Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

DLTPulseGenerator: A library for the simulation of lifetime spectra based on detector-output pulses

Danny Petschke [*], Torsten E.M. Staab

University Wuerzburg, Department of Chemistry, LCTM Roentgenring 11, D-97070 Wuerzburg, Germany

ARTICLE INFO

Article history:
Received 20 October 2017
Received in revised form 9 April 2018
Accepted 10 April 2018

Keywords:
Lifetime spectroscopy
Signal processing
Pulse simulation

ABSTRACT

The quantitative analysis of lifetime spectra relevant in both life and materials sciences presents one of the *ill-posed* inverse problems and, hence, leads to most stringent requirements on the hardware specifications and the analysis algorithms. Here we present *DLTPulseGenerator*, a library written in native *C++ 11*, which provides a simulation of lifetime spectra according to the measurement setup. The simulation is based on pairs of non-TTL detector output-pulses. Those pulses require the Constant Fraction Principle (CFD) for the determination of the exact timing signal and, thus, the calculation of the time difference i.e. the lifetime. To verify the functionality, simulation results were compared to experimentally obtained data using Positron Annihilation Lifetime Spectroscopy (PALS) on pure tin.

Code metadata

Current code version	v1
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-17-00077
Legal Code License	BSD-3-clause
Code versioning system used	GitHub
Software code languages, tools, and services used	C/C++ and Python
Compilation requirements, operating environments & dependencies	**OS**: Microsoft Windows **Compilation requirements (for DLTPulseGenerator.h/.cpp only):** should work with any C++ compiler (has to provide C++11 standard) – recommended: MS-VSCompiler (at least version 2013) **Dependencies for example C++ project - AppDLTPulseGenerator:** Microsoft Visual Studio 2015 **Dependencies for C++ wrapper in Python - pyDLTPulseGenerator.py:** ctypes-library **Dependencies for example project in Python - pyDLTPulseGeneratorApp.py:** matplotlib, NumPy
If available Link to developer documentation/manual	A ***Readme.md*** *file* can be found on **GitHub**: https://github.com/dpscience/DLTPulseGenerator/blob/master/README.md
Support email for questions	danny.petschke@uni-wuerzburg.de

1. Introduction and significance

Lifetime spectroscopy has become an established method in life science, physics and materials science over the last decades. A first step was taken in the early *1960's* by several groups measuring the lifetime distributions of antielectrons (positrons) in materials using photomultiplier–scintillator combinations to detect the gamma rays, which are emitted when they are annihilated with an electron [1–3]. This method, known as Positron Annihilation Lifetime Spectroscopy (PALS), is used for microstructure investigations in a broad range of material classes from metals [4–9] and semiconductors [10,11] to polymers [12,13] and porous glasses [14,15]. The characteristic or specific lifetimes are highly sensitive to the kind and size (from several angstroms to nanometers) of

* Corresponding author.
E-mail addresses: danny.petschke@uni-wuerzburg.de (D. Petschke), torsten.staab@uni-wuerzburg.de (T.E.M. Staab).

https://doi.org/10.1016/j.softx.2018.04.002

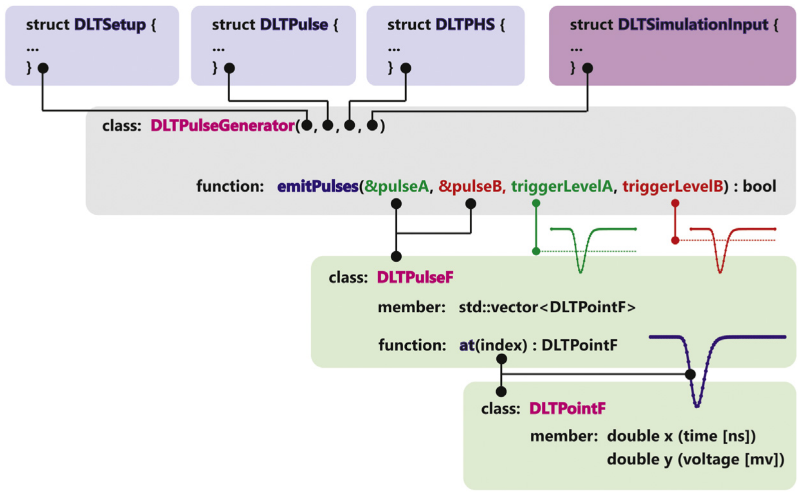
struct DLTSetup { ... }
struct DLTPulse { ... }
struct DLTPHS { ... }
struct DLTSimulationInput { ... }
class: DLTPulseGenerator(●, ●, ●, ●)
function: emitPulses(&pulseA, &pulseB, triggerLevelA, triggerLevelB) : bool
class: DLTPulseF
member: std::vector<DLTPointF>
function: at(index) : DLTPointF
class: DLTPointF
member: double x (time [ns])
double y (voltage [mv])

Fig. 1. Schematic illustration of the library architecture. *DLTPulseGenerator* expects four structures for initialization. *DLTPulseF* holds a vector of the data points (time vs. voltage) in double precision represented by the class *DLTPointF*.

material defects (e.g. impurities in semiconductors, vacancies in metals or pores in glasses) and range from several picoseconds to microseconds. After the technical realization of single photon sensitive detectors (photodiodes, APD — avalanche photodiodes), methods such as Fluorescence Lifetime Spectroscopy (FLS), Fluorescence Lifetime Imaging Microscopy (FLIM) or Fluorescence Lifetime Correlation Spectroscopy (FLCS) became feasible and are used nowadays to investigate (life cell) protein interactions [16] or diffusion dynamics [17].

Those lifetime spectra are mathematically described by a sum of exponential lifetime distributions f_i convoluted with an Instrument Response Function (IRF) g. According to the number of components N, the resulting function f is given as

$$f(t) = \sum_{i=0}^{N-1} f_i(t) = \sum_{i=0}^{N-1} \frac{I_i}{\tau_i} \exp\left\{-\frac{t}{\tau_i}\right\}, \quad (1)$$

where τ_i as the ith component *specific lifetime* and I_i its corresponding *intensity*. Mostly, the IRF is analytically approximated by a Gaussian distribution function[1]

$$g(t|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-0.5\left(\frac{t-\mu}{\sigma}\right)^2\right\}, \quad (2)$$

where σ is the standard deviation and μ is the mean.

A quantitative analysis of the lifetime spectra and, thus, the extraction of the relevant information, i.e. the *specific lifetimes* and its *intensities*, needs to solve the *ill-posed* inverse problem, which means that the uniqueness or even the existence of the solution is not always assured [18]. Various approaches are used: the *Maximum-Likelihood Method*, the most commonly used *Least-Square Fitting* (software: e.g. Positronfit [19–21], PALSFit [22,23], LT [24,25], FluoFit [26] or SymPhoTime 64 [27]) or the *Bayesian* approach using the quantified *Maximum Entropy Method* (MEM or MaxEnt) (software: e.g. MELT [28,29]). Especially for spectra consisting of multiple lifetime components and/or having shorter *specific lifetimes* than the instrumental resolution (means: $\tau_i < \sigma$), an IRF with minimum deviations from the model function Eq. (2) is required to solve this *ill*-conditioned problem, i.e. this is decisive for an exact data treatment. Therefore, the setup and the parameters relevant to determine the correct timing signal from the received pulses, using the Constant Fraction Principle (CFD), needs to be fully optimized to guarantee reproducible and comparable results.

DLTPulseGenerator library generates pairs of detector pulses (non-TTL signals) with exponentially distributed (Eq. (1)) time differences, i.e. the lifetime, by taking the pulse shape and the uncertainties (Eq. (2)) of the most relevant hardware components into account (Fig. 2). This allows the user to study the influences of both the measurement setup and configuration on the lifetime spectrum without being connected to the hardware.

2. Software description

DLTPulseGenerator is written in native *C++ 11* (ISO/IEC 14882:2011). It provides the optional compilation as static or linked library to make it easy accessible to other programming languages which are preferentially used in research and engineering, e.g. *Matlab* (using mex-library) or *Python* (using ctypes-library,[2]).

The ***class DLTPulseGenerator*** expects four structures (C/C++ syntax: *struct*) for initialization (Fig. 1):

i. **DLTSetup**
ii. **DLTPulse**
iii. **DLTPHS**
iv. **DLTSimulationInput**

These contain the specifications of the setup and information on pulse shape, pulse height distribution and the lifetime distributions. A pulse is represented by the ***class DLTPulseF*** and consists of a vector (*std::vector*) which holds the data points (***class DLTPointF***).

Calling the function *DLTPulseGenerator::emitPulses* expects:

I. a pointer to an object of the *class DLTPulseF*, which is then manipulated and filled with data points (time [ns] vs. voltage [mV]) in double precision and
II. the trigger-level.

In the following chapters, a detailed overview of the physical and mathematical background, which the simulation is based on, is given while relating to the mentioned structures (C/C++ syntax: *struct*).

[1] Different distribution functions such as Lorentz/Cauchy or Voigt as well as superpositions and skewing of distribution functions are not considered in this work but could be easily implemented.

[2] ***pyDLTPulseGenerator***: a class in *Python* showing the functionality and use of *ctypes-library* in combination with *DLTPulseGenerator* (compiled as linked library), is provided by the author.

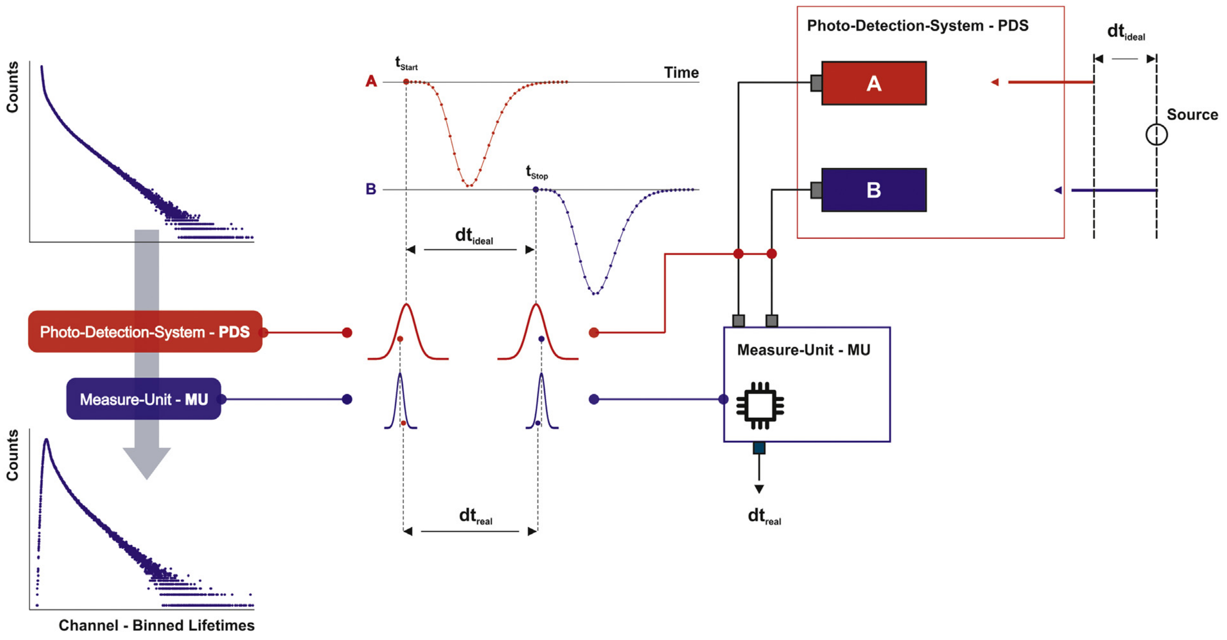

Fig. 2. Right: Simplified setup consisting of the Photo Detection System (PDS) and the Measure Unit (MU) only (right). The ideal lifetime dt_{ideal}, picked out from the *LTSelector*, is separated into the start and stop timestamp. These are then modified by the influences of the PDS and MU. So, the real lifetime dt_{real} and, thus, the real lifetime spectrum (left-bottom) is obtained.

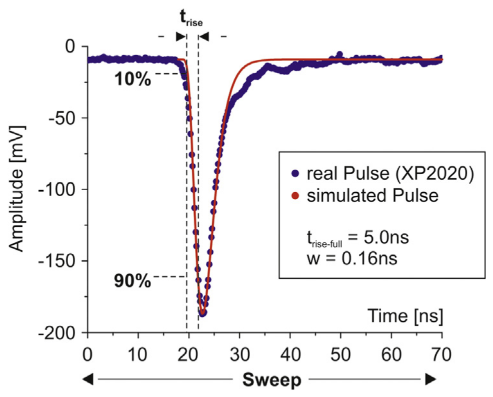

Fig. 3. Comparison between an anode pulse measured with a Philipps *XP2020* photomultiplier (blue) and the modeled pulse described by a log-normal distribution function (red) (Eq. (4)). The digitized pulses are stored in the Sweep which defines the readout range.

2.1. Software architecture

2.1.1. struct DLTSetup

Depending on the application, different modifications of the lifetime spectroscopy setup are required. In principle, any setup can be simplified to be described by two basic components:

I. the **Photo Detection System** (**PDS:** i.e. PMT (photomultiplier), APD or photodiode) and
II. the **Measure Unit** (**MU:** i.e. digitizer/digital oscilloscope including the electronics (e.g. ADC, FPGA)), which is necessary for the acquisition and digitization of the detected photons.

The variables ***PDSUncertaintyA*** and ***PDSUncertaintyB*** describe the standard deviations of the IRF of the detectors according to Eq. (2) in nanoseconds. For photomultipliers (PMT) this reflects the Transit Time Spread (TTS), whereas for photodiodes this represents the Jitter. The resolution of the MU (variable ***MUUncertainty*:** standard deviation of Eq. (2) in nanoseconds) describes how precise identical pulses can be acquired.

In principle, in a real setup the detected pulses are digitized by the ADC and stored in the Sweep (variable ***sweep***), which defines the readout range (***class DLTPulseF***) in nanoseconds (Fig. 3). The sample rate (or sample speed) ν in GHz is further given by

$$\nu = (N - 1)/\mathrm{Sweep}, \qquad (3)$$

where N indicates the number of sample points (variable ***numberOfCells***). In the case of different cable lengths, which connect the detectors A and B with the MU, an additional constant Arrival Time Spread (variable ***ATS*** in nanoseconds) is observed between the pulses. This leads subsequently to an offset in the lifetime spectrum.

2.1.2. struct DLTPulse

A non-TTL pulse originating from a PMT[3] or photodiode can mathematically be described with the log-normal distribution function

$$U(t|w, t_{\mathrm{rise-full}}) = A\ \exp\left\{-0.5\frac{\ln\left(\frac{t}{t_{\mathrm{rise-full}}}\right)^2}{w^2}\right\},\ t > 0 \qquad (4)$$

where $t_{\mathrm{rise-full}}$ denotes the rise time (variable ***riseTime*** in nanoseconds) from zero to the maximum amplitude A (variable ***amplitude***). Moreover, $t_{\mathrm{rise-full}}$ is the *mode* of the log-normal distribution function. The pulse width w is defined by the variable ***pulseWidth*** in nanoseconds. The rise time t_{rise} is defined by the time a pulse needs to rise from 10% to 90% of its amplitude A (Fig. 3). According

[3] Since the rise time is independent from the amplitude (same rise time for all amplitudes) for both variants of PMT pulses, the anode (unipolar) and the dynode signal (bipolar), the region, which is relevant for the timing determination (rising edge) using the CFD principle, can be modeled by a log-normal distribution (Eq. (4)) as seen in Fig. 3.

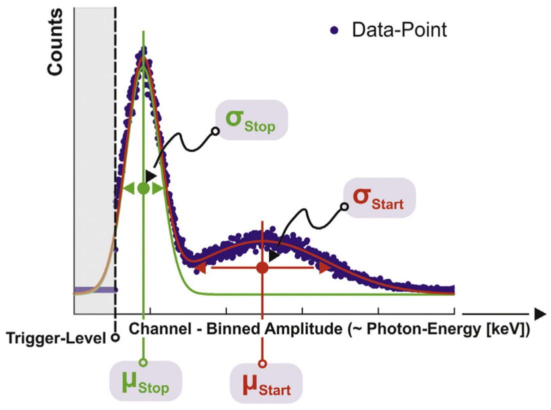

Fig. 4. Modeled pulse height spectrum (PHS) of the plastic scintillator BC422, often used in PALS experiments: The function is given by a linear combination of two Gaussian distribution functions, where each one is representing the photon energy distribution for the start ($\mu_{\text{Start}}, \sigma_{\text{Start}}$) and stop quantum ($\mu_{\text{Stop}}, \sigma_{\text{Stop}}$), respectively. The trigger-level (arguments in function *DLTPulseGenerator::emitPulses*) defines the minimum of the accepted pulse amplitude.

to Eq. (4), the conversion is given by:

$$t_{\text{rise}} = t_{U=0.9A} - t_{U=0.1A} = t_{\text{rise-full}} \left[\exp\left\{ w\sqrt{2}\sqrt{\ln\left(\frac{10}{9}\right)} \right\} - \exp\left\{ w\sqrt{2}\sqrt{\ln(10)} \right\} \right]. \quad (5)$$

Since the pulse shows a negative polarity in Fig. 3, the variable ***isPositivePulsePolarity*** has set to be *false*. Furthermore, a trigger-delay in nanoseconds can be added to the variable ***delay*** to shift the pulse-pair by an absolute value within the Sweep.

2.1.3. struct DLTPHS

In Positron Annihilation Lifetime Spectroscopy (PALS), the gamma quanta, detected as start and stop signal, carry different energies (Na-22: start = 1274 keV, stop = 511 keV). Therefore, an energy selection of the detected pulses is essential. The detection of energy resolved gamma rays is realized by a scintillator–photomultiplier combination. Due to Compton and/or backscattering effects within the scintillator materials (e.g. plastic, BaF_2 or LSO [30]), a distribution of the pulse amplitude, known as pulse height spectrum (PHS), is generated (Fig. 4). For subsequent analysis purposes, a simple model for the PHS of any scintillator materials is created as a linear combination of two Gaussian distribution functions (Eq. (2)), i.e. one Gaussian function for the start and stop pulse amplitude distribution. The variables for the detectors A and B (Fig. 2) are named as followed:

i. ***meanOfStartA, meanOfStartB*** [mV]
ii. ***meanOfStopA, meanOfStopB*** [mV]
iii. ***stddevOfStartA, stddevOfStartB*** [mV]
iv. ***stddevOfStopA, stddevOfStopB*** [mV]

From Eq. (2), μ represents ***mean*** while σ (standard deviation) is related to ***stddev*** (Fig. 4). The value of μ should be between the trigger-level and the maximum amplitude A (Eq. (4)) of the pulses. For photodiodes, the standard deviation can be set to zero, which then represents the amplitude distribution as a delta-function.

2.1.4. struct DLTSimulationInput

This structure contains the specific lifetimes and associated intensities according to Eq. (1). The number of input components for the simulation is limited to a maximum of five per spectrum

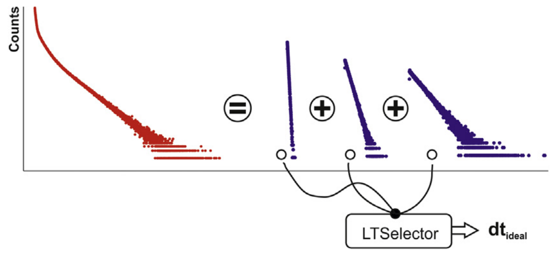

Fig. 5. The *LTSelector* picks out the time differences dt_{ideal} from each of the exponential lifetime distributions (blue) considering the weights I_i. Moreover, the occurrences of prompt ($dt_{\text{ideal}} = 0$) and background events are considered.

(variables ***tauX*** (in nanoseconds) and ***intensityX,*** where X as an integer value between 1 and 5). Each component to be simulated must be enabled by setting the related variable ***ltX_activated*** *true*. The sum of all enabled intensities must be equal to one. Otherwise an error (*enum DLTErrorType*) is issued, which can be handled inheriting from the *class DLTCallback*. Furthermore, the library provides the possibility to take the occurrence of prompt events (variable ***intensityOfPromptOccurrance***) as well as an artificially generated background (variable ***intensityOfBackgroundOccurrance***) into account. The intensities of prompt and background events are treated separately. The lifetimes are generated to receive the start and stop pulses alternately from detector A and B (Fig. 2). This can be disabled by setting the variable ***isStartStopAlternating*** *false*.

2.2. Lifetime generation

2.2.1. The LTSelector — Picking out the ideal lifetime

The *LTSelector* relates to the *struct DLTSimulationInput* and picks out the ideal lifetimes dt_{ideal} with double precision from the exponential distributions (Fig. 5)

$$p(t|\lambda) = \lambda \exp\{-\lambda t\}, \quad \text{where } \lambda = \tau^{-1} \quad (6)$$

considering the given weights I_i and the occurrences of prompt (w_{prompt}) and background events (w_{bkgrd}) using the piecewise constant distribution function

$$p(x|b_0, \ldots, b_n, w_0, \ldots, w_{n-1}) = \frac{w_i}{\sum_{k=0}^{n-1} w_k(b_{k+1} - b_k)}. \quad (7)$$

For our case $b_{k+1} - b_k = 1$. The intensities I_i of N lifetime components τ_i (e.g. three components shown in Fig. 5), with $\sum_{i=0}^{N-1} I_i \equiv 1$, are scaled by the factor

$$w_{\text{LT}} = 1 - w_{\text{prompt}} - w_{\text{bkgrd}} \quad (8)$$

due to the condition

$$w_{\text{LT}} \sum_{i=0}^{N-1} I_i + w_{\text{prompt}} + w_{\text{bkgrd}} \equiv 1, \quad (9)$$

which then represents the true weights $w_{\text{LT}} \cdot I_i$ of the lifetime components for the *LTSelector*.

2.2.2. Adding the influences of the Instrumental Response

Subsequently, the ideal lifetime dt_{ideal} picked out from the *LTSelector* is separated into the start-

$$t_{\text{Start}} = \text{delay}$$

and stop-time stamp (Fig. 2)

$$t_{\text{Stop}} = dt_{\text{ideal}} + \text{delay} - \text{ATS},$$

where the defined trigger-delay (variable ***DLTPulse::delay***) represents the reference time stamp *delay* and *ATS* the Arrival Time Spread (variable ***DLTSetup::ATS***).

Table 1
Results of an experimentally obtained lifetime spectrum with PALS: 4N-Tin (τ_1) measured using Na-22 within a Kapton-foil (thickness: 7μ m) as positron-source (τ_2 represents the combined component originating from the Na-22 source and the Kapton-foil). The longest component τ_3 results from the lifetime of positrons which form hydrogen-like atoms (positronium) in between the source covering foil and the source (Na-22) and in between the source covering foil and the sample, respectively.

i	τ_i [ps]	I_i
1	192.59 ± 0.17	0.8342 ± 0.0021
2	385.26 ± 0.11	0.1616 ± 0.0018
3	3620 ± 24	0.00350 ± 0.00094
	σ [ps]	FWHM [ps]
	84.00 ± 0.12	197.80 ± 0.28

Finally, the Gaussian distributed (Eq. (2)) deviations resulting from the influences of the PDS and MU (*struct DLTSetup*) are added, which yields to the real lifetime dt_{real}. This leads to a smearing in the lifetime spectrum as shown in Fig. 2.

3. Illustrative example — Verification of the validity and functionality

To verify the validity and functionality of this library, a PALS spectrum of pure Tin (99.99% - 4N) was obtained using two photomultiplier (H1949-50/WA-5309 by *Hamamatsu*) with ultra-fast BC422Q (0.5% Benzophenone) scintillators and the DRS4 Evaluation Board [31] as MU, whose application for PALS was already shown by Petriska et al. [32] and Bin et al. [33].

The lifetime analysis of the obtained spectrum with the self-written *C++* software *DQuickLTFit*[4] [34], based on the algorithm as described by P. Kirkegaard et al. [20], resulted in the lifetime components listed in Table 1, which were then taken as input for the simulation. The *Hamamatsu* anode pulses were modeled by using a rise time (variable DLTPulse::***riseTime***) of 4.4 ns, a pulse width (variable DLTPulse::***pulseWidth***) of 0.14 ns and a maximum amplitude (DLTPulse::***amplitude***) of 500 mV as it is the measuring range of the DRS4 Evaluation Board. For both, the experiment and the simulation, a CFD-level of 25% (both branches A and B) and a cubic spline interpolation was used for the determination of the exact timing, which serves for the calculation of the time differences dt, i.e. the lifetimes forming the spectrum.

Measuring a split signal from one photomultiplier to both inputs A and B produces an artificial prompt signal (zero lifetime). Thus, the standard deviation σ_{MU} of the MU's IRF was experimentally determined to be $\sigma_{\text{MU}} = (13.16 \pm 1.21)$ ps. With the knowledge of the experimentally determined IRFs (Table 1 and MU), the standard deviations of the detectors were calculated as described in the following:

The IRFs of both branches A and B (g_{A} and g_{B}, respectively) (Fig. 2) are mathematically described by a convolution of the Gaussian distribution functions (Eq. (2)) of both MU and PDS

$$g_{\text{A}}\left(t|\mu_{\text{A}}, \sigma_{\text{A}}\right) = g\left(t|\mu^{\text{A}}_{\text{PDS}}, \sigma^{\text{A}}_{\text{PDS}}\right) * g\left(t|\mu_{\text{MU}}, \sigma_{\text{MU}}\right) \tag{10}$$

$$g_{\text{B}}\left(t|\mu_{\text{B}}, \sigma_{\text{B}}\right) = g\left(t|\mu^{\text{B}}_{\text{PDS}}, \sigma^{\text{B}}_{\text{PDS}}\right) * g\left(t|\mu_{\text{MU}}, \sigma_{\text{MU}}\right). \tag{11}$$

This results in Gaussian distribution function as well. Thus, the respective variances σ^2 are linked as followed:

$$\sigma^2_{\text{A}} = {\sigma^{\text{A}}_{\text{PDS}}}^2 + {\sigma^{\text{A}}_{\text{MU}}}^2 \tag{12}$$

$$\sigma^2_{\text{B}} = {\sigma^{\text{B}}_{\text{PDS}}}^2 + {\sigma^{\text{B}}_{\text{MU}}}^2. \tag{13}$$

[4] *DQuickLTFit* software [34] implements the *MPFIT* library [35] written in *C* for solving the non-linear least-square problem using the *Levenberg–Marquardt* algorithm. *MPFIT* is ported from *MINPACK-1 library* [36].

Table 2
Results of the simulated spectrum using the retrieved parameters from Table 1 as simulation input.

i	τ_i [ps]	I_i
1	192.34 ± 0.12	0.8324 ± 0.0023
2	384.33 ± 0.11	0.1634 ± 0.0021
3	3574 ± 20	0.00420 ± 0.00085
	σ [ps]	FWHM [ps]
	84.71 ± 0.10	199.48 ± 0.24

The final standard deviation σ_{result} of the lifetime spectrum, further the retrieved standard deviation from Table 1, is calculated by the Gaussian error propagation:

$$g_{\text{result}}\left(t\right): \sigma_{\text{result}} = \sqrt{{\sigma_{\text{A}}}^2 + {\sigma_{\text{B}}}^2}. \tag{14}$$

Under the ideal condition of using a complete symmetric setup (equivalent detectors for A and B):

$$g\left(t|\mu^{\text{A}}_{\text{PDS}}, \sigma^{\text{A}}_{\text{PDS}}\right) \equiv g\left(t|\mu^{\text{B}}_{\text{PDS}}, \sigma^{\text{B}}_{\text{PDS}}\right) \rightarrow \sigma_{\text{A}} \equiv \sigma_{\text{B}}, \tag{15}$$

and Eq. (14) simplifies to

$$g_{\text{result}}\left(t\right): \sigma_{\text{result}} = \sqrt{2}\sigma_{\text{A}} = \sqrt{2}\sigma_{\text{B}}. \tag{16}$$

With this assumption, the standard deviations of the photomultipliers (PDS) can be calculated from the experimentally obtained values, using Eqs. (12), (13), (16), to $\sigma_{\text{PDS}} = (59.38 \pm 1.21)$ ps.

In Fig. 6 the experimentally obtained and simulated spectra are compared. Both spectra are based on a statistic of approximately 12 Mio. counts acquired. The analysis of the simulated spectrum uses the same software and, thus, the same routine as in the analysis of the experimental spectrum (Table 1) by leaving all parameters free. The results are listed in Table 2 and indicate a good agreement with the experiment (compared to Table 1) and, therefore, verify the validity and functionality of *DLTPulseGenerator* library.

4. Impact and conclusion

Simulated positron lifetime spectra were typically used to evaluate the correctness and robustness of the analysis algorithms and to test the performance of new software [24,28,37]. First, simulations of multi-component spectra enabled the investigation of its decomposability [38,39] and, thus, to study the reliability and self-consistency of source corrections in PALS [40]. Therefore, different approaches were proposed for the generation of realistic lifetime spectra [38,39,41], where an analytical function according to Eqs. (1) and (2) containing Poisson noise and a constant background is the most commonly applied.

On the contrary, *DLTPulseGenerator* library uses a hardware-based approach and was developed to analyze the most significant influences resulting from the setup components (PDS and MU) and the given parameters, required for the determination of the exact timing signal (e.g. CFD level, number of pulse interpolation points), on the lifetime spectrum. The detector-output signal (non-TTL) is modeled by a log-normal distribution function (Eq. (4)), where the most significant parameters, rise time and pulse width, can be manipulated. An additional IRF spread could result from the interaction of the rise time and the sample speed of the acquisition electronics (ADC, FPGA) due to not-well matched values and, hence, it can be fully analyzed with respect to the pulse-interpolation/-fitting algorithm, the number of interpolation points and the CFD level. Furthermore, for energy resolved measurements, a pulse height spectrum (PHS), defined by a linear combination of two Gaussian distribution functions, is considered. In case of using plastic scintillators (e.g. BC422(Q)), Compton scattering becomes important and can be modeled well by overlapping the two Gaussian distribution functions of the start and stop branch, as seen in

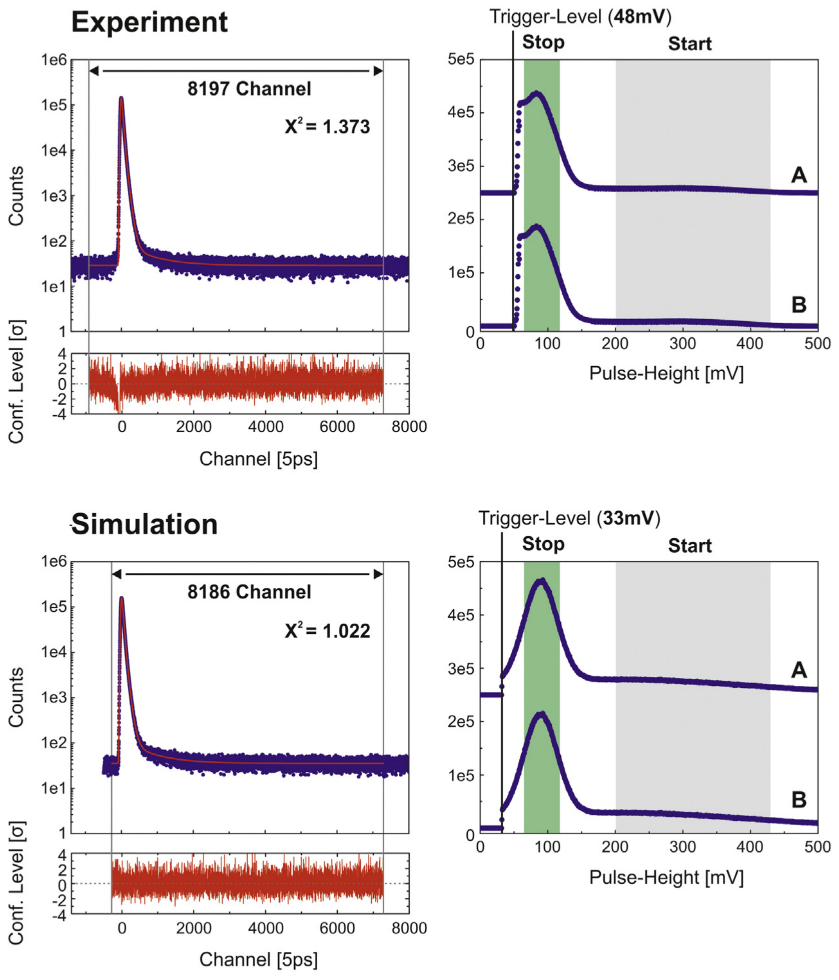

Fig. 6. Positron lifetime spectra (blue dots, left) and the recorded pulse height spectra (blue dots, right) of the experiment measuring tin with a purity of 99.99% (4N) (top), and the simulation (bottom), which is based on the parameters listed in Table 1. The model function from fitting is indicated by the red curve in the positron lifetime spectra (left). The resulting residuals are displayed below within the confidence levels of $\pm 4\sigma$. The pulse height spectrum (PHS) of the simulation was adapted to model the PHS of the used BC422Q (0.5% Benzophenone) scintillators best and, thus, the same windows for the start (gray shaded) and the stop quantum (green shaded) were chosen.

Figs. 4 and 6 (bottom-right). The pulses, resulting from Compton scattered photons, lead to a misinterpretation of being the start or stop signal and, thus, to an additional background and spread in the IRF. Therefore, the influences of the given windows (the accepted pulse heights for the start and stop energy) related to the IRF and the background are feasible to analyze.

Moreover, this library can be used for demonstration purposes or for the validation of algorithms on signal processing (e.g. filter, pulse interpolation or pulse fitting routines) or for spectra analysis in new software or libraries for lifetime spectroscopy.

Its validity and functionality were verified by PALS, measuring the lifetime spectrum of tin (Sn) with a purity of 99.99% (4N), where the experimentally retrieved lifetimes and intensities served as a simulation input subsequently.

References

[1] Bell RE, Graham RL. Time distribution of positron annihilation in liquids and solids. Phys Rev 1953;90:644–54. http://dx.doi.org/10.1103/PhysRev.90.644.

[2] Brandt W, Spirn I. Positron lifetime spectra in molecular substances. Phys Rev 1966;142:231–7. http://dx.doi.org/10.1103/PhysRev.142.231.

[3] Bergersen B, Stott MJ. The effect of vacancy formation on the temperature dependence of the positron lifetime. Solid State Commun 1969;7:1203–5. http://dx.doi.org/10.1016/0038-1098(69)90177-X.

[4] Connors DC, West RN. Positron annihilation and defects in metals. Phys Lett A 1969;30:24–5. http://dx.doi.org/10.1016/0375-9601(69)90018-8.

[5] Hautojärvi P, Tamminen A, Jauho P. Trapping of positrons by dislocations in aluminum. Phys Rev Lett 1970;24:459–61. http://dx.doi.org/10.1103/PhysRevLett.24.459.

[6] Crisp VHC, Lock DG, West RN. An investigation of positron trapping in indium. J Phys F Met Phys 1974;4:830–8. http://dx.doi.org/10.1088/0305-4608/4/6/011.

[7] Seeger A. Investigation of point defects in equilibrium concentrations with particular reference to positron annihilation techniques. J Phys F Met Phys 1973;3:248–94. http://dx.doi.org/10.1088/0305-4608/3/2/003.

[8] Seeger A. The study of defects in crystals by positron annihilation. Appl Phys 1974;4:183–99. http://dx.doi.org/10.1007/BF00884229.

[9] Seeger A. The temperature dependence of the diffusion coefficient and of the trapping rate of positrons in metals. Appl Phys 1975;7:257–63. http://dx.doi.org/10.1007/BF00900321.

[10] Saarinen K, Hautojärvi P, Corbel C. Positron annihilation spectroscopy of defects in semiconductors. In: Identif. Defects Semicond. Elsevier Ltd.; 1998. p. 209–85. http://dx.doi.org/10.1016/S0080-8784(08)63057-4.

[11] Krause-Rehberg R, Leipner HS. Positron annihilation in semiconductors: Defect studies. Berlin: Springer; 1999.

[12] Dlubek G, Kilburn D, Bondarenko V, Pionteck J, Krause-Rehberg R, Alam MA. Positron annihilation: A unique method for studying polymers. Macromolecular 2004;210:1–11.

[13] Harms S, Rätzke K, Faupel F, Schneider GJ, Willner L, Richter D. Free volume of interphases in model nanocomposites studied by positron annihilation lifetime spectroscopy. Macromolecules 2010;43:10505–11. http://dx.doi.org/10.1021/ma1022692.

[14] Zaleski R. Ortho-positronium localization in pores of vycor glass at low temperature. J Phys Conf Ser 2013;443:12062. http://dx.doi.org/10.1088/1742-6596/443/1/012062.

[15] Thränert S, Enke D, Dlubek G, Krause-Rehberg R. Positron lifetime spectroscopy on controlled pore glass porosimetry and pore size distribution.

Mater Sci Forum 2009;607:169–72. http://dx.doi.org/10.4028/www.scientific.net/MSF.607.169.

[16] Festy F, Ameer-Beg SM, Ng T, Suhling K, Spagnuolo C, Martinez OE, Jares-Erijman EA, Jovin TM, van Driel R, Chernoff J, Burshtyn DN, Frend PMW, Davis DM, Sandison A, Wallace A, Davis DM, Lever J, Neil MAA, Phillips D, Stamp GW, French PMW. Imaging proteins in vivo using fluorescence lifetime microscopy. Mol Biosyst 2007;3:381. http://dx.doi.org/10.1039/b617204k.

[17] Gura Sadovsky R, Brielle S, Kaganovich D, England JL. Measurement of rapid protein diffusion in the cytoplasm by photo-converted intensity profile expansion. Cell Rep 2017;18:2795–806. http://dx.doi.org/10.1016/j.celrep.2017.02.063.

[18] Willoughby RA. Solutions of ill-posed problems (A. N. Tikhonov and V. Y. Arsenin). SIAM Rev 1979;21:266–7. http://dx.doi.org/10.1137/1021044.

[19] Kirkegaard P, Eldrup M. POSITRONFIT: A versatile program for analysing positron lifetime spectra. Comput Phys Comm 1972;3:240–55. http://dx.doi.org/10.1016/0010-4655(72)90070-7.

[20] Kirkegaard P, Eldrup M. Positronfit extended: A new version of a program for analysing position lifetime spectra. Comput Phys Comm 1974;7:401–9. http://dx.doi.org/10.1016/0010-4655(74)90070-8.

[21] Kirkegaard P, Eldrup M, Mogensen OE, Pedersen NJ. Program system for analysing positron lifetime spectra and angular correlation curves. Comput Phys Comm 1981;23:307–35. http://dx.doi.org/10.1016/0010-4655(81)90006-0.

[22] Olsen JV, Kirkegaard P, Pedersen NJ, Eldrup M. PALSfit: A new program for the evaluation of positron lifetime spectra. Phys Status Solidi 2007;4:4004–6. http://dx.doi.org/10.1002/pssc.200675868.

[23] Mostgaard M, Kirkegaard P, Olsen JV, Eldrup M. PALSfit3: A software package for analysing positron lifetime spectra. Kgs. Lyngby: Technical University of Denmark (DTU; 2017.

[24] Kansy J. Microcomputer program for analysis of positron annihilation lifetime spectra. Nucl Instrum Methods Phys Res Sect A 1996;374:235–44. http://dx.doi.org/10.1016/0168-9002(96)00075-7.

[25] Giebel D, Kansy J. A new version of LT program for positron lifetime spectra analysis. Mater Sci Forum 2010;666:138–41. http://dx.doi.org/10.4028/www.scientific.net/MSF.666.138.

[26] PicoQuant GmbH. FluoFit - Global fluorescence decay data analysis software. n.d.; https://www.picoquant.com/products/category/software/fluofit-global-fluorescence-decay-data-analysis-software.

[27] PicoQuant GmbH. SymPhoTime 64 - Fluorescence lifetime imaging and correlation software. n.d.; https://www.picoquant.com/products/category/software/symphotime-64-fluorescence-lifetime-imaging-and-correlation-software.

[28] Shukla A, Peter M, Hoffmann L. Analysis of positron lifetime spectra using quantified maximum entropy and a general linear filter. Nucl Instrum Methods Phys Res Sect A 1993;335:310–7. http://dx.doi.org/10.1016/0168-9002(93)90286-Q.

[29] Shukla A, Hoffmann L, Manuel AA, Peter M. Melt 4.0 a program for positron lifetime analysis, 255–257 (1997) 233–237. http://dx.doi.org/10.4028/www.scientific.net/MSF.255-257.233.

[30] Haaks M, Valentini R, Vianden R. First test of LSO scintillators for positron lifetime spectroscopy. Phys Status Solidi 2007;4:4036–9. http://dx.doi.org/10.1002/pssc.200675869.

[31] Ritt S. Design and performance of the 6 GHz waveform digitizing chip DRS4. In: IEEE Nucl Sci Symp Conf Rec. IEEE; 2008. p. 1512–5. http://dx.doi.org/10.1109/NSSMIC.2008.4774700.

[32] Petriska M, Sojak S, Slugeň V. Positron lifetime setup based on DRS4 evaluation board. J Phys Conf Ser 2014;505:12044. http://dx.doi.org/10.1088/1742-6596/505/1/012044.

[33] Bin C, Liu Y-F, Bang-Jiao Y, Kong W, Ritt S. A new positron annihilation lifetime spectrometer based on DRS4 waveform digitizing board. Chinese Phys C 2014;38:56001. http://dx.doi.org/10.1088/1674-1137/38/5/056001.

[34] Petschke D. dpscience/dquickltfit: DQuickLTFit v2.06. 2018, http://dx.doi.org/10.5281/ZENODO.1168286.

[35] Markwardt CB. Non-linear least squares fitting in IDL with MPFIT. Astron Data Anal Softw Syst XVIII ASP Conf Ser 2009;411:251.

[36] Moré JJ. The Levenberg–Marquardt algorithm: Implementation and theory. Berlin, Heidelberg: Springer; 1978. p. 105–16. http://dx.doi.org/10.1007/BFb0067700.

[37] Gregory RB, Zhu Y. Analysis of positron annihilation lifetime data by numerical laplace inversion with the program contin. Nucl Instrum Methods Phys Res Sect A 1990;290:172–82. http://dx.doi.org/10.1016/0168-9002(90)90358-D.

[38] Somieski B, Staab TEM, Krause-Rehberg R. The data treatment influence on the spectra decomposition in positron lifetime spectroscopy Part 1: On the interpretation of multi-component analysis studied by Monte Carlo simulated model spectra. Nucl Instrum Methods Phys Res Sect A 1996;381:128–40. http://dx.doi.org/10.1016/0168-9002(96)00584-0.

[39] Sormann H, Kindl P, Puff W. Investigations on the reliability of a multi-component analysis of positron lifetime spectra, using a new method of producing computer-simulated test spectra. Nucl. Instrum Methods Phys Res 1983;206:203–9. http://dx.doi.org/10.1016/0167-5087(83)91260-7.

[40] Staab TEM, Somieski B, Krause-Rehberg R. The data treatment influence on the spectra decomposition in positron lifetime spectroscopy Part 2: The effect of source corrections. Nucl. Instrum Methods Phys Res Sect A 1996;381:141–51. http://dx.doi.org/10.1016/0168-9002(96)00585-2.

[41] Lawther DW, Gaudin A, Stewart A. Simulated positron lifetime (SimPL): A Monte Carlo simulator for positron diffusion, trapping, and lifetime spectra. Phys Status Solidi 2007;4:4007–10. http://dx.doi.org/10.1002/pssc.200675768.