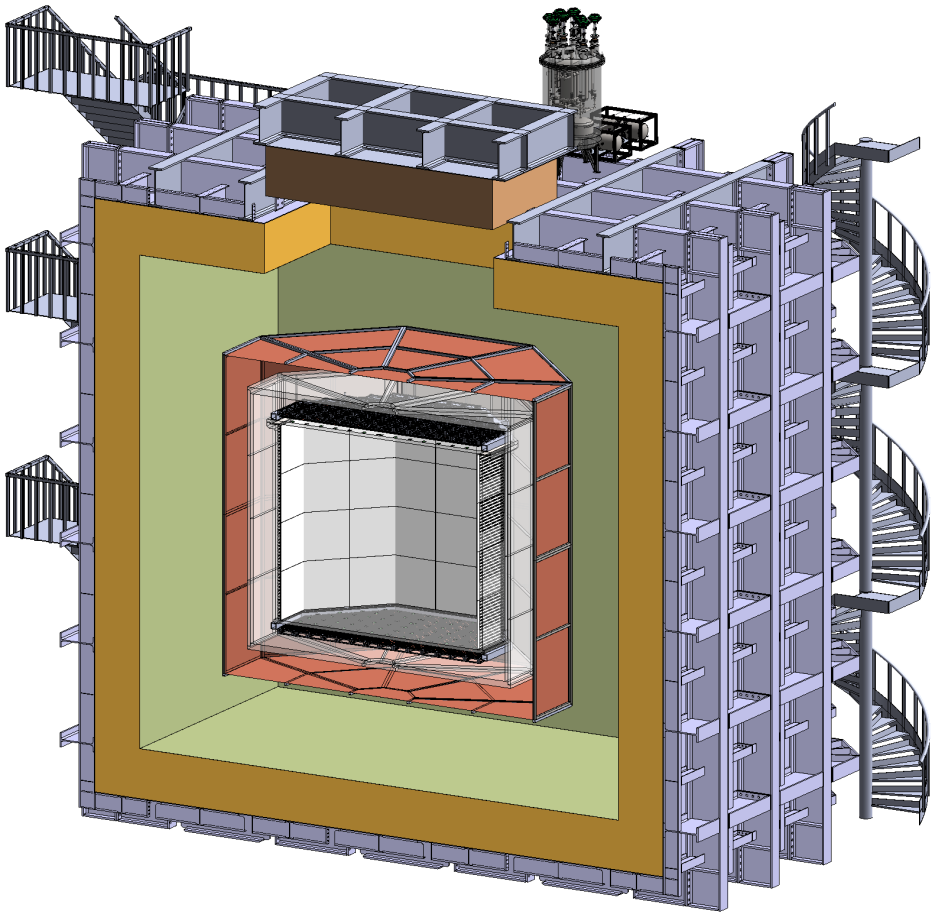


Figure 2.1: Schematic cross section of DarkSide20k. From [Aal+19, p. 2]. The innermost cylinder is the TPC, wrapped in the two layers of the veto detector. The outer container is the cryostat, copied from the ProtoDune experiment.

while the photons reach immediately the photodetectors, placed on the top and bottom faces.

Figure 2.1 shows a scheme of DarkSide20k. The innermost cylinder is the TPC. The two successive shells delimit two layers of LAr which are used as a veto detector. Cosmic rays can produce neutrons when they hit the detector, but first they have to pass through the veto detector, which tells the system to ignore the signal.

The outermost tank is the cryostat, which is completely filled with LAr. In this design the argon circuit of the TPC is separate, because it is filled with argon extracted underground (UAr) and purified to have about 1/1400 the amount of radioactive ^{39}Ar isotope relative to argon distilled from the atmosphere (AAr), which is used instead for the rest of the system.

At the top of the TPC vessel, just below the photodetectors, a "pocket" facing downward, like a diving bell, is kept filled with gaseous argon. An additional grid provides an higher electric field in the gaseous phase, called "extraction field". When the electrons emerge from the liquid phase, they are accelerated toward the anode and produce scintillation in the argon vapor. This process is called electroluminescence, or secondary scintillation. The extraction field is not intense enough to induce further ionization when the electrons pass, so the information on the number of ionized electrons is preserved. Most of this light hits the top photodetection plane concentrated in a narrow spot.

The prompt photons produced by the recoil are called S1, while the light emitted by extracted electrons S2. The names stand for "first signal"

and "second signal", since S2 arrives after S1 due to the drift time. This delay allows to deduce the depth of the recoil, i.e., the z coordinate. Since most of the S2 light is concentrated, it is easy to measure its position on the horizontal photodetector plane. The electrons drift vertically, with little transverse diffusion, so the reconstructed xy S2 position matches the xy coordinates of the recoil. Thus the TPC permits a 3D reconstruction of the signals.

2.3 Photodetectors

DarkSide-50 uses photomultiplier tubes (PMTs) for photodetection. DarkSide20k instead will be equipped with silicon photomultipliers (SiPMs). The SiPMs are large matrices of photodiodes connected in parallel and operated in Geiger mode. The advantages of SiPMs compared to PMTs are:

- better single photon resolution;
- higher filling factor, i.e., they tile more densely the surface;
- low bias voltage;
- better radio-purity;
- higher photon detection efficiency (PDE).

The basic unit of the photodetection system is the photodetector module (PDM), consisting of a Tile, i.e., a 6×4 matrix of SiPMs totaling $25\,\mathrm{cm}^2$, connected to a front end board (FEB) hosting the biasing circuit and the preamplifier. Matrices of 5×5 PDMs are bound together to form a motherboard (MB). Each motherboard is paired with a "steering module" controlling the PDMs and transmission electronics, forming a photodetector unit (PDU). Figure 2.2 shows a Tile, a FEB, a PDM and a MB.

The Tiles are wired as follows. There are four 3×2 SiPMs quadrants, visible in Figure 2.2. In each quadrant, SiPMs are connected in series in pairs, and then the three branches in parallel. The quadrants are then connected to the FEB. For each quadrant there is a trans-impedance amplifier (TIA), which converts the current signal to a voltage signal. The outputs of the TIAs are summed analogically. The choice of grouping multiple SiPMs in a PDM is made to limit the number of channels to read out, for reasons of cost, complexity, and limitation of the radiation contamination from electronic components. The division in quadrants with separate preamplifiers, instead of a single parallel connection, is made to reduce the input capacitance seen by the TIAs. Electrical noise limits the number of SiPMs that can be summed analogically into a single channel, and thus the size of the PDM.

The signals will be carried out of the cryostat to digitizer boards equipped with 14 bit 125 MSa/s ADCs connected to an FPGA and a controller CPU. The digitizer board is expected to implement basic single photon pulse identification and send selected data through an ethernet link to a front end processor (FEP), a normal CPU station, that refines the analysis. Finally, an event builder farm will take the information from all the FEPs, organize the sets of photon hits into events, and decide what to record to permanent storage.

The readout of SiPMs presents some additional issues relative to PMTs. Each SiPM is a matrix of photodiodes connected in parallel. Since in the PDMs the SiPMs are either connected together or summed, we can treat approximatively the whole PDM as a single large SiPM. The random electrical

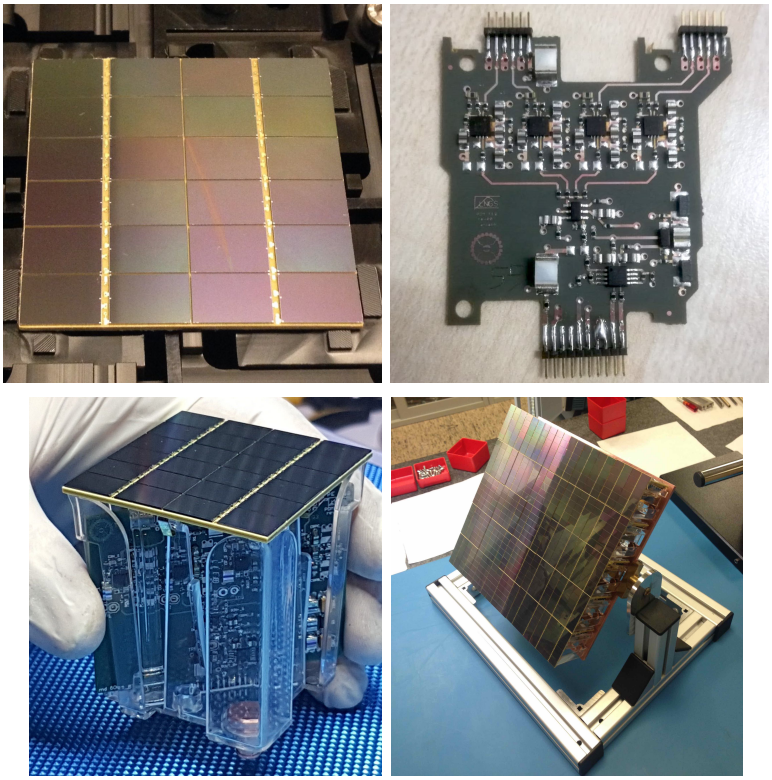


Figure 2.2: Top row, left panel: a Tile, and right panel: a FEB, from [Mar19, p. 42]. Bottom row, left panel: a photodetector module (PDM), i.e., a Tile attached to a FEB, and right panel: a motherboard with 25 PDMs, from [Aal+19, p. 4].

noise is summed in quadrature, so it is proportional to the square root of the area. This means that the noise is quite high for the DarkSide20k PDMs. In practice, as we will see in chapter 4, the noise RMS is comparable to the amplitude of the single photon signals. Another issue is correlated noise. Each SiPM output current pulse can produce other pulses, recursively, either delayed from the parent pulse or stacked. The production of correlated noise is stochastic, i.e., the number of additional pulses for each primary pulse fluctuates. This implies that the extension of the dynamic range can not be compensated by tuning the gain, and that the loss of photon count resolution is substantial.

The SiPM photodiodes are reverse biased below breakdown voltage. The production of free carriers by an incident photon starts an avalanche, regulated with passive quenching. The amplitude of the resulting current pulse is approximately proportional to the difference between the bias and the breakdown voltage, called *overvoltage*. The electrical noise, instead, depends weakly on the overvoltage, thus its effect can be mitigated by increasing the overvoltage. However, this also increases correlated noise, because an higher electric field in the junctions facilitates the production of charge carriers. This means that the overvoltage is a critical parameter that must be tuned to operate the SiPM. We defer a more complete description of the SiPM to chapter 7.

Chapter 3

Data

We drew on two types of datasets: 1) data collected with the Proto0 DarkSide prototype at CERN, and 2) data collected with a cryogenic test stand at Laboratori Nazionali del Gran Sasso (LNGS).

Both datasets contain digital traces of photodetector modules (PDMs) output, divided in continuous chunks of fixed length that we call "events". Optionally, a pulsed laser is shone at the detector and the temporal window of the event waveform is synchronized with the laser.

3.1 Proto0

Proto0 is a small prototype of time projection chamber (TPC) in liquid argon (LAr). It was built and operated in CERN building 182 in 2019, and now will be moved and operated again in Napoli in 2021. Information is provided in the CERN wiki page [Dar21a]. Note that the descriptions in [Luz20, sec. 4.3.2] and in the Yellow Book [Aal+18, p. 65] contain outdated information relative to the status of the setup when the data we are considering was recorded.

The TPC size is $30\,\text{cm} \times 30\,\text{cm} \times 20\,\text{cm}$. On top of the TPC sits a photodetector unit (PDU), i.e., data transmission electronics plus a motherboard. This specific one is dubbed "Motherboard 2" (MB2), and consists of a matrix of 5×5 PDMs. The silicon photomultipliers (SiPMs) Tiles mounted in the PDMs were fabricated by Fondazione Bruno Kessler (FBK) in 2019. The tiling scheme of the motherboard is shown in Figure 3.2, while in Figure 3.1 there are some photos of the apparatus.

The PDM outputs are each sent to a channel of a CAEN V1725 analog to digital converter (ADC) board. The ADC has 16 bit and sampling frequency 250 MSa/s, downsampled in software to 125 MSa/s to match the specifics which will be adopted in DarkSide20k. The downsampling is simple decimation without antialiasing, i.e., every two samples, one is discarded. The response of the SiPMs will be described briefly in chapter 4 and more in detail in chapter 7.

Of the Proto0 data, we used only a run collected with the PDMs biased below breakdown voltage and thus almost insensitive to light. So the waveforms contain only electrical noise. We analyzed 1000 events, each 0.5 ms long. In Table 3.1 we list for reference the conditions for this run. A persistence plot of the data for Tile 53 is shown in Figure 6.5.

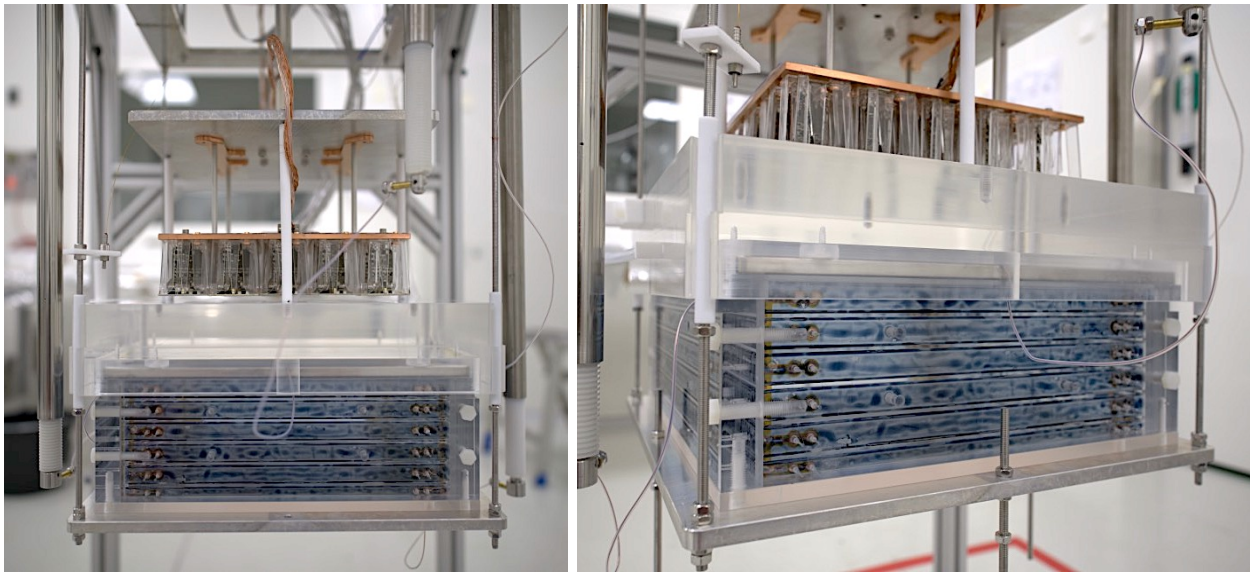


Figure 3.1: The Proto0 TPC outside of the cryostat. Photos by Tom Thorpe (GSSI) [Dar21b]. When in the cryostat, the whole object is completely submerged in LAr. Inside the bluish box there is a uniform electric field pointing top to bottom. The stripes on the sides are conductive and are the steps of a voltage divider, to create boundary conditions that keep the field uniform even near the borders. The metal frame visible just over the blue box, behind the semitransparent plastic, is the support of the grid used to create an high field region above it up to the anode. That region is contained in a "gas pocket", i.e., something like a cup kept upside-down underwater. The pocket is kept filled with gaseous argon. The copper frame on the top supports the PDMs, with their characteristic plastic case holding the front end board (FEB) attached orthogonally to the SiPMs Tile.

PDM 1		PDM 2		PDM 3		PDM 4		PDM 5	
	31		32		39		64		55
126	57	132	44	136	42	142	52	149	53
0	0	0	2	0	4	0	6	0	8
PDM 6		PDM 7		PDM 8		PDM 9		PDM 10	
	30		59		57		37		29
127	33	133	41	138	37	144	31	150	43
0	10	0	12	1	0	1	2	1	4
PDM 11		PDM 12		PDM 13		PDM 14		PDM 15	
	38		36		58		62		60
129	46	134	48	139	60	145	59	151	50
1	6	1	8	1	10	2	0	2	2
PDM 16		PDM 17		PDM 18		PDM 19		PDM 20	
	41		61		66		63		52
130	47	135	32	140	40	146	56	152	35
2	4	2	6	2	8	2	10	3	0
PDM 21		PDM 22		PDM 23		PDM 24		PDM 25	
	34		53		54		65		42
131	38	137	58	141	34	148	51	153	45
3	2	3	4	3	6	3	8	3	10

PDM slot	
	tile (run2)
PDM	feb
V1725[0..3]	Ch[0..15] (top)

Figure 3.2: Schematic of the motherboard installed in Proto0, showing the SiPM Tile identification number of each PDM and the ADC channels. The color of the Tile number sets apart the two set of SiPMs used: green "run2" have a higher breakdown voltage than violet "run4"; being all operated at the same bias, the run2 Tiles have overvoltage 5.5 V while run4 6.5 V. The PDMs marked in red are the inputs of the data trigger, which is not used in the data we consider.

Field	Value	Field	Value
run	886	gas pocket	ON
date(dd-mm)	5-11	laser intensity	
run type	baseline run	trigger	external (50 Hz)
Quality	good	trigtime (µs)	100
Problem		post-trigger (µs)	400
Nevents (k)		Time gate (µs)	500
Edrift (V/cm)	200	PDM Coincidence	
Eextraction (kV/cm)	2.8	Threshold extent (ns)	80
SiPM bias voltage (V)	50	TPC pressure (mbarg)	>450

Table 3.1: Metadata for the Proto0 baseline run 886.

3.2 LNGS

We also used laser-run data collected at LNGS with a test stand built to characterize individual SiPMs or Tiles, briefly described in the Yellow Book [Aal+18, p. 34], and presented in greater detail in [Ace+17] and [Sav18, ch. 3]. It achieves a temperature stability of 0.1 K and accuracy 3 K in the operative range 40 K to 300 K. The digitizer is a CAEN V1751. The laser light is diffused prior to reaching the SiPM.

These datasets were collected between 2018 and 2021 by Lucia Consiglio. They contain all the FBK Tiles used in Proto0 apart from Tile 55 (data from 2018–2019), plus other newer Tiles produced by LFoundry, Tiles 15, 21, 22, and 23 (data from 2020–2021). The data was retrieved from the bookkeeping directories `SiPM/Tiles/FBK/NUV/MB2-LF-3x/` and `SiPM/Tiles/LFOUNDRY/pre-production-test/`. The directory name `MB2-LF-3x` stands for "Motherboard 2, low field, triple doping". The data for all Tiles are collected using the same reference FEB.

The events are laser-triggered and recorded in `wav` files. Each file contains the Tile output, and in most runs there is also the recorded laser trigger in an additional channel, containing a square pulse marking the trigger location, which occurs at the same place relative to the event start varying in a 23 ns range (see Figure 7.5). Of the two-channels files we used, the channel layout is 0:PDM, 1:trigger for FBK, swapped for LFoundry.

The format of the `wav` files is the following. The files are arrays of 16 bit signed integers, divided into events. Each event starts with 20 values of metadata, followed by 15 001 values for each channel. The ADC resolution is 10 bit, so the values are always in the range 0 to 1023. The sampling frequency is 1 GSa/s.

Persistence plots of the data for tiles 15, 21, 53, 57 and 59 are shown in Figure 6.6, A.4, 6.5, and 4.1.

Chapter 4

Signal to noise ratio of filtered photodetection signals

When a silicon photomultiplier (SiPM) detects a photon, the electrical output is a sudden current spike, with a rise time on the order of nanoseconds, which decays slowly in some microseconds.

This signal is immersed in electrical noise, which is proportional to the square root of the area of the SiPM, since all photodiode cells are connected in parallel. Given the relatively large area of the SiPMs Tiles, $25\,\mathrm{cm}^2$, the amplitude of the noise can be comparable to the amplitude of the signals.

This implies that to read out the detector it is important to filter the waveform to suppress the noise. In this chapter we will study the performance of some common filters on a PDM output.

4.1 Data

We use the LNGS data for Tile 57, see section 3.2. The data file is `MB2-LF-3x/NUV-LF_3x_57/nuvhd_lf_3x_tile57_77K_64V_6VoV_1.wav`. The Tile is operated at 6.0 V overvoltage, to be compared to 6.5 V in Proto0, see Figure 3.2. The amplitude of the signals is proportional to the overvoltage, while the noise should not depend on it, so this difference has to be taken into account when comparing the performances in the Proto0 and LNGS setups.

In Figure 4.1 we show the persistence plot of the waveforms, both on the whole acquisition window and zoomed on the pulses produced by the laser. The persistence plot is a two-dimensional histogram of the events, where the variables are the sample number in the event and the ADC value. It is like stacking on each other all the waveforms. We have still not explained in full detail how a SiPM works, for now we will do with what we can see with our own eyes in the data. We defer an in-depth explanation to section 7.1.

From the whole event visualization it is evident that there are other pulses occurring at random times beside those produced by the laser. We will need to measure the height of pulses, and to do that we need a reference value, i.e., a baseline. The straightforward way to compute the baseline is taking the average of the pre-trigger region before the laser pulse, but this average would be biased by the presence of a pulse. So we filter away all events where there is a pre-trigger sample below 700, choosing this threshold by looking at Figure 4.1. There are 72 such events out of 10 005.

Looking at the zoom on the laser pulses, we see that the amplitude of the pulses is discrete. The laser does not emit only one photon, we expect a Poisson distributed number of photons to hit the Tile. So we interpret the discrete amplitudes as corresponding to the number of detected photons. We use "PE", standing for "photoelectron", as a unit of measure to indicate

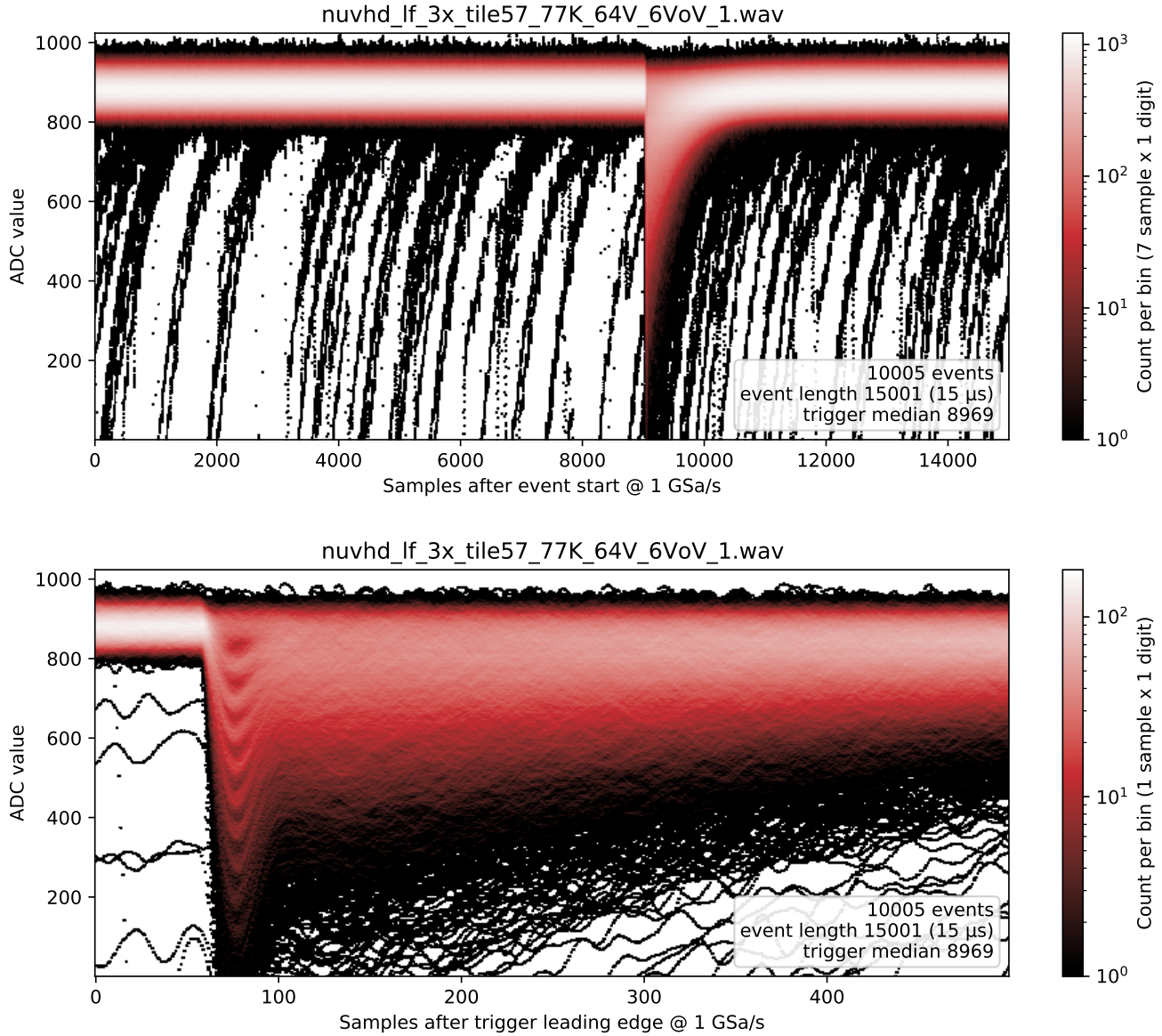


Figure 4.1: Time-value histogram of Tile 57 in LNGS data. Top panel: whole events. Bottom panel: zoomed on the laser pulses and synchronized with the laser trigger. (`fighist2dtile57.py`)

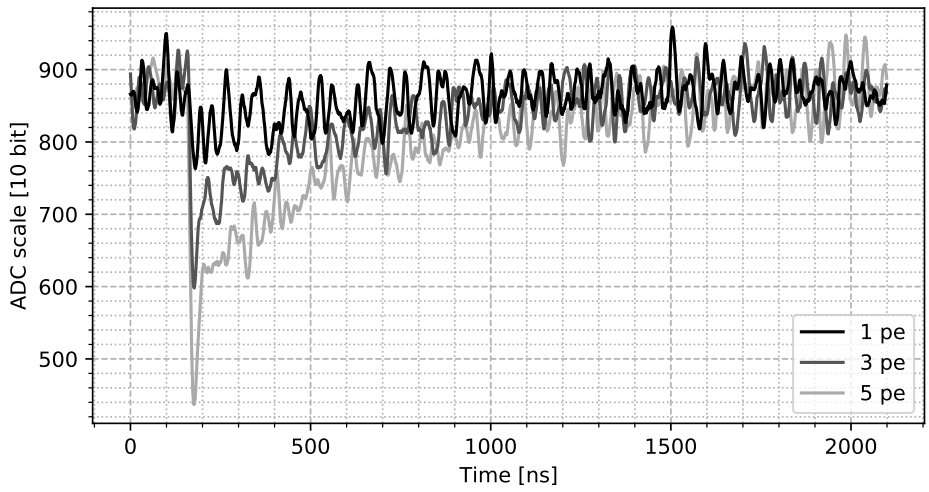


Figure 4.2: Single laser pulses. (`figsignals.py`)

these steps. So the 1 PE pulses are the lowest ones, the first non-straight band in the figure, then there are the 2 PE pulses, and so on.

In chapter 7 we will see that the PE do not correspond precisely to the number of photons, due to the presence of other avalanche-generating mechanisms, but for now we just care about the discretization. In Figure 4.2 we show some individual pulses with different PE.

4.2 Filters

A filter operates by converting the original sequence of ADC samples $(x_1, x_2, \ldots)$ to a new "filtered" sequence $(y_1, y_2, \ldots)$. The filters we consider are causal, i.e., the filtered sample y_n can be computed only using the original samples up to x_n. This limitation is introduced because we are interested in using the filters online, updating the filter output continuously as samples are read.

We tested three linear filters: the moving average, the exponential moving average (also known as autoregressive filter), and the cross correlation filter. These are standard filters and are the ones currently considered by the DarkSide collaboration for the implementation of the online readout system.

We will always normalize the filters to behave like an average, in the sense that a constant input waveform is mapped to itself. This is a convenient choice because it allows to subtract the baseline value after filtering.

All filters have a temporal scale parameter, which we will tune in the analysis.

4.2.1 Moving average

The moving average consists in taking the average of the last N samples:

$$y_n = \frac{1}{N} \sum_{i=1}^{N} x_{n-N+i}. \tag{4.1}$$

It is efficient to compute because at each step the average can be updated by subtracting the N-th past sample and adding the incoming one:

$$y_{n+1} = y_n + \frac{1}{N} x_{n+1} - \frac{1}{N} x_{n-N+1}. \tag{4.2}$$

The memory requirement is a buffer with the N samples.

Since the SiPM output is a current signal, the moving average is computing the charge in a given time window, so it is also called "charge integration" in this context.

4.2.2 Exponential moving average

The exponential moving average weighs past samples with an exponentially decaying coefficient, and can be written recursively as

$$y_n = a y_{n-1} + (1 - a) x_n, \quad a \in (0, 1). \tag{4.3}$$

The scale of the exponential decay is given by

$$\tau = -\frac{1}{\log a}, \tag{4.4}$$

$$\approx \frac{1}{1 - a} \text{ for } a \text{ close to 1.} \tag{4.5}$$

This is the most efficient filter possible since it processes exactly once each sample and requires no working memory.

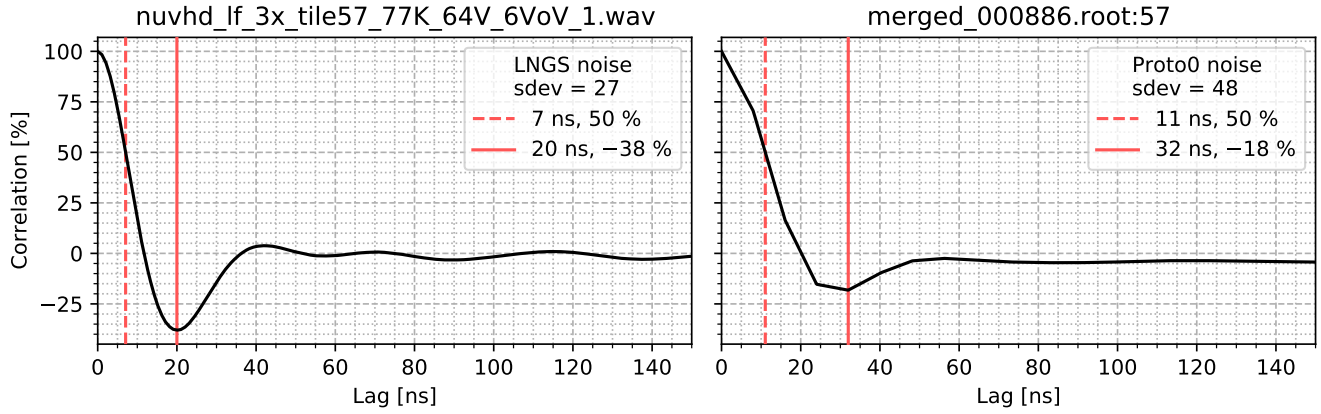


Figure 4.3: Autocorrelation of the LNGS and Proto0 noise, computed on the part of the events before the trigger. The autocorrelation at lag t is the correlation between a sample and another sample occurring a time t after the first. (`figautocorrlngs.py`)

4.2.3 Cross correlation filter

The cross correlation filter consists in computing the cross correlation of the input with the expected signal shape. In some texts this is called matched filter, term which we reserve for a different case explained later.

Let $\mathbf{h} = (h_1, h_2, \ldots, h_N)$ be a *template* of the signal waveform we want to detect. This means $\mathbf{h}$ should ideally match the shape of the signal waveform we want to find in the noisy data. The filter is then the cross correlation of $\mathbf{x}$ with $\mathbf{h}$:

$$y_n = \sum_{i=1}^{N} h_i x_{n-N+i}, \quad \sum_{i=1}^{N} h_i = 1. \tag{4.6}$$

Under the assumption that the data is white noise plus a signal that perfectly matches the template apart from amplitude, this filter is optimal in the sense that in the filter output there will be a peak corresponding to the signal and this peak will have the maximum possible height relative to the standard deviation of the filtered noise.

The differences we have from the ideal case are:

1. the shape of the signal probably changes a bit each time;
2. the signal is not aligned always in the same way to the ADC clock;
3. most importantly, the noise is not white.

We defer to section 4.4 the calculation of the template $\mathbf{h}$. See Figure 4.4 for an example of filtered waveform.

4.2.4 Matched filter

With the cross correlation filter, a non-white noise spectrum can be kept into account by appropriately transforming $\mathbf{h}$, in this case the filter is called *matched filter* [Fer15, p. 161]. Let $\mathbf{s}$ and $\mathbf{w}$ be the signal and noise, such that the waveform to be filtered is $\mathbf{x} = \mathbf{s} + \mathbf{w}$. Let V be the noise covariance matrix, i.e., $V_{ij} = \mathrm{Cov}[w_i, w_j]$. Then the template for the matched filter is

$$\mathbf{h}_\mathrm{m} = V^{-1}\mathbf{h}. \tag{4.7}$$

This filter is in some sense equivalent to doing a linear least squares fit of the data with one parameter which multiplies the model $\mathbf{h}$. This can be seen

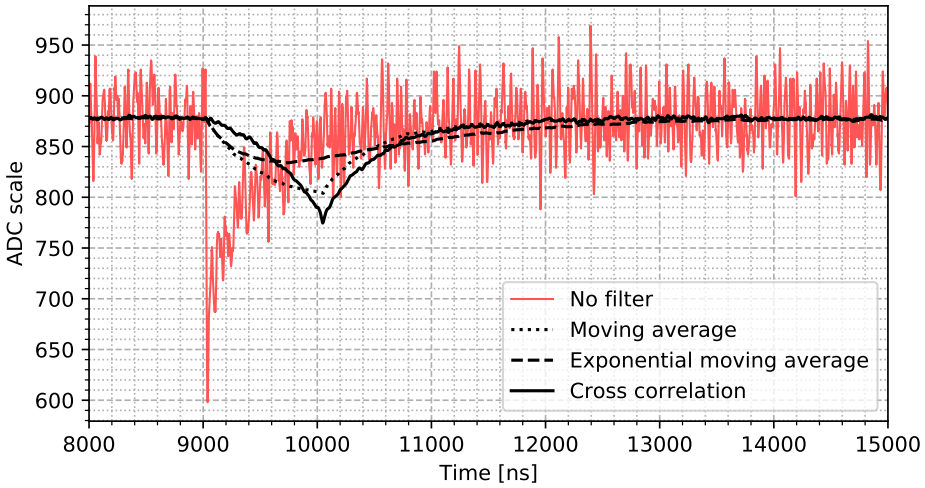


Figure 4.4: A filtered event. The number of averaged samples in the moving average filter, the scale of the autoregressive filter, and the length of the template of the cross correlation filter are all 1024 samples, at 1 GSa/s. (`figfilters.py`)

by writing the cross correlation with the template, $\mathbf{h}^\top V^{-1}\mathbf{x}$, which compared to the least squares estimator [Zyl+20, p. 628] is missing the multiplicative factor $1/(\mathbf{h}^\top V^{-1}\mathbf{h})$.

We tried implementing the matched filter with mixed results, and we will not report these studies in detail. We computed the noise covariance matrix on the event samples before the signal. Since the noise is stationary, V is a Toeplitz matrix, i.e., the covariance depends only on the lag between two samples. In Figure 4.3 we show the correlation of LNGS and Proto0 noise.

The matched filter worked slightly better than the cross correlation filter, as expected, but only for filter length N sufficiently small, getting worse as N increased. This could be due to numerical accuracy problems in solving the linear system V in Equation 4.7 as the size of the matrix increases.

The template obtained from the transformation is much different than the initial signal shape. There are large cancellations, so it is more sensitive to numerical accuracy, and may not be appropriate for an implementation on FPGA. Moreover, it has not been considered by the rest of the collaboration, so we did not investigate it further, and limited ourselves to the characterization of the proposed filters.

4.3 The fingerplot

To measure the performance of filters, we define a signal to noise ratio (SNR) variable as follows: the SNR is the ratio of the value that, on average, the peak of the filtered waveform yields on 1 PE pulses (without noise), over the standard deviation of the filtered noise.

We could consider other similar measures, for example we could include the standard deviation of the peak filter output value since that influences where we should place a threshold to discriminate signals, however it is sufficient to use any reasonable definition for the purpose of comparing different filters. Occasionally we computed the SNR without strictly adhering to the definition above, and this will be signaled in the text where appropriate.

For each event we compute the filter output at a fixed temporal delay from the leading edge of the laser trigger pulse, see Figure 4.5. Then we compute the average of the samples before the trigger pulse and take that value as "baseline". We subtract the baseline from the filter output, and

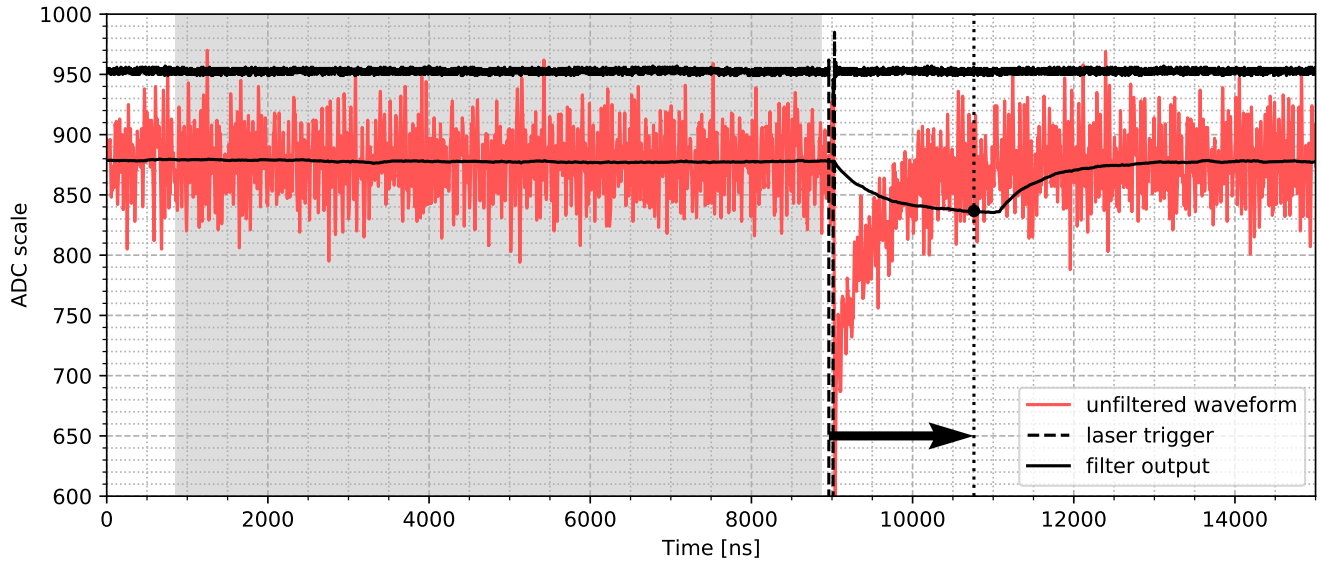


Figure 4.5: Example of how a signal amplitude is computed in an event for the purpose of computing the SNR. The samples in the shaded region are averaged to compute the baseline. The filter is evaluated at a fixed offset, indicated by the arrow, from the leading trigger edge. The amplitude then is the difference between the baseline and the filter value. The shaded region ends 100 samples before the trigger. (`figfiltersample.py`)

finally we change the sign to obtain positive values since the signals are negative.

We take the list of these baseline and sign corrected filter outputs and compute an histogram. One of these histograms is shown in Figure 4.6. It is called "fingerplot" due to the descending peaks reminding of fingers.

The first peak is centered in zero and thus corresponds to cases where no laser photon is detected. Since the instant where we are evaluating the filter output in each event is independent of the output itself, instead of, e.g., being determined by peak finding, this peak is the distribution of the noise after passing through the filter.

The various other peaks correspond to an increasing number of PE. The second peak is the distribution of the filter output at a fixed instant for 1 PE signals. If the instant where we evaluate the filter is the one associated to the highest signal response, the mean of the second peak divided by the standard deviation of the first yields the SNR.

Since the peaks are often overlapping, and that we will have to repeat the calculations for many fingerplots automatically without checking each one, we use robust measures of location and scale instead of the average and standard deviation. We run a peak finding algorithm on the fingerplot and divide the data by putting boundaries halfway between peaks, so that we assign each datapoint to a peak. For the second peak, we take the median instead of the average. For the first peak, we take half of the symmetric 68 % interquantile range instead of the standard deviation, i.e., half the difference between the 0.84 and 0.16 quantiles.

On a Gaussian distribution these are equivalent to the mean and standard deviation, however they are less sensible to the tails of the distribution, which is a desirable property since the boundaries could cut away the tails and there could be contamination from the tails of the neighboring peaks. By definition, when the SNR is high, the peaks are well separated, so in that case the result can be trusted to match precisely the quantiles of the single

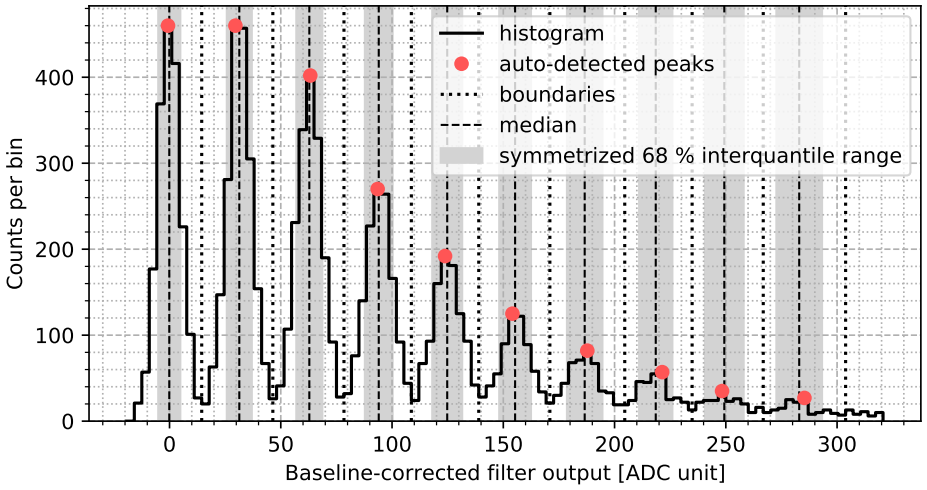


Figure 4.6: The fingerplot for the moving average filter with 128 samples evaluated 128 samples after the trigger. (`figfingerplot.py`)

isolated peak.

Moreover, the code always automatically checks that the median of the 0 PE peak is zero within its error. If it is not, the SNR is considered to be zero, such that it is well visible in the results.

4.4 Cross correlation filter template

We compute the template for the cross correlation filter by taking the average of 1 PE signals, selected using the fingerplot boundaries obtained with a 1500-samples average, counting the samples starting from the trigger position. The baseline is computed with the median. We consider three alignment methods for the waveforms prior to averaging:

- unaligned, or rather aligned to the acquisition window;
- aligned using the laser trigger;
- aligned using a cross correlation filter done with the unaligned template, where the alignment offset is the position of the minimum in the filter output.

We show the obtained templates in Figure 7.7. The trigger square pulse is sampled at 1 GSa/s like the PDM output, so we assume that it gives a precise alignment. Indeed, the pulse peak appears sharper than with the unaligned template. The filter-aligned template has an additional sinc-like oscillation centered on the peak. It is similar to the autocorrelation of the noise, see Figure 4.3, so we interpret it as the effect of a non-null coherence in the averaged noise, due to the fact that the position of the filter peak minimum will depend both on the position of the signal and on the overlapping noise oscillations, which have a width similar to the one of the signal peak.

The trigger-aligned template appears to be the best one, so we will use it in this chapter and in chapter 5. However chapter 7 analyzes data without trigger information, forcing us to use the filter-aligned template in that case. See [CH01] for an alignment method that may offer some improvements over our vanilla approach.

When using the template, we normalize it to unit sum, as explained in section 4.2. Since we will try various lengths of the filter template, we have

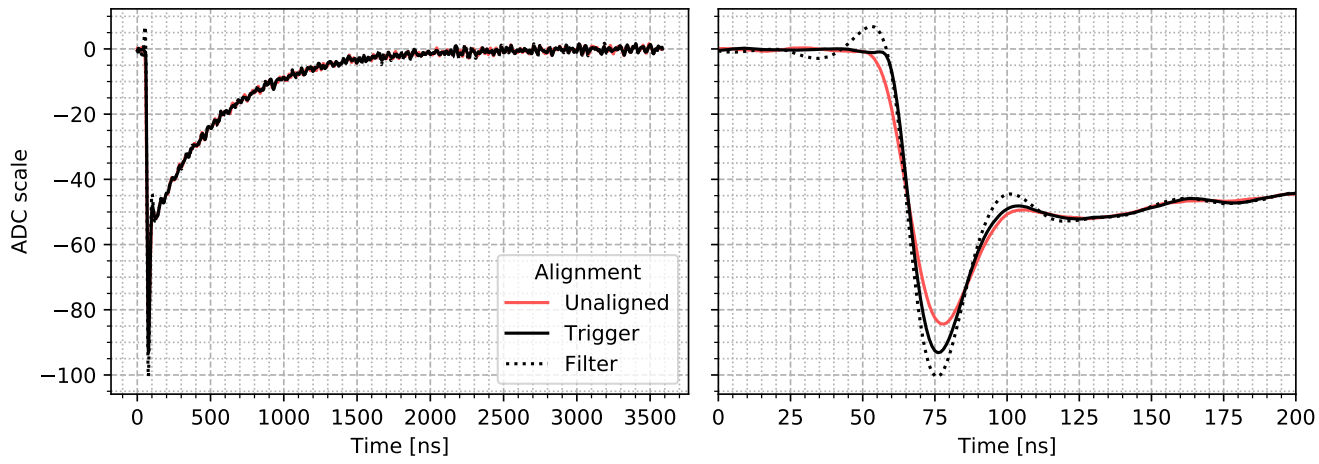


Figure 4.7: Templates obtained averaging 1 PE laser pulses, selected with a 1.5 µs charge integration fingerplot, with different pre-averaging alignment choices. The template we pick is the trigger-aligned one. (`figtemplate.py`)

to decide how to truncate the full template. When truncating to N samples, we pick N contiguous samples from the template such that their sum of squares is maximized. Of course normalizing the template to unit sum is done after truncation.

We did not smooth the template nor fitted it with a model. For how to fit this kind of pulses, we refer to, e.g., [Luz20]. We think that our vanilla average is fine enough, but we did not actually compare it with a fitted template.

We conclude this section with a technical detail: when computing the filter-aligned template, we first smooth the unaligned template with a 2-sample moving average. This is to remove a coherent noise at the Nyquist frequency, i.e., even samples are always biased in the same direction relative to odd samples, in all waveforms. If we do not correct this, the noise appears in the template, then when using it the filter matches the noise in template to the one in the waveform, and gives a peak output only on even or odd samples.

4.5 SNR versus filter length

When determining how to compute the SNR from the fingerplot we assumed that we were evaluating the filter output at the optimal instant. Since we do not know it a priori, we repeat the computation for a range of values of the filter evaluation point. A simpler solution that comes to mind is taking the minimum (the signals are negative) of the filter output in each event, however this would bias upward the SNR measure because the minimum can yield lower values due to noise peaks. Instead, by fixing the point independently from the data, the random oscillation due to noise is symmetrical and the averaging recovers the actual amplitude of the filter output for the signal.

The resulting SNR curves are shown in Figure 4.8, repeated for a range of values of the filter-length parameter. For the moving average and cross correlation filter, the length parameter is the number of samples, N. For the exponential moving average, it is the scale of the exponential decay τ.

The maximum of each curve gives the actual SNR figure we are interested in. We expect by intuition that the width of the maximum is approximately proportional to the temporal resolution we could achieve if we used the filter

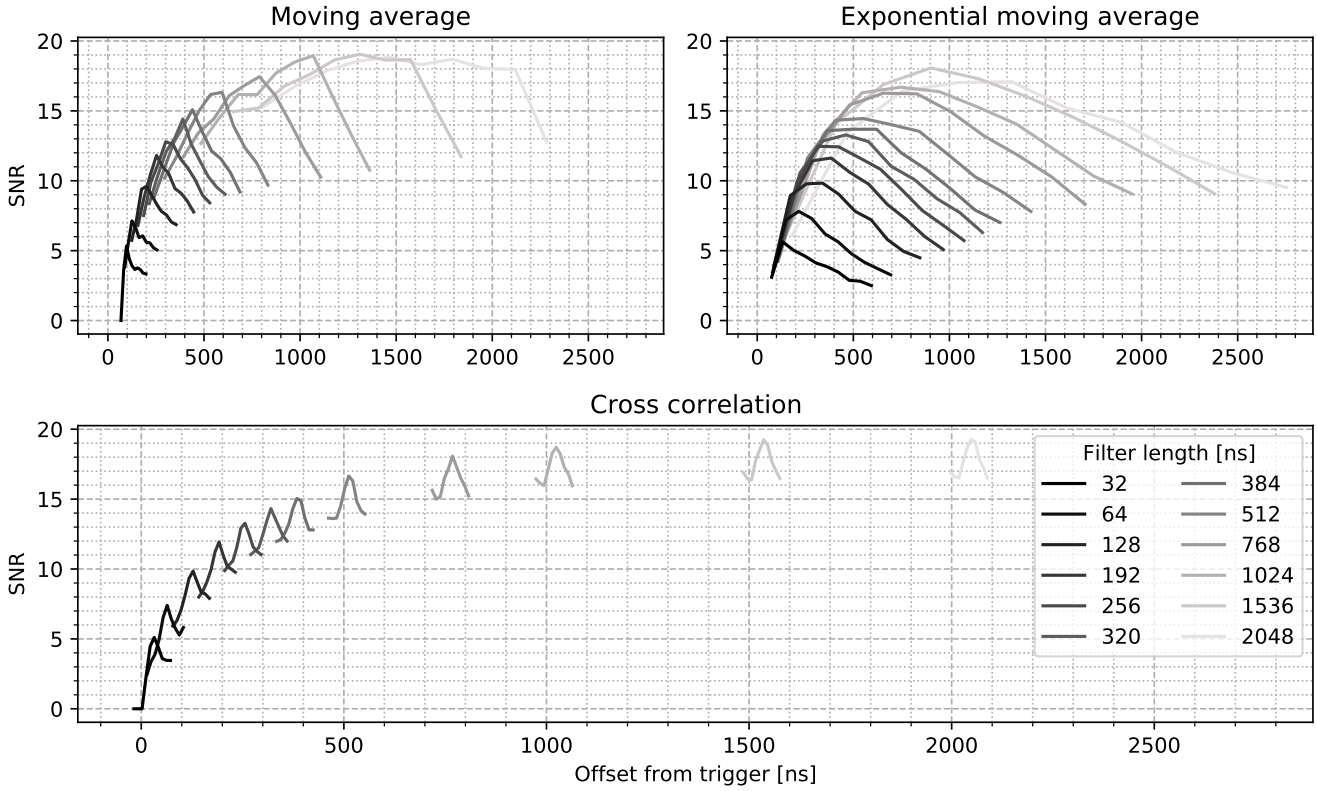


Figure 4.8: The SNR as a function of delay from trigger (x-axis) and filter length (shade of gray) for the three filters. (`figsnrplot.py`)

output to locate temporally the signal. So we do another plot, Figure 4.9 left column, where we show the maximum SNR value and the width of the peak versus the filter length parameter. We measure the width as the distance between the two points where the SNR is 1 less than its maximum value.

4.6 Effect of the baseline computation

In constructing the fingerplot, we subtract from the filter output value the baseline, i.e., the average value of the waveform in absence of signals. We compute the baseline for each event as the average of the samples before the signal. As a default, we average 8000 samples. The value obtained varies randomly due to the noise; its standard deviation is that of the noise divided by $\sqrt{8000}$. The maximum filter length we use is 2000, so in that case the width of the peaks gets an additional contribution (summed in quadrature) of roughly $\sqrt{2000/8000} = 1/2$ of the width we would have with a noiseless baseline, i.e., the width of the first peak is $\sqrt{1 + 1/2} - 1 = 22\,\%$ larger, lowering the SNR by the same percentage. We note that, since the noise is autocorrelated, the standard deviation of the mean does not scale with $1/\sqrt{N}$, giving a more detrimental effect on the SNR than the one just estimated, and the width of the 0 PE peak depends on the filter.

These considerations are relevant to real-world applications, because the baseline computation is a relevant part of the signal finding, and it requires either a fast on-digitizer algorithm estimating it reliably or sending enough samples to subsequent stages in the data processing chain. The value of the baseline is expected to change in the detectors, so it may not be feasible to compute it once with a very long average and then keep it fixed for a while.

To get an idea of the effect of the precision of the baseline calculation,

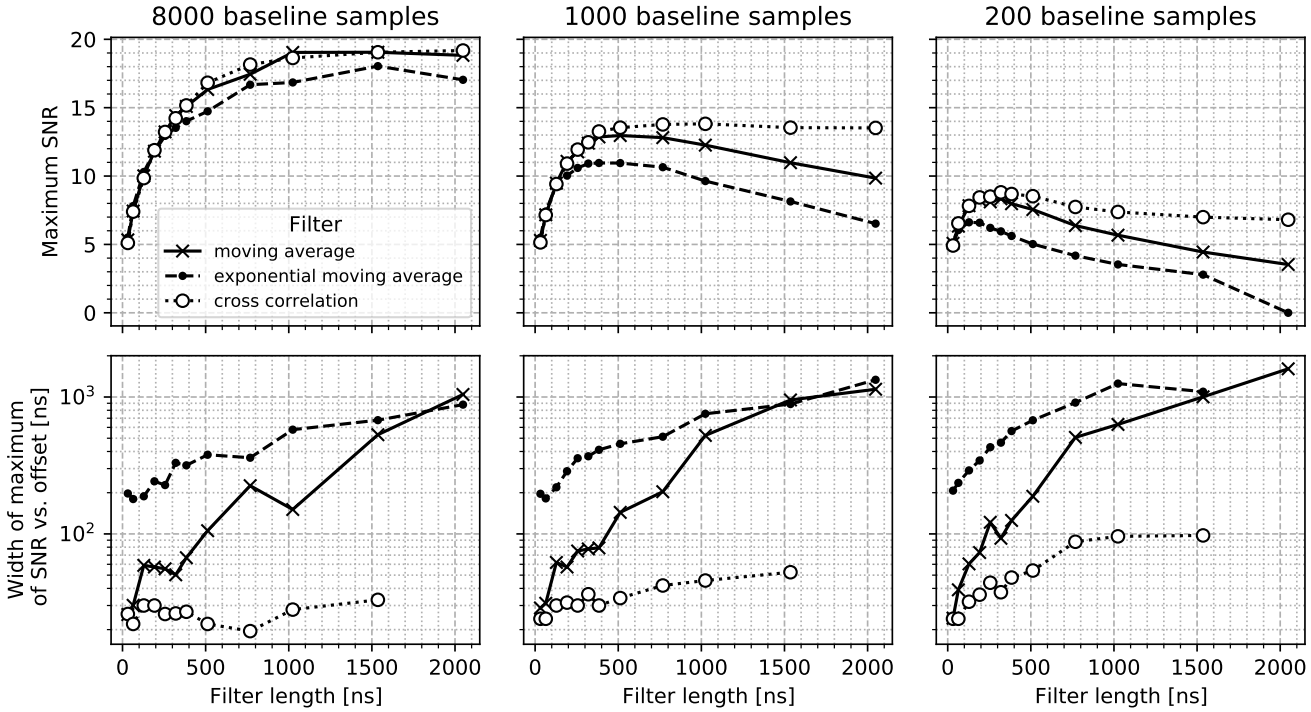


Figure 4.9: Top row: maximum SNR achieved with each filter (different curves) and filter length (x-axis). Bottom row: the width of the maxima, computed as the length of the interval of offset from trigger with endpoints where the SNR is 1 less than the maximum. The cross correlation filter maintains almost constant width with increasing SNR, while the other two filters have increasing width. In each column the computation is repeated changing the number of pre-signal samples used to compute the baseline. The performance decreases with a lower number of samples because the error on the baseline increases and the baseline is compared to the filtered signal amplitude to discriminate signals from noise. (`figchangebs.py`)

we repeat everything computing the baseline with just 1000 or 200 samples. The result is in the central and rightmost columns of Figure 4.9. We see that, for example, going from 8000 to 200 baseline samples the maximum SNR drops from 19 to 9.

It may appear strange that, with an imprecise baseline, the SNR starts dropping with increasing filter length at somewhat short lengths. We note that the filters behave like an average, so their peak value on the signal decreases as they get longer. The filtered noise contribution decreases too, but the baseline error is constant.

4.7 Noise spectrum

To support our previous statement that the noise is not white, in this section we report the frequency spectrum of the noise.

We take the region of each event before the trigger pulse, compute its periodogram, and take the median across events for each frequency. We use the median instead of the average in case there were spurious signals or irregularities we missed. We do the same for the same tile (57) in Proto0 data, with no need to isolate the pre-trigger region in that case. The spectra thus obtained are shown in Figure 4.10. We also compute the spectrum of the signal template and compare it to the noise in Figure 4.11. Note that we do not window the signal because it is not a stationary process.

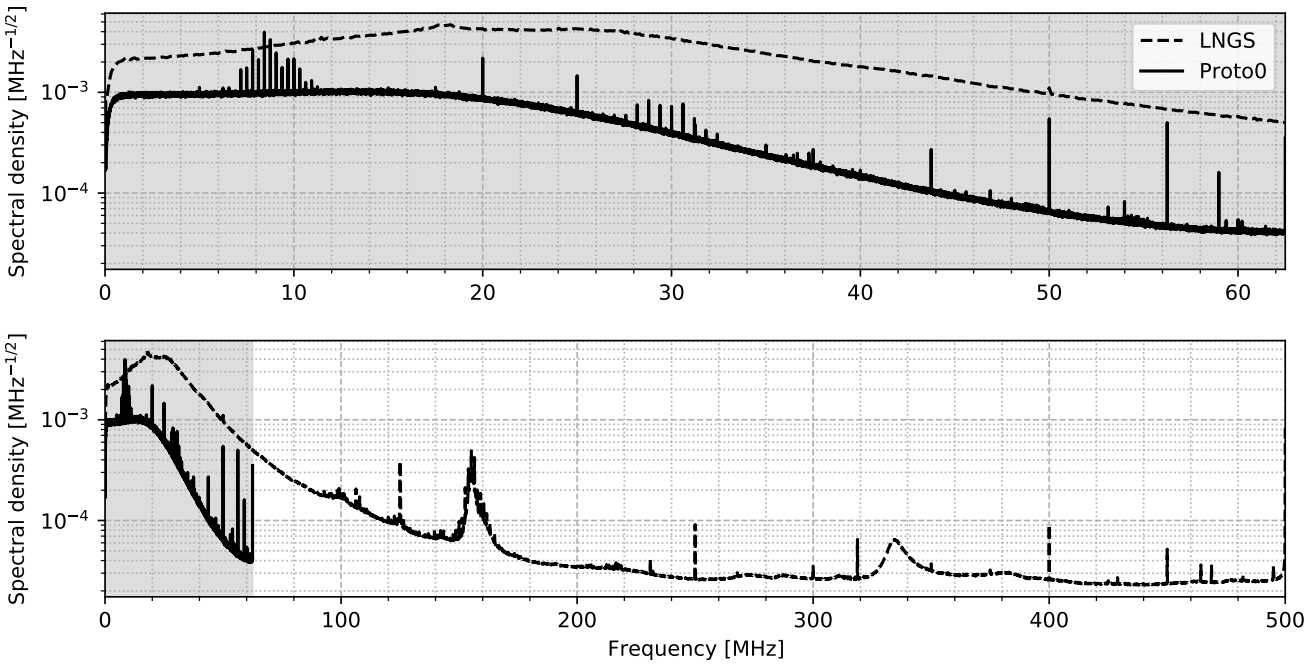


Figure 4.10: Noise spectrum of Tile 57 in LNGS and Proto0. The vertical scale has arbitrary units. (`figspectra.py`)

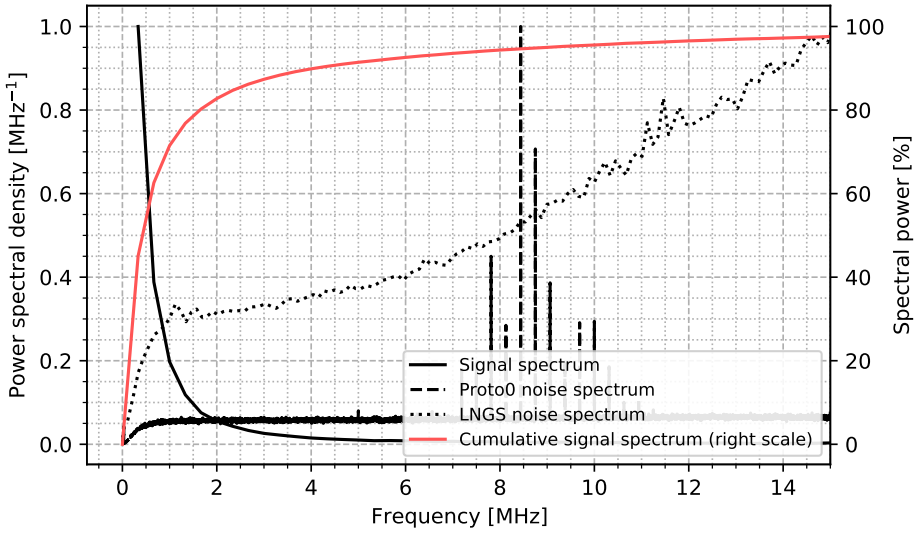


Figure 4.11: Comparison of the spectra of signal and noise. The red curve shows the percentage of signal power below a given frequency. The signal spectrum is computed with the discrete Fourier transform of the cross correlation filter template (Figure 4.7) without windowing. (`figtemplsp.py`)

Chapter 5

Temporal resolution of pulse detection

In this chapter we measure the temporal localization precision once the presence of a signal pulse is established.

To this end, we simulate events each containing only one signal at a known position. In principle we could use the LNGS data (section 3.2), but we do not know the jitter of the trigger pulse and we may reach a temporal resolution below the sampling period, while in the simulation we know the exact actual temporal location of signals.

5.1 Event simulation

Every event is the sum of a 1 PE signal and a noise waveform. We do not add a baseline, so the noise has mean zero and the signals taper down to zero. The signals are negative. We use the same scale of the LNGS data; the scale does not affect the results of this study since we are not simulating digitalization. The following paragraphs describe in detail the event simulation procedure.

5.1.1 Signal generation

We generate the signal pulse shape according to the trigger-aligned template from 1 PE laser pulses in Tile 57, see section 4.4 and Figure 4.7.

The template is sampled at 1 GSa/s, but the simulated events are sampled at 125 MSa/s, which is the sampling frequency planned for the DarkSide20k digitizers. The randomly generated signal time is not required to be aligned with either clock. Given the generated temporal position, we round it by excess *and* defect to the 1 ns clock tick. Then we downsample the template by averaging samples in groups of 8. This is done twice: once with the groups aligned to the floor-rounded temporal position, once with the ceiling-rounded one. Finally we interpolate linearly between the two downsampled templates. Figure 5.1 shows a series of waveforms generated following this procedure. Here averaging before downsampling has the role of an antialias filter. While more refined antialiasing filters exist, a simple average is sufficient for our application.

In each simulation event we vary the amplitude of the signal by an additive Gaussian random variable, which has the standard deviation observed in the LNGS data, 2.9 % of the average 1 PE amplitude. This value is obtained by computing the difference in quadrature between the "quantile standard deviations" (see section 4.3) of the 1 PE and 0 PE peaks in the fingerplot done with a 1.5 µs average, the same used for the template in section 4.4, and dividing it by the median of the 1 PE peak.

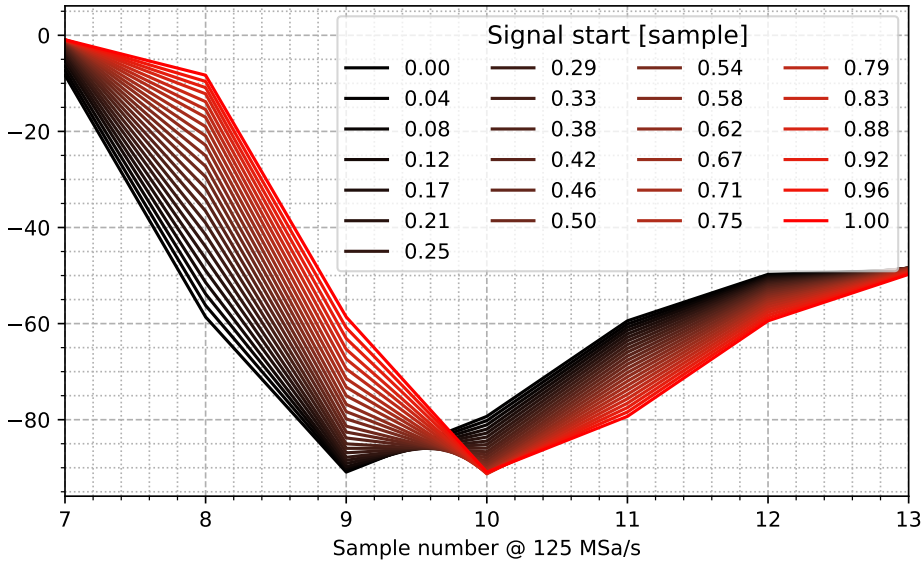


Figure 5.1: The signal template downsampled from 1 GSa/s to 125 MSa/s and translated continuously instead of by discrete steps with linear interpolation. (`figinterptempl.py`)

5.1.2 Noise simulation

To study the dependence of the algorithms on noise, we simulate three different noise distributions: Gaussian white noise; noise sampled from the LNGS data; noise sampled from the Proto0 data.

The white noise is generated in the simulation. The LNGS noise is sampled from the pre-trigger region of Tile 57 data, the same data used for the signal template, ignoring any event with any sample less than 700 as in section 4.1. The Proto0 noise is copied from data collected operating Tile 57 below the breakdown voltage, keeping the whole events without selection. A persistence plot of the Proto0 data is shown in Figure 6.6. The spectra are shown in section 4.7, the autocorrelations in Figure 4.3.

The preprocessing applied on the pre-trigger window of LNGS data does not fully reject 1 PE pulses, and spurious 1 PE pulses may appear in the simulation. In the analysis we will use robust statistics, i.e., quantiles, to deal with outliers caused by this or any other unanticipated feature of the data.

When the noise waveforms for multiple simulated events are extracted from the same data event, we skip 1 µs between each waveform segment in the data, to avoid correlations between the simulated events.

We downsample the noise in the same way as the signal, by averaging nearby samples. The noise spectra from both sources (LNGS and Proto0) decrease with frequency, so an antialiasing with an average should suffice. The Proto0 data, as is available to us, is pre-downsampled without antialiasing from 250 MSa/s to 125 MSa/s.

After downsampling, the noise obtained from data is normalized to the desired variance. The order matters because downsampling with averaging reduces the variance of the noise, see Figure 5.2. We normalize the variance separately for each *data* event, also fixing the mean to zero, such that in the simulated events the variance has a realistic variation.

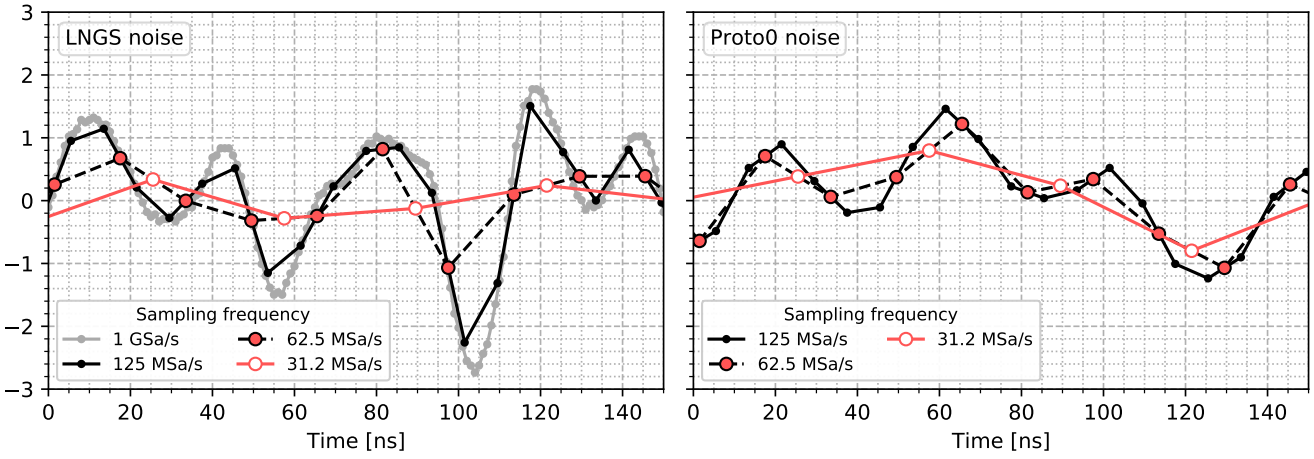


Figure 5.2: The LNGS and Proto0 noise at the original sampling frequency (normalized to zero mean and unit variance) and downsampled. (`fignoise.py`)

5.1.3 Event layout

Each event is the sum of a noise waveform and a shorter signal waveform. Before the beginning of the signal there is a noise-only region which is chosen long enough for the filters to be in a stationary state when the signal occurs. The length of this region is set to be 2304 ns, i.e., longer than the largest filter-length parameter considered in our study, 2048 ns.

The simulation is repeated for various raw signal to noise ratios (SNR), calculated as the peak height relative to the baseline of the original 1 GSa/s signal template over the noise standard deviation.

The reason we consider the signal amplitude at 1 GSa/s and not at the actual sampling frequency of the simulation is because downsampling reduces the peak height. For convenience, we want to keep the definition of signal height comparable at different sampling frequencies.

Simulations with different raw SNR differ only in the multiplicative constant of the noise, so we use exactly the same noise and signal arrays for every SNR to speed up the code. This means that there is no random variation between results obtained at different SNR (or with different filters), keep this in mind if the smoothness of some curves would seem to suggest that the Monte Carlo error is negligible.

Figure 5.3 shows a complete example event.

5.2 Temporal localization

To reconstruct the time position of the signals, We run the three filters described in section 4.2 (moving average, exponential moving average, cross correlation), and take the minimum (the signals are negative) of the filtered waveform as the location of the signal. We also take the minimum of the unfiltered waveform as a baseline comparison.

The minimum of the filter output occurs at a shifted position relative to the signal location, but this is not a problem since the choice of the point of the signal to be taken as reference is arbitrary, and, for each filter, the shift is constant from one event to another.

To build the template for the cross correlation filter, we first truncate it as described in section 4.4, and then downsample it in the same way we downsample the signal template and the noise.

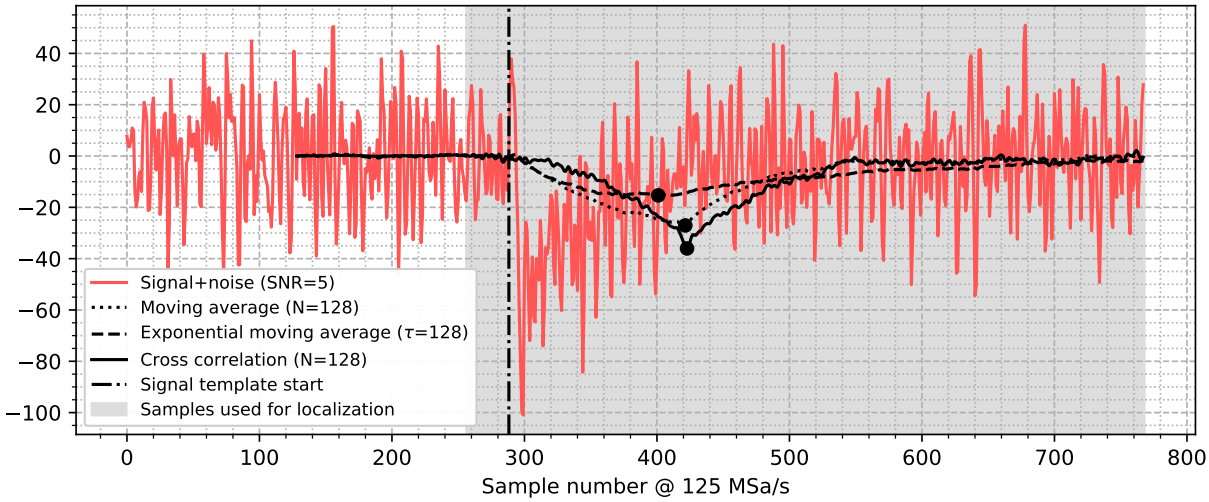


Figure 5.3: A simulation event. The dots are the minima of the filters output. The minima are searched in the shaded region only; this makes no difference with high enough SNR like in this example, but in the limit SNR = 0 the minimum fluctuates around uniformly: the search range sets the endpoints of this distribution. (`figtoyevent.py`)

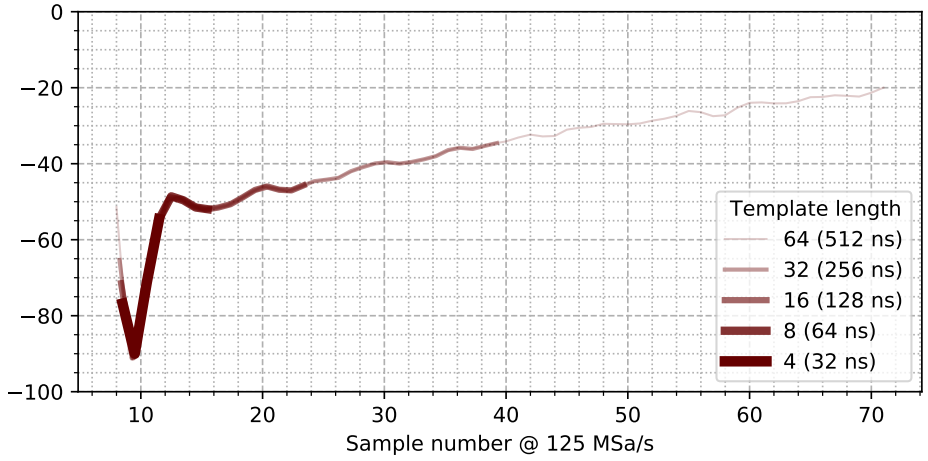


Figure 5.4: Some cross correlation filter templates for different lengths. It may appear strange that the endpoint on the left has a different height than the endpoint on the right for a given template, since we choose the truncation to maximize the norm; it happens because we downsample *after* truncation. (`figtoyfilttempl.py`)

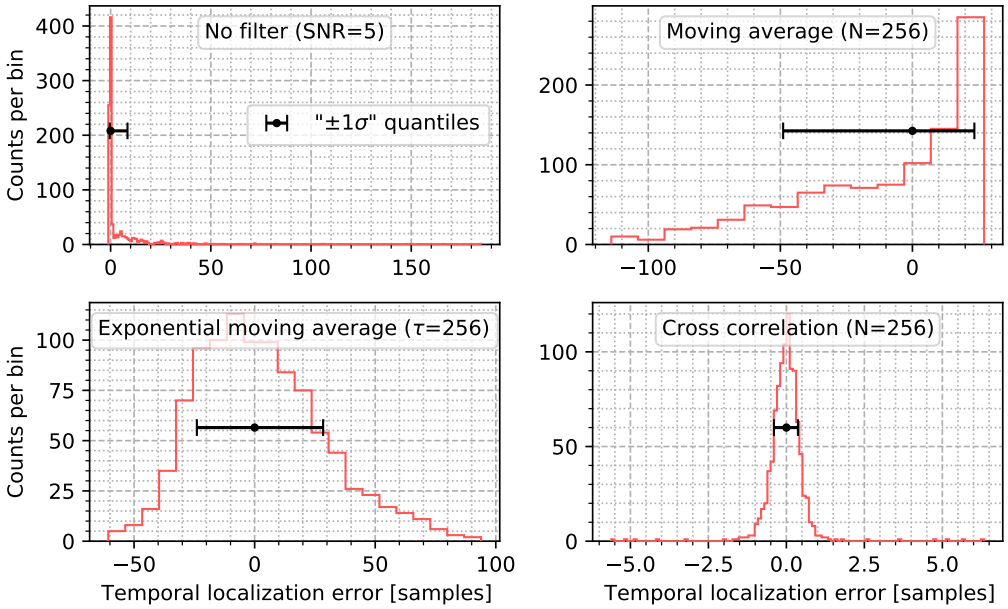


Figure 5.5: Histograms of the temporal localization error, i.e. the difference between the filter output minimum and the signal template start, translated to have zero median, for a choice of SNR and filters length. The error bars mark the 16 % and the 84 % quantiles. As definition of temporal resolution we take half the distance between those quantiles. The sampling step is 8 ns. (`figlochist.py`)

To allow for a localization more precise than the sampling clock bin, we interpolate the minimum sample and its first neighbors with a parabola. We also try upsampling the waveform to 1 GSa/s (with sample repetition) prior to filtering to check if it improves performance.

5.2.1 Time resolution results

Assuming Proto0 noise spectrum, we simulate 1000 events and repeat the time position reconstruction varying the filter, filter length parameter, and raw SNR. The signal template position is generated uniformly within one clock bin. Figure 5.5 shows the histograms of the temporal localization for all filters for a choice of SNR and filter length.

We see that the distribution of reconstructed signal time positions can be non-Gaussian, so to quantify the resolution we use, instead of the standard deviation, half the distance between the 16 % and 84 % quantiles, which is equivalent to a standard deviation for a Gaussian, but gives a meaningful measure for the width of the distribution even when it is highly skewed or with heavy tails.

Figure 5.6 shows the temporal resolution thus defined for each filter, filter length, and raw SNR. The exponential moving average has a consistently poor performance compared to the other filters. The cross correlation filter is the best one, with performance improving with filter-length, and at a length of 96 samples (768 ns) is already practically optimal. The moving average can get close to the cross correlation filter by choosing appropriately the number of samples.

In the experiment The online processing of the PDM output will happen in two steps: the digitizers must find the signals, then send them to the front end processors (FEPs) for further analysis. The computational resources of the digitizers are limited compared to those available in the FEPs.

The exponential moving average can be implemented on the digitizers

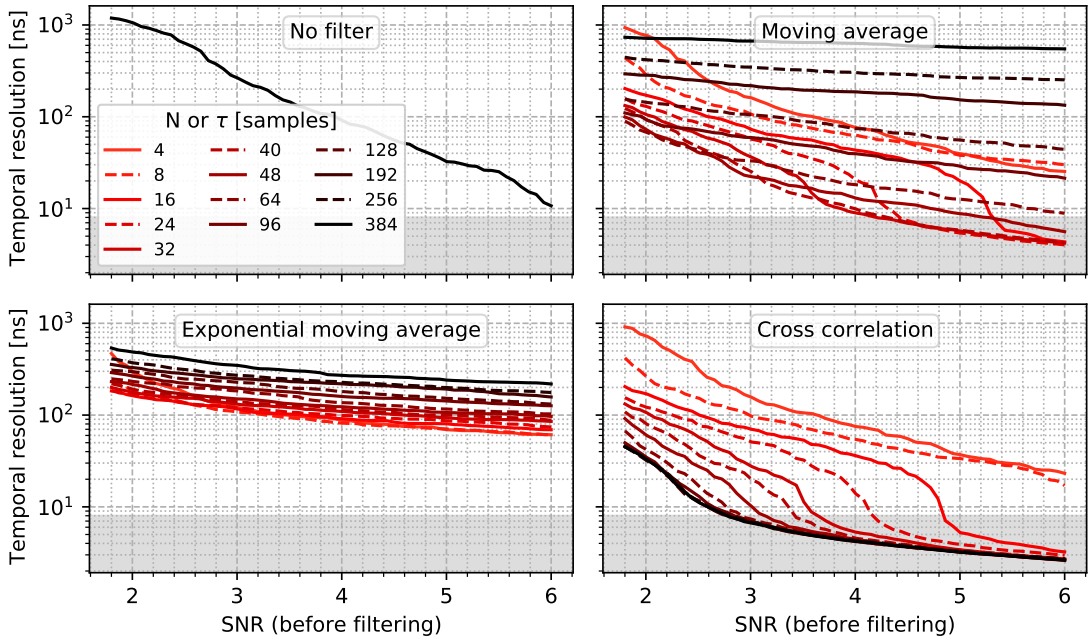


Figure 5.6: Pulse detection temporal resolution for a range of raw SNR and filter lengths. The shaded region marks the sampling step 8 ns. The right endpoint of the cross correlation filter curves is at 2.6 ns. (`figrescurve.py`)

with few logic resources. The cross correlation with 64 samples could probably be performed with the resources available on the digitizers since a computation with similar complexity was implemented in firmware and run on an evaluation card featuring the same FPGA installed on the DarkSide digitizer boards.

The FEPs can and should probably use the best filter, so they would run a long cross correlation filter, since achieving a good temporal resolution may be beneficial for offline analysis.

To summarize, out of all the temporal resolution curves the most relevant are:

- the best time resolution we can achieve with the exponential moving average and moving average;
- the long cross correlation filters;
- the 64 samples cross correlation filter.

We plot these curves together in Figure 5.7, adding the resolution plots obtained in the best configuration when changing the noise spectrum in the simulation, to show the impact of the noise spectrum on the performances. We note in particular that a different noise spectrum makes a large difference at low SNR. Finally, we plot a curve computed with and without upsampling, which shows that upsampling does not improve significantly the performance.

5.3 Data reduction

In this section we study the effect of some data reduction strategies at the digitizer level on the time resolution that can be achieved in subsequent data processing stages. We said that the digitizers must find signals in the waveform stream and send them to the FEPs for further processing. Depending on the background rate, the bandwidth of the connection between

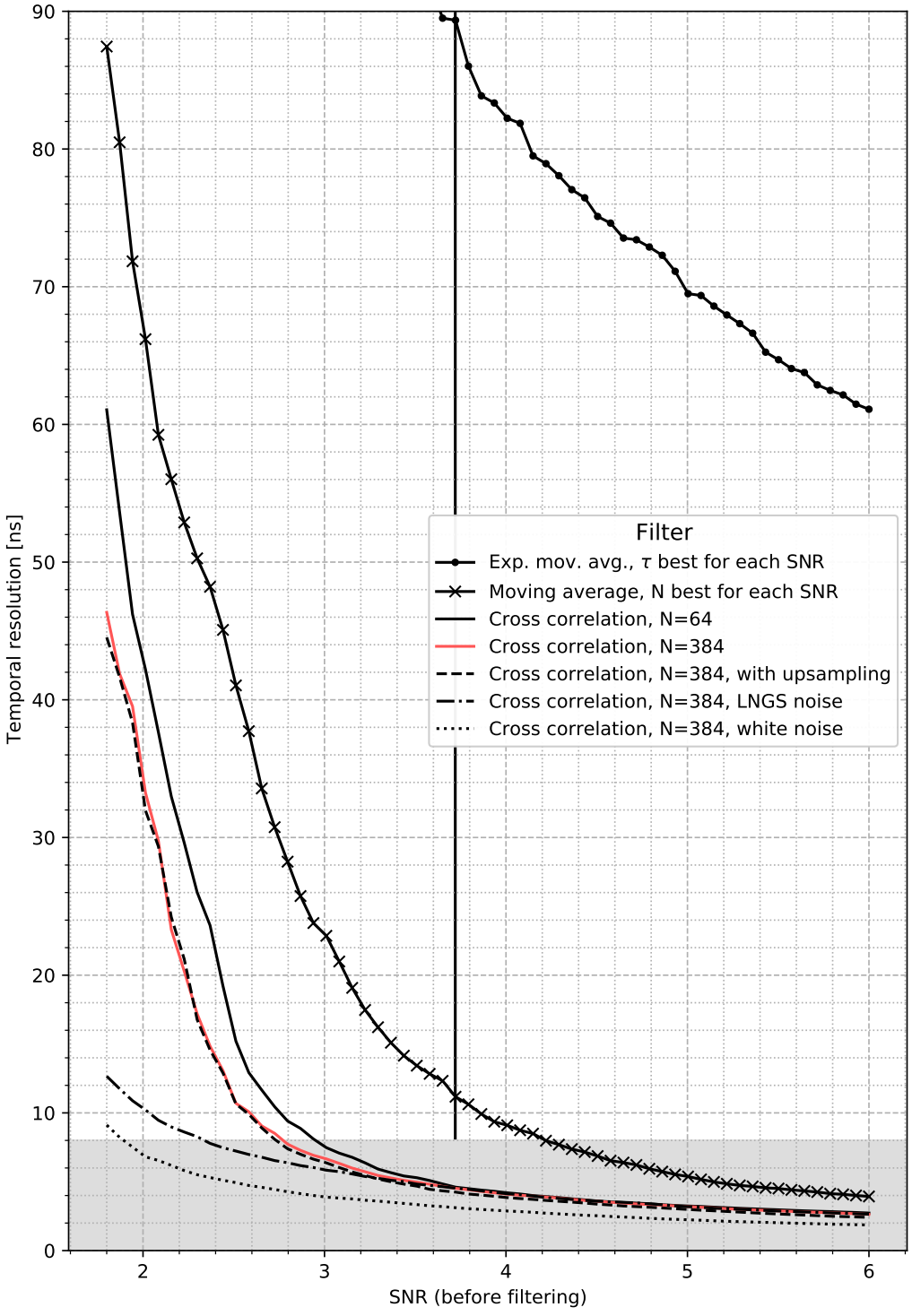


Figure 5.7: Pulse detection temporal resolution vs. SNR for various filters. The shaded region marks the sampling step 8 ns. The hatched band is the interval of SNR observed in Proto0; the vertical line is the SNR in the LNGS data, after downsampling to 125 MSa/s. Where not specified, the noise is from Proto0. (`figrescomp.py`)

the digitizers and the FEPs can be a bottleneck. Two possible ways of reducing the amount of transmitted data are keeping only the minimum number of samples around each signal, and reducing the sampling frequency. Both have an effect on the temporal resolution, which we assess in the next paragraphs.

5.3.1 Waveform truncation

We repeat the simulation, as in section 5.2, but this time we use only a fixed smaller number of samples in each event to compute the filter output. We call this selection of samples a "window". On the window we run only a long cross correlation filter since that is what would be done on the FEPs. As past and future boundary condition we use zero. We evaluate the filter even after the sample window end because the window can be shorter than the filter.

In this study we did not attempt any optimization of the left/right balance of the window. The number of samples to be stored *before* the onset of the signal is driven by the requirement to allow a proper baseline subtraction procedure. However, by applying the zero padding just described, we are implicitly assuming that a proper baseline subtraction procedure has been applied prior to running the filter. Thus, we only focus on determining the number of samples that should be saved *after* the onset of the signal, by considering windows that very skewed to the right. Keep into account that the measure we are looking at, the temporal resolution, does not depend critically on getting the baseline right.

While the length of the window is fixed, its placement is not fixed relative to the true signal location. Instead we use the temporal localization with another filter feasible on the digitizers, calibrated to have the median aligned to the beginning of the signal template. The window then extends for a given number of samples to the left and to the right of this localization.

Figure 5.8 shows this procedure graphically for a single event. Figure 5.9 shows the temporal resolution versus unfiltered SNR curves for various choices of window length, noise, and filter used to align the window, where for reasons of computation time the latter was computed at a fixed SNR that does not follow the value on the x-axis.

From Figure 5.9 we conclude that it would be necessary to save at least 1 µs of waveform after the onset of the signal to avoid degrading the temporal resolution. On the other hand, we also observe that the performances improve quickly to almost optimal ones when increasing the window length. In the top right panel, i.e., with Proto0 noise and centering with an exponential moving average, the resolution does not converge to the value without windowing as the window length increases. This is due to the standard deviation of the distribution of the window center, 17 Sa, being not small enough compared to the left window margin, 32 Sa. This means that in a non-negligible fraction of cases, the window does not include the leading edge of the signal. We show this problem intentionally to underline the importance of the left/right balance.

5.3.2 Downsampling

Another way of reducing the data throughput is downsampling. In Figure 5.10 we show the temporal resolution achieved with a long cross correlation filter at different sampling frequencies. The downsampling is computed averaging nearby samples. So, in other words, we are comparing applying the cross correlation at the full sampling frequency to first applying an an-

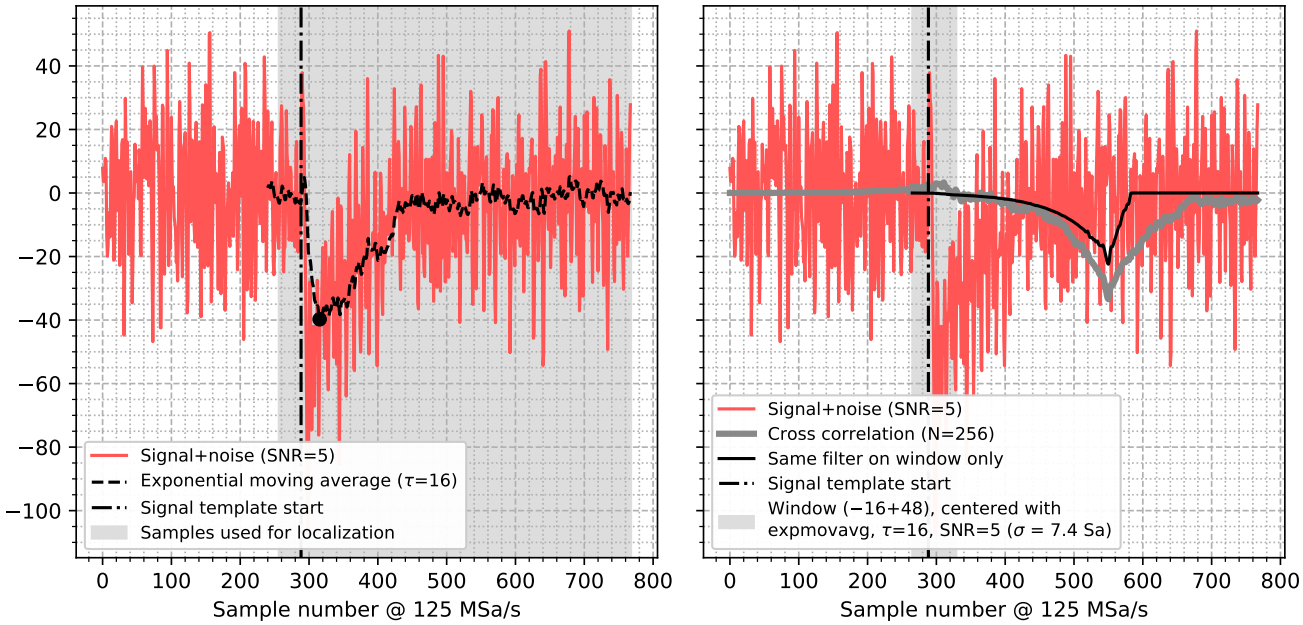


Figure 5.8: Left panel: a simulation event filtered with the exponential moving average. Right panel: the same event filtered with a long cross correlation filter, both using the whole waveform and using only the samples in the shaded window, which is centered using the localization from the filter in the left panel. `(figwindowevent.py)`

	SNR after over before filtering			
Noise	1 GSa/s	125 MSa/s	62.5 MSa/s	31.2 MSa/s
Proto0		3.3	3.3	3.3
LNGS	5.6	5.5	5.7	6.0
White	4.3	4.3	4.2	4.2

Table 5.1: Ratio of SNR after over before filtering with a cross correlation filter with template length 2048 ns. The 125 MSa/s column contains the actual SNR ratios of the simulations, while the values for the other sampling frequencies are divided by the noise standard deviation reduction with downsampling relative to 125 MSa/s to make them comparable.

tialiasing filter, downsampling and then computing the cross correlation with a downsampled template. We observe that downsampling by a factor of 2 from 125 MSa/s to 62 MSa/s maintains almost the same temporal resolution, while going to 31 MSa/s lowers it visibly.

When downsampling a waveform, the variance of the noise is reduced. At each sampling frequency the simulation sets the SNR looking at the standard deviation of the already downsampled noise, so the SNR scales are off by the factor of the noise amplitude reduction. To make the simulations comparable, we should start from a common "master simulation" at 1 GSa/s, then downsample it various times. Our code does not implement this and repeats the simulation from scratch at each sampling frequency, renormalizing the downsampled noise to unitary variance. To account for this, in Figure 5.10 we apply a correction factor on the raw SNR before plotting the time resolution results.

We also check if downsampling is associated to signal to noise ratio degradation in the cross correlation filter output. In Table 5.1 we report the ratio between SNR after and before filtering. It does not appear to change significantly.

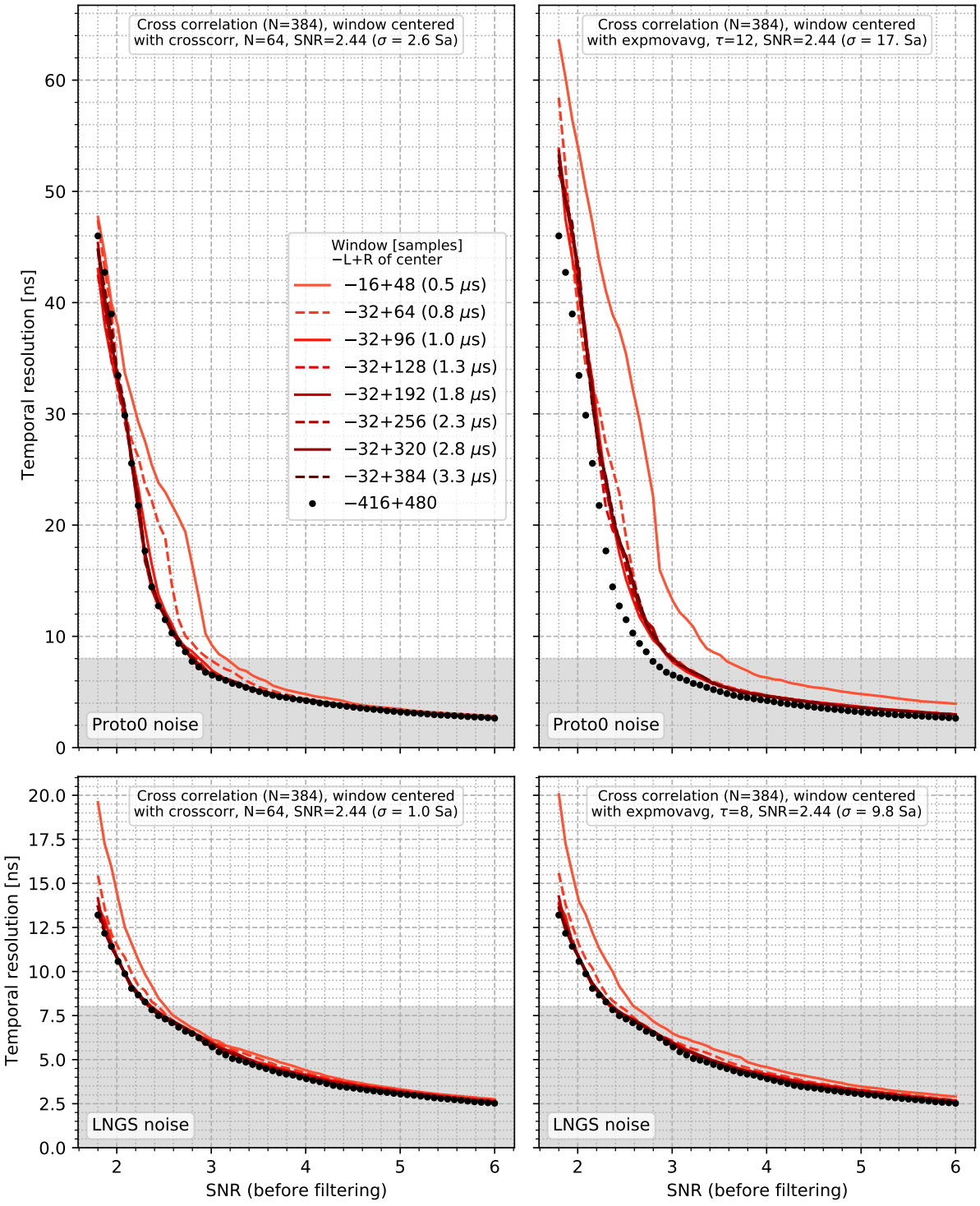


Figure 5.9: Pulse detection temporal resolution with a long cross correlation filter applied only on a short window of samples centered using a shorter cross correlation filter (left panels) or an exponential moving average (right panels). The various curves correspond to different window lengths, while the black dots are the resolution without windowing. (`figwindowtempres.py`)

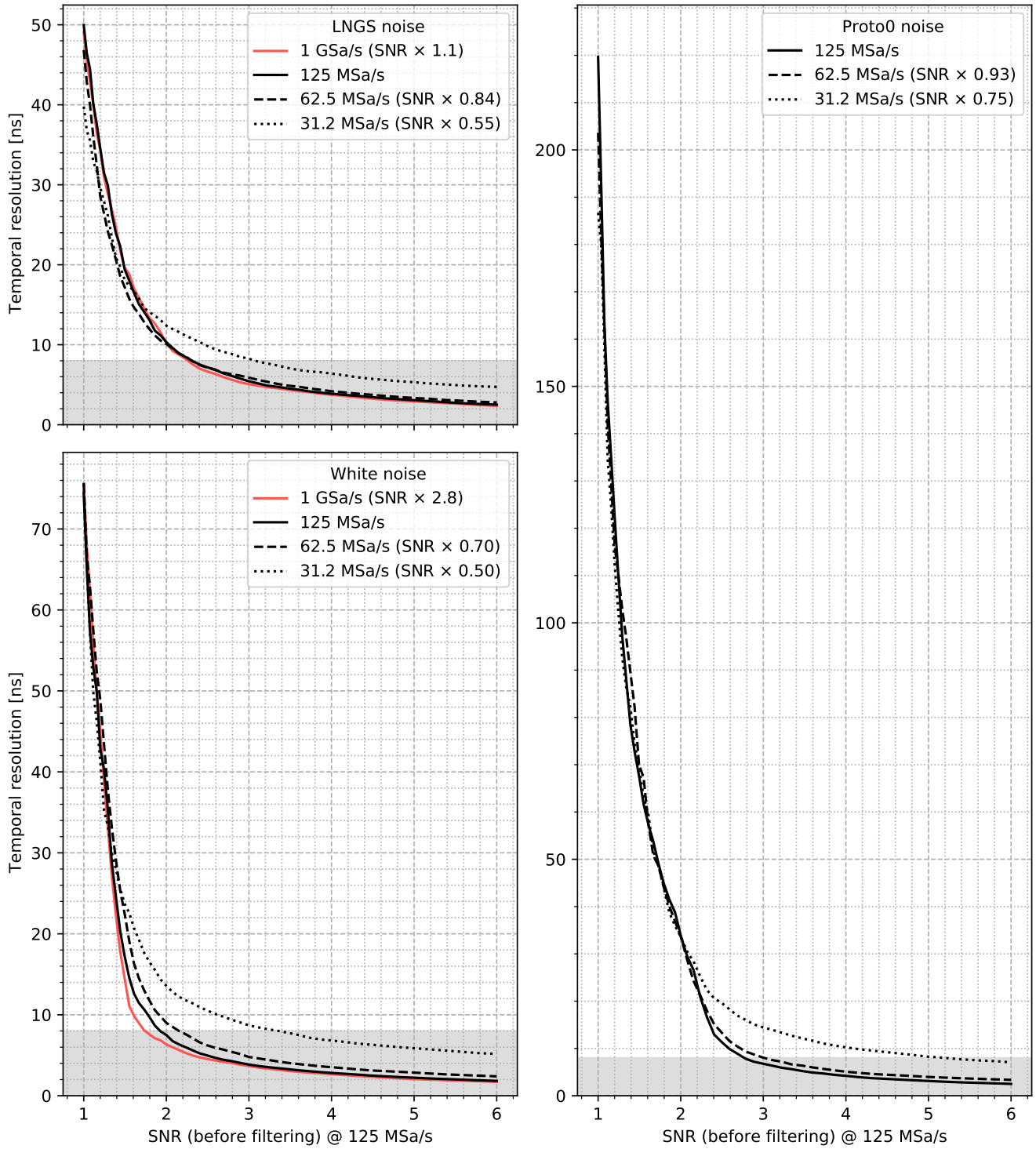


Figure 5.10: Pulse detection temporal resolution at different sampling frequencies with a cross correlation filter with template length 2048 ns. The SNR scale is at 125 MSa/s; curves for different sampling frequencies are rescaled horizontally by the factor written in the legend to account for the noise variance reduction with downsampling, as described in the text. (`figtempresdowns.py`)

Chapter 6

Fake rate of pulse detection

In chapter 4 we measured the signal to noise ratio after filtering. The point of using the SNR is expressing the signal height relative to the width of the noise distribution, because the threshold required to reject noise with a given probability is proportional to that scale. So it is convenient to express the threshold relative to the same scale. When setting a threshold on the filtered amplitude, as done in data acquisition to reduce the noise rate, an SNR well above such a rescaled threshold will ensure that a good efficiency for signal is retained.

This chapter deals with the calculation of the noise rate resulting from random fluctuations of the noise above the threshold.

Even assuming the noise to be Gaussian, the probability that a random fluctuation gets above the threshold is not given simply by computing the survival function (i.e., the integral to $+\infty$) of the Gaussian distribution at the threshold.

More precisely, the probability that any given sample is above the threshold is given by such integral. What we need, however, is the *rate* of threshold *crossings*. The noise is not white, but even if it was, after applying the filter, which combines linearly many input samples for each output sample, the waveform is autocorrelated at least up to the length of the filter. Intuitively, if a smooth function crosses a threshold, it takes some time to go down before it can cross the threshold again.

We will present the method used to compute the threshold crossing rate of the noise, or fake rate, for a running average filter, and check that the calculated rates are consistent with the results obtained in data.

The choice of the running average filter is due to the fact that this was the filter implemented as part of the first prototype firmware for the FPGA of the digitizer board. This constitutes the first processing stage in the DarkSide DAQ chain, and the reduction of the noise is a key goal of that processing. However, the method we present can be applied to any linear filter of choice, and allows to extrapolate efficiently to very low rates.

6.1 Model

We expect the noise to be Gaussian. Even if it were not prior to any processing, when filtering many samples are linearly combined, and the sum of random variables tends to have a Gaussian distribution independently of the initial one. So Gaussianity is a reasonable assumption in our study.

6.1.1 From the continuous case

We have a discrete sequence of samples. The continuous equivalent is a Gaussian process. We can expect the discrete case to be equivalent to the continuous case if the autocorrelation time is larger enough than the sampling step, which should hold from the consideration above.

Also, even though the values are initially discrete too, after filtering the possible non integer values between two consecutive integers are at least the length of the filter (think about an average). So we take the formula for the continuous case and adapt it.

The mean number of threshold upcrossings r in the interval $(0, 1)$ by a zero-mean stationary and appropriately smooth Gaussian process is given by [RK06, p. 81]

$$r = \sqrt{-\frac{k''(0)}{2\pi}}\,\mathrm{gauss}(u; 0, \sigma) = \tag{6.1}$$

$$= \frac{1}{2\pi}\frac{\sqrt{-k''(0)}}{\sigma}\exp\left(-\frac{1}{2}(u/\sigma)^2\right), \tag{6.2}$$

where u is the threshold, σ the noise standard deviation (the RMS), $\mathrm{gauss}(x; \mu, \sigma)$ a Gaussian probability density on x with mean μ and standard deviation σ, and k the autocovariance function, i.e., $k(x) = \mathrm{Cov}[f(t), f(t + x)]$ for any t (for example, $k(0) = \sigma^2$), where $f(t)$ is the continuous waveform.

We have to map the second derivative of the autocovariance function to a discrete equivalent. We first do a manipulation in the continuous realm. Since the covariance operator is an integral, it commutes with derivation:

$$k''(x) = \frac{\partial^2}{\partial x^2}\mathrm{Cov}[f(t), f(t + x)] = \tag{6.3}$$

$$= \mathrm{Cov}[f(t), f''(t + x)], \tag{6.4}$$

thus $k''(0) = \mathrm{Cov}[f(t), f''(t)]$. We estimate the second derivative with a finite difference:

$$f(t \pm \Delta t) = f(t) \pm f'(t)\Delta t + \frac{1}{2}f''(t)\Delta t^2 + O(\Delta t^3) \to \tag{6.5}$$

$$\to f''(t)\Delta t^2 = f(t + \Delta t) + f(t - \Delta t) - 2f(t) + O(\Delta t^3). \tag{6.6}$$

Choosing $\Delta t = 1/f_s$, where f_s is the sampling frequency, and calling $y_i = f(t_0 + i\Delta t)$ the samples, we have:

$$k''(0) \mapsto f_s^2 k_2, \tag{6.7}$$

$$k_2 \equiv \mathrm{Cov}[y_i, y_{i+1} + y_{i-1} - 2y_i], \tag{6.8}$$

$$r = f_s\frac{1}{2\pi}\frac{\sqrt{-k_2}}{\sigma}\exp\left(-\frac{1}{2}(u/\sigma)^2\right). \tag{6.9}$$

Discretizing directly $k''(0)$ yields the same result.

The covariance in Equation 6.8 can be estimated with the sample covariance on a filtered waveform array $\mathbf{y}$.

6.1.2 Direct discrete derivation

Since the formula we derived is approximate, as a cross check we derive another approximate one following a different path.

A threshold crossing happens when a sample is below the threshold and the next one is above: $y_i \le u$, $y_{i+1} > u$. Fix $i = 0$ and let $p(y_0, y_1)$ be

the joint distribution of the two samples. The probability of crossing at any given point then is

$$P = \int_{-\infty}^{u} \mathrm{d}y_0 \int_{u}^{\infty} \mathrm{d}y_1 \, p(y_0, y_1). \tag{6.10}$$

In general we can not obtain the crossing rate just by multiplying P by the sampling frequency because of correlations. However in practice we are interested in low crossing rates, less than 10 cps, to be compared to the filter length $2\,\mu\mathrm{s} = 1/(500\,\mathrm{kHz})$. If the typical time between crossings is much longer than the autocorrelation time, then we can ignore correlations. Thus the crossing rate is $r = f_s P$.

The integrand in Equation 6.10 is a bivariate Gaussian distribution, which explicitly is

$$p(y_0, y_1) = \frac{1}{2\pi\sqrt{\sigma^4 - c^2}} \exp\left(\frac{1}{2}\begin{pmatrix} y_0 & y_1 \end{pmatrix}\begin{pmatrix} \sigma^2 & c \\ c & \sigma^2 \end{pmatrix}^{-1}\begin{pmatrix} y_0 \\ y_1 \end{pmatrix}\right), \tag{6.11}$$

where $c = \mathrm{Cov}[y_0, y_1]$.

We do not know how to the integral analytically, so we break down the joint distribution as $p(y_0, y_1) = p(y_1|y_0)p(y_0)$ and discretize the integral over y_0:

$$p(y_0) = \mathrm{gauss}(y_0; 0, \sigma), \tag{6.12}$$

$$p(y_1|y_0) = \frac{p(y_0, y_1)}{p(y_0)} = \mathrm{gauss}\left(y_1; \frac{c}{\sigma^2} y_0, \sqrt{\sigma^2 - \frac{c^2}{\sigma^2}}\right), \tag{6.13}$$

$$P \approx \sum_{k=0}^{N-1} \Delta u \, p(y_0(k, u, \Delta u)) \int_{u}^{\infty} \mathrm{d}y_1 \, p(y_1|y_0(k, u, \Delta u)), \tag{6.14}$$

$$y_0(k, u, \Delta u) \equiv u - k\Delta u. \tag{6.15}$$

The integral on y_1 can be computed using the error function. Δu should be chosen small compared to σ, while N large relative to $\sigma/\Delta u$.

In Figure 6.1 we compare formula (6.9) (with $f_s = 1$) with P and with the Gaussian survival function. To make the comparison we have to use a k_2 that matches c:

$$\begin{aligned} k_2 &= \mathrm{Cov}[y_i, y_{i+1} + y_{i-1} - 2y_i] = \\ &= \mathrm{Cov}[y_i, y_{i+1}] + \mathrm{Cov}[y_i, y_{i-1}] - 2\,\mathrm{Cov}[y_i, y_i] = \\ &= 2(c - \sigma^2). \end{aligned} \tag{6.16}$$

We use $\sigma = 1$, $c = 99\,\%$, $\Delta u = 1/100$, $N = 500$ (we decreased Δu until convergence). We see that our derivation and the formula for Gaussian processes agree very well, while differing visibly from the survival function. We will henceforth use the continuous formula for its simplicity.

6.1.3 Dead time

One part of the data acquisition system (DAQ) for which the study of threshold crossings is particularly relevant are the digitizers. Due to limited logic resources in the installed FPGAs, a digitizer can not do complicated processing. Instead, a simple filter is applied, and the filtered waveform is compared to the threshold. Whenever the threshold is crossed, a fixed slice of waveform is sent to the front end processing (FEP) for further analysis (identify multiple signals, locate them precisely, use a better filter, etc.). A threshold crossing that happens too close in time to a previous crossing will be ignored.

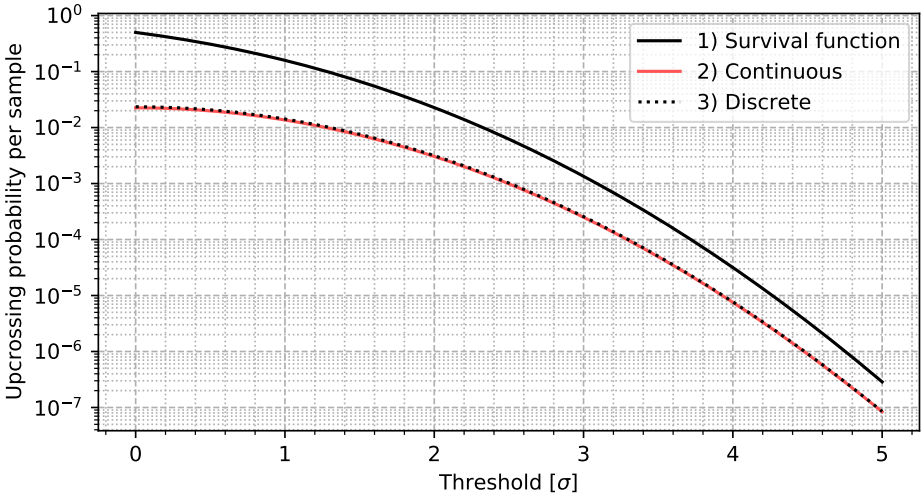


Figure 6.1: The threshold upcrossing rate expressed as per-sample crossing probability (i.e., the rate if the sampling frequency is 1) for an autocorrelated Gaussian waveform, estimated using three formulae: 1) the probability for a single sample to be higher than the threshold, 2) a formula for continuous processes (Equation 6.9), 3) an approximation of the discrete case (Equation 6.14). `(figcrossingprob.py)`

This means that we have a dead time T. We model it as a non-restartable dead time, i.e., a crossing that happens within T of a previous one is ignored only if the latter has not been ignored itself.

Assuming that the crossings are a Poisson process, the formula to correct a rate R for the effect of the dead time is [Kno10, p. 120]:

$$R \mapsto \frac{R}{1 + RT}. \tag{6.17}$$

We note that the crossings of a Gaussian process are not in general a Poisson process. We just need one counterexample to show this. Consider the process with autocovariance function $k(x) = \cos(x)$. This is positive definite because it is a linear combination of external products: $\cos(x - y) = \cos x \cos y + \sin x \sin y$. Since cos is orthogonal to sin, they are the eigenfunctions, so a realization of the process is a random linear combination of harmonic functions, which means that it is a shifted cosine, so the crossings are exactly periodic.

However, in practice we expect that there will just be a "repulsion" or "attraction" of crossings within the scale of the autocorrelation time, so for low crossing rate the formula should work.

6.2 Application to real electrical noise data

We want to test formula (6.9) on actual electrical noise.

6.2.1 Data

We will use the Proto0 run 886, collected when operating the SiPMs below their breakdown voltage, see section 3.1. Tiles 53, 57 and 59 (used in Proto0) will be also studied with LNGS data, while for Tile 15 just LNGS data are available. The LNGS data files are:

`FBK/NUV/MB2-LF-3x/NUV-LF_3x_53/nuvhd_lf_3x_tile53_77K_64V_6VoV_1.wav`
`FBK/NUV/MB2-LF-3x/NUV-LF_3x_53/nuvhd_lf_3x_tile53_77K_66V_7VoV_1.wav`

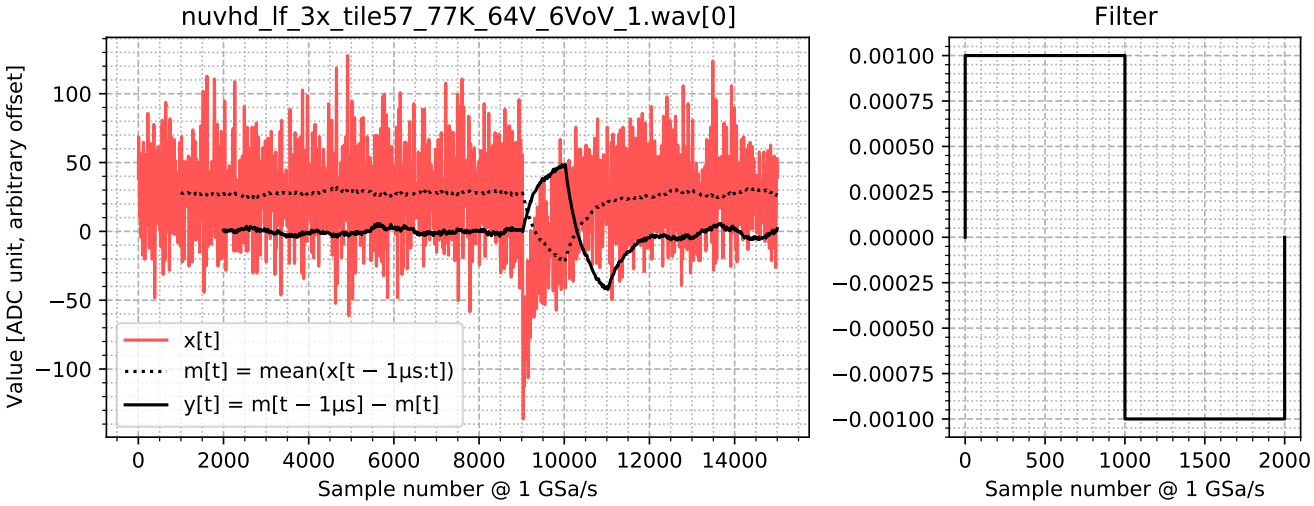


Figure 6.2: A filtered waveform. The computation is split into a moving average m (dotted line) and its finite difference y (solid black line). The right plot shows the overall filter shape. (We use an event with a signal just to see how the filter behaves, the analysis is done on noise only.) (`figsqfilt.py`)

```
FBK/NUV/MB2-LF-3x/NUV-LF_3x_57/nuvhd_lf_3x_tile57_77K_64V_6VoV_1.wav
FBK/NUV/MB2-LF-3x/NUV-LF_3x_59/nuvhd_lf_3x_tile59_77K_64V_6VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_55V_0VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_59V_2VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_63V_4VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_67V_6VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_71V_8VoV_1.wav
LFOUNDRY/pre-production-test/TILE_15/LF_TILE15_77K_73V_9VoV_1.wav
```

The Proto0 data consists purely of noise, so no preprocessing is required. For the LNGS data we take the pre-trigger part of the events, and ignore events where any pre-trigger sample is less than 750 (860 for Tile 15).

We plot the time-value histogram of the data for Tile 53 in Proto0 and LNGS (Figure 6.5), for Tile 15 at maximum overvoltage, and for Tiles 57 and 59 (Figure 6.6).

6.2.2 Filter

We filter using a 1 µs moving average with 1 µs of baseline and 1 µs of dead time, without delay between the baseline and the signal averages. In Figure 6.2 we show an example filtered waveform and the filter shape.

Although we know from chapter 4 and 5 that this is not the optimal filter by SNR neither by temporal resolution, this is a simple filter that would be suitable for implementation in the digitizer's FPGA.

Since on LNGS data the events are short compared to the filter length (9 µs vs. 2 µs), we need to take into account boundary effects. The output length of the filtered waveform is (initial length) − 2 µs + 1 sample; to compute the rate we have to divide by this quantity instead of the initial waveform length.

The dead time does not play nicely with borders, because a hypothetical unseen crossing within 1 µs before the event start could kill a crossing in the event. Moreover, at low thresholds, the first crossing will happen almost immediately, and again immediately after the dead time ends, thus the number of crossings per event is quantized.

A complete solution to these problems would be to avoid counting the crossings which happen within 1 µs after the event start (although they are detected and do project a dead time on the following ones), and in the remaining region further select a subregion which is 1 µs shorter but has a uniformly random starting position.

However we are mostly interested in the low-rate regime, in which the dead time boundary effects are negligible, so we keep the whole filtered region.

6.2.3 Algorithm

The simplest way to count the threshold crossings as a function of the threshold is to repeat the calculation varying the threshold. The computational complexity is $O(nN)$ where n is the number of thresholds and N the number of samples.

To produce a smooth curve (large n) we use instead a reverse histogram. We choose an evenly spaced range of thresholds. For each pair of consecutive samples, if the second sample is higher than the first, we determine the subrange of thresholds that falls between the samples, and increment their counts. To apply the dead time, we keep a per-threshold last occurence time.

Since the range of thresholds is evenly spaced, the subrange can be found arithmetically, so the complexity is just $O(N)$, i.e., it does not depend on the number of thresholds.

6.2.4 Results

In Figure 6.3 we compare the measured threshold crossings with the continuous-derived formula (6.9) for a single Tile over the entire threshold range. The coefficients σ and k_2 for the formula are computed on each filtered event and then averaged. Finally, the dead time is accounted for with (6.17). For the data, the conversion from count to rate is done dividing by the length of the filtered waveform (so 2 µs less than the initial length per event).

A discrete agreement can be observed at low and high rates, less so in the intermediate region. The agreement at high rate is probably improved by saturation due to dead time, since the maximum rate allowed by dead time is $1/(1\,\mu\text{s}) = 1\,\text{Mcps}$. Although it cannot be fully appreciated from Figure 6.3 due to the logarithmic scale, even where the theory and data lines are closer they still differ by a factor 1.3, while the Poisson error is approx. 3 % since the count is 1000.

In Figure 6.4 we show the same comparison together for all datasets, divided in three groups (Tile 15 LNGS data, LNGS other Tiles, Proto0 data). The parameters for the formula, and the rates at threshold $4\,\sigma$, are listed in Table 6.1.

For comparison, in Figure 6.4 we also show the rate predicted by [Sav18, p. 98], which provides the following formula:

$$r = \frac{R_0}{2} \exp\left(-\frac{1}{2}(u/\sigma)^2\right), \tag{6.18}$$

$$R_0 = \frac{2}{\sqrt{3}} f_u, \quad f_u = 40\,\text{MHz}. \tag{6.19}$$

We see that it overestimates the true rate about by a factor of 5-10. Note, however, that the formula was derived without taking into account filtering, so the overestimation is expected.

In the second group above some threshold the measured rate stops decreasing and remains constant at approximately 10 counts. This is probably

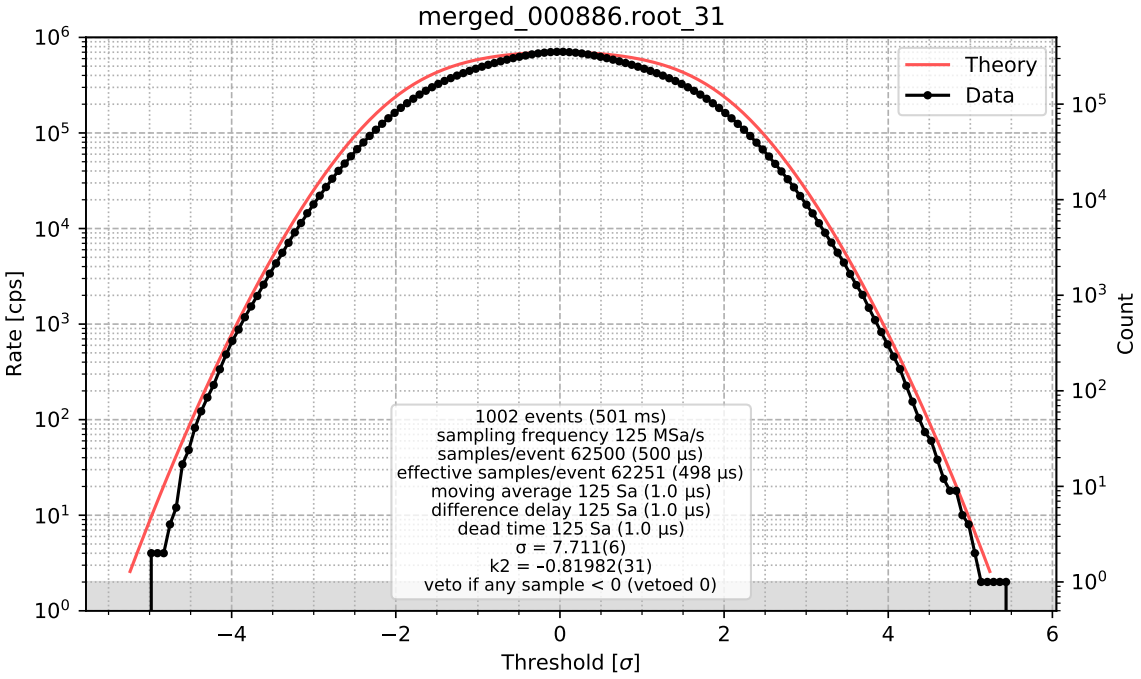


Figure 6.3: Measured and predicted (with Equation 6.9) fake rate for Tile 31 in Proto0 with a noise-only acquisition. The right scale shows the actual count of threshold crossings for the data, with a gray band marking one crossing. (`figfakerate1.py`)

due to real pulses which our very simple preprocessing can not filter away, most likely 1 PE dark noise or random pulses from light leaks.

In Figure 6.4 we highlight the measured rates for Tile 53 because they are evident outliers. The rate remains higher than the theory predicts and than the other Tiles as the threshold increases. This is visible in Proto0, and in LNGS at 6 VoV, but not at 7 VoV. Analogously, from Table 6.1 we see that the quantity $f_s\sqrt{-k_2}/(2\pi\sigma)$ that multiplies the exponential in (6.9) is different from the others for Tile 53, in the same cases as the data, but with the opposite trend. This variation seems to depend only on an increase in σ and not on k_2.

Comparing the 2D histograms for Tile 53 (Figure 6.5) to the others (Figure 6.6) there is no apparent difference. Three possible explanations come to mind: 1) a violation of Gaussianity, 2) stray pulses, 3) low frequency electrical noise.

Regarding Gaussianity, we checked the distribution of the samples before filtering and it agrees very well with a Gaussian. If they are stray pulses, they do not have the same height, otherwise they would show up as a flat rate curve. It could be oscillating noise with a frequency of approx. $1/(2\,\mu s) = 500\,kHz$, since our filter would be very good at picking that up, or sudden variations of the baseline.

We note that this behavior arises for the same Tile in different setups, showing up both as an increased noise RMS and an higher threshold crossing rate. We also note that the Proto0 data were collected after the LNGS data. As for the absence of this behavior in the LNGS data collected at 7 VoV, we cannot formulate hypotheses as we do not know in what order the LNGS data at 6 VoV and 7 VoV were collected, i.e., whether the tile was damaged during the testing procedure.

We did not investigate further the discrepancy for Tile 53, since for all the other Tiles the agreement between our model and data is consistent and satisfactory.

We now want to estimate the minimum amount of data required for the procedure. From Table 6.1 we see that the relative error on k_2 times the square root of the time, $C \equiv \mathrm{Std}[\bar{k_2}]\sqrt{T}/|k_2|$, is approximately $8\,\mathrm{ns}^{1/2}$ in all cases. The error should be proportional to $T^{-1/2}$, so to have an 1 % error we need $T_{1\,\%} = (100C)^2 = 0.7\,\mathrm{ms}$.

We conclude this chapter by summarizing all the steps we took to compute the fake rate:

1. Acquire at least 1 ms of noise data.
2. Filter the data (including baseline subtraction) producing a filtered waveform $\mathbf{y}$.
3. Compute the standard deviation σ and $k_2 = \mathrm{Cov}[y_i, y_{i+1} + y_{i-1} - 2y_i]$.
4. Compute the fake rate for threshold u using $r = f_s\sqrt{-k_2/(2\pi)}\,\mathrm{gauss}(u; 0, \sigma)$ where f_s is the sampling frequency.

Alternatively, if one has a noise spectrum available but not the noise waveform, it is possible to obtain the autocovariance by computing the discrete Fourier transform of the power spectrum [Fer15, p. 84] and then normalizing it to be σ^2 in 0. Then k_2 is obtained by the covariance at lag 1 c with (6.16). If the spectrum was obtained from the discrete Fourier transform of a noise waveform, c is the first coefficient after the central one in the autocovariance.

The model was satisfactory in 24 of the 25 Tiles we considered. The uncertainty in the low rate regime is $\pm 50\,\%$ and the result is an overestimate with probability 90 %. These statements are educated guesses based on Table 6.1.

Data								Rate @ $4\,\sigma$	
Setup	Tile	Overvoltage [V]	T [ms]	σ [u]	k_2 [u^2]	Std$[\bar{k_2}]\sqrt{T}$ [u^2 ns$^{1/2}$]	$f_s\sqrt{-k_2}/(2\pi\sigma)$ [Mcps]	Theory [kcps]	Data [kcps]
LNGS	15	0	138	1.6	−0.0017	0.012	4.2	1.4	1.1
LNGS	15	2	138	1.5	−0.0017	0.012	4.3	1.4	1.2
LNGS	15	4	138	1.5	−0.0017	0.012	4.3	1.4	1.3
LNGS	15	6	138	1.5	−0.0017	0.012	4.4	1.5	1.1
LNGS	15	8	138	1.5	−0.0017	0.012	4.5	1.5	1.4
LNGS	15	9	138	1.4	−0.0017	0.012	4.6	1.5	1.3
LNGS	53	6	69	3.8	−0.0044	0.030	2.7	0.92	2.6
LNGS	53	7	69	2.1	−0.0043	0.029	5.0	1.7	1.0
LNGS	57	6	68	2.2	−0.0043	0.031	4.8	1.6	1.2
LNGS	59	6	69	2.2	−0.0040	0.028	4.6	1.5	1.0
Proto0	29	<0	499	8.3	−0.86	10.0	2.2	0.74	0.77
Proto0	30	<0	499	6.9	−0.75	6.0	2.5	0.84	0.67
Proto0	31	<0	499	7.7	−0.82	7.0	2.3	0.78	0.60
Proto0	32	<0	499	7.0	−0.76	6.6	2.5	0.83	0.67
Proto0	34	<0	499	6.5	−0.78	6.2	2.7	0.90	0.62
Proto0	36	<0	499	7.7	−0.84	7.1	2.4	0.79	0.63
Proto0	37	<0	499	6.5	−0.69	5.4	2.5	0.85	0.67
Proto0	38	<0	499	7.7	−0.84	7.1	2.4	0.80	0.59
Proto0	39	<0	499	7.6	−0.91	7.5	2.5	0.84	0.68
Proto0	41	<0	499	7.5	−0.87	7.1	2.5	0.83	0.71
Proto0	42	<0	499	7.0	−0.82	6.8	2.6	0.87	0.63
Proto0	52	<0	499	7.8	−0.82	6.5	2.3	0.77	0.61
Proto0	53	<0	499	10.1	−0.90	8.0	1.9	0.63	2.3
Proto0	54	<0	499	8.0	−0.89	7.4	2.4	0.79	0.59
Proto0	55	<0	499	8.0	−0.85	7.1	2.3	0.77	0.63
Proto0	57	<0	499	7.8	−0.88	7.4	2.4	0.80	0.69
Proto0	58	<0	499	7.6	−0.82	6.4	2.4	0.79	0.65
Proto0	59	<0	499	7.6	−0.79	6.6	2.3	0.77	0.55
Proto0	60	<0	499	7.5	−0.77	6.3	2.3	0.78	0.61
Proto0	61	<0	499	8.0	−0.89	7.4	2.4	0.79	0.60
Proto0	62	<0	499	7.6	−0.80	6.6	2.3	0.78	0.62
Proto0	63	<0	499	8.1	−0.86	7.1	2.3	0.76	0.61
Proto0	64	<0	499	7.9	−0.81	6.9	2.3	0.76	0.66
Proto0	65	<0	499	7.5	−0.80	6.4	2.4	0.80	0.62
Proto0	66	<0	499	8.2	−0.91	7.5	2.3	0.77	0.61

Table 6.1: The coefficients measured on the filtered waveforms needed to evaluate the formula for the threshold upcrossing rate. T is the total duration, σ the standard deviation, k_2 the covariance of the waveform with its second derivative (Equation 6.8), Std$[\bar{k_2}]$ the uncertainty on the value of k_2 determined as the standard deviation of the sample mean of k_2 values across events. The unit "u" is the ADC digit. (`figfakerate.py`)

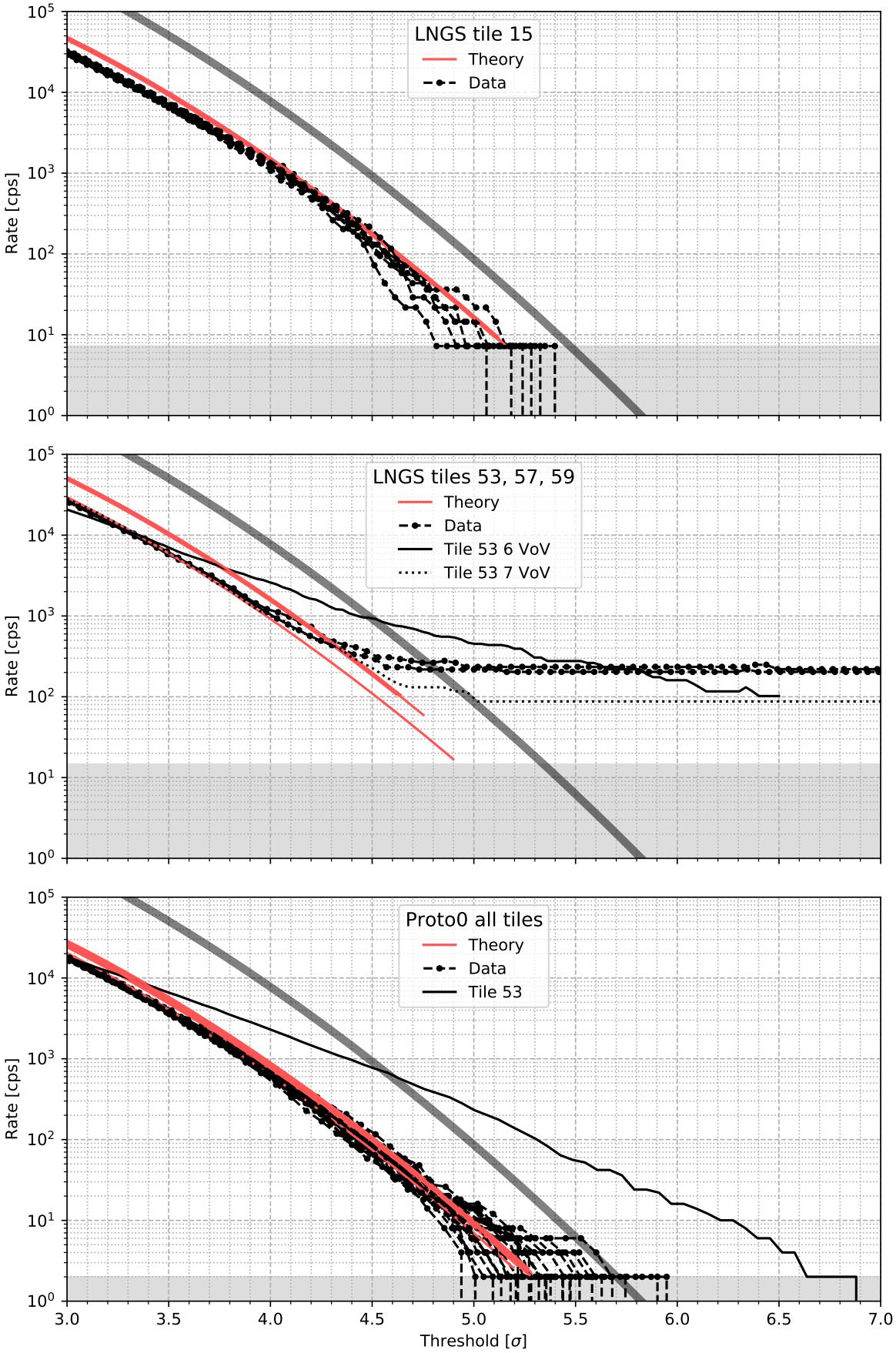


Figure 6.4: The threshold upcrossing rate both counting directly the crossings on data and computing it with formula (6.9). The gray band marks the rate corresponding to a single crossing counted in the data. The thick gray curve is the rate predicted by [Sav18, p. 98]. (`figfakerate.py`)

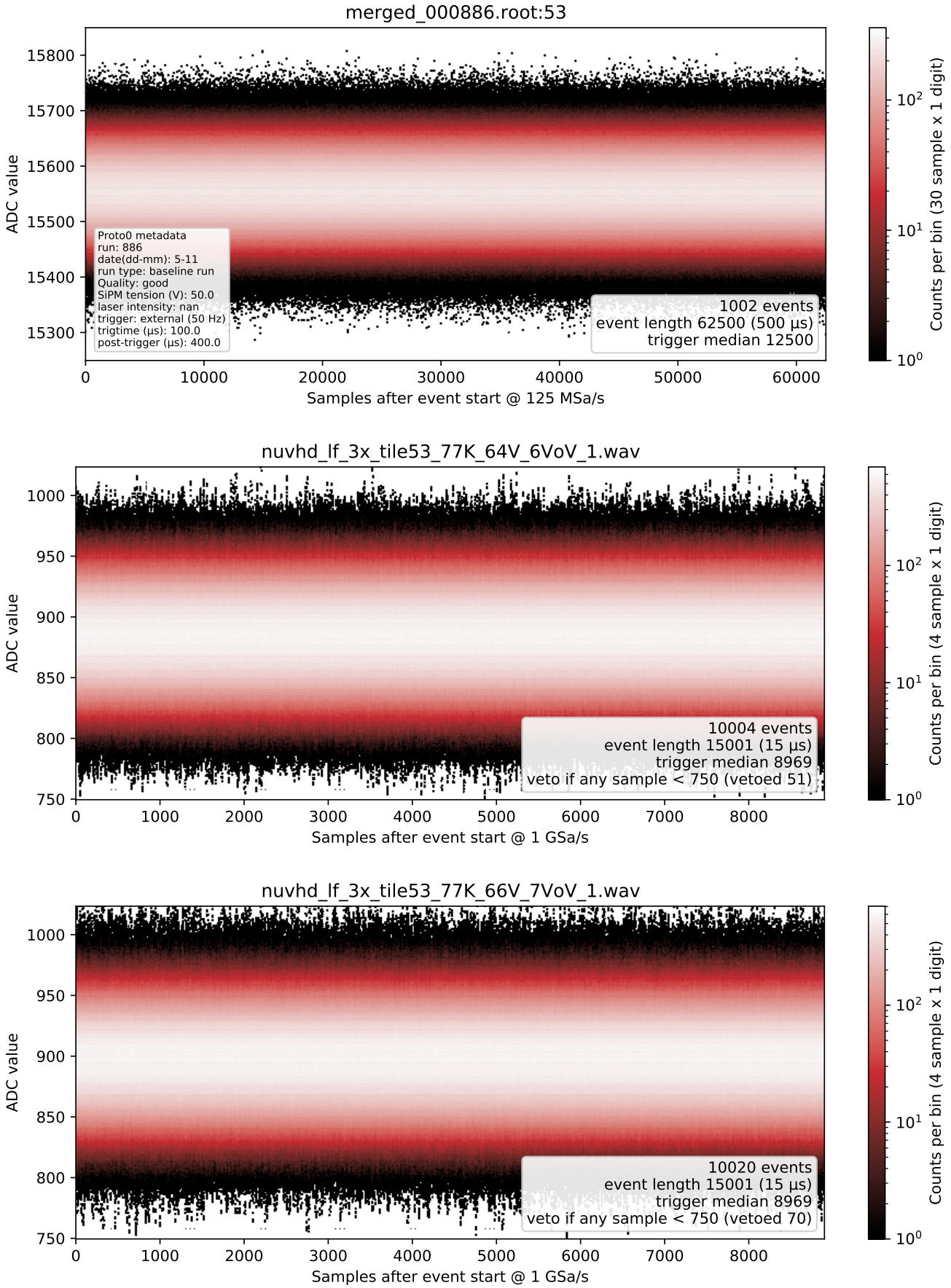


Figure 6.5: Time-value histograms of Tile 53 noise in Proto0 with a baseline acquisition, and in LNGS pre-trigger at overvoltage 6 V and 7 V. (`fighist2dtile53.py`)

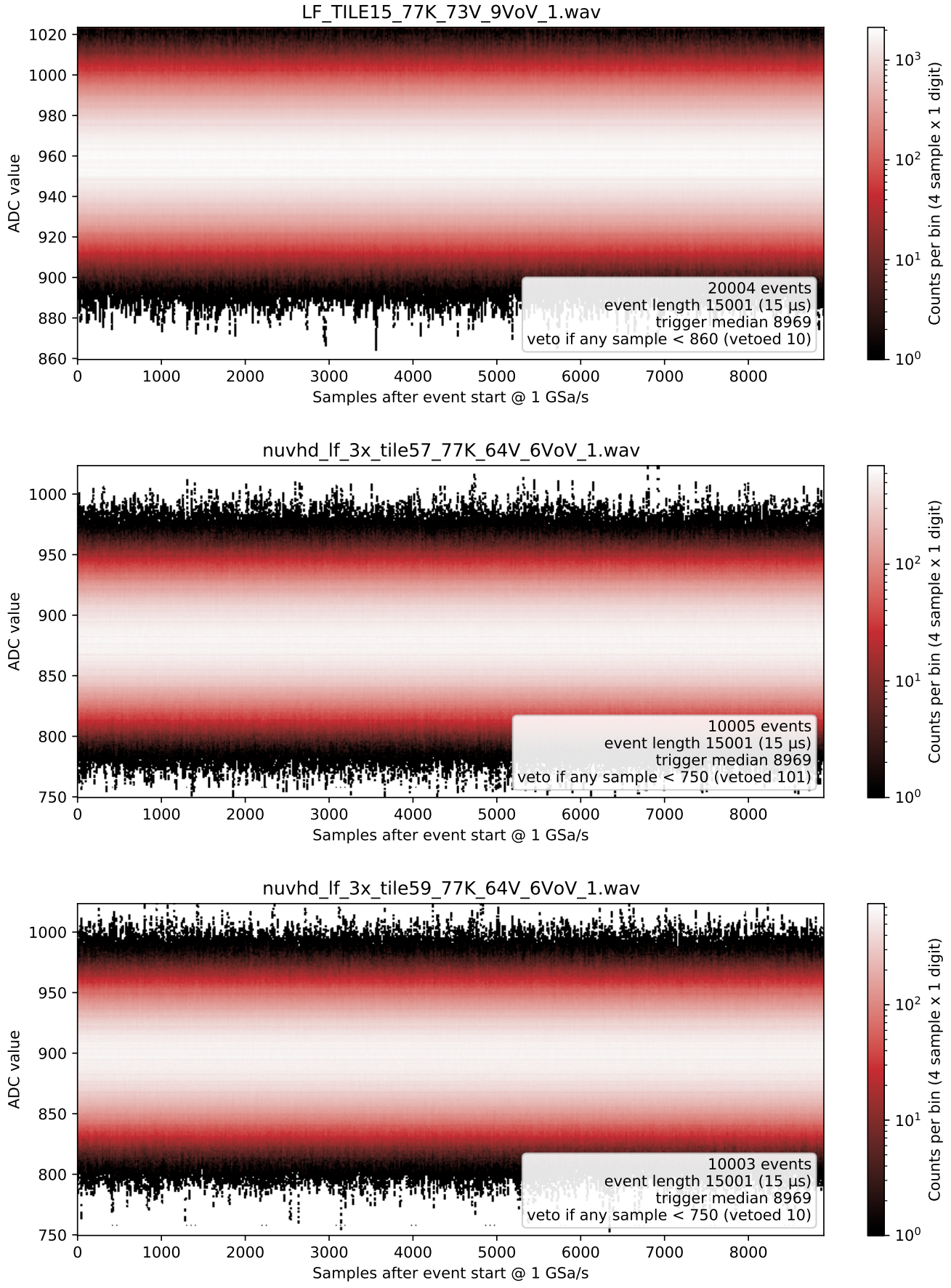


Figure 6.6: Time-value histograms of Tiles 15, 57 and 59 noise in LNGS data. (`fighist2dtile155759.py`)

Chapter 7

Study of dark count rate and correlated noise in a DarkSide SiPM

In the previous chapters we studied the impact of electrical noise in the procedure to identify and measure the signal produced by an incident photon. However the photodetector itself produces pulses which are not directly caused by an external photon. Such pulses can be produced independently of incident light, as in the case of thermally-generated avalanches, or result from secondary-avalanche generation mechanisms triggered by a primary avalanche. In the latter case we refer to them as correlated noise.

We will give a brief explanation and classification of the SiPM noise and then characterize it on a specific Tile from the LFoundry production, using laser data collected with the LNGS test stand. In doing this, we rely on several of the tools introduced in the previous chapters.

There are two goals in characterizing the SiPM noise. First, a quantitative model of the noise is needed for the DarkSide20k simulation, and we will provide an estimate of the parameters that can be used to simulate the behavior of the most recent version of the DarkSide TPC photosensors. Second, using the fast frontend electronics of the TPC PDMs we can classify noise pulses by looking at their amplitude and time distribution. This provides a validation sample against which one can test more indirect noise characterization methods, necessary, e.g., when relying on slower front-end electronics. This is a topic of interest for monitoring applications, e.g., for the sensors installed in the VETO, which feature the same Tile as the TPC ones, but slower electronics.

7.1 Theory

In chapter 2 and 4 we briefly introduced the silicon photomultiplier (SiPM). We now recap and expand the explanation, mainly following [Sav18, ch. 3]. For a general introduction to semiconductor detectors, but not the SiPM, see [Kno10, ch. 11].

The difference between a SiPM and older kinds of semiconductor detectors is that the SiPM has a binary response: the amplitude of the output is not proportional to the energy released by the detected particle. In this sense they are analogous to Photomultiplier Tubes (PMTs), because they are designed to detect just the presence of a photon with high efficiency ($\sim 50\,\%$).

A SiPM is composed by a grid of *microcells*. The microcells are Single

Photon Avalanche Photodiodes (SPADs). A SPAD is a photodiode operated in reverse bias above its breakdown voltage in series with a *quenching resistor*. Since the bias is above breakdown, if a current starts in the photodiode it will be self sustaining and would normally destroy the diode. The resistor in series lowers the potential difference on the diode when a current flows through it, stopping the current because the diode goes below breakdown, so the output pulse will have duration and amplitude uniquely determined by the resistor and the capacitance of the diode junction, independently of the initial release of energy that triggered the current.

The difference between the bias and the breakdown voltage is called *overvoltage*, and is often indicated with the unit "VoV" meaning "Volt overvoltage".

The output from all the microcells is summed analogically, so if multiple microcells are triggered simultaneously, the amplitude of the output pulse is discretized and proportional to the number of fired microcells. As we already said in chapter 4, we will use the term "PE", standing for "photoelectrons", as a unit when indicating the number of microcells corresponding to a pulse.

7.1.1 SiPM noise

The initial creation of an electron-hole pair that starts the avalanche in the photodiode can either be caused by an absorbed photon or by a thermal fluctuation. The latter case happens randomly with probability that depends on temperature, and the resulting rate of random pulses is called *dark count rate*, where "dark" stands for the fact that these pulses occur even when the photodiode is kept isolated from light.

A "primary" pulse, either thermally- or photon-generated, can induce other pulses, that can themselves recursively produce additional pulses in the same way. The main mechanisms for the proliferation of pulses are listed here and illustrated in Figure 7.1:

Afterpulse (AP) During the avalanche, charge carriers can remain trapped into impurities and imperfections of the crystal. They are released afterwards at random times, starting another avalanche in the same microcell.

Direct cross talk (DiCT) A photon emitted by an avalanche can trigger a nearby microcell.

Delayed cross talk (DeCT) Instead of hitting directly another microcell, the photon can be absorbed in the shared crystal substrate and generate a hole that travels up until it passes through a microcell and triggers an avalanche there.

Delayed afterpulse In the latter case, if the hole hits the originating microcell, we call it delayed afterpulse instead of DeCT.

To remove ambiguity, the DCR as we intend here may be called "primary DCR", as in [Ace+17, fig. 3] for example, to be distinguished from the total rate with includes the correlated noise produced by the primary DCR.

The microcells have a pitch of 35 µm, so the DiCT pulse starts within 1 ps of the originating pulse, while as we saw in chapter 4 even just the peak of the pulse lasts ~ 10 ns. This means that the effect of the DiCT is multiplying the height of the pulse by an integer factor, because the overlapping pulses are well aligned. Afterpulses and DeCT, instead, can arrive with a significative delay, and thus can be distinguished from the originating pulse.

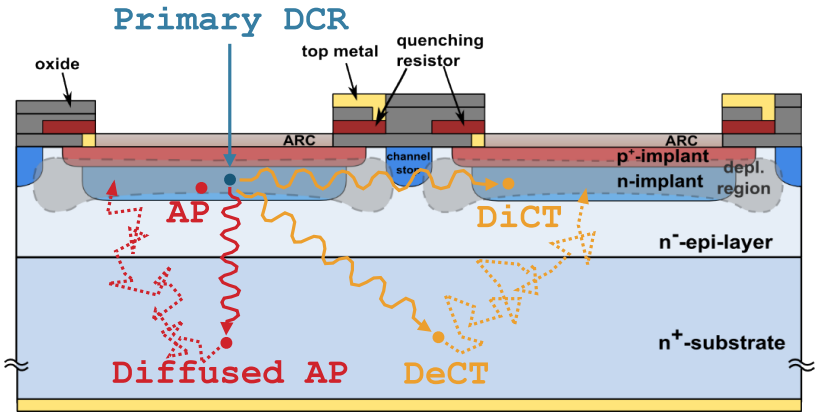


Figure 7.1: Schematic of SiPM noises overlaid on the cross section of the device: AP (afterpulse), DiCT (direct cross talk), DeCT (delayed cross talk). From [Sav18, p. 53].

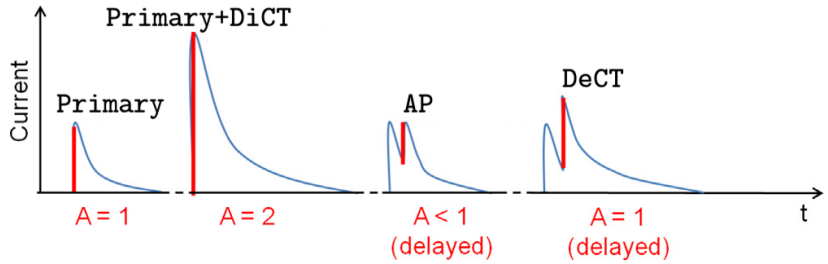


Figure 7.2: Schematic of noise pulses. From [Nag+14, p. 4].

The reason why we classify separately the delayed pulses, instead of having an overarching category of "delayed noise", is that afterpulses have a different amplitude. The shape of the pulse is a sharp peak followed by an exponentially decaying tail. This tail is due to the capacitance of the reverse-biased junction that recharges through the quenching resistor after being discharged inside the diode by the avalanche. A pulse generated in the same microcell before its complete recharge can only use up the charge present in that moment. Delayed cross talk, instead, has full amplitude because it involves a different microcell. See Figure 7.2.

A category of correlated noise we did not mention, and will not study, is external cross talk (eCT). It is like DiCT, but mediated by an external object. For example, when a SiPM is attached to a scintillator, the correlated noise increases due to photons emitted by the avalanches that are reflected back by the scintillator [Gol+14, p. 8]. We make the assumption that our data is free from eCT. An anecdotal estimate of the expected eCT probability in DarkSide20k at 9 VoV is 10 %.

From [Sav18, tab. 3.1 p. 62] we report typical values for the DCR and correlated noise probabilities in FBK devices similar to the LFoundry device we characterize here. The dark count rate, converted from rate per area to rate per PDM (area 25 cm^2), is 25 cps. The correlated noise probabilities are: DiCT 17 %, AP 17 %, DeCT 1.5 %. These are the probabilities for a 1 PE pulse to generate at least one pulse of the specified category.

Among correlated noise mechanisms, in the following we focus on DiCT and AP. For the DeCT we will just give a very rough estimate, since we are not able to select cleanly this class of pulses. From [Sav18, fig. 3.8 p. 54], in fact, we can see that DeCT occurs within 50 ns and our analysis is not sensitive to secondary pulses with delays below 100 ns. Also, since the probability is smaller than the other noise modes, we completely neglect second order effects due to DeCT.

7.1.2 DiCT model

Like primary pulses, noise pulses can themselves produce other noise pulses. We can imagine quite complicated interactions, for example: a DiCT induces a DeCT on the original cell, thus producing a "secondary" delayed afterpulse. Or a DiCT induces an afterpulse that induces a DiCT on the originating cell, again with an afterpulse as final outcome. If the probabilities involved are small enough, we should get by with the following simplifying assumption: that for each pulse, the DiCT involves new cells that were not previously involved in the chain, and the distribution of the number of DiCT at each step is fixed.

For the generation model of DiCT, we follow [Vin12]. Even when multiple DiCT steps are chained, the combined delay is very small compared to the duration of the pulse, so the overall effect of the complete DiCT "tree" is still to multiply the amplitude of the first pulse. This means that at the end we just need to know the distribution of the total number of consecutive DiCT.

For the single DiCT step, we consider two distributions: Bernoulli and Poisson. In the first case we assume that a cell can induce a DiCT in at most just another cell. In the second, we assume that there is a infinite population of other cells, each with a fixed probability to be triggered. Clearly these two cases are "opposite" approximations. In the first case, the distribution of the total number of PE k is Geometric. If p is the probability of generating a DiCT at each step, the distribution is

$$P_G(k;p) = p^{k-1}(1-p), \quad k \geq 1, \quad p \in [0,1). \tag{7.1}$$

Note that the conventional parametrization of the Geometric distribution is different, with p and $1-p$ interchanged. We use this formulation because it is more intuitively comparable with the other case. If the branching distribution is Poisson with mean μ_B, the total distribution is the Borel,

$$P_B(k;\mu_B) = e^{-k\mu_B}\frac{(k\mu_B)^{k-1}}{k!}, \quad k \geq 1, \quad \mu_B \in [0,1). \tag{7.2}$$

If it were $\mu_B \geq 1$, k would diverge with nonzero probability.

The formula for the mean is formally the same for the two distributions,

$$E[k] = \frac{1}{1-p} = \frac{1}{1-\mu_B}, \tag{7.3}$$

thus, since the interesting quantity is the amount of excess pulses due to noise, it makes sense to compare the two distributions when $p = \mu_B$. We make such comparison in Figure 7.3 for $p = \mu_B = 0.5$. We note that the Borel distribution has higher probability in the no-DiCT case, and at the same time a fatter tail, while being lower in the few-DiCT cases.

Since we use laser data, there can be multiple photons hitting the SiPM. The light is diffused before reaching the photodetector (see chapter 3), so we expect at most one photon per cell, and thus a Poisson distribution for the initial number of fired cells. So for the analysis we will need the distribution of the total number of PE $n \geq 0$ for an initial Poisson-distributed number of cells, each with its DiCT tree.

For the geometric model, the resulting distribution is called geometric Poisson or Pólya-Aeppli. Let μ_P be the initial Poisson mean. The probability

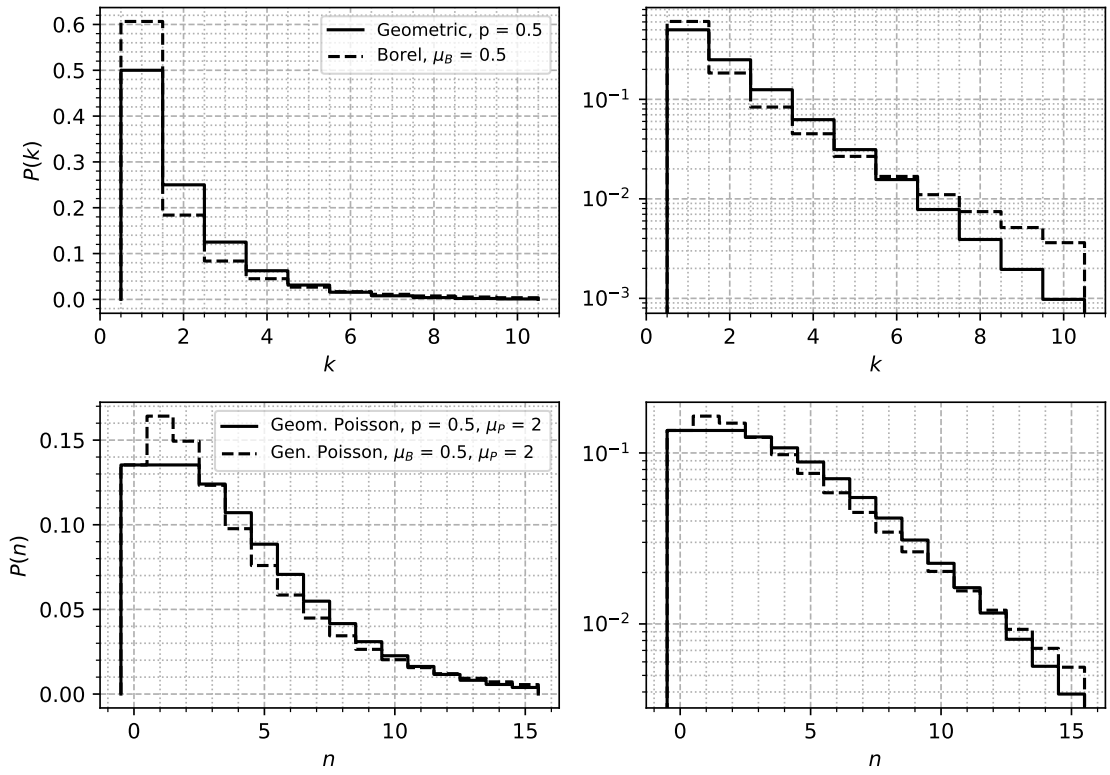


Figure 7.3: Top panels: comparison of the geometric and Borel distributions (Equations 7.1 and 7.2), used for the total number of pulses in the DiCT tree, including the initial pulse. Bottom panels: geometric and generalized Poisson (Equations 7.8 and 7.9), for the total number of PE when the number of initial pulses is Poisson distributed. The right panels are the same plots on the left in logarithmic scale. (`figgeomborel.py`)

mass function P_{GP} can be computed with this recursion [Nue08, p. 5]:

$$P_n \equiv P_{GP}(n; \mu_P, p), \tag{7.4}$$

$$z \equiv \mu_P \frac{1-p}{p}, \tag{7.5}$$

$$P_0 = e^{-\mu_P}, \tag{7.6}$$

$$P_1 = e^{-\mu_P} zp, \tag{7.7}$$

$$P_n = \frac{2n-2+z}{n} pP_{n-1} + \frac{2-n}{n} p^2 P_{n-2}. \tag{7.8}$$

For the Borel model, the distribution is the generalized Poisson:

$$P_{BP}(n; \mu_P, \mu_B) = e^{-(\mu_P + n\mu_B)} \frac{\mu_P(\mu_P + n\mu_B)^{n-1}}{n!}. \tag{7.9}$$

The means of the two distributions are, unsurprisingly, $\mu_P/(1-p)$ and $\mu_P/(1-\mu_B)$. Note that for $n = 0$ the probabilities are equal, since of course the cross talk does not affect the case with zero initial pulses.

The referenced article finds a much better agreement with the Borel model when looking at the tails of the PE distribution for DiCT only [Vin12, p. 3 fig. 1], while it does not find a visible difference for the Poisson+DiCT distribution with PE ≤ 11 [Vin12, p. 4 fig. 2].

7.1.3 Afterpulse model

For the afterpulses we have to model not just the number of pulses, but also their temporal distribution and amplitude.

In a generally accepted explanation, afterpulses are produced by carriers trapped during the avalanche that get released afterwards [Nag+14], [CLR91]. The release of trapped carriers is a thermodynamical process regulated by the potential difference that the carriers must overcome to jump out of their traps. This means that it also depends on the external bias. If there were only one kind of trap, the carrier release temporal distribution would be exponential. However we expect to have various possible trapping energy levels, so the distribution is in general a mixture of exponentials.

The released carriers also have a non-unitary probability of generating an avalanche. This probability also depends on the electric field and thus on the bias, like the release probability. The bias in turn depends on the recharge state of the cell, thus deforming the exponential distribution for low delays. While [Nag+14, p. 2] tries to keep into account this deviation by multiplying the afterpulse temporal distribution with the fraction of recovered charge

$$1 - \exp\left(-\frac{\Delta t}{\tau_{\text{rec}}}\right), \tag{7.10}$$

where τ_{rec} is the exponential scale of the pulse shape tail, and Δt is the delay from the primary pulse, other authors [Gar+14, p. 4] parametrize the time distribution with pure exponentials. However, we notice from [Nag+14, p. 5 fig. 11], reported here in Figure 7.4, that they do not actually have data at low delays to test the accuracy of their model, since the relatively high threshold they use for peak finding truncates the afterpulse distribution at $\approx 80\,\text{ns}$. Even though the recharge time is $\tau_{\text{rec}} = 207\,\text{ns}$, thus making the correction term (7.10) significative even above the temporal cut, by looking at the plotted histogram we think that a vanilla exponential with a different amplitude and scale could fit as well.

[Gar+14] has approximately the same temporal cut and parameters, and gets a good fit with either one or two uncorrected exponentials. We anticipate

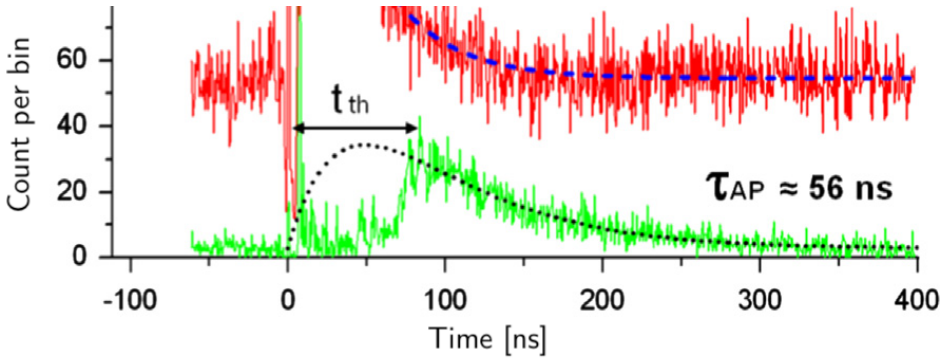


Figure 7.4: Excerpt of Figure 11 from [Nag+14], showing the histogram of the delay of pulses successive to initial 1 PE pulses, selected with an upper bound on their amplitude to get afterpulses while removing dark count, and fitted with an exponential decay with constant τ_{AP} multiplied by the correction in Equation 7.10 with $\tau_{\mathsf{rec}} = 207\,\mathrm{ns}$.

that in our analysis the temporal cut, relative to τ_{rec}, will be even more extended than in the referenced articles, so we will not be able to discriminate the two models. For simplicity, we will keep the plain exponentials.

Another aspect to consider at small delays is the amplitude reduction. Here all the references we consulted agree on multiplying the 1 PE amplitude by the recharge factor (7.10), thus assuming the amplitude to be proportional to overvoltage. We will discuss this in relation with the results of our data analysis.

Finally, the afterpulse avalanche itself produces trapped carriers, resulting in multiple afterpulses. However, since it may have smaller amplitude than a primary avalanche, we expect it to produce fewer carriers than a complete discharge. Other two possible effects that we hypothesize, although we are not sure about them, are: that if traps are not too far from saturation after an avalanche, the additional afterpulses will be even more suppressed, and that an avalanche may partially "purge" traps.

Keeping into account all these interactions is tedious, so we make a simplifying assumption, inspired by the last consideration: that each avalanche resets the cell to a fixed state. In this model, a pulse can have at most one afterpulse. An eventual successive afterpulse must be produced by the first afterpulse, not the initial pulse. [CLR91] implicitly consider this not to be the case, since in their statistical analysis they keep into account the possibility of multiple afterpulses related to the same initial population of trapped carriers. Since these are second order effects, we anticipate that the probabilities involved are small and that we probably would not be able to falsify the simplified model with our data.

Bringing the DiCT model into the picture, each afterpulse will have its own DiCT tree. We suppose that a smaller avalanche should emit less photons and thus have less cross talk; again, for the sake of simplicity, we will assume that the DiCT probability remains the same instead.

Since the DiCT involves separate cells, even with our "resetting afterpulse" model if the initial pulse has more than one PE then multiple afterpulses due to the same initial pulse are allowed. In the following we refer to this case as *parallel afterpulses*, while the case with an afterpulse of an afterpulse as *series afterpulses*.

Figure 7.5: Cumulative histogram of the trigger leading edge (the first sample less than 600 in the recorded waveform) for 96 LNGS files. The distribution has the shape of a convolution between two uniforms with length 16 and 8. (`figtriggerhist.py`)

7.2 Data

We characterize DCR and correlated noise for Tile 21 from the LFoundry production. The following data files have been collected at LNGS at different overvoltages:

```
LFOUNDRY/pre-production-test/TILE_21/LF_TILE21_77K_54V_65VoV_X.wav
LFOUNDRY/pre-production-test/TILE_21/LF_TILE21_77K_54V_69VoV_X.wav
LFOUNDRY/pre-production-test/TILE_21/LF_TILE21_77K_54V_73VoV_X.wav
```

The '`X`' suffix in the file name stands for an index which goes from 1 to 10. Each file contains $\approx 20\,000$ events, so we have 200 000 events per overvoltage.

The overvoltage is obtained by subtracting the first voltage from the second in the file name and dividing by two. The first is the breakdown voltage while the second is the applied reverse bias; the division by two is due to the SiPM connection layout in the PDM consisting of several parallel branches each with two SiPMs in series. Thus the overvoltages are 5.5 V, 7.5 V, 9.5 V.

We show the time-value histograms of a single file per overvoltage in Figure A.4. Looking at the histograms it can be inferred that the laser fires at $\approx 9\,\mu s$. These files do not include the laser trigger waveform, but the external trigger from the laser ensures these waveforms are aligned within 22 ns. Since the DAQ configuration in the LNGS setup was not modified between the various data taking campaigns, we check this alignment by plotting in Figure 7.5 the position of the trigger rising edge from the runs in which it was recorded. The full range of the distribution is (8969 ± 11) Sa.

In some of the datasets for Tile 21 we observe double peaks in the fingerplot, i.e. there are two possible pulse amplitudes corresponding to the same number of PE. This is particularly evident in the first file at 5.5 VoV, `LF_TILE21_77K_54V_65VoV_1.wav`. In Figure 7.6 we show the fingerplot, both global and as a function of time, computed with 1.5 µs charge integration. It is evident that the pulse charge changes during the acquisition. Since it is still possible to separate the PE peaks, even if they are doubled, this will not be an issue. Moreover, with the filter we will use in the analysis the doubling will be much less marked.

In this data there could be a significative amount of light that hits the detectors after being reflected. Since the apparatus is small, we conservatively assume an upper bound of 1 m for the distance traveled by light, which corresponds to a 3 ns delay. This delay is smaller than the scale of the variations of the signal shape and of the noise, so we can neglect this problem.

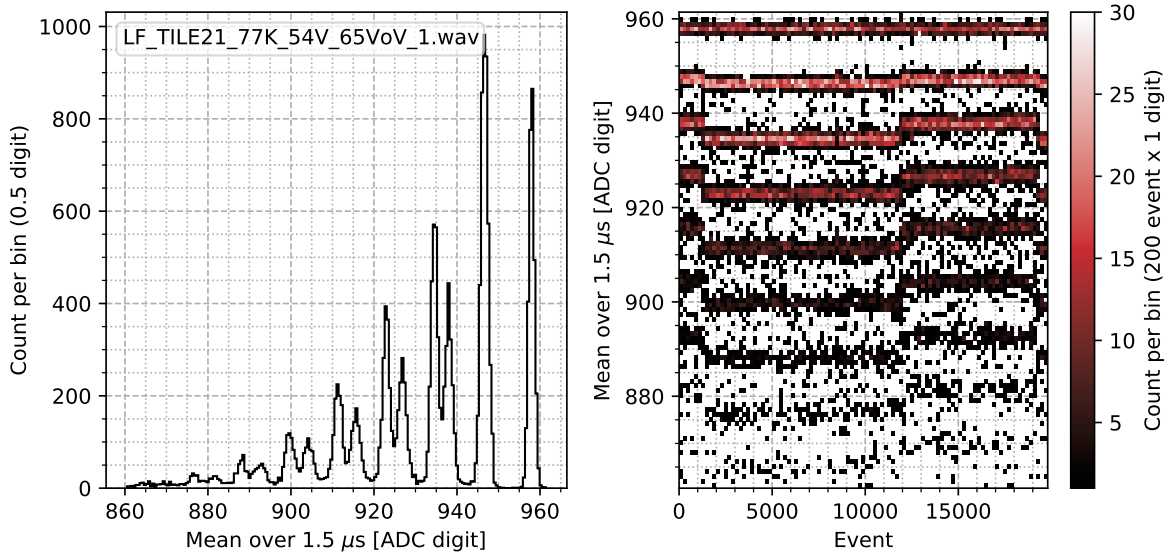


Figure 7.6: The distribution of the mean of the waveforms from sample 8958 to 10 457 (1500 samples) in the first file at 5.5 VoV. Left panel: global distribution. Right panel: the same distribution by groups of 200 events. (`figdoublepeak.py`)

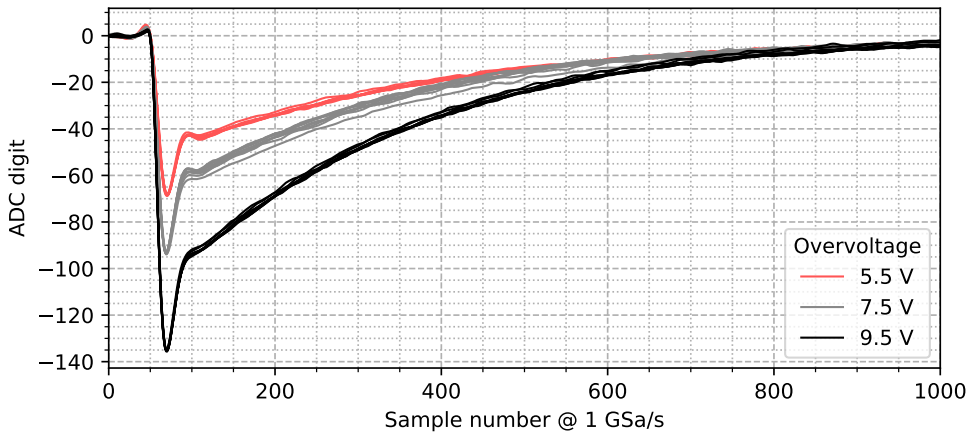


Figure 7.7: Templates of the pulse shape obtained with averaging for each file. (`figtemplates.py`)

7.3 Peak finding

To characterize the LFoundry Tile and classify the noise pulses we measure the amplitude and temporal position of all pulses in data. We filter the waveforms to suppress noise and then run a peak finding algorithm. This sections described the various steps of the signal processing.

7.3.1 Filtering

We filter each event using a cross correlation filter (see section 4.2). We follow the procedure outlined in section 4.4 to build the filter template. We determine the template separately for each file, to check for unexpected variations in the pulse shape. The templates obtained are shown in Figure 7.7. Although the variations appear to be reasonably small, we use each template only for its own file. An unexpected feature is the nonlinearity of the pulse amplitude with overvoltage. A possible explanation is a bookkeeping mistake, i.e., that the recorded operating operating bias voltage is wrong. We

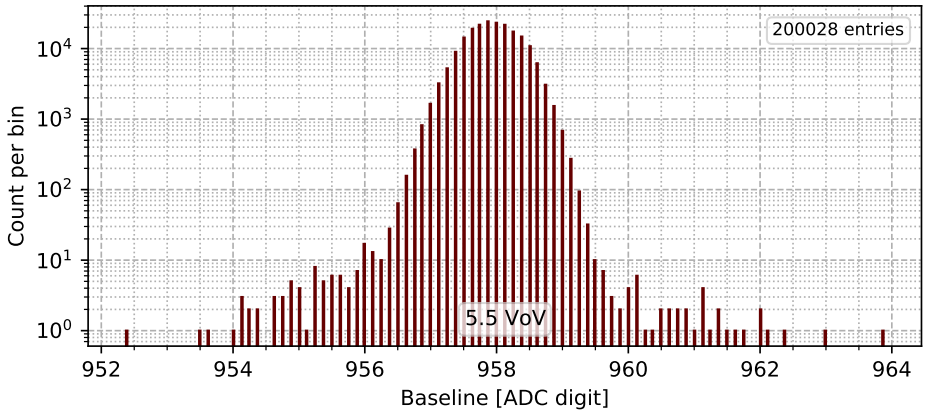


Figure 7.8: Distribution of the waveform baseline measured in the pre-trigger region of the events. See Figure A.1 for all overvoltages. (`figbaseline.py`)

will come back to this discussion after the analysis.

Filtering increases the signal to noise ratio, but degrades the separation between close peaks. Since we are studying correlated pulses, it is thus important to use a filter as short as possible. We filter the waveform with a logarithmic range of filter lengths: 32 ns, 64 ns, ..., 2048 ns. The computations we will describe are carried in parallel with all filter lengths, and then we choose in the analysis which lengths to use. The truncation of the template to the desired length is done as explained in section 4.4, keeping the fixed length subrange of samples that has maximum squared norm.

The template is normalized to unit sum for filtering, such that the filter behaves like an average and the baseline can be computed independently of the filter. This also means that signals will stay negative after filtering.

To evaluate the filter near the boundaries, the waveform is prolonged with the estimated baseline value, described in the next section. We defer to subsection 7.3.5 the description of how we select the filter length.

7.3.2 Baseline

To compute the height of the peaks we have to subtract the baseline value. We compute the baseline using the pre-trigger region of the waveforms. For robustness against deviations we use the median instead of the average. Since the input sequence is quantized with 10 bit resolution and the median can output only one of the values of the input sequence, the median is also quantized. To have a more continuous output we divide the array in 8 interleaved subarrays, i.e., the first subarray contains samples 0, 8, 16, ..., the second 1, 9, 17, etc., take the median separately on each subarray, and then average the medians. If any pre-trigger sample in an event is less than 700, for that event we reuse the baseline obtained in a previous event.

In Figure 7.8 we show the histogram of the obtained baseline values for the 5.5 VoV data. We will carry on the discussion always on the 5.5 VoV dataset as example, unless otherwise necessary. Appendix A contains additional plots that complete the picture.

The baseline distribution has a small tail to the left (note that the scale is logarithmic) and some far outliers. In Figure 7.9 and Figure 7.10 we show the events corresponding to the lowest and highest measured baselines and a pair of events from the lower tail respectively. (The event visualization contains elements which we have not introduced yet, they will soon be explained.)

In all cases the extreme baseline events do not show anomalies, they have

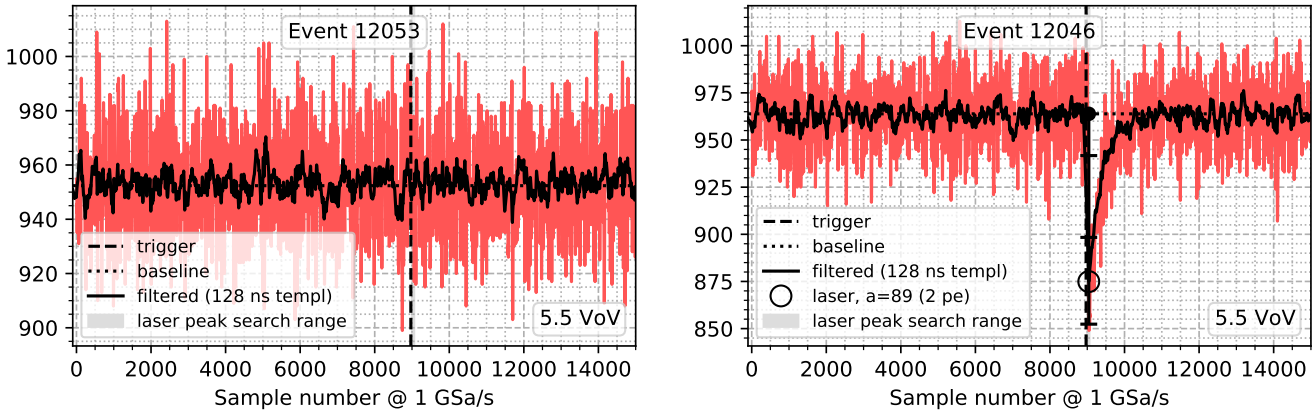


Figure 7.9: The events with the lowest and highest baseline. See Figure A.2 for all overvoltages. (`figbsoutlier.py`)

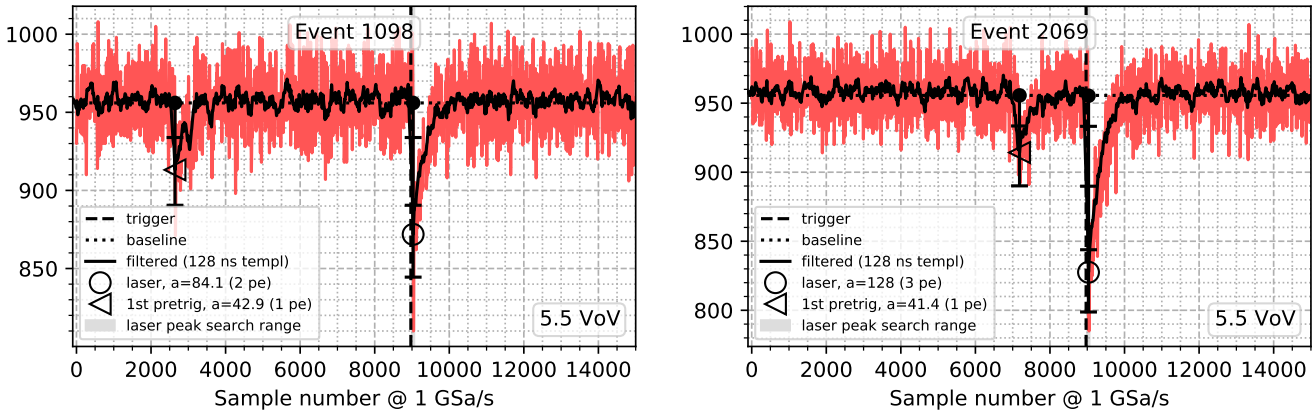


Figure 7.10: A pair of events from the low baseline tail (baseline between 955 and 956). See Figure A.3 for all overvoltages. (`figbstail.py`)

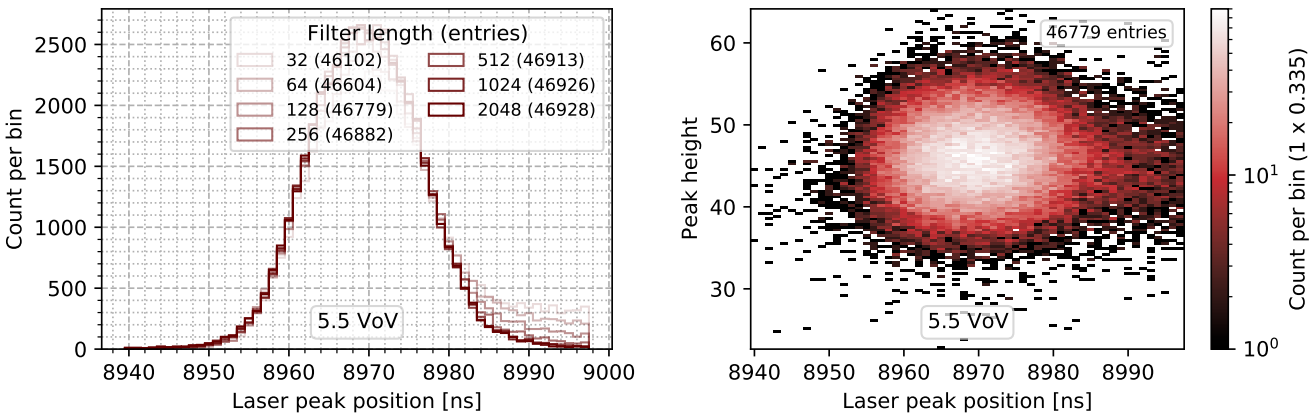


Figure 7.11: Left panel: histogram of the laser peak position, only for 1 PE peaks, for all filter lengths. The zero of the scale is such that the expected position is 8969. Right panel: 2D histogram of the laser peak position and the amplitude, for the same selection of peaks, but only with the 128 ns filter. See Figure A.5 for all overvoltages. (`figlaserpos.py`)

genuinely unusual baselines. At 5.5 VoV and 9.5 VoV the events corresponding to the lowest and highest baseline have close progressive indices, so they must occur close in time; we suppose then that these deviations are due to low-frequency transient oscillations.

The events in the tail all have a pre-trigger pulse (we checked only the two ones we plotted, we did not cherrypick). This means that we will systematically underestimate the amplitude of pre-trigger pulses. However, as is already evident by looking at the event plots, the bias is small compared to the pulse height. If data with lower SNR was to be processed, we would raise (recall the signals are negative) the baseline veto value from 700 to something as close as possible to the noise pedestal, and segment the baseline computation to detect inhomogeneity, but in the present case this is not deemed necessary.

7.3.3 Identification of the laser peak

The laser pulse occurs at a fixed offset from the start of the recorded waveform, so we search for the primary signal associated to the laser light emission in a range of ±30 samples around the expected position after filtering. When using different filters, we adjust the search window to account for the offset introduced by the filtering procedure. In this window we take the minimum local minimum, i.e., we consider the samples which have higher neighboring samples, and take the minimum of these. If there is no local minimum, which happens when the waveform is monotonically increasing or decreasing within the 60 selected samples, we mark the laser peak as missing.

As a crosscheck for this procedure we look at the distribution of the peak positions for 1 PE pulses (Figure 7.11, left panel). We select 1 PE pulses with a cut on the pulse height, described later in subsection 7.3.5. The distribution is centered in the expected place. However, it has a tail to the right when the filter length is short. A possible explanation could be that when the SNR is not high enough, the peak finder sometimes selects a close afterpulse instead of the primary pulse. Another possible interpretation is that this tail is due to random fluctuations.

As a first diagnostic, we look at the joint distribution of the peak position and height (Figure 7.11, right panel). If the minimum were moving to the right due to an additional close peak, we would expect a positive cor-

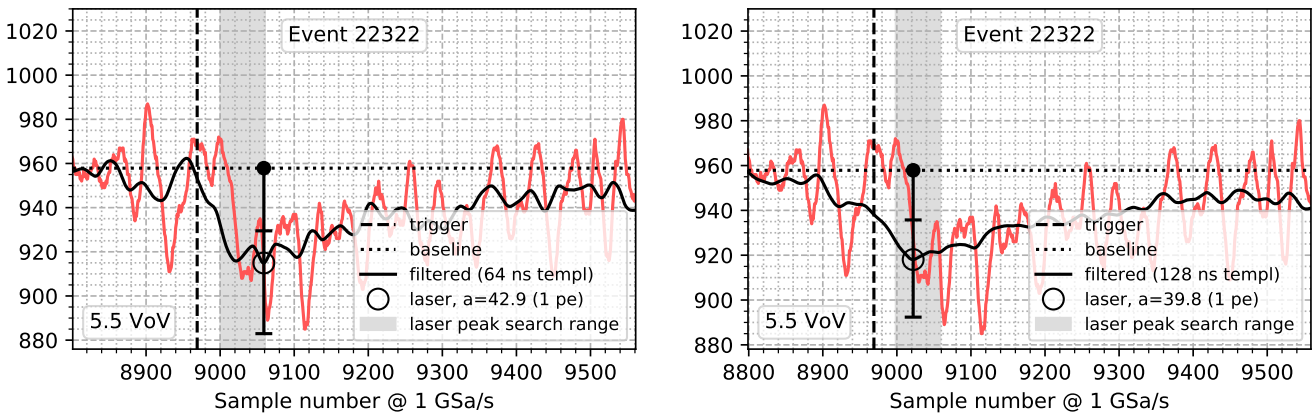


Figure 7.12: Left panel: an event with unusually late laser peak position for 1 PE peaks with filter length 64 ns. Right panel: the same event with filter length 128 ns. See Figure A.6 for all overvoltages. `(figlptail.py)`

relation between height and position in the tail. Instead, we see a negative correlation.

Then we look at the events themselves. In Figure 7.12 we show an event in the tail at 64 ns, filtered with 64 ns and 128 ns. With the longer filter the position goes back to the center of the distribution. We inspected a lot of events in the tail and they are almost all like the one we show.

Finally, the tail contracts as the overvoltage increases (see Figure A.5). From these observations we induce that the right shift is caused by random fluctuation. The asymmetry of the tail reflects the asymmetry of the shape of the signal.

We said that with our procedure a laser peak can be missing. However we know that the laser is always present; even when there are no pulses, we need to count the event as a 0 PE laser signal to fit the Poisson+DiCT distributions.

First, we count these "missing laser" events for each filter length (Figure 7.13, left panel). They are about 1.5 % of the total number of events in the dataset, 200 000, so not negligible. We went through a lot of these events and they are almost all genuinely missing pulses. It turns out that the noise is actually quite likely not to produce a local minimum in a 60 ns window after filtering. Nevertheless, there are some events which have a pulse which for some weird combination of noise oscillations is not detected properly.

As in the case of the position distribution tail, the anomalies tend to appear only for a particular filter length choice and disappear on the same event with other lengths. So as a simple solution we use the shortest filter which yields a non-missing peak when the preferred filter does not work. The right panel of Figure 7.13 shows how the fraction of missing events decreases as we allow more lengths to choose from. Even with this fix, however, there is a hard core of events which do not have a local minimum in any case, about 0.15 %.

These events are more interesting, so we show 12 of them at random for each overvoltage in Figures A.7, A.8, and A.9. By eye we identify three cases: 1) true missing pulses, 2) delayed pulses which fall out of the window, and 3) pulses which are shadowed by a very high close consecutive pulse. In Table 7.1 we list the total number of hard misses and the counts of the three categories in the sample for all overvoltages.

The delayed pulses could be photons that produce carriers in an inactive region that then migrate to a diode junction (we are not sure about this, it

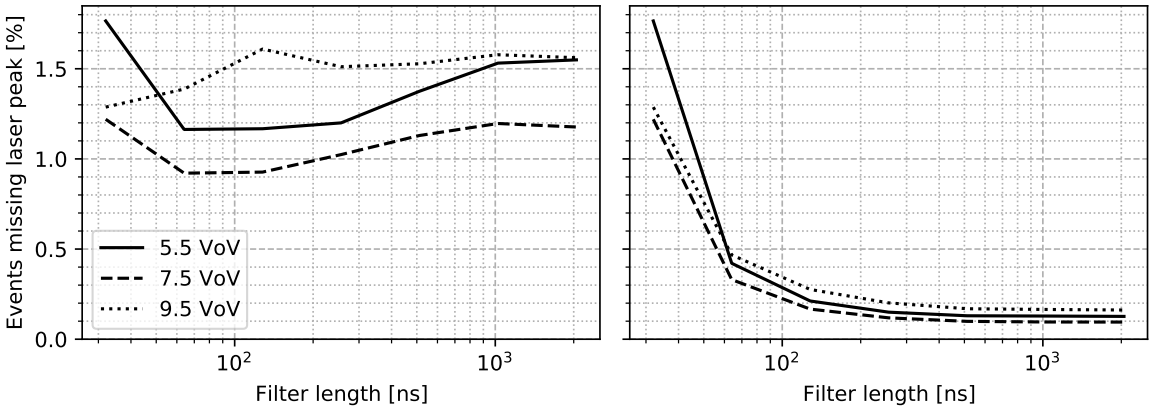


Figure 7.13: Left panel: for each overvoltage, the fraction of events where there is no local minimum in the laser peak search range, as a function of filter length. Right panel: the fraction of events where the minimum is missing for all filter lengths less than or equal to the one on the abscissa. (`figmissing.py`)

		Count of 12		
Overvoltage [V]	Total	True miss	Delayed	Close consecutive
5.5	253	10	1	1
7.5	190	7	3	2
9.5	324	7	2	3

Table 7.1: The number of events where the laser peak is missing with all filter lengths, and the counts, in a random sample of 12 of these events, for the three categories of configurations that appear in this selection. (`figmissing.py`)

is just speculation). The consecutive pulses could be DeCT with high DiCT, they would have the time scale we expect from [Sav18, fig. 3.8 p. 54] already mentioned in subsection 7.1.1, or less likely afterpulses.

Since these events are a small fraction of the sample we did not investigate this further. We will just keep in the back of our minds that there are these events and check case by case that ignoring them has a negligible effect.

7.3.4 Identification of secondary peaks and peaks not associated to the laser pulse

We identify other pulses using a prominence-based peak finder. In this section we describe the peak-finding algorithm as applied to positive pulses (low-to-high-to-low). By changing the sign of the input, the same algorithm can be used to find minima, i.e., peaks corresponding to negative signals.

The topographical prominence is defined as follows: starting from the peak whose prominence is to be measured, go to the right and to the left until an higher elevation is found. For each side, take the minima between these stops and the peak. The prominence is the difference in elevation between the peak and the maximum of the two minima. See Figure 7.14 for an illustration.

We search the non-laser pulses separately in the pre- and post-laser peak regions. We use the position of the laser peak, when present, to delimit the two regions. This means that even if the laser peak is lower than the peak, the prominence "exploration range" stops there. When the laser peak is missing, we use the left edge of the laser peak search window as delimiter.

If one of the minima occurs at the edge, it is ignored and the prominence

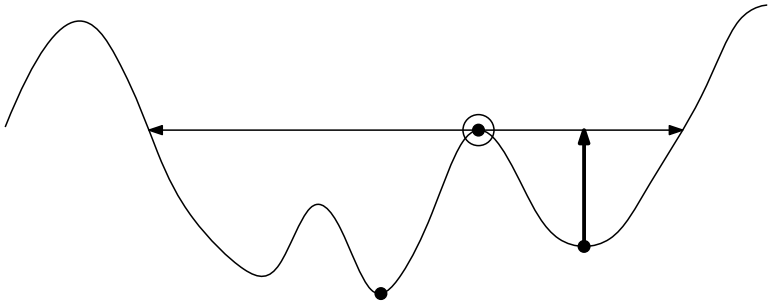


Figure 7.14: Illustration of the definition of topographical prominence. The prominence of the circled peak is the length of the thick vertical arrow.

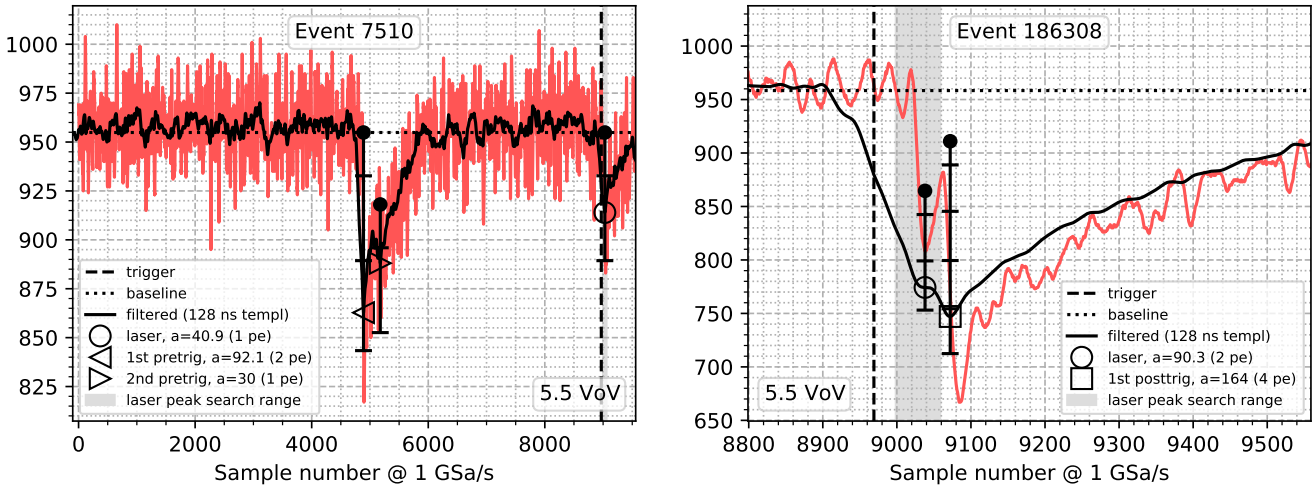


Figure 7.15: The events with the closest pair of pre-trigger pulses (left panel) and post-trigger pulse closest to the laser pulse (right panel) for at least 1 PE peaks. (`figpeakfinder.py`)

is computed using the minimum on the other side, unless both minima occur at the edge. This is to avoid assigning a very low prominence to a pulse whose shape is truncated due to being close to a boundary.

The minima are capped to the baseline: if a minimum is lower than the baseline, the value of the baseline is used instead to compute the prominence. This increases the ratio between the prominence of close pulses to the one of isolated pulses, because if the explored range is longer it is easier to find a deeper random oscillation.

For each region, we save the two most prominent peaks. We do not apply a threshold on the prominence, we always fill the four slots per event. In this way we can observe the distribution of the height of random peaks and choose a good threshold afterwards.

In peak finding algorithms it is customary to require a minimum distance between peaks. This is to avoid picking up close high peaks due to random oscillation that actually correspond to a single pulse. Selecting by prominence avoids this problem if the waveform is smooth, because close peaks will have only a shallow dip between them and the prominence of the lower one will be measured relative to that, resulting in a very low prominence. Our waveforms are smooth on the scale of the filter length, so it did not turn out necessary to add a distance criterion.

In Figure 7.15 we show two examples of events with close pulses correctly identified.

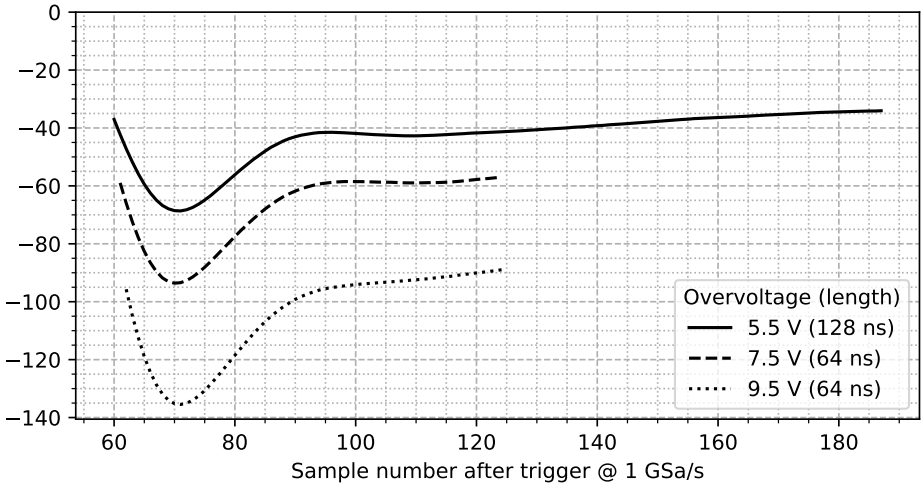


Figure 7.16: The principal filter templates used in the analysis, shown unnormalized. (`figtrunc.py`)

7.3.5 Number of photoelectrons associated to a peak

To assign the number of PE to the peaks we need to define bins for the height of the peaks. We want to choose the filter length that yields the optimal separation between PE in the height distribution. A longer filter suppresses the random height fluctuation due to noise; at the same time, however, it picks up afterpulses with the template tail, adding an upper tail to each PE height distribution. Since the afterpulse probability increases with the number of PE of the primary pulse, the distributions for 0 and 1 PE are the least affected.

In our analysis we will care particularly about selecting 1 PE pulses and separating the afterpulses. So we use the following criterion: we take the shortest filter that allows to clearly separate the 1 PE distribution. These filter lengths are respectively 128 ns, 64 ns, 64 ns for 5.5 VoV, 7.5 VoV, 9.5 VoV. We show the templates for these lengths in Figure 7.16.

We tried selecting away the afterpulse tails using the peak finder to locate afterpulses, however we did not manage to reduce them significantly. We interpret this as the tail being composed mainly of afterpulses occurring shortly after the primary peak, that can not be recognized individually due to their low height. Being located on the slope of the laser peak, the prominence of a close afterpulse tends to be lower than that of an isolated pulse with the same height. For close enough afterpulses, the prominence will often be lower than noise fluctuations, and the peak finder will fail in identifying them.

There are other two minor factors that deform the height distribution: the saturation of the digitizer, and the tails of pre-trigger pulses close to the laser one. Thus we histogram the laser peak height excluding events with saturation and events where the peak finder locates a pre-trigger pulse higher than the upper endpoint of the random peaks height distribution within 2.5 µs before the trigger using the longest filter. This histogram is shown in Figure 7.17 for 5.5 VoV and Figure A.10 for all overvoltages.

We define the PE bin boundaries in Figure 7.17 taking the midpoints between the two most distant consecutive heights in the range between each pair of consecutive height distribution peaks. Laser peak heights above the last boundary are assigned to an overflow bin when necessary. The ruler ticks that accompany the peaks in the event visualizations are these boundaries.

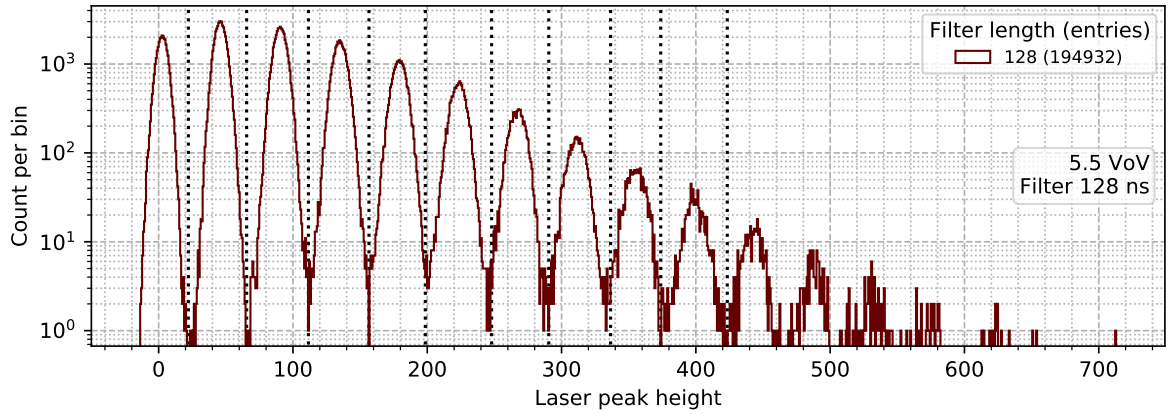


Figure 7.17: Histogram of the laser peak height, with a selection to avoid biased heights. The dotted lines are the boundaries of the PE bins. See Figure A.10 for all overvoltages. (`figpe.py`)

Note the dips between the peaks in the histograms (the vertical scale is logarithmic). The lowest bin count is at most 4 in all cases. Let us put a rough upper bound on the contamination. This allows us to place a rough upper bound of 1 % on the bin-to-bin contamination. This in particular is surely less than the Poisson error on the count for each PE. The 1 PE peaks are well separated, with a row of 0 or 1 count bins around them, so we estimate a contamination of 0.01 %.

7.3.6 Determination of the amplitude of close pulses

When two pulses are close to each other, the height contains a contribution from the tail of the other pulse. We need to compute the amplitude of the single pulse, i.e., the height it would have if it was isolated. To simplify the problem, we make the following assumptions:

1. the position of a peak does not depend on the presence of other pulses;
2. the shape of afterpulses is the same as normal pulses.

About the first assumption: we have seen that the peaks in the output of the cross correlation filter have a sharp cusp. A perfectly sharp cusp, even if summed to a sloping shape, remains in the same position. A rounded peak, instead, will move slightly in the uphill direction when summed to a slope. Given the short filters we are using, these cusps are not really sharp, as shown in the right panel of Figure 7.15: the peak in the square box has a radius of less than 10 ns. Thus we estimate the typical bias due to an underlying exponential tail to be around 5 ns.

The second assumption is commonly made in the literature. But it rests on the assumption that the pulse shape does not change with overvoltage, such that for an avalanche occurring while the cell is recharging, only the overall amplitude of the pulse will be affected. Figure 7.18 shows that while the shape of the signal pulses varies with overvoltage for the Tile under study, the difference is small enough to justify the working hypothesis.

Now, let $\mathbf{y}$ be the output of the filter applied to a single noiseless pulse, which we compute by filtering the signal template. Let t_α be the position of peak α and a_α the unknown amplitude. Since the filter is linear, the filtered sum of the pulses and the noise is the sum of each component filtered

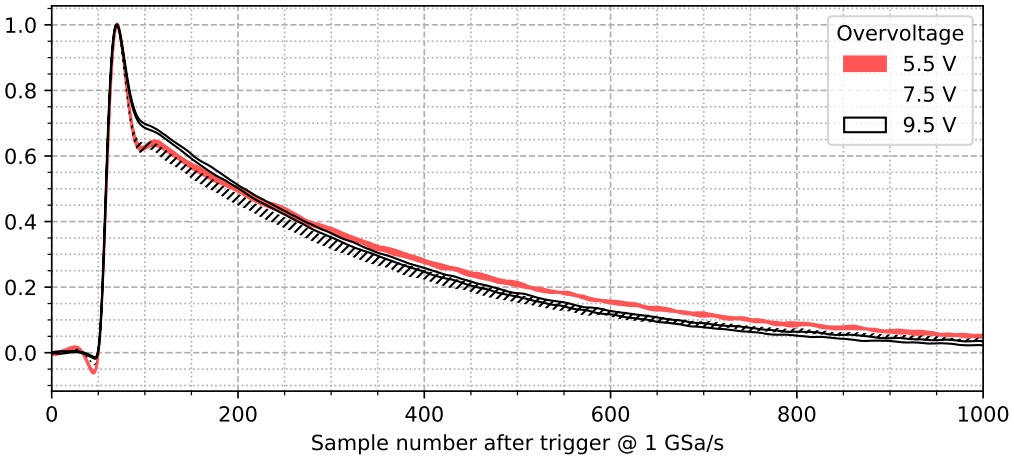


Figure 7.18: The signal templates at different overvoltages compared fixing the peak amplitude to 1. The thickness of the stroke is twice the standard deviation over the 10 files per overvoltage, while the center of the stroke is the mean. (`figshape.py`)

separately, so the model for the complete filtered waveform $\mathbf{w}$ is

$$w_i = \sum_\alpha a_\alpha y_{i-t_\alpha} + \epsilon_i, \tag{7.11}$$

where $\boldsymbol{\epsilon}$ is the filtered noise. There is some freedom in the definition of $\mathbf{y}$: we fix that the peak occurs at y_0 and that $y_0 = 1$, such that if there is only one peak then $w_{t_1} = a_1 + \epsilon_{t_1}$.

It is well known that the minimum variance in estimating $\mathbf{a}$ is obtained by using least squares with the inverse of the covariance matrix V of $\boldsymbol{\epsilon}$ as quadratic form. Let $H_{i\alpha} = y_{i-t_\alpha}$, then the solution is [Zyl+20, p. 628]

$$\hat{\mathbf{a}} = (H^\top V^{-1} H)^{-1} H^T V^{-1} \mathbf{w}. \tag{7.12}$$

We note, however, that to solve the system it is just necessary to have as many datapoints as the number of peaks. It then comes natural to solve for simplicity this system of equations instead:

$$w_{t_\beta} = \sum_\alpha \hat{a}_\alpha y_{t_\beta - t_\alpha}, \tag{7.13}$$

i.e., we use only the observed peak heights w_{t_β} instead of the full waveform.

To justify this simplification, we recall that in section 4.2 we said that the matched filter is equivalent to linear least squares. We now indeed show that, if we were using the matched filter, solving (7.13) would be equivalent to least squares and thus optimal. Just for this proof, let us use all the above notation, but without the filter applied, i.e., $\mathbf{w}$ is the unfiltered waveform, $\mathbf{y}$ is the unfiltered signal, V is the covariance of the unfiltered noise.

The model (noise implicit) is

$$\mathbf{w} = H\mathbf{a}. \tag{7.14}$$

The filter is applied by operating with the matrix

$$F_{ij} \equiv V^{-1}_{jk} y_{k-i}. \tag{7.15}$$

Here we can not write the filter as the cross correlation with a fixed template because we are not assuming the stationarity of the noise. If that was the case, V^{-1} would be invariant under the paired translation of its indices, and

thus (7.15) could be expressed with a template, and would depend only on $j - i$. In other words, think that the template changes at each point due to the different noise spectrum. Applying the filter to (7.14) we have

$$F\mathbf{w} = FH\mathbf{a}. \tag{7.16}$$

We want to use only the peak heights in the filter output, assuming that we know exactly the true signal positions. These are

$$W_\beta \equiv (F\mathbf{w})_{t_\beta} = V^{-1}_{jk} y_{k-t_\beta} w_j, \tag{7.17}$$

which in vector form becomes

$$\mathbf{W} = H^\top V^{-1} \mathbf{w}. \tag{7.18}$$

So, considering only the W_β on the left hand side of (7.16) and putting the indices on the right hand side, we have

$$W_\beta = V^{-1}_{jk} y_{k-t_\beta} H_{j\alpha} a_\alpha = \tag{7.19}$$

$$= (H^\top V^{-1} H)_{\beta\alpha} a_\alpha, \tag{7.20}$$

thus the solution is

$$\mathbf{a} = (H^\top V^{-1} H)^{-1} H^\top V^{-1} \mathbf{w}, \tag{7.21}$$

which is the least squares estimator (7.12). □

Our filter differs in two ways from the matched filter: it does not keep into account the noise correlation, and it is truncated. So our method for computing the amplitude is worse than the optimal one as much as the filter is worse than the matched filter.

In each event we have to decide which peaks to input in (7.13). Most of the time, peaks are random noise oscillation. If the amplitude of random peaks had mean zero, adding a fake peak would increase the error but not introduce a bias. But, since we are selecting peaks by highest prominence, it is guaranteed that in a region of some microseconds we will find a peak with height comparable to the noise standard deviation. Indeed in the next section we will look at the height distribution for the peak finder output and see that it is positively biased. So we compute the amplitude only for peaks higher than a threshold. As threshold we pick the PE bin boundary between 0 and 1 PE.

When we use the computed pulse amplitude instead of the peak height, we would like to also see the distribution of the fake peaks such that we can be sure at a glance that we have no contamination, but most of the fake peaks do not have an amplitude since they do not pass the threshold prerequisite. The y normalization we have chosen makes the amplitude "have the same units" of the height, so we construct a variable which is the amplitude when available, and the height otherwise. In the following sections we mean this variable when we talk about "amplitude".

Note that with this selection we are not excluding low height afterpulses from the amplitude computation, because even if their amplitude is smaller, sitting on the tail of their parent pulse their height from baseline is still above threshold.

In the event visualizations, the ruler associated to each peak starts from a dot placed such that the distance from the dot to the peak is the amplitude. In the legend, the amplitude is given under "$a = \ldots$".

We checked that recomputing the PE bins with the laser pulse amplitude instead of the height gives almost the same result. In the analysis, anyway, we use the bins computed with the amplitude for complete consistency.

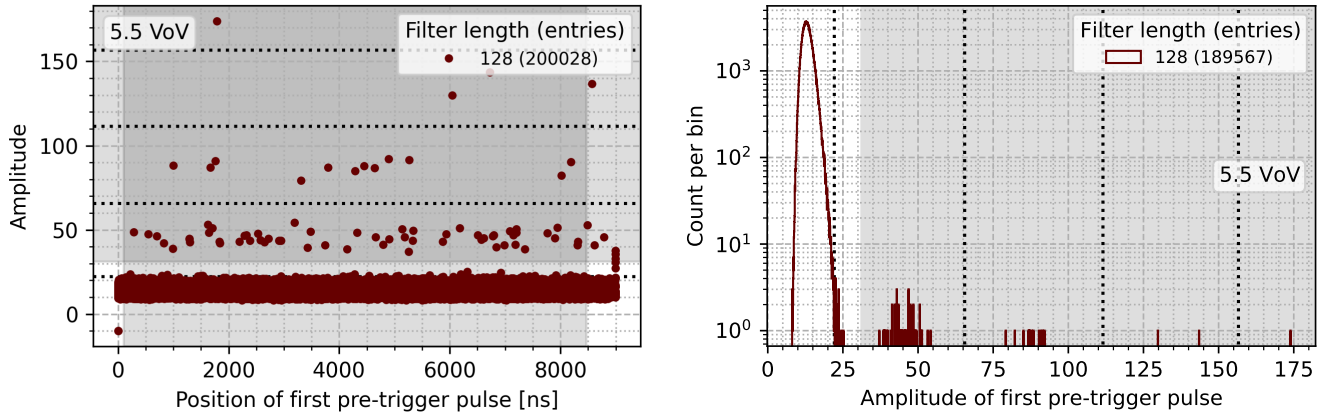


Figure 7.19: Left panel: scatter plot amplitude vs. position of the chronological first pre-trigger peak in each event. Right panel: histogram of the amplitude for the peaks contained in the vertical gray band in the left plot. See Figure A.11 for all overvoltages. (`figpretrigger.py`)

7.4 Random pulses rate

When counting afterpulses there will be a background from the dark count rate, which we have to subtract, so we measure it in the pre-trigger region. The laser trigger frequency should be less than 1 kHz, so the events are distant between each other and there is not the possibility of correlated noise from a laser pulse leaking into the successive event.

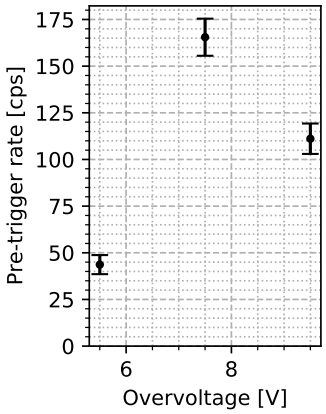
The title of this section reads "random pulses rate" instead of dark count rate because we will see we have reason to believe that in the data there are actually more random pulses than the dark count. It does not matter for background subtraction though.

By looking at the time-value histograms in Figure A.4 we see that in 20 000 events there are just a few pre-trigger pulses, so we neglect the case of a double random pulse in the same event. If there are two pre-trigger pulses, one must be the afterpulse of the other, so the event counts as one random pulse. For the rate we do not care to know which pulse is the primary, but we will need it when fitting the DiCT model to the random pulses.

The primary pulse is of course the first in chronological order, but to distinguish the one pulse from the two pulses case we are forced to put a threshold on the amplitude before looking at the distribution. We do something similar as we did for the amplitude. The initial variables are the position, height and amplitude of the two most prominent pre-trigger peaks. When both heights are above the 0 to 1 PE boundary, the first set of variables is set to the first peak in chronological order; otherwise to the most prominent peak.

In the left panel of Figure 7.19 we show the scatter plot of the first pulse amplitude versus position. There are some problems at the edges. On the left edge, the negative amplitudes are actually very large values that we mapped to -10. They are probably due to the amplitude linear system (7.13) being degenerate, but we did not investigate why this may happen. On the right edge conversely there are some zeroes and anomalously small values. Thus we ignore the first 100 ns and the last 500 ns of the pre-trigger region. The wider margin on the right is just to be safe in case the presence of the laser pulse was having some effects. The selected region is marked by the gray vertical band in the plot.

The right panel of Figure 7.19 shows the histogram of the amplitude of the



Overvoltage [V]	Events	Time		Pulses	
		Per event [µs]	Total [s]	Count	Rate [cps]
5.5	200 028	8.37	1.67	73	44 ± 5
7.5	200 021	8.37	1.67	277	165 ± 10
9.5	200 025	8.37	1.67	186	111 ± 8

Table 7.2: The rate of primary pre-trigger pulses in a fiducial region and the intermediate quantities used to compute it. (`figptrate.py`)

pulses in the temporal cut. The separation between random peaks and 1 PE pulses is clear; the amplitude threshold we use is marked with a gray band also reported in the left panel. The dotted lines are the PE bin boundaries. As we anticipated, the distribution of the amplitude of the random peaks is biased upward, and leaks a bit above the 0 to 1 PE boundary.

To compute the rate, we count the number of pulses satisfying the cuts, and divide by the number of events times the pre-trigger region duration minus the cuts. The uncertainty is given by the Poisson error of the count. In Table 7.2 we report the obtained values.

We notice that the rate at 7.5 VoV is substantially higher than the one at 9.5 VoV. The dark count rate steadily increases with overvoltage (see for example [Sav18, fig. 3.13 p. 61]), because an higher field in the diode lowers the potential barrier that a carrier must overcome to become conducive. So we induce that there must be an additional source of random pulses. Maybe the experimental setup was not light-tight. Anyway, these values can be considered an upper bound for the dark count rate. For reference, the maximum dark count rate allowed by the DarkSide20k specifications is 250 cps [Sav18, tab. 3.1 p. 62].

7.5 Afterpulses

To study afterpulses we consider the post-trigger peaks in events with a 1 PE laser pulse. Using the terminology introduced at the end of subsection 7.1.3, a single PE initial pulse implies that there are no parallel afterpulses. The parameter we are interested in is the probability for a single cell to generate an afterpulse, thus even if there are two series afterpulses, we only care about the first. So, like we did for pre-trigger peaks, when there are two peaks with height above the 0 to 1 PE boundary we select the first.

The case with an afterpulse and a random pulse is quite unlikely. From Table 7.2 we compute that the probability of having a random pulse in the post-trigger region is within 0.1 %. We will see that the probability of an afterpulse is about 5 %, so the coincidence probability is less than 0.005 %, in less than 50 000 events, so we expect less than 2.5 afterpulse+random events. Since the distribution of randoms is flat, while the afterpulses are concentrated at short delays, and keeping into account that even if both were uniform in half of the cases the afterpulse would come first by chance, we are sure that the expected afterpulses lost due to random are less than 1 and so we neglect this problem.

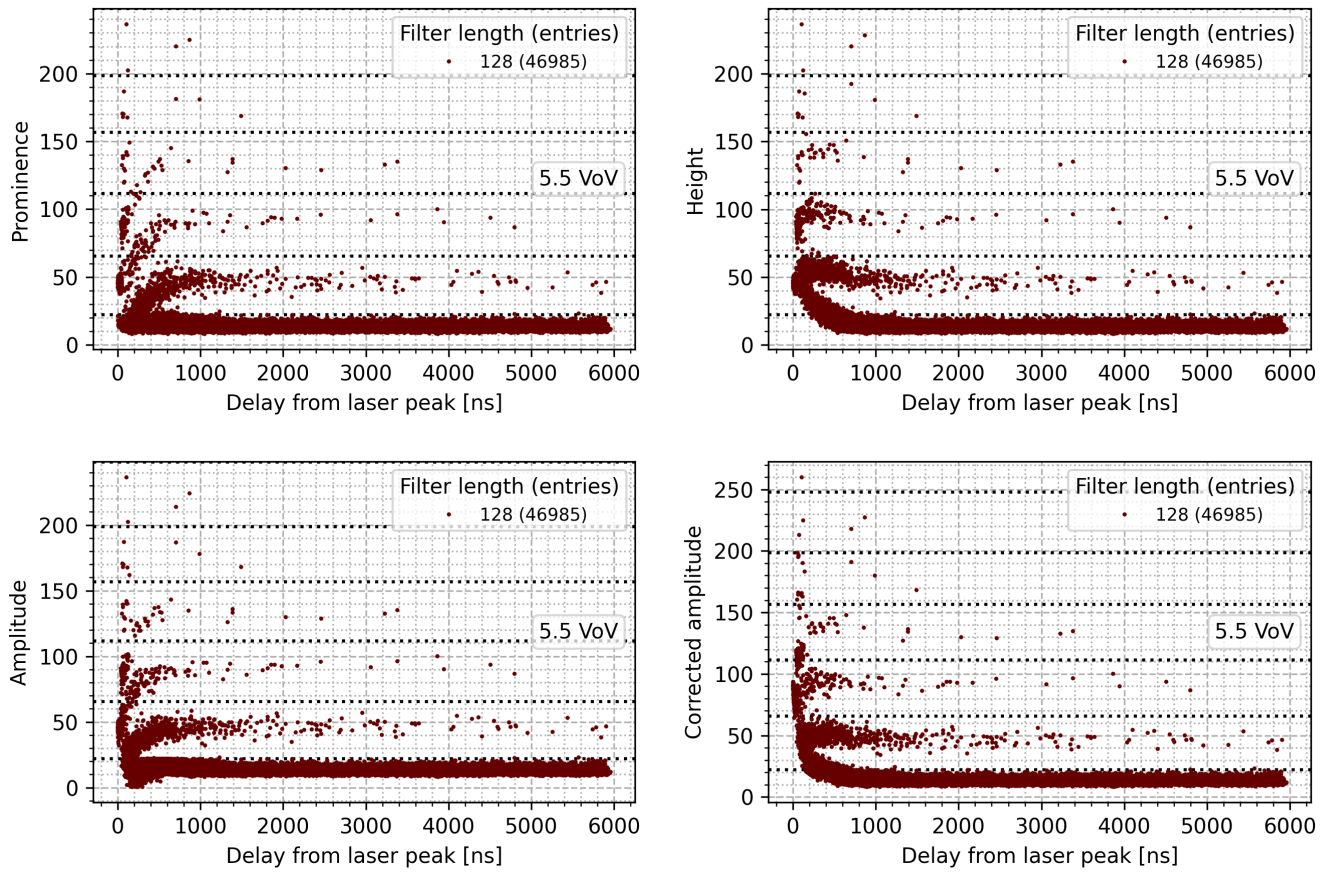


Figure 7.20: Various ways of measuring the magnitude of post-trigger pulses. Top left: the prominence computed by the peak finder; it decreases with short delays so it does not allow a straight cut. Top right: the height from the baseline; it is straighter but leaks a bit above the 2 PE threshold. Bottom left: the amplitude; it yields a better separation than the prominence but has to be rectified. Bottom right: the corrected amplitude which we use in our analysis. (`figapampl.py`)

The amplitude of the afterpulses is shorter than normal at low delays. It would be convenient to select them with a straight cut. This will be even more useful when fitting the DiCT model on the afterpulses. We empirically observe that the height of the afterpulses is approximately constant, although a bit higher at short delays, like if the rule was that an afterpulse reaches a normal height when "sitting" on the tail of its parent (see the top right panel of Figure 7.20). The slight increase for close pulses can be interpreted as an effect of the smearing of the filter. So we do the following: we take the *unfiltered* signal template, normalize it to have peak amplitude as a 1 PE *filtered* pulse, take its value at a delay from its peak equal to the delay from the post-trigger to the laser peak, and sum this to the post-trigger pulse amplitude.

The obtained *corrected amplitude* is shown in the bottom right panel of Figure 7.20. Note that this correction is not equivalent to dividing by the expected recharge factor (7.10), it is just an empirical procedure that rectifies the observed distribution. This may indicate that the model is inaccurate, however we did not investigate further. In the event visualizations, the amplitude shown for post-trigger pulses is actually the corrected amplitude.

We can now discuss an additional issue with selecting the first peak when there are two of them. Before computing the amplitude we select by height. Pulses sitting on the tail of the laser pulse have a height bonus, and we said this is useful to avoid selecting away short afterpulses at this stage. However

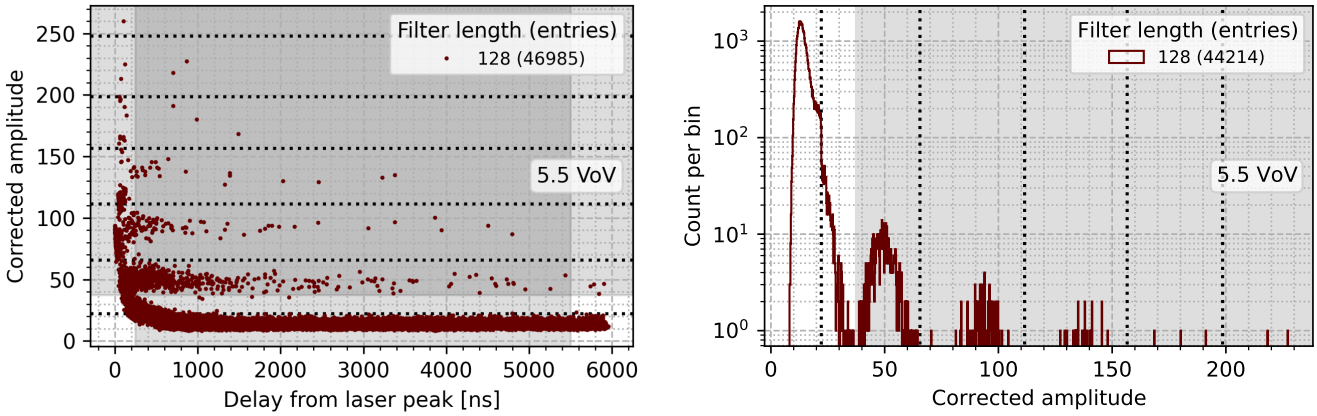


Figure 7.21: Left panel: corrected amplitude versus delay from laser peak; the gray bands mark the selected ranges. Right panel: the histogram of the amplitude for the points that fall in the vertical band in the scatterplot. See Figure A.12 for all overvoltages and a zoom in of the scatterplot. (`figapscatter.py`)

some of these short pulses will turn out to be noise. If it happens that there is a second true pulse, it will be hidden by the first noise pulse. So we take the first chronological only when both have their corrected amplitude above the 0 to 1 PE threshold. We checked that at 5.5 VoV there would be at most 13 cases with this problem, to be compared to about 500 selected post-trigger pulses, so the effect would be small anyway.

To select the afterpulses we have to ignore short delays because we can not separate too short pulses from the noise, and because the peak finder could be losing some pulses if they are short and close to the laser peak. Moreover, like with the random pulses, the threshold on the corrected amplitude has to be set higher that the 1 PE boundary because the amount of noise is overwhelming compared to the pulses. We also truncate the delay distribution at approximately 500 ns from the end of the waveform just to stay safe from boundary effects. We first do the temporal selection, and then determine the threshold with our usual procedure of taking the midpoint between the two most distant consecutive samples. These selections are shown in Figure 7.21.

We have done a temporal cut, thus to count the afterpulses we have to calculate how many we have lost. To do this we need to determine the temporal distribution and compute the probability mass that falls outside of the selected range. We histogram the delay of the pulses and fit two distributions, one with an exponential decay component and one with two.

Since we are selecting the first pulse after the laser peak, normally it would be necessary to correct the distribution for a "first arrived" effect: an eventual second pulse would not be counted, so the observed counts decay faster with delay than the actual distribution. Simple example: start from a uniform distribution, and take the first event after a fixed point in time; the distribution transforms into the well known exponential. [CLR91, p. 2] and [Gar+14, p. 4] both take this into account. In our case, however, we have a model where caring only about the first pulse is "built-in" so to say. Just for reference, asymptotically the correction would amount to the probability of afterpulses, which as we will see is about 5 %.

This is not true for the background of random pulses, which would need the correction, but its probability is so small that the correction can be neglected. In other words: we approximate the exponential background of random pulses with a uniform because the rate is very low compared to the

observed time interval.

We do an approximate Bayesian fit using least squares. This is described in detail in Appendix B; for the impatient, just consider that this is almost a standard least squares fit of a histogram, with additional squared terms that represent the prior.

The priors of the fit are:

- The expected uniform background, computed from the random pulse rate in Table 7.2. The variance is determined only from the variance of the rate; the afterpulse count i.e., the histogram normalization is an errorless input.
- For the one component fit, the prior on the logarithm of the exponential decay constant in nanoseconds is $\log(1000) \pm 1$. Consider than a ± 1 error on the natural logarithm corresponds to a 100 % relative error with first order propagation.
- For the two components fit, the priors on the logarithms are $\log(400) \pm 1$, $\log(800) \pm 1$, while the prior on the relative weight of the short component is a uniform in $(0, 1)$.

All the parameters are bounded, so they are fit transformed. The background density and the exponential scales are transformed with the logarithm, while the component weight is mapped to the unitary interval using the error function, which makes the prior on it uniform.

Since the density varies a lot over the observed range of about 5 µs, we use uneven bins for the fit. As simple criterion we compute the bins that would yield uniform counts with an exponential with scale 1.5 µs:

$$b_i = t_L - \tau_0 \log\left(1 - \frac{i}{n}\left(1 - \exp\left(-\frac{t_R - t_L}{\tau_0}\right)\right)\right), \quad i = 0, \ldots, n, \quad (7.22)$$

where there are n bins, b_i is the i-th bin edge, $\tau_0 = 1.5\,\mu s$, and (t_L, t_R) is the total range. For n we take $3/4$ of the square root of the total count. With these criteria it never happened to have a bin with a count less than 5.

We are using somewhat large bins, so we have to fit against the integral of the distribution instead of the usual approximation of taking the density in the center of the bin. The cumulative density functions for the two models are:

$$P_1(t_L < t' < t; \tau, R) = \left(1 - \frac{R}{N}\right)\left(1 - e^{-t/\tau}\right) + \frac{R}{N}\frac{t}{t_R - t_L}, \quad (7.23)$$

$$\begin{aligned} P_2(t_L < t' < t; \tau_1, \tau_2, p_1, R) = {} & \left(1 - \frac{R}{N}\right)\left[p_1\left(1 - e^{-t/\tau_1}\right)\right. \\ & \left. + (1 - p_1)\left(1 - e^{-t/\tau_2}\right)\right] \\ & + \frac{R}{N}\frac{t}{t_R - t_L}, \end{aligned} \quad (7.24)$$

where τ, τ_1 and τ_2 are the exponential scales, p_1 is the weight of the component with the shorter prior, R is the number of random pulses, and N is the total histogram count.

Once we have the fitted parameters, we correct the afterpulse count by subtracting the fitted background and dividing by the probability mass contained in the range:

$$N_1 = \frac{N - R}{\exp(-t_L/\tau) - \exp(-t_R/\tau)}, \quad (7.25)$$

$$N_2 = \frac{N - R}{p_1(e^{-t_L/\tau_1} - e^{-t_R/\tau_1}) + (1 - p_1)(e^{-t_L/\tau_2} - e^{-t_R/\tau_2})}, \quad (7.26)$$

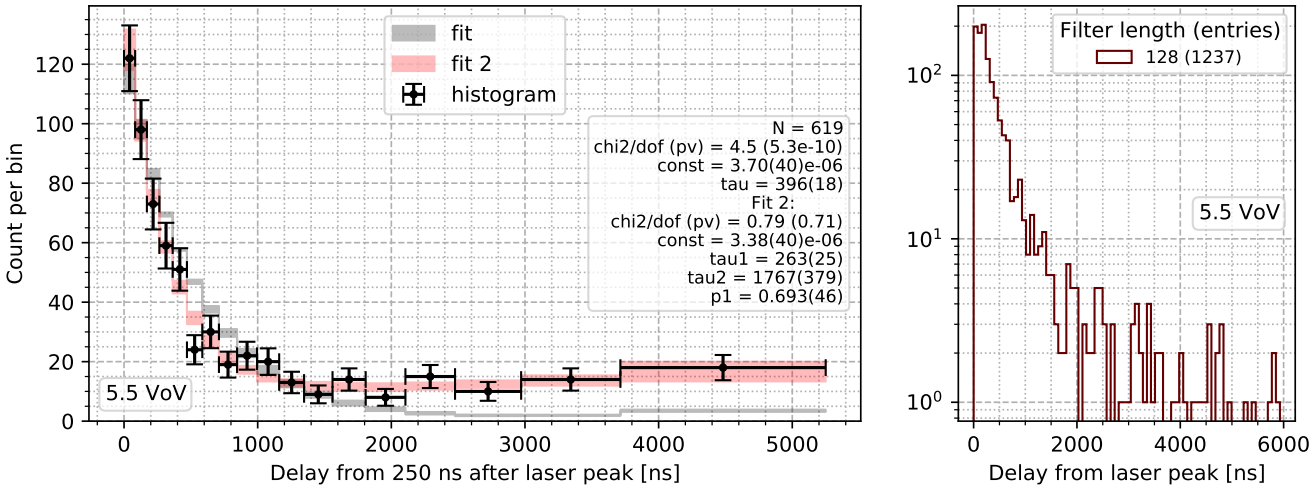


Figure 7.22: Left panel: fit of the temporal distribution of post-trigger pulses. Right panel: the histogram with even bins and without the temporal cuts. In the legend, 'const' is the background excess density, i.e., $(R/N)/((t_R - t_L)(1 - R/N))$, in ns^{-1}. See Figure A.13 for all overvoltages. (`figapfit.py`)

	Delay range						Fit parameters						Fit quality		
OV [V]	tL [ns]	tR [ns]	Events	N	Time [s]	R prior	R	tau1 [ns]	tau2 [µs]	p1 [%]	Correction	Prob. [%]	chi2	dof	pvalue
5.5	250	5500	46 985	619	0.25	10.8(13)	11.8(12)	396(18)			1.88(12)	2.43(18)	81	18	<1e-6
5.5	250	5500	46 985	619	0.25	10.8(13)	10.8(13)	263(25)	1.77(38)	69.3(46)	1.921(65)	2.49(13)	14	18	0.71
7.5	250	5500	38 400	704	0.20	33.4(20)	35.6(20)	382(17)			1.93(12)	3.35(24)	83	19	<1e-6
7.5	250	5500	38 400	704	0.20	33.4(20)	33.5(20)	263(23)	1.92(49)	71.8(41)	1.964(66)	3.43(18)	20	19	0.38
9.5	150	5500	28 297	1072	0.15	16.8(12)	17.9(12)	299(11)			1.652(83)	6.15(36)	187	24	<1e-6
9.5	150	5500	28 297	1072	0.15	16.8(12)	16.9(12)	142(12)	0.957(84)	57.8(35)	1.785(45)	6.66(26)	23	24	0.55

Table 7.3: Intermediate quantities and results of the afterpulse analysis. For each overvoltage the data is fitted either with one or two exponential decay components. "Prob." is the afterpulse probability. "Correction" is the inverse of the denominator in (7.25) or (7.26). (`tabap.py`)

then we divide by the number of events to compute the afterpulse probability. In Figure 7.22 we show the fit, while in Table 7.3 and Figure 7.23 we summarize the results for all overvoltages. In all the calculations, when necessary, we have kept into account the correlations to propagate the error.

The one component fit has poor quality, so to compute the errors on the correction and on the afterpulse probability, we rescale the covariance matrix of the fitted parameters dividing by the factor χ^2/dof. The two components fit seems good so we do not rescale the errors. See Appendix B for details on this procedure. In Table 7.3, the errors on the fit parameters are not rescaled, while the errors on the parameters entering into the calculation of "Correction" and "Prob." are.

Even if the two models differ, the resulting corrections and thus afterpulse probabilities are compatible with each other for each overvoltage.

In the top left panel of Figure 7.20, which shows the prominence versus delay from the laser pulse of post-trigger pulses, we notice that at short delays, less than 100 ns, there is a dense lump of points around the 1 PE height. These may appear as good candidates for delayed cross talk; however, they are all without exception cases where the laser peak is delayed with the short filter and classified as post-trigger pulse, and the event is still selected as 1 PE laser because we use a longer filter for the selection. This was discussed in subsection 7.3.3. The same goes for the lump at 2 PE, but by

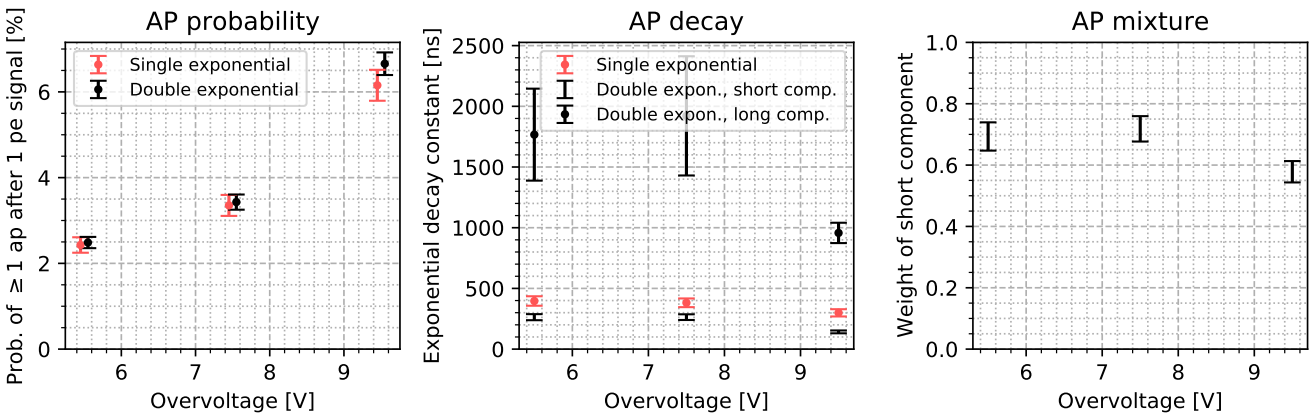


Figure 7.23: Results of the afterpulse analysis. The first panel gives the probability for a 1 PE pulse to generate at least one afterpulse. The second shows together the exponential decay constants for the one and two components models. The third contains the relative weight of the short component for the two components model. (`figapresults.py`)

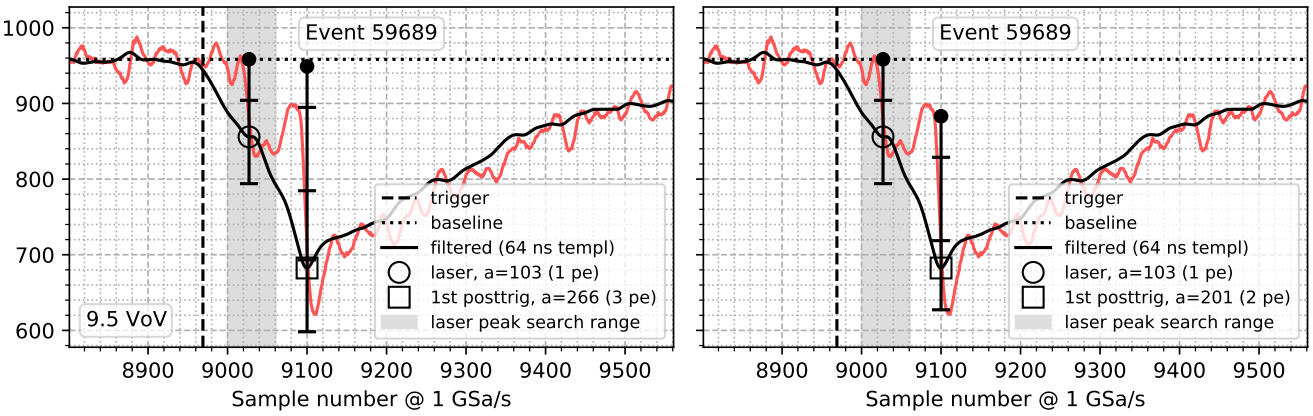


Figure 7.24: An event where the post-trigger pulse appears to have an integer number of PE with normal amplitude (right panel) instead of corrected amplitude (left panel). (`figdectevent.py`)

inspecting the events one by one we found a single event at 9.5 VoV that has a good chance of being DeCT. There are about 30 000 selected events for that overvoltage; supposing to have missed some cases since we inspected by hand, we give this rough estimate: $10/30\,000 = 0.03\,\%$. The event we are talking about is shown in Figure 7.24.

7.6 Direct cross talk

We take the random pulses and afterpulses selected in the previous sections and the laser pulses and fit the DiCT models on the PE distributions. We assign the pulses to the PE bins using the amplitude. For the afterpulses we use the corrected amplitude.

Since the PE bins do not cover the full amplitude range, we put all the uncovered pulses in an overflow bin which is fitted against the survival function of the distribution. Saturated pulses are included.

The bins are fixed a priori independently of the effective data range, so it does happen that there are empty bins for high PE. So for each dataset we aggregate bins to the overflow bin starting from the last one until there

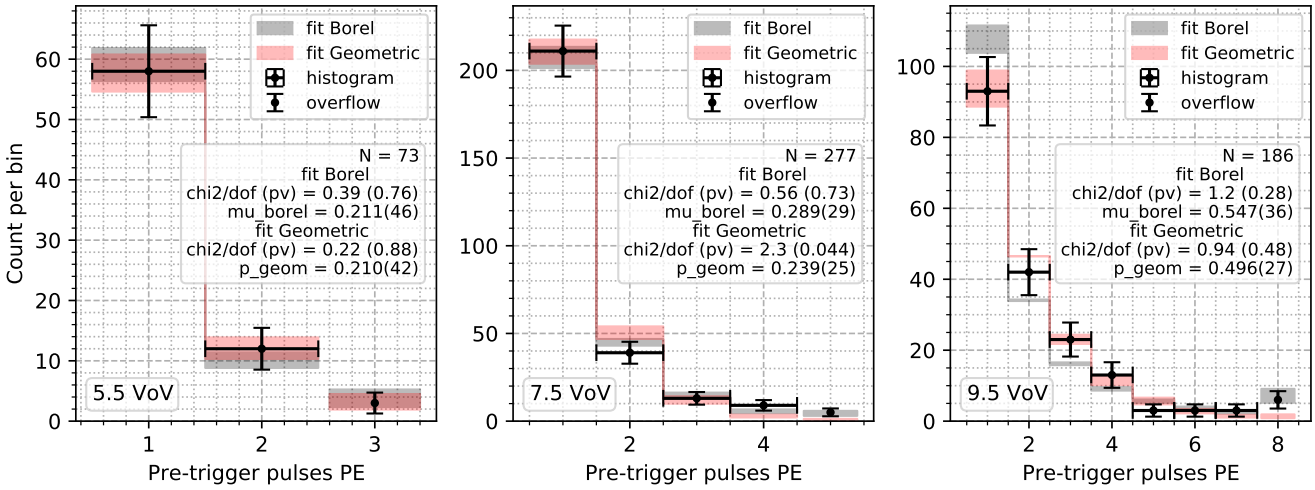


Figure 7.25: Fit of the DiCT models (7.2) and (7.1) on pre-trigger pulses for all overvoltages. The selection of pulses is described in section 7.4. `(figctfitrandom.py)`

are at least 4, sometimes 5, samples in the overflow bin.

The priors for the fit parameters are: uniform in $(0, 1)$ for μ_B and p, and 0 ± 1 for $\log(\mu_P)$.

For the 9.5 VoV laser data the overflow bin seems to be a particular source of data-model discrepancy, so we also fit it without overflow. In this case the bins if necessary are aggregated to the last bin. In the spirit of avoiding arbitrary choices, we do the same at all overvoltages and also for the afterpulses, reporting both results.

The 0 PE probability in the laser model only depends on the initial Poisson mean μ_P. Since in most of the cases the laser fit has poor quality, we are interested in "fixing" the first datapoint, i.e., assuming that the Poisson model is correct and that all the problems come from the DiCT model. As usual we will rescale the fit errors with $\sqrt{\chi^2/\text{dof}}$, so what we want to obtain is that, after the rescaling, the error on the first datapoint is the Poisson error. So we would like to first divide that error by what would turn out to be the $\sqrt{\chi^2/\text{dof}}$ considering the modified error. As an approximation, we instead divide the error by the $\sqrt{\chi^2/\text{dof}}$ obtained with the normal fit.

In Figure 7.25 and 7.26 we show individual fits. In Figure 7.27 we show together the obtained values for the DiCT parameters μ_B (Borel model) and p (Geometric model) for the three categories of pulses. Additionally we also show the same comparison with the DiCT probability, i.e., the probability of having more than one PE, which is $1 - e^{-\mu_B}$ for the Borel model and just p for the Geometric model, and with the average excess PE, i.e., the mean number of PE minus one, which is respectively $1/(1-\mu_B) - 1$ or $1/(1-p) - 1$. Note that to first order these quantities are equal to the parameter.

In Figure 7.28 and 7.29 we compare the results for the laser and afterpulse datasets varying the two options, which are including or not the overflow and fixing or not the 0 PE bin. Finally, in Table 7.4 we list the complete results for all fits. As can be seen in the table, there is a clear separation between poor and good quality fits. In the comparison plots and in the table the errors are shown with the χ^2/dof correction if applied, while in the individual fit plots they are χ^2/dof-less.

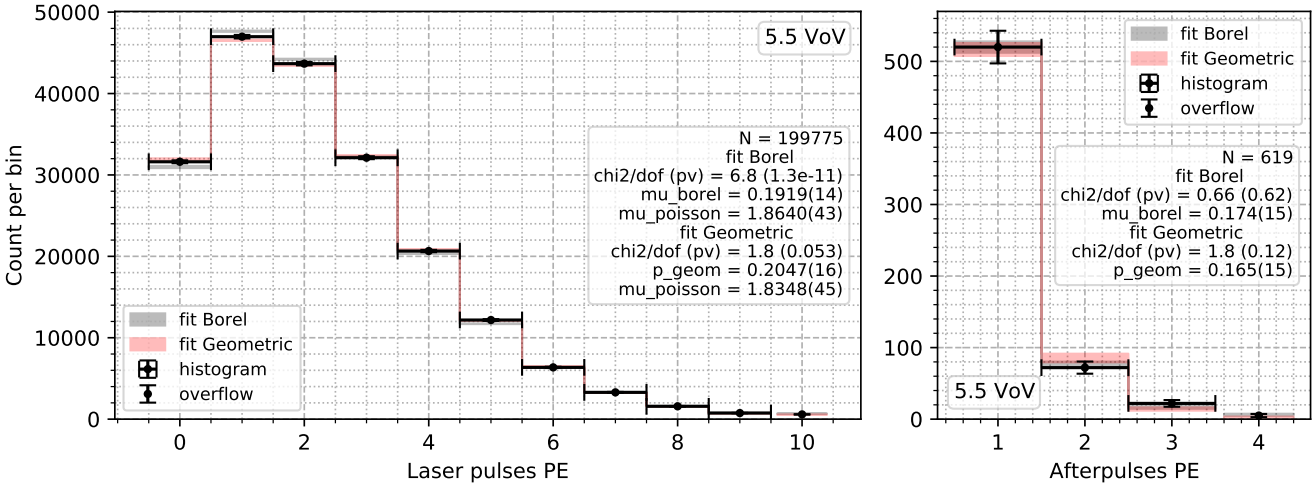


Figure 7.26: Fit of the DiCT models on laser pulses (Equations 7.9 and 7.8) and afterpulses (Equations 7.2 and 7.1) at 5.5 VoV. For all options and overvoltages, see Figures A.14, A.15 and A.16. (`figctfitaplaser.py`)

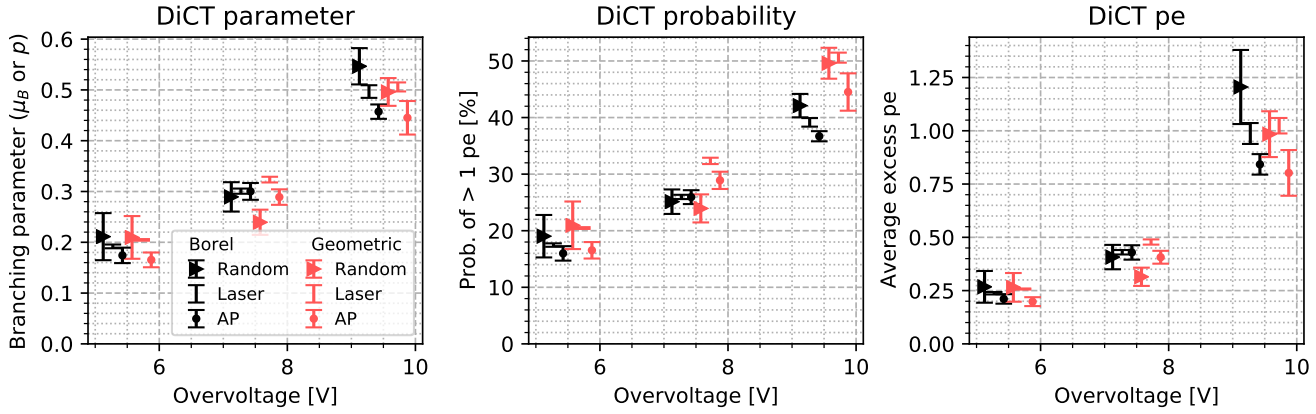


Figure 7.27: Comparison of the DiCT parameters between models and datasets. For the afterpulses and laser pulses, the fits used include the overflow bin. For the laser pulses, the 0 PE bin is *not* fixed. The left panel shows the parameters, the central panel the DiCT probability ($1 - e^{-\mu_B}$ or p) and the right panel the average excess PE ($1/(1-\mu_B) - 1$ or $1/(1-p) - 1$). (`figctresults.py`)

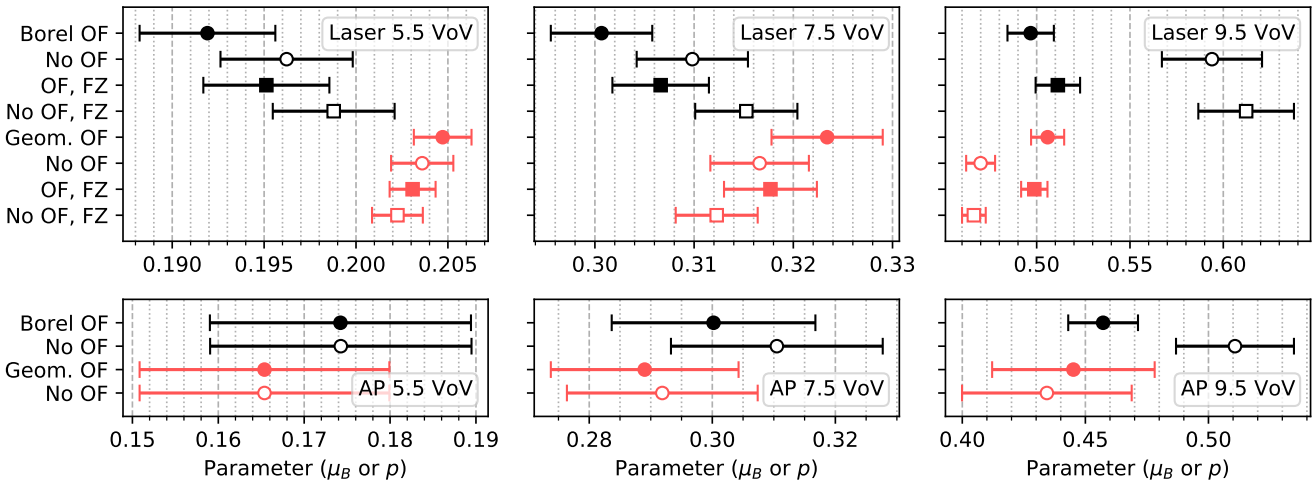


Figure 7.28: Comparison of the DiCT parameters for afterpulses and laser pulses varying the options. 'OF' stands for overflow bin included, while 'FZ' stands for 0 PE bin "fixed". (`figctopt.py`)

Data			Options			Fit parameters					Fit quality		
OV	Pulses	N	OF	FZ	Model	muB	p	muP	Prob. [%]	pe	chi2	dof	pvalue
5.5	Random	73	Yes		Borel	0.21(5)			19(4)	0.27(7)	1	3	0.76
5.5	Random	73	Yes		Geom.		0.21(4)		21(4)	0.27(7)	1	3	0.88
5.5	Laser	199 775	Yes	No	Borel	0.192(4)		1.864(11)	17.46(30)	0.237(6)	75	11	<1e-6
5.5	Laser	199 193	No	No	Borel	0.196(4)		1.860(10)	17.82(30)	0.244(6)	48	10	<1e-6
5.5	Laser	199 775	Yes	Yes	Borel	0.1951(34)		1.848(6)	17.73(28)	0.242(5)	93	11	<1e-6
5.5	Laser	199 193	No	Yes	Borel	0.1988(33)		1.849(6)	18.03(27)	0.248(5)	57	10	<1e-6
5.5	Laser	199 775	Yes	No	Geom.		0.2047(16)	1.835(4)	20.47(16)	0.2574(25)	19	11	0.05
5.5	Laser	199 193	No	No	Geom.		0.2036(17)	1.836(5)	20.36(17)	0.2557(27)	16	10	0.09
5.5	Laser	199 775	Yes	Yes	Geom.		0.2031(13)	1.8417(21)	20.31(13)	0.2548(20)	22	11	0.02
5.5	Laser	199 193	No	Yes	Geom.		0.2023(14)	1.8411(24)	20.23(14)	0.2535(22)	18	10	0.05
5.5	AP	619	Yes		Borel	0.174(15)			16.0(13)	0.211(22)	3	4	0.62
5.5	AP	619	No		Borel	0.174(15)			16.0(13)	0.211(22)	3	4	0.62
5.5	AP	619	Yes		Geom.		0.165(15)		16.5(15)	0.198(21)	7	4	0.12
5.5	AP	619	No		Geom.		0.165(15)		16.5(15)	0.198(21)	7	4	0.12
7.5	Random	277	Yes		Borel	0.289(29)			25.1(22)	0.41(6)	3	5	0.73
7.5	Random	277	Yes		Geom.		0.239(25)		23.9(25)	0.31(4)	11	5	0.04
7.5	Laser	199 831	Yes	No	Borel	0.301(5)		2.041(18)	26.0(4)	0.430(10)	179	12	<1e-6
7.5	Laser	197 778	No	No	Borel	0.310(6)		2.032(15)	26.6(4)	0.449(12)	114	11	<1e-6
7.5	Laser	199 831	Yes	Yes	Borel	0.307(5)		2.006(7)	26.4(4)	0.442(10)	242	12	<1e-6
7.5	Laser	197 778	No	Yes	Borel	0.315(5)		2.009(7)	27.0(4)	0.460(11)	142	11	<1e-6
7.5	Laser	199 831	Yes	No	Geom.		0.323(6)	1.971(19)	32.3(6)	0.478(12)	177	12	<1e-6
7.5	Laser	197 778	No	No	Geom.		0.317(5)	1.978(15)	31.7(5)	0.463(11)	98	11	<1e-6
7.5	Laser	199 831	Yes	Yes	Geom.		0.318(5)	1.999(7)	31.8(5)	0.466(10)	213	12	<1e-6
7.5	Laser	197 778	No	Yes	Geom.		0.312(4)	1.996(6)	31.2(4)	0.454(9)	115	11	<1e-6
7.5	AP	704	Yes		Borel	0.300(17)			25.9(12)	0.429(34)	9	7	0.26
7.5	AP	703	No		Borel	0.310(17)			26.7(13)	0.45(4)	4	6	0.74
7.5	AP	704	Yes		Geom.		0.289(15)		28.9(15)	0.406(30)	11	7	0.14
7.5	AP	703	No		Geom.		0.292(16)		29.2(16)	0.412(31)	9	6	0.16
9.5	Random	186	Yes		Borel	0.55(4)			42.1(21)	1.21(17)	10	8	0.28
9.5	Random	186	Yes		Geom.		0.496(27)		49.6(27)	0.98(11)	7	8	0.48
9.5	Laser	199 701	Yes	No	Borel	0.497(12)		2.15(4)	39.2(8)	0.99(5)	637	9	<1e-6
9.5	Laser	167 134	No	No	Borel	0.594(27)		2.205(31)	44.8(15)	1.46(16)	186	8	<1e-6
9.5	Laser	199 701	Yes	Yes	Borel	0.511(12)		2.072(7)	40.0(7)	1.05(5)	890	9	<1e-6
9.5	Laser	167 134	No	Yes	Borel	0.612(26)		2.190(33)	45.8(14)	1.58(17)	218	8	<1e-6
9.5	Laser	199 701	Yes	No	Geom.		0.506(9)	2.03(4)	50.6(9)	1.02(4)	417	9	<1e-6
9.5	Laser	167 134	No	No	Geom.		0.470(8)	2.016(17)	47.0(8)	0.887(28)	85	8	<1e-6
9.5	Laser	199 701	Yes	Yes	Geom.		0.499(7)	2.069(5)	49.9(7)	0.995(28)	486	9	<1e-6
9.5	Laser	167 134	No	Yes	Geom.		0.466(6)	2.028(7)	46.6(6)	0.874(22)	91	8	<1e-6
9.5	AP	1072	Yes		Borel	0.457(14)			36.7(9)	0.84(5)	16	8	0.04
9.5	AP	1060	No		Borel	0.511(24)			40.0(14)	1.04(10)	7	7	0.46
9.5	AP	1072	Yes		Geom.		0.445(33)		44.5(33)	0.80(11)	52	8	<1e-6
9.5	AP	1060	No		Geom.		0.434(34)		43.4(34)	0.77(11)	46	7	<1e-6

Table 7.4: List of all the DiCT fits. For fits with p-value less than 1e–6, the errors on the fit parameters and on the derived quantities are rescaled with $\sqrt{\chi^2/\mathrm{dof}}$. 'OF' stands for overflow bin included, while 'FZ' stands for 0 PE bin "fixed". "Prob." is the probability of having more than 1 PE under the model, while "pe" is the average number of PE minus 1. (`tabct.py`)

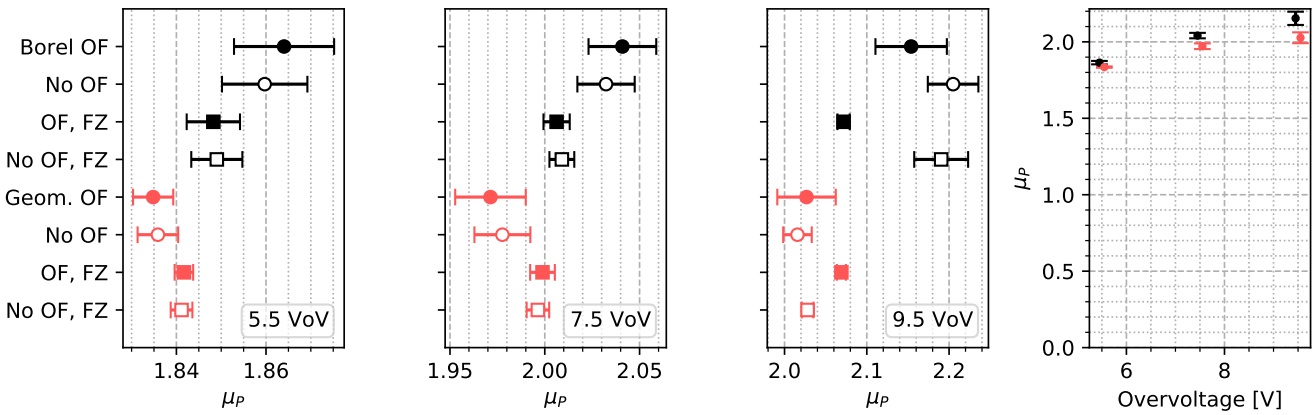


Figure 7.29: Initial Poisson mean μ_P for the laser pulses fit, compared varying the options. 'OF' stands for overflow bin included, while 'FZ' stands for 0 PE bin "fixed". The last panel shows the 'OF' values vs. overvoltage. (`figctpoisson.py`)

7.7 Discussion

We analyzed the additional pulses noise of the LFoundry Tile 21. The key results are reported in Table 7.2, Figure 7.23 and Figure 7.27.

For the random pulses rate we are quite confident that there must be an additional source of random pulses other than the dark count rate, so our estimates are an upper bound on DCR, and are anyway within the DarkSide20k requirements.

The afterpulse probability for 1 PE pulses goes from $\approx 2.5\,\%$ at 5.5 VoV to $\approx 6.5\,\%$ at 9.5 VoV. The temporal distribution is well explained by two exponential decay components but not by one, where the two decay constants are $\approx 270\,\text{ns}$ and $\approx 2\,\mu\text{s}$ at 5.5 VoV and 7.5 VoV, and $\approx 150\,\text{ns}$ and $\approx 1\,\mu\text{s}$ at 9.5 VoV. The amount relative to the total of the short component is $\approx 65\,\%$.

The temporal distribution was fit with delays respectively above 250 ns, 250 ns, 150 ns for the three overvoltages, to be compared with the recharge time $\approx 300\,\text{ns}$. In computing the AP probability, we assumed that the distribution would continue with exponentials up to zero delay. This is likely not accurate but there is not a consensus in the literature on how the distribution is suppressed towards zero. Quite likely our assumption gives an upper bound, while a lower bound is obtained by completely neglecting the excluded delays. The distribution correction factors in Table 7.3 are less than 2, so the probabilities are at least half of those we report. Note that fitting with a suppression term included would likely change the resulting decay constants.

We guess that for afterpulses at short delays it would be more useful to know the probability weighted with their relative amplitude since it goes to zero at zero delay. However we are not confident in our understanding of how the amplitude goes to zero. We expected it to follow the recharge factor, which would mean that by summing an exponential to the amplitude it should go to 1 PE at all delays, because $(1 - e^{-t}) + e^{-t} = 1$. However, we rectified it by summing the unfiltered pulse shape with *peak amplitude* normalized to 1 PE, which means we are adding a smaller exponential. And looking at Figure A.12 it still seems to be biased upward.

We can not exclude that there could be a problem in our procedure for computing the amplitude due to the short filter. In hindsight, we should have followed a different procedure. After identifying the peaks and measuring their position with the short filter, we should have computed the amplitude

with the long filter. Or maybe a different combination to determine the position—but the point is using the long filter output for the amplitude, because it is less noisy and because having determined the position in another way we are not taking a peak in the filter output and thus we avoid an upward bias. We noticed that this method was already presented in [Kri20].

With the DiCT we get consistent results between the three samples: random pulses, laser pulses, afterpulses. The probability goes from $\approx 20\,\%$ to $\approx 45\,\%$.

The results obtained with the Borel model are not always compatible with the ones from the Geometric model, but we can not determine which one is better. Looking at Table 7.4, the poor quality fits are: all the laser fits apart from 5.5 VoV Geometric, and the afterpulse Geometric. Even when bad, the Geometric model has lower χ^2 on the laser fits. But looking at Figure A.14 and A.15, the Geometric model is visibly getting wrong a feature of the distribution: the relative heights of the 0, 1, 2 PE bins. Another difference between the models is how μ_P varies with overvoltage in Figure 7.29. "Fixing" the 0 PE bin the values become at the same time more precise and compatible. If you trust the Poisson model, those are the values to look at.

Including or excluding the overflow, and fixing or not the 0 PE bin, do not have a substantial effect on the result, apart from the overflow at 9.5 VoV. Two concurrent explanations come to our mind: that at high DiCT probability the discrepancy from the model in the tail of the distribution is more visible, and that having defined less PE bins at 9.5 VoV due to saturation—see Figure A.10—it is just the wider overflow bin that is making the difference.

Overall, looking at the summary in [Sav18, tab. 3.1 p. 62], which is given at 5 VoV for FBK Tiles, and comparing it to our values at 5.5 VoV, the DiCT probability is a bit smaller than ours, while the afterpulse probability is always much higher. We are not sure, but we think that the values in that table just take into account a truncated afterpulse distribution, so they should be compared to roughly half of our estimates. Considering the slightly higher overvoltage, the DiCT would probably be actually the same. The DeCT is much higher, but we do not trust our estimate. So we can say that this LFoundry Tile has the same DiCT, 4 to 30 times less AP, and probably and possibly much less DeCT, than those FBK Tiles.

If this work was to be carried further, we would model and fit also the events with two post-trigger pulses, and with more than 1 PE laser pulses, since they have the potential to break our simple assumptions on the generation of afterpulses, which were actually never tested in this analysis. Moreover, we would try to investigate models for the amplitude of afterpulses and intermediate alternatives between the Borel and Geometric models. Mind that these DiCT models are probably already accurate enough for the necessities of DarkSide20k, although we do not know for sure. We are less confident in the adequacy of the afterpulse model.

Another unexpected feature worth investigating is the nonlinearity of the amplitude respect to the overvoltage, which was given for granted in the literature; but it could well be that if we went through an electric model of the SiPM with attention, for example in [Sav18, ch. 3], we would find the answer already laid out somewhere.

But there is another possibility we already discussed in subsection 7.3.1. In general the 9.5 VoV data feels like an outlier. Computing the peak amplitude to overvoltage ratio of the templates from Figure 7.7, in order by overvoltage we get 12.7, 12.7, 14.2. With the afterpulse probability from Table 7.3 it is 0.45, 0.45, 0.69. For the DiCT probability, which we take

from default options Borel laser fits from Table 7.4, the ratios are 3.2, 3.5, 4.1. Either the SiPM enters a nonlinear regime somewhere between 7.5 VoV and 9.5 VoV, or the 9.5 VoV data is mislabeled. The latter case is a concrete possibility that should be checked.

An issue we did not flesh out is the quality of the template. We are using simple averaging with a selection of 1 PE pulses using charge integration. Our sample surely contains short afterpulses which are not big enough to push the charge over the upper selection boundary. This changes the shape of the template. How large is this effect? Since afterpulses increase with overvoltage, we can get an idea from Figure 7.18 where the shapes at different overvoltages are compared. If afterpulses are what is driving the difference between the templates, they seem small enough to not have a substantial influence on the results. It would be interesting to compute the template keeping only the lower half of the distribution.

From [Sav18] and [Nag+14] we guess that the data for this kind of analysis is normally collected by reading continuously the SiPM output and triggering on dark count pulses, running a peak finder online, instead of shooting laser pulses and saving complete waveforms synced with the laser trigger. We have encountered no obstacle in using this other kind of dataset, and we think that for an in-depth analysis having the full waveforms recorded is convenient for studying in detail filtering and peak finding procedures. The size of our data is 6 GB per overvoltage in uncompressed 16 bit `wav` files, so there are no disk space problems for a single analysis.

The other principal approach to correlated noise measurement is fitting the amplitude distribution with charge integration, keeping into account DiCT and afterpulses all into a single formula. This is done thoroughly in [Chm+17]. However to use this approach with confidence one must be sure of the underlying model. For example, the referenced article states that the afterpulse distribution is unexpected [Chm+17, app. A]. They derive the expected distribution assuming one exponential component and amplitude reduction with the recharge factor, which as we have seen are probably inaccurate assumptions.

In DarkSide20k this situation arises with the VETO system, which employs simpler electronics than the time projection chamber (TPC). To check these detectors after installation it would be necessary to employ a charge integration method without recorded waveforms. We think that an analysis of this kind should be carried on a Tile which was previously analyzed pulse-by-pulse, such that the validity of the models can be assessed. For example, if our analysis is deemed sufficient, Tile 21 could be reanalyzed following [Chm+17].

Another application of a model-sensitive analysis is for defining the correlated noise simulation in the DarkSide20k software. The current version of the SiPM simulation is online at `https://gitlab.in2p3.fr/darkside/pyreco/-/blob/a37a0e40ee2112ece28a28e981b67db6cb69085b/elec/sipm.py` (access restricted). By reading all the code, we are quite sure that the simulation implemented almost matches the model we described in section 7.1: Borel distribution for DiCT, resetting afterpulses, afterpulse amplitude reshaped with the recharge factor, recursive generation of noise with the same distributions applied at each step.

The only difference is that the temporal delay of afterpulses is drawn in this way: first, a variable t_{inv} is drawn from an exponential with scale $1/\tau$, the delay is then $t = 1/t_{\text{inv}}$. Transforming the probability density, the distribution of t is $e^{-1/t}/t^2$ ($\tau = 1$). This is suppressed for $t \to 0$, but it also goes down as $1/t^2$ for $t \to \infty$, like the Cauchy. This is an unusual feature.

This distribution does not even have a mean. Physically it would imply that the distribution of the trapping energy levels goes arbitrarily deep. We presume that it is intended just as a quick way of sampling a zero-suppressed distribution in a work in progress code. Anyway, we take the occasion to notice that the model used to fit the data and measure the parameters must be the same used by the simulation which is fed those parameters.

Chapter 8

Conclusions

We studied the effect of electrical noise on the identification and reconstruction of output pulses in DarkSide20k silicon photomultipliers (SiPM) Tiles in chapter 4, 5 and 6, and measured the correlated noise of a Tile in chapter 7. In the following paragraphs we summarize and comment the results from all chapters.

The main takeaway from chapter 4 is in Figure 4.9. The maximum signal to noise ratio (SNR) that can be reached with filtering on that data is 19, shown on the left panel with the cross correlation filter and precise baseline calculation. However the constraints of the available resources on the digitizers may turn out to impose a moving average filter with imprecise baseline, lowering the SNR to 13, as shown in the central panel. So the SNR at the digitizer stage can be as low as 65 % of the optimal one.

In the DarkSide collaboration it was proposed to use an "autoregressive moving average" (ARMA) filter to implement a long cross correlation filter in the digitizers. The ARMA combines linearly two filters: 1) a cross correlation with a template only for the peak of the pulse, and 2) an autoregressive filter, run on the *temporally reversed* waveform. This reproduces a cross correlation with the peak template plus an exponentially decaying tail. Although the asymptotical complexity of the algorithm is excellent on paper, the actual amount of logic resources necessary for the implementation on FPGA may exceed the available ones, mainly due to the necessity of keeping a buffer for the temporal reversal, and to glue appropriately the reversed temporal slices with dynamical boundary conditions or partially overlapping slices. From Figure 4.9 it is clear that a simple moving average can achieve performances comparable to a cross correlation filter, thus it may not be necessary to devise a way to implement the cross correlation filter on the digitizers.

In chapter 5 there are various things to learn:

- The temporal resolution diverges below a certain SNR, giving a somewhat rigid bound on the working point, and the behavior at low SNR heavily depends on the noise spectrum, Figure 5.7. The DarkSide20k simulation currently implements only white noise.
- For what concerns temporal resolution, it is sufficient to extract 1 µs of waveform per pulse from the digitizers, Figure 5.9.
- Downsampling from 125 MSa/s to 62.5 MSa/s does not deteriorate the temporal resolution, Figure 5.10 and Table 5.1. If a compromise with the data rate is necessary, 31.2 MSa/s might still be good enough.
- Upsampling is not necessary, Figure 5.7.

The temporal resolution required by the specifications of DarkSide20k is 10 ns [Aal+18, p. 30]. A worse resolution would decrease the selective power

of pulse shape discrimination (PSD). With the noise spectrum of the Proto0 prototype, we find that the temporal resolution reaches 10 ns at pre-filter SNR 2.6, to be compared with the SNR observed in the Proto0 data of at least 3.3, see Figure 5.7.

Chapter 6 has its own summary in subsection 6.2.4. In general the formula for the fake rate is sufficiently precise, but there is an exception with higher rate than expected that should be investigated better, see Figure 6.4. With the tested filter, which matches the one mentioned above for the SNR = 13 figure, the fake rate is 10 cps with the threshold set at 5 filtered noise standard deviations (including baseline subtraction), to be compared to the maximum total primary noise rate required, 250 cps/PDM [Aal+18, p. 30].

The chapter on correlated noise contains more material, so it includes a rather long discussion of the results in section 7.7. To summarize further:

- Compared to the Fondazione Bruno Kessler (FBK) Tiles, the analyzed LFoundry Tile has similar direct cross talk (DiCT) and much less afterpulsing (AP).
- The afterpulse models should be studied and validated better.
- It should be determined whether Tile 21 enters a nonlinear regime between 7.5 VoV and 9.5 VoV, or if the 9.5 VoV data is mislabeled.
- Tile 21 should be analyzed with charge spectrum methods and the results compared to ours.
- The SiPM simulation must be consistent with the model used in the analysis of real data.

We put upper bounds on the dark count rate (DCR), respectively 50 cps, 170 cps, and 120 cps at overvoltages 5.5 V, 7.5 V, and 9.5 V, to be compared with the required upper limit 250 cps. We give upper bounds for AP probabilities of 2.5 %, 3.5 % and 6.5 %, and estimate DiCT probabilities 20 %, 30 % and 50 %. These are the probabilities of said noises being generated by a 1 PE pulse. The DarkSide20k specifications require the sum of correlated noise probabilities to be less than 60 % [Aal+18, p. 30] in order to avoid performance degradation, mainly dynamic range reduction on ionization signals, where there is a lot of pile-up. Furthermore, some safety margin should be considered for the aging of the SiPMs, which is not well studied, in case the correlated noise probability turned out to increase with time. Recall that, for a correlated noise probability p, the ratio of total pulses to primary pulses is $1/(1-p)$, which diverges quite fast as p goes to 1. [Gol+19, p. 8] mention that in practice the maximum manageable probability is 50 %.

Appendix A

Additional figures for chapter 7

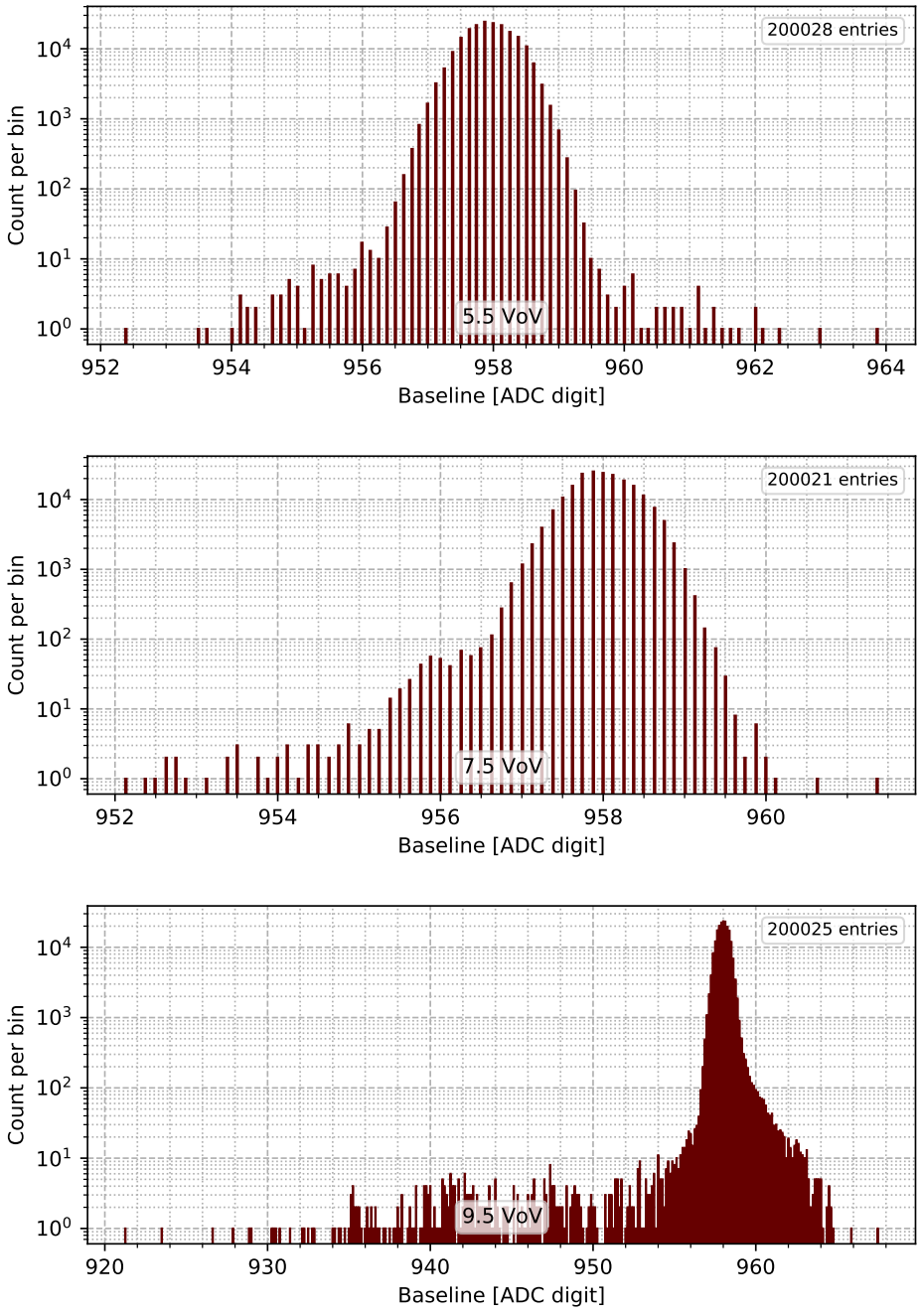


Figure A.1: Distribution of the waveform baseline measured in the pre-trigger region of the events. (`figbaseline2.py`)

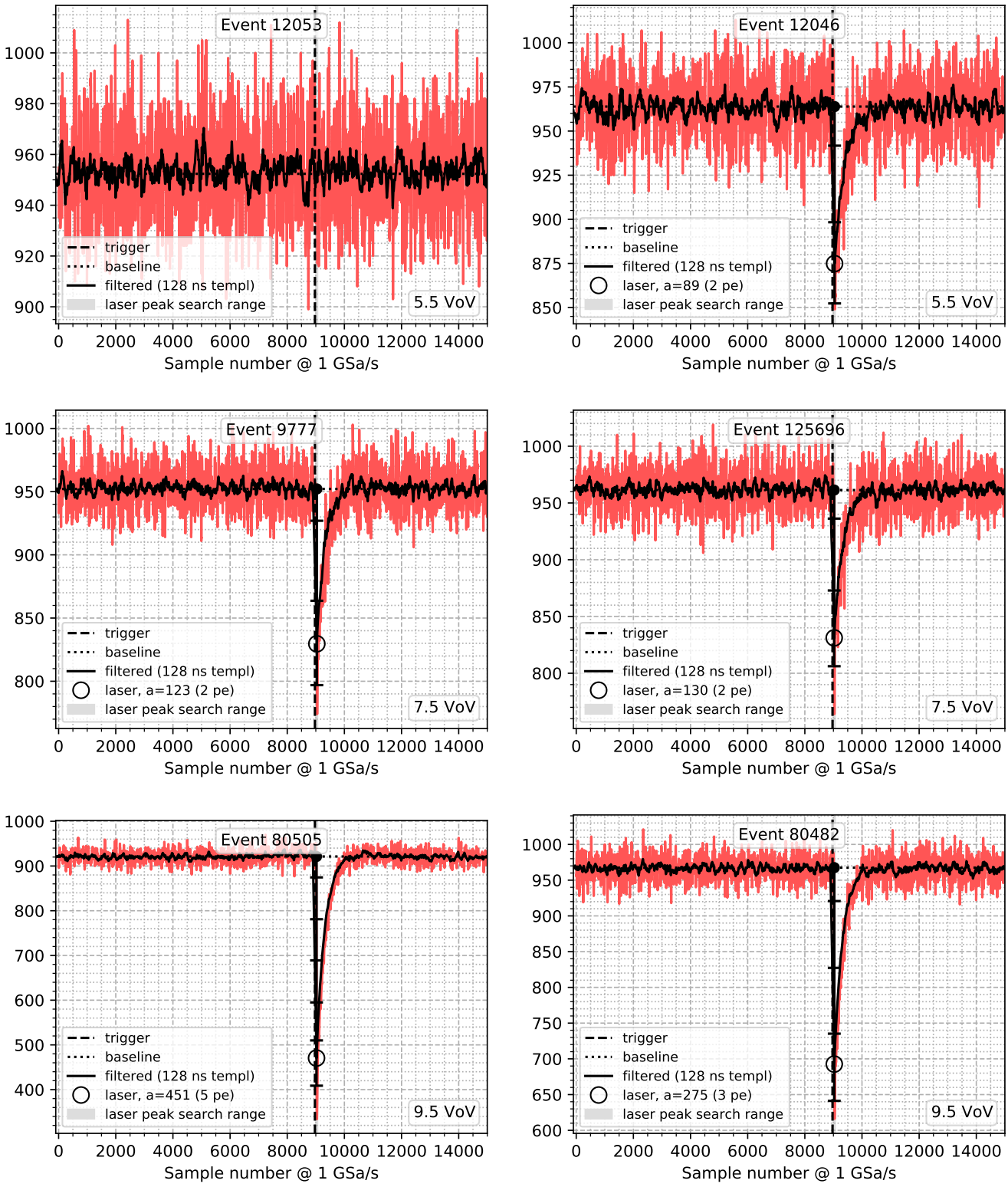


Figure A.2: The events with the lowest and highest baseline. (`figbsoutlier2.py`)

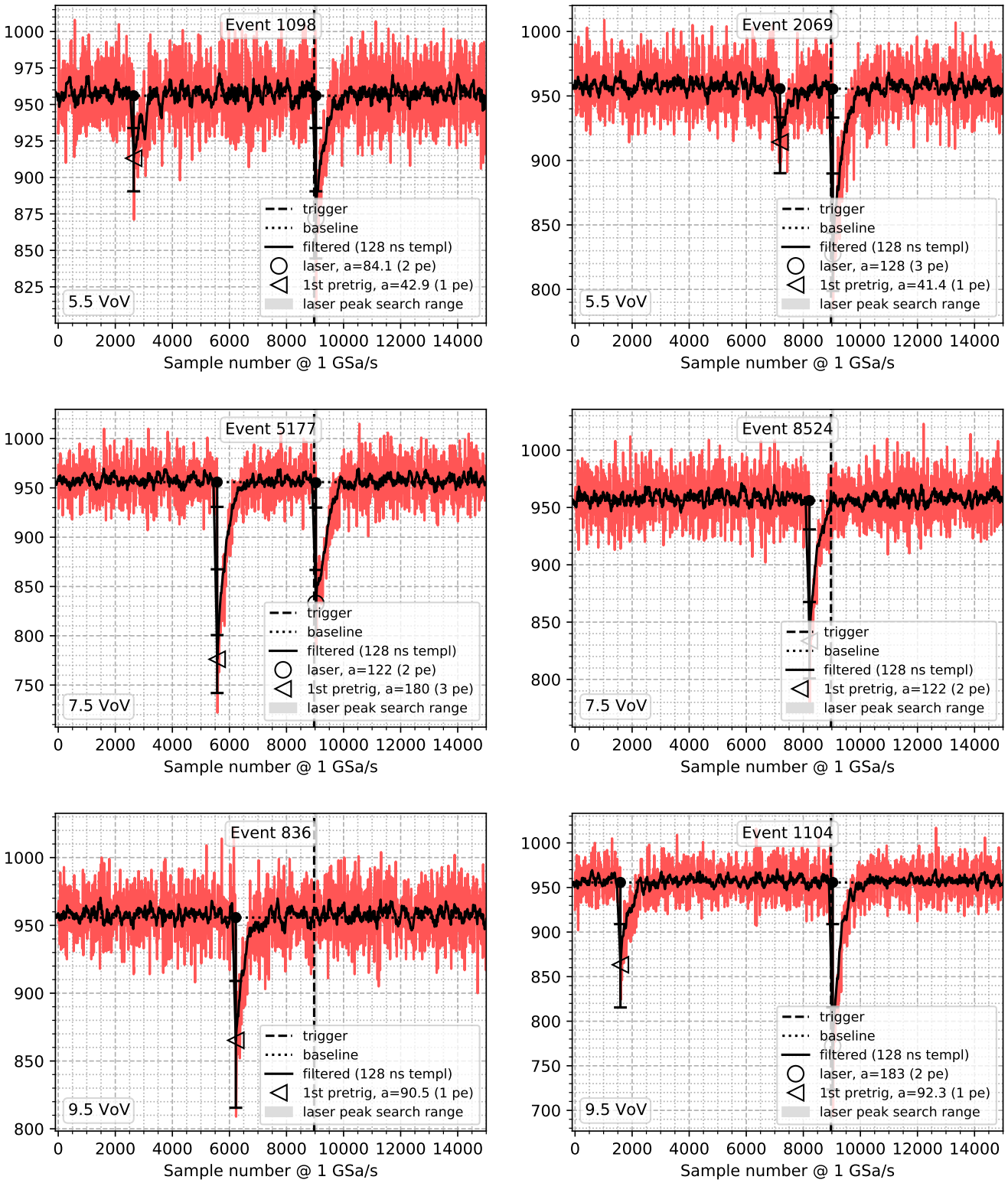


Figure A.3: A pair of events from the low baseline tail (baseline between 955 and 956). (`figbstail2.py`)

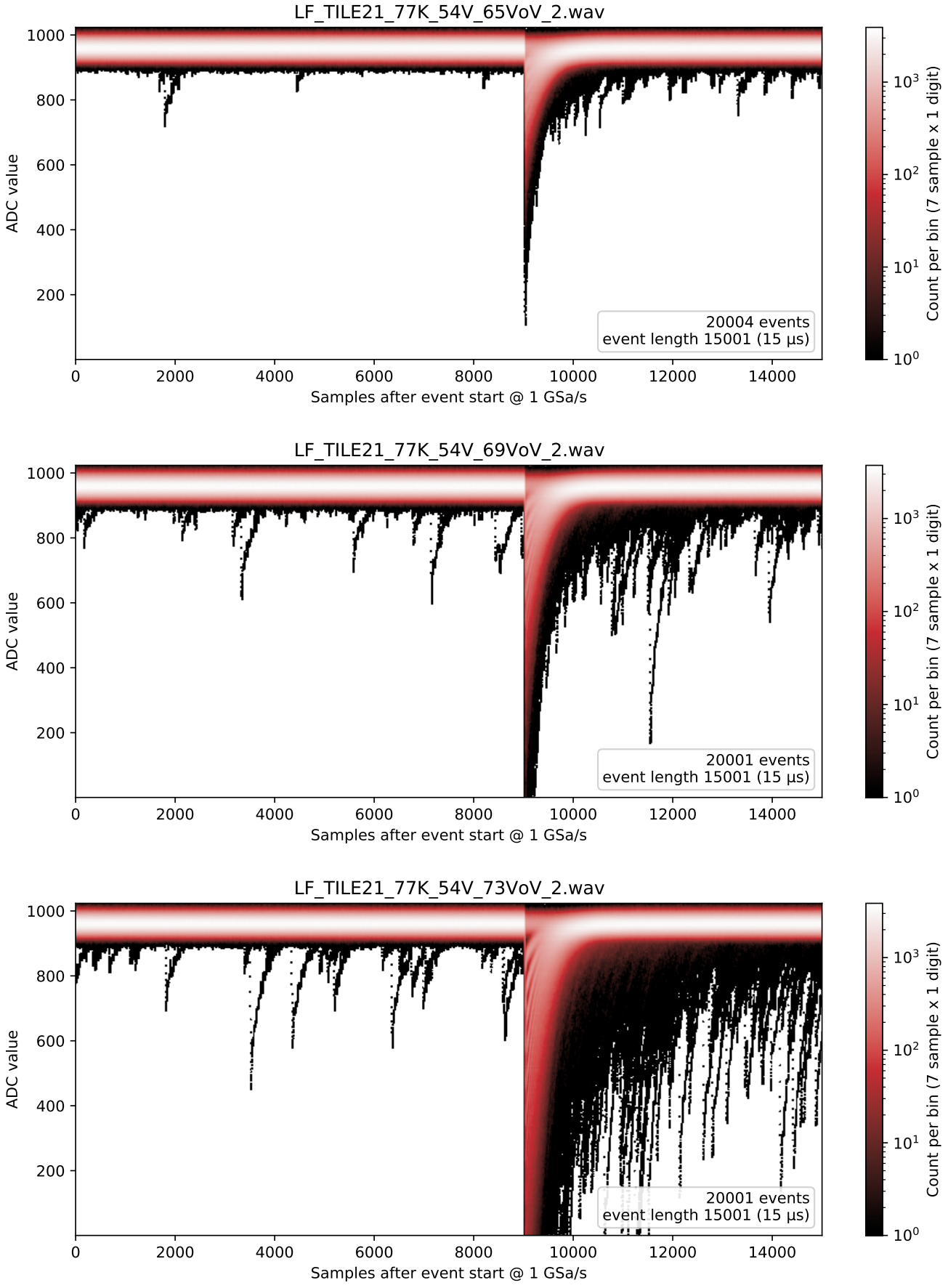


Figure A.4: Time-value histograms of LFoundry Tile 21 in LNGS laser data at overvoltages 5.5 V, 7.5 V, and 9.5 V. (`fighist2dtile21.py`)

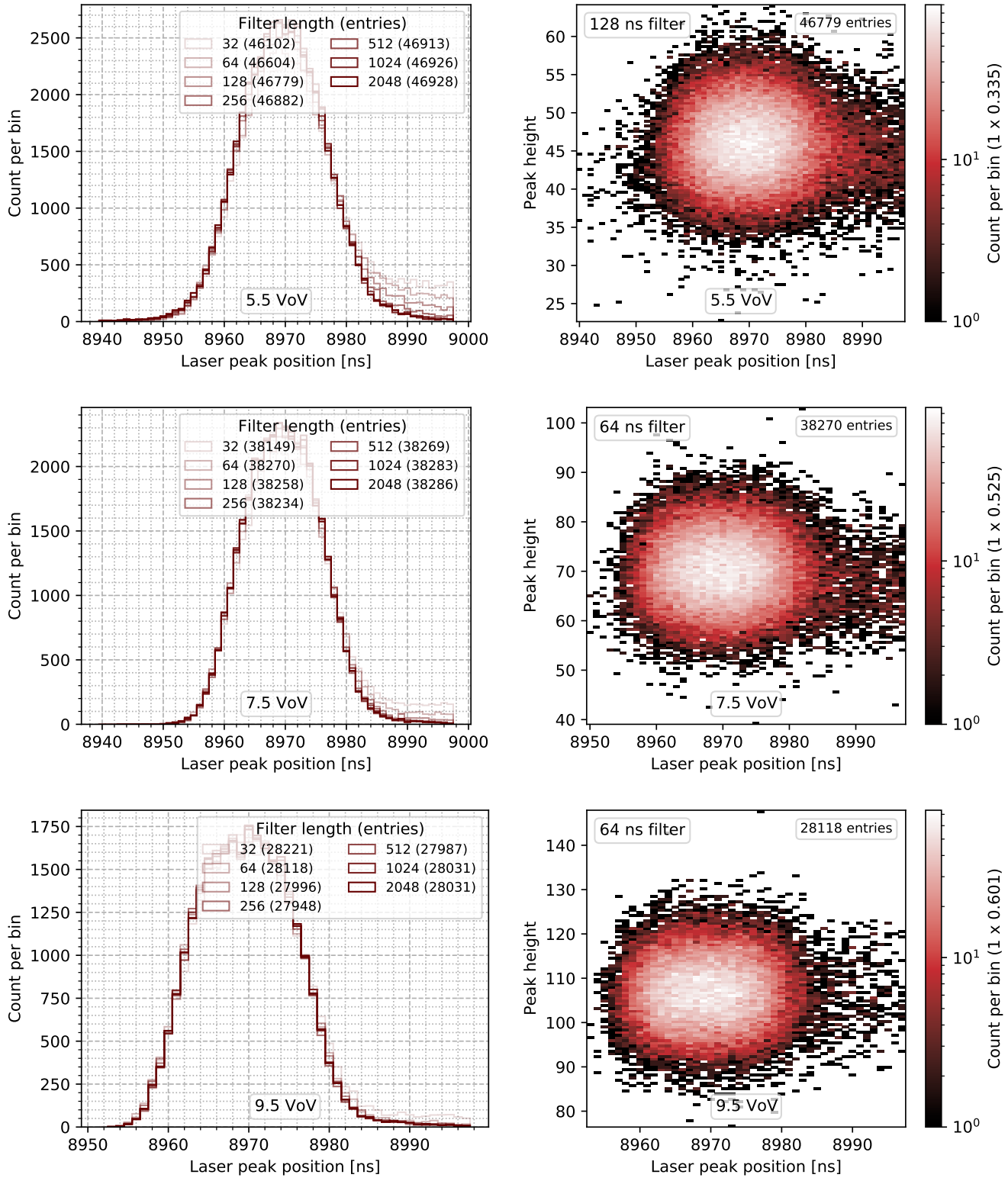


Figure A.5: In each row, left panel: histogram of the laser peak position, only for 1 PE peaks, for all filter lengths. The zero of the scale is such that the expected position is 8969. Right panel: 2D histogram of the laser peak position and the amplitude, for the same selection of peaks, but only with the principal filter length used in the rest of the analysis. (`figlaserpos2.py`)

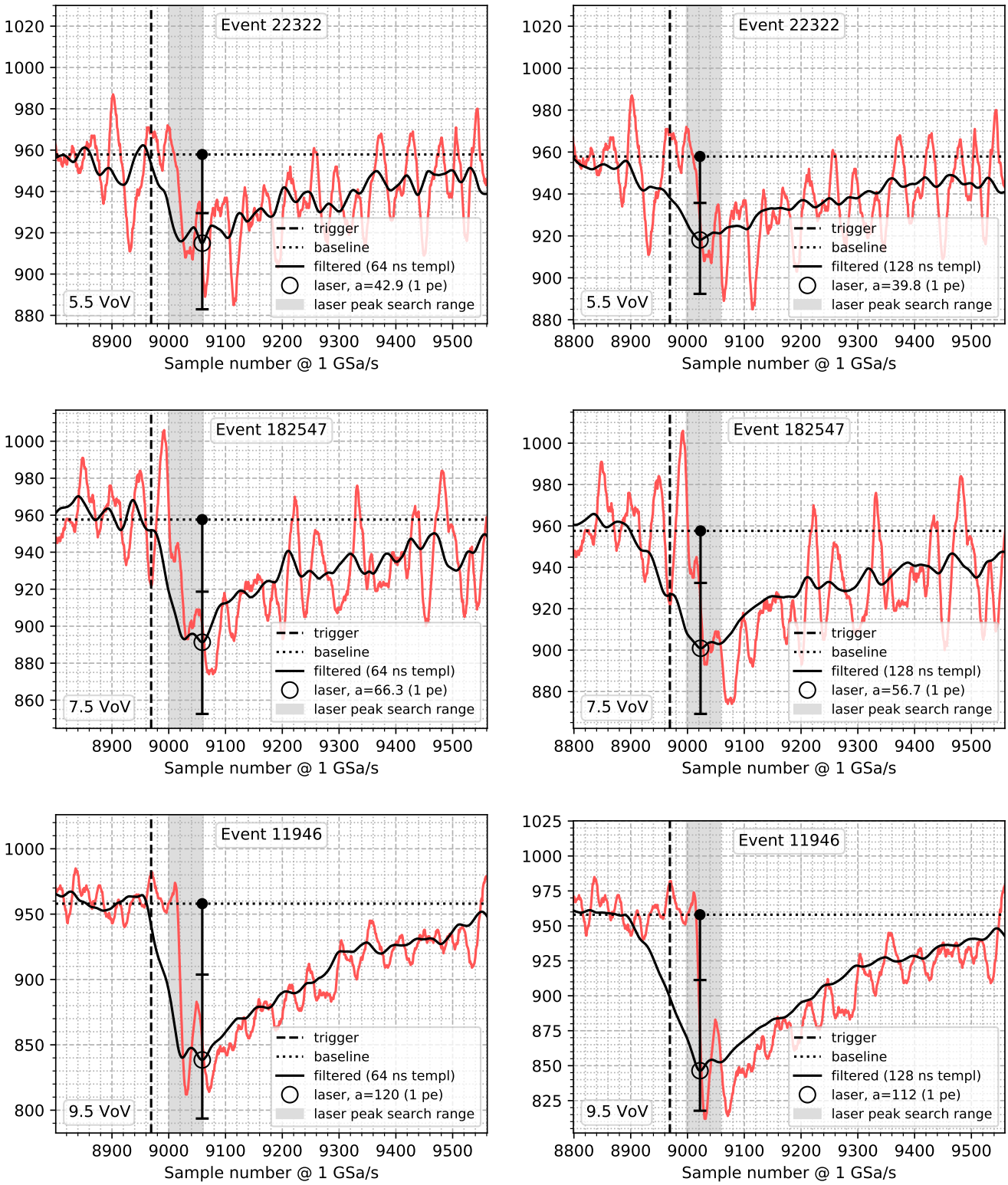


Figure A.6: In each row, left panel: an event with unusually late laser peak position for 1 PE peaks with filter length 64 ns. Right panel: the same event with filter length 128 ns. (`figlptail2.py`)

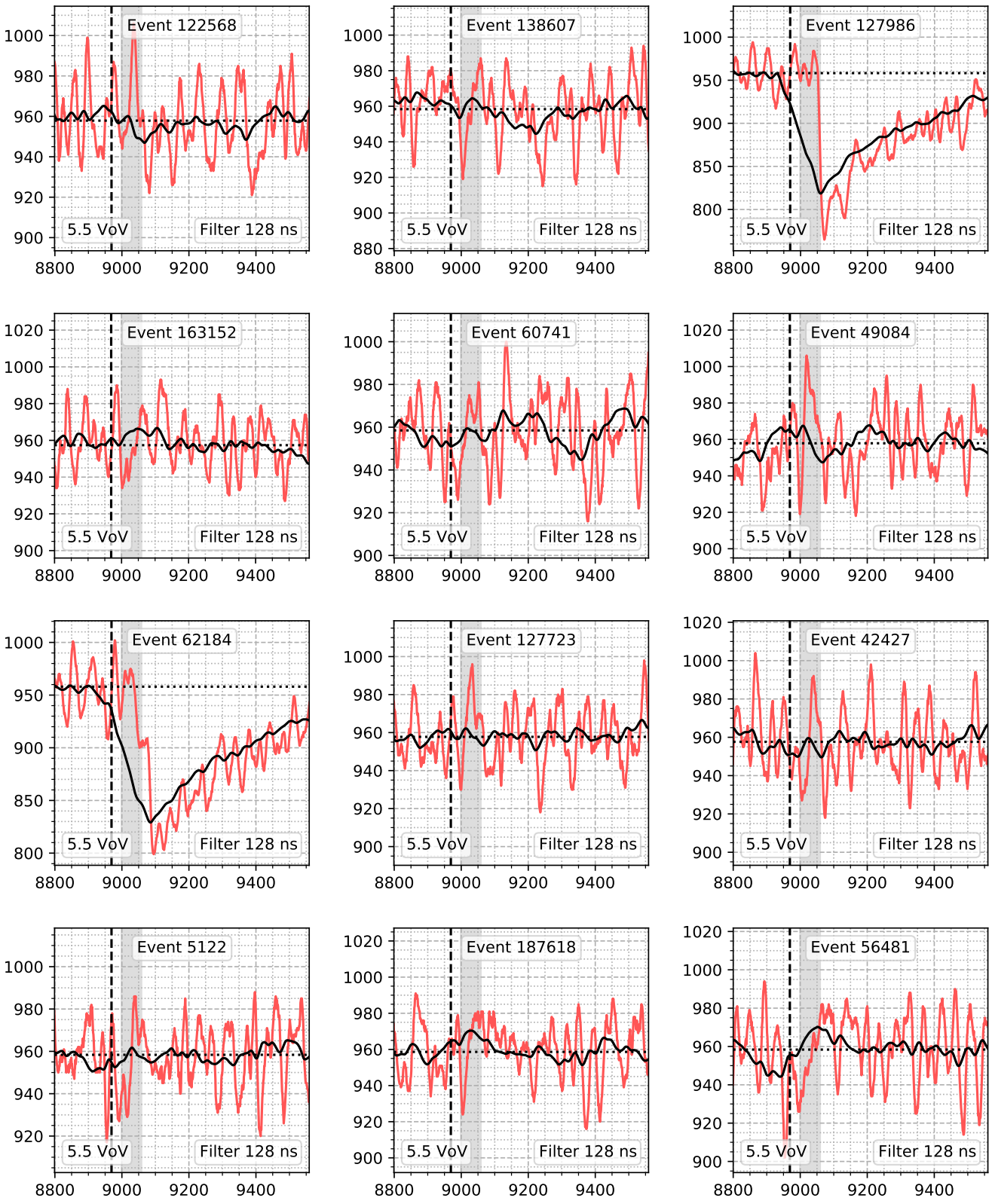


Figure A.7: Twelve randomly selected events at 5.5 VoV where with all filter lengths there is no local minimum in the laser peak search range. (`figverymissing.py`)

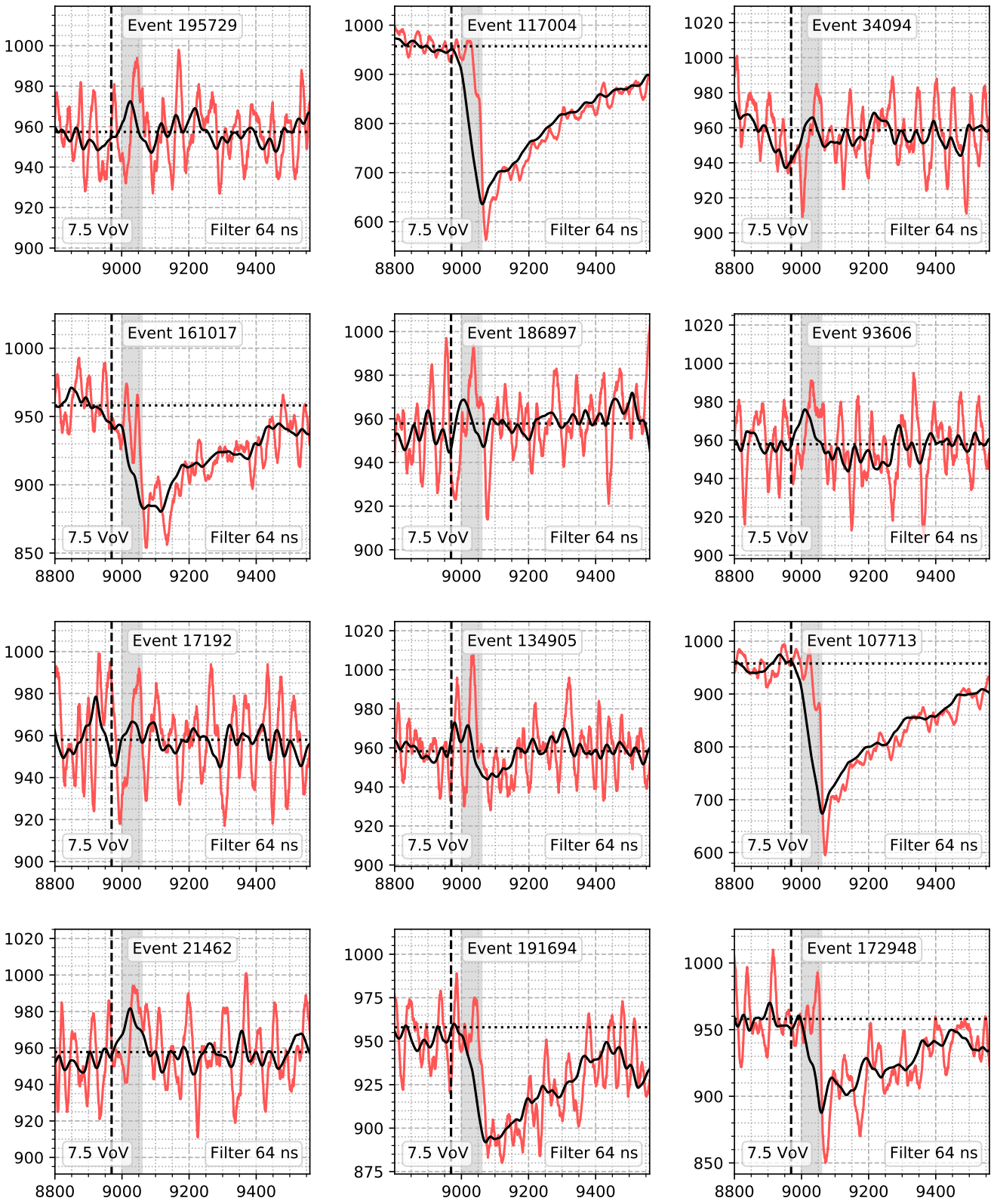


Figure A.8: Twelve randomly selected events at 7.5 VoV where with all filter lengths there is no local minimum in the laser peak search range. (`figverymissing.py`)

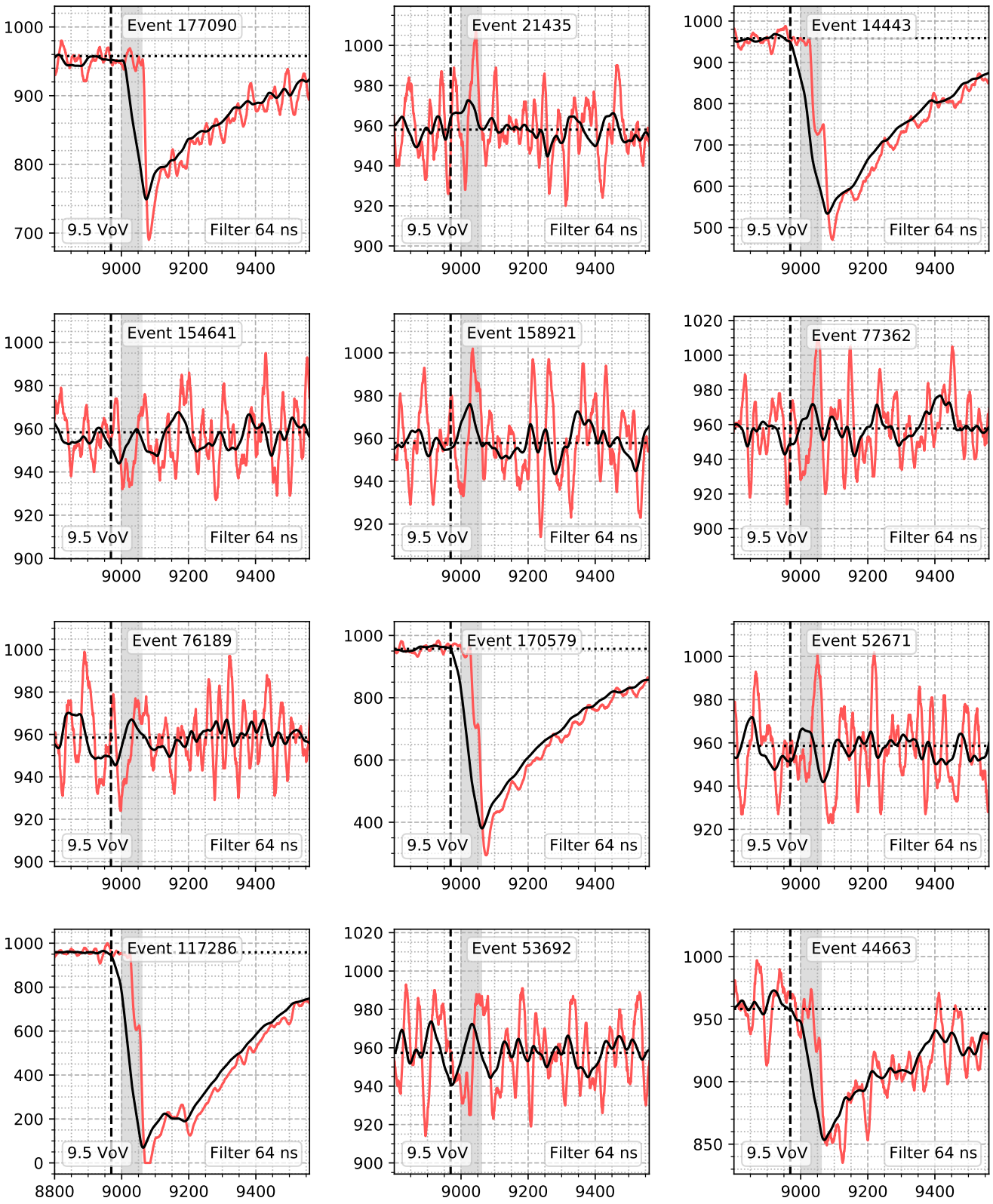


Figure A.9: Twelve randomly selected events at 9.5 VoV where with all filter lengths there is no local minimum in the laser peak search range. (`figverymissing.py`)

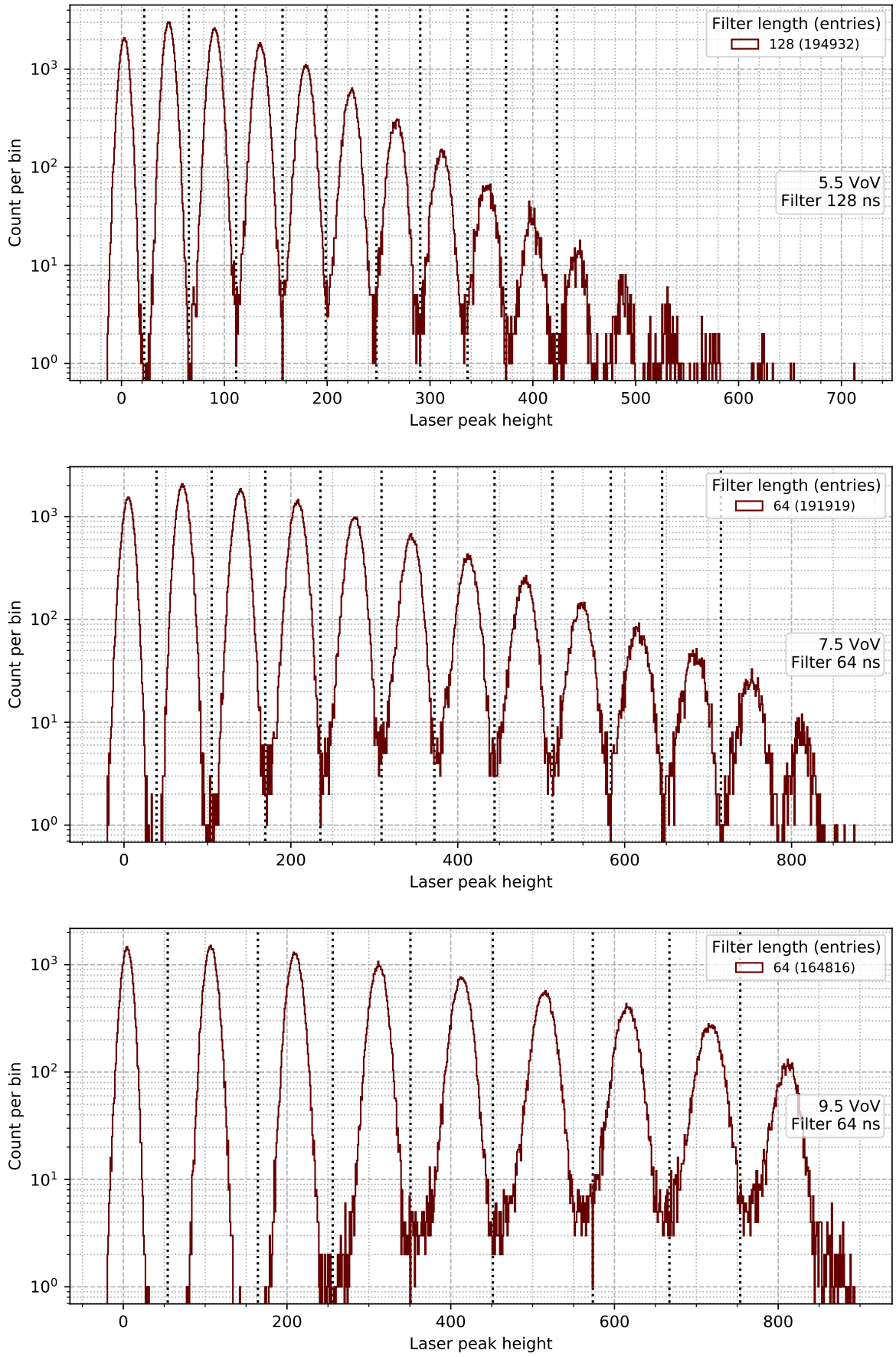


Figure A.10: Histogram of the laser peak height, with a selection to avoid biased heights. The dotted lines are the boundaries of the PE bins. (`figpe2.py`)

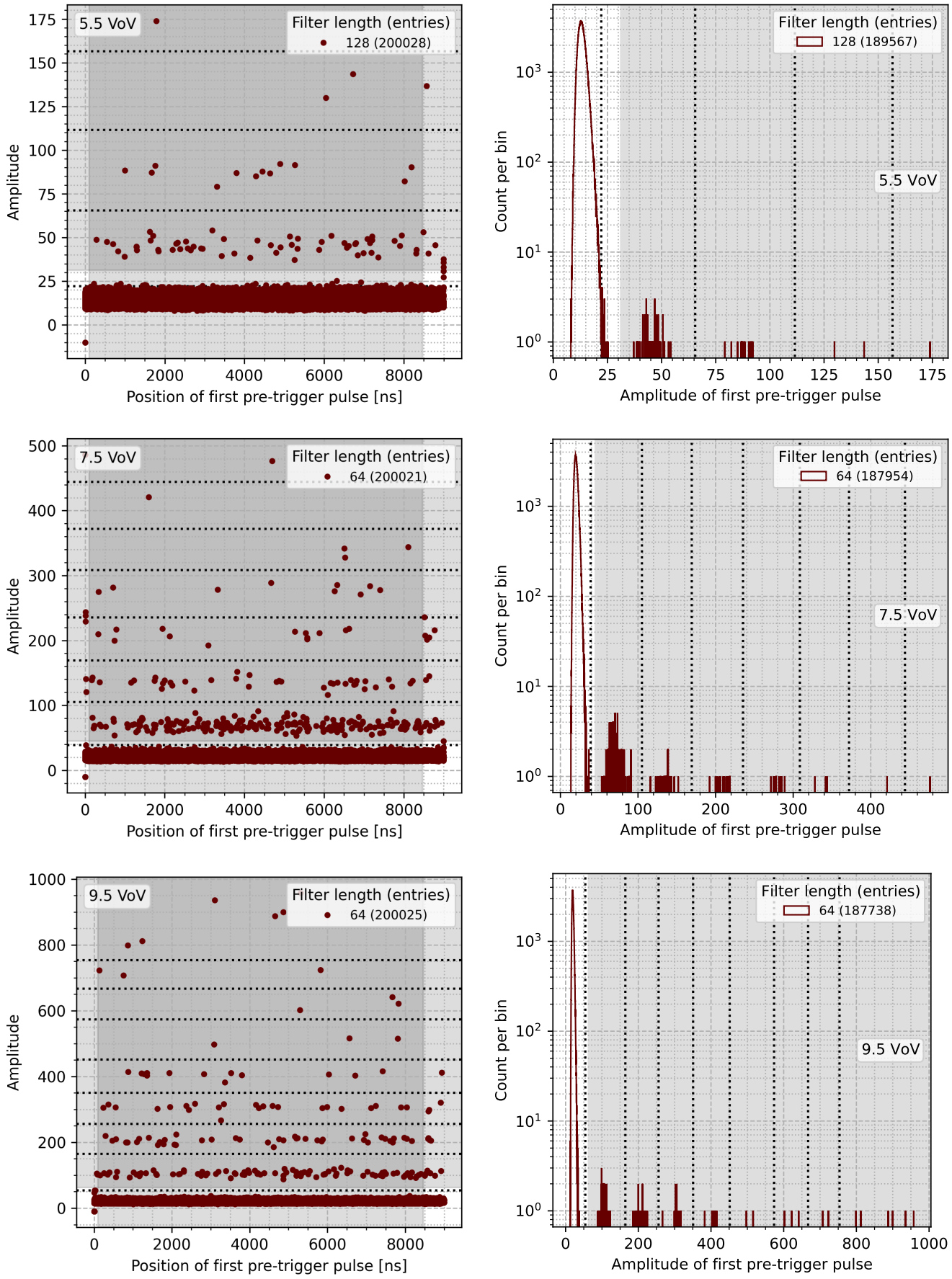


Figure A.11: In each row, left panel: scatter plot amplitude vs. position of the chronological first pre-trigger peak in each event. Right panel: histogram of the amplitude for the peaks contained in the vertical gray band in the left plot. (`figpretrigger2.py`)

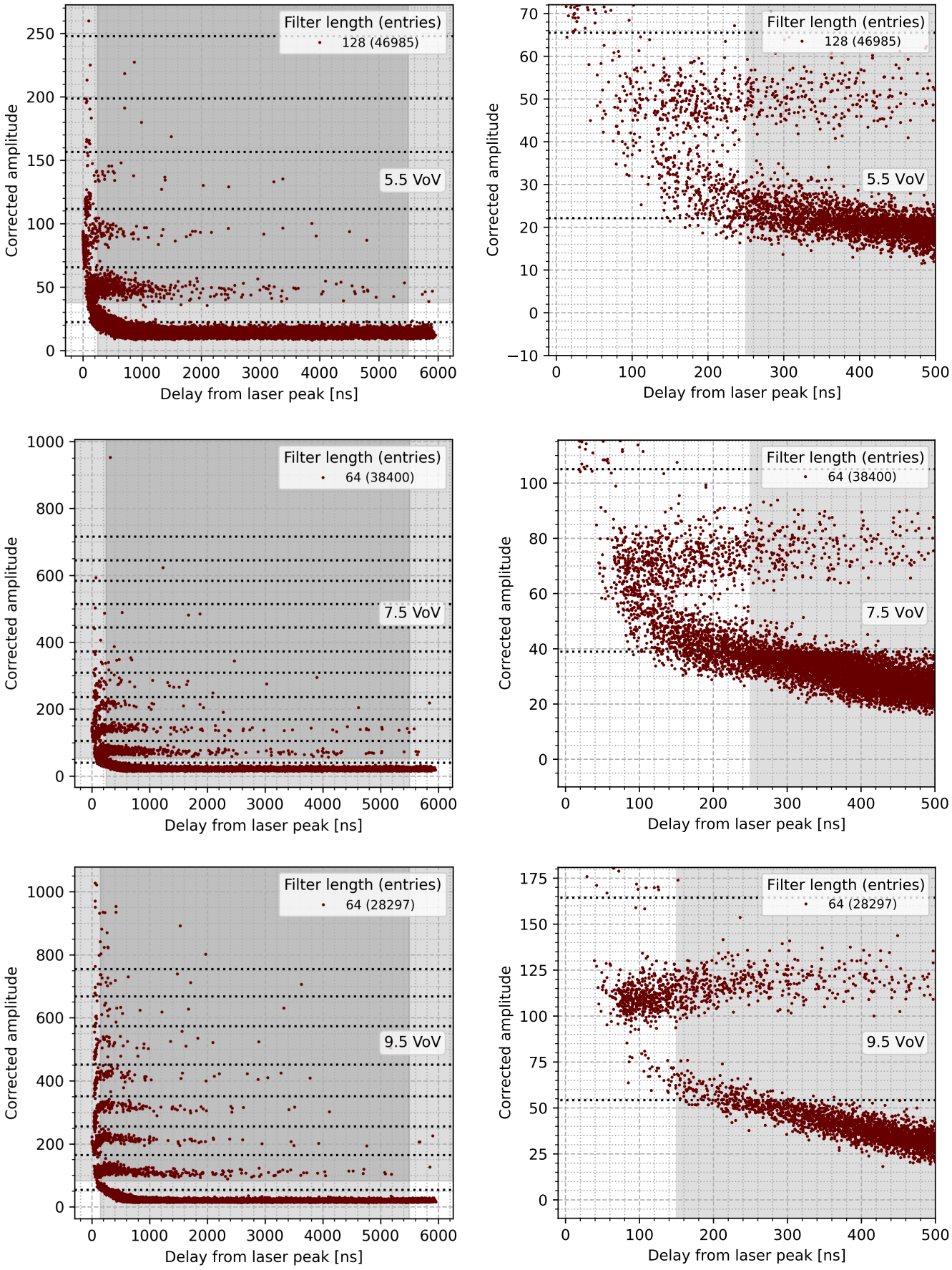


Figure A.12: In each row, left panel: corrected amplitude versus delay from laser peak; the gray bands mark the selected ranges. Right panel: zoom on the selection corner. (`figapscatter2.py`)

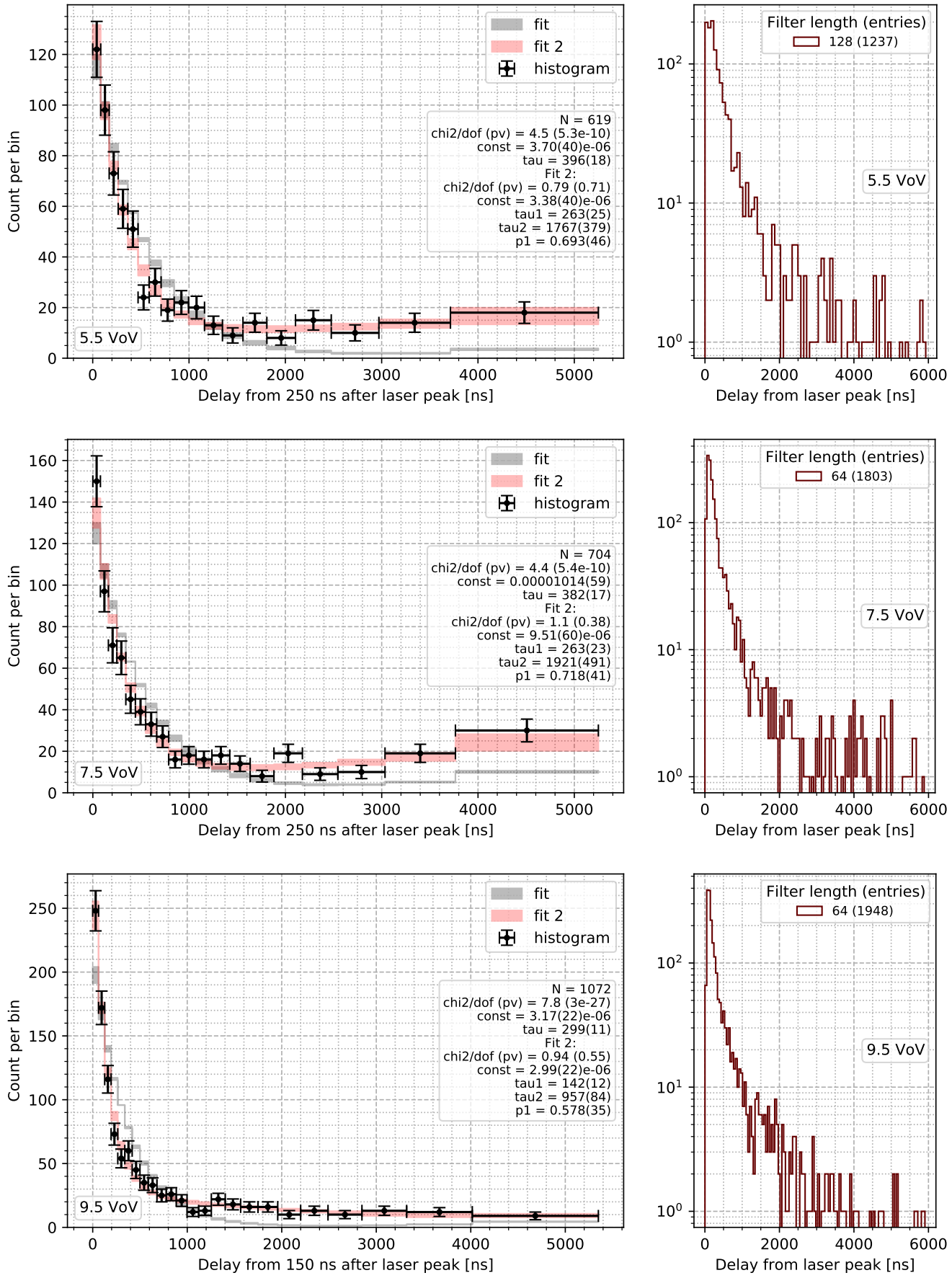


Figure A.13: In each row, left panel: fit of the temporal distribution of post-trigger pulses. Right panel: the histogram with even bins and without the temporal cuts. In the legend, 'const' is the background excess density, i.e., $(R/N)/((t_R - t_L)(1 - R/N))$, in ns^{-1}. (`figapfit2.py`)

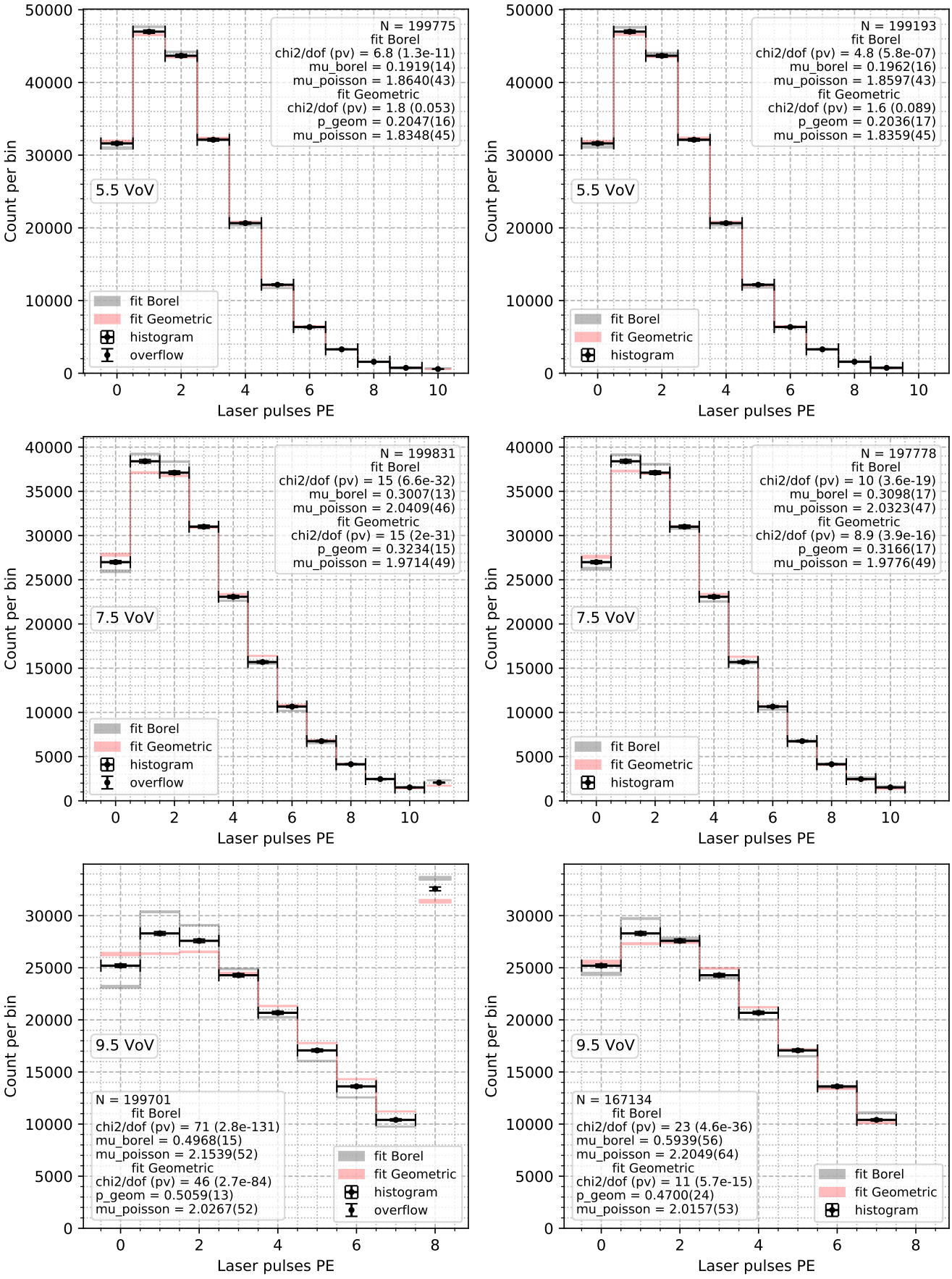


Figure A.14: Fit of the DiCT models on laser pulses (Equations 7.9 and 7.8). Left panels with overflow bin, right panels without. (`figctfitlaser.py`)

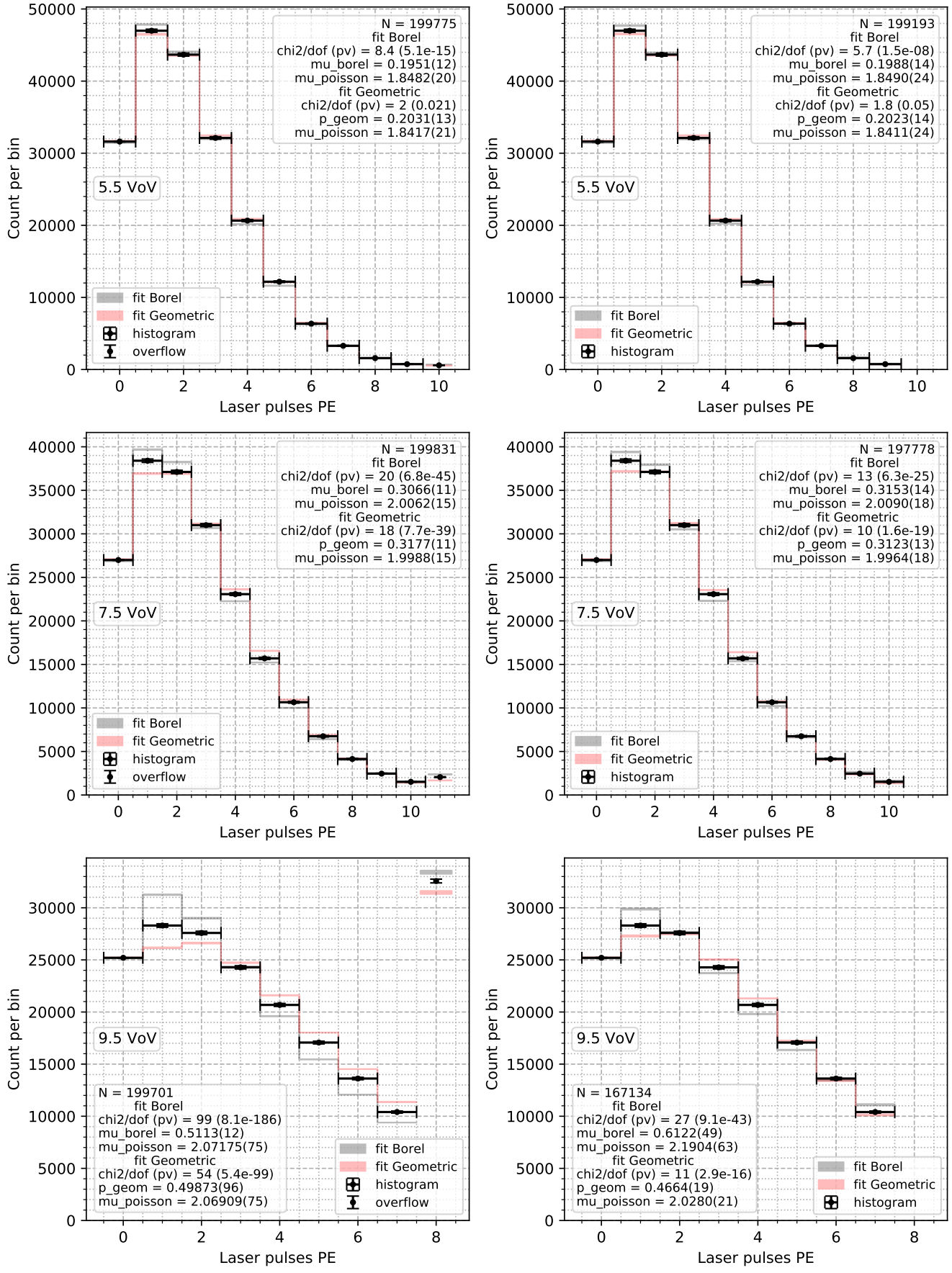


Figure A.15: Fit of the DiCT models on laser pulses (Equations 7.9 and 7.8), "fixing" the 0 PE bin. Left panels with overflow bin, right panels without. (`figctfitlaser.py`)

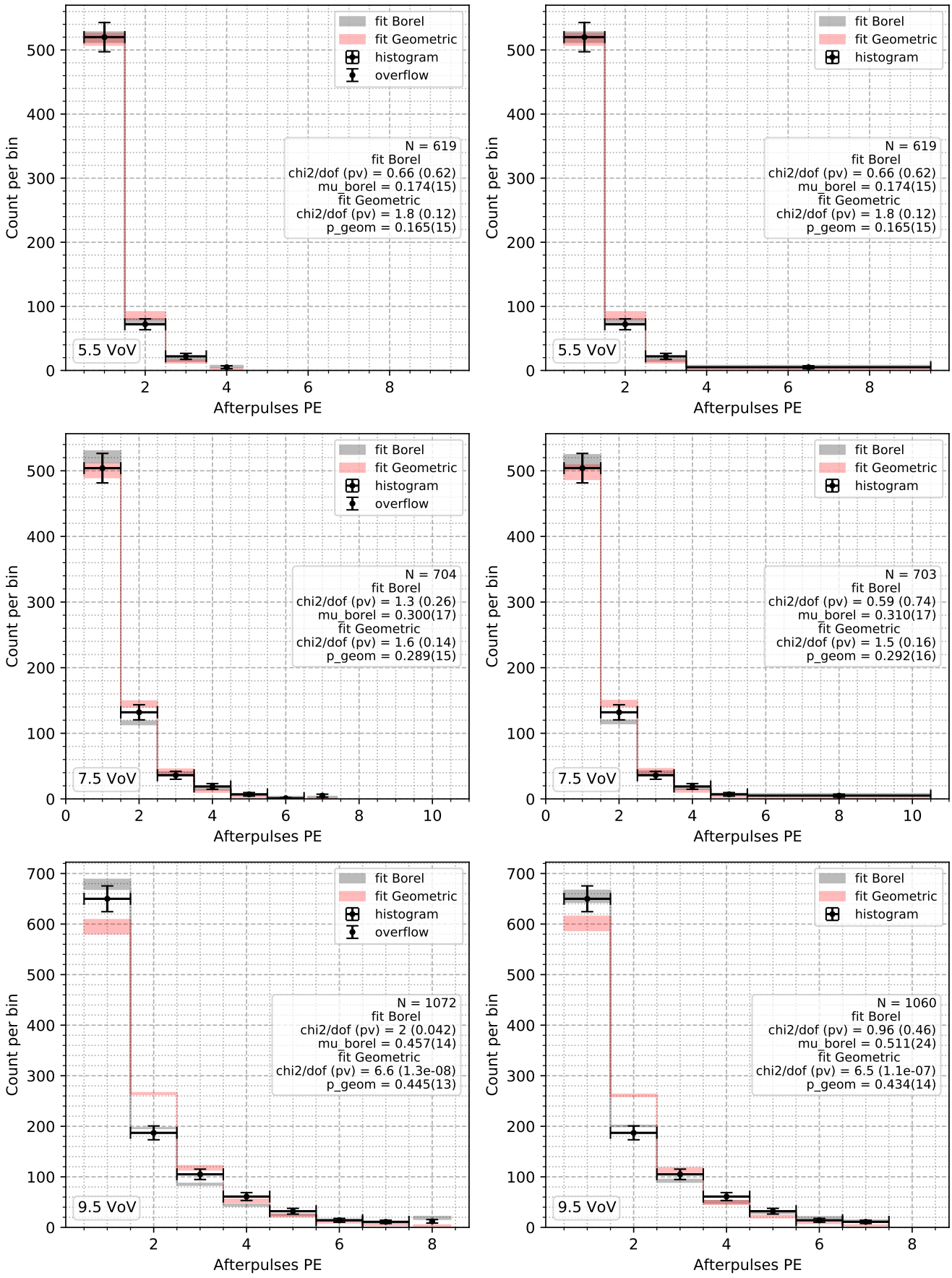


Figure A.16: Fit of the DiCT models on afterpulses (Equations 7.2 and 7.1). Left panels with overflow bin, right panels without. (`figctfitap.py`)

Appendix B

Fit of histograms

This chapter illustrates how the fits were done in chapter 7. They will turn out to be almost exactly a standard least squares fit of a histogram, but we will give a Bayesian interpretation to all the procedures. If you do not favor Bayesian methods, just do not think about it. To actually do the fits, we used the Python package `lsqfit` [LG21], which we warmly recommend.

B.1 Bayesian least squares

Bayesian inference is based on applying Bayes' theorem to get the probability distribution of the parameters conditional on the observed data. Call θ the parameters and y the data, then it reads

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)}. \tag{B.1}$$

The term $p(y|\theta)$ is the likelihood, how the data depends on the parameters. $p(y)$ is the unconditional probability of the data; whatever that means, we can compute it with $\int \mathrm{d}\theta\, p(y|\theta)p(\theta)$. $p(\theta)$ is the unconditional probability distribution of the parameters. It is called *prior* because it encodes what one knows about the parameters before having the data. By the same logic, $p(\theta|y)$ is called *posterior*.

Least squares is a general procedure to produce an estimator, but a standard interpretation is also as a maximum likelihood fit when the likelihood is Gaussian. We can do the same in the Bayesian case if the prior is Gaussian too. We switch to logarithms and drop addends which do not depend on the parameters:

$$\log p(\mathbf{y}|\boldsymbol{\theta}) = -\sum_{i=1}^{n} \log \sigma_i - \frac{1}{2}\sum_{i=1}^{n} \left(\frac{y_i - \mu_i(\boldsymbol{\theta})}{\sigma_i}\right)^2, \tag{B.2}$$

$$\log p(\boldsymbol{\theta}) = -\sum_{j=1}^{k} \log \sigma_j^\theta - \frac{1}{2}\sum_{j=1}^{k} \left(\frac{\mu_j^\theta - \theta_j}{\sigma_j^\theta}\right)^2. \tag{B.3}$$

Now these two terms have to be summed to obtain the log posterior. If we extend the $\mathbf{y}$, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ vectors, we can see the two sums together as a

single sum of squares:

$$y_{n+j} \equiv \mu_j^\theta, \tag{B.4}$$

$$\mu_{n+j}(\boldsymbol{\theta}) \equiv \theta_j, \tag{B.5}$$

$$\sigma_{n+j} \equiv \sigma_j^\theta, \qquad j = 1, \ldots, k, \tag{B.6}$$

$$\log p(\boldsymbol{\theta}|\mathbf{y}) = -\sum_{i=1}^{n+k} \log \sigma_i - \frac{1}{2} \sum_{i=1}^{n+k} \left(\frac{y_i - \mu_i(\boldsymbol{\theta})}{\sigma_i} \right)^2 \equiv \tag{B.7}$$

$$\equiv -\sum_{i=1}^{n+k} \log \sigma_i - \frac{1}{2}\chi^2. \tag{B.8}$$

So formally the prior is equivalent to additional datapoints, one for each parameter. We have not dropped the sum with the $\log \sigma$ although it does not depend on $\boldsymbol{\theta}$; it will be needed later. Note that the degrees of freedom of the χ^2 now are $(n+k) - k = n$, instead of the usual $n - k$ without prior.

In the analysis we said we put uniform priors over the parameters defined in the interval $(0, 1)$. This is implemented by fitting a transformed parameter with a standard Gaussian prior with zero mean and unitary variance, and applying as inverse transformation the Gaussian cumulative density function (cdf). Say, if we want to put a uniform prior on θ_j, we fit the transformed parameter θ'_j, with

$$\mu_j^{\theta'} = 0, \tag{B.9}$$

$$\sigma_j^{\theta'} = 1, \tag{B.10}$$

$$\theta_j(\theta'_j) = \Phi(\theta'_j) \equiv \int_{-\infty}^{\theta'_j} \mathrm{d}u \, \frac{e^{-u^2/2}}{\sqrt{2\pi}}. \tag{B.11}$$

Bayesianly, this is equivalent to omitting the squared term for the prior and putting bounds on the parameter value, without transforming. However it is convenient for the fitting routine. By the same logic, when we use the logarithm to map a positive parameter to the real line, the prior on the untransformed parameter is a log-Gaussian distribution.

Now the least squares estimator would be obtained by minimizing the χ^2, while a Bayesian cares about the posterior as a distribution. If we approximate the posterior as a Gaussian, we can do formally the same calculations. To compute the Gaussian approximation we expand the logarithm of the distribution to second order around the maximum:

$$\log p(\boldsymbol{\theta}|\mathbf{y}) \approx -\frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\top V^{-1} (\boldsymbol{\theta} - \boldsymbol{\theta}_0), \tag{B.12}$$

$$\boldsymbol{\theta}_0 = \arg\min_{\boldsymbol{\theta}} \chi^2, \tag{B.13}$$

$$V_{jk}^{-1} = \frac{1}{2} \left. \frac{\partial^2 \chi^2}{\partial \theta_j \partial \theta_k} \right|_{\boldsymbol{\theta}_0}. \tag{B.14}$$

Now let us carry further the calculation of the precision matrix, to discuss an additional issue:

$$\frac{\partial \chi^2}{\partial \theta_j} = 2 \sum_i \frac{y_i - \mu_i(\boldsymbol{\theta})}{\sigma_i^2} \frac{\partial \mu_i}{\partial \theta_j}, \tag{B.15}$$

$$\frac{\partial^2 \chi^2}{\partial \theta_j \partial \theta_k} = 2 \sum_i \frac{1}{\sigma_i^2} \left(-\frac{\partial \mu_i}{\partial \theta_j} \frac{\partial \mu_i}{\partial \theta_k} + (y_i - \mu_i(\boldsymbol{\theta})) \frac{\partial^2 \mu_i}{\partial \theta_j \partial \theta_k} \right). \tag{B.16}$$

In standard least squares, the precision matrix of the estimator is estimated with the above expression, but with the term with the second derivative dropped. The reason is for convenience, but also because the expectation

of the residuals $y_i - \mu_i(\boldsymbol{\theta}_0(\mathbf{y}))$ is approximately zero. A frequentist would like to compute this with the true value of the parameters, but has to content herself with the least squares estimate, getting something which on average is the true precision matrix. So any term which is zero on average can be removed.

For a Bayesian, instead, there is not some ideal error corresponding to an ideal parameter value; there is only the covariance of the posterior obtained with the particular values at hand. Different data, different posterior. However in our fits the residuals term is missing, as we said just because this is convenient for the implementation. In practice, the difference should be small.

B.2 Least squares fit of a histogram

The standard model of a histogram is Poisson distributions for the bin counts. Let c_i be the count in bin i, $f_i(\boldsymbol{\theta})$ the model normalized distribution of the bins, and M a parameter for the mean of the total (Poisson) number of samples. Then the likelihood is

$$P(\mathbf{c}|M,\boldsymbol{\theta}) = \prod_i P(c_i|M,\boldsymbol{\theta}) = \prod_i \text{poisson}(c_i; M f_i(\boldsymbol{\theta})), \tag{B.17}$$

where $\text{poisson}(\cdot;\mu)$ is a Poisson probability mass function (pmf) with mean μ.

We do not care about M, only about $\boldsymbol{\theta}$, so we marginalize the posterior over M:

$$p(\boldsymbol{\theta}|\mathbf{c}) = \int \mathrm{d}M\, p(M,\boldsymbol{\theta}|\mathbf{c}) = \tag{B.18}$$

$$= \int \mathrm{d}M\, \frac{P(\mathbf{c}|M,\boldsymbol{\theta})p(M,\boldsymbol{\theta})}{P(\mathbf{c})}. \tag{B.19}$$

Now we assume that a priori M is independent from $\boldsymbol{\theta}$, i.e., $p(M,\boldsymbol{\theta}) = p(M)p(\boldsymbol{\theta})$. Then, continuing the calculation:

$$p(\boldsymbol{\theta}|\mathbf{c}) = \frac{p(\boldsymbol{\theta})}{P(\mathbf{c})} \int \mathrm{d}M\, p(M) \prod_i \text{poisson}(c_i; M f_i(\boldsymbol{\theta})) = \tag{B.20}$$

$$= \frac{p(\boldsymbol{\theta})}{P(\mathbf{c})} \int \mathrm{d}M\, p(M) \prod_i e^{-M f_i(\boldsymbol{\theta})} \frac{(M f_i(\boldsymbol{\theta}))^{c_i}}{c_i!} = \tag{B.21}$$

$$= \frac{p(\boldsymbol{\theta})}{P(\mathbf{c})} \int \mathrm{d}M\, p(M) e^{-M\sum_i f_i(\boldsymbol{\theta})} M^{\sum_i c_i} \prod_i \frac{f_i(\boldsymbol{\theta})^{c_i}}{c_i!} \tag{B.22}$$

Since f_i is normalized the sum over it yields 1, while we call N the total count, so

$$p(\boldsymbol{\theta}|\mathbf{c}) = \frac{p(\boldsymbol{\theta})}{P(\mathbf{c})} \prod_i \frac{f_i(\boldsymbol{\theta})_i^c}{c_i!} \int \mathrm{d}M\, p(M) e^{-M} M^N. \tag{B.23}$$

The integral does not depend on $\boldsymbol{\theta}$, so we call it $I(N)$ and do not care about its value. Now we take the logarithm:

$$\log p(\boldsymbol{\theta}|\mathbf{c}) = -\log P(\mathbf{c}) + \log I(N) - \sum_i \log c_i! + \log p(\boldsymbol{\theta}) + \sum_i c_i \log f_i(\boldsymbol{\theta}). \tag{B.24}$$

This is the exact expression for the posterior, which we want to compare to least squares, which yields

$$Q(\boldsymbol{\theta}) = \log p(\boldsymbol{\theta}) - \frac{1}{2} \sum_i \frac{(c_i - N f_i(\boldsymbol{\theta}))^2}{c_i}. \tag{B.25}$$

Note the particular flavor of chisquare we have picked: there is no parameter for the normalization, which is fixed to the actual total count N, and the denominator is the observed count c_i.

To do the comparison, we expand both expressions to second order, ignoring the shared prior term which is already matched:

$$L(\boldsymbol{\theta}) \equiv \log p(\boldsymbol{\theta}|\mathbf{c}), \tag{B.26}$$

$$\frac{\partial L}{\partial \theta_j} = \sum_i \frac{c_i}{f_i(\boldsymbol{\theta})} \frac{\partial f_i}{\partial \theta_j}, \tag{B.27}$$

$$\frac{\partial^2 L}{\partial \theta_j \partial \theta_k} = \sum_i c_i \left[-\frac{1}{f_i(\boldsymbol{\theta})^2} \frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} + \frac{1}{f_i(\boldsymbol{\theta})} \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} \right]. \tag{B.28}$$

For Q we start from (B.16), obtaining

$$\frac{\partial^2 Q}{\partial \theta_j \partial \theta_k} = -N^2 \sum_i \frac{1}{c_i} \left[\frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} - \left(\frac{c_i}{N} - f_i(\boldsymbol{\theta}) \right) \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} \right]. \tag{B.29}$$

We do not discuss the different results for the maximum because it is well known that when there at least five counts per bin the least squares approximation is good enough. The covariance instead needs a discussion. Now we manipulate $\partial_{jk} L$ to show it is similar to $\partial_{jk} Q$. First, we approximate the observed counts with the model, $c_i \approx N f_i(\boldsymbol{\theta})$:

$$\frac{\partial^2 L}{\partial \theta_j \partial \theta_k} = -N^2 \sum_i \frac{c_i}{N^2 f_i(\boldsymbol{\theta})^2} \left[\frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} - f_i(\boldsymbol{\theta}) \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} \right] \approx \tag{B.30}$$

$$\approx -N^2 \sum_i \frac{1}{c_i} \left[\frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} - f_i(\boldsymbol{\theta}) \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} \right]. \tag{B.31}$$

Next, we observe that since f is normalized, $\sum_i \partial_{jk} f_i = 0$, so we can add the missing term to obtain the residuals:

$$\frac{\partial^2 L}{\partial \theta_j \partial \theta_k} \approx -N^2 \sum_i \frac{1}{c_i} \left[\frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} + \left(\frac{c_i}{N} - f_i(\boldsymbol{\theta}) \right) \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} \right]. \tag{B.32}$$

Comparing (B.29) with (B.32), we see that the difference is the sign of the residuals term. As we explained above, in the least squares fit that term is dropped altogether, so notwithstanding the Poisson model, we end up doing the same kind of approximation we did starting from Gaussians.

B.3 The χ^2/dof correction

Starting from the Gaussian model posterior (B.8), we add a parameter s that rescales uniformly all the errors:

$$\log p(\boldsymbol{\theta}|\mathbf{y}) = -\sum_{i=1}^{n+k} \log \sigma_i - \frac{1}{2}\chi^2 \mapsto \tag{B.33}$$

$$\mapsto \log p(\boldsymbol{\theta}, s|\mathbf{y}) = \log p(s) - \sum_{i=1}^{n+k} \log(s\sigma_i) - \frac{1}{2} \sum_{i=1}^{n+k} \left(\frac{y_i - \mu_i(\boldsymbol{\theta})}{s\sigma_i} \right)^2 = \tag{B.34}$$

$$= \log p(s) - (n+k) \log s - \sum_{i=1}^{n+k} \log \sigma_i - \frac{1}{2} \frac{\chi^2}{s^2}. \tag{B.35}$$

Since s is a scale parameter, it makes sense to use a uniform prior over $\log s$, i.e., $p(s) = 1/s$, because it is scale-invariant. Now that we have let errors vary, we marginalize away s, freely dropping normalizations as usual:

$$p(\boldsymbol{\theta}|\mathbf{y}) = \int \mathrm{d}s\, p(\boldsymbol{\theta}, s|\mathbf{y}) = \tag{B.36}$$

$$= \int \frac{\mathrm{d}s}{s^{n+k+1}} \exp\left(-\frac{1}{2}\frac{\chi^2}{s^2}\right) = \tag{B.37}$$

$$= \frac{1}{(\chi^2)^{(n+k)/2}}. \tag{B.38}$$

Now again we take the logarithm and expand to second order:

$$P(\boldsymbol{\theta}) \equiv \log p(\boldsymbol{\theta}|\mathbf{y}) = -\frac{n+k}{2}\log\chi^2, \tag{B.39}$$

$$\frac{\partial P}{\partial\theta_j} = -\frac{n+k}{2}\frac{1}{\chi^2}\frac{\partial\chi^2}{\partial\theta_j}, \tag{B.40}$$

$$\frac{\partial^2 P}{\partial\theta_j\partial\theta_k} = -\frac{n+k}{2}\left[-\frac{1}{(\chi^2)^2}\frac{\partial\chi^2}{\partial\theta_j}\frac{\partial\chi^2}{\partial\theta_k} + \frac{1}{\chi^2}\frac{\partial^2\chi^2}{\partial\theta_j\partial\theta_k}\right]. \tag{B.41}$$

In the maximum $\partial_j P = 0$, so the first term in (B.41) vanishes, leaving

$$\frac{\partial^2 P}{\partial\theta_j\partial\theta_k} = -\frac{1}{2}\frac{1}{\chi^2/(n+k)}\frac{\partial^2\chi^2}{\partial\theta_j\partial\theta_k}. \tag{B.42}$$

This is to be compared with the precision matrix in (B.14). The only difference is that the matrix has been divided by $\chi^2/(n+k)$. Note that dof $= n$, so we obtained a different prescription than the conventional one. Nevertheless, in the analysis we used χ^2/dof just because it is what people expect; compared to (B.42), this gives errors up to 25 % larger on our fits, see Table 7.4.

Similarly, we used the pvalue; since we always obtained a stark contrast between good and bad fits, we did not bother giving a precise Bayesian equivalent.

Appendix C

Source code

The code written for this thesis is open-source. For each figure and table we provide a Python script that reproduces the content. Each script is referenced at the end of the caption like this: `(fignoise.py)`. In the PDF it links is to a preview of the file in an online repository, `https://github.com/Gattocrucco/sipmfilter`. To run these scripts, clone or download the repository. The scripts are located in the directory `figthesis`. The file `README.md` provides detailed instructions on how to set up the working environment.

Some scripts require no input. Others require large data files which are not included in the repository. Of the latter, some have a cache of the data they need, others just can not be executed without the original data. For people outside of the DarkSide collaboration it may not be possible to obtain the data files; in any case, they can check what the code is doing exactly.

The LaTeX code for this thesis is itself available at `https://github.com/Gattocrucco/thesis`. Moreover, the `sipmfilter` repository contains slides on this work which we presented at DarkSide meetings, although they shall be considered outdated respect to this document. The Python code is covered by the MIT license, while the thesis and the slides are covered by the CC-BY 4.0. These licenses grant anyone the right to reuse this material, even without releasing themselves the modifications as open source, provided they cite the original author and keep the copyright and license notices. The licenses do not cover figures which we copied from other sources nor the DarkSide data.

Bibliography

[Aal+18] C. E. Aalseth et al. "DarkSide-20k: A 20 tonne two-phase LAr TPC for direct dark matter detection at LNGS". In: *The European Physical Journal Plus* 133.3 (2018), p. 131. ISSN: 2190-5444. DOI: 10.1140/epjp/i2018-11973-4. URL: https://link.springer.com/article/10.1140/epjp/i2018-11973-4.

[Aal+19] C. E. Aalseth et al. *DarkSide-20k Preliminary Design Report: The Global Argon Dark Matter Collaboration*. Tech. rep. Research Publications at Politecnico di Milano, 2019. URL: http://hdl.handle.net/11311/1169015.

[Ace+17] Fabio Acerbi et al. "Cryogenic Characterization of FBK HD Near-UV Sensitive SiPMs". In: *IEEE Transactions on Electron Devices* 64.2 (2017), pp. 521–526. DOI: 10.1109/TED.2016.2641586.

[Apr+06] Elena Aprile et al. *Noble Gas Detectors*. John Wiley & Sons, Ltd, 2006. ISBN: 9783527610020. DOI: 10.1002/9783527610020. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/9783527610020.

[Bax+21] D. Baxter et al. *Recommended conventions for reporting results from direct dark matter searches*. 2021. arXiv: 2105.00599 [hep-ex].

[BLF19] Jatan Buch, Shing Chau (John) Leung, and JiJi Fan. "Using Gaia DR2 to constrain local dark matter density and thin dark disk". In: *Journal of Cosmology and Astroparticle Physics* 2019.04 (Apr. 2019), pp. 026–026. DOI: 10.1088/1475-7516/2019/04/026. URL: https://iopscience.iop.org/article/10.1088/1475-7516/2019/04/026/meta.

[CH01] K. J. Coakley and P. Hale. "Alignment of noisy signals". In: *IEEE Transactions on Instrumentation and Measurement* 50.1 (2001), pp. 141–149. DOI: 10.1109/19.903892. URL: https://ieeexplore.ieee.org/abstract/document/903892.

[Chm+17] V. Chmill et al. "On the characterisation of SiPMs from pulse-height spectra". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 854 (2017), pp. 70–81. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2017.02.049. URL: https://www.sciencedirect.com/science/article/pii/S0168900217302334.

[Clo+06] Douglas Clowe et al. "A Direct Empirical Proof of the Existence of Dark Matter". In: *The Astrophysical Journal* 648.2 (Aug. 2006), pp. L109–L113. DOI: 10.1086/508162. URL: https://iopscience.iop.org/article/10.1086/508162.

[CLR91] S. Cova, A. Lacaita, and G. Ripamonti. "Trapping phenomena in avalanche photodiodes on nanosecond scale". In: *IEEE Electron Device Letters* 12.12 (1991), pp. 685–687. DOI: 10.1109/55.116955.

[Dar21a] DarkSide collaboration. 2021. URL: https://twiki.cern.ch/twiki/bin/view/Sandbox/DarkSideProto0.

[Dar21b] DarkSide collaboration. 2021. URL: https://drive.google.com/drive/folders/1j9xyqWUhxyX4fAmQfCJ3QVC2gZPCc5xI.

[Dru+15] Andrzej Drukier et al. *New Dark Matter Detectors using DNA or RNA for Nanometer Tracking*. 2015. arXiv: 1206.6809 [astro-ph.IM].

[Fer15] Isidoro Ferrante. *Elaborazione dei segnali per la fisica*. Pisa University Press, 2015. ISBN: 978-88-6741-548-9.

[Gar+14] E. Garutti et al. "Afterpulse effect in SiPM and neutron irradiation studies". In: *2014 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*. Nov. 2014, pp. 1–7. DOI: 10.1109/NSSMIC.2014.7430746.

[Gol+14] Alberto Gola et al. "SiPM optical crosstalk amplification due to scintillator crystal: effects on timing performance". In: *Physics in Medicine and Biology* 59.13 (June 2014), pp. 3615–3635. DOI: 10.1088/0031-9155/59/13/3615. URL: https://iopscience.iop.org/article/10.1088/0031-9155/59/13/3615/meta.

[Gol+19] Alberto Gola et al. "NUV-Sensitive Silicon Photomultiplier Technologies Developed at Fondazione Bruno Kessler". In: *Sensors* 19.2 (2019). ISSN: 1424-8220. DOI: 10.3390/s19020308. URL: https://www.mdpi.com/1424-8220/19/2/308.

[Har+15] David Harvey et al. "The nongravitational interactions of dark matter in colliding galaxy clusters". In: *Science* 347.6229 (2015), pp. 1462–1465. ISSN: 0036-8075. DOI: 10.1126/science.1261381. eprint: https://science.sciencemag.org/content/347/6229/1462.full.pdf.

[Kno10] Glenn F. Knoll. *Radiation detection and measurement*. Third edition. John Wiley & Sons, 2010. ISBN: 0-471-07338-5.

[Kri20] Ben Krikler. "A readily-interpretable fully-convolutional autoencoder-like model for unlabelled waveform analysis". In: *DANCE Machine Learning Workshop*. 2020. URL: https://indico.physics.lbl.gov/event/1192/contributions/4934/.

[LG21] Peter Lepage and Christoph Gohlke. *lsqfit*. Version 11.8. Feb. 2021. DOI: 10.5281/zenodo.4568470. URL: https://github.com/gplepage/lsqfit.

[Luz20] Ludovico Luzzi. "Characterisation of SiPM detectors for dark matter detection with the liquid-argon time-projection chamber of the DarkSide prototype". MA thesis. Sapienza Università di Roma, May 2020. URL: http://www.infn.it/thesis/thesis_dettaglio.php?tid=14656.

[Mar19] Andrea Marasciulli. "La prima Motherboard di DarkSide-20k: dal montaggio al primo test in argon liquido". MA thesis. Università di Pisa, 2019. URL: https://etd.adm.unipi.it/t/etd-09252019-101839/.

[Nag+14] Ferenc Nagy et al. "Afterpulse and delayed crosstalk analysis on a STMicroelectronics silicon photomultiplier". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 759 (2014), pp. 44–49. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2014.04.045. URL: https://www.sciencedirect.com/science/article/pii/S0168900214004501.

[Nue08] Grégory Nuel. "Cumulative distribution function of a geometric Poisson distribution". In: *Journal of Statistical Computation and Simulation* 78.3 (2008), pp. 385–394. DOI: 10.1080/10629360600997371. URL: https://www.tandfonline.com/doi/abs/10.1080/10629360600997371.

[Pla+20] Planck Collaboration et al. "Planck 2018 results - I. Overview and the cosmological legacy of Planck". In: *A&A* 641 (2020), A1. DOI: 10.1051/0004-6361/201833880. eprint: https://www.aanda.org/articles/aa/pdf/2020/09/aa33880-18.pdf.

[RK06] Carl Edward Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*. 2006. ISBN: 0-262-18253-X. URL: http://www.gaussianprocess.org/gpml/.

[Sav18] Claudio Savarese. "A novel light detector for DarkSide-20k". PhD thesis. Gran Sasso Science Institute, 2018. URL: http://hdl.handle.net/20.500.12571/9683.

[TY18] Sean Tulin and Hai-Bo Yu. "Dark matter self-interactions and small scale structure". In: *Physics Reports* 730 (2018), pp. 1–57. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2017.11.004. URL: https://www.sciencedirect.com/science/article/pii/S0370157317304039.

[Vin12] S. Vinogradov. "Analytical models of probability distribution and excess noise factor of solid state photomultiplier signals with crosstalk". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 695 (2012). New Developments in Photodetection NDIP11, pp. 247–251. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2011.11.086. URL: https://www.sciencedirect.com/science/article/pii/S0168900211021565.

[VZL12] Mark Vogelsberger, Jesus Zavala, and Abraham Loeb. "Subhaloes in self-interacting galactic dark matter haloes". In: *Monthly Notices of the Royal Astronomical Society* 423.4 (July 2012), pp. 3740–3752. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2012.21182.x. eprint: https://academic.oup.com/mnras/article-pdf/423/4/3740/4916526/mnras0423-3740.pdf.

[Zwi33] F. Zwicky. "Die Rotverschiebung von extragalaktischen Nebeln". In: *Helvetica Physica Acta* 6.II (1933), p. 110. ISSN: 0018-0238. DOI: 10.5169/seals-110267.

[Zyl+20] P. A. Zyla et al. "Review of Particle Physics". In: *Progress of Theoretical and Experimental Physics* 2020.8 (Aug. 2020). 083C01. ISSN: 2050-3911. DOI: 10.1093/ptep/ptaa104. eprint: https://academic.oup.com/ptep/article-pdf/2020/8/083C01/34673722/ptaa104.pdf. URL: https://pdg.lbl.gov/.