

GrainBoundary.py

This module provides functionalities to generate matrices encoding site-wise Fourier transforms of Bloch functions for systems containing grain boundaries in 2 or 3 dimensions. It includes utilities to generate lattices representing grain boundaries and their mappings, as well as functions to create the relevant Fourier transform matrices.

Function Descriptions:

1. `lattices_2d(...)`:
 - Description: Generates 2D labeled lattices representing grain boundaries and their mapped versions. This function returns the original and mapped lattices with specific configurations based on parameters like the number of defects, defect spacing, and lattice dimensions.
2. `lattices_3d(...)`:
 - Description: Generates 3D labeled lattices representing grain boundaries and their mapped versions. This function returns the original and mapped lattices with specific configurations based on parameters like the lattice depth, number of defects, defect spacing, and lattice dimensions.
3. `gen_matrices_2d(...)`:
 - Description: Generates sparse matrices encoding the Fourier transforms of Bloch functions for a given 2D lattice. These matrices represent different Fourier components, such as $\cos(k_x \cdot a)$, $\cos(k_y \cdot a)$, $\sin(k_x \cdot a)$, $\sin(k_y \cdot a)$, etc., and their interplay with the grain boundary.
4. `gen_matrices_3d(...)`:
 - Description: Similar to the 2D version, this function generates sparse matrices encoding the Fourier transforms of Bloch functions for a 3D lattice. These matrices represent Fourier components and their interactions in a 3D system with grain boundaries.
5. `grain_boundary_2d(...)`:
 - Description: A wrapper function that generates the 2D lattice and then produces the Fourier transform matrices for a 2D system with grain boundaries based on specified parameters. It can return either the labeled lattice alone or both the labeled lattice and the Fourier transform matrices.
6. `grain_boundary_3d(...)`:
 - Description: A wrapper function that produces 3D lattices and Fourier transform matrices for a 3D system with grain boundaries based on specified parameters. Similar to its 2D counterpart, it can return either the labeled lattice alone or both the labeled lattice and the Fourier transform matrices.

RealSpace.py

This module focuses on generating Hamiltonians in real space for systems with grain boundaries. It provides functionalities for generating Hamiltonians for 2D and 3D systems with different interaction orders. The module leverages functionalities from the GrainBoundary module to create grain boundary configurations and then constructs real-space Hamiltonians based on these configurations, with potential for disorder. The main output is the eigenvalues and eigenvectors of these Hamiltonians.

Function Descriptions:

1. `multi_binary(n, dense=False)`:
 - Description: Generates a basis of mutually anti-commuting Hermitian matrices which generalize the Pauli matrices.
2. `disorder_matrix(w, lattice, order, dense=False)`:
 - Description: Generates a disorder matrix based on a given lattice. The matrix represents random on-site energies for each site in the lattice, and it can be in either dense or sparse format.
3. `dim2ord2(geom, Delta1, Delta2, theta, num_modes, w=None)`:
 - Description: Constructs the Hamiltonian for a 2D system with second-order topology, given the grain boundary geometry and other system parameters. This function returns the lattice, eigenvalues, and eigenvectors of the Hamiltonian.
4. `dim3ord2(geom, Delta1, Delta2, theta, num_modes, w=None)`:
 - Description: Constructs the Hamiltonian for a 3D system with second-order topology. Similar to its 2D counterpart, it returns the lattice, eigenvalues, and eigenvectors based on the provided grain boundary geometry and system parameters.
5. `dim3ord3(geom, Delta1, Delta2, Delta3, theta, num_modes, w=None)`:
 - Description: Constructs the Hamiltonian for a 3D system but with third-order topology. This function generates the Hamiltonian based on the grain boundary geometry, system parameters, and interaction strengths, and then returns the lattice, eigenvalues, and eigenvectors of the Hamiltonian.

ReciprocalSpace.py

This module provides functionalities related to the reciprocal space representation of systems with grain boundaries. It primarily focuses on the transformation of eigenvectors from real space to reciprocal space, using Fourier transformations. The module supports transformations for both 2D and 3D systems.

Function Descriptions:

1. `eigenvectors_to_lattice_format(lattice, eigenvectors)`:
 - Description: Reformats the eigenvectors, which are given in a flat array, into a shape consistent with the spatial lattice format. This helps with indexing and further transformations.
2. `fourier_transform_1D(psi_r_basis, k_res=2000)`:
 - Description: Performs a 1D Fourier transform on the given eigenvectors (in real space) to convert them to reciprocal space. The transformation focuses on grain boundary systems, capturing the important features around the boundary.
3. `fourier_transform_2D(psi_r_basis, kz_res=256, ky_res=256)`:
 - Description: Performs a 2D Fourier transform for 3D real space eigenvectors, capturing features in the ky-kz plane. Similar to its 1D counterpart, it emphasizes the grain boundary regions, but also focuses on momentum in the stacking direction (kz).
4. `fourier_transform_wrapper(lattice, eigenvectors)`:
 - Description: A wrapper function that determines whether to perform a 1D or 2D Fourier transform based on the dimensionality of the input lattice. It then calls the appropriate transformation function.

DataProcess.py

This module provides various functionalities to process and analyze the eigenvalue and eigenvector data of systems in real and reciprocal space. It offers tools for deriving the density of states (DOS), reducing the dimensionality of data, and detecting the significant peaks in the reciprocal space DOS to derive the grain boundary band structure.

Function Descriptions:

1. `n_closest(arr, val, n):`
 - Description: Finds the `n` closest values to `val` in the array `arr` and returns their indices.
2. `DOS_arrays(lattice, eigenvectors):`
 - Description: Computes the density of states (DOS) arrays for the provided eigenvectors. The DOS is computed based on the magnitude squared of the eigenvectors.
3. `reduce_dimensions(DOS_data, dimension):`
 - Description: Reduces the dimensions of the given DOS data by summing along all dimensions except the one specified by the `dimension` argument.
4. `enhanced_peak_detection(peak_data, threshold, extension, **kwargs):`
 - Description: Detects peaks in the provided data with enhanced accuracy by extending the data and normalizing it. The function uses the `find_peaks` function from `scipy.signal` to detect the peaks.
5. `grain_boundary_band_structure(eigenvalues, eigenvectors_k, reciprocal_lattice, peak_threshold=0.9, extension_size=50, **kwargs):`
 - Description: Computes the band structure of grain boundaries by detecting significant peaks in the density of states and associating them with eigenvalues (energies). The function returns arrays of momentum values and associated energy values.