sensors

MDPI

Article

A Comparative Study of Regression Methods for Solving the Timepix Calibration Task

Jan Broulím [1,2,*], Matěj Prokop [1], Libor Nouzák [2] and Pavel Smrčka [1]

[1] Czech Technical University, Jugoslavskych Partyzanu 1580/3, 16000 Prague, Czech Republic; prokom12@fjfi.cvut.cz (M.P.); smrcka@fbmi.cvut.cz (P.S.)
[2] Charles University, Ovocny trh 560/5, 11636 Prague, Czech Republic; libor.nouzak@mff.cuni.cz
* Correspondence: jan.broulim@cvut.cz

Abstract

In this article, we provide a study of the energy calibration model used for Timepix-type detectors. The Timepix detectors, operating in Time-over-Threshold mode, measure information that needs to be mapped into the corresponding energies using a non-linear function. We consider three iterative algorithms, Gradient-Descent, Gauss–Newton and Levenberg–Marquardt algorithm, which we modify according to the calibration model constraints to perform better in terms of the convergence properties. Moreover, based on the variable projection method, we suggest a partial linearization of the calibration problem and provide results for this novel method.

Keywords: Timepix; Timepix3; Timepix4; Levenberg–Marquardt; regression; optimization algorithms; variable projection; calibration

check for updates

Academic Editor: Guillermo Villanueva

Received: 5 July 2025
Revised: 18 October 2025
Accepted: 23 October 2025
Published: 3 November 2025

Citation: Broulím, J.; Prokop, M.; Nouzák, L.; Smrčka, P. A Comparative Study of Regression Methods for Solving the Timepix Calibration Task. *Sensors* **2025**, *25*, 6714. https://doi.org/10.3390/s25216714

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The goal was to create an efficient and robust algorithm for fitting the energy Time-over-Threshold (ToT) calibration curve to data measured by a specific pixel of the Timepix-type detector. The problem requires constrained non-linear regression, for which several algorithms were implemented.

This article provides calibration solutions based on various mathematical algorithms and validates the effectiveness of the calibration.

1.1. Timepix Detectors

Timepix-type detectors [1–4] are hybrid pixel detectors based on the readout chip [1] and a bump-bonded sensor layer, typically made of Si, SiC, GaAs, or CdTe. They contain a matrix of pixels with the size of 55 μm × 55 μm per pixel and a granularity of 256 × 256 for Timepix, Timepix2, and Timepix3, or 512 × 448 for Timepix4. Each pixel represents an independent spectroscopic channel, providing information about the energy and time of acquired events in terms of the ToT and the Time of Arrival (ToA) values. Since the Timepix3 generation, both values can be measured simultaneously [3,5,6].

When measuring in the ToT mode, the detector counts the number of clock cycles during a certain period depending on the programmable threshold and deposited energy. The ToT value represents the incremental number of clock cycles during which the incoming signal remains above a settable threshold. The signal consists of two parts: a rapid rise as a result of charging the capacitor, followed by a slower discharge. The amplitude is proportional to the energy deposited. When the amplified signal exceeds the settable

threshold, the ToT counter is incremented based on the ticks of the clock. When the signal is below the threshold, it does not contribute to the measurement, which introduces non-linearity in the calibration function. This process is illustrated in Figure 1. The wire-bonded sensor is shown in Figure 2.

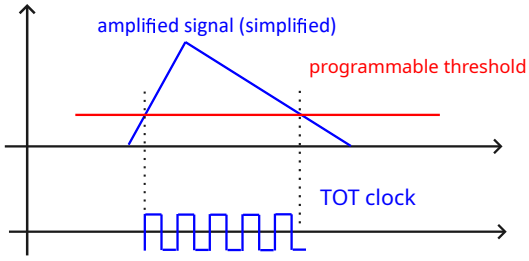


Figure 1. A simplified principle of measuring in the ToT mode.

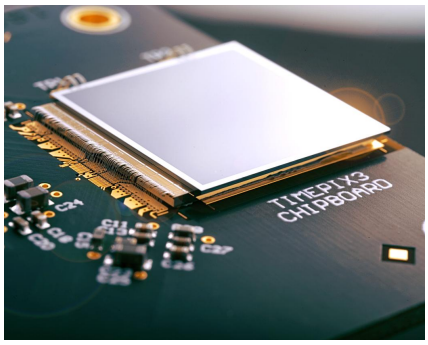


Figure 2. The Timepix3 wire-bonded on the chipboard [7].

The calibration setup, measurement conditions, task definition, and algorithms used are listed below.

1.2. Calibration Setup

The setup consists of an X-ray tube, metal foils, and a Timepix detector. The X-ray tube irradiates the metal foil, producing characteristic X-rays at known energies. The characteristic X-rays are acquired by the Timepix detector. Alternatively, the small calibration sources are used.

Pixel detectors exploit the charge-sharing effect [8], which means that the deposited energy is spread over multiple pixels. This creates multi-pixel events, which we call clusters. Based on their morphology, we can distinguish the type of particle [9–11]. The sum of the energies per cluster (track) corresponds to the total deposited energy. This type of calibration is applied to the energies up to 1 MeV, before the volcano effect appears [12–14]. The volcano effect is caused by saturation within the pixel ASIC, resulting in two pulses in the preamplifier. The calibration fit for these energies differs from the equation used in this paper. The term volcano refers to the blobs observed from heavy, high-energy particles, which show lower acquired values at their centers.

In this work, we are interested in single-pixel events, where all deposited energy is acquired within a single pixel. This is common for X-rays. The measured value, called Time-over-Threshold (ToT), is mapped onto a known energy expressed in keV. A filter to exclude multi-pixel events is applied. The filter searches for hits in neighboring pixels within a given time window, and those hits are deleted.

We acquired several datasets; each dataset corresponds to a material producing X-rays (Cd, Ti, Cu, and Zr) and another dataset for the Am-241 source. Since we are interested only in single-pixel events, we can consider that measurements were taken at 65,536 channels, which corresponds to the matrix size. For five calibration points, this results in five datasets,

each measured at 65,536 channels. The Fe-55 dataset was dropped due to low statistics. We also worked with the second detector to confirm and validate the measurements.

Figures 3 and 4 show acquired histograms of ToT values for Am-241. We are interested in the last peak, which corresponds to the full energy deposition at 59.6 keV, which is around the maximum energy for single pixel hits in the Si sensor used. Other peaks and Compton scattering can also be identified in the spectrum. Figure 3 depicts values acquired within one selected pixel, while Figure 4 shows the sum of events in the entire matrix. The spectra after the calibration, i.e., when ToT is recalculated to energies, are plotted in Figure 5 for Am-241 and in Figure 6 for Cd irradiated by the X-ray tube. The principle of the calibration is described in the following section, and the energies for these spectra are obtained using Equation (29). The energy resolution of the detector is expressed using σ or Full Width at Half Maximum (FWHM), which quantifies the peak widths at particular energies.

Tables 1 and 2 list the calibration energies. For calibration purposes, we are interested in the peak positions. We can search for the maximal values or fit peaks with the use of a Gaussian approximation, which may provide better results by means of estimating peak positions between the two ToT integer values. This approach can produce slightly better results compared to the method that searches for the maximal values within histograms.

Table 1. Selected characteristic X-rays commonly used for calibration.

Material	$K\alpha_1$
Ti	4.51 keV
Cu	8.05 keV
Zr	15.78 keV
Cd	23.17 keV
In	24.21 keV

Table 2. Calibration sources.

Material	γ Ray Energy
Am-241	59.6, 26.3, 20.7 keV
Fe-55	5.89 keV

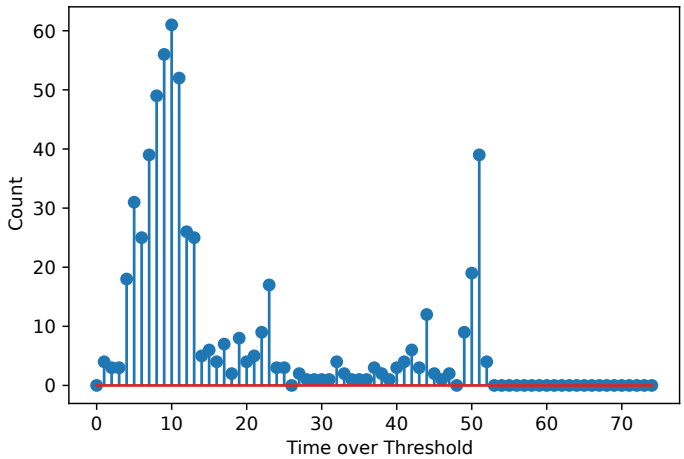


Figure 3. Acquired points based on the measurement with the Am-241 source. The spectrum shows the ToT values for one selected pixel. The red color represents the baseline (also in the following figures).

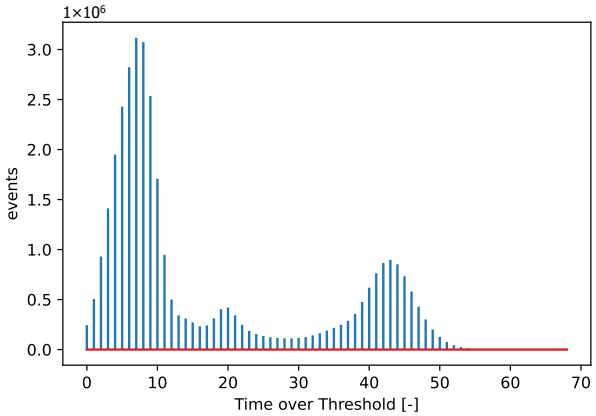


Figure 4. Acquired points based on the measurement with the Am-241 source. The spectrum shows a summation of ToT values for all pixels.

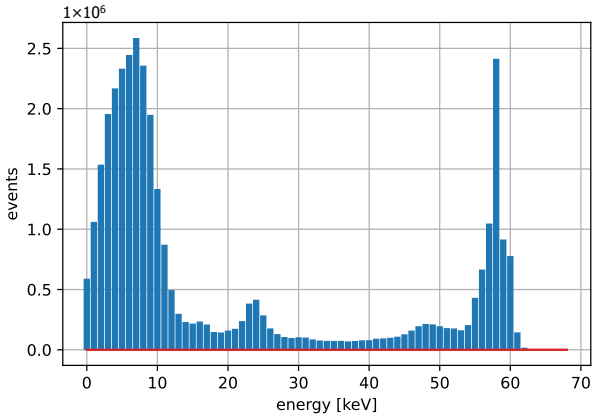


Figure 5. Acquired data points are based on measurements with the Am-241 source after the calibration. The spectrum represents the sum of the energy values from all pixels, expressed in keV. A clear 59.6 keV peak is observed, which appears significantly narrower when compared to the corresponding ToT values.

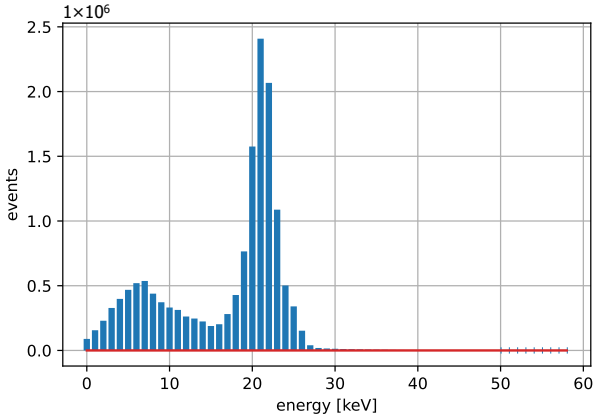


Figure 6. The spectrum obtained for Cd producing characteristic X-rays. All pixels are summed up.

2. Task Definition

The task is to model the dependence between measured ToT values and the energy expressed in keV. According to the detector description and the electronics used we expect the following relation [15–20]:

$$y = f(x, \boldsymbol{\beta}) = \beta_a x + \beta_b - \frac{\beta_c}{x - \beta_d} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)$ are the measured energies expressed in keV, $y = (y_1, y_2, \ldots, y_n)$ are the corresponding measured ToT values (non-negative). This equation is not valid at high energies (above 1 MeV), where ASIC saturation occurs. The calibration fit in this region is almost linear, but higher energies correspond to lower ToT values.

The task is to determine coefficients $\boldsymbol{\beta} = (\beta_a, \beta_b, \beta_c, \beta_d)$ that minimize the value of the cost function $F(\boldsymbol{\beta})$, given by the RSS value (Residual Sum of Squares):

$$F(\boldsymbol{\beta}) = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - f(x_i, \boldsymbol{\beta}))^2. \tag{2}$$

The values $r_i = y_i - f(x_i, \boldsymbol{\beta})$ are referred to as i-th component of the residual vector $\boldsymbol{r}$.

The value of β_d solution is limited by the condition that it should be less than the minimum of a measured calibration point, as follows:

$$\beta_d < \min_i(x_i) \tag{3}$$

Also, large negative values can signal a problematic solution. The value of β_a and β_c are expected to be positive, written as follows:

$$\beta_a > 0, \tag{4}$$

$$\beta_c > 0. \tag{5}$$

3. Iterative Algorithms

We have evaluated the performance of the following three algorithms [21,22] for calibrating detectors:

- Gradient Descent (GD),
- Levenberg–Marquardt (LM),
- Gauss–Newton (GN).

Iterative algorithms start with setting the initial guess $\boldsymbol{\beta}^{(0)}$. Then, they iterate in order to update the current estimate, with the goal to lower the cost function. The next iteration estimate during the algorithm run is calculated as follows:

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k + \boldsymbol{\delta}^k, \tag{6}$$

where $\boldsymbol{\delta}^k$ is the step updating k-th iteration estimate. Iterative algorithms differ in the way of calculating $\boldsymbol{\delta}^k$.

3.1. Gradient Descent

A gradient descent is the algorithm for finding the local maximum or minimum of a given function. We minimize the Function (2), based on performing steps against its gradient. The value of $\boldsymbol{\delta}^{(k)}$ for the gradient descent algorithm is given by the equation:

$$\boldsymbol{\delta}^{(k)} = -\gamma^{(k)} \nabla F(\boldsymbol{\beta})^{(k)}, \tag{7}$$

where γ is called the step size. In the simplest case, it could have a fixed value, independent of k. However, for better performance, we use a backtracking line search to find the value of the step size [23]. For simplification, we omit k, denoting the current iteration index, in the following expressions. At the beginning of each iteration, we start with γ_0 (the

maximum step size), which is a fixed algorithm parameter. Then we multiply γ with the fixed adjusting constant $\tau < 1$, written as follows:

$$\gamma_{i+1} = \tau\gamma_i, \tag{8}$$

while the condition given by (Armijo–Goldstein condition negation) [23]

$$F(\boldsymbol{\beta} - \gamma\boldsymbol{\nabla}F(\boldsymbol{\beta})) > F(\boldsymbol{\beta}) - \frac{1}{2}\gamma||\boldsymbol{\nabla}F(\boldsymbol{\beta})||^2, \tag{9}$$

is satisfied. When this process is finished, we have found our best step size value estimate, and we perform the step with this value. This method provides a general approach for solving the tasks; however, other methods exhibit better convergence properties, as shown in the following figures.

3.2. Gauss–Newton

The Gauss–Newton algorithm is used for solving non-linear least squares problems. Given the residual functions $\boldsymbol{r} = (r_1, r_2, ..., r_m)$ and variables $\boldsymbol{\beta} = (\beta_i)$, the value of $\boldsymbol{\delta}^{(k)}$ is found by solving the matrix equation given by the following:

$$(\boldsymbol{J}^T\boldsymbol{J})\boldsymbol{\delta}^{(k)} = \boldsymbol{J}^T\boldsymbol{r}^{(k)}. \tag{10}$$

Assuming Equation (6), we can express the next $\boldsymbol{\beta}$ values as follows:

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k + (\boldsymbol{J}^T\boldsymbol{J})^{-1}\boldsymbol{J}^T\boldsymbol{r}^{(k)}. \tag{11}$$

where $\boldsymbol{J}$ is the Jacobian matrix. The Formula (10) allows us to find desired values directly from solving the system of linear equations, which can be performed faster than the calculation of the inverse matrix (11).

3.3. Levenberg–Marquardt (LM)

The idea of Levenberg–Marquardt is to improve the convergence of the Gauss–Newton method by introducing a damping parameter [22]. The algorithm is based on the $\boldsymbol{\delta}$ calculation from the matrix equation given by the following:

$$(\boldsymbol{J}^T\boldsymbol{J} + \lambda^{(k)}\boldsymbol{I})\boldsymbol{\delta}^{(k)} = \boldsymbol{J}^T\boldsymbol{r}^{(k)}, \tag{12}$$

where $\boldsymbol{I}$ is the identity matrix and λ is called the damping parameter.

For the adaptation of the damping parameter, we use the method proposed by Marquardt [22]. Let $v > 1$ be the adjusting constant of the algorithm, and let λ^0 be the initial damping parameter of the algorithm.

At the beginning of the k-th iteration, let $\lambda^{(k-1)}$ be the damping parameter value from the previous iteration, or we set $\lambda^{(-1)} = \lambda_{initial}$ for the first iteration. Our goal is to find the best damping parameter for the current iteration, denoted as $\lambda^{(k)}$ ($k = 0, 1, 2, ...$). Consider the cost function $F_{\lambda^{(k-1)}/v}$ calculated for $\lambda^{(k-1)}/v$. Then the following algorithm is used [22]:

- If $F_{\lambda^{(k-1)}/v} \leq F_{\lambda^{(k-1)}}$ we set $\lambda^{(k)} = \lambda^{(k-1)}/v$;
- If $F_{\lambda^{(k-1)}/v} > F_{\lambda^{(k-1)}}$ and $F_{\lambda^{(k-1)}} \leq F(\boldsymbol{\beta}^{(k)})$ we set $\lambda^{(k)} = \lambda^{(k-1)}$;
- If $F_{\lambda^{(k-1)}/v} > F_{\lambda^{(k-1)}}$ and $F_{\lambda^{(k-1)}} > F(\boldsymbol{\beta}^{(k)})$ we multiply $\lambda^{(k-1)}$ by v for some smallest w until $F_{\lambda^{(k-1)}v^w} \leq F(\boldsymbol{\beta}^{(k)})$ is satisfied. Then we set $\lambda^{(k)} = \lambda^{(k-1)}v^w$.

3.4. Analytical Derivations Used for Calculations

Here we derive partial derivatives, which are required for the algorithm implementations.

3.4.1. Jacobian Matrix

Jacobian matrix $\boldsymbol{J}$ is a matrix whose columns are the gradients of the regression function for the i-th data point. That is described by the following equations:

$$J_{i1} = \frac{\partial f}{\partial \beta_a}(x_i, \boldsymbol{\beta}) \tag{13}$$

$$J_{i2} = \frac{\partial f}{\partial \beta_b}(x_i, \boldsymbol{\beta}) \tag{14}$$

$$J_{i3} = \frac{\partial f}{\partial \beta_c}(x_i, \boldsymbol{\beta}) \tag{15}$$

$$J_{i4} = \frac{\partial f}{\partial \beta_d}(x_i, \boldsymbol{\beta}), \tag{16}$$

where the partial derivatives within the Jacobian matrix are expressed as

$$\frac{\partial f}{\partial \beta_a}(x_i, \boldsymbol{\beta}) = x_i \tag{17}$$

$$\frac{\partial f}{\partial \beta_b}(x_i, \boldsymbol{\beta}) = 1 \tag{18}$$

$$\frac{\partial f}{\partial \beta_c}(x_i, \boldsymbol{\beta}) = -\frac{1}{x_i - \beta_d} \tag{19}$$

$$\frac{\partial f}{\partial \beta_d}(x_i, \boldsymbol{\beta}) = -\frac{\beta_c}{(x_i - \beta_d)^2} \tag{20}$$

3.4.2. Gradient of the Cost Function

The analytic expression for components of the cost function gradient can be found by the differentiation of Equation (2).

$$\frac{\partial F}{\partial \beta_a}(\boldsymbol{\beta}) = -2\sum_{i=1}^{n} x_i y_i + 2\beta_a \sum_{i=1}^{n} x_i^2 + \tag{21}$$

$$+2\beta_b \sum_{i=1}^{n} x_i - 2\beta_c \sum_{i=1}^{n} \frac{x_i}{x_i - \beta_d} \tag{22}$$

$$\frac{\partial F}{\partial \beta_b}(\boldsymbol{\beta}) = -2\sum_{i=1}^{n} y_i + 2\beta_a \sum_{i=1}^{n} x_i + \tag{23}$$

$$+2\beta_b n - 2\beta_c \sum_{i=1}^{n} \frac{1}{x_i - \beta_d} \tag{24}$$

$$\frac{\partial F}{\partial \beta_c}(\boldsymbol{\beta}) = 2\sum_{i=1}^{n} \frac{y_i}{x_i - \beta_d} - 2\beta_a \sum_{i=1}^{n} \frac{x_i}{x_i - \beta_d} - \tag{25}$$

$$-2\beta_b \sum_{i=1}^{n} \frac{1}{x_i - \beta_d} + 2\beta_c \sum_{i=1}^{n} \frac{1}{(x_i - \beta_d)^2} \tag{26}$$

$$\frac{\partial F}{\partial \beta_d}(\boldsymbol{\beta}) = 2\beta_c \sum_{i=1}^{n} \frac{y_i}{(x_i - \beta_d)^2} - 2\beta_a\beta_c \sum_{i=1}^{n} \frac{x_i}{(x_i - \beta_d)^2} \tag{27}$$

$$-2\beta_b\beta_c \sum_{i=1}^{n} \frac{1}{(x_i - \beta_d)^2} + 2\beta_c^2 \sum_{i=1}^{n} \frac{1}{(x_i - \beta_d)^3} \tag{28}$$

The sums from this equation that are independent of $\boldsymbol{\beta}$ can be precomputed. The sums dependent on $\boldsymbol{\beta}$ must be computed again with every iteration.

3.5. Algorithm Modifications

We modify the algorithms to have better convergence properties. We must consider that the algorithm can converge to a different minimum than the desired one.

We must avoid cases where the algorithm terminates outside the expected bounds by properly identifying poor convergence behavior through consideration of the bounds (3)–(5). For improving convergence and enlarging the region of initial estimates, which terminates in good solutions, we suggest using *randomization* step during the algorithm run. If the $\boldsymbol{\beta}$ vector goes out of the given bounds, random numbers are added to its elements.

All iterative algorithms are performed within a cycle until the maximal number of iterations is reached. This stopping condition is further extended by checking the convergence criteria, e.g., the gradient of the cost function or the iteration step size. A Gauss–Newton algorithm is described in terms of pseudo code in Algorithm 1. We calculate $\boldsymbol{\delta}$ according to Equation (10) in order to avoid the calculation of the inverse matrix when $\boldsymbol{\beta}$ is expressed directly Equation (11). The values of the **J** matrix are expressed analytically. The Gradient-Descent algorithm implementation is an intuitive modification of the pseudo code. Algorithm 2 shows the LM regression, extended by our randomization. Compared to the GN algorithm, it uses a damping parameter to improve the convergence. Both algorithms provide $\boldsymbol{\beta}$ coefficients as the output.

The comparative results for the algorithms are shown in the following sections. The cases of wrong convergence, incorrect β_d values, and the algorithm comparisons are also included in the study.

Algorithm 1 Iterative algorithm—GN.

input: $\boldsymbol{\beta_0}$
output: $\boldsymbol{\beta}$ {vector of 4 coefficients}
$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta}_0$; {initial guesses, initial damping parameter}
$k \leftarrow 0$; {iteration index}
while $(k < k_{max})$ **do**
  $\boldsymbol{\delta} \leftarrow$ solve $\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{\delta} = \boldsymbol{J}^T\boldsymbol{r}$
  $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \boldsymbol{\delta}$
  if stoppingCriterion$(\boldsymbol{\delta}, \boldsymbol{\beta})$ **then**
    break {stops updating if the delta value or the gradient of the cost function is below a given threshold}
  end if
  $k \leftarrow k + 1$
end while
return $\boldsymbol{\beta}$

Algorithm 2 Iterative algorithm—LM.

input: $\lambda_{initial}$; v; $\boldsymbol{\beta_0}$
output: $\boldsymbol{\beta}$ {vector of 4 coefficients}
$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta}_0$; $\lambda \leftarrow \lambda_{initial}$; {initial guesses, initial damping parameter}
$k \leftarrow 0$; {iteration index}
while $(k < k_{max})$ **do**
  $\boldsymbol{J}^T \leftarrow$ jacobianTransposed$(\boldsymbol{\beta})$
  $\boldsymbol{r} \leftarrow$ residualsVector$(\boldsymbol{\beta})$
  $\boldsymbol{\delta_1} \leftarrow$ solve $(\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{I})\boldsymbol{\delta} = \boldsymbol{J}^T\boldsymbol{r}$
  $\boldsymbol{\delta_2} \leftarrow$ solve $(\boldsymbol{J}^T\boldsymbol{J} + \frac{\lambda}{v}\boldsymbol{I})\boldsymbol{\delta} = \boldsymbol{J}^T\boldsymbol{r}$
  if $F(\boldsymbol{\beta} + \boldsymbol{\delta_1}) \geq F(\boldsymbol{\beta} + \boldsymbol{\delta_2})$ **then**
    $\lambda \leftarrow \frac{\lambda}{v}$
    $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \boldsymbol{\delta_2}$ {a part specific for LM regression}
  else if $F(\boldsymbol{\beta} + \boldsymbol{\delta_1}) \leq F(\boldsymbol{\beta})$ **then**
    $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \boldsymbol{\delta_1}$
  else
    while $F(\boldsymbol{\beta} + \boldsymbol{\delta_1}) > F(\boldsymbol{\beta})$ **do**
      $\lambda \leftarrow \lambda v$
      $\boldsymbol{\delta_1} \leftarrow$ solve $(\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{I})\boldsymbol{\delta} = \boldsymbol{J}^T\boldsymbol{r}$ {this is recommended to limit by the maximum number of steps}
    end while
    $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \boldsymbol{\delta_1}$
  end if
  if !insideConstraints$(\boldsymbol{\beta})$ **then**
    $\boldsymbol{\beta} \leftarrow$ randomize()
  else if isSolution$(\boldsymbol{\beta})$ **then**
    return $\boldsymbol{\beta}$
  end if
  $k \leftarrow k + 1$
end while
return

4. Results for Iterative Algorithms

Figure 7 shows a comparison for the LM algorithm when sweeping with the damping parameter. We can see that the high values of the damping parameter can slow down the convergence. While the damping parameter equal to 10 provides very slow but stable convergence, the damping parameter equal to 0.01 results in unstable convergence at the beginning, followed by a rapid decrease in the cost function. The introduction of the damping parameter extends the GN algorithm and improves its properties.

The fitted curve is shown in Figure 8 for the two different algorithms. The comparison depicts the iteration count dependence of the fitted curve for the GD and the LM algorithms. We can recognize that the convergence of the LM algorithm is much faster than the convergence of the GD algorithm. A significant difference is observed in the comparison, where the LM algorithm performed better after 50 iterations compared to the GD algorithm after 2000 iterations.

In Figures 9 and 10, we compare the dependency of the cost function value on the iterations performed. We show results for two different initial guesses. Figure 9 corresponds to the initial guess $\boldsymbol{\beta}_0 = (0, 0, 25, 1)$, Figure 10 to $\boldsymbol{\beta}_0 = (0, 0, 5, 1)$. In the first case the LM algorithm produced the best results (with the fixed damping used). A comparison of the different adjusting constants for the LM algorithm is plotted in Figure 11 with the initial guess $\boldsymbol{\beta}_0 = (1, 1, 5, 1)$. The filter that selects the minimum cost up to the current iteration is applied.

In Figures 12–15 we show the histogram of the cost function for all pixels. Figures 13 and 15 show results after 5 iterations performed. The convergence behavior of

the three algorithms was illustrated in Figure 16 in terms of the 3D plot. The convergence trajectory is indicated in Figure 17.

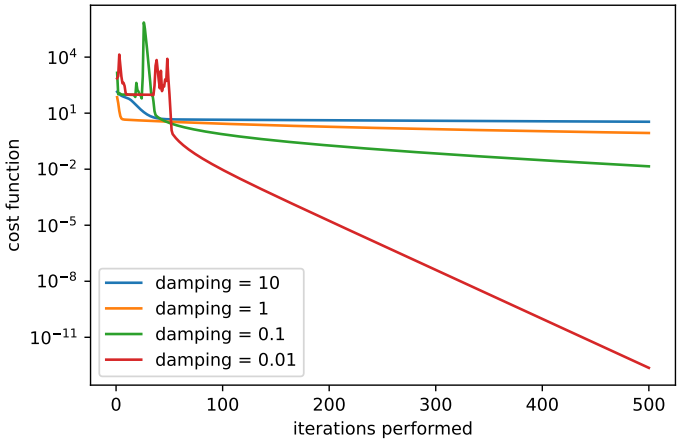


Figure 7. The convergence of LM algorithm, comparison for various damping parameters, $\boldsymbol{\beta}_0 = (0, 0, 25, 1)$.

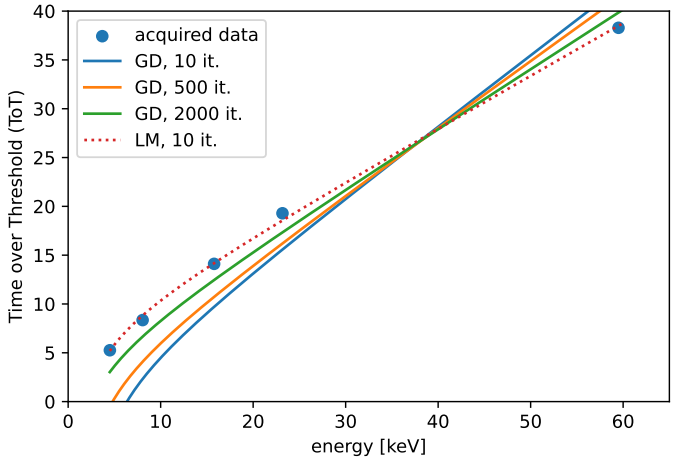


Figure 8. The convergence of the GD algorithm, comparison with the LM algorithm. The initial guesses are set to $\boldsymbol{\beta}_0 = (0, 0, 25, 1)$.

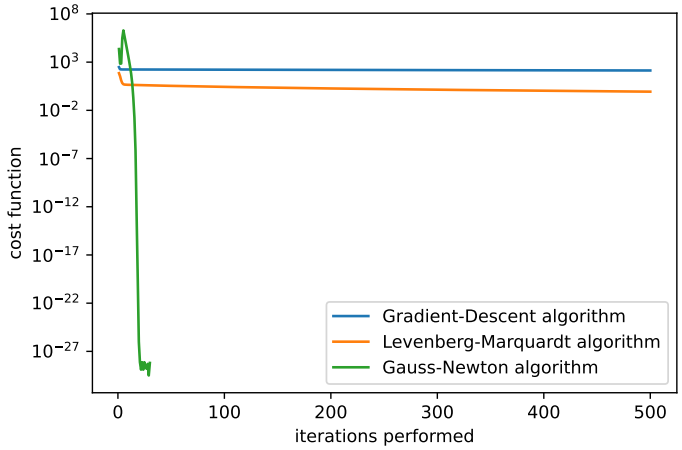


Figure 9. The cost function of algorithms. The initial guesses set to $\boldsymbol{\beta}_0 = (0, 0, 25, 1)$, damping parameter = 1, rate = 0.0001. We can see that the Gauss–Newton algorithm can in some cases converge very quickly.

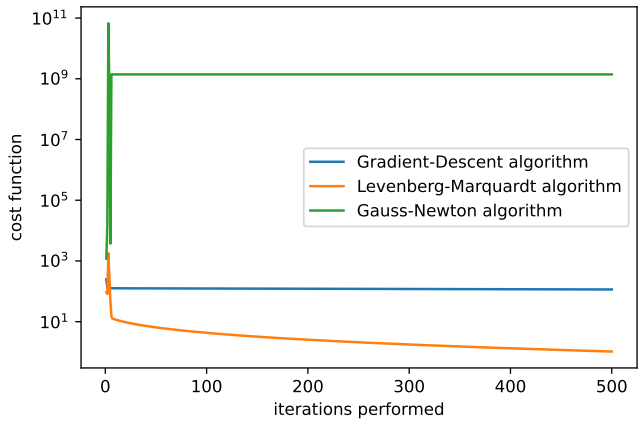


Figure 10. The cost function of the algorithms. The initial guesses are set to $\beta_0 = (0, 0, 5, 1)$, damping parameter = 1, and rate = 0.0001. We can recognize the best performance of the LM algorithm.

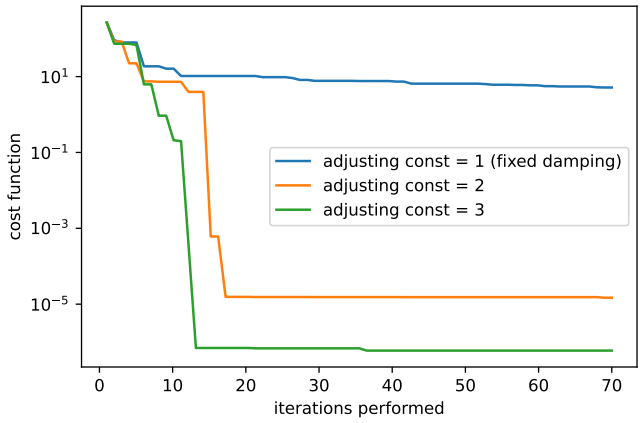


Figure 11. The cost function of the LM algorithm with the different adjusting constants. The initial guesses set to $\beta_0 = (1, 1, 5, 1)$ are used and the initial damping is set to 1.

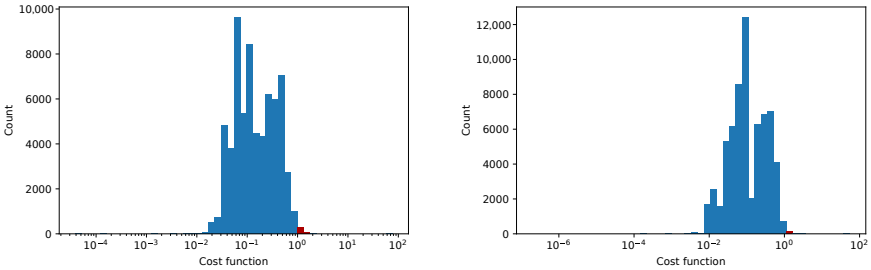


Figure 12. The convergence of LM algorithm after 50 iterations, all pixels included, a comparison for various damping parameters is shown, the initial guesses were $\beta_0 = (0, 0, 0, 1)$. The (**left**) plot shows fixed steps, the (**right**) picture uses adjusting constant equal to 5.

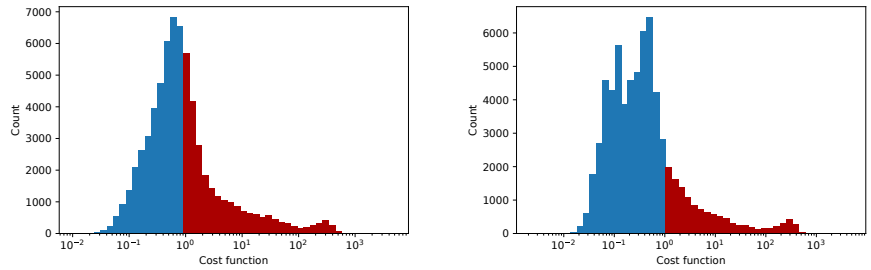


Figure 13. The convergence of the LM algorithm after 5 iterations, all pixels are included, a comparison for various damping parameters is shown, the initial guesses were $\beta_0 = (1, 1, 25, 1)$. The (**left**) plot shows the fixed steps, the (**right**) picture uses adjusting constant equal to 3.

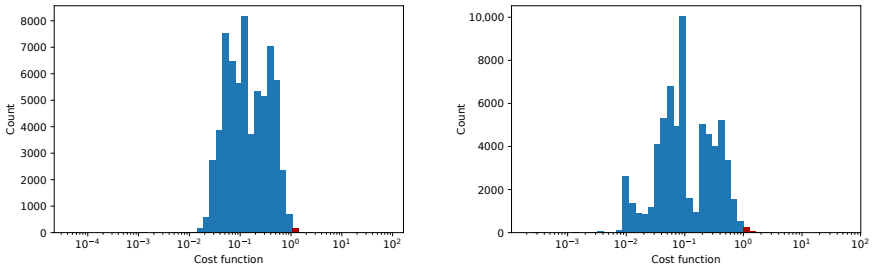


Figure 14. The convergence of the LM algorithm after 50 iterations, all pixels are included, a comparison for various damping parameters is shown, the initial guesses were $\boldsymbol{\beta}_0 = (1, 1, 25, 1)$. The (**left**) plot shows fixed steps, the (**right**) picture uses adjusting constant equal to 3.

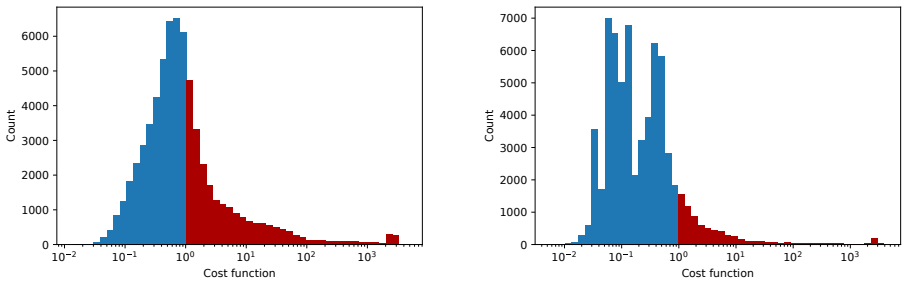


Figure 15. The convergence of the LM algorithm after 5 iterations, all pixels are included, comparison for various damping parameters is shown, the initial guesses were $\boldsymbol{\beta}_0 = (0, 0, 0, 1)$. The (**left**) plot shows fixed steps, the (**right**) picture uses adjusting constant equal to 5.

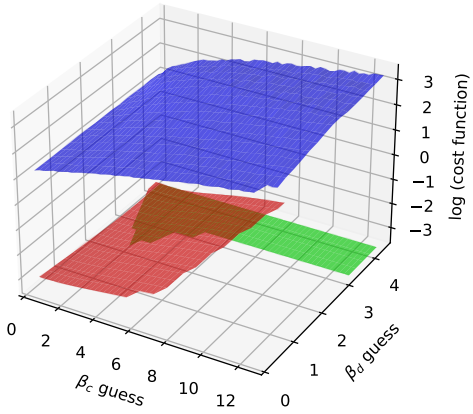


Figure 16. The convergence plot for the three different algorithms after 100 iterations. The red color stands for the LM algorithm (fixed damping 1), the green color for the GN algorithm, the blue color for the GD algorithm (maximal rate 100, adjusting 0.5). The initial guesses for $\beta_a = 1$, $\beta_b = 1$. Figures 29–31.

Energy Spectra

The calibration function converting the energy E to the ToT value is expressed in Equation (1).

We denote $t = F(E)$, $E > \beta_d$, and derive the inversion function as follows:

$$E = \frac{t + \beta_a\beta_d - \beta_b \pm \sqrt{(t - \beta_a\beta_d - \beta_b)^2 + 4\beta_a\beta_c}}{2\beta_a}, \tag{29}$$

where t is the ToT value.

Roots of the quadratic formula corresponds to positive and negative brach of the calibration function, $E_+ \geq \beta_d$ and $E_- \leq \beta_d$.

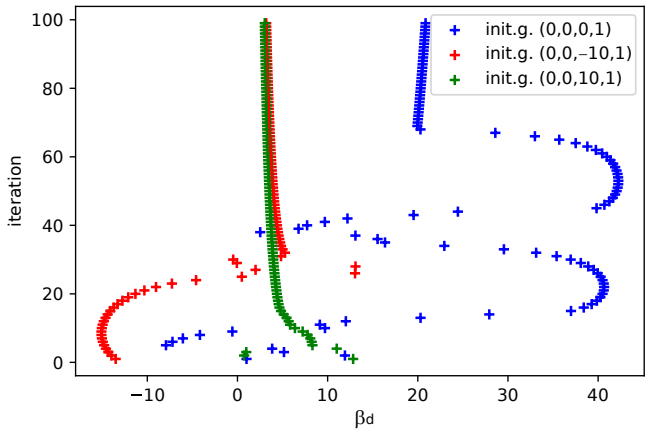


Figure 17. Visualization of the convergence trajectory in terms of β_d value, the LM algorithm is used. The blue trajectory shows a failure in convergence, however, it can be easily identified as β_d cannot be between measured points.

Comparisons for different iterative algorithms and different numbers of iterations are shown in Figures 18 and 19, with Ti as the source of X-rays. The center value of the peak for Ti, determined using the LM algorithm with 50 iterations, is located between 4.0 and 4.5 keV for all initial guesses except for $\boldsymbol{\beta}_0 = (0, 0, 50, 10)$. Considering the resolution and the associated uncertainty, the result is acceptable. We observe shifts of peak centers towards the desired energies with an increasing number of iterations. The LM algorithm is highlighted for its superior performance. The calibrated peaks in the energy spectrum are plotted in Figure 20. The Gaussian fits and the centers of the corresponding peaks are shown in Figure 21 and in Table 3, where we compare the results for several initial guesses.

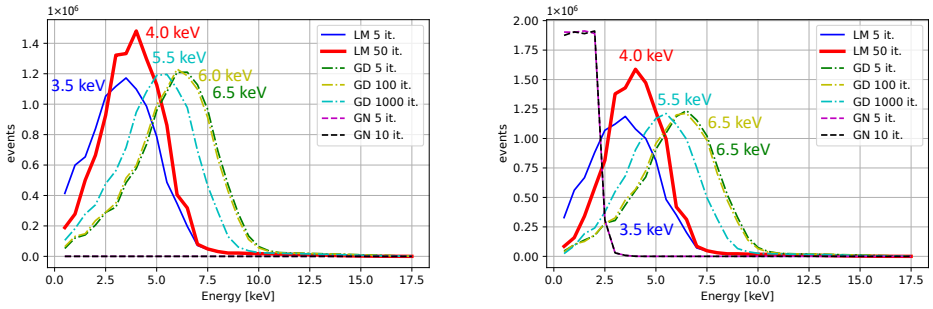


Figure 18. The measurement with Ti after the calibration. The initial guesses were $\boldsymbol{\beta}_0 = (0, 0, 0, 0)$ for the left plot and $\boldsymbol{\beta}_0 = (5, 0, 0, 1)$. The peak maxima with respect to the binning are indicated (no fitting was applied).

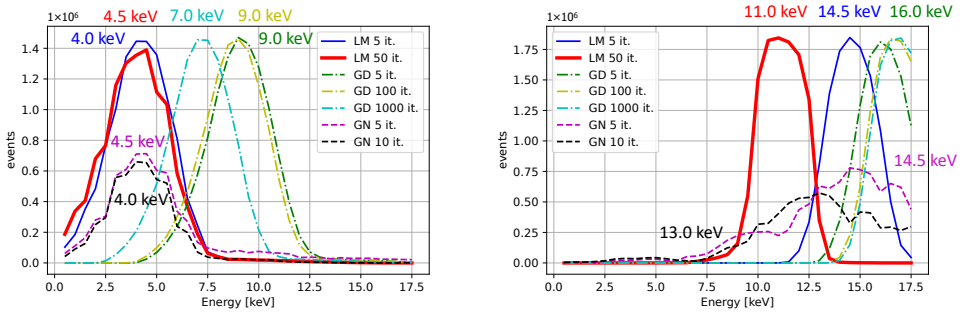


Figure 19. The measurement with Ti after the calibration. The initial guesses were $\boldsymbol{\beta}_0 = (0, 0, 25, 0)$ for the left plot and $\boldsymbol{\beta}_0 = (0, 0, 50, 10)$. The peak maxima with respect to the binning are indicated (no fitting was applied).

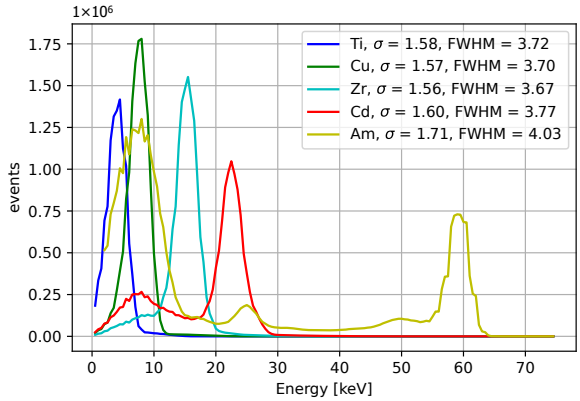


Figure 20. The energy spectrum for 4 X-rays datasets measured X-rays and the Am-241 source. The LM algorithm at 50 iterations was used, the initial guess were $\boldsymbol{\beta}_0 = (0, 0, 20, 0)$.

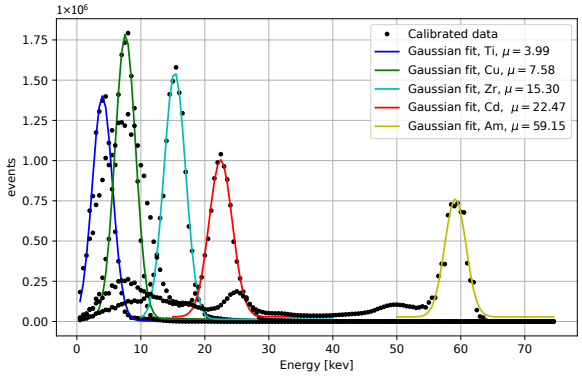


Figure 21. Gaussian fits for the energy spectrum for 4 X-rays datasets measured X-rays and the Am-241 source. The LM algorithm at 50 iterations was used, the initial guess were $\boldsymbol{\beta}_0 = (0, 0, 20, 0)$. The peak centers obtained from the fits are included.

Table 3. Selected peak centers after applying the LM algorithm with 5 iterations. The last column corresponds to 50 iterations of the LM algorithm. The damping parameter was set to 1.

$\boldsymbol{\beta}_0$:	$(0, 0, 20, 1)$	$(0, 0, 50, 1)$	$(0, 0, 50, 5)$	$(0, 0, 0, 0)$	$(0, 0, 50, 5)$, 50 it.
Ti	4.23 keV	4.5 keV	8.15 keV	3.29 keV	4.96 keV
Cu	7.55 keV	7.64 keV	9.82 keV	7.58 keV	7.54 keV
Zr	15.10 keV	14.78 keV	16.17 keV	15.83 keV	15.59 keV
Cd	22.26 keV	21.8 keV	22.99 keV	23.06 keV	22.90 keV
In	59.27 keV	59.52 keV	58.49 keV	58.84 keV	58.97 keV

5. Variable Projection

Inspired by the work on variable projection [24], we introduce a method, which moves the calibration partially towards analytical solution. Suppose that the optimal cost function depends only on β_d. Then, β_a, β_c, and β_d are given as the solution to the linear least squares problem with dependence on β_d.

We define a reduced vector $\boldsymbol{\beta}_{reduced} = (\beta_a(\beta_d), \beta_b(\beta_d), \beta_c(\beta_d))$ satisfying the minimum of the cost function (2).

According to the derivations in Section 3, we can express the solution as follows:

$$\begin{pmatrix} \sum y_i x_i \\ \sum y_i \\ \sum k_i y_i \end{pmatrix} = \begin{pmatrix} \sum x_i^2 & \sum x_i & -\sum x_i k_i \\ \sum x_i & n & -\sum k_i \\ \sum x_i k_i & \sum k_i & -\sum k_i^2 \end{pmatrix} \begin{pmatrix} \beta_a \\ \beta_b \\ \beta_c \end{pmatrix} \tag{30}$$

where $k_i = \frac{1}{x_i - \beta_{d0}}$.

The associated reduced cost function is then written as follows:

$$s_{nl}(\beta_d) = \sum_{i=1}^{n} \left(y_i - \beta_a(\beta_d) x_i - \beta_b(\beta_d) + \frac{\beta_c(\beta_d)}{x_i - \beta_d} \right)^2. \tag{31}$$

For $\forall \beta_d \in [0, x_{min})$ the reduced cost function s_{nl} is a differentiable function of the variable β_d.

If no stationary point lies in the interval $(0, x_{min})$, the extrema of $s_{nl}(\beta_d)$ within the interval $[0, x_{min})$ must lie on the boundary $\beta_d = 0$ or $\beta_d \to x_{min}$. This is because $s_{nl}(\beta_d)$ is differentiable on the $[0, x_{min})$ interval.

The variable projection method can be extended for $\beta_d < 0$ by adding the additional condition ensuring $E \geq 0$ to the linear system with the variable $\boldsymbol{\beta}_{reduced}$. The case $\beta_c = 0$ corresponds to a linear fit.

We show the plots obtained by two types of datasets. The first dataset provides a minimum within the expected bounds. The cost function is shown in Figure 22. We can see that the calibration fit corresponds to the measured points.

A good fit for 5 points is shown in Figure 23. For the practical use of the algorithm, we must ensure that β_c is positive. In Figure 24 we can see two minima, local and global; however, both are out of bounds. The corresponding plot of the fits is shown in Figure 25. The red and cyan curves show the fits, which are not correct. If no minimum is found inside the given bounds, we must select a β_d point within the bounds, reasonably close to the boundary with the minimal cost value. Then, we have to choose β_d within the expected bounds. The plot of all pixels after the variable projection method is performed is shown in Figure 26. Statistical parameters for the LM algorithm and the variable projection method as a box plot are shown in Figure 27. The coefficient values are shown in Figure 28.

The results for the variable projection method in the selected area of the matrix are shown in Figures 29–31. The cost function and β_d and β_c values are plotted. The bound conditions are applied within the algorithm. The energy spectra for different β_d values are depicted in Figure 32, where Ti is used as the X-ray source. The plot in Figure 16 corresponds to the pixel coordinates $(x, y) = (0, 3)$.

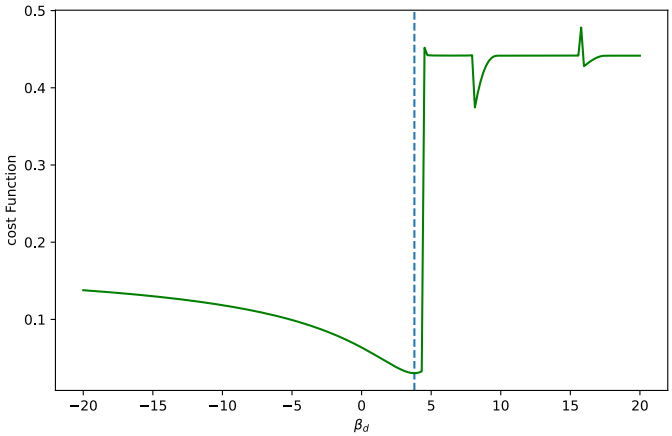


Figure 22. The cost function of the analytic solutions for varying β_d values, while a good dataset is processed. Only one minimum is observed. In cases of poor data quality, we must ensure that $\beta_d > 0$. The upper bound is given by the first calibration point. If $\beta_d > 0$ exceeds this limit, the fit results in an infinite discontinuity. The dotted line marks to the minimum value.

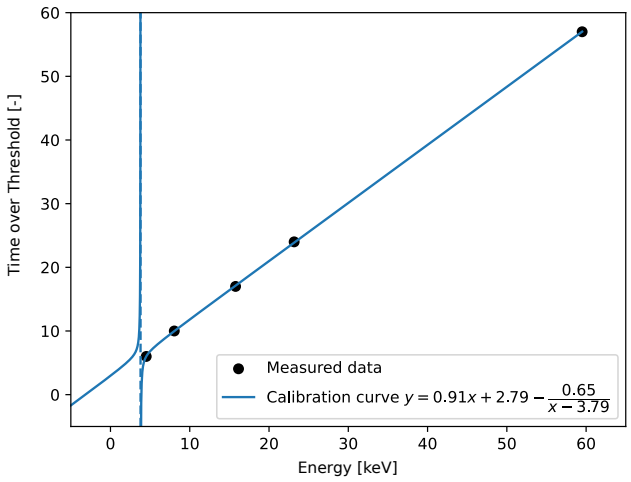


Figure 23. The fit for good data, the resulting equation is provided. The fit corresponds to the pixel (0, 3) in the related matrix plots. The blue dotted line indicates the discontinuity in the fit. Its position is given by the value of β_d.

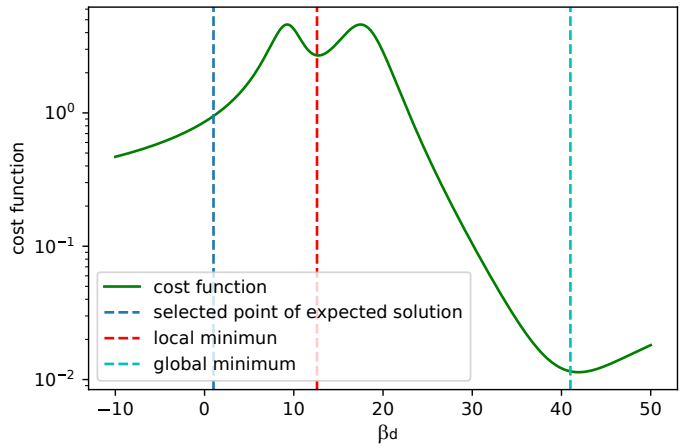


Figure 24. The cost function calculated analytically for given β_d values. This example shows a case, where minima do not correspond to the proper fits. A suitable β_d is around the blue line (corresponding picture is in Figure 25).

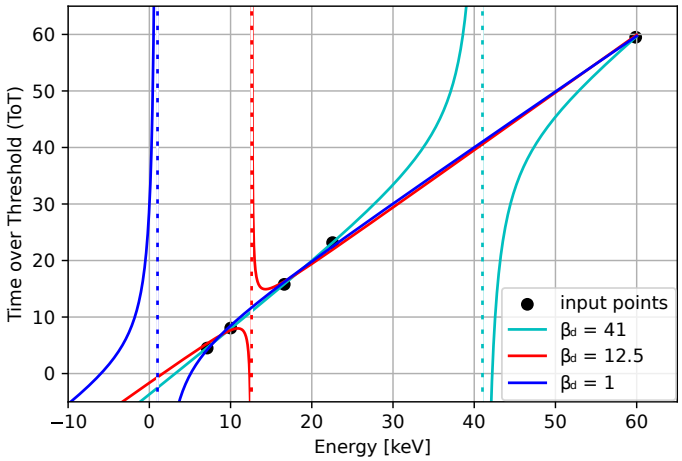


Figure 25. Fitted curves for fixed β_d. The red color ($\beta_d = 12.5$) corresponds to a local minimum of the cost function, and the cyan curve ($\beta_d = 41$) corresponds to the global minimum of the cost function. Both are incorrect, as is evident from the figure. Here we have to fix β_d to the expected value, as shown in the blue function.

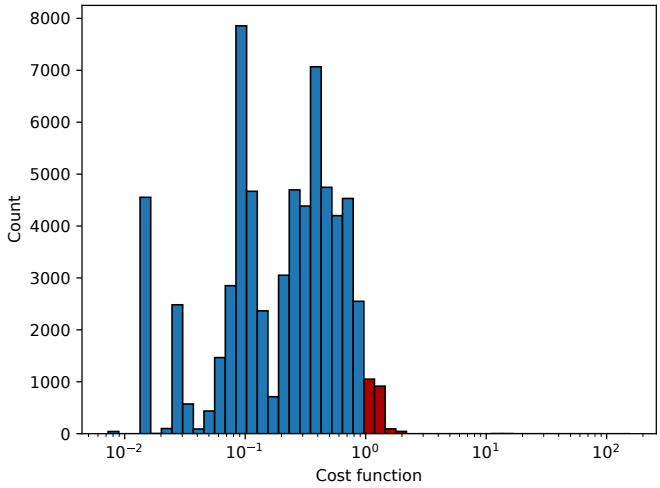


Figure 26. The cost function for all pixels when the variable projection method is used. The value β_c is bounded to be larger than 0.

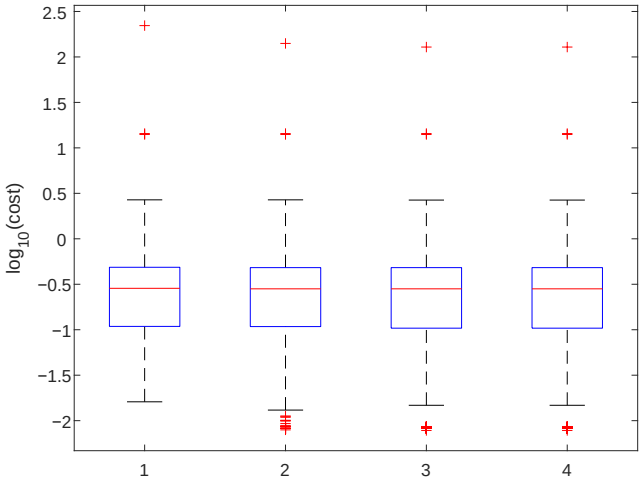


Figure 27. The box plot for the cost function. We show values respectively as follows: 1—LM (damping 1, adjusting constant 2), 10 iterations; 2—LM, 50 iterations; 3—LM—1000 iterations; and 4—variable projection, $\beta_c > 0$.

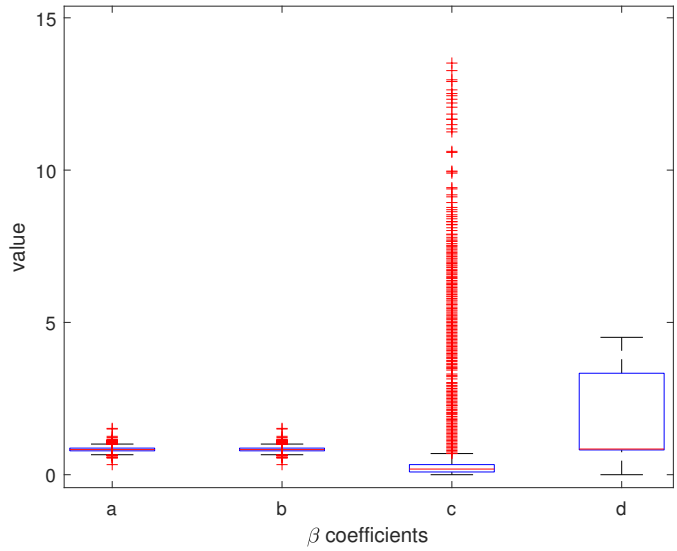


Figure 28. The box plot for β_a, β_b, β_c, and β_d coefficients. The LM algorithm at 50 iterations and randomized initial guesses were used.

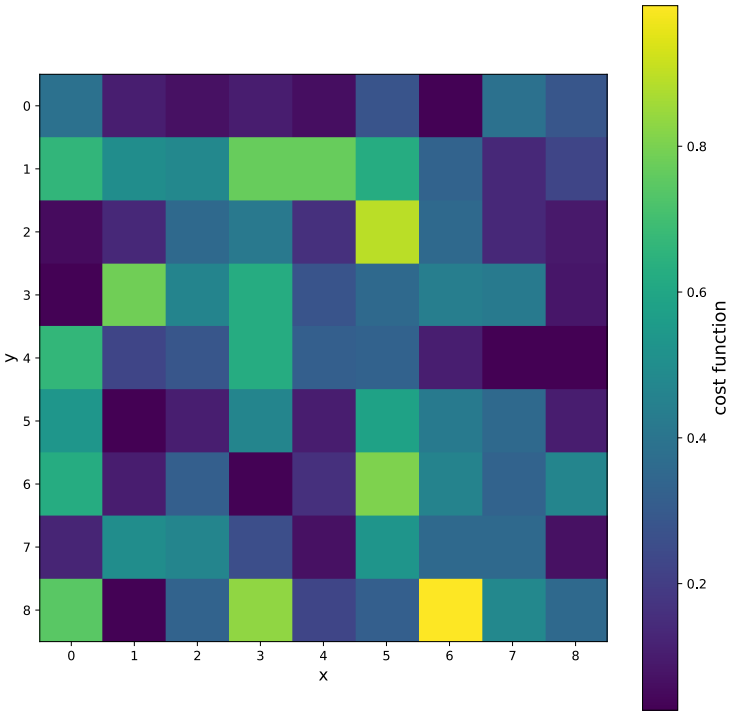


Figure 29. The selected area of the matrix, where x and y are pixel coordinates. The cost function for several pixels when the variable projection method is used.

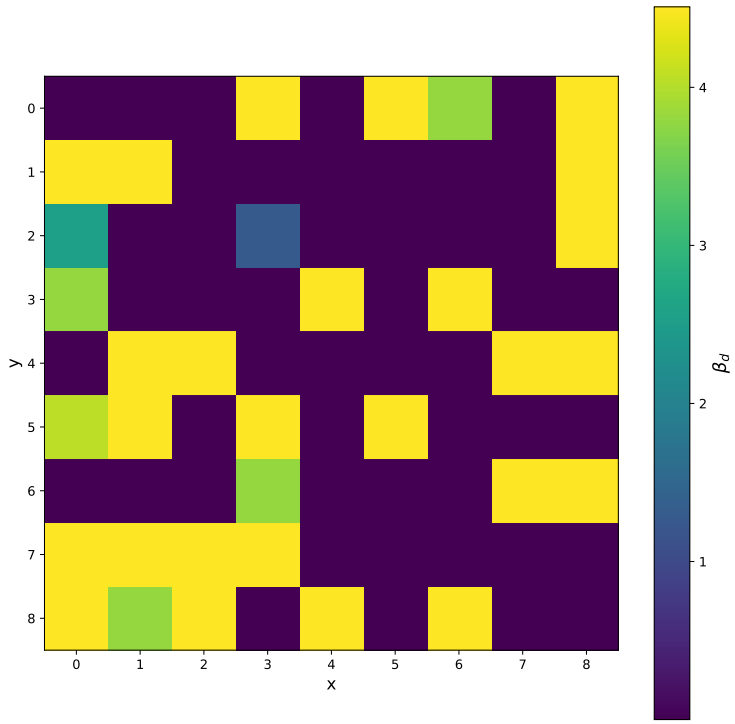


Figure 30. The selected area of the matrix, where x and y are pixel coordinates. The β_d for all pixels when the variable projection method is used. We can see the range of β_d values lower than the minimal measured energy point, corresponding to the bound (3).

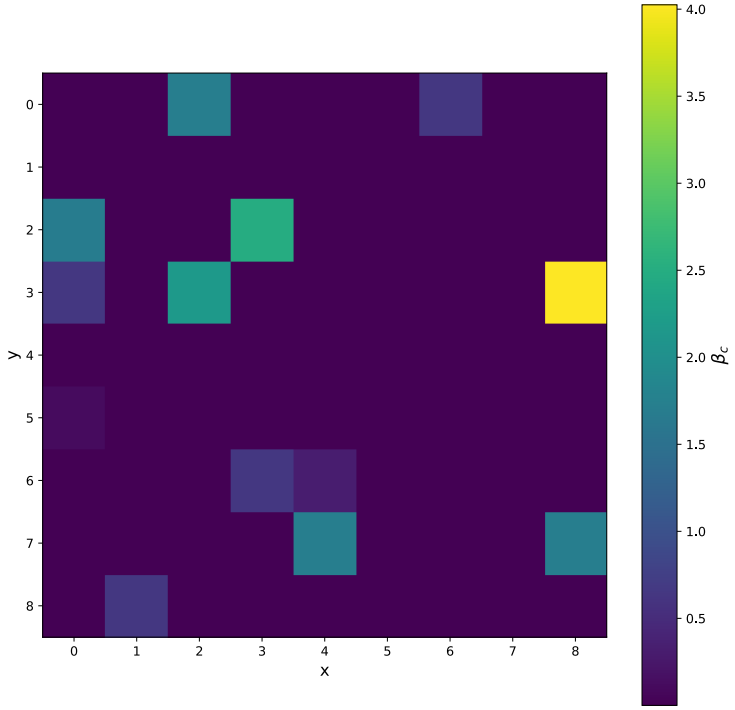


Figure 31. The selected area of the matrix, where x and y are pixel coordinates. The $\beta_c > 0$ for all pixels when the variable projection method is used.

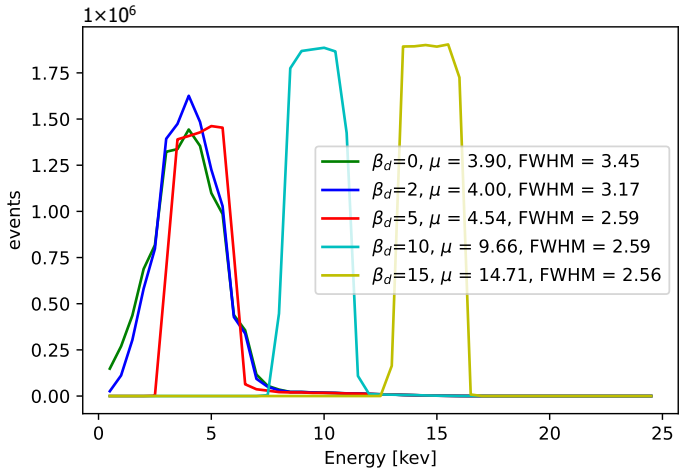


Figure 32. The comparison in terms of energy spectra for different β_d values, when we fix this parameter (no other constrains were applied). Ti X-rays were used. The parameter β_d was fixed to be the same constant for pixels. Better results can be obtained when the value β_d is not fixed for all pixels and is instead determined by searching through the cost function. The cost function takes values around 1×10^{-1} for $\beta_d = 2$ and $\beta_d = 5$ and increases several times for $\beta_d = 15$. The calibration points, different for each pixel, correspond to the peaks in the ToT values of each pixel.

6. Conclusions

In this article, we studied a calibration model commonly used for the Timepix detectors with several approaches to solve non-linear regression problems. We show the convergence of the algorithms in terms of the cost function dependence on the iteration number and verify the area of initial guesses that converge to a good solution. Compared with the other two iterative algorithms, the LM regression with the proposed randomization tends to perform the best in all tested cases. The convergence trajectories for the LM algorithm were also plotted. The modified version with the additional randomization provided better results. Generally, the LM algorithm applied together with bounds checking is highly

recommended to use. The GD algorithm is an abstract algorithm, not limited to solving least squares problems. Its convergence is slower and more dependent on the parameters compared to the LM algorithm.

As an alternative approach, we have also shown that the calibration can be solved as a variable projection into a one-dimensional optimization problem. For the given β_d values we can calculate β_a, β_b, and β_c as the least squares problem solution. The β_d value must be lower than the first measured energy. The main advantage of this method is that it does not need the initial guesses as the algorithm input, while the iterative algorithms need guesses in all their modifications. Moreover, it can resolve convergence problems of the iterative algorithms.

The future work will be focused on the speed optimizations and implementation of the accelerated version using parallel algorithms. Moreover, χ^2 minimization is a method worth considering for studies and comparisons.

Author Contributions: J.B. contributed to conducting experiments, analysis, writing, and conceptualization; M.P. contributed variable projection derivation, analysis, and validation, L.N. contributed to the validation; P.S. contributed to the methodology. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data are currently not publicly available. Please contact the authors for access.

Acknowledgments: We would like to thank the Medipix Collaboration at CERN for the opportunity to participate in this research. We extend our thanks to the AANL laboratory in Yerevan, particularly to Hrachya Marukyan, for making it possible to conduct the measurements. We also appreciate the welcoming environment of the restaurant Hamburg and Centrala Club in Prague, as well as Charlie's Burger & Bar in Pilsen, which supported the preparation of this article. The authors also thank Josef Janecek for useful advice in the topic of Timepix detectors; Jan Jakubek for his never-ending contributions to the Medipix Collaboration; and Farrukh Waheed for his contributions during the initial phase of this work.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ToT	Time-over-Threshold
GD	Gradient Descent
LM	Levenberg–Marquardt
GN	Gauss–Newton
RSS	Residual Sum of Squares
FWHM	Full Width at Half Maximum

References

1. Llopart, X.; Ballabriga, R.; Campbell, M.; Tlustos, L.; Wong, W. Timepix, a 65k Programmable Pixel Readout Chip for Arrival Time, Energy and/or Photon Counting Measurements. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2007**, *581*, 485–494. [CrossRef]
2. Llopart, X.; Alozy, J.; Ballabriga, R.; Campbell, M.; Casanova, R.; Gromov, V.; Heijne, E.; Poikela, T.; Santin, E.; Sriskaran, V.; et al. Timepix4, a Large Area Pixel Detector Readout Chip Which Can Be Tiled on 4 Sides Providing Sub-200 ps Timestamp Binning. *J. Instrum.* **2022**, *17*, C01044. [CrossRef]
3. Poikela, T.; Plosila, J.; Westerlund, T.; Campbell, M.; De Gaspari, M.; Llopart, X.; Gromov, V.; Kluit, R.; van Beuzekom, M.; Zappon, F.; et al. Timepix3: A 65K Channel Hybrid Pixel Readout Chip with Simultaneous ToA/ToT and Sparse Readout. *J. Instrum.* **2014**, *9*, C05013. [CrossRef]
4. Bromberger, H.; Pennicard, D.; Ballabriga, R.; Trippel, S.; Küpper, J. Timepix3: Single-Pixel Multi-Hit Energy-Measurement Behaviour. *J. Instrum.* **2024**, *19*, P11008. [CrossRef]

5. Turecek, D.; Jakubek, J.; Pospisil, S.; Vykydal, Z.; Kroupa, M. Pixelman: A Multi-Platform Data Acquisition and Processing Software Package for Medipix2, Timepix and Medipix3 Detectors. *J. Instrum.* **2011**, *6*, C01046. [CrossRef]
6. Broulim, J.; Broulim, P.; Burian, P.; Holik, M.; Mora, Y.; Pospisil, S.; Solar, M. j-Pix—A Multiplatform Acquisition Package for Timepix3. *J. Instrum.* **2019**, *14*, C06004. [CrossRef]
7. CERN Knowledge Transfer. Timepix3. Available online: https://kt.cern/technologies/timepix3 (accessed on 5 July 2025).
8. Jakubek, J. Energy-Sensitive X-Ray Radiography and Charge Sharing Effect in Pixelated Detector. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2009**, *607*, 192–195. [CrossRef]
9. Holy, T.; Heijne, E.; Jakubek, J.; Pospisil, S.; Uher, J.; Vykydal, Z. Pattern Recognition of Tracks Induced by Individual Quanta of Ionizing Radiation in Medipix2 Silicon Detector. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2008**, *591*, 287–290. [CrossRef]
10. Granja, C.; Solc, J.; Gajewski, J.; Rucinski, A.; Stasica, P.; Rydygier, M.; Marek, L.; Oancea, C. Composition and Spectral Characterization of Mixed-Radiation Fields with Enhanced Discrimination by Quantum Imaging Detection. *IEEE Trans. Nucl. Sci.* **2024**, *71*, 921–931. [CrossRef]
11. Granja, C.; Jakubek, J.; Köster, U.; Platkevic, M.; Pospisil, S. Response of the Pixel Detector Timepix to Heavy Ions. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2011**, *633* (Suppl. 1), S198–S202. [CrossRef]
12. George, S.P.; Kroupa, M.; Wheeler, S.; Kodaira, S.; Kitamura, H.; Tlustos, L.; Campbell-Ricketts, T.; Stoffle, N.N.; Semones, E.; Pinsky, L. Very high energy calibration of silicon Timepix detectors. *J. Instrum.* **2018**, *13*, P11014. [CrossRef]
13. Hoang, S.; Vilalta, R.; Pinsky,L.; Kroupa, M.; Stoffle, N.; Idarraga, J. Data Analysis of Tracks of Heavy Ion Particles in Timepix Detector. *J. Phys*. Conf. Ser. **2014**, *523*, 012026. [CrossRef]
14. Sommer, M.; Granja, C.; Kodaira, S.; Ploc, O. High-Energy Per-Pixel Calibration of Timepix Pixel Detector with Laboratory Alpha Source. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2022**, *1022*, 165957. [CrossRef]
15. Jakubek, J. Precise Energy Calibration of Pixel Detector Working in Time-Over-Threshold Mode. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2011**, *633* (Suppl. 1), S262–S266. [CrossRef]
16. Turecek, D.; Jakubek, J.; Kroupa, M.; Soukup, P. Energy Calibration of Pixel Detector Working in Time-Over-Threshold Mode Using Test Pulses. In Proceedings of the 2011 IEEE Nuclear Science Symposium Conference Record, Valencia, Spain, 23–29 October 2011; pp. 1722–1725. [CrossRef]
17. Ponchut, C.; Ruat, M. Energy Calibration of a CdTe X-ray Pixel Sensor Hybridized to a Timepix Chip. *J. Instrum.* **2013**, *8*, C01005. [CrossRef]
18. Amoyal, G.; Ménesguen, Y.; Schoepff, V.; Carrel, F.; Michel, M.; Angélique, J.C.; de Lanaute, N.B. Evaluation of Timepix3 Si and CdTe Hybrid-Pixel Detectors' Spectrometric Performances on X- and Gamma-Rays. *IEEE Trans. Nucl. Sci.* **2021**, *68*, 229–235. [CrossRef]
19. Jakubek, J. Semiconductor Pixel Detectors and Their Applications in Life Sciences. *J. Instrum.* **2009**, *4*, P03013. [CrossRef]
20. Delogu, P.; Biesuz, N.V.; Bolzonella, R.; Brombal, L.; Cavallini, V.; Brun, F.; Cardarelli, P.; Feruglio, A.; Fiorini, M.; Longo, R.; et al. Validation of Timepix4 Energy Calibration Procedures with Synchrotron X-Ray Beams. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2024**, *1068*, 169716. [CrossRef]
21. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004; ISBN 978-0-521-83378-3.
22. Marquardt, D.W. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *J. Soc. Ind. Appl. Math.* **1963**, *11*, 431–441. Available online: https://www.jstor.org/stable/2098941 (accessed on 17 December 2024). [CrossRef]
23. Armijo, L. Minimization of Functions Having Lipschitz Continuous First Partial Derivatives. *Pac. J. Math.* **1966**, *16*, 1–3. [CrossRef]
24. O'Leary, D.; Rust, B. Variable Projection for Nonlinear Least Squares Problems. *Comput. Optim. Appl.* **2013**, *54*, 579–593. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.