CHEMICAL ENGINEERING
UNIVERSITY of WASHINGTON

Random Forests and Image Processing for Photoluminescence Prediction

Wesley Beckner, Garrett Davidson, and Jessica Kong

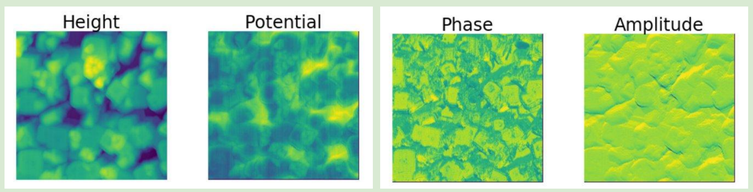
https://github.com/wesleybeckner/afm-miner

Background

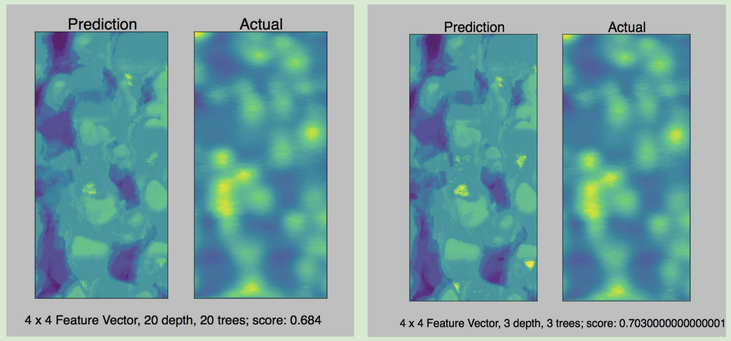
AfmMiner uses **Random Forests**, a kind of non-parametric supervised learning algorithm, to discover materials properties from atomic force microscopy (AFM) images. In particular, afmMiner can predict the photoluminesence of a material given some topographical and electrical AFM data. AfmMiner addresses a fundamental question: **can we predict photoluminescence from things that are not diffraction limited,** and allows the user to uncover relationships between topographical, electrical, and photographic measurements of their material.

Functions and Overview

afmMiner, what's under the hood? afmMiner uses the sklearn ensemble module RandomForestRegressor to process sets of AFM images and predict non-diffraction limited spatial mapping of the material's photoluminescence. This ensemble learner trains a number of classifying decision trees on the user's provided training data (pixels representing the AFM measurements) and averages them to create the final model, example input images are shown below (note: a photoluminescence image is also provided for training/testing, see figures at bottom).



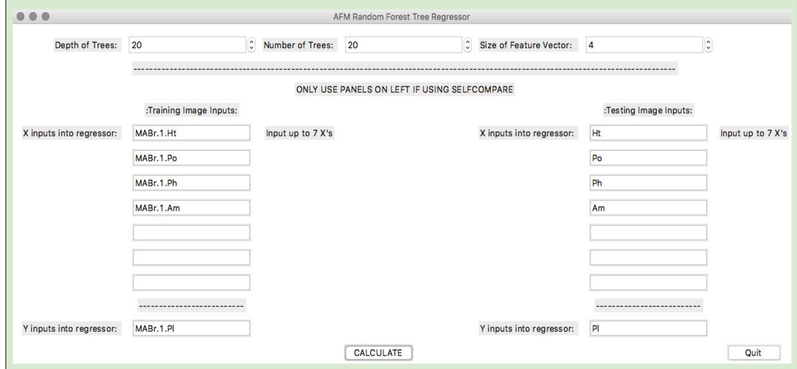
The afmMiner user has control over tree depth, number of trees, input images, and a final parameter, the "feature vector" that represents the array size of surrounding pixels that are used as inputs to predict the center pixel. This last option was implemented as a way to contextualize a given pixel with its neighbors and reduce the effect of noisey data.



afmMiner.selfCompare() output

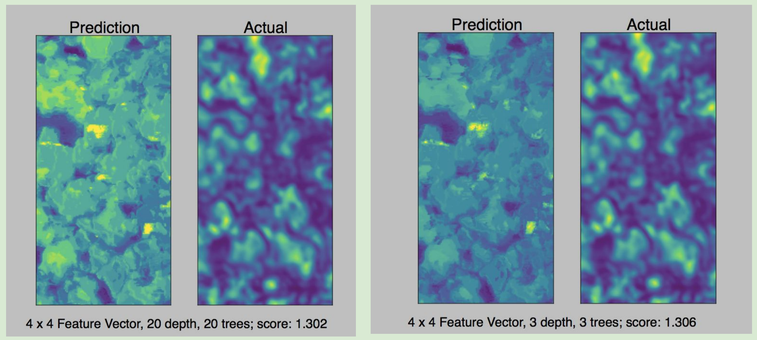
GUI Design

The GUI uses tkinter to display the interface. The window allows users to input the depth of the trees, the number of trees in the forest, the size of the feature vector and the input images to train and test against. Based on the columns used for the inputs, the function will run either selfCompare or crossCompare and output the final prediction. As a result, the user can easily adjust the parameters to find the best model for the images. The images use 'viridis' color mapping to differentiate high and low pixel values.



afmMiner.selfCompare(): selfCompare trains and tests on the same material essentially removing material-specific artifacts that may arise between training and testing sets. Increased trees and depth improve the prediction, see figures to the bottom left.

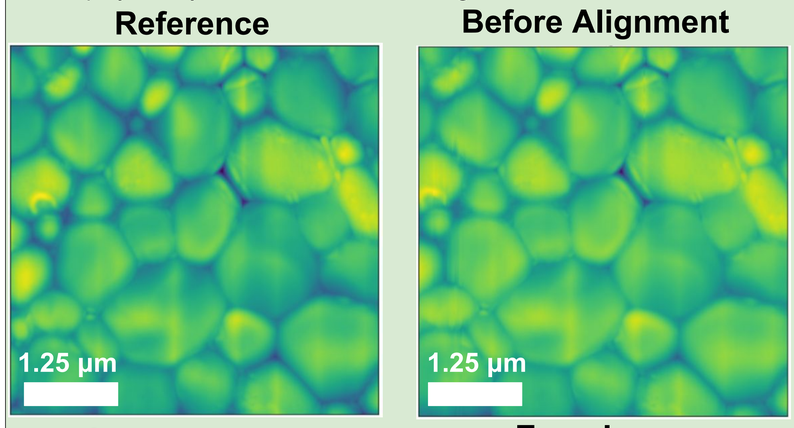
afmMiner.crossCompare(): crossCompare trains and tests on separate materials, given that the AFM input-types are the same. The overall higher error than selfCompare indicates that some material-specific artifacts may arise, see figures below.

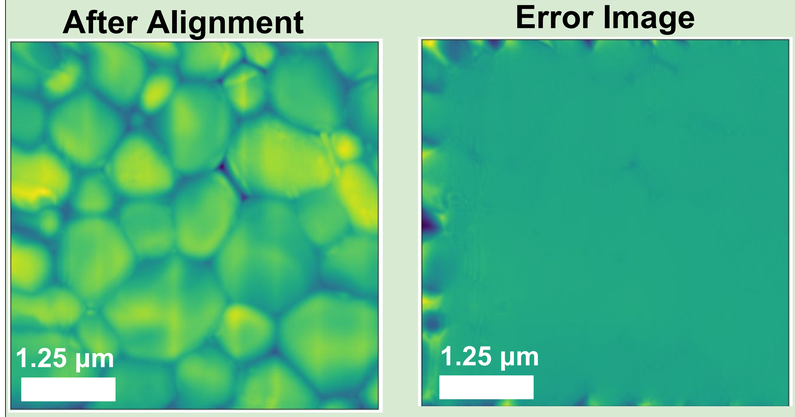


afmMiner.crossCompare() output

Image Alignment

Before accurate predictions can be made, the images provided to the algorithm need to be physically aligned with one another according to a topographical reference. For this purpose we employ the Symmetric Normalization algorithm.





The Symmetric Normalization algorithm proposed by Avants, et al registers two topography images. A Gaussian pyramid is built and the alignment is optimized at each level. The transformation is optimized with a built in Symmetric Diffeomorphic Registration function which produces transformation matrices that can be used to map images back and forth.

References

[1] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[2] Garyfallidis E, Brett M, Amirbekian B, Rokem A, van der Walt S, Descoteaux M, Nimmo-Smith I and Dipy Contributors (2014). Dipy, a library for the analysis of diffusion MRI data. Frontiers in Neuroinformatics, vol.8, no.8.

[3] Avants, B. B., Epstein, C. L., Grossman, M., & Gee, J. C. (2009). Symmetric Diffeomorphic Image Registration with Cross- Correlation: Evaluating Automated Labeling of Elderly and Neurodegenerative Brain, 12(1), 26-41.

Seattle, WA, March 2017