RESEARCH

Materials Knowledge Systems in Python - A Data Science Framework for Accelerated Development of Hierarchical Materials

David B Brough[1], Daniel Wheeler[2] and Surya R. Kalidindi[1,3*]

[*]Correspondence: surya.kalidindi@me.gatech.edu
[1]School of Computational Science and Engineering, Georgia Institute of Technology, 30332, Atlanta, USA
Full list of author information is available at the end of the article

RANDOM IDEAS
TALK ABOUT PYTHON, SCIKIT-LEARN METHODOLOGY BEING A WAY TO UNIFY DIFFERENT MATERIALS SCIENCE LEARNING AND DATA TRANSFORMS INTO SINGLE PIPELINES (EITHER ABSTRACT OR CONCLUSION OR ELSEWHERE). EASY TO SHARE PIPELINES AND EMBED IN OTHER MORE COMPLEX PIPELINES.
TALK ABOUT WHAT PYMKS ADDS ABOVE AND BEYOND SCIKIT-LEARN IN ABSTRACT. IT ADDS BOTH THE CONVOLUTION LEARNING AND PERIODICTY/INDEXING ISSUES TO DO WITH MICROSTRUCTURE. ANSWER QUESTION OF WHY NOT JUST USE SCIKIT-LEARN? NEEDS TO BE CLEAR.

Abstract

To Do

Keywords: TBD

1 Introduction

Current practices for developing tools and infrastructure used in multiscale materials design, development, and deployment are generally highly localized (sometimes even within a single organization) resulting in major inefficiencyies (duplication of effort, lack of code review, not engaging the right talent for the right task, etc.). Although it is well known that the pace of discovery and innovation significantly increases with effective collaboration [1–4], scaling such efforts to large heterogeneous communities such as those engaged in materials innovation has been very difficult.

The advent of information technology has facilitated massive electronic collaborations (generally referred to as e-collaborations) that have lead to significant advances in several domains including the discovery of the Higg's boson [5], the sequencing of the human genome [6], the Polymath project [7], the monitoring of species migration [8, 9] and numerous open source software projects. E-collaborations allow experts from complementary domains to create highly productive collaborations that transcend geographical, temporal, cultural, and organizational distances. E-collaborations require a supporting cyber-infrastructure that allows team members to generate, analyze, disseminate, access, and consume information at dramatically increased pace and/or quantity [10]. A key element of this emerging cyber-infrastructure is open source software as it eliminates collaboration hurdles due to software licenses and can help foster truly massive e-collaborations. In other words, even with collaborations involving proprietary data, open source cyberinfrastructure provide a common language that can facilitate e-collaborations with large numbers of team members (could even become a community effort).

Several recent national and international initiatives [11–13] have been launched with the premise that the adoption and utilization of modern data science and informatics toolsets offers a new opportunity to accelerate dramatically the design and deployment cycle of new advanced materials in commercial products. More specifically, it has been recognized that innovation cyber-ecosystems are needed to allow experts from the materials science and engineering, design and manufacturing, and data science domains to collaborate effectively. The challenge in integrating these traditionally disconnected communities comes from the vast differences in how knowledge is captured, curated, and disseminated in these communities [14]. More specifically, knowledge systems in the materials field are rarely captured in a digital form. In order to create a modern materials innovation ecosystem, it is imperative that we design, develop, and launch novel collaboration platforms that allow automated distilling of materials knowledge from large amounts of heterogeneous data acquired through customized protocols that are necessarily diverse (elaborated next). It is also imperative that this curated materials knowledge is presented to the design and manufacturing experts in highly accessible (open) formats.

Customized materials design has great potential for impacting virtually all emerging technologies, with significant economic consequences [12, 13, 15–23]. However, materials design (including the design of a manufacturing process route) resulting in the combination of properties desired for a specific application is a highly challenging inverse problem due to the hierarchical nature of materials internal structure. Material properties are controlled by the hierarchical internal structure (over multiple length scales which spans from atomic to macroscopic) as well as coupled physical phenomena which can occur at different timescales at each of the hierarchical length scales. Characterization of the structure at each of these different length scales is often in the form of images which come from different experimental/computational techniques resulting in highly heterogeneous data. As a result, tailoring the material hierarchical structure to yield desired combinations of properties or performance characteristics is enormously difficult. Figure 1 provides a collection of materials images depicting material structures at different length scales, which are generally acquired using diverse protocols and are captured in equally diverse formats.

While the generation (from experiments and computer simulations) and dissemination of datasets consisting of heterogeneous images are necessary elements in a modern materials innovation ecosystem, there is an equally critical need for customized analytics that take into account the stochastic nature of these data at multiple length scales in order to extract high value, transferable, knowledge. Data-driven Process-Structure-Property (PSP) linkages [25] provides a systemic, modular, and hierarchical framework for community engagement (i.e., several people making complementary or overlapping contributions to the overall curation of materials knowledge). Computationally cheap PSP linkages also communicate effectively the curated materials knowledge to design and manufacturing experts in highly accessible formats.

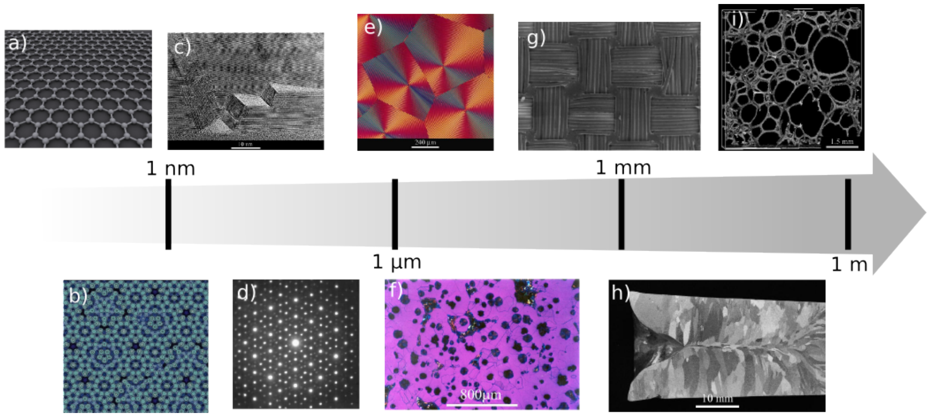


Figure 1 Heirarchical Materials structure at multiple length scales a). Simulated graphene crystalline structure. b). Simulated fivefold icosahedral Al-Ag quasicrystals. c).High resolution electron microscopy image of delamination cracks in h-BN particles subjected to compressive stress in the (0001) planes (within a silicon nitride particulate-reinforced silicon carbide composite. d). Electron diffraction pattern of an icosahedral Zn-Mg-Ho quasicrystal. e). Cross-polarised light image of spherulites in in poly-3-hydroxy butyrate (PHB) f). Cast iron with magnesium induced spheroidised graphite. g). SEM micrograph of a taffeta textile fragment h). Optical microscopy image of a cross-section of an aluminium casting i). X-ray tomography image of open cell polyurethane foam. Images courtesy of Core-Materials [24].

The Materials Knowledge Systems in Python project (PyMKS) is the first open source materials data analytics toolkit that can be used to create high value PSP linkages for hierarchical materials in large scale efforts driven and directed by an entire community of users. In this regard, it could be a foundational element of the cyber-infrastructure needed to realize a modern materials innovation ecosystem.

2 Current Materials Innovation Ecosystem

Open access materials databases and computational tools are critical components of the cyber-infrastructure needed to curate materials knowledge through effective e-collaborations [26]. Several materials science open source computational toolsets and databases have emerged in recent years to help realize the vision outlined in the Materials Genome Initiative (MGI) and the Integrated Computational Materials Engineering (ICME) paradigm [12, 13, 15–23]. Yet, the creation and adoption of a standard materials taxonomy and database schema has not been established due to the unwieldy size of material descriptors and heterogeneous data. Additionally, the coupled physical phenomena that govern material properties is too complex to model all aspects of a material simultaneously using a single computational tool. Consequently, current practices have resulted in the development of computation tools and databases with a narrow focus on specific length/structure scales, material classes, or properties.

NIST Data Gateway contains over 100 free and paid query-able web-based materials databases. These databases contain atomic structure, thermodynamics, kinetics, fundamental physical constants, x-ray spectroscopy, among other features [27]. NIST DSpace provides a curation of links to several materials community databases [28]. NIST Materials Data Curation Systems (MDCS) is a general online database that aims to facilitate the capturing, sharing, and transforming of materials data [29]. Open Quantum Materials Database (OQMD) is an open source data repository for phase diagrams and electronic ground states computed using density functional theory [30]. MatWeb is a database containing materials properties for over 100,000 materials [31]. Atomic FLOW of Materials Discovery (AFLOW) databases millions of materials and properties and hosts computational tools that can be used for atomic simulations [32]. The Materials Project (and the tool pyMatgen) [33, 34] provides open web-based access to computed information on known and predicted materials as well as analysis tools for electronic band structures. The Knowledgebase of Interatomic Models (OpenKIM) hosts open source tools for potentials for molecular simulation of materials [35]. PRedictive Integrated Structural Materials Science (PRISMS) hosts a suite of ICME tools and datastorage for the metals community focused on microstructure evolution and mechanical properties [36].

SPPARKS Kinetic Monte Carlo Simulator (SPPARKS) is a parallel Monte Carlo code for on-lattice and off-lattice models [37]. MOOSE is a parallel computational framework for coupled systems of nonlinear equations [38]. Dream3D is a tool used for synthetic microstructures generation, image processing and mesh creation for finite element [39].

While there exits a sizable number of standard analytics tools [40–49], none of them are tailored to create PSP linkages from materials structure image data and their associated properties. PyMKS aims to seed and nurture an emergent user group in the materials data analytics for establishing homogenization and localization (PSP) linkages by leveraging open source signal processing and machine learning packages in Python. An overview of the PyMKS project accompanied with several examples is presented here. This paper is a call to others interested in participating in this open science activity.

3 Theoretical Foundations of Materials Knowledge Systems

Material properties are controlled by their internal structure and the diverse physical phenomena occurring at multiple time and length scales. Generalized composite theories [50, 51] have been developed for hierarchical materials exhibiting well separated length scales in their internal structure. Generally speaking, these theories either address homogenization (i.e., communication of effective properties associated with the structure at a given length scale to a higher length scale) or localization (i.e., spatiotemporal distribution of the imposed macroscale loading conditions to the lower length scale). Consequently, homogenization and localization are the essential building blocks in communicating the salient information in both directions between hierarchical length/structure scales in multiscale materials modeling. It is also pointed out that localization is significantly more difficult to establish, and implicitly provides a solution to homogenization.

The most sophisticated composite theory available today that explicitly accounts for the full details of the material internal structure (also simply referred as microstructure) comes from the use of perturbation theories and Green's functions [50, 52–63]. In this formalism, one usually arrives at a series expansion for both homogenization and localization, where the individual terms in the series involve convolution integrals with kernels based on Green's functions. This series expansion was refined and generalized by Adams and co-workers [62, 64, 65] through the introduction of the concept of a microstructure function, which conveniently separates each term in the series into a physics-dependent kernel (based on Green's functions) and a microstructure-dependent function (based on the formalism of n-point spatial correlations [56–61]).

Materials Knowledge Systems (MKS) [66–72] complements these sophisticated physics-based materials composite theories with a modern data science approach to create a versatile framework for extracting and curating multiscale PSP linkages. More specifically, MKS employs a discretized version of the composite theories mentioned earlier to gain major computational advantages. As a result, highly adaptable and templatable protocols have been created and used successfully to extract robust and versatile homogenization and localization metamodels with impressive accuracy and broad applicability over large microstructure spaces.

The MKS framework is based on the notion of a microstructure function. The microstructure function provides a framework to represent material structure terms of quantities such as phase identifiers, lattice orientation, chemical composition, defect types and densities, among others refered to as local states. The microstructure function, $m_j\left(h;s\right)$, represents a probability distribution for the given local state, $h \in H$, at each position, $s \in S$, in a given microstructure, j [73–76]. The introduction of the local state space H (i.e., the complete set of all potential local states) provides a consolidate variable space for combining the diverse attributes (often a combination of scalar and tensor quantities) needed to describe the local states in the material structure. The MKS framework requires a discretized description of m_j, which is denoted here as $m_j\left[h;s\right]$, where the $[\cdot;\cdot]$ represent the discretized space (in contrast to $(\cdot;\cdot)$, which defines the continuous space). The ";" symbol divides indices in physical space to the right of ";" from indices in state space to the left of ";". In most applications, S is simply tessellated into voxels on a regular (uniform) grid such that the position can be given by $s \to i, j, k$ in three dimensions.

As noted earlier, the local state space in most advanced materials is likely to demand sophisticated representations. In prior work [68, 77, 78], it was found that spectral representations on functions on the local state space offered many advantages both in compact representation as well as in reducing the computational cost. In such cases, h indexes the spectral basis functions employed. The selection of these functions depends on the nature of local state descriptors. Examples include: (i) the primitive basis (or indicator functions) used to represent simple tessellation schemes [66, 67, 69–74, 79], (ii) generalized spherical harmonics used to represent functions over the orientation space [68, 77], and (iii) Legendre polynomials used to represent functions over the concentration space [78].

3.1 Homogenization

Comparing different microstructures is quite difficult even after expressing them in convenient discretized descriptions mainly due to the lack of a reference point or a natural origin for the index s in the tessellation of the microstructure volume. Yet the relative spatial distributions of the local states provide a valuable representation of the microstructure that can be used effectively to quantify the microstructure and compare it with other microstructures in robust and meaningful ways [72–74, 76, 79]. The lowest order of spatial correlations comes in the form of 2-point statistics and

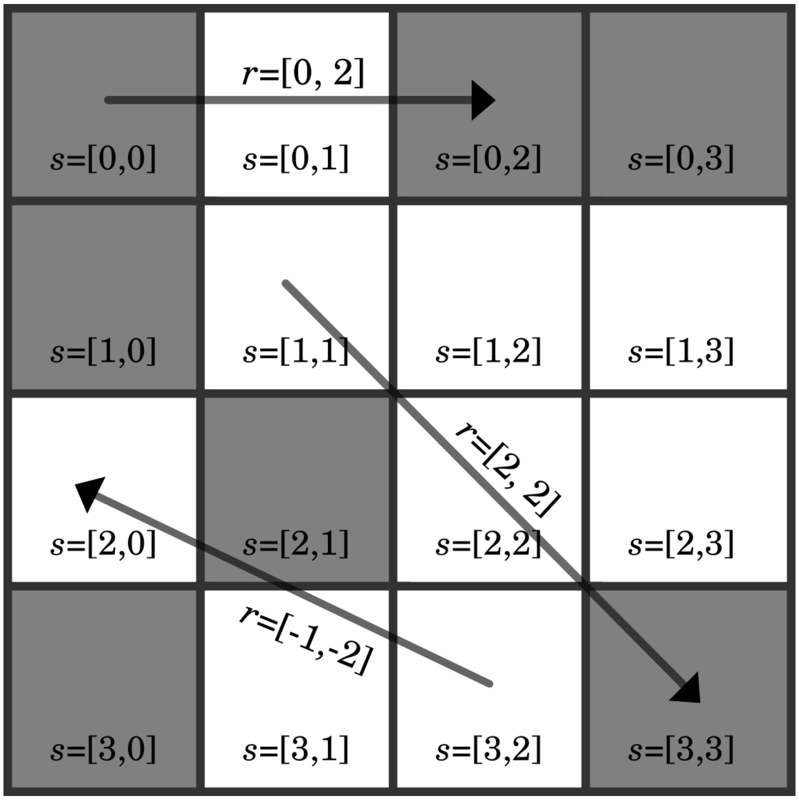


Figure 2 Caption

can be computed as a correlation of a microstructure function such that

$$f_j[h, h'; r] = \frac{1}{\Omega_j\,[r]} \sum_s m_j[h; s] m_j[h'; s + r] \tag{1}$$

where r is a discrete spatial vector within the voxelated domain specified by s, $f_j[h, h'; r]$ is one set of 2-point statistics for the local stats h and h' and $\Omega_j\,[r]$ is a normalization factor that depends on r [79]. The subscript j refers to a sample microstructure used for analysis (i.e. each j could refer to a microstructure image). The physical interpretation of the 2-point statistics is explained in Fig. 2 with a highly simplified two-phase microstructure (the two phases are colored white and gray). If the the primitive basis is used to discretize both the spatial domain and the local state space then $f_j[h, h'; r]$ can be interpreted as the probability of finding local states h and h' at the tail and head of the vector r, respectively.

2-Point statistics provide a meaningful representation of the microstructure, but create an extremely large feature space that often contains redundant information. Dimensionality reduction can be used to create low dimensional microstructure descriptors from the sets of spatial correlations (based on different selections of h and h') with principal component analysis (PCA). The PCA dimensionality reduction can be mathematically expressed as

$$f_j\,[l] \approx \sum_{k \in K} \mu_j\,[k]\,\phi\,[k, l] + \overline{f[l]} \tag{2}$$

In Eq. 2, $f_j\,[l]$ are the low rank approximations of 2-point statistics, where the $f_j\,[h,h';r]$ indices from Eq. 1 are represented as a vector with single index given by $f_j\,[l]$ where l is a unique integer for every combination of h, h' and r. The $\mu_j[k]$ are low dimensional microstructure descriptors (the transformed 2-point statistics) or principal component scores (PC scores). The $\phi\,[k,l]$ are the calibrated principle components (PCs) and the $\overline{f[l]}$ are the mean values from the calibration ensemble of $f_j\,[l]$ for each l. The $k \in K$ indices refer to the $\mu_j\,[k]$ in decreasing order of significance and are independent of l, l' and r. The main advantage of this approach is that the $f_j\,[l]$ can be reconstructed to sufficient fidelity with only a small subset of $\mu_j\,[k]$ [**?**].

After obtaining the needed dimensionality reduction in the representation of the material structure, machine learning models can be used to create homogenization PSP linkages of interest. As an example, a generic homogenization linkage can be expressed as

$$p_j^{\text{eff}} = \mathcal{F}(\mu_j[k]) \tag{3}$$

In Eq. 3, p_j^{eff} is the effective materials response (reflecting an effective property in structure-property linkages or an evolved low dimensional microstructure descriptor in process-structure linkages), and $\mathcal{F}$ is a machine learning function that links $\mu_j[k]$ to p_j^{eff}.

3.2 Localization

MKS Localization linkages are significantly more complex than the homogenization linkages. These are usually expressed in the same series forms that are derived in the general composite theories, while employing discretized kernels based on Green's functions [50, 52–63]. Mathematically, the MKS localization linkages are expressed as

$$p_j[s] = \sum_{h;r} \alpha[h;r] m_j[h;s-r] + \sum_{h,h';r,r'} \alpha[h,h';r,r'] m_j[h;s-r] m_j[h';s-r'] + ... \tag{4}$$

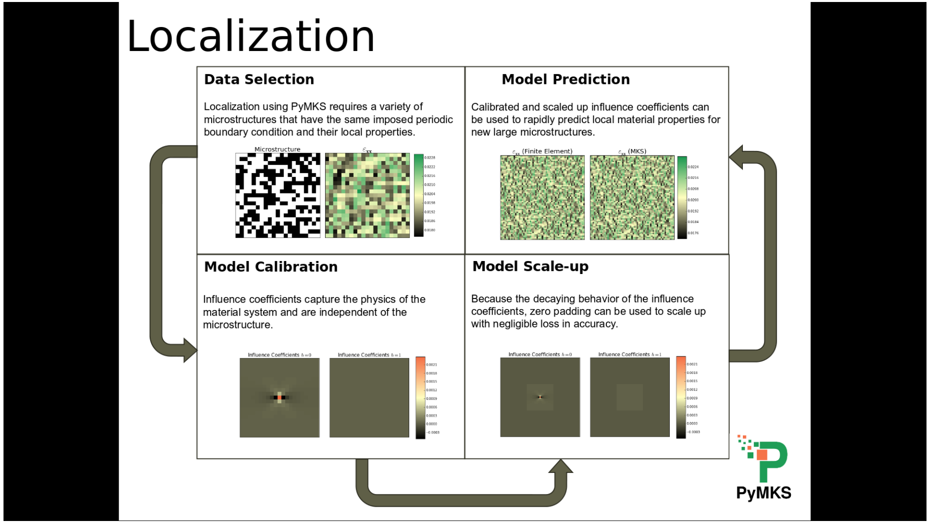
In Eq. 4, $p_j[s]$ is the spatially resolved (localized) response field (e.g. a response variable such as stress or strain rate in a structure-property linkage, or an evolved microstructure function in a process-structure linkage), and $\alpha[h;r]$ are the Green's function based discretized influence kernels. These digital kernels are calibrated using regression methods [66–69, 77, 78].

Figs. 4 and 3 provide schematic overviews of the MKS homogenization and localization workflows. THE CURRENT IMAGES ARE JUST PLACE HOLDERS. More detailed explanations on the MKS homogenization and localization linkages can be found in prior literature [66–74, 79].

4 Materials Knowledge Systems in Python

PyMKS is an object-oriented numerical implementation to the MKS theory developed in the literature [67]. It provides a high-level, computational efficient framework to implement data pipelines for classification, cataloging and quantifying materials structures for PSP relationships. PyMKS is written in Python, a natural

Figure 3 Localization Figure To Do - Place holder



Localization

Data Selection

Localization using PyMKS requires a variety of microstructures that have the same imposed periodic boundary condition and their local properties.

Model Prediction

Calibrated and scaled up influence coefficients can be used to rapidly predict local material properties for new large microstructures.

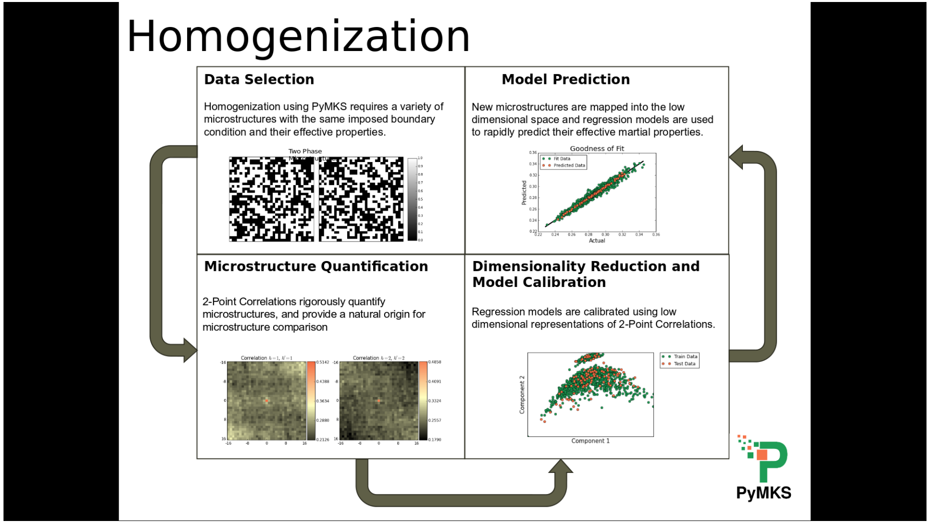
Model Calibration

Influence coefficients capture the physics of the material system and are independent of the microstructure.

Model Scale-up

Because the decaying behavior of the influence coefficients, zero padding can be used to scale up with negligible loss in accuracy.

Figure 4 Homogenization Figure To Do - Place holder



Homogenization

Data Selection

Homogenization using PyMKS requires a variety of microstructures with the same imposed boundary condition and their effective properties.

Model Prediction

New microstructures are mapped into the low dimensional space and regression models are used to rapidly predict their effective martial properties.

Microstructure Quantification

2-Point Correlations rigorously quantify microstructures, and provide a natural origin for microstructure comparison

Dimensionality Reduction and Model Calibration

Regression models are calibrated using low dimensional representations of 2-Point Correlations.

choice for scientific computing due to its ubiquitous use among the data science community as well as many other favorable attributes [80]. PyMKS is licensed under the permissive MIT license [81] which allows for unrestricted distribution in commercial and non-commercial systems.

4.1 Core Functionality

PyMKS consists of four main components including a set of tools to compute 2-point statistics, tools for both homogenization and localization linkages and tools for discretizing the microstructure. In addition, PyMKS has modules for generating data sets using conventional numerical simulations and a module for custom visualization of microstructures. PyMKS is based on the Scikit-learn data pipelining methdology. This is a high level system for combining multiple data and machine learning transformations into a single customizable pipeline with only minimal required code. This approach makes cross-validation and parameter searches simple to implement and avoids the complicated book keeping issues associated with training, testing and validating data pipelines in machine learning.

The starting point for an MKS homogenization analysis is to use 2-point statisics as outlined in Eq. 1 and provided in PyMKS by the `MKSStructureAnalysis` object, which calculates the objective low dimensional structure descriptors, $\mu_j[k]$. The default dimensionality reduction technique is PCA, but any model that uses the `transform_fit` or a "transformer" object can be substituted. After calculating the descriptors, the `MKSHomogenizationModel` is used to create linkages between the $\mu_j[k]$ and the effective material response, p_j^{eff}, as indicated in Eq. 3. The default machine learning algorithm is a polynomial regression, but any estimator with the `fit` and `predict` methods can be substitued to create the linkages between $\mu_j[k]$ and p_j^{eff}.

The `MKSLocalizationModel` object provides the MKS localization functionality. It calibrates the first order influence kernels $\alpha[h;r]$ used to predict local materials responses, $p_j[s]$, as outlined in Eq. 4. The calibration of the influence kernels is achieved using a variety of linear regression techniques described in numerous previous studies [66–68,78]. The `MKSLocalizationModel` object uses `fit` and `predict` methods to follow the standard interface for a Scikit-learn estimator object.

To use either the homogenization or the localization models in PyMKS, the microstructure first needs to be represented by a microstructure function, $m_j\left[h,s\right]$. The `bases` module in PyMKS contains four objects for generating the $m_j\left[h,s\right]$ using a variety of discretization methods [66–72,78]. A `PrimitiveBasis` object uses indicator (or hat) functions and is well suited for microstructures that have discrete local states (e.g., distinct thermodynamic phases). The `LegendreBasis` and `FourierBasis` objects create spectral representations of microstucture functions defined on nonperiodic and periodic continuous local state spaces, respectively. For example, functions over a range of chemical compositions can be described using `LegendreBasis`, while functions over orientations in two-dimensional space can be described using `FourierBasis`. Furthermore, `GSHBasis` creates compact spectral representations for functions over lattice orientation space (such as those needed to describe polycrystalline microstructures) [82–93].

PyMKS contains modest data generation tools (in the `datasets` module) that are used in both the PyMKS examples and the PyMKS test suite. The `MicrostructureGenerator` object creates stochastic microstructures using digital filters. This assists users in creating PyMKS workflows even when data is unavailable. PyMKS has objects for generating sample data from both a spinodal decomposition simulation (using the `CahnHilliardSimulation` object) and a linear elasticity simulation (using the `ElasticFESimulation` object). PyMKS comes with custom functions for visualizing microstructures in elegant ways (in the `tools` module). These are are used extensively in the PyMKS example notebooks to minimize incidental code associated with visualization.

4.2 Underlying Technologies

PyMKS is built upon the highly optimized Python packages NumPy [94], SciPy [95], and Scikit-learn [42]. NumPy arrays are the primary data structure used throughout PyMKS and provide the basic vector and matrix manipulation operations. SciPy's signal processing and numerical linear algebra functions are used to calibrate models and generate synthetic data. PyMKS is highly integrated with Scikit-learn and mimics its simple API in order to leverage from Scikit-learn's data pipeling methodology for machine learning and data transformations. In addition, PyMKS uses the Pytest framework to automate execution of the test suite [96].

Optional packages that can be used with PyMKS include Simple Finite Elements in Python (SfePy) [97], the python wrapper for the FFTW library (pyFFTW) [98] and the plotting package Matplotlib [99]. SfePy is used to simulate linear elasticity to create sample response field data. PyFFTW is a hightly optimized Fast Fourier Transform library that enhances the efficiency of PyMKS and enables parallel computations in PyMKS. Matplotlib is used to generate custom microstructure visualizations.

4.3 Development Practices

PyMKS leverages from existing tools, standards and web resources wherever possible. In particular the developers are an open community that use GitHub for issue tracking and release management (see https://github.com/materialsinnovation/pymks). Additionally a Google group is used as a public forum to discuss the project development, support and announcements (see pymks-general@googlegroups.com). The Travis CI continuous integration tool is used to automate running the test suite for branches of the code stored on GitHub. Code standards are maintained by following the Python PEP8 standards and by reviewing code using pull requests on GitHub. Detailed administrative guidelines are outlined in the `ADMINISTRATA.md` document and potential developers are encouraged to follow them.

5 Examples of Homogenization and Localization with PyMKS

5.1 Prediction of Effective Stiffness using Homogenization

5.1.1 Data Generation

A set of periodic microstructures and their volume averaged elastic stress values $\bar{\sigma}_{xx}$ can be generated by importing the `make_elastic_stress_random` function from `pymks.datasets`. This function has several arguments. `n_samples` is the number of samples that will be generated, `size` specifies the dimensions of the microstructures, `grain_size` controls the effective microstructure feature size, `elastic_modulus` and `poissons_ratio` are used to indicate the material property for each of the phases, `macro_strain` is the value of the applied uniaxial strain, and the `seed` can be used to change the the random number generator seed.

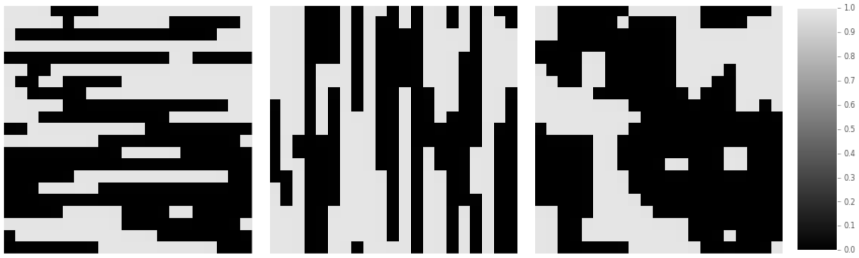
In this homogenization example, 200 samples from six different microstructures classes with dimensions 21 x 21 were created totaling 1,200. Each of the six classes have different microstructure feature sizes. The function will return and the microstructures and their associated volume averaged stress values.

```
import numpy as np
```

```
from pymks.datasets import make_elastic_stress_random

sample_size = 200
grain_size = [(15, 2), (2, 15), (7, 7), (8, 3), (3, 9), (2, 2)]
n_samples = [sample_size] * 6
elastic_modulus = (310, 200)
poissons_ratio = (0.28, 0.3)
macro_strain = 0.001
        size = (21, 21)

X, y = make_elastic_stress_random(
    n_samples=n_samples, size=size, grain_size=grain_size,
    elastic_modulus=elastic_modulus, poissons_ratio=poissons_ratio,
    macro_strain=macro_strain, seed=0)
```

Lets take a look at 3 types the microstructures to get an idea of what they look like. We can do this by importing `draw_microstructures`.

```
from pymks.tools import draw_microstructures

X_examples = X[::sample_size]
draw_microstructures(X_examples[:3])
```



5.1.2 Calibration of Homogenization Model

In order to make an instance of the `MKSHomogenizationModel`, we need to pass an instance of a basis class. For this particular example, there are only 2 discrete phases, so we will use the `PrimitiveBasis` from `pymks.bases`. We only have two phases denoted by 0 and 1, therefore we have two local states and our domain is 0 to 1. Let's make an instance of the `MKSHomgenizationModel`.

```
from pymks import MKSHomogenizationModel
from pymks.bases import PrimitiveBasis

p_basis = PrimitiveBasis(n_states=2, domain=[0, 1])
model = MKSHomogenizationModel(basis=p_basis, periodic_axes=[0, 1],
                               correlations=[(0, 0), (1, 1)])
```

The default machine model used to create the homogenization linkage is polynomial regression. We need to optimize the number of principal components and the degree of polynomial. To do this we are going to split the data into test and training sets. This can be done using the train_test_spilt function from sklearn.

```
from sklearn.cross_validation import train_test_split

flat_shape = (X.shape[0],) + (X[0].size,)
X_train, X_test, y_train, y_test = train_test_split(
    X.reshape(flat_shape), y,test_size=0.2, random_state=3)
print(X_train.shape)
print(X_test.shape)
```

```
(960, 441)
(240, 441)
```

We will use cross validation with the testing data to find the optimal value for our model parameters using GridSeachCV from sklearn. We will pass a dictionary params_to_tune with the range of the degree of the polynomial and the number of principal components we want to search. Let's vary n_components from 1 to 11 and degree from 1 to 3.

```
from sklearn.grid_search import GridSearchCV

params_to_tune = {'degree': np.arange(1,4),
                  'n_components': np.arange(2, 12)}
fit_params = {'size': X[0].shape}
gs = GridSearchCV(model, params_to_tune,
                  fit_params=fit_params).fit(X_train, y_train)
```

The default score method for the MKSHomogenizationModel is R-squared.

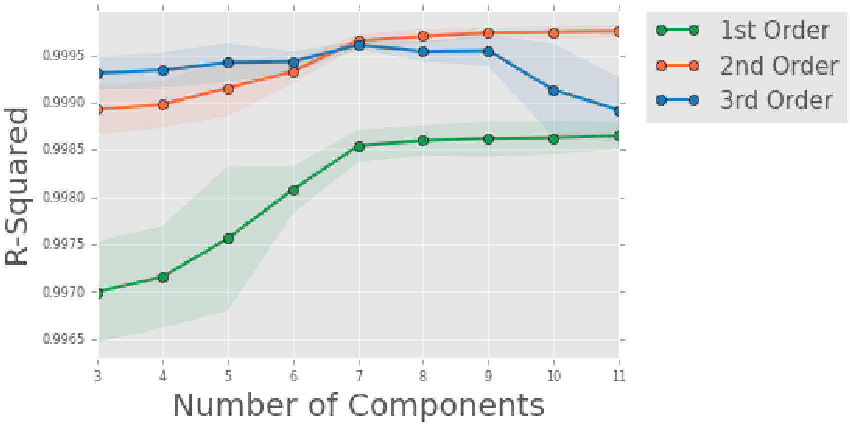
```
print('Order of Polynomial'), (gs.best_estimator_.degree)
print('Number of Components'), (gs.best_estimator_.n_components)
print('R-squared Value'), (gs.score(X_test, y_test))
```

```
Order of Polynomial 2
Number of Components 11
R-squared Value 0.999808062591
```

For the parameter range that we searched, we have found that a model with 2nd order polynomial and 11 components had the best R-squared value. Let's look at the results using draw_grid_scores, and set the model to have those parameter values.

```
from pymks.tools import draw_gridscores
```

```
gs_deg_1 = [x for x in gs.grid_scores_
            if x.parameters['degree'] == 1][1:]
gs_deg_2 = [x for x in gs.grid_scores_
            if x.parameters['degree'] == 2][1:]
gs_deg_3 = [x for x in gs.grid_scores_
            if x.parameters['degree'] == 3][1:]

draw_gridscores([gs_deg_1,  gs_deg_2, gs_deg_3], 'n_components',
                 data_labels=['1st Order',
                              '2nd Order', '3rd Order'],
                 param_label='Number of Components',
                 score_label='R-Squared')
```



```
model = gs.best_estimator_
```

Now that we have found optimal values for the parameters `n_components` and `degree`, lets fit the model with the data.

```
model.fit(X, y)
```

5.1.3 Prediction of Effective Stiffness

Let's generate some more data to validate our model. We are going to generate 20 samples of all six different types of microstructures using the same `make_elastic_stress_random` function.

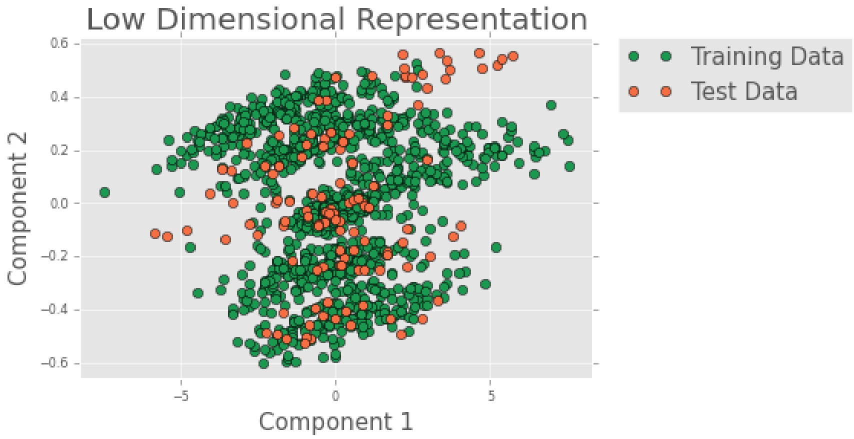
```
test_sample_size = 20
n_samples = [test_sample_size] * 6
X_new, y_new = make_elastic_stress_random(
    n_samples=n_samples, size=size, grain_size=grain_size,
    elastic_modulus=elastic_modulus, poissons_ratio=poissons_ratio,
    macro_strain=macro_strain, seed=1)
```

Now let’s predict the stress values for the new microstructures using the `predict` method.

```
y_predict = model.predict(X_new)
```

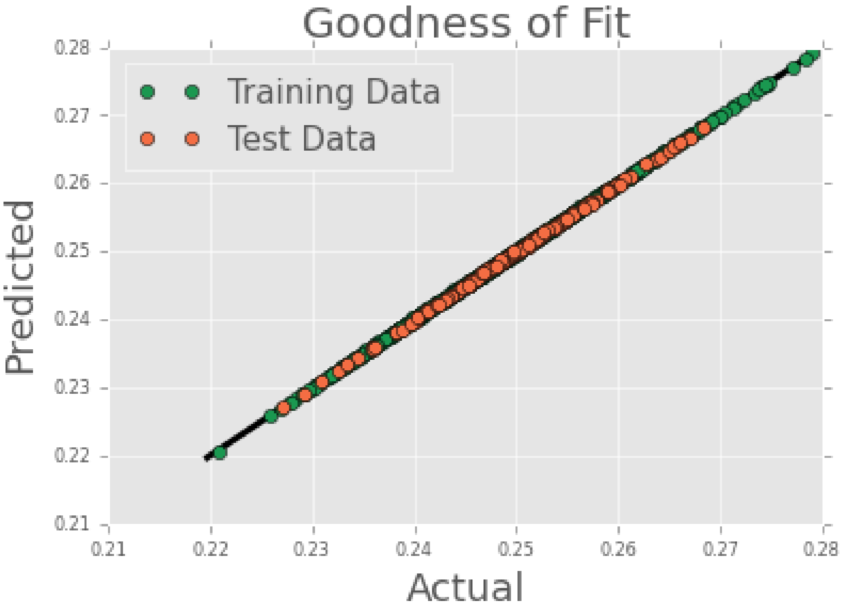
We can look to see, if the low-dimensional representation of the new data is similar to the low-dimensional representation of the data we used to fit the model using `draw_components_scatter` from `pymks.tools`.

```
from pymks.tools import draw_components_scatter

draw_components_scatter([model.reduced_fit_data[:, :2],
                         model.reduced_predict_data[:, :2]],
                         [’Training Data’, ’Test Data’])
```



The predicted data seems to be reasonably similar to the data we used to fit the model with. Now let’s look at the score value for the predicted data. We can evaluate our prediction by looking at a goodness-of-fit plot. We can do this by importing `draw_goodness_of_fit` from `pymks.tools`.

```
from pymks.tools import draw_goodness_of_fit

 fit_data = np.array([y, model.predict(X)])
 pred_data = np.array([y_new, y_predict])
 draw_goodness_of_fit(fit_data, pred_data,
                      [’Training Data’, ’Test Data’])
```



5.2 Prediction of Local Strain Field with Localization

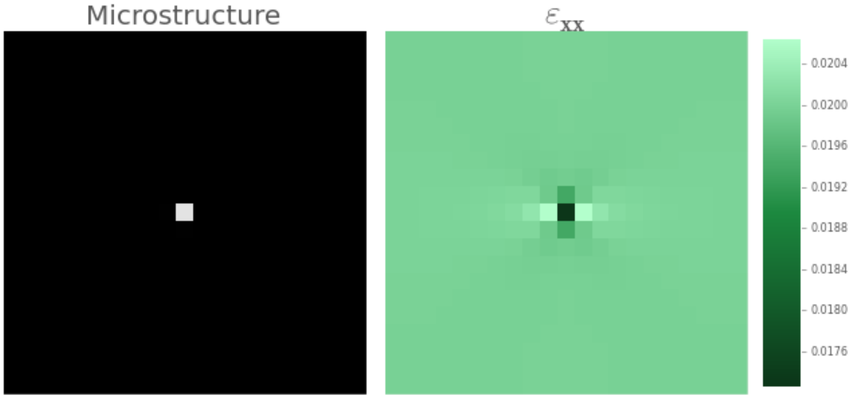
5.2.1 Generating Calibration Data

In this example, lets look at a three phase microstructure with elastic moduli values of 80, 100 and 120 and Poisson's ratio values all equal to 0.3. Let's also set the macroscopic imposed strain equal to 0.02. All of these parameters used in the simulation must be passed into the `make_elasticFEstrain_delta` function from `pymks.datasets`. The number of Poisson's ratio values and elastic moduli values indicates the number of phases.

```
import numpy as np
from pymks.tools import draw_microstructures
from pymks.datasets import make_delta_microstructures

n = 21
n_phases = 3
from pymks.datasets import make_elastic_FE_strain_delta
from pymks.tools import draw_microstructure_strain

elastic_modulus = (80, 100, 120)
poissons_ratio = (0.3, 0.3, 0.3)
macro_strain = 0.02
size = (n, n)

X_delta, strains_delta = make_elastic_FE_strain_delta(
    elastic_modulus=elastic_modulus,
    poissons_ratio=poissons_ratio,
    size=size, macro_strain=macro_strain)
```

Let's take a look at one of the delta microstructures and the ε_{xx} strain field.

```
draw_microstructure_strain(X_delta[0], strains_delta[0])
```



5.2.2 Calibrating Localization Model

Now that we have the delta microstructures and their strain fields, we will calibrate the influence coefficients by creating an instance of the `MKSLocalizatoinModel` class. Because we are going to calibrate the influence coefficients with microstructures that contain discrete local states (in this case phases), we can create an instance of `PrimitiveBasis` with `n_states` equal to 3, and use it to create an instance of `MKSLocalizationModel`.

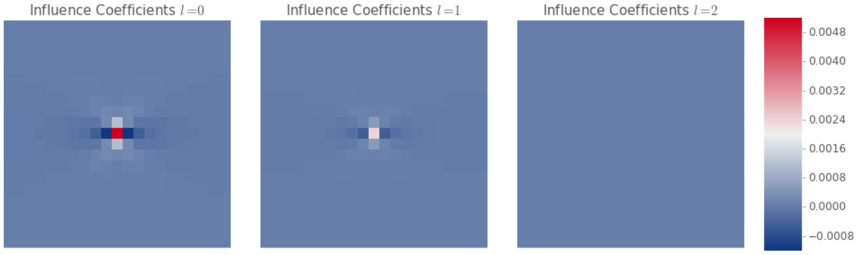
```
from pymks import MKSLocalizationModel
from pymks import PrimitiveBasis

p_basis =PrimitiveBasis(n_states=3, domain=[0, 2])
model = MKSLocalizationModel(basis=p_basis)
```

Now, pass the delta microstructures and their strain fields into the `fit` method to calibrate the first-order influence coefficients.

```
model.fit(X_delta, strains_delta)
```

That's it, the influence coefficient have been calibrated. Let's take a look at them.

```
from pymks.tools import draw_coeff

draw_coeff(model.coef_)
```

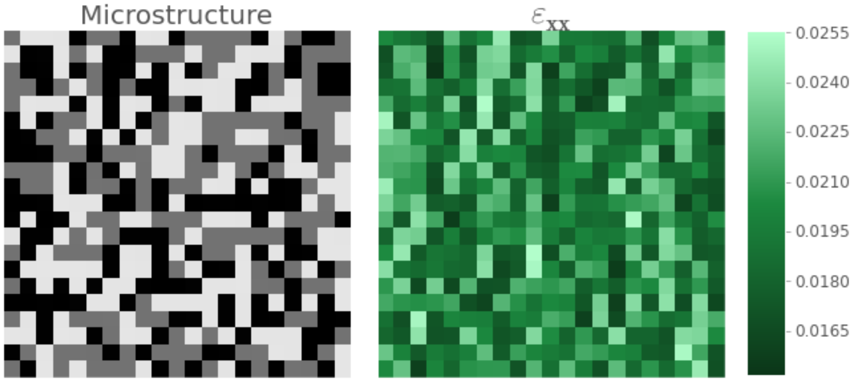


5.2.3 Prediction of the Strain Field for a Random Microstructure

Let's now use our instance of the `MKSLocalizationModel` class with calibrated influence coefficients to compute the strain field for a random two-phase microstructure and compare it with the results from a finite element simulation.

The `make_elasticFEstrain_random` function from `pymks.datasets` is an easy way to generate a random microstructure and its strain field results from finite element analysis.

```
from pymks.datasets import make_elastic_FE_strain_random

np.random.seed(101)
X, strain = make_elastic_FE_strain_random(
    n_samples=1, elastic_modulus=elastic_modulus,
    poissons_ratio=poissons_ratio, size=size,
    macro_strain=macro_strain)
draw_microstructure_strain(X[0] , strain[0])
```



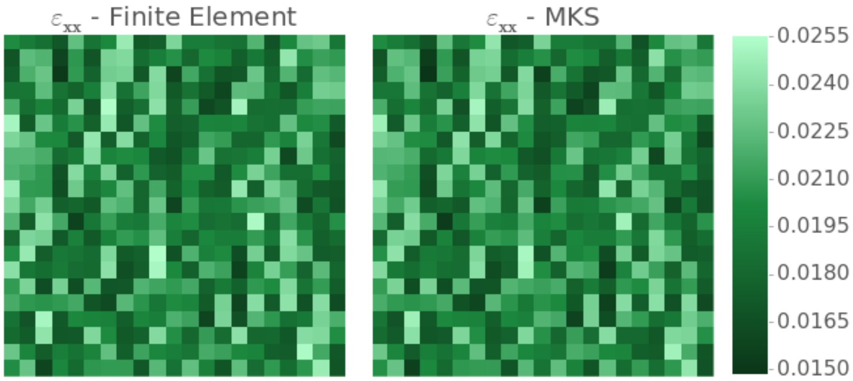
Now, to get the strain field from the `MKSLocalizationModel`, just pass the same microstructure to the `predict` method.

```
strain_pred = model.predict(X)
```

Finally let's compare the results from finite element simulation and the MKS model.

```
from pymks.tools import draw_strains_compare
```

```
draw_strains_compare(strain[0], strain_pred[0])
```



6 Conclusion

To Do

Author details
[1]School of Computational Science and Engineering, Georgia Institute of Technology, 30332, Atlanta, USA. [2]Materials Science and Engineering Division, Material Measurement Laboratory, National Institute of Standards and Technology, 20899, Gaithersburg, USA. [3]George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, 30332, Atlanta, USA.

References

1. Sawhney, M., Verona, G., Prandelli, E.: Collaborating to create: The internet as a platform for customer engagement in product innovation. Journal of interactive marketing **19**(4), 4–17 (2005)
2. Edwards, A.M., Bountra, C., Kerr, D.J., Willson, T.M.: Open access chemical and clinical probes to support drug discovery. Nature Chemical Biology **5**(7), 436–440 (2009)
3. Bayne-Smith, M., Mizrahi, T., Garcia, M.: Interdisciplinary community collaboration: Perspectives of community practitioners on successful strategies. Journal of Community Practice **16**(3), 249–269 (2008)
4. Boudreau, K.: Open platform strategies and innovation: Granting access vs. devolving control. Management Science **56**(10), 1849–1872 (2010)
5. Aad, G., Abajyan, T., Abbott, B., Abdallah, J., Khalek, S.A., Abdelalim, A., Abdinov, O., Aben, R., Abi, B., Abolins, M., *et al.*: Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc. Physics Letters B **716**(1), 1–29 (2012)
6. Lander, E.S., Linton, L.M., Birren, B., Nusbaum, C., Zody, M.C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., *et al.*: Initial sequencing and analysis of the human genome. Nature **409**(6822), 860–921 (2001)
7. Cranshaw, J., Kittur, A.: The polymath project: lessons from a successful online collaboration in mathematics. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1865–1874 (2011). ACM
8. Dickinson, J.L., Zuckerberg, B., Bonter, D.N.: Citizen science as an ecological research tool: challenges and benefits. Annual review of ecology, evolution and systematics **41**, 149–72 (2010)
9. Hochachka, W.M., Fink, D., Hutchinson, R.A., Sheldon, D., Wong, W.-K., Kelling, S.: Data-intensive science applied to broad-scale citizen science. Trends in ecology & evolution **27**(2), 130–137 (2012)
10. Atkins, D.: Revolutionizing science and engineering through cyberinfrastructure: Report of the national science foundation blue-ribbon advisory panel on cyberinfrastructure (2003)
11. Anderson, A.: Report to the president on ensuring american leadership in advanced manufacturing. Executive Office of the President (2011)
12. National Science and Technology Council Executive Office of the President: Materials Genome Initiative for Global Competitiveness. http://www.whitehouse.gov/sites/default/files/microsites/ostp/materials_genome/_initiative-final.pdf Accessed 2011-06-30
13. Materials Genome Initiative National Science and Technology Council Committee on Technology Subcommittee on the Materials Genome Initiative: Materials Genome Initiative Strategic Plan. http://www.whitehouse.gov/sites/default/files/microsites/ostp/NSTC/mgi_strategic_plan_-_dec_2014.pdf Accessed 2014-12-30
14. Kalidindi, S.R.: Data science and cyberinfrastructure: critical enablers for accelerated development of hierarchical materials. International Materials Reviews **60**(3), 150–168 (2015)
15. Ward, C.: Materials genome initiative for global competitiveness. In: 23rd Advanced Aerospace Materials and Processes (AeroMat) Conference and Exposition (2012). Asm
16. Allison, J., Backman, D., Christodoulou, L.: Integrated computational materials engineering: a new paradigm for the global materials profession. JOM **58**(11), 25–27 (2006)
17. Allison, J.: Integrated computational materials engineering: A perspective on progress and future steps. JOM Journal of the Minerals, Metals and Materials Society **63**(4), 15–18 (2011)

18. Olson, G.B.: Designing a new material world. Science **288**(5468), 993–998 (2000)
19. on Integrated Computational Materials Engineering, N.R.C.U.C.: Integrated Computational Materials Engineering: a Transformational Discipline for Improved Competitiveness and National Security. National Academies Press, ??? (2008)
20. Schmitz, G.J., Prahl, U.: Integrative Computational Materials Engineering: Concepts and Applications of a Modular Simulation Platform. John Wiley & Sons, ??? (2012)
21. Robinson, L.: TMS Study charts a course to Successful ICME Implementation. Springer (2013)
22. Allison, J.E.: Integrated computational materials engineering (icme): A transformational discipline for the global materials profession. Metals and Materials, 223
23. A Study Organized by The Minerals, Metals & Materials Society: Integrated Computational Materials Engineering (ICME): Implementing ICME in the Aerospace, Automotive, and Maritime Industries
24. CORE-Materials: CORE-Materials - A resource repository contains a large number of open educational resources (OERs) in Materials Science and Engineering. https://www.flickr.com/people/core-materials/. [Online; accessed 6-April-2016] (2009)
25. Kalidindi, S.R.: Hierarchical Materials Informatics: Novel Analytics for Materials Data. Elsevier, ??? (2015)
26. Bhat, T.N., Bartolo, L.M., Kattner, U.R., Campbell, C.E., Elliott, J.T.: Strategy for extensible, evolving terminology for the materials genome initiative efforts. JOM **67**(8), 1866–1875 (2015)
27. of Standards, N.I., Technology: NIST Data Gateway. http://srdata.nist.gov/gateway/ Accessed 2016-04-01
28. Laboratory, N.M.M.: NIST Repositories DSpace. https://materialsdata.nist.gov/dspace/xmlui/ Accessed 2016-04-01
29. of Standards, N.I., Technology: NIST Data Curation System. https://mgi.nist.gov/materials-data-curation-system Accessed 2016-04-01
30. Saal, J.E., Kirklin, S., Aykol, M., Meredig, B., Wolverton, C.: Materials design and discovery with high-throughput density functional theory: the open quantum materials database (oqmd). Jom **65**(11), 1501–1509 (2013)
31. MatWeb, L.: MatWeb - Materials Property Data. http://www.matweb.com/ Accessed 2016-04-01
32. Curtarolo, S., Setyawan, W., Hart, G.L., Jahnatek, M., Chepulskii, R.V., Taylor, R.H., Wang, S., Xue, J., Yang, K., Levy, O., *et al.*: Aflow: an automatic framework for high-throughput materials discovery. Computational Materials Science **58**, 218–226 (2012)
33. Ong, S.P., Richards, W.D., Jain, A., Hautier, G., Kocher, M., Cholia, S., Gunter, D., Chevrier, V.L., Persson, K.A., Ceder, G.: Python materials genomics (pymatgen): A robust, open-source python library for materials analysis. Computational Materials Science **68**, 314–319 (2013)
34. Jain, A., Ong, S.P., Hautier, G., Chen, W., Richards, W.D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G., *et al.*: Commentary: The materials project: A materials genome approach to accelerating materials innovation. Apl Materials **1**(1), 011002 (2013)
35. Project, K.: OpenKIM - The Knowledgebase of Interatomic Models. https://openkim.org/ Accessed 2016-04-01
36. Project, P.: PRedictive Integrated Structural Materials Science (PRISMS). http://www.prisms-center.org/#/home Accessed 2016-04-01
37. Plimpton, S., Thompson, A., Slepoy, A.: SPPARKS kinetic Monte Carlo simulator (2012)
38. Gaston, D., Newman, C., Hansen, G., Lebrun-Grandie, D.: Moose: A parallel computational framework for coupled systems of nonlinear equations. Nuclear Engineering and Design **239**(10), 1768–1778 (2009)
39. Groeber, M.A., Jackson, M.A.: Dream. 3d: a digital representation environment for the analysis of microstructure in 3d. Integrating Materials and Manufacturing Innovation **3**(1), 1–17 (2014)
40. Littell, R.C.: Sas. Wiley Online Library, ??? (2006)
41. Seabold, S., Perktold, J.: Statsmodels: Econometric and statistical modeling with python. In: Proceedings of the 9th Python in Science Conference, pp. 57–61 (2010)
42. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.*: Scikit-learn: Machine learning in python. The Journal of Machine Learning Research **12**, 2825–2830 (2011)
43. Albanese, D., Visintainer, R., Merler, S., Riccadonna, S., Jurman, G., Furlanello, C.: mlpy: Machine learning python. arXiv preprint arXiv:1202.6548 (2012)
44. Goodfellow, I.J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., Bergstra, J., Bastien, F., Bengio, Y.: Pylearn2: a machine learning research library. arXiv preprint arXiv:1308.4214 (2013)
45. McKinney, W.: Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. " O'Reilly Media, Inc.", ??? (2012)
46. Müller, A.C., Behnke, S.: Pystruct: learning structured prediction in python. The Journal of Machine Learning Research **15**(1), 2055–2060 (2014)
47. Demšar, J., Zupan, B., Leban, G., Curk, T.: Orange: From Experimental Machine Learning to Interactive Data Mining. Springer, ??? (2004)
48. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
49. Van Der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T.: scikit-image: image processing in python. PeerJ **2**, 453 (2014)
50. Hill, R.: Elastic properties of reinforced solids: some theoretical principles. Journal of the Mechanics and Physics of Solids **11**(5), 357–372 (1963)
51. Hashin, Z.: Analysis of composite materials—a survey. Journal of Applied Mechanics **50**(3), 481–505 (1983)
52. Brown Jr, W.F.: Solid mixture permittivities. The Journal of Chemical Physics **23**(8), 1514–1517 (1955)
53. Kröner, E.: Statistical modelling. In: Modelling Small Deformations of Polycrystals, pp. 229–291. Springer, ??? (1986)

54. Kröner, E.: Bounds for effective elastic moduli of disordered materials. Journal of the Mechanics and Physics of Solids **25**(2), 137–155 (1977)
55. Kröner, E.: Statistical Continuum Mechanics. Springer, ??? (1972)
56. Etingof, P., Adams, B.L.: Representations of polycrystalline microstructure by n-point correlation tensors. Texture, Stress, and Microstructure **21**(1), 17–37 (1993)
57. Adams, B.L., Olson, T.: The mesostructure—properties linkage in polycrystals. Progress in Materials Science **43**(1), 1–87 (1998)
58. Fullwood, D.T., Adams, B.L., Kalidindi, S.R.: A strong contrast homogenization formulation for multi-phase anisotropic materials. Journal of the Mechanics and Physics of Solids **56**(6), 2287–2297 (2008)
59. Torquato, S.: Random Heterogeneous Materials: Microstructure and Macroscopic Properties vol. 16. Springer, ??? (2013)
60. Li, D., Saheli, G., Khaleel, M., Garmestani, H.: Quantitative prediction of effective conductivity in anisotropic heterogeneous media using two-point correlation functions. Computational Materials Science **38**(1), 45–50 (2006)
61. Milhans, J., Li, D., Khaleel, M., Sun, X., Garmestani, H.: Prediction of the effective coefficient of thermal expansion of heterogeneous media using two-point correlation functions. Journal of Power Sources **196**(8), 3846–3850 (2011)
62. Adams, B.L., Kalidindi, S., Fullwood, D.T.: Microstructure-sensitive Design for Performance Optimization. Butterworth-Heinemann, ??? (2013)
63. Garmestani, H., Lin, S., Adams, B., Ahzi, S.: Statistical continuum theory for large plastic deformation of polycrystalline materials. Journal of the Mechanics and Physics of Solids **49**(3), 589–607 (2001)
64. Adams, B.L., Gao, X.C., Kalidindi, S.R.: Finite approximations to the second-order properties closure in single phase polycrystals. Acta Materialia **53**(13), 3563–3577 (2005)
65. Binci, M., Fullwood, D., Kalidindi, S.R.: A new spectral framework for establishing localization relationships for elastic behavior of composites and their calibration to finite-element models. Acta Materialia **56**(10), 2272–2282 (2008)
66. Landi, G., Niezgoda, S.R., Kalidindi, S.R.: Multi-scale modeling of elastic response of three-dimensional voxel-based microstructure datasets using novel dft-based knowledge systems. Acta Materialia **58**(7), 2716–2725 (2010)
67. Kalidindi, S.R., Niezgoda, S.R., Landi, G., Vachhani, S., Fast, T.: A novel framework for building materials knowledge systems. Computers, Materials, & Continua **17**(2), 103–125 (2010)
68. Yabansu, Y.C., Patel, D.K., Kalidindi, S.R.: Calibrated localization relationships for elastic response of polycrystalline aggregates. Acta Materialia **81**, 151–160 (2014)
69. Al-Harbi, H.F., Landi, G., Kalidindi, S.R.: Multi-scale modeling of the elastic response of a structural component made from a composite material using the materials knowledge system. Modelling and Simulation in Materials Science and Engineering **20**(5), 055001 (2012)
70. Kalidindi, S.R., Niezgoda, S.R., Salem, A.A.: Microstructure informatics using higher-order statistics and efficient data-mining protocols. Jom **63**(4), 34–41 (2011)
71. Gupta, A., Cecen, A., Goyal, S., Singh, A.K., Kalidindi, S.R.: Structure–property linkages using a data science approach: Application to a non-metallic inclusion/steel composite system. Acta Materialia **91**, 239–254 (2015)
72. Çeçen, A., Fast, T., Kumbur, E., Kalidindi, S.: A data-driven approach to establishing microstructure–property relationships in porous transport layers of polymer electrolyte fuel cells. Journal of Power Sources **245**, 144–153 (2014)
73. Niezgoda, S.R., Kanjarla, A.K., Kalidindi, S.R.: Novel microstructure quantification framework for databasing, visualization, and analysis of microstructure data. Integrating Materials and Manufacturing Innovation **2**(1), 1–27 (2013)
74. Niezgoda, S.R., Yabansu, Y.C., Kalidindi, S.R.: Understanding and visualizing microstructure and microstructure variance as a stochastic process. Acta Materialia **59**(16), 6387–6400 (2011)
75. Qidwai, S.M., Turner, D.M., Niezgoda, S.R., Lewis, A.C., Geltmacher, A.B., Rowenhorst, D.J., Kalidindi, S.R.: Estimating the response of polycrystalline materials using sets of weighted statistical volume elements. Acta Materialia **60**(13), 5284–5299 (2012)
76. Niezgoda, S.R., Turner, D.M., Fullwood, D.T., Kalidindi, S.R.: Optimized structure based representative volume element sets reflecting the ensemble-averaged 2-point statistics. Acta Materialia **58**(13), 4432–4445 (2010)
77. Yabansu, Y.C., Kalidindi, S.R.: Representation and calibration of elastic localization kernels for a broad class of cubic polycrystals. Acta Materialia **94**, 26–35 (2015)
78. Brough, D.B., Wheeler, D., Warren, J.A., Kalidindi, S.R.: Microstructure-based knowledge systems for capturing process-structure evolution linkages. Current Opinion in Solid State and Materials Science (2016)
79. Cecen, A., Fast, T., Kalidindi, S.R.: Versatile algorithms for the computation of 2-point spatial correlations in quantifying material structure. Integrating Materials and Manufacturing Innovation **5**(1), 1–15 (2016)
80. Pérez, F., Granger, B.E., Hunter, J.D.: Python: An ecosystem for scientific computing. Computing in Science & Engineering **13**(2), 13–21 (2011). doi:10.1109/MCSE.2010.119
81. The MIT License (MIT). `https://opensource.org/licenses/mit-license.php`. Accessed: 2016-05-18
82. Kalidindi, S.R., Duvvuru, H.K., Knezevic, M.: Spectral calibration of crystal plasticity models. Acta Materialia **54**(7), 1795–1804 (2006)
83. Shaffer, J.B., Knezevic, M., Kalidindi, S.R.: Building texture evolution networks for deformation processing of polycrystalline fcc metals using spectral approaches: applications to process design for targeted performance. International Journal of Plasticity **26**(8), 1183–1194 (2010)
84. Knezevic, M., Levinson, A., Harris, R., Mishra, R.K., Doherty, R.D., Kalidindi, S.R.: Deformation twinning in az31: influence on strain hardening and texture evolution. Acta Materialia **58**(19), 6230–6242 (2010)
85. Al-Harbi, H.F., Knezevic, M., Kalidindi, S.R.: Spectral approaches for the fast computation of yield surfaces and first-order plastic property closures for polycrystalline materials with cubic-triclinic textures. Computers, Materials, & Continua **15**(2), 153–172 (2010)

86. Duvvuru, H.K., Knezevic, M., Mishra, R.K., Kalidindi, S.R.: Application of microstructure sensitive design to fcc polycrystals. In: Materials Science Forum, vol. 546, pp. 675–680 (2007). Trans Tech Publ
87. Li, D., Garmestani, H., Schoenfeld, S.: Evolution of crystal orientation distribution coefficients during plastic deformation. Scripta Materialia **49**(9), 867–872 (2003)
88. Li, D., Garmestani, H., Adams, B.: A texture evolution model in cubic-orthotropic polycrystalline system. International journal of plasticity **21**(8), 1591–1617 (2005)
89. Li, D., Garmestani, H., Ahzi, S.: Processing path optimization to achieve desired texture in polycrystalline materials. Acta materialia **55**(2), 647–654 (2007)
90. Li, D.S., Bouhattate, J., Garmestani, H.: Processing path model to describe texture evolution during mechanical processing. In: Materials Science Forum, vol. 495, pp. 977–982 (2005). Trans Tech Publ
91. Creuziger, A., Hu, L., Gnäupel-Herold, T., Rollett, A.D.: Crystallographic texture evolution in 1008 steel sheet during multi-axial tensile strain paths. Integrating Materials and Manufacturing Innovation **3**(1), 1 (2014)
92. Sundararaghavan, V., Zabaras, N.: A multi-length scale sensitivity analysis for the control of texture-dependent properties in deformation processing. International Journal of Plasticity **24**(9), 1581–1605 (2008)
93. Sundararaghavan, V., Zabaras, N.: Linear analysis of texture–property relationships using process-based representations of rodrigues space. Acta materialia **55**(5), 1573–1587 (2007)
94. Van Der Walt, S., Colbert, S.C., Varoquaux, G.: The numpy array: a structure for efficient numerical computation. Computing in Science & Engineering **13**(2), 22–30 (2011)
95. Jones, E., Oliphant, T., Peterson, P.: {SciPy}: open source scientific tools for {Python} (2014)
96. Pytest. http://pytest.org (2016)
97. Cimrman, R.: Sfepy-write your own fe application. arXiv preprint arXiv:1404.6391 (2014)
98. Frigo, M., Johnson, S.G.: Fftw: An adaptive software architecture for the fft. In: Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference On, vol. 3, pp. 1381–1384 (1998). IEEE
99. Hunter, J.D., *et al.*: Matplotlib: A 2d graphics environment. Computing in science and engineering **9**(3), 90–95 (2007)