Binary 2D Morphologies of Polymer Phase Separation: Dataset and Python Toolbox

Viraj Shah*, Ameya Joshi, Balaji Sesha Sarath Pokuri, Sambuddha Ghosal,
Soumik Sarkar, Baskar Ganapathysubramanian, Chinmay Hegde

Iowa State University

March 2019

1 Introduction

Study of the intricate connection between the design of material distributions (also called morphology or microstructure) and the final properties of the material system has been an attractive research theme for material science community. Such analysis provides ability to synthesize the microstructures exhibiting desired properties. This theme encompasses several material systems including porous materials [26], steels and welds [2], composites [14], powder metallurgy [28], 3D printing [22], energy storage devices as batteries [10], and energy converting devices like bulk hetero-junction solar cells [20]. Microstructure-sensitive design has been used to tailor a wide variety of properties including strengths, heat and mass diffusivities, energy storage capacity and lifetime, and energy conversion efficiency.

Ability to acquire or generate the miscrostructure is a key requirement in establishing the connection between microstructures and the properties. There are extensive efforts to experimentally image the microstructure (including X-ray, optical, and electron microscopy). However, getting a complete virtual instance of a microstructure is non-trivial, as is the process of perturbing specific features of the microstructure. Thus, an entire sub-field in computational material science is devoted to the development of methods for the *simulation* of microstructures [8, 9, 21]. Over the past several decades, a number of methods have been developed for microstructure simulations, including Gaussian random fields [21], optimization-based methods [31], multi-point statistics [5], and layer-by-layer reconstruction [25]. Consequently, these approaches are computationally intensive and prohibitively slow, requiring several compute hours even to simulate just *one* synthetic example.

In contrast with these computational approaches, in our recent work [24], we proposed data-driven methods employing physics-aware generative models as the alternative. Availability of large dataset of microstructures play a significant role in the success of data-driven generative models such as Wasserstein Generative Adversarial Network in [24]. Though the simulation methods to synthesize microstructures are widely known in the literature, the large dataset of microstructures generated using such simulations are not readily available to be used to train data-driven models. Generating such large datasets from scratch requires dedicated time and computational resources and if not standardized, fails to address the need of larger research community. To overcome this shortcoming and to accelerate the research in the field of data-driven microstructure generation, in this work we curate a large dataset of 2-phase microstructures by solving Cahn-Hilliard equation, and make it publicly available [18]. We call our dataset the **CH dataset**. Moreover, we develop an **open-source**

*contact:viraj@iastate.edu

Funding Info: This work is supported by a grant from the DARPA AIRA program.

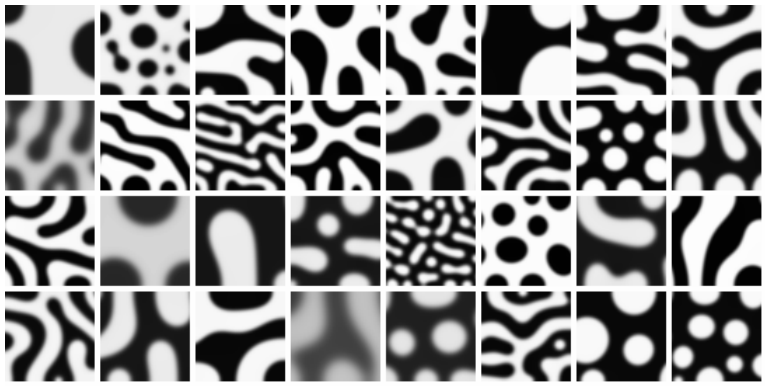

Figure 1: (a) Sample images from Cahn-Hilliard dataset

python toolbox with our dataset, that provides several tools related to setup, usage and analysis functionality for a smoother interface. In this document, we introduce our dataset, the method of generation, its format, and functionality of our python toolbox.

2 The Cahn-Hilliard Dataset

Our dataset is a collection of 34672 microstructure images generated by solving the Cahn-Hilliard equation. Since we solve the CH equation on a 101×101 grid in pixel domain, every image is 101×101 pixels in resolution. Each image is grayscale, with the value of each pixel ranging between 0 to 1. However, due to the nature of the solution, these values lie very close to either 0 or 1, producing a near-binary image.

For efficient data storage, we use HDF5 format to store our data. HDF stands for Hierarchical Data Format (HDF) and is a filesystem-like format designed to store and organize large amounts of data [1]. HDF is supported by many commercial and non-commercial software platforms, including Java, C/C++, MATLAB, Scilab, Octave, Mathematica, IDL, Python, R, Fortran, and Julia. Our dataset is a single HDF5 file containing a folder named `morphologies`. This folder stores the 34672×10201 matrix, where each row is a flattened microstructure image.

We host our dataset on Zenodo, a general-purpose open-access repository that allows researchers to deposit data sets, research software, reports, and any other research related digital artifacts [32]. For each submission, a persistent Digital object identifier (DOI) is minted, which makes the stored items easily citeable. It is hosted on high-performance computing infrastructure and provides longevity to the dataset. Our dataset is released under the Creative Commons licence [4] allowing easy public distribution. Our dataset and supporting metadata can be downloaded manually from the Zenodo website, or can be downloaded automatically using a Python toolbox that we provide on Github. Section 3 covers details about the usage and functionality of our toolbox.

2.1 Dataset Generation

We generate the CH dataset of microstructural images using a thermodynamic consistent binary phase separation simulation. This is done by solving the well known Cahn-Hilliard equation [3]. Originally proposed to study phase separation in alloys, this equation can account for such phenomena in polymers, ceramics, and other material systems.

The Cahn-Hilliard equation tracks the local volume fraction of each material (ϕ_i):

$$\begin{aligned} \frac{\partial \phi_i}{\partial t} &= \nabla \left(M\left(\phi_i\right) \nabla \mu_i \right) \\ \mu_i &= \frac{\partial f}{\partial \phi_i} - \epsilon^2 \nabla^2 \phi_i \end{aligned} \tag{1}$$

where $M(\phi_i)$ is the mobility of component i and μ_i represents the chemical potential of component i. The chemical potential as defined in Eq.1 is the variational derivative of the total free energy of the system. The total free energy comprises of the bulk free energy f and the interfacial energy. The interfacial free energy is characterized as $0.5\epsilon^2|\nabla\phi_i|^2$, where ϵ is the interfacial energy parameter. ϵ is usually correlated with the thickness of the interface between the components. The bulk free energy is described using the Flory-Huggins ([6]) energy representation:

$$f = \frac{\phi_1}{N_1} \log \phi_1 + \frac{\phi_2}{N_2} \log \phi_2 + \chi_{12} \phi_1 \phi_2 \tag{2}$$

The degree of polymerization of the components is denoted by N_i while χ_{ij} represents the severity of interaction between the components. The values for χ are either estimated using molecular simulations ([7, 13]), or experimentally ([16]), or calculated through empirical methods ([11]).

We use an in-house modular finite element software [29] to solve this equation 1 to generate time evolving microstructures. This process generates time series of images that can be treated as independent images for the sake of training. In order to generate numerous consistent morphologies, we perform 100 simulations of the above equation (10 values of χ_{12} with 10 values of initial concentration), with morphologies outputted at every 20 timesteps (which provides distinguishable morphologies across timesteps). Independent solutions were obtained for a range of system parameters like volume fraction (ratio of black pixels to white pixels) and immiscibility parameters (that determine the degree of purity of domains). Previous analysis using these images can be found in [30]. A characteristic of this procedure for generating morphologies through simulation is their similarity to morphologies in real active layers produced during thermal annealing. In the morphology, the domains are similar in size and have smooth interface contours.

3 Python Toolbox

To aid in the usage and analysis of our dataset, we provide a set of utility functions in the form of a python toolbox [23] publicly available on Github page under MIT licence. This section serves as the help and reference guide to our toolbox. We also provide a Python notebook [1] depicting the usage of all of the following functionalities through examples.

3.1 Setup

Our toolbox is written in Python 3.5 and is compatible with any version of Python>=3.5. Some key python libraries required for our toolbox are: `h5py` [1], `numpy` [15], `scipy` [15], `matplotlib` [12], `Pillow` [17], `urllib` [27]. An extensive list of the requirements are enclosed with the toolbox. We provide an installation script `setup.py` to install all the required packages. Once the setup is complete, the data can be downloaded automatically using the `download` method as explained below. We also provide an exemplar Jupyter notebook [19] to demonstrate the usage of our toolbox.

[1] `https://github.com/shahviraj/2d-morphologies-data-utils/blob/master/Example.ipynb`

3.2 `Class:AIRADataset`

This class is a wrapper around the h5py handler functions. Class methods include the basic functionality required to load the dataset and perform basic data handling operations. We expose a random access scheme for the data allowing for indexed access as well as slicing (similar to `numpy` arrays). In addition, we include a resizer to automatically resize the images to user specifications. We describe the API of our toolbox in the following subsection.

3.2.1 API

`AIRADataset.__init__`: The constructor exposes the following functionalities;

- Selection of the dataset *(huge/tiny)*.
- Automatic download of the selected dataset.
- Initialization of user-specified resizing of images.

`AIRADataset.download_dataset`: Function to download the dataset automatically from the dataset URL.

`AIRADataset.get_dataset`: Function to return the raw dataset without any preprocessing. This returns a `h5py` array.

`AIRADataset.generate_thumbnails`: A utility to visualize a random selection of data. It selects 16 microstructures randomly from the dataset, and returns a tiled array of these 16 images stacked into 4×4 grid – which can be used to plot the images for visualization. Alternatively, `show_random_images` method can also be used for visualization.

`AIRADataset.show_random_images`:

A wrapper for the `AIRADataset.generate_thumbnails`. Primarily used for visualization in `Jupyter notebooks`.

3.3 Utility functions for data analysis

Calculating the statistical properties of the microstructures, and partitioning the dataset into different subsets based on the values of such statistical properties are often encountered tasks in data-driven material discovery. We provide utility functions to aid in such tasks for two most common statistical properties: one-point correlation or volume fraction (p_1), and two-point correlation (p_2). The detailed discussion about the formulae used for the calculations are available in [24] or in appendix(section 4).

3.3.1 `calculate_p1`

p_1 refers to volume fraction of the microstructure, calculated as the mean of all the pixel values of the microstructure. This method calculates the p_1 values and return the p1 distribution over the dataset or a segment of it. The function returns a histogram plot for better visualization of the output.

3.3.2 `calculate_p2`

Returns a two-point correlation vector for the input microstructure image. We calculate the p2 vector by taking the radial mean of the autocorrelation of the input image. Generated p_2 vector can be plotted using any standard plotting library as shown in the example python notebook.

3.3.3 `get_segmented_ds`

This function is to segment the dataset based on the p_1 value. The desired range of p_1 value is given as the input to get a `numpy` array containing all the images with p_1 value in the given range.

4 Appendix

We consider the underlying material to be a two-phase homogeneous, isotropic material. Our setup for statistical characterization of microstructure follows with Torquato [26]. A phase function $\phi(\cdot)$ is used to characterize this two-phase system, defined as:

$$\phi^{(1)}(\mathbf{r}) = \begin{cases} 1, \mathbf{r} \in V_1, \\ 0, \mathbf{r} \in V_2, \end{cases} \tag{3}$$

where $V_1$is the region occupied by phase 1 and $V_2$is the region occupied by phase 2.

Given this microstructure defined by the phase function, ϕ, we can calculate the n-point correlation functions for $n = 1, 2, 3, ...$ For homogeneous and isotropic media, the 1-point correlation function, p_1, commonly known as *volume fraction* for phase 1, $p_1^{(1)}$, is defined as:

$$p_1^{(1)} = \mathbb{E}_{\mathbf{r}} \phi^{(1)}(\mathbf{r}).$$

Also, the 2−point correlation is defined as:

$$p_2^{(1)}(r_{12}) = \mathbb{E}_{\mathbf{r_1},\mathbf{r_2}} \left[\phi^{(1)}(\mathbf{r}_1) \phi^{(1)}(\mathbf{r}_2) \right].$$

The 2−point correlation is one of the most important statistical descriptors of microstructures. An alternate interpretation of 2-point correlation is the probability that two randomly chosen points $\mathbf{r}_1$ and $\mathbf{r}_2$ a certain distance apart both share the same phase.

References

[1] In: *HDF5 Technologies* (2019). URL: https://support.hdfgroup.org/HDF5/whatishdf5.html.

[2] H. K. D. H. (Harshad Kumar Dharamshi Hansraj) Bhadeshia and R. W. K. (Robert William Kerr) Honeycombe. *Steels : microstructure and properties.* Butterworth-Heinemann, 2017, p. 461. ISBN: 9780081002704.

[3] John W Cahn. "On spinodal decomposition". In: *Acta metallurgica* 9.9 (1961), pp. 795–801.

[4] *Creative Commons Legal Code*. In: *Creative Commons* (). URL: https://creativecommons.org/licenses/by/4.0/.

[5] Junxi Feng et al. "Accelerating multi-point statistics reconstruction method for porous media via deep learning". In: *Acta Materialia* 159 (Oct. 2018), pp. 296–308. ISSN: 13596454. DOI: 10.1016/j.actamat.2018.08.026. URL: https://www.sciencedirect.com/science/article/pii/S1359645418306633?dgcid=rss%7B%5C_%7Dsd%7B%5C_%7Dall.

[6] Paul J. Flory. "Thermodynamics of High Polymer Solutions". In: *The Journal of Chemical Physics* 10.1 (1942), pp. 51–61. DOI: http://dx.doi.org/10.1063/1.1723621. URL: http://scitation.aip.org/content/aip/journal/jcp/10/1/10.1063/1.1723621.

[7] Yao-Tsung Fu, Chad Risko, and Jean-Luc Bredas. "Intermixing at the Pentacene-Fullerene Bilayer Interface: A Molecular Dynamics Study". In: *Advanced Materials* 25.6 (2013), pp. 878–882. ISSN: 1521-4095. DOI: 10.1002/adma.201203412. URL: http://dx.doi.org/10.1002/adma.201203412.

[8] Baskar Ganapathysubramanian and Nicholas Zabaras. "A non-linear dimension reduction methodology for generating data-driven stochastic input models". In: *Journal of Computational Physics* 227.13 (June 2008), pp. 6612–6637. ISSN: 0021-9991. DOI: 10.1016/J.JCP.2008.03.023. URL: https://www.sciencedirect.com/science/article/pii/S0021999108001721.

[9] Baskar Ganapathysubramanian and Nicholas Zabaras. "Modeling diffusion in random heterogeneous media: Data-driven models, stochastic collocation and the variational multiscale method". In: *Journal of Computational Physics* 226.1 (Sept. 2007), pp. 326–353. ISSN: 0021-9991. DOI: 10.1016/J.JCP.2007.04.009. URL: https://www.sciencedirect.com/science/article/pii/S002199910700160X.

[10] Ramiro García-García and R. Edwin García. "Microstructural effects on the average properties in porous battery electrodes". In: *Journal of Power Sources* 309 (Mar. 2016), pp. 11–19. ISSN: 0378-7753. DOI: 10.1016/J.JPOWSOUR.2015.11.058. URL: https://www.sciencedirect.com/science/article/pii/S0378775315305577.

[11] Charles M. Hansen. *Hansen Solubility Parameters, A User's Handbook, Second Edition*. CRC Press, 2007.

[12] *Installation*. In: *Matplotlib: Python plotting - Matplotlib 2.2.3 documentation* (2019). URL: https://matplotlib.org/.

[13] Chong Meng Kok and Alfred Rudin. "Prediction of Flory–Huggins interaction parameters from intrinsic viscosities". In: *Journal of Applied Polymer Science* 27.2 (1982), pp. 353–362. ISSN: 1097-4628. DOI: 10.1002/app.1982.070270203. URL: http://dx.doi.org/10.1002/app.1982.070270203.

[14] James Chen-Min Li. *Microstructure and Properties of Materials, Volume 2*. WORLD SCIENTIFIC, Oct. 2000, p. 436. ISBN: 9810241801. DOI: 10.1117/12.874247. URL: https://books.google.com/books?id=4c31Fipmc5AC%7B%5C&%7Dpgis=1.

[15] *NumPy*. In: *NumPy* (2019). URL: https://www.numpy.org/.

[16] R. A. Orwoll. "The Polymer-Solvent Interaction Parameter X". In: *Rubber Chemistry and Technology* 50.3 (1977), pp. 451–479. DOI: 10.5254/1.3535155. eprint: http://dx.doi.org/10.5254/1.3535155. URL: http://dx.doi.org/10.5254/1.3535155.

[17] *Pillow*. In: *Pillow - Pillow (PIL Fork) 3.0.0 documentation* (2019). URL: https://pillow.readthedocs.io/.

[18] Balaji Sesha Sarath Pokuri et al. *Binary 2D morphologies of polymer phase separation Binary 2D morphologies of polymer phase separation*. In: (Feb. 2019). DOI: 10.5281/zenodo.2580293. URL: https://doi.org/10.5281/zenodo.2580293.

[19] *Project Jupyter*. In: *Project Jupyter* (2019). URL: https://jupyter.org/.

[20] Jonathan Rivnay et al. "Quantitative Determination of Organic Semiconductor Microstructure from the Molecular to Device Scale". In: *Chemical reviews* (2012). DOI: 10.1021/cr3001109. URL: https://pubs.acs.org/sharingguidelines.

[21] Anthony P Roberts. "Statistical reconstruction of three-dimensional porous media from two-dimensional images". In: *Physical Review E* 56.3 (1997), p. 3203.

[22] Christian Schumacher et al. "Microstructures to control elasticity in 3D printing". In: *ACM Transactions on Graphics (TOG)* 34.4 (2015), p. 136. URL: https://core.ac.uk/download/pdf/77926908.pdf.

[23] Viraj Shah and Ameya Joshi. *2d morphologies data utils*. In: *GitHub* (Mar. 2019). URL: https://github.com/shahviraj/2d-morphologies-data-utils.

[24] Rahul Singh et al. "Physics-aware Deep Generative Models for Creating Synthetic Microstructures". In: *arXiv preprint arXiv:1811.09669* (2018).

[25] Pejman Tahmasebi, Ardeshir Hezarkhani, and Muhammad Sahimi. "Multiple-point geostatistical modeling based on the cross-correlation functions". In: *Computational Geosciences* 16.3 (2012), pp. 779–797.

[26] Salvatore Torquato. *Random heterogeneous materials: microstructure and macroscopic properties*. Vol. 16. Springer Science & Business Media, 2013.

[27] *urllib - Open arbitrary resources by URL.* In: *threading - Higher-level threading interface - Python 2.7.15 documentation* (2019). URL: https://docs.python.org/2/library/urllib.html.

[28] Hongtao Wang, Z Zak Fang, and Pei Sun. "A critical review of mechanical properties of powder metallurgy titanium". In: *International Journal of Powder Metallurgy (Princeton, New Jersey)* 46.5 (2010), pp. 45–57.

[29] Olga Wodo and Baskar Ganapathysubramanian. "Modeling morphology evolution during solvent-based fabrication of organic solar cells". In: *Computational Materials Science* 55 (Apr. 2012), pp. 113–126. ISSN: 0927-0256. DOI: 10.1016/J.COMMATSCI.2011.12.012. URL: https://www.sciencedirect.com/science/article/pii/S0927025611006732.

[30] O. Wodo et al. "Automated, high throughput exploration of process–structure–property relationships using the MapReduce paradigm". In: *Materials discovery* 1 (2015), pp. 21–28.

[31] C. L. Y. Yeong and S. Torquato. "Reconstructing random media". In: *Phys. Rev. E* 57 (1 Jan. 1998), pp. 495–506. DOI: 10.1103/PhysRevE.57.495. URL: https://link.aps.org/doi/10.1103/PhysRevE.57.495.

[32] *Zenodo,Research. Shared.* In: *Zenodo* (2019). URL: https://zenodo.org/.