2056
doi:10.1017/S1431927621007455
Microsc. Microanal. 27 (Suppl 1), 2021

NeXL: A Platform for Innovation in Microanalysis

Nicholas Ritchie and Dale Newbury

National Institute of Standards & Technology, Gaithersburg, Maryland, United States

NeXL is a series of loosely coupled software packages in the Julia language (https://julialang.org/) implementing X-ray microanalysis routines. NeXLCore (https://github.com/usnistgov/NeXLCore.jl, https://doi.org/10.18434/M32275) sits at the center of NeXL providing fundamental atomic and X-ray physics data and algorithms. In addition, it defines basic algorithms for handling materials, k-ratios and a Monte Carlo simulator of electron transport. NeXLMatrixCorrection (https://github.com/usnistgov/NeXLMatrixCorrection.jl, https://doi.org/10.18434/M32276) implements a handful of common matrix correction algorithms and a basic iteration algorithm. NeXLSpectrum (https://github.com/usnistgov/NeXLSpectrum.jl, https://doi.org/10.18434/M32286) implements energy dispersive X-ray spectrum manipulation and analysis tools for both spectra and hyper-spectra. It can read and write ISO/EMSA files and LISPIX-style RPL/RAW hyperspectra. NeXLUncertainties (https://github.com/usnistgov/NeXLUncertainties.jl, https://doi.org/10.18434/M32274) implements multivariate measurement models. Each package comes with a test suite to validate the code. User documentation is linked from the GitHub repository and each package is available for download through the Julia package manager.

Julia is an innovative computer language recently developed at MIT and used extensively within the mathematical and scientific community [Bezanson, 2017]. It combines the Python-like scripting with C-like speed. This combination makes it suitable for implementing low-level algorithms like Monte Carlo simulators that would be prohibitively slow in Python. Julia has an extensive third-party ecosystem of packages implementing sophisticated mathematical, scientific and computer science algorithms.

We will present two ways we've combined NeXL with third-party Julia libraries to facilitate innovative microanalysis. The first example is a *k-ratio database*. This combines the spectrum analysis tools in NeXLSpectrum with a Structured Query Language (SQL) database to facilitate data management and discovery. Spectra from materials of known composition along with suitable reference spectra are stored in the database. The known spectra are filter-fit and the resulting k-ratios are stored in the database. The measured k-ratios can be compared with k-ratios computed from the known compositions using the NeXLMatrixCorrection library. If the matrix correction model is correct, we expect the ratio of the measured k-ratio over the calculated k-ratio to be approximately unity. When the ratio deviates, this suggests either a measurement error or a matrix correction model failure. We have populated a database with measurements of materials from NIST's extensive collection of stoichiometric compounds, Standard Reference Materials, Certified Reference Materials and other established standard materials. This database can then be queried to extract and display spectra based on search criteria. For example, the topic of matrix correction model failure in transition metal L-lines has been of much recent interest [Gopon, 2013, Llovet, 2016]. An interactive Pluto notebook (https://github.com/fonsp/Pluto.jl) has been implemented which allows the user to extract materials by element and to plot a sub-set. In the example in Figure 1, four sets of measurements from the element Nickel at 15 keV were selected. The k-ratios from both the K- and L- families were fitted relative to a pure Ni standard and compared to the K-ratios computed using the Pouchou and Pichoir's XPP matrix correction model [Pouchou, 1991]. The ratio of the measured to computed k-ratios is plotted for the K- and L-families. The K-family measurements cluster around unity while the L-family measurements are seen to cluster at 1.2 but also near 1.5. This suggests

that the classic matrix correction algorithms are failing for Ni L but that C71300, NiTi and Pentlandite can work as standards for L-family measurement for each other but not for NiSi.

The second example is a neural network for matrix correction. This uses the Monte Carlo simulator in NeXLCore to compute thousands of $\varphi(\rho z)$ for various materials, atomic shells, and beam energies. These curves are then used to train a neural network implemented using the Flux package (https://fluxml.ai/) [Innes, 2018] and executed on a GPU using the CUDA (https://github.com/JuliaGPU/CUDA.jl) package. While a neural network is a black box, it is also capable of universal function approximation [Hornik, 1989]. A neural network trained on a high-quality physics model could potentially generate accurate $\varphi(\rho z)$ curves with the computational efficiency of a neural network. We have had some success training networks as is demonstrated in Figure 1. However, when we do more extensive validation, we find that a small fraction of the generated $\varphi(\rho z)$ curves deviates far enough to become useless. While current implementations fall short of our goal, they represent a promising start. With NeXL and the Julia package infrastructure, it was possible to implement a functional prototype in a couple days.

While NIST DTSA-II (https://cstl.nist.gov/div837/837.02/epq/dtsa2/index.html) has seen broad use, we have been disappointed that few people have taken the opportunity to extend it and repurpose it. In part, we believe this is because the graphical user interface makes interactive use easy but code reuse harder. NeXL comes from the opposite direction. The initial learning curve is steep, but it is designed to extend and repurpose. The loosely coupled architecture of the NeXL packages allows users to pick the parts they need and allows users to contribute either directly to NeXL or in the form of their own NeXL-friendly packages. We hope this will help others in our community to take their ideas quickly from conception to implementation to community use.

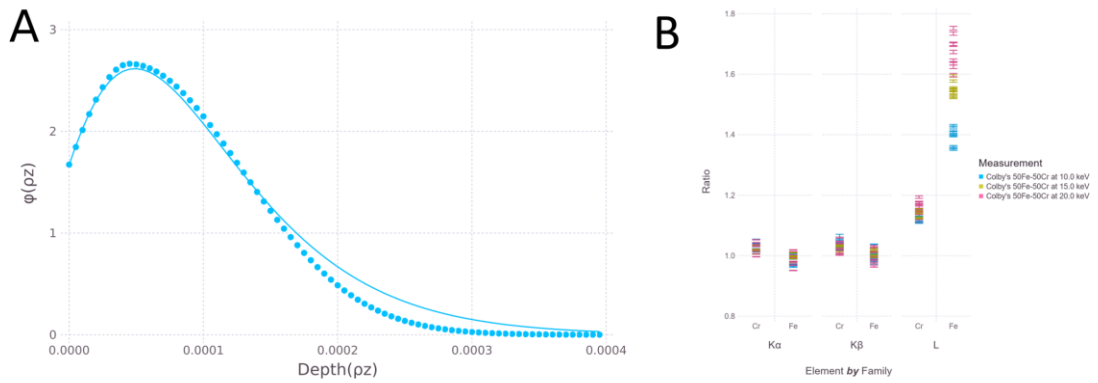

Figure 1. A) Comparing a curve computed by neural network (dots) with one computed using XPP for Al K in Fe at 10 keV. B) Comparing the ratio of the measured k-ratio over the calculated k-ratio for measurements of a 50% Fe / 50% Cr alloy at 10 keV, 15 keV and 20 keV. Nominally the ratio should be unity when the matrix correction algorithm is working correctly.

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, *59*(1), 65-98.

Hornik, Stinchcombe & White (1989) *Multilayer Feedforward Networks are Universal Approximators* Neural Networks, Vol. 2, pp. 359-366.

Innes, M. (2018). Flux: Elegant machine learning with Julia. *Journal of Open Source Software*, *3*(25), 602.

https://doi.org/10.1017/S1431927621007455 Published online by Cambridge University Press

Gopon, P., Fournelle, J., Sobol, P. E., & Llovet, X. (2013). Low-voltage electron-probe microanalysis of Fe-Si compounds using soft X-rays. *Microscopy and Microanalysis*, *19*(6), 1698.
Llovet, X., Pinard, P. T., Heikinheimo, E., Louhenkilpi, S., & Richter, S. (2016). Electron probe microanalysis of Ni silicides using Ni-L X-ray lines. *Microscopy and Microanalysis*, *22*(6), 1233-1243.
Pouchou, J. L., & Pichoir, F. (1991). Quantitative analysis of homogeneous or stratified microvolumes applying the model "PAP". In *Electron probe quantitation* (pp. 31-75). Springer, Boston, MA.

https://doi.org/10.1017/S1431927621007455 Published online by Cambridge University Press