DetectorCalibGUI

Detector Energy Calibration Tool

Comprehensive User Manual

Author: Siddharth Parashari

Version 1.0 — 2025

1. Overview

DetectorCalibGUI is a cross-platform Python application for performing energy calibration of multi-channel particle and radiation detector systems. It supports PET scanners, Compton cameras, and general-purpose gamma-ray detector arrays. The tool reads raw ROOT data files, detects or accepts user-defined calibration peaks, fits a calibration model, and exports the resulting coefficients for downstream analysis.

1.1 Key Features

- **Dual ROOT backend:** PyROOT (native) or uproot (pure Python) — detected automatically.
- **Flexible data loading:** TTree filter mode, array-branch mode, custom Draw() expressions, or pre-filled TH1 histograms.
- **Intelligent peak detection:** ROOT TSpectrum (PyROOT) or SciPy fallback, with user-adjustable sigma and threshold.
- **Peak propagation:** Detect peaks on one reference channel and automatically search all other channels within a configurable ADC window.
- **Two calibration models:** Linear (E = P0 + P1·Q) and nonlinear scintillator model (E = P0·(P1^Q)^P2 + P3·Q − P0), plus user-defined custom expressions.
- **Calibrated spectrum viewer:** Apply coefficients (from session or file) and display energy-calibrated spectra.
- **Energy resolution analysis:** Interactive Gaussian fitting with FWHM extraction and per-channel resolution trend plots.
- **Batch processing:** Fit all channels simultaneously in a background thread with a live progress bar.
- **Rich output:** Human-readable calibration log, compact coefficients table, and resolution results file.

1.2 Application Tabs at a Glance

Tab	Name	Purpose
1	🔬 Single Channel	Spectrum view, peak detection & assignment, single-channel calibration fit
2	📊 Overview Grid	All-channel thumbnail grid colour-coded by fit status (green / red)
3	📈 Coefficient Trends	Plot calibration parameters vs channel ID with error bars
4	⚡ Calibrated Spectrum	View energy-calibrated spectra; send channel to resolution analysis

5	📐 Resolution	Interactive Gaussian peak fitting, FWHM extraction, trend plots

2. Installation & Requirements

2.1 Python Dependencies

Install the required Python packages before first launch:

```
pip install PyQt5 matplotlib numpy scipy
```

2.2 ROOT Backend (choose one)

Option A — PyROOT (recommended)

If ROOT 6 is installed on your system, source it before launching the application:

```
source /path/to/root/bin/thisroot.sh
python3 main.py
```

On macOS with Homebrew:

```
source $(brew --prefix root)/bin/thisroot.sh
```

The application auto-detects ROOT even if thisroot.sh was not sourced, by searching ROOTSYS, root-config --prefix, and common installation directories such as /opt/homebrew, /usr/local, and ~/root.

Option B — uproot (no ROOT installation required)

```
pip install uproot awkward
python3 main.py
```

> **NOTE** The active backend is displayed in the status bar after loading a file, e.g. [PYROOT backend] or [UPROOT backend]. TSpectrum peak detection requires PyROOT; scipy find_peaks is used automatically when uproot is active.

2.3 Launching

```
cd DetectorCalibGUI
python3 main.py
```

3. Loading Data

3.1 Opening a ROOT File

Click 📂 **Open ROOT File** in the toolbar. A file dialog opens; select any `.root` file. The application probes the file and counts available TTrees and TH1 histograms, then opens the **Configure Data Source** dialog.

3.2 Data Mode

Choose between two data modes:

- **TTree mode:** Raw event-level data stored in a TTree. The application reads branches and builds histograms per channel.
- **TH1 Histogram mode:** Pre-filled TH1F/TH1D histograms, one per channel. Channel ID is parsed from trailing digits in the histogram name (e.g. spectrum_ch0042 → channel 42).

3.3 TTree Draw Modes

When TTree mode is selected, choose one of three draw styles that determine how the application builds per-channel histograms:

Mode	ROOT equivalent	When to use
Filter	Draw("energy","channelID ==N")	Standard layout: separate branch for ADC values, separate branch for channel ID. Channel IDs are auto-discovered.
Array	Draw("energy[N]")	Array branch where each event stores a value for every channel (e.g. energy[64]). Requires channel range to be set.
Custom	Draw("expr, cut") with %d	Full control. Use %d as a placeholder for the channel number in both the draw expression and the cut string. Example: sqrt(energy[%d]), channelID==%d

NOTE The Draw preview label in the dialog shows the exact ROOT call that will be made, updated live as you change branch names or mode.

3.4 Branch Selection

- **Channel branch:** Branch containing the channel identifier (integer). Used in Filter mode as the selection variable.
- **Energy / ADC branch:** Branch containing the raw ADC value or deposited energy. This becomes the x-axis of the spectrum.

The branch dropdowns are populated automatically from the TTree. Select the correct branches using the dropdowns.

3.5 Channel Range & Entry Limits

Control	
First / Last / Step	Explicit channel range. Last = “Auto” (value −1) lets the application auto-discover channels from data (Filter mode only).

Custom list	Comma-separated list of channel IDs, e.g. 0,1,5,10. Overrides First/Last/Step when non-empty.
Max entries per channel	Limit events read per channel. 0 = All entries. Useful for fast preview of large files (e.g. set 100000 for a quick look).
Histogram bins	Number of bins for the resulting histogram. Default 1024. Reduce for fewer statistics.

4. Peak Detection & Assignment

4.1 Peak Detection Controls

The **Peak Detection** group in the left panel controls automatic peak finding. These settings apply when you click 🔍 **Detect Peaks (this channel)**.

Control	
Threshold (slider)	Minimum peak height as a fraction of the spectrum maximum (0.01 to 0.50). Lower values find smaller peaks but may produce more false positives.
Sigma (bins)	Gaussian smoothing width in bins applied before peak search. Increase for noisy spectra; decrease for sharp peaks.
Max peaks	Maximum number of peaks to return. Peaks are ranked by prominence and the top N are kept.

With the PyROOT backend, detection uses ROOT `TSpectrum::Search()` with the call:

```
sp.Search(h, sigma, "nobackground nodraw", threshold)
```

With the uproot backend, SciPy `find_peaks()` with Gaussian smoothing and centroid refinement is used automatically.

4.2 Detected Peak Table

After clicking **Detect Peaks**, the **Assign Energies to Detected Peaks** table is populated with the ADC positions of all found peaks (shown as orange dashed lines on the spectrum). To assign a physical energy to a peak:

1. **Double-click** any row in the detected peaks table, or select a row and click
2. The Assign Energy dialog opens with the ADC position pre-filled.
3. Type any energy value in keV in the “Known energy” spinbox, or click a row in the quick-select table to fill from common gamma-ray sources.
4. Add an optional label (e.g. Cs-137 661.7 keV) and click OK.

The assigned peak appears in the **Calibration Points** table and as a green dashed line on the spectrum.

4.3 Manual Peak Entry

- 🖱️ **Click mode:** Toggle click mode with the Click button. The cursor changes to a crosshair. Click anywhere on the spectrum to open the Assign Energy dialog at that ADC position.
- **+ Manual:** Opens the Assign Energy dialog with ADC position set to zero. Type any ADC position and energy.

4.4 Quick-Select Energy Table

The Assign Energy dialog contains a scrollable table of 12 common gamma-ray sources for convenience. Click any row to fill the energy field. You may always type any value manually.

Energy (keV)	Source	Common use
511.0	Na-22 / annihilation	PET, pair production
1274.5	Na-22	PET calibration
661.7	Cs-137	General-purpose standard
1332.5	Co-60	Gamma calibration
1173.2	Co-60	Gamma calibration
122.1	Co-57	Low-energy calibration
344.3	Eu-152	Multi-energy standard
1460.8	K-40	Natural background
59.5	Am-241	Low-energy / X-ray range
88.0	Cd-109	X-ray / low energy

4.5 Global vs Per-Channel Peaks

Peaks can be defined globally (applied to all channels as a default) or overridden per channel:

- **Global peaks:** When the Override checkbox is not ticked, any peak you add goes into the global list. These are used as the calibration points for any channel that has no channel-specific override.
- **Per-channel override:** Tick the Override checkbox to create a channel-specific peak list. This channel will use its own list instead of the global one.
- **↑ Use Global:** Removes the per-channel override for the current channel and reverts to the global list.
- **− Remove:** Deletes the selected calibration point from the current channel’s list (or from the global list if no override is active).

5. Peak Propagation

The **Propagate Peaks to All Channels** feature lets you define or detect peaks on one representative channel and automatically find the corresponding peaks in all remaining channels within a configurable search window. This is the most efficient workflow for multi-channel systems where all channels exhibit similar spectra.

5.1 Workflow

5. Select a representative channel using the channel selector.
6. Detect peaks (click 🔍 Detect Peaks) or manually assign peaks with known energies.
7. Set the Search window (± ADC) spinbox. A value of 50 means the application will look in [ref_ADC − 50, ref_ADC + 50] for each peak.
8. Select the peak source: current channel (uses all detected + assigned peaks) or Global peak list only.
9. Tick Override checkbox to store refined positions as per-channel overrides (recommended).
10. Click 🔄 Propagate Peaks → All Channels.

A status line reports how many channels were successfully matched, e.g. *Propagated 2 peak(s) to 15/16 channels.*

5.2 Search Algorithm

For each reference peak, the algorithm:

11. Slices the target channel's spectrum to the window [ref_ADC − window, ref_ADC + window].
12. Runs TSpectrum or SciPy inside that narrow window to find peak candidates.
13. Selects the candidate closest to the reference ADC position.
14. Falls back to the centroid of the local maximum if no peak is detected by the finder.

The resulting peaks carry the same **known_energy** as the reference, but with ADC positions refined to each channel. This correctly accounts for gain variation across the detector.

> TIP If a channel's spectrum has very different gain from the reference, widen the search window. If false peaks are being matched, reduce the window.

6. Calibration Fitting

6.1 Calibration Models

Linear (default)

$$E = P0 + P1 \cdot Q$$

Simple linear model. P0 is the offset (keV), P1 is the gain (keV/ADC). Requires ≥ 2 calibration points. With exactly 2 points, the fit is an exact interpolation (NDF = 0) which is valid and will never be flagged as a bad channel.

Nonlinear Scintillator Model

$$E = P0 \cdot (P1\,\hat{}\,Q)\,\hat{}\,P2 + P3 \cdot Q - P0$$

Physically motivated model for scintillator detectors where the light yield has a nonlinear response at low energies. Parameters:

Param	Typical value	Meaning
P0	~10 keV	Nonlinearity amplitude. Set to 0 for purely linear response.
P1	~0.999	Exponential base. Must be in (0, 2).
P2	~1.0	Exponent shaping factor.
P3	keV/ADC	Linear gain (dominant term). Same as P1 in the linear model.

Requires ≥ 4 calibration points. Initial guesses are computed automatically from the slope of the first and last points.

Custom Expression

Select **Custom** in the model dropdown. A text field appears in the toolbar. Type any mathematical expression using `x` as the ADC variable and single letters (not x) as parameters:

```
a*x**2 + b*x + c
a*np.exp(-b*x) + c*x
a*np.log(x) + b
```

Available functions: `np.exp`, `np.log`, `np.sqrt`, `np.abs`, `np.sin`, `np.cos`. Parameters are discovered automatically from the expression. Requires ≥ N calibration points where N is the number of parameters.

6.2 Fitting a Single Channel

After assigning calibration peaks, click ▸ **Fit This Channel** in the left panel. The calibration curve is drawn immediately in the Single Channel tab, annotated with each peak's energy in keV and showing the χ^2/NDF value in the title.

6.3 Batch Fitting All Channels

Click ⚡ **Fit All Channels** in the toolbar. Fitting runs in a background thread; a progress bar in the status bar updates in real time. On completion:

- **Overview Grid tab:** All channel thumbnails are redrawn with a green border (good fit) or red border (bad channel).
- **Coefficient Trends tab:** Parameter plots are updated.
- **Status bar:** Reports total channels fitted and number of bad channels.

6.4 Bad Channel Flagging

A channel is flagged as **bad** only if the fit genuinely fails, i.e. the covariance matrix contains NaN (numerically unstable fit). A channel is **never** flagged bad merely because χ²/NDF is large or uncertainties are wide — those are informational only. This design ensures that a 2-point linear calibration always succeeds.

> **NOTE :** If a channel cannot be fit even after several attempts, check that the calibration peaks are correctly assigned and that the ADC range is within the spectrum.

7. GUI Reference

7.1 Toolbar

Control	
📂 **Open ROOT File**	Opens a ROOT file and shows the Configure Data Source dialog.
Model dropdown	Choose calibration model: Linear (default), Nonlinear, or Custom.
Custom expression field	Visible when Custom is selected. Type the mathematical expression here.
⚡ **Fit All Channels**	Batch-fit all loaded channels in a background thread.
💾 **Export**	Saves calibration results (log + coefficients files) to a chosen directory.

7.2 Left Panel

Group	
Channel	Previous / Next arrows, channel combo box, per-channel info label (entries, source, fit status).
Peak Detection	Threshold slider (0.01–0.50), Sigma spinbox, Max peaks spinbox, Detect Peaks button, detected count label.
Propagate Peaks	Search window (± ADC), source selector (current channel / global), override checkbox, Propagate button, status label.
Assign Energies	Detected peaks table (double-click or select + Assign to open energy dialog), manual add, click mode toggle.
Calibration Points	Assigned peaks table, Remove button, Use Global button, Override checkbox.

Fit This Channel	Fit the current channel immediately using assigned peaks.

7.3 Single Channel Tab

The top canvas shows the raw ADC spectrum in log-y scale. Orange dashed lines mark detected-but-unassigned peaks; green dashed lines mark assigned peaks with their energy label.

The bottom canvas shows the calibration curve (fit line + annotated calibration points) with the model label and χ^2/NDF in the title.

A Matplotlib navigation toolbar above the spectrum allows zoom, pan, and save.

7.4 Overview Grid Tab

Displays thumbnails of all loaded channel spectra in an 8-column scrollable grid. Each thumbnail has a coloured border: **green** = successful fit, **red** = bad channel, **grey** = not yet fitted. Click any thumbnail to navigate to that channel in the Single Channel tab.

7.5 Coefficient Trends Tab

Plots the value of a chosen calibration parameter (P0, P1, P2, P3) against channel ID with error bars. Bad channels are marked with a red vertical line. Use the parameter dropdown to switch between parameters. A horizontal mean line is shown for reference.

8. Calibrated Spectrum Tab

The ⚡ **Calibrated Spectrum** tab converts raw ADC spectra to energy spectra (keV axis) using the fitted calibration coefficients.

8.1 Loading Calibration

- 📋 **From Session:** Uses calibration results from the current fitting session. Click this after running Fit All Channels.
- 📂 **Load coeffs.txt:** Loads a calibration_coeffs.txt file produced by a previous export. This allows applying calibration from a prior session or from a different dataset to any ROOT file.

The calibration source and number of calibrated channels are shown next to the buttons.

8.2 Energy Range & Binning

Min (keV)	Low edge of the output energy axis. 0 = auto (uses minimum calibrated ADC).
Max (keV)	High edge of the output energy axis. 0 = auto.

Bins	Number of bins in the output calibrated histogram. Default 1024.

8.3 Applying Calibration

- **Single channel:** Select a channel in the Channel combo box. The calibrated spectrum is computed and displayed immediately.
- **▸ Apply to All:** Computes calibrated spectra for every channel with a valid calibration. Required before batch resolution analysis.
- 📐 **Analyse Resolution →:** Sends the current channel's calibrated spectrum to the Resolution tab and switches to it automatically.

9. Resolution Analysis Tab

The 📐 **Resolution** tab performs interactive Gaussian peak fitting on calibrated spectra to extract energy resolution (FWHM and R%).

9.1 Interactive Fit Workflow

15. Send a calibrated spectrum from the Calibrated Spectrum tab using 📐 Analyse Resolution →.
16. The spectrum is displayed in the left canvas with a draggable SpanSelector (yellow band).
17. Drag horizontally on the spectrum to select the energy range around a peak.
18. Set the Peak label (keV) spinbox to the nominal energy of this peak.
19. Choose the background model: Gaussian + linear BG (default) or Gaussian only.
20. Click ⚡ Fit Peak. The Gaussian is fitted within the selected range and overlaid in red. FWHM and R% are annotated directly on the spectrum.
21. Repeat for additional peaks on this channel.
22. To fit the same peak across all channels, click ⚡ Fit All Channels (same range).

9.2 Fit Model

The fit function is a Gaussian on a linear background:

$$f(E) = A \cdot \exp[-0.5 \cdot ((E - \mu) / \sigma)^2] + B + C \cdot E$$

From the fitted μ and σ, the following quantities are derived:

Quantity	Formula	Description
FWHM	2.3548 × σ	Full width at half maximum (keV)
R%	FWHM / μ × 100	Relative energy resolution (%)

9.3 Results Table

Each successful fit adds a row to the Results Table showing: Channel, Peak (keV), Centroid (keV), FWHM ± error (keV), R% ± error, χ²/NDF, and status. Failed fits show a red row with the failure reason.

9.4 Resolution Trend Plot

The Resolution Trend sub-tab plots FWHM (keV) or R% vs channel ID with error bars and a mean value line. Use the Peak dropdown to switch between different fitted peaks, and the Show dropdown to toggle between FWHM and R%.

9.5 Export

Click 💾 **Export** to save all resolution results to `resolution_results.txt`. The file contains a full table with FWHM, uncertainty, R%, χ²/NDF, and status for every (channel, peak) combination that was fitted.

10. Exporting Calibration Results

After fitting, click 💾 **Export** in the toolbar and select an output directory. Two files are written:

10.1 calibration_log.txt

Full human-readable report with one section per channel: model label, χ²/NDF, all parameters with uncertainties, and a table of ADC positions, assigned energies, and residuals.

10.2 calibration_coeffs.txt

Compact machine-readable table, one row per channel:

```
Channel          P0          P1    Chi2/NDF  Status
      0  -1.23456e+00  +5.00000e-01   0.0000    OK
      1  -1.18900e+00  +4.99800e-01   0.0000    OK
     42             nan         nan      nan    BAD: Covariance ...
```

This file can be loaded directly by the Calibrated Spectrum tab to apply calibration in a future session without re-fitting.

11. Typical Workflows

11.1 Standard Multi-Channel Calibration

23. Open ROOT file → configure TTree / branch names → OK.
24. Navigate to a representative channel.
25. Click 🔍 Detect Peaks. Adjust Threshold / Sigma until the correct peaks are found.
26. Double-click each detected peak row → assign known energy → OK.

27. Verify green markers align with the correct spectral features.
28. Set Search window (± ADC). Click 🔄 Propagate Peaks → All Channels.
29. Select Linear model (or Nonlinear for scintillators). Click ⚡ Fit All Channels.
30. Review Overview Grid. Investigate any red-bordered channels individually.
31. Click 💾 Export to save calibration_log.txt and calibration_coeffs.txt.

11.2 Applying Saved Calibration to a New File

32. Open the new ROOT file and configure data source as usual.
33. Switch to the ⚡ Calibrated Spectrum tab.
34. Click 📂 Load coeffs.txt and select the previously exported calibration_coeffs.txt.
35. Click ▸ Apply to All.
36. Inspect calibrated spectra for each channel.

11.3 Resolution Analysis Workflow

37. Complete calibration fitting (sections 4–6) or load calibration from file.
38. In the ⚡ Calibrated Spectrum tab, click ▸ Apply to All.
39. Select any channel and click 📐 Analyse Resolution →.
40. In the Resolution tab, drag a span over the 511 keV (or desired) peak.
41. Set Peak label = 511.0, click ⚡ Fit Peak. Verify the Gaussian overlay.
42. Click ⚡ Fit All Channels (same range) to propagate to all channels.
43. Check the Resolution Trend tab for FWHM vs channel and R% vs channel plots.
44. Click 💾 Export to save resolution_results.txt.

12. Troubleshooting

Problem	Solution
No ROOT backend found	Source thisroot.sh (PyROOT) or install uproot: pip install uproot awkward
No channels found after loading	For Array/Custom mode, set the channel range (First/Last) manually. For Filter mode, verify the channel branch name is correct.
All channels flagged bad	This should not happen with the current version. If it does, check that calibration points are assigned correctly and that the model has enough points (linear ≥2, nonlinear ≥4).
TSpectrum finds wrong peaks	Lower threshold or increase sigma. Alternatively, use Click mode to place peaks manually.

Propagation misses peaks	Widen the search window (e.g. 100–200 ADC). If channels have very different gains, propagation may not be reliable; assign peaks manually per channel.
Calibration curve looks wrong	Verify peak assignments: green markers should be at the correct spectral features. Remove and reassign if needed.
Gaussian fit fails in resolution tab	Widen the span selection to include more of the peak. Switch to Gaussian only mode if the background model causes instability.
Export produces no files	Run Fit All Channels or Fit This Channel before exporting. At least one successful fit is required.

13. Module Reference

13.1 File Structure

File	
main.py	Entry point. Initialises Qt, runs backend check, launches MainWindow.
src/root_loader.py	ROOTFileLoader class. Handles all ROOT I/O for both backends and all draw modes.
src/peak_manager.py	PeakManager class. Stores global and per-channel peaks. TSpectrum / SciPy detection. find_peaks_in_windows() for propagation.
src/calib_fitter.py	CalibrationFitter class. scipy.optimize.curve_fit for linear, nonlinear, and custom models. FitResult dataclass.
src/output_writer.py	OutputWriter class. Writes calibration_log.txt and calibration_coeffs.txt.
src/calib_spectrum.py	CalibSpectrumEngine class. Converts ADC spectra to energy spectra using stored coefficients.
src/resolution.py	ResolutionCalculator class. Gaussian + linear BG fitting. FWHM, R% extraction and export.
src/main_window.py	MainWindow and all PyQt5 widgets, canvases, dialogs, and tab builders.
src/calib_spectrum_tab.py	CalibratedSpectrumTab widget (Tab 4).

src/resolution_tab.py	ResolutionTab widget with SpanSelector and trend plot (Tab 5).

Appendix A — Calibration Formula Reference

Model	Formula	Notes
Linear	E = P0 + P1·Q	Default. 2 params. Works with 2+ peaks.
Nonlinear	E = P0·(P1^Q)^P2 + P3·Q − P0	4 params. Requires 4+ peaks. P1 ∈ (0,2).
Custom	User-defined in toolbar	Parameters: any single letter except x.

Appendix B — Keyboard & Mouse Quick Reference

Action	
Next / previous channel	Arrow buttons (◀ ▶) in left panel, or change combo box
Place a peak manually	Toggle 🖱️ Click button, then left-click on spectrum
Assign energy to detected peak	Double-click any row in the Assign Energies table
Select Gaussian fit range	Drag horizontally on the calibrated spectrum in Resolution tab
Zoom spectrum	Matplotlib toolbar: magnifier icon, or scroll wheel
Pan spectrum	Matplotlib toolbar: arrow icon, then drag
Reset view	Matplotlib toolbar: home icon
Navigate to channel from grid	Click any thumbnail in Overview Grid tab

End of User Manual — *DetectorCalibGUI v1.0* — *Siddharth Parashari*