Development of Python-based Toolkit to Improve Analysis of Atom Probe Tomography Data

Vishal Kanigicherla[1], Dr. Keith Knipling[2], and Dr. Brian Kennedy[1]

[1]Thomas Jefferson High School for Science and Technology
[2]U.S. Naval Research Laboratory Materials Science and Technology Division

September 2020 - May 2021

Contents

Abstract

The purpose of the project is to develop a Python-based toolkit to optimize and streamline analysis of data produced by atom probe tomography (APT), addressing many of the issues with the analysis process such as the difficulties of mass spectrum peak decomposition, mass spectrum ranging, and error calculation in reported compositions. This will be achieved using established Python libraries such as Pandas, processing large amounts of binary data generated by APT into workable CSV files, and addressing each issue in APT analysis with discrete scripts, finally compiling them into a toolkit available on Github.

1 Purpose

This project seeks to holistically address APT data analysis using Python, liberating APT data analysis from proprietary software while also improving existing data analysis techniques by implementing various algorithms employing optimization and machine learning.

2 Hypothesis

This project demonstrated the purpose by automating various menial and/or tedious tasks associated with APT data analysis, implementing methods to calculate important metrics integral to APT analysis, and tentatively will optimize industry standards for the specific algorithms involved in cluster identification and more. The automation was measured by consistency with past results of NRL APT data analysis, while the optimization will be measured against industry standards through efficiency and error comparison. The automation's consistency was a perfect equivalent to past calculations, and the spatial error in the optimized algorithm will be statistically significantly less than that of the maximum separation algorithm.

3 Introduction

Atom probe tomography (APT) is a precise atom-by-atom dissection of a material volume, producing 250×250×1000 nm^3 three-dimensional reconstructions, with sub-nanometer resolution, and chemical sensitivities approaching 10 atomic ppm [1]. The ability to detect individual atomic species, directly in three dimensions, provides researchers with an unprecedented understanding of a material's nanostructure, which ultimately dictates that material's properties and performance [2].

Specimens for APT are in the form of sharply pointed needles, typically with an end radius less than 100 nm. These specimens are subjected to cryogenic temperature (approximately 30 K) and high voltage (approximately 10 kV), causing the atoms at the apex of the specimen tip to ionize and accelerate away from the positively charged tip and towards a detector. The detector records both the ion's time-of-flight (and hence its mass-to-charge ratio — its chemical identity) and its impact position (and hence its original location on the tip). APT specimens are generally prepared by milling in a dual-beam focused ion beam/scanning electron microscope (FIB/SEM) [2].

The ability to detect individual atomic species, directly in three dimensions, gives researchers an unprecedented understanding of a material's nanostructure, which ultimately dictates that material's properties and performance. APT also enables the intelligent design of future structural, electronic, optoelectronic, and functional materials for various applications.

4 Challenges of Atom Probe Tomography

Though APT is an incredibly powerful analytical technique, often considered a "holy grail" of such techniques, it has its downsides. FIB milling implants gallium ions into the specimen which can introduce damage and must be corrected for during analysis [3]. APT also has a relatively small field of view, limiting the size of the nanoscale features that can be analyzed [4]. Also, there are many artifacts associated with the technique — including local magnification and isotopic overlaps in the mass spectrum — which can complicate data analysis and interpretation. Developing new algorithms to overcome some of these challenges is one of the objectives of this proposed project.

Data analysis automation and optimization is another challenge with APT data, especially considering that modern APT datasets can contain hundreds of millions of atoms. One process which would benefit from automation is mass spectrum peak identification, which becomes cumbersome due to ambiguities in mass-to-charge ratio. A common example in steels occurs at 27 Da, which could be $^{27}\mathrm{Al}^{1+}$, $^{54}\mathrm{Fe}^{2+}$, or $^{54}\mathrm{Cr}^{2+}$. This ambiguity of isotopic overlap is normally resolved through user-interaction, but could easily be automated considering the natural abundances of the atomic species in question and their non-overlapping peaks. The primary data format of APT data is the POS file, which contains position (x, y, z) and mass-to-charge ratio for each atomic data point, and an accompanying range file, which identifies and assigns element information to peaks on a mass spectrum. New tools will be developed to work with these data files in Python. A common analytical technique in APT is the proximity histogram, or "proxigram," which plots the concentrations of elements with respect to a three-dimensional isoconcentration surface in the data. The proxigram is commonly used in processes of precipitation, where the precipitates can be delineated by an isoconcentration surface and then co-positions of the precipitates and the surrounding matrix can be computed. All of these tasks can be streamlined into one toolkit, again simplifying many menial processes central to APT analysis.

There are other issues that may be addressed with this toolkit. These include implementing algorithms for detecting solute clusters, which are based on the maximum separation algorithm. This process, however, is heavily dependent on user-identified parameters such as d$_{max}$ (maximum distance between atoms in a cluster), N$_{min}$ (minimum number of atoms in a cluster), and K, the order of the nearest-neighbour model sometimes used to determine d$_{max}$ [5]. Recent research has identified a possible avenue of optimization of this rather sensitive and subjective process utilizing Ripley's K-function and machine learning, but the problem remains open for further investigation [6]. Other problems with APT data analysis include the optimization of existing simulations for field evaporation behavior such as TAPSim; while fruit for further discussion, may lay outside the scope of the current project [7].

The application of APT is extremely important in materials science. Research using the technique extends to diverse areas of interest, such as high-entropy alloys that have application as substitutions for titanium in corrosive environments, the characterization of biological metal-interacting compounds such as porphyrin rings or proteins such as ferritin, and geosciences and investigation of the makeup of our planet [8] [9] [1]. The significance of this work is that implementation of such a toolkit will allow for an easily accessible and optimized data analysis procedure with a variety of scripts and algorithms necessary for APT data analysis, giving researchers the flexibility to turn their attention more towards the impact of their research itself, rather than the menial tasks associated with the tool that is APT. In stating such, the proposed toolkit will improve efficiency in the fields of research associated with APT, and will impact the diverse and extensive fields of research that APT finds application in.

5 Intended Outcome

The intended outcome of this work will be measured in two ways. First, for the sections of the toolkit intended to simply streamline certain data analysis processes, said sections will process past data collected by the NRL, and resultant statistics should be found consistent with past reports. For the exploratory sections of the toolkit with no prior implementation in NRL work, I will compare statistical distributions of data processed using new algorithms with distributions I generate through the implementation of standard algorithms using necessary two-sample T-tests and one-way ANOVAs as necessary.

6 Materials

Materials that were used for this project include Python libraries (pandas, numpy, matplotlib, tkinter, struct, sklearn) to interact with files and data types for processing, a laptop with Python and necessary environment installed (currently Lenovo Thinkpad) including IVAS LT, MATLAB, a text editor (VSCode), and an HDF file viewer. Finally, I made use of POS/EPOS files, range files, and proxigram files publicly available or generated from past and current work at the NRL, in addition to Excel workbooks with formulae kindly provided by Dr. Colin Stewart at the NRL MSTD.

7 Procedure

Python code was written in order to:

- Generate initial CSV files from proxigram XLSX.
- Take user-input for manual peak decomposition in proxigrams.

- Convert proxigram atom counts to at% values.
- Generate profiles of proxigram data after discarding irrelevant isotopes.
- Create core statistics file from user identification of matrix and precipitate of profile.
- Turn large POS files into workable CSVs.
- Calculate and display sample composition error statistics for proxigram and CSV.
- Take inputs for calculating average radius, vol fraction, and number density, do the calculation, and report the uncertainty.
- Generate a mass spectrum graph and analysis from POS CSV.
- Consolidate programs into GUI using Py2App and tkinter for better user experience.

8 CSV Generation and Manipulation

The *pandas* library is a powerful tool that can be used to manipulate large amounts of data in comma-separated values, or CSV files. One other industry standard file format is Hierarchical Data Format (HDF4, HDF5), though interaction with and export capability is also integrated into pandas.

In order to streamline the Excel data manipulation, proxigram files were exported as CSVs. POS and EPOS files, though, do not have this built in capability. Accordingly, a program was written to parse the binary formatting of reference POS files using struct, and write the 16 or 44 char string (.pos and .epos respectively) to a CSV, to be processed by later programs. A problem initially faced was the difficulty of creating an algorithm that properly parsed through the file and stopped when it reached the end. The solution developed was more simple: parse through the file once counting the number of characters, and divide by the number of characters which should exist in each line. For a POS file, this number would be 16. For an EPOS file, this number would be 44. In order to add functionality for working with EPOS files to the applet, the same program could be used - all that is required is to change all instances in the code of the number '16' to '44,' and change the ouput type from the POS format's four floating point numbers to the correct nine floating point numbers and two integers.

A bulk of the code was written for interaction with proxigrams, due to their easier manageability as a result of smaller file sizes. One concern, as reflected earlier, was the decomposition of overlapping peaks, previously being resolved at the NRL with a manual input of known isotope abundances and requiring much work to conduct. The script written to automate much of this process interacts with three files. The first is the original CSV proxigram, from which unknown peaks are separated into their constituent elements, and the file is rewritten into an intermediate. This intermediate file is processed and resolved but still a standard proxigram file, making it extremely relevant for other calculations. The third file is the final, with percentages instead of atom counts, and data deemed irrelevant to analysis by the user discarded; that file is used to generate the scatter plot to visualize concentration distribution, while the intermediate second file will be used after 'core' region identification for further statistics. Scripts automating many of these subtasks which also included a proxigram profile plotter, core composition statistics generation, isotopal composition error CSV generation, and more were also compiled into a master script in the repository as *ProxigramPeakDecomp.py* and also divided into constituent tasks and placed in the repository.

9 Gibbsian Interfacial Excess of Solute

The Gibbsian interfacial excess of solute (Γ_s) is a quantity that relates the concentration of solute within an infinitesimally small interface, a "Gibbs dividing surface," between two bulk phases of a material. APT data, which is comprised of spatially resolved atom counts, is ideally suited for calculating Γ_s with respect to any complex interface.

In many prior formulations, knowledge of the area of the interface is necessary to computing Γ_s. This often results in inherently error-producing somewhat arbitrary estimations of curves and planes. This is resolved by calculating area through other statistics that can be generated from APT: shell thickness, atomic density, and number of atom counts in each slice. Hellman & Seidman (2002) provides a formula using summations to calculate Γ_s from concentrations in shells from the interface of interest [10].

The purpose of a proximity histogram, or proxigram (ubiquitous across APT research) is to visualize atom counts in the vicinity of an isoconcentration surface, usually a nonplanar interface of interest. The proxigram, therefore, is ideal for calculating Γ_s from the methods established in this article. Summations ($\sum$) are easily executable in code through for-loops. Thus, I wrote a script to parse through a proxigram and sum concentrations of solutes of interest, and use statistics calculated in other package scripts (number density, particle volume, percent concentrations, etc.) to implement the formulae devised. The advantage of the formulation presented by Hellman & Seidman (2002) was that it eliminated the need for *a priori* knowledge of slice area, and through dimensional analysis substituted a formula for it [10].

$$A_i = \frac{N_i}{\rho \Delta l}$$

The difficulty arises when calculating an atomic number density ρ.

This was circumvented by prompting the user for the lattice parameter of the element of interest, and assuming a Bravais lattice geometry (possibly open for input), we calculated the ideal atomic number density and accordingly approximated the area of the unit of interest. Ultimately, I was able to develop a script to calculate the important thermodynamic quantity of Gibbs interfacial excess of solute. The script for this is available in the Github repository as *GibbsInterfacialExcess.py.*

10 Cluster Analysis and Detection

Isotope cluster analysis is also a major point of interest for research using APT. IVAS, a commonly used commercial software for APT data analysis, has the ability to output a dataset of an atom probe sample with cluster counts of certain isotopes in regions determined by the user. The region predetermined by the user, often an isoconcentration surface, may clip the edge of certain clusters. Consequently, a workaround used by the NRL MSTD is to export two cluster searches: one that is ”bounded,” meaning that it is for clusters completely enclosed within the dataset, and one that is ”clipped,” or partially clipped by the boundary of the needle dataset.

Quantities that are of research interest include the average equivalent volume and radius of these clusters, the standard deviation of the equivalent radii, the total number of precipitates in the dataset, and the number density of the particles in the dataset. In order to calculate these values, code was written to approximate the clusters detected by IVAS as spheres. The complete code can be found in the script *VolumeRadCalc.py.*

During internal discussion of cluster analysis of IVAS, it became apparent that the method detailed above sometimes over- or underestimated the number of clusters in an APT sample. Due to this, I believed that use of another cluster analysis algorithm might be beneficial for future APT data analysis. Thus, I implemented a data clustering algorithm called DBSCAN, or density-based clustering of applications with noise, which uses a nearest neighbour algorithm to group points closely packed within other points (hence, noise) [11]. This was implemented using the package sklearn. Currently, there are no weightings and the algorithm itself is a skeleton. The four points it takes into account are spatial coordinates (x, y, z) and mass to charge ratio of the particle. After implementation of the RRNG to CSV and integration with the original CSV derived from the POS file, this algorithm can be more effectively used to categorize clusters. Due to their incomplete nature, the equivalent volume and radius calculator as well as the DBSCAN implementation are both not included in the applet, but can still be found in the Github.

11 Results and Discussion

All relevant documentation along with sample data sets, errors, and inputs and outputs are available on a Github repository. The README will soon be updated to reflect more details about specific aspects of each file, the prerequisite information required, as well as helpful tips for users. At the moment, all algorithms and routines written have been mainly for the purpose of automation, streamlining the process of menial tasks: because the programs written complete their tasks of automation, reorganization, and calculation the results are that all goals have been met to their intended degree.

In a discussion of the implemented formulation of Gibbs interfacial excess of solute, we must address the use of prompting lattice parameters and the underlying assumptions that follow. The ρ term in our use is for an extremely specific case; the spatial error may be calculated and revealed to negligible, yet it is still relevant to address the assumption of Bravais lattice structure in number density ($\rho = \frac{lattice\,points}{lattice\,parameter^3}$). We can begin to address the specifics by referencing the initial formulations for Krakauer & Seidman et al. (1993) for an APFIM TEM calculation of Gibbsian excess as well as Cahn's formulation, which does have limited application, to achieve a more rigorous description of ρ to calculate the excess [10] [12].

The question of user experience is also an important one, which has been addressed in brief but merits further consideration. The best choice is always for there to be complete automation, no work or *a priori* considerations necessary for use of programs (including experience in chemical analysis or computer science). While this could not be circumvented in earlier stages of the project, it was instead addressed by clear, thorough prompts for value inputs. The current interface application works by interacting with the terminal. In conjunction with the README, while there is much improvement to be made, this will function as a stopgap until consolidation, whether into a package with documentation or a multi-platform compliant applet, is completed.

12 Conclusion

After developing the toolkit outlined in the proposal, we wanted to answer the question of whether APT data analysis can be improved with respect to modularity, ease-of-use, streamlining, and optimization. The value of this project is in the consolidation and improvement of established analytical tools as well as the development of new analysis techniques, resulting in improvements to any field of research with the viability of using such a technique. Tentative extensions to this toolkit include the implementation of alternatives to the standard maximum separation algorithm used in cluster identification and an exploration of atom probe tip behavior simulations.

13 In Progress

There are many tasks for which automation is a possibility in the realm of Python-based APT data analysis. First, the question of consolidation and packaging of scripts must be answered. Currently, I am developing both a Python package as well as an independent applet through which users may work with proxigram and POS/EPOS CSV data files. The scripts that I am working on at the moment and will tentatively complete by the end of June 2021 are listed below. This is a complete list of next steps in and of itself, but higher priorities moving forward.

- Automating proxigram generation by converting range (RRNG) file to CSV and integrating with POS developed from CSV
- Executing peak detection and automatic ranging and isotope assignment using available isotope abundances table
- Modelling matrix composition to track precipitation as a function of time
- Developing DBSCAN implementation to further work with elemental clusters
- Developing SQL database in order to work with large amounts of data points more efficiently
- Developing colocation algorithm for particle clusters
- Optimization of Gibbsian interfacial excess of solute calculation algorithm
- Completing downloadable application using py2app or additional Github functionality
- *Additional tasks are to be determined*

14 Author's Note

Due to the nature of the current COVID-19 pandemic, it is difficult to develop research consistent with past APT research conducted through the Mentorship program at the NRL. Thank you for your continued patience and understanding during this time.

References

[1] S. M. Reddy, et al., "Atom Probe Tomography: Development and Application to the Geosciences," *Geostandards and Geoanalytical Research*, **44**(1), 5–50 (2020), doi:10.1111/ggr.12313.

[2] G. Da Costa, "Chapter Six - Atom Probe Tomography: Detector Issues and Technology," in *Atom Probe Tomography*, edited by W. Lefebvre-Ulrikson, F. Vurpillot, and X. Sauvage (Academic Press, 2016), pp. 155–181, ISBN 978-0-12-804647-0, doi:https://doi.org/10.1016/B978-0-12-804647-0.00006-1.

[3] M. Tamura, et al., "Focused ion beam gallium implantation into silicon," *Applied Physics A Solids and Surfaces*, **39**(3), 183–190 (1986), doi:10.1007/bf00620733.

[4] T. F. Kelly and M. K. Miller, "Atom probe tomography," *Review of Scientific Instruments*, **78**(3), 031101 (2007), doi:10.1063/1.2709758.

[5] S. Dhara, et al., "Atom probe tomography data analysis procedure for precipitate and cluster identification in a Ti-Mo steel," *Data in Brief*, **18**, 968–982 (2018), doi:10.1016/j.dib.2018.03.094.

[6] G. B. Vincent, A. P. Proudian, and J. D. Zimmerman, "Three dimensional cluster analysis for atom probe tomography using Ripley's K-

function and machine learning," *Ultramicroscopy*, **220**, 113151 (2021), doi:10.1016/j.ultramic.2020.113151.

[7] M. Kühbach, et al., "Building a Library of Simulated Atom Probe Data for Different Crystal Structures and Tip Orientations Using TAPSim," *Microscopy and Microanalysis*, **25**(2), 320–330 (2019), doi:10.1017/s1431927618016252.

[8] K. E. Knipling, P. U. Narayana, and L. T. Nguyen, "Microstructures and Properties of As-Cast AlCrFeMnV, AlCrFeTiV, and AlCrMnTiV High Entropy Alloys," *Microscopy and Microanalysis*, **23**(S1), 702–703 (2017), doi:10.1017/s1431927617004172.

[9] D. E. Perea, et al., "Atom Probe Tomographic Mapping Directly Reveals the Atomic Distribution of Phosphorus in Resin Embedded Ferritin," *Scientific Reports*, **6**(1) (2016), doi:10.1038/srep22321.

[10] O. C. Hellman and D. N. Seidman, "Measurement of the Gibbsian interfacial excess of solute at an interface of arbitrary geometry using three-dimensional atom probe microscopy," *Materials Science and Engineering: A*, **327**(1), 24–28 (2002), doi:10.1016/s0921-5093(01)01885-8.

[11] J. Sander, et al., "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications," *Data Mining and Knowledge Discovery*, **2**(2), 169–194 (1998), doi:10.1023/a:1009745219419.

[12] B. W. Krakauer and D. N. Seidman, "Absolute atomic-scale measurements of the Gibbsian interfacial excess of solute at internal interfaces," *Physical Review B*, **48**(9), 6724–6727 (1993), doi:10.1103/physrevb.48.6724.