doi:10.1017/S1431927621004116
Microsc. Microanal. 27 (Suppl 1), 2021

The evolution of an open source file format: a version control story

Benjamin Savitzky[1], Steven Zeltmann[2], Luis Rangel DaCosta[2], Peter Ercius[3], Mary Scott[4], Andrew Minor[4] and Colin Ophus[1]

[1]Lawrence Berkeley National Laboratory, California, United States, [2]UC Berkeley, United States, [3]Lawrence Berkeley National Laboratory, United States, [4]UC Berkeley, Berkeley, California, United States

Good data management is essential for good scientific practice, and is intimately related to the basic principle of reproducibility. Within the broad category of data management, the file format might be thought of as a basic abstraction level, and design choices made at this level have the potential to greatly simplify, or confound, downstream systems. Here, we discuss the past, present, and future of the EMD (Electron Microscopy Dataset) format.

An EMD 0.1 file is an HDF5 based fileformat designed to store arbitrary N-dimensional arrays while directly attaching the key metadata required for analysis and interpretation [1]. The HDF5 format is used widely in scientific computing, enables cross-platform and high performance data access, and allows flexible definition of directory hierarchies containing both data and metadata. It is enhanced by open source interfaces such as the h5py python interface, which make well-structured HDF5 files easy to interact with, either as human-readable files or for automated extraction. For the purposes of accessibility, reproducibility, and ease-of-use, HDF5 is therefore an excellent alternative to closed-source, proprietary file formats. Additional important strengths of the EMD 0.1 specification include its simplicity, and the tight grouping of relevant metadata with each data array, ensuring appropriate calibrations are always available. However, not all data is best stored as an N-dimensional array. A generalization of this format that allows for additional datastructures, but retains its basic simplicity and tight grouping of data/metadata, is desirable.

Four-dimensional scanning transmission electron microscopy (4D STEM) describes scanned probe experiment where a 2D image of the scattered electron beam is collected at each scan position in a 2D raster pattern. 4D STEM generates large data volumes, and is an area where good file and data management practices are particularly germane [2]. py4DSTEM is an open source python package for analysis of 4D STEM data [3]. It uses an HDF5 based format initially derived from the EMD 0.1 format to store N-D arrays, which also contains several additional datastructures that are useful in 4D STEM data analysis. These include an H5 representation of the numpy structured array class, a gridded set of structured arrays intended to map one such array to each scan position, and (separately) a sparse array implementation for electron counted data. By generalizing these structures, keeping specifications simple and metadata/data paired as appropriate, we've extended the original EMD file to broaden its utility while maintaining its basic principles. Implemented as EMD 0.8, we discuss the general approach, the implementation in py4DSTEM as a specific example, and the similarities and differences to other open formats, such as the USID format [4]. We aspire to make this format as clear, simple and accessible as possible to allow for maximal collaborative development with the academic and vendor communities.

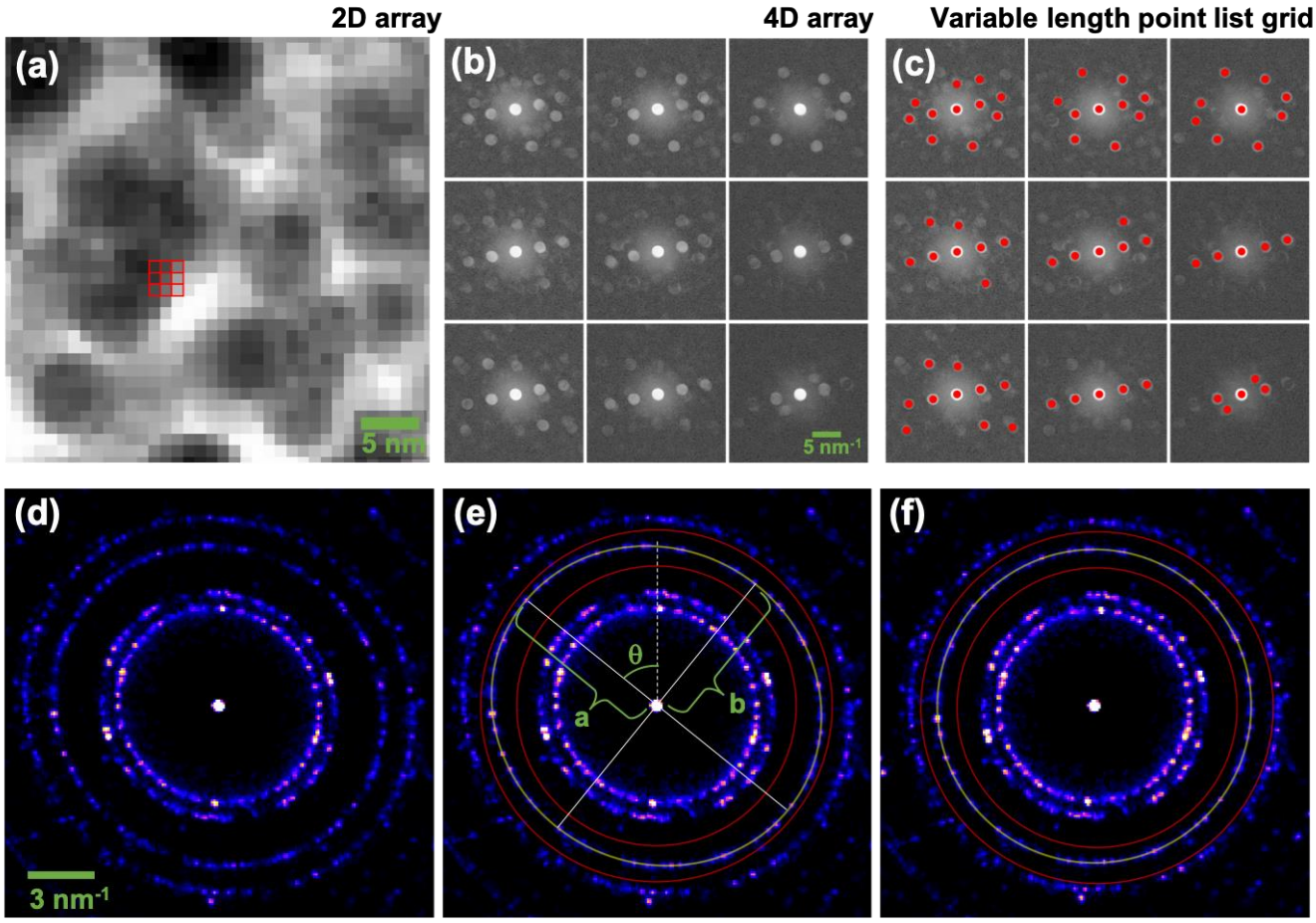

Figure 1.. A 4D STEM calibration workflow highlighting multiple data structures and levels of metadata. (a) A virtual bright field image, a 2D array of data, generated from a simulated 4D STEM dataset of polydisperse gold nanoparticles. (b) A small 3x3 subgrid of diffraction patterns, a 4D array of data, from the positions indicated in red in (a). (c) A set of positions indicating Bragg scattering vectors overlaid over their associated diffraction patterns. This data corresponds to a variable length list of m-dimensional points (here m=3: x, y and intensity). It is not best described as an N-D array, and should carry its own unique set of metadata. (d-f) An elliptical distortion correction routine, showing (d) a Bragg vector map, generated by summing the Bragg scattering from all diffraction patterns, (e) measurement of elliptical distortion of the data, and (f) re-measurement after elliptical correction. Throughout, metadata is shown in green, and we emphasize the existence of distinct levels of metadata: the scalebars (a,b,d) represent metadata at the level of specific datablocks, while the elliptical distortions (e) are metadata that is relevant to some subset of the data in the pipeline.

References

[1] https://emdatasets.com/format/

[2] Nord, Magnus, et al. "Fast pixelated detectors in scanning transmission electron microscopy. Part I: data acquisition, live processing, and storage." *Microscopy and Microanalysis* 26.4 (2020): 653-666.

[3] Savitzky, Benjamin H., et al. "py4DSTEM: A software package for multimodal analysis of four-dimensional scanning transmission electron microscopy datasets." *arXiv preprint arXiv:2003.09523* (2020).
[4] Somnath, Suhas, et al. "USID and Pycroscopy–Open Source Frameworks for Storing and Analyzing Imaging and Spectroscopy Data." *Microscopy and Microanalysis* 25.S2 (2019): 220-221.
[5] The authors gratefully acknowledge support from the Toyota Research Institute. Work at the Molecular Foundry was also supported by the Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. *SEZ was supported by STROBE, a NSF Science and Technology Center.*