PYTHON SPECTRAL ANALYSIS TOOL (PYSAT) FOR PREPROCESSING, MULTIVARIATE ANALYSIS, AND MACHINE LEARNING WITH POINT SPECTRA R.B. Anderson[1], N. Finch[1], S. Clegg[2], T. Graff[3], R.V. Morris[3], J. Laura[1]; [1]U.S. Geological Survey, Astrogeology Science Center, Flagstaff, AZ (rbanderson@usgs.gov); [2]Los Alamos National Laboratory, [3]NASA Johnson Space Center.

Introduction: Spectroscopic planetary data sets are inherently more difficult to work with than images, because they often require a deeper familiarity with the instrument and data processing steps to derive scientifically meaningful results. However, spectroscopic data are also vital to understand the properties and geologic history of planetary bodies.

The quantity of spectroscopic data available from surface instruments is also increasing rapidly. For example, ChemCam on the Curiosity rover [1,2] has returned more than 400,000 spectra to date and SuperCam on the upcoming Mars 2020 mission [3] is expected to be similarly prolific. It is vital that the planetary science community be able to easily analyze the data from these and other spectroscopic instruments.

We are therefore developing a library of spectral analysis software and an associated graphical user interface to enable any scientist to process point spectral data without requiring programming expertise. The focus of development to date has been on implementing the types of preprocessing, multivariate analysis and machine learning methods that are used in the calibration and analysis of data from ChemCam, but these methods can easily be applied to other spectral data, and additional capabilities such as clustering and classification will be added in the future. This work is complementary to PySAT development geared toward orbital data [4].

Data Format: PySAT relies on the Pandas [5] library to efficiently store spectra and associated metadata in a single data frame. The PySAT format for point spectral data is a comma-separated value (.csv) file with spectra and associated metadata stored in rows. Two header rows represent multi-indexes for the data frame, with the top row indicating broad categories of data (e.g. wavelength, metadata, composition) and the second row indicating specific categories (e.g. '240.811', 'Target Name', 'SiO_2'). PySAT has a built-in function to read individual PDS-format data files from ChemCam into the PySAT .csv format, and laboratory data in the proper format will be included in the repository.

Capabilities:

Preprocessing: To ensure that regression models can be trained, rows of the dataframe that are missing the variable of interest can easily be filtered out. Spectra in one data frame can also be interpolated onto the spectral channels from another data frame, a first step in combining data from different instruments. A mask, specified by a simple .csv file, can be applied to all of the spectra in a data frame, and spectra can be normalized to the integrated signal within specified wavelength range(s). Spectra can also be grouped into a number of "folds" in preparation for model validation and testing, with the folds stratified on a single column of the data frame's metadata (for example, the SiO_2 content) to ensure a similar distribution of that variable in each fold. Dimensionality reduction, using principal components analysis (PCA) or two different independent component analysis (ICA) algorithms can also be applied as a preprocessing step. The PySAT library also includes a number of continuum removal algorithms provided by [6]. These have not yet been connected to the graphical interface.

Regression: Regression methods can be used to generate models capable of converting observed spectra into chemical compositions or other predictions of the target properties. PySAT makes use of the scikit-learn machine learning library [7] to allow users to train a variety of regression models. Currently the following methods are implemented: ordinary least squares (OLS), partial least squares (PLS), Gaussian process regression (GPR), orthogonal matching pursuit (OMP), and Lasso (refer to [7] for references for these and other methods). To assist users in identifying the optimal parameters for these methods, a flexible cross validation option is available using the stratified folds defined in preprocessing.

Once a regression model has been trained, it can be used for prediction. If multiple models have been trained, they can be blended together to implement submodel regression [8]. The current ChemCam calibration uses blended PLS submodels [9], but there is no requirement in PySAT that the same regression method be used for all submodels, providing added flexibility. The ranges over which the submodels are blended in PySAT can optionally be optimized based on performance on training data.

Visualization: PySAT also includes Matplotlib-based [10] options for producing point and line plots, as well as plots of scores and loadings to visualize PCA and ICA results. Figure 1 shows an example plot comparing several different regression algorithms.

Interface: We have also developed a graphical user interface based on PyQt4 [11] to make all of the above capabilities accessible for non-programmers. The graphical interface is based on the concept of "workflows", comprising individual processing steps

or "modules" arranged in a user-specified order. The interface includes a progress bar to indicate when the program is running a calculation. As each module is run, it is grayed-out to indicate progress through the workflow (Fig. 2).

Once the user has determined the optimal order for the steps in the workflow, it can be saved and restored at a later date. This saving and restoration process uses Python's "Pickle" capability, and provides a convenient way to store and later replicate the exact processing steps involved in deriving results from a data set. For example, the methods and plotting commands used to produce the results and figures in a publication could be stored and included as supplemental information along with the manuscript.

Future Work: The PySAT point spectra tool GUI is available at [12] and the underlying code containing the core functionality of PySAT for point spectra is at [13]. As development continues, several additional capabilities will be added, including: upgrading to PyQt5, adding a graphical interface for continuum removal; basic spectral math; peak area calculations; more dimensionality reduction and regression options; clustering and classification; calibration transfer; and improved data input, manipulation, and visualization options. The course of development and prioritization of different features will also be guided by feedback from users.

Although the tool was designed primarily with laser-induced breakdown spectroscopy (LIBS) analysis in mind, it can be easily applied to any spectral data if formatted correctly. The tool provides a flexible and powerful framework enabling scientists to process and analyze spectral data using multivariate and machine learning methods, leading to improved scientific results.

References: [1] Maurice, S., et al. (2012) Space Sci. Rev. 170, 95–166. doi:10.1007/s11214-012-9912-2 [2] Wiens, R.C., et al. (2012) Space Sci. Rev. 170, 167–227. doi:10.1007/s11214-012-9902-4 [3] Wiens, R.C., et al. (2016) 47th LPSC, #1322. [4] Gaddis et al., this volume. [5] http://pandas.pydata.org/ [6] Giguere, S., Carey, C.J., Boucher, T., et al. (2013) Proc. 5th IJCAI Workshop on Artificial Intelligence in Space. [7] http://scikit-learn.org/ [8] Anderson, R.B., et al. (2017) Spectrochim. Acta B, 129, 49–57. doi:http://dx.doi.org/10.1016/j.sab.2016.12.003 [9] Clegg, S. et al. (2017), Spectrochim. Acta B, 129, 64-85. http://doi.org/10.1016/j.sab.2016.12.003 [10] http://matplotlib.org/ [11] https://riverbankcomputing.com/software/pyqt/intro [12] https://github.com/USGS-Astrogeology/PySAT_Point_Spectra_GUI [13] https://github.com/USGS-Astrogeology/PySAT

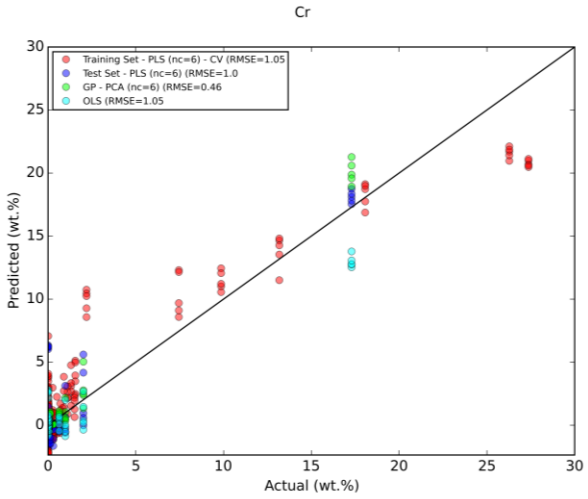

Figure 1: Example plot showing predicted chromium content using ChemCam laboratory spectra and several different algorithms, illustrating how the tool can be used to rapidly develop and compare results from different workflows.

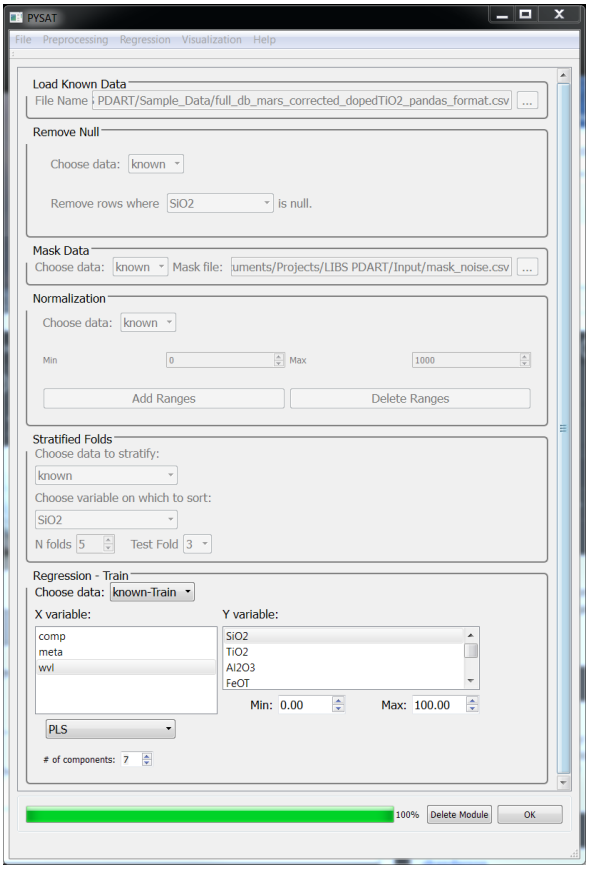

Figure 2: Example workflow in the PySAT point spectra GUI. The workflow loads data, removes spectra with no SiO_2 composition, masks and normalizes the spectra, separating them into 5 stratified folds with fold 3 set aside as a test set, and trains a PLS regression model with 7 components.