The Journal of Open Source Software

# APyT: A modular open-source framework for atom probe tomography data evaluation

**Sebastian M. Eich** [1]

**1** Department for Materials Physics, Institute for Materials Science, University of Stuttgart, Germany

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:** Lucy Whalley

**Reviewers:**

- @mkuehbach
- @zhangjy-ge

**Submitted:** 14 October 2025
**Published:** unpublished



## Summary

Atom probe tomography (APT) is a microscopy and mass-spectrometry technique that provides three-dimensional, nanoscale insight into materials, including chemical composition and spatial distribution of atoms. However, processing APT data involves a complex sequence of steps — such as voltage and detector hit-position corrections, mass-to-charge calibration, peak identification, and three-dimensional reconstruction — which have traditionally relied on closed-source commercial software with limited transparency, reproducibility, and accessibility.

The **APyT** package (Eich, 2025) offers an open-source, modular, and fully scriptable framework for processing, reconstructing, and analyzing APT data. Implemented entirely in Python, APyT supports both interactive use via a Matplotlib-based graphical user interface and automated, reproducible workflows through its well-documented application programming interface (API). Its design emphasizes transparency, reproducibility, and ease of integration into custom research pipelines. A small example dataset is included, allowing users to explore the complete workflow immediately after installation.

## Statement of need

In the APT community, the most widely used analysis environments — such as **IVAS** or **AP Suite** by *CAMECA* (CAMECA SAS, 2025) — are closed-source and require commercial licenses. While these software suites provide comprehensive graphical interfaces, they offer only limited scripting capabilities and lack transparency regarding internal data processing and reconstruction algorithms. As a result, reproducibility, automation, and integration with modern data-science frameworks remain challenging.

The **Atom Probe Toolbox** (Heller et al., 2024) is an open-source alternative; however, it depends on the commercial MATLAB environment, which limits accessibility and cross-platform deployment. Other tools, such as **APAV** (Smith & Young, 2023) and **APTTools** (APTTools developers, 2025), address specific analysis tasks, but do not provide an end-to-end workflow from raw measurement data to full three-dimensional reconstruction.

**APyT** (Eich, 2025) addresses this gap by providing an open-source, Python-based ecosystem for APT data processing and analysis. It is designed for both educational and research-oriented use, enabling flexible scripting, reproducible analyses, and integration with external Python-based workflows. The framework follows a strict separation between user interface and computational logic, supporting continuous development and scientific extensibility.

## Audience and scope

The intended audience of APyT includes researchers in materials science, surface physics, and microscopy who work with APT. In addition, the modular Python architecture makes the

framework attractive for data scientists seeking to integrate APT data into broader multi-modal analysis pipelines, such as correlative microscopy or atomistic modeling workflows.

## Features and implementation

APyT provides a complete workflow for APT analysis:

- **Mass spectrum alignment**: Corrects voltage and hit-position effects to obtain sharp peaks including accurate alignment.
- **Spectrum fitting**: Chemical identification based on analytic peak-shape models; supports both atoms and molecules.
- **Three-dimensional reconstruction**: Classic voltage-based radius model or taper geometry model with predefined taper angle and radius.
- **Data import**: Support for vendor and custom raw data formats.
- **Database integration**: Central SQL or YAML-based local databases for metadata management.
- **Command-line interface**: Lightweight wrappers for alignment, fitting, and reconstruction modules enable script-free use, lowering the entry barrier for new users.
- **Example dataset**: Small raw measurement dataset included for hands-on demonstration.
- **Extensive documentation**: API, guides, example workflows, configuration details, and release notes.

APyT employs a layered architecture that separates data I/O, numerical processing, and visualization. Each task — alignment, fitting, and reconstruction — is encapsulated in a dedicated module with a clear API, ensuring reproducibility, testability, and seamless integration with Python workflows such as Jupyter notebooks or automated pipelines. Flexible database support and command-line wrappers make it suitable for both centralized data management and individual use.

## Example usage

The APyT package is shipped with an exemplary measurement dataset, allowing users to perform test reconstructions out of the box. The documentation provides a detailed guide for performing spectrum calibration and reconstruction via the command-line interface (CLI) with simple GUIs from Matplotlib. The example dataset and workflow description are available in the online documentation (Eich, 2025).

## Comparison with existing software

APyT stands out as an open, scriptable, and extensible framework designed for reproducibility and integration into larger data processing pipelines. It provides a complete set of modules, from mass spectrum calibration to three-dimensional reconstruction, and emphasizes compatibility with standard Python scientific libraries such as NumPy, SciPy, and Matplotlib.

The following table provides an overview of available software packages:

| Software | Open source | Interface | Script-able | Full workflow | DB management |
|---|---|---|---|---|---|
| **APyT** | Yes | CLI + Python API | Yes | Yes | Yes |
| IVAS/AP Suite | No | GUI | No | Yes | No |
| Atom Probe Toolbox | Yes (Matlab req.) | Matlab | Yes | Yes | No |
| APAV | Yes | Python API | Yes | No | No |

| Software | Open source | Interface | Script-able | Full workflow | DB man-agement |
|---|---|---|---|---|---|
| APTTools | Yes | GUI + limited API | Partial | No | No |

## Data architecture and storage considerations

APT measurements generate large binary datasets — typically millions of detection events, each defined by position, time-of-flight, and auxiliary detection information. Efficient storage and retrieval of these data, along with their associated metadata (e.g., measurement conditions, reconstruction parameters, and analysis results), are core design considerations in APyT.

APyT employs a **hybrid storage model** that combines:

- **Binary raw data files** for central storage on a file server.
- **An SQL database** for metadata, measurement conditions, reconstruction parameters, and analysis results.

This architecture offers several advantages:

- Fast metadata queries using SQL, enabling efficient access even for large repositories.
- Lightweight updates when analysis parameters are added or refined.
- Clear separation between large, immutable raw data and small, frequently updated metadata.

In addition to the SQL-based approach, APyT supports **local metadata management** through a simple YAML file that mirrors the structure and content of the central database. This dual setup allows users to run APyT tools locally without requiring access to a central database infrastructure, facilitating both standalone and collaborative workflows.

### Comparison with HDF5-based storage

HDF5 is a widely used format for scientific data management, valued for its self-describing structure and hierarchical organization of datasets and metadata. However, when compared to SQL in the context of APT data, several key differences emerge.

The **SQL + binary data model** used in APyT offers faster and more flexible metadata queries, efficient concurrent access through standard database servers, and lightweight updates without the need to rewrite large files. It also provides native network access and transactional consistency, making it well suited for distributed repositories and datasets that evolve over time. In contrast, **HDF5** performs best in archival or self-contained scenarios, but is slower for large-scale querying, less efficient for incremental updates, and limited in concurrent or networked access.

In summary, SQL excels at fast, flexible metadata management, while HDF5 is preferable for long-term archival storage or data portability. For central repositories with network access and frequently updated metadata — as in typical APT workflows — the hybrid SQL + binary model adopted by APyT remains the more practical and scalable solution. Nevertheless, APyT's architecture remains flexible and can accommodate future HDF5 integration for long-term archiving or standardized data exchange formats, such as those promoted by FAIR data initiatives (Wilkinson et al., 2016).

## Future development

Planned developments for APyT include a unified graphical interface to improve accessibility for non-specialist users, native support for HDF5 as a standardized format for long-term data

archiving, and an expanded set of analysis routines. These enhancements aim to improve usability, strengthen data interoperability, and broaden the range of supported APT workflows.

## Software availability

The source code, issue tracker, and documentation for APyT are available at https://github.com/sebi-85/apyt and https://apyt.mp.imw.uni-stuttgart.de, respectively. The APyT package can be installed directly from PyPI.

## Acknowledgments

Development of APyT was carried out at the Institute for Materials Science, University of Stuttgart. The author gratefully acknowledges colleagues for their valuable feedback on usability and testing throughout the development of the package. Special thanks are extended to Dr. Jianshu Zheng for insightful discussions on database design and software architecture, as well as for his contributions to testing and validation.

## References

APTTools developers. (2025). *APTTools*. https://sourceforge.net/projects/apttools/

CAMECA SAS. (2025). *CAMECA IVAS and AP Suite Software*. https://www.cameca.com/

Eich, S. M. (2025). *APyT: A modular open-source framework for atom probe tomography data evaluation*. https://apyt.mp.imw.uni-stuttgart.de/

Heller, M., Ott, B., Dalbauer, V., & Felfer, P. (2024). A MATLAB toolbox for findable, accessible, interoperable, and reusable atom probe data science. *Microscopy and Microanalysis*, *30*(6), 1138–1151. https://doi.org/10.1093/mam/ozae031

Smith, J. D., & Young, M. L. (2023). APAV: An open-source Python package for mass spectrum analysis in atom probe tomography. *Journal of Open Source Software*, *8*(83), 4862. https://doi.org/10.21105/joss.04862

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1). https://doi.org/10.1038/sdata.2016.18