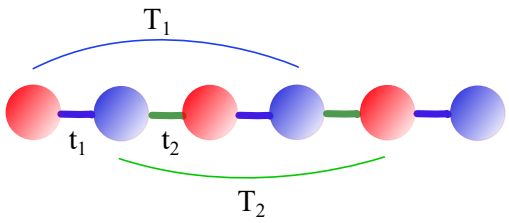


FIG. 7. Schematic set-up of a 1D SSH system. Here t_1 and t_2 are nearest-neighbor hoppings while T_1 and T_2 are second nearest-neighbor hoppings.

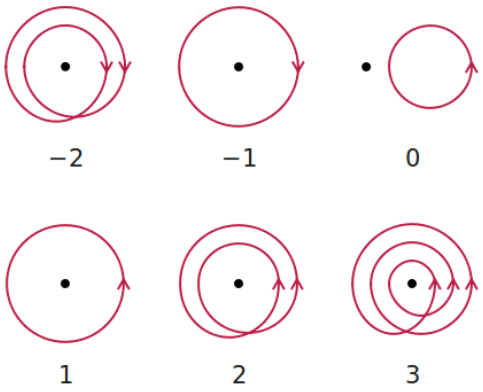


FIG. 8. Winding number. The winding number of a closed, oriented curve with respect to a reference point is a topological invariant that counts how many times the curve winds around the point. Picture credits: Jim Belk, public domain.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial or non-financial interests.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to N. L. Holanda.

SUPPLEMENTARY MATERIAL

The SSH model

The SSH model [58] describes the movement of free electrons along a dimerized chain whose basic units consist of two distinct atoms. This movement, usually called "hopping" in the literature, can be made either between atoms in a unit cell or between unit cells, and the allowed hopping rules for a given system completely determine its Hamiltonian. This is because the kinetic energies of the electrons are parameterized by a vector of real numbers $\mathbf{t}$ that also encodes hopping terms, thus allowing for a compact mathematical description of a Hamiltonian in terms of creation/annihilation operators as

$$\mathbf{H}(\mathbf{t}) = \mathbf{c}^{\dagger} H(\mathbf{t}) \mathbf{c} \tag{22}$$

where the column vector

$$\mathbf{c} = \left(c_1^A, c_1^B, \cdots, c_{\frac{N}{2}}^A, c_{\frac{N}{2}}^B \right)^T$$

contains annihilation operators $c_p^{A(B)}$ that erase electrons at atom A (B) and lattice site p and similarly the row vector

$$\mathbf{c}^{\dagger} = \left(c_1^{A\dagger}, c_1^{B\dagger}, \cdots, c_{\frac{N}{2}}^{A\dagger}, c_{\frac{N}{2}}^{B\dagger} \right)$$

contains creation operators $c_p^{A(B)\dagger}$ that produce electrons at atom A (B) and lattice site p. Please note that N is twice the number of unit cells in the chain and therefore an even integer.

The convenience of equation (22) is that all information about a system such as its eigenstates and eigenenergies can be recovered from the $N \times N$ matrix $H(\mathbf{t})$. We can thus think of the vectors $\mathbf{t}$ in parameter space as very compact representations of SSH models: each point in $\mathbf{t}$-space can be mapped to a $N \times N$ matrix $H(\mathbf{t})$ whose eigenvectors and eigenvalues can then be computed, as is usually done in quantum mechanics. As an example, a general matrix $H(\mathbf{t})$ describing a SSH system with hoppings between nearest and second nearest neighbors is given by

$$H(t_1, t_2, T_1, T_2) = \begin{pmatrix} 0 & t_1 & 0 & T_1 & 0 & \cdots \\ t_1 & 0 & t_2 & 0 & T_2 & \cdots \\ 0 & t_2 & 0 & t_1 & 0 & \cdots \\ T_1 & 0 & t_1 & 0 & t_2 & \cdots \\ 0 & T_2 & 0 & t_2 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}_{N \times N} . \tag{23}$$

Knowing the vector $\mathbf{t} = (t_1, t_2, T_1, T_2)$ corresponding to a particular system described by the Hamiltonian in equation (23) should suffice to compute any of its physical properties, including its topological phase. Figure 7 depicts a SSH system described by equation (23).

The reason why topological materials garnered so much interest in recent years is that their physical properties are topologically robust. This means that these properties are stable under continuous (i.e., adiabatic) mathematical operations performed on the system's underlying wave functions. This topological robustness is expressed theoretically in terms of a topological invariant that characterizes different phases of a system. In the particular case of the SSH model, the topological invariant used to classify the topological phases is the winding number.

The winding number is a topological property of any closed, oriented curve that measures how many times the curve winds around a point that does not belong to itself. It can be any integer and is usually chosen to be positive when the curve winds in counterclockwise motion with respect to the reference point (equivalently, when a closed, oriented curve winds in clockwise motion around a reference point its winding number is negative). Figure 8 shows several closed, oriented curves and their winding numbers computed with respect to a given point.

An interesting property of topological invariants like the winding number is that they are a global feature of geometric objects: for each of the curves in figure 8 for example the winding number is a property of the whole curve that cannot be defined locally for each of its points. This fundamental characteristic of topological invariants makes the study of topological phases of matter from local lattice data a challenging task.

For SSH systems with translational symmetry like the finite systems with periodic boundary conditions investigated in the article, the winding number is usually computed in wavevector space via

$$W = \frac{1}{4\pi i} \int_0^{2\pi} dk Tr(\sigma_3 H(k)^{-1} \partial_k H(k)), \tag{24}$$

where $H(k)$ is the kernel in wavevector space of a Hamiltonian $\mathbf{H}(\mathbf{t})$ and σ_3 is the chiral operator. Equation (24) can be evaluated for several Hamiltonians by varying the parameter $\mathbf{t}$, resulting in phase diagrams in parameter space like the ones shown in figure 1 in the article.

Learning topological phases from real space data

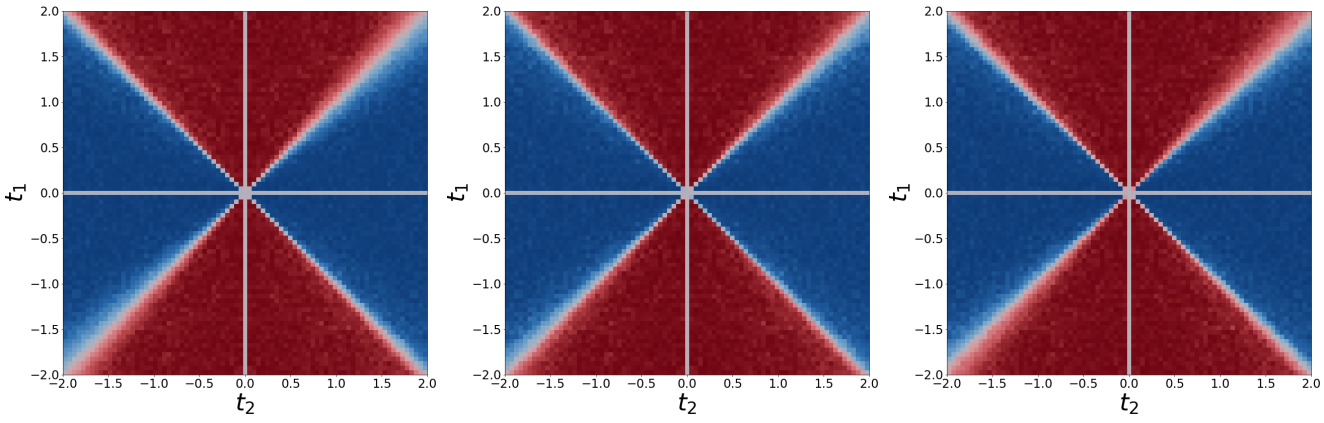
We now discuss the principles that lead us to the intuition that learning topological phases from local real space data should be possible.

First, while the topological invariants that characterize distinct topological phases are usually computed in wavevector space, the topological properties of a Hamiltonian are the same regardless of the basis in Hilbert space used to represent it. Thus, information on the topological phase of a Hamiltonian should still be available when it is represented in real lattice space.

Second, even though the topological properties of a Hamiltonian are global, meaning that in general they cannot be said to be localized at a particular lattice site, in parameter space topology is indeed a local property: knowing the vector $\mathbf{t}$ associated with a Hamiltonian completely determines its topological phase.

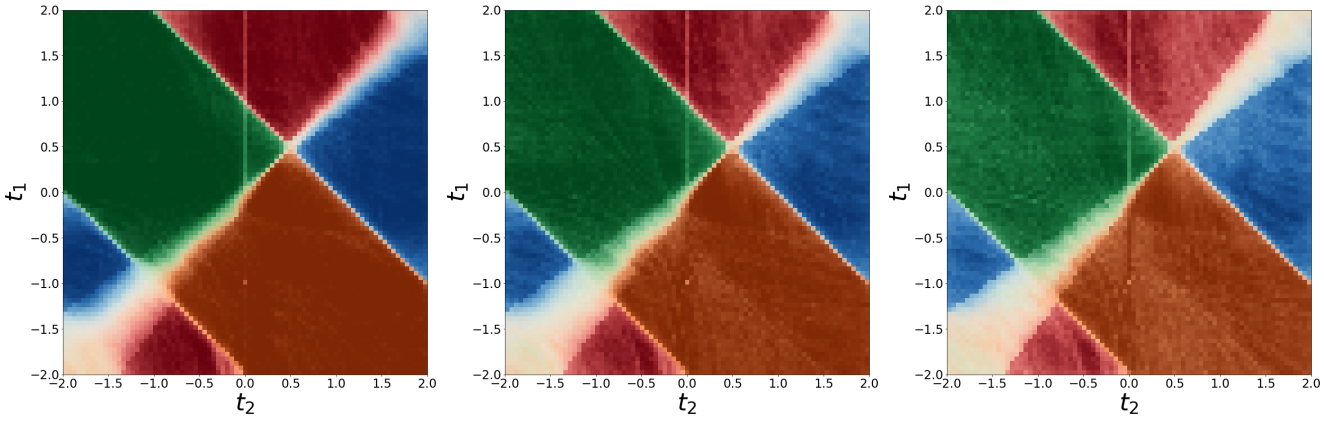
In a data-driven approach, locality is often exploited by means of a local constancy hypothesis [59]. Mathematically, this policy prescribes the value of a function $W(\mathbf{t}')$ at points where it is unknown in a vicinity of a data point $\mathbf{t}$ as approximately equal to its known value $W(\mathbf{t})$,

$$W(\mathbf{t} + \boldsymbol{\delta}) \approx W(\mathbf{t}). \tag{25}$$



(a) (b) (c)

FIG. 9. Phase diagrams learned from compressed representations of the SSH 1 system. (a) Phase diagram learned using the real space features $X_{\mathcal{S}_1}$. (b) Phase diagram learned from the DCT topological features $\hat{X}^{c}_{\mathcal{S}_1,\mathcal{E}_1}$. (b) Phase diagram learned from the DST topological features $\hat{X}^{s}_{\mathcal{S}_1,\mathcal{O}_1}$.



(a) (b) (c)

FIG. 10. Phase diagrams learned from compressed representations of the SSH 2 system. (a) Phase diagram learned using the real space features $X_{\mathcal{S}_2}$. (b) Phase diagram learned from the DCT topological features $\hat{X}^{c}_{\mathcal{S}_2,\mathcal{E}_2}$. (b) Phase diagram learned from the DST topological features $\hat{X}^{s}_{\mathcal{S}_2,\mathcal{O}_2}$.

That such a policy will be successful in classifying topological phases in parameter space can be visualized in figure 1 in the article where we draw phase diagrams of SSH models with first-neighbor (figure 1(a)) and first- and second-neighbor (figure 1(b)) hoppings. In figure 1(a) for example, it is clear that knowing a particular Hamiltonian $H(t_1, t_2)$ with winding number $W = 0$ (that is, in one of the red regions) means that there is a small neighborhood around (t_1, t_2) in which all Hamiltonians belong to the same topological phase. Were we able to collect data on the topological phases of several Hamiltonians in parameter space, the problem of learning phase boundaries in a supervised setting would reduce to a standard problem of curve estimation which could be tackled with conventional machine learning algorithms.

It does not immediately follow, however, that the same strategy will be successful in real space. Indeed, at first sight it may appear that a local constancy policy should be able to easily exploit locality in real space through the diagonalization maps $v^{(j,l)} : \mathbb{R}^h \to \mathbb{R}^N$,

$$\mathbf{t} = (t_1, .., t_h) \to \left(v^{(j,1)}(t_1, ..., t_h), ..., v^{(j,N)}(t_1, ..., t_h)\right) = \mathbf{v}^{(j)}(\mathbf{t}) \tag{26}$$

where $j = 1, ..., N$ and $\mathbf{v}^{(j)}(\mathbf{t})$ is an eigenvector of the Hamiltonian $H(\mathbf{t})$. The trouble with this reasoning is that it disregards the high dimensionality of real space, i.e., the fact that $h \ll N$.

This fact is well illustrated by the numerical experiments discussed in the article. Although it may seem from figures 2(b) and 3(b) (from the article) in 2D parameter space that we have used a large number of data points for this learning task, it is important to note that in the numerical experiments the decision trees have taken as inputs 100D eigenvectors in real lattice space. In such high-dimensional spaces, the data should be much sparser.

The difficulty arising from machine learning problems in high-dimensional spaces is commonly referred to as *the curse of dimensionality* [60]. It essentially expresses the fact that the amount of data needed to ensure a machine learning algorithm will generalize well out of its training set grows exponentially with the dimensionality of feature space.

These apparently conflicting facets of our learning problem are harmonized by the manifold hypothesis [61, 62]: even though the eigenvectors exist in a high-dimensional space (100D in the numerical experiments), they are actually much lower-dimensional surfaces (2D in the numerical experiments) embedded in this space. Furthermore, the different classes in our problem correspond to different submanifolds as can easily be seen in parameter space (this is often referred to as the manifold hypothesis for classification [63]). As we have demonstrated in the article, only a small fraction of the 2D surfaces (i.e., eigenvector lattice coordinates $v^{(j,l)}(t_1, t_2)$) were needed to retrieve the 2D parameter space phase diagrams from the 100D real space data, which reveals that there is a lot of redundancy in the information content of real space eigenvectors. This fact can be exploited to generate compressed representations of SSH systems that still carry most of the relevant topological information, as can be seen in figures 9 and 10.

Numerical explorations on longer lattices

A natural question that comes to mind regarding the information entropy signatures presented here is whether these signals are artifacts of the numerical procedure used to generate them.

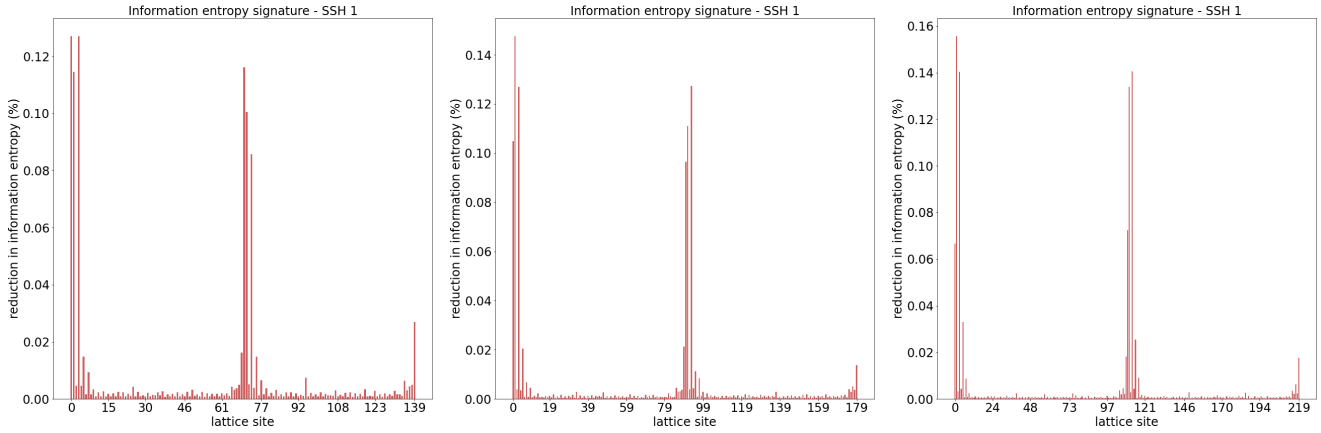
The eigenvector ensembling algorithm used in this work contains steps for generating data (step 1), sampling training data (step 2) and training a supervised learning algorithm on the sampled training data (step 3). Each of these steps can generate misleading artifacts that do not represent real properties of the physical systems we are investigating.

Artifacts resulting from randomization in the eigenvector ensembling algorithm can be traced to sampling (step 2) or any random components in the supervised learning algorithm used (step 3). As an example, random forests allocate subsets of features stochastically to each of its decision trees, thus generating a randomization effect. The bootstrapping (step 5) is designed to remove artifacts originating from randomization in the eigenvector ensembling algorithm.

There are still artifacts that might arise from the hyperparameters used to generate the data. These hyperparameters include lattice size and grid specifications (i.e., choices regarding the discretization of parameter space). Such artifacts can only be controlled for by bootstrapping over different hyperparameter settings, which may be computationally prohibitive. Intuitively, the hyperparameter we identify as likely having the most noticeable effect in the information entropy signatures is lattice size.

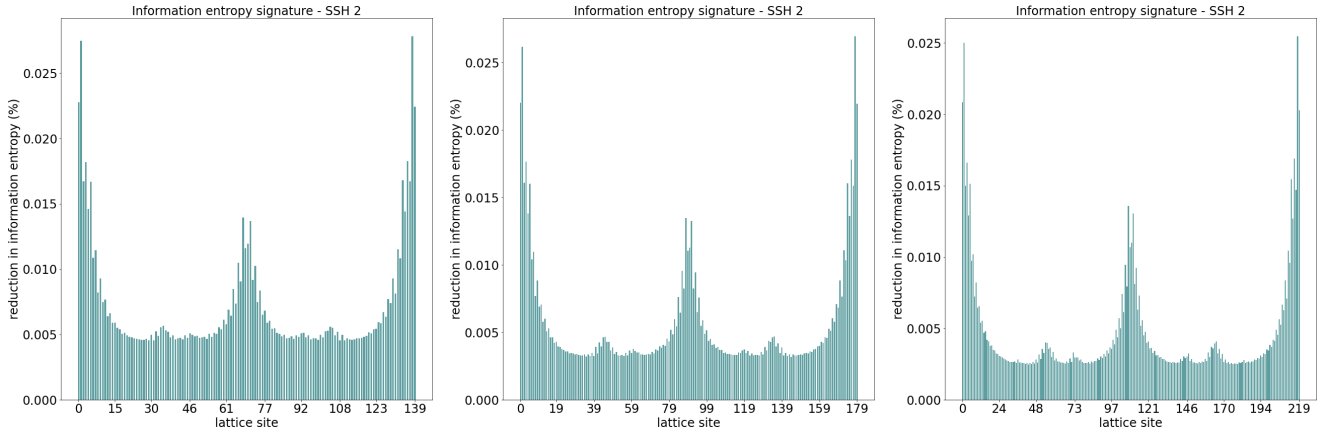
All results presented in the article were obtained for lattices with 50 unit cells. Each cell contains two different atoms, thus leading to 100×100 Hamiltonian matrices and their corresponding eigenvectors in $\mathbb{R}^{100}$. Here we present the information entropy signatures obtained for lattices with 70, 90 and 110 unit cells for both experiments (figures 11 and 12). These signals were generated in exactly the same way as the information entropy signatures obtained for 50 unit cells (i.e., after bootstrapping $n_{exp} = 100$ times and averaging lattice site relevances across all iterations).

It is clear from figures 11 and 12 that the patterns seen with 50 unit cells (figures 6(a) and 6(b) in the article) are stable across higher lattice sizes, with finer details emerging in the signals as the lattice size is increased. This naturally leads us to speculation about the continuum limit of information entropy signatures, a matter formally explored in the article.



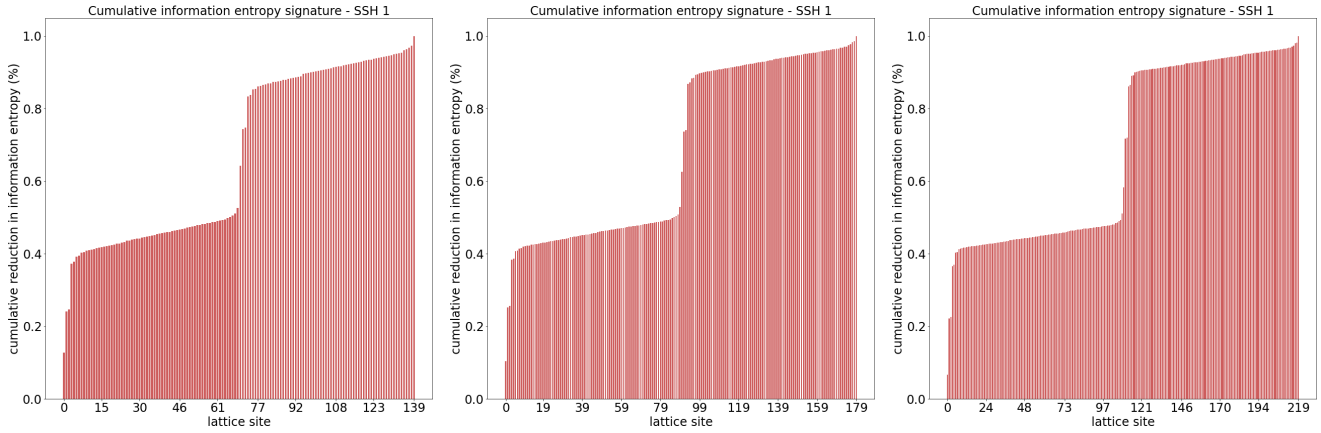
(a) (b) (c)

FIG. 11. Information entropy signatures obtained for experiment 1 with higher lattice sizes. (a) Information entropy signature for 70 unit cells. (b) Information entropy signature for 90 unit cells. (c) Information entropy signature for 110 unit cells.



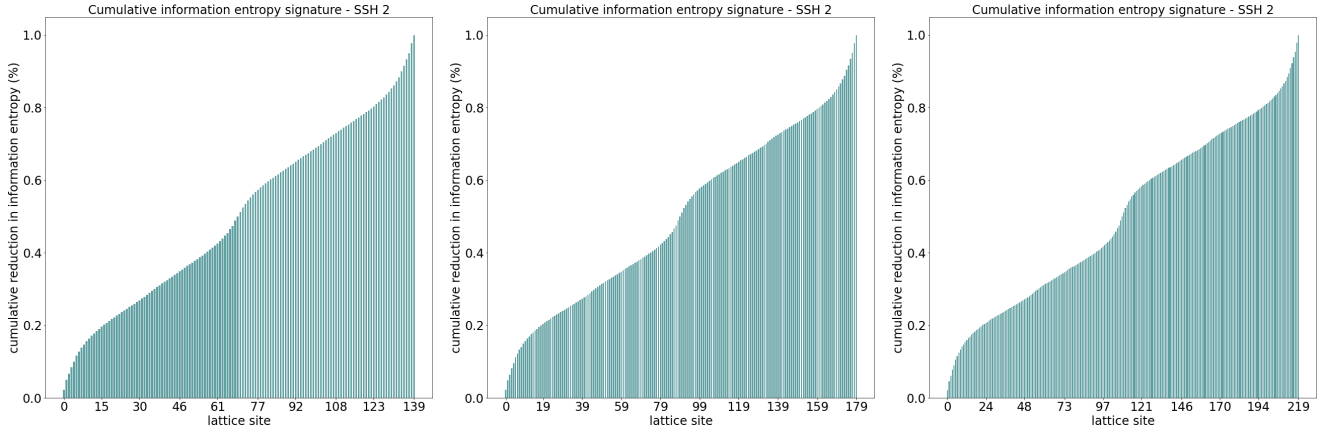
(a) (b) (c)

FIG. 12. Information entropy signatures obtained for experiment 2 with higher lattice sizes. (a) Information entropy signature for 70 unit cells. (b) Information entropy signature for 90 unit cells. (c) Information entropy signature for 110 unit cells.



(a) (b) (c)

FIG. 13. Cumulative entropy distributions for higher lattice sizes in experiment 1. (a) Cumulative entropy distribution for 70 unit cells. (b) Cumulative entropy distribution for 90 unit cells. (c) Cumulative entropy distribution for 110 unit cells.



(a) (b) (c)

FIG. 14. Cumulative entropy distributions for higher lattice sizes in experiment 2. (a) Cumulative entropy distributions for 70 unit cells. (b) Cumulative entropy distributions for 90 unit cells. (c) Cumulative entropy distributions for 110 unit cells.