Topological two-dimensional polymers Supporting Information

Maximilian A. Springer[1,2], Tsai-Jung Liu[2], Agnieszka Kuc[1], and Thomas Heine[1,2]

[1]Helmholtz-Zentrum Dresden-Rossendorf, Institute of Resource Ecology, Research Site Leipzig, Permoserstrasse 15, 04138 Leipzig, Germany
[2]TU Dresden, Faculty for Chemistry and Food Chemistry, Bergstrasse 66c, 01069 Dresden, Germany

Contents

1 Band structures and sketches for 2D nets

On the following pages, Tables S1-S4 show the tight-binding band structures that were calculated using scripts similar to the accompanying example script, based on the pythTB package.[1] The Fermi level was estimated by integrating over a grid of equidistant points in the first Brillouin zone. The investigated topologies are separated by the manifold of faces and vertices. First, Table S1 shows the band structures for the face- and vertex-transitive regular nets **hcb**, **hxl**, and **sql**. In Table S2 and S3, band structures for the semiregular (one type of vertex but different types of faces) and demiregular nets (exactly two types of vertices) can be found. Additionally, we show the tight-binding band structures for selected topologies that do not fit into one of these groups in Table S4.

Table S1: Sketches and tight-binding band structures for the regular nets **hcb**, **hxl**, and **sql**.

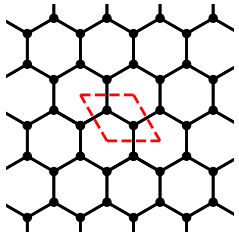
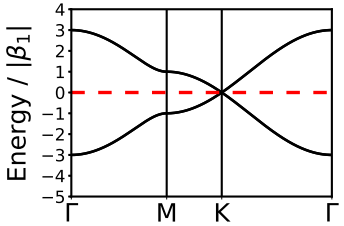
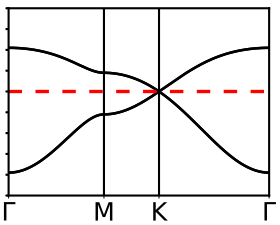
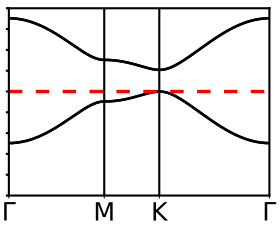
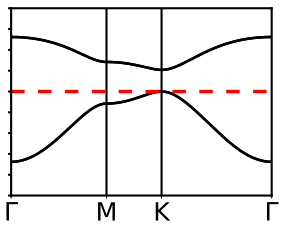
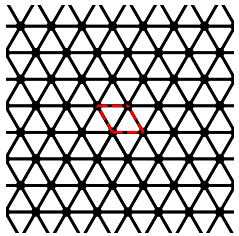
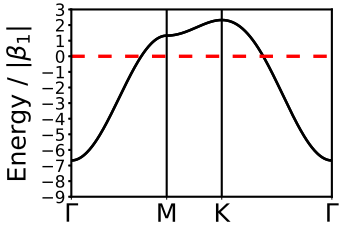
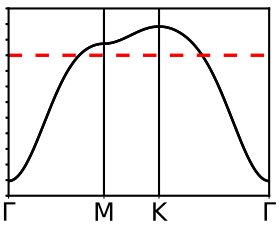
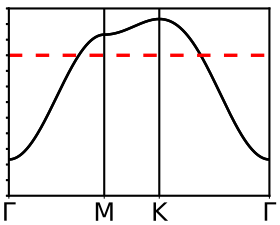
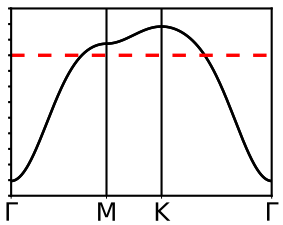
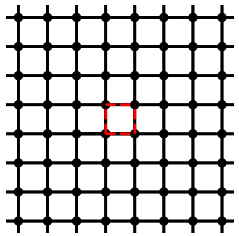
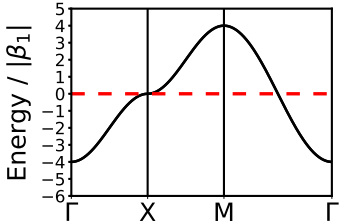
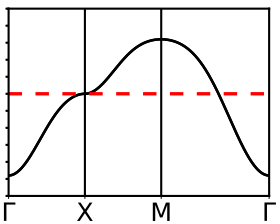
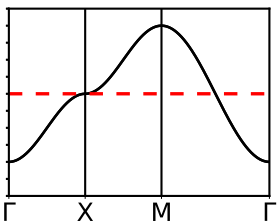
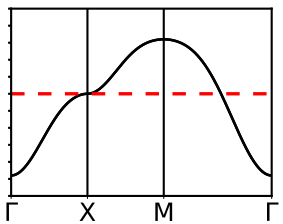
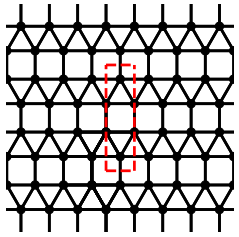
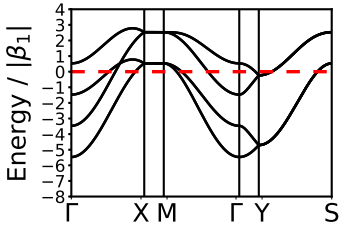
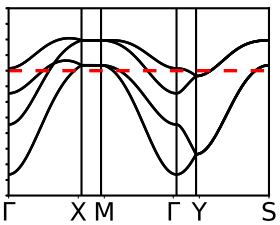
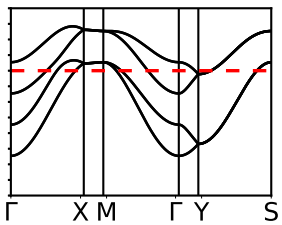
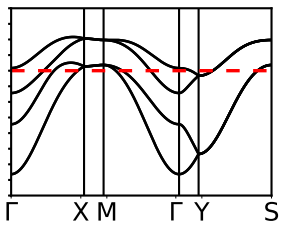
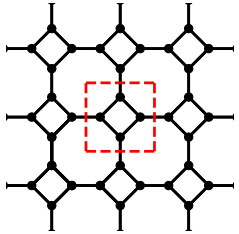
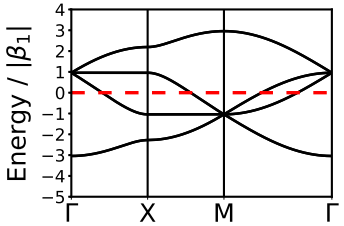
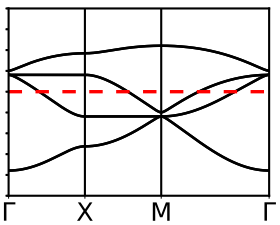
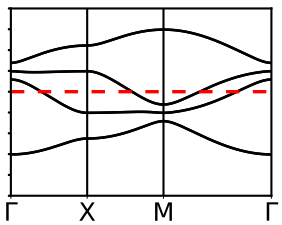
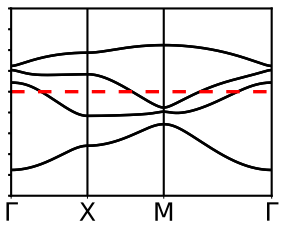
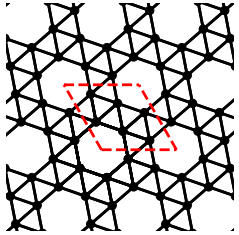
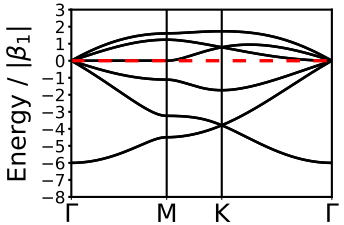
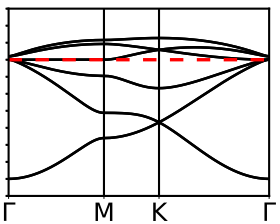
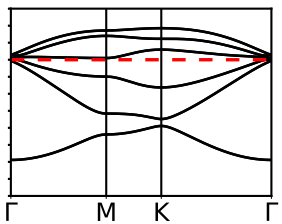
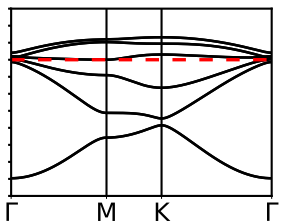
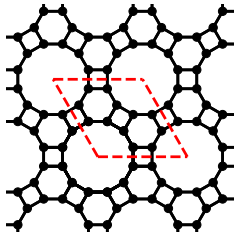
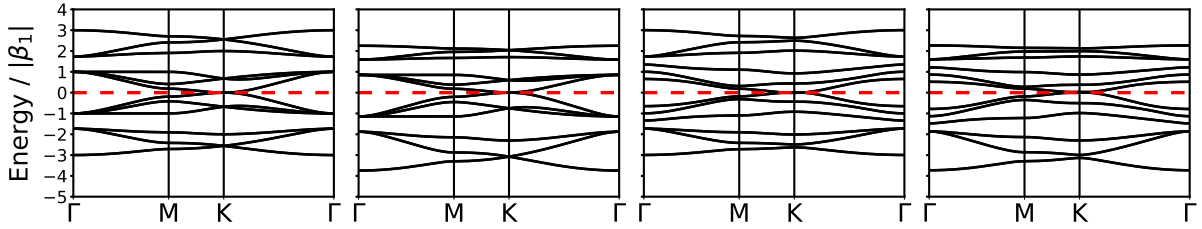
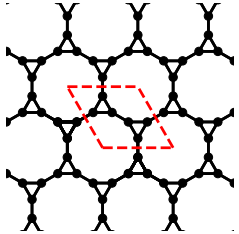
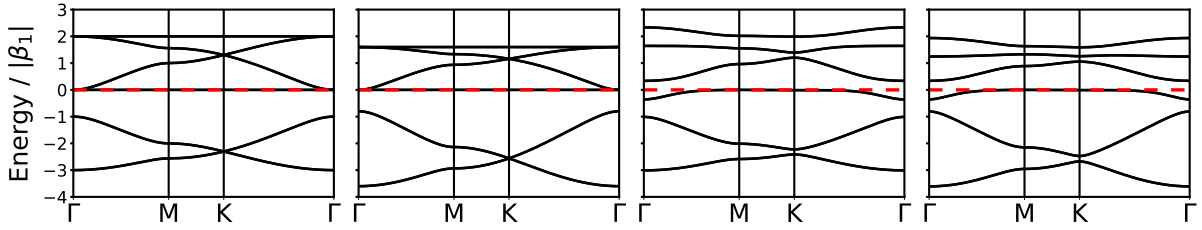
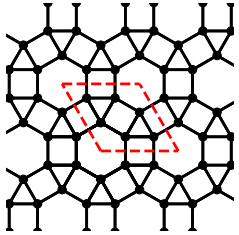
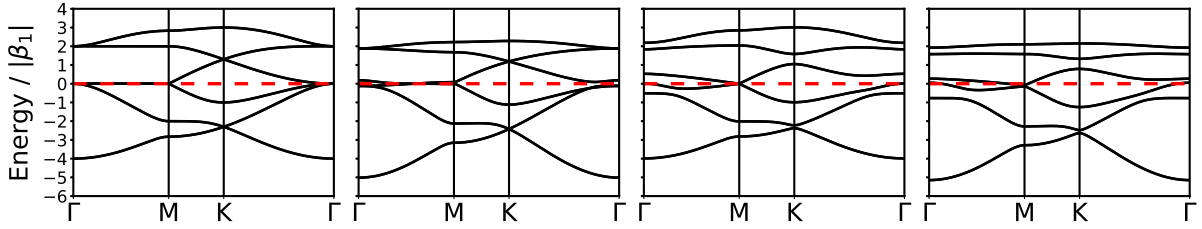
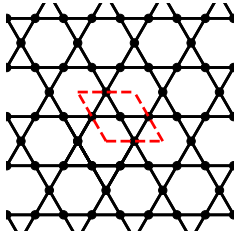
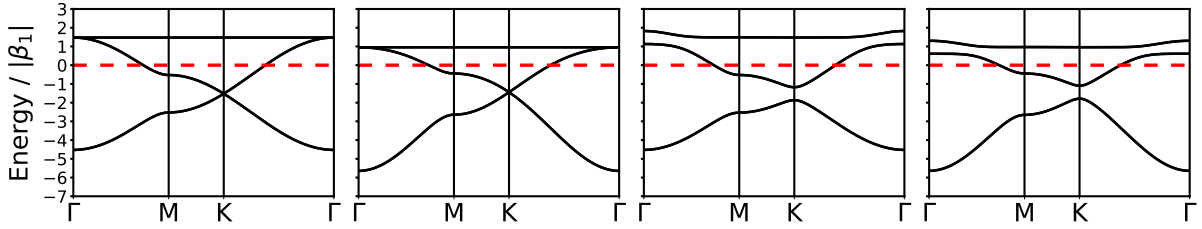
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
hcb					
hxl					
sql					

Table S2: Sketches and tight-binding band structures for the semiregular nets **cem**, **fes**, **fsz**, **fxt**, **hca**, **htb**, **kgm**, and **tts**.

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cem					
fes					
fsz					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
fxt					
hca					
htb					
kgm					

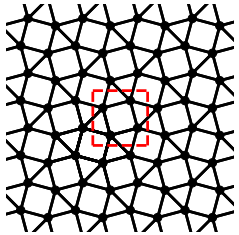
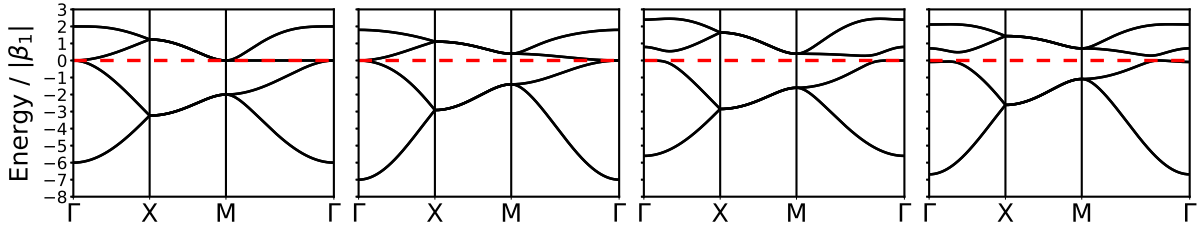
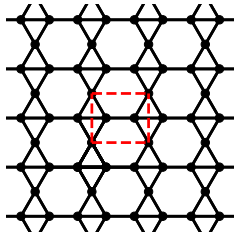
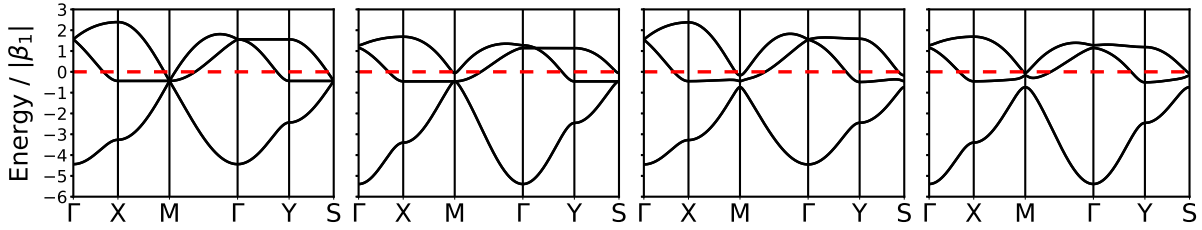
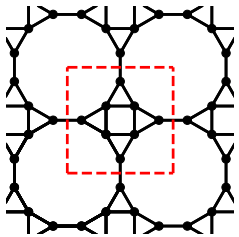
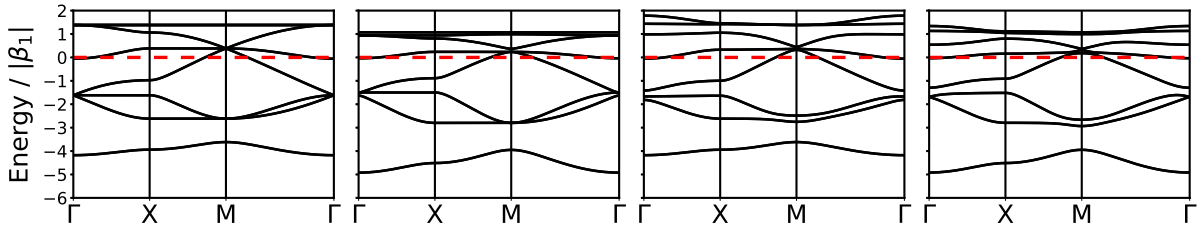
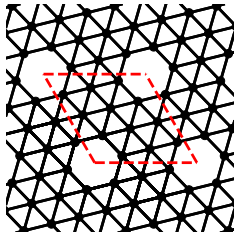
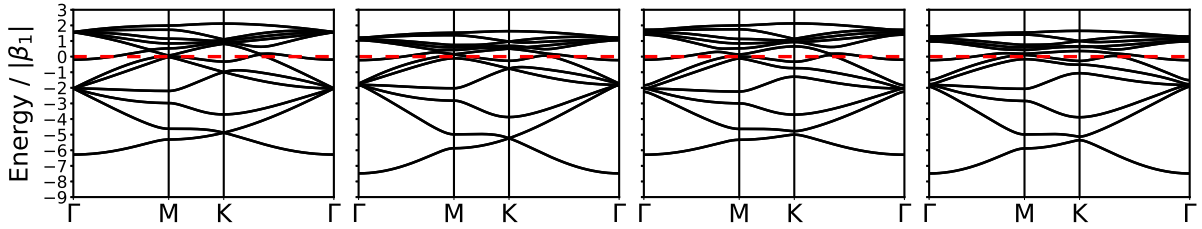
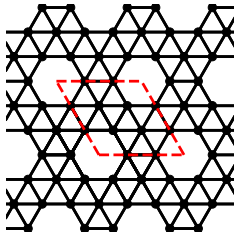
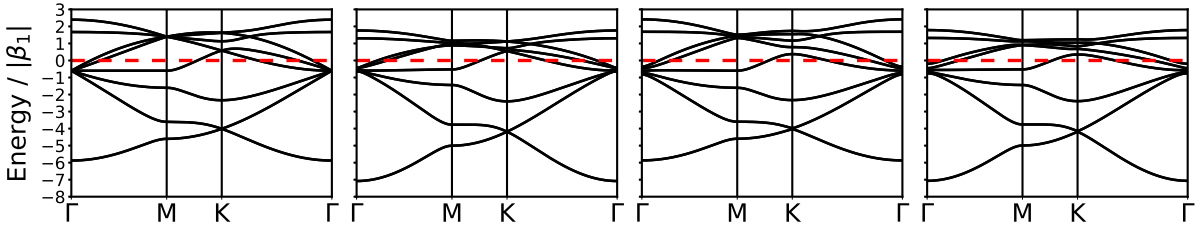
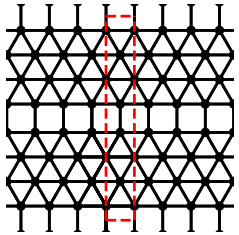
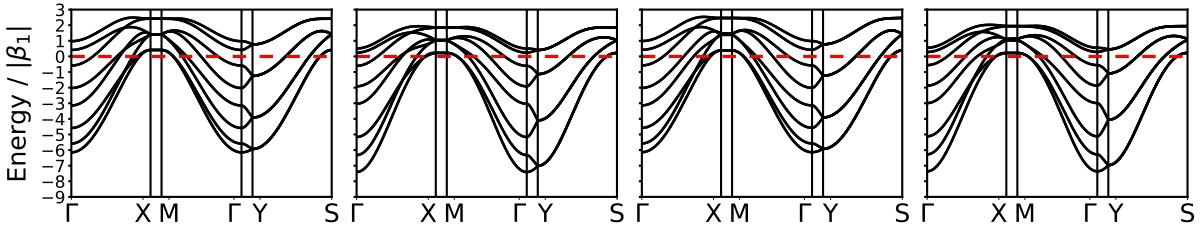
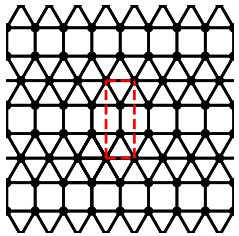
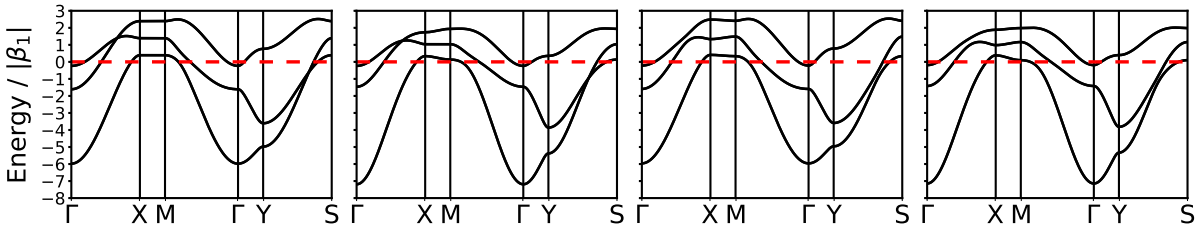
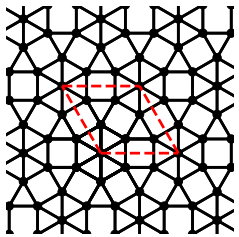
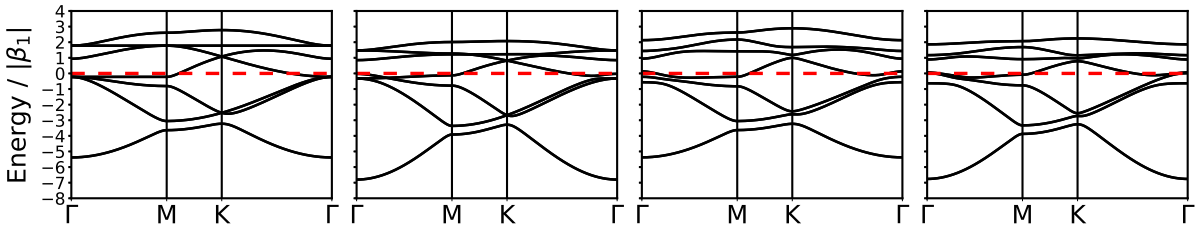
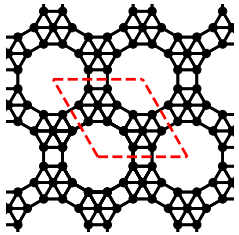
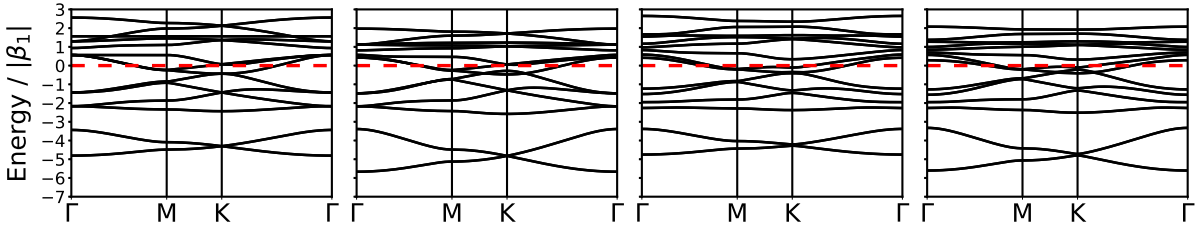
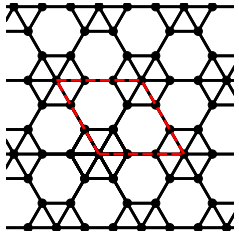
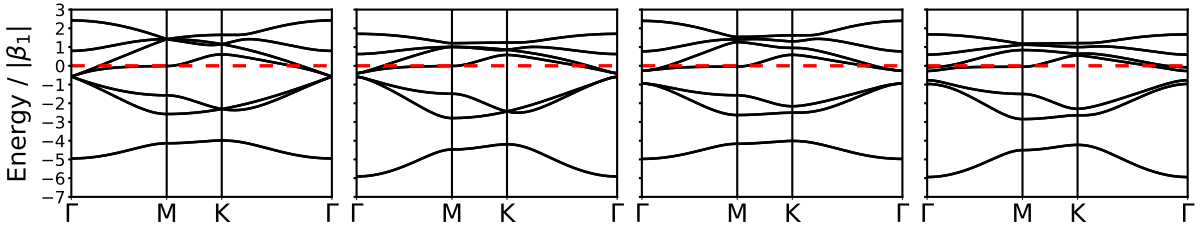
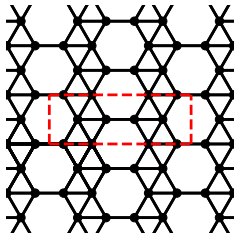
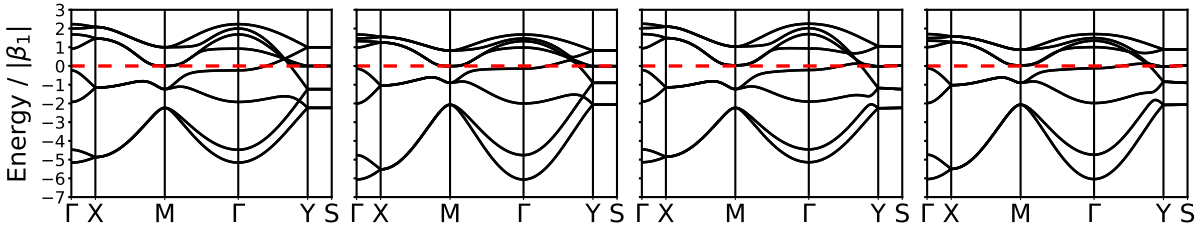
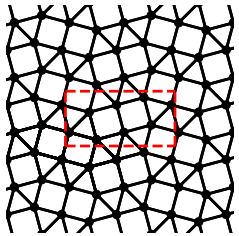
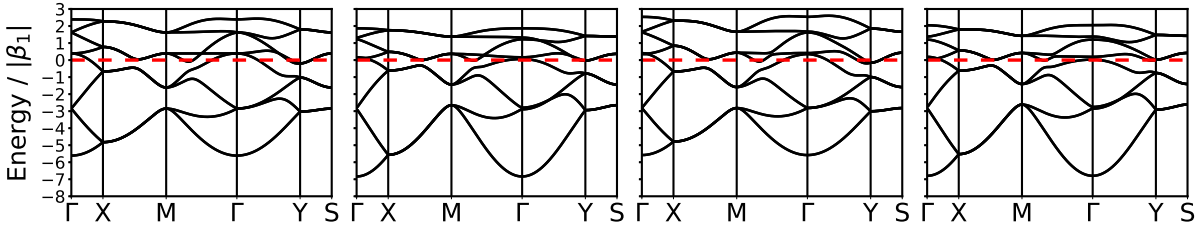
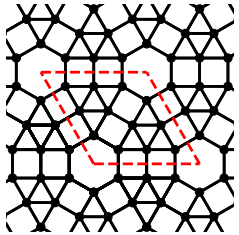
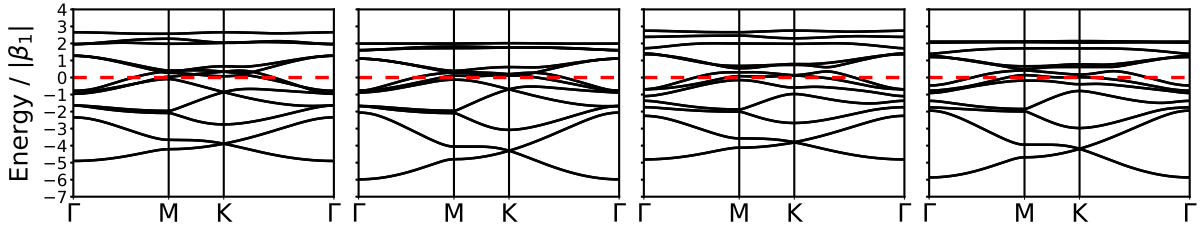
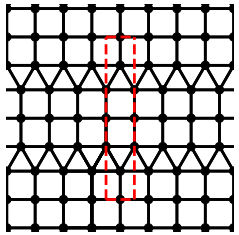
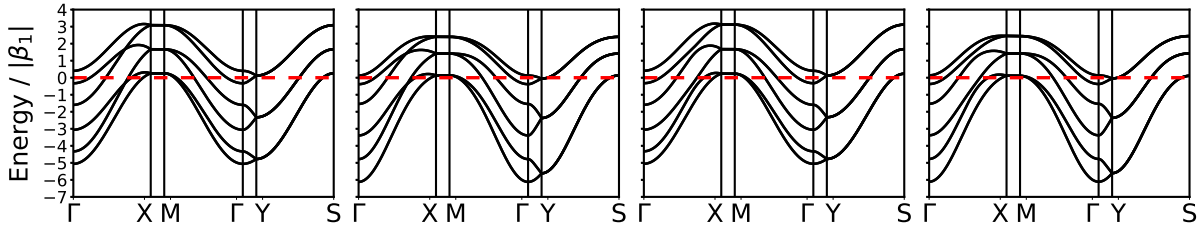
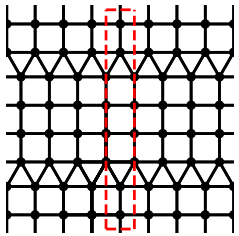
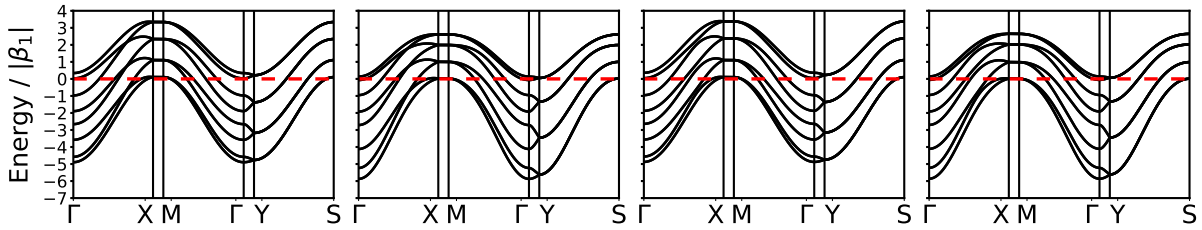
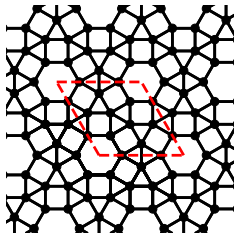
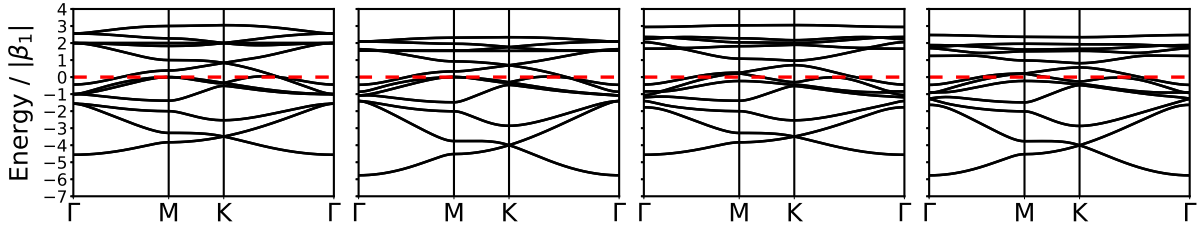
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
tts					

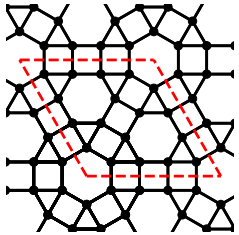
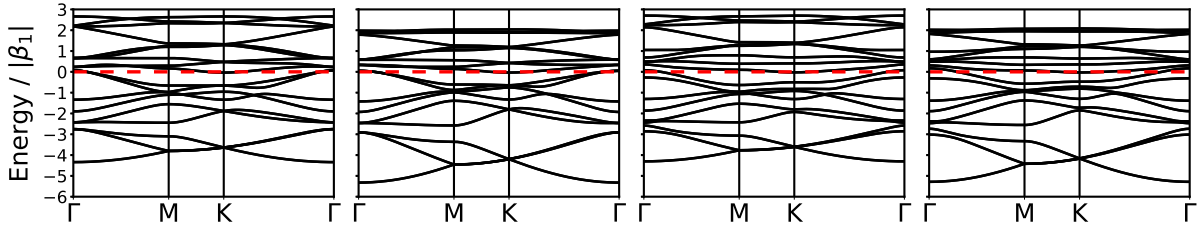
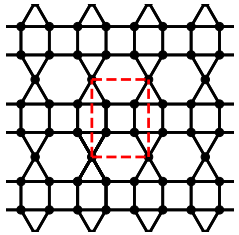
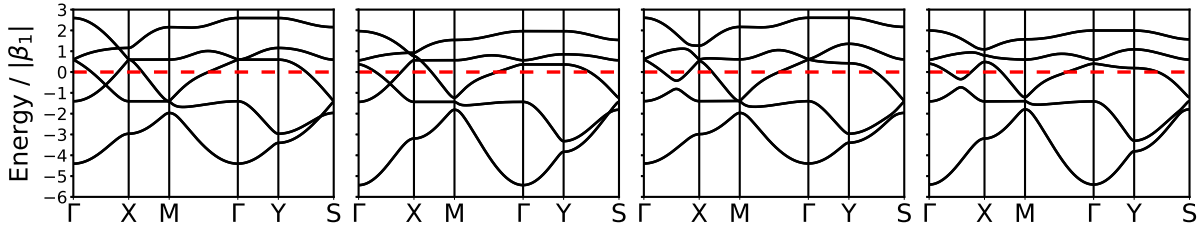
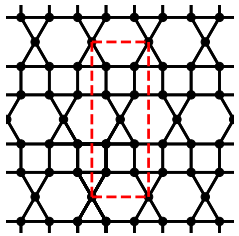
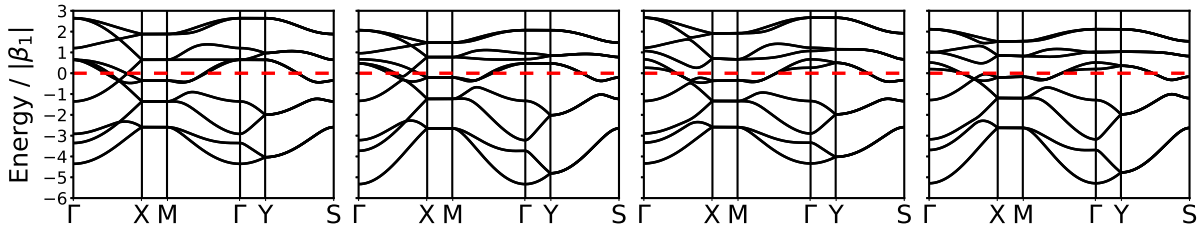
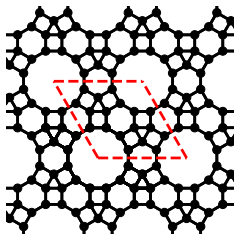
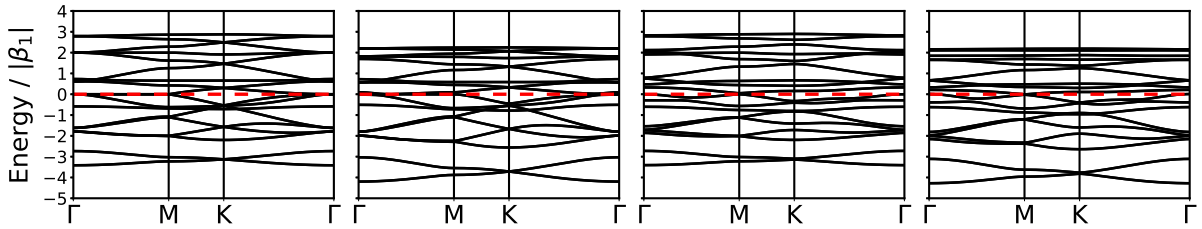
Table S3: Sketches and tight-binding band structures for the demiregular nets **bew**, **cph**, **kra**, **krb**, **krc**, **krd**, **kre**, **krf**, **krg**, **krh**, **krj**, **krk**, **krl**, **krm**, **krn**, **krq**, **krr**, **krs**, **krt**, and **usm**.

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
bew					
cph					
kra					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
krb					
krc					
krd					
kre					

Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
krf					
krg					
krh					
krj					

Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
krk					
krl					
krm					
krn					

Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
krq					
krr					
krs					
krt					

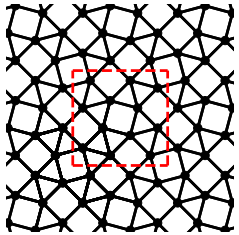
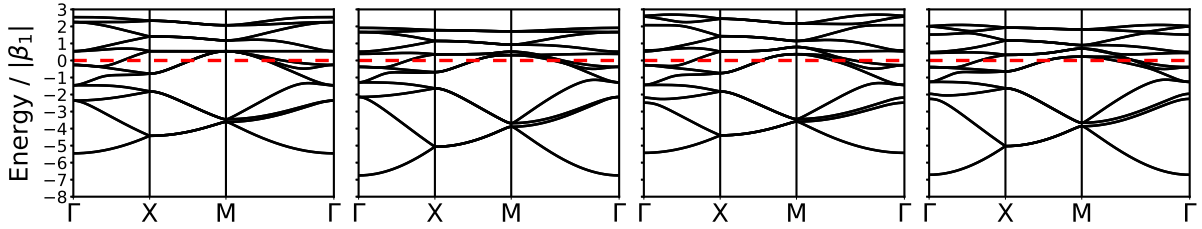
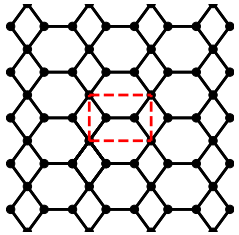
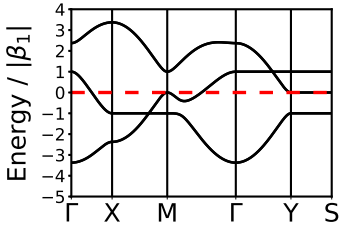
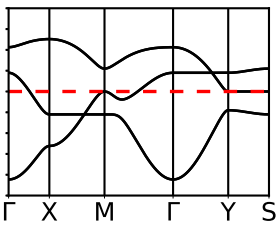
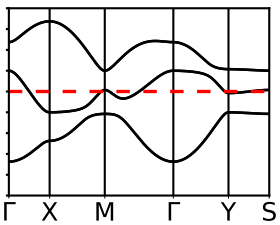
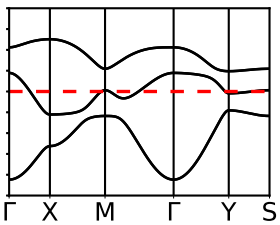
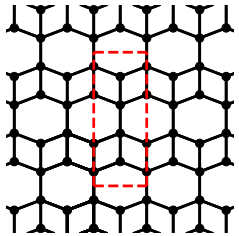
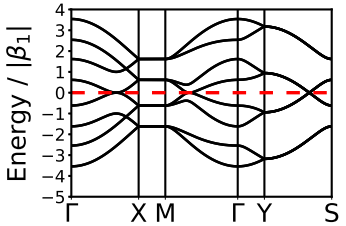
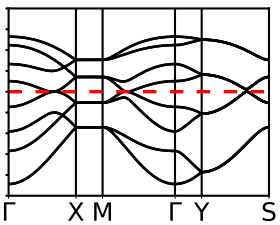
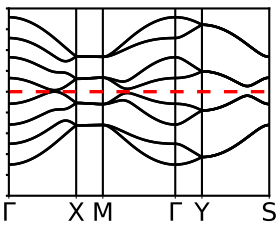
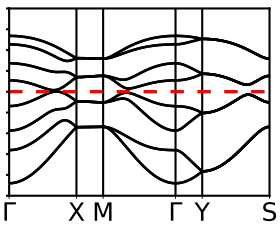
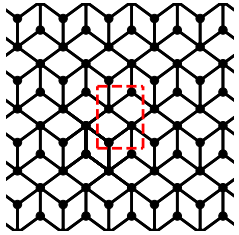
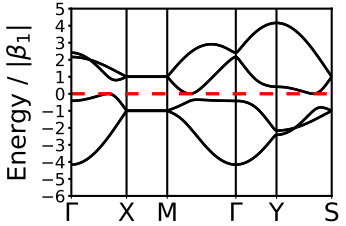
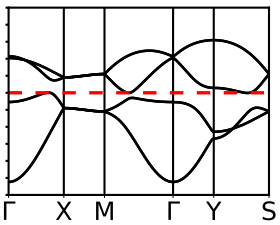
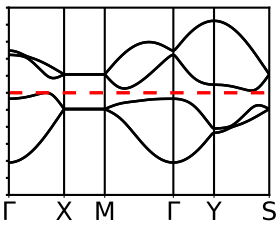
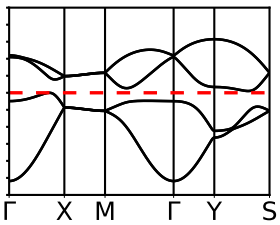
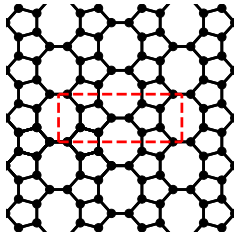
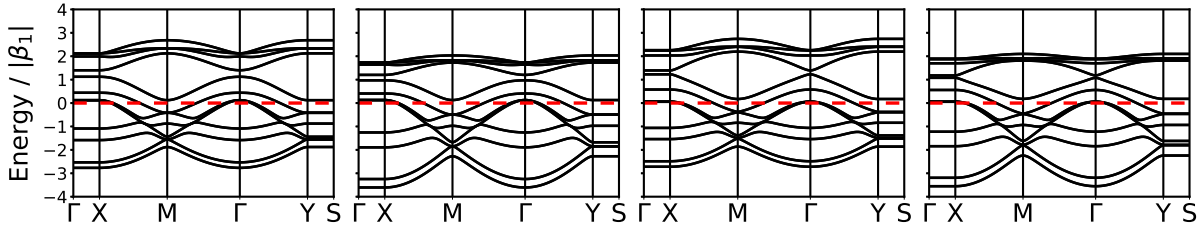
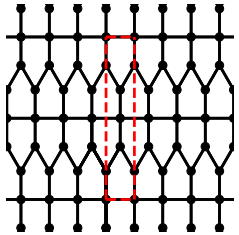
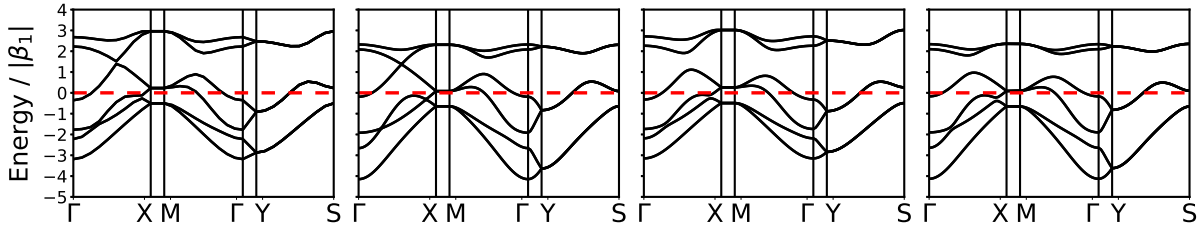
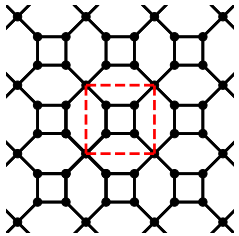
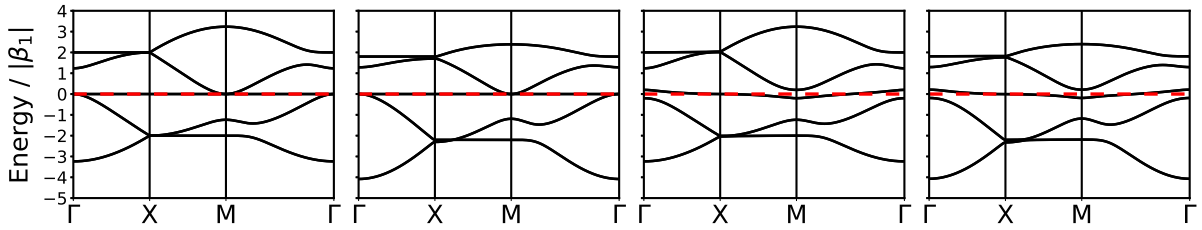
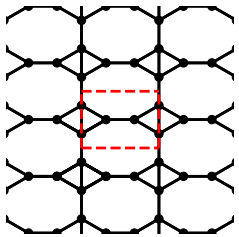
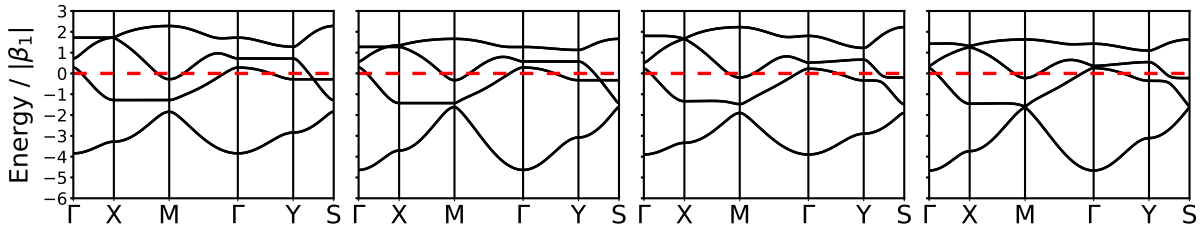
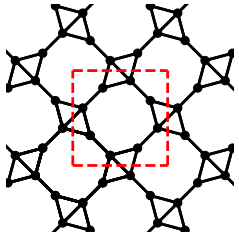
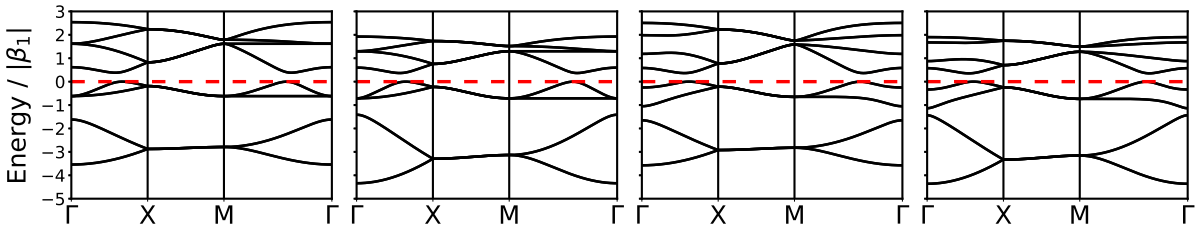
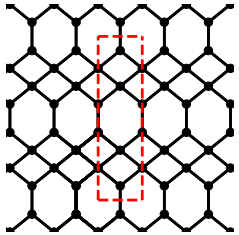
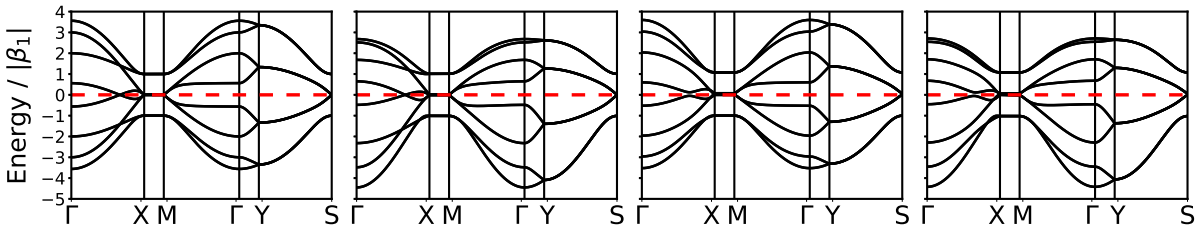
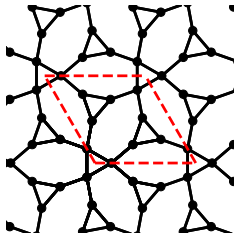
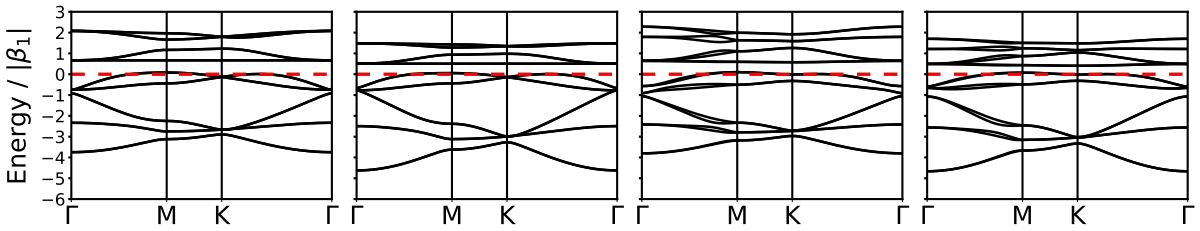
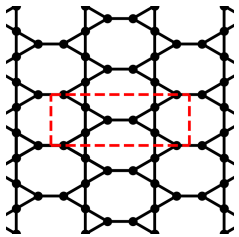
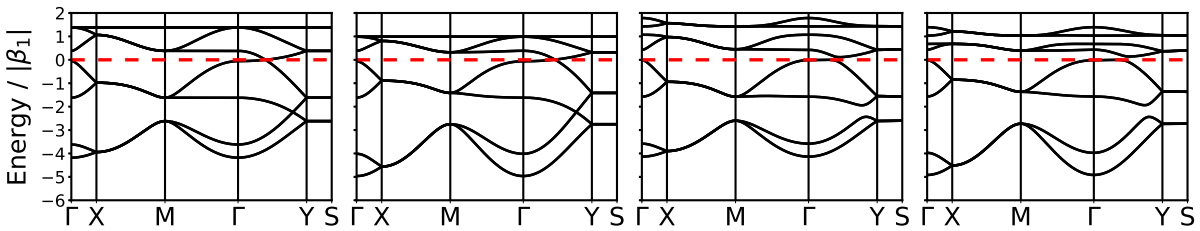
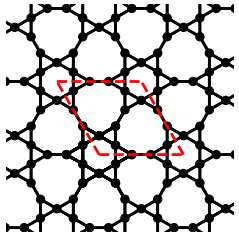
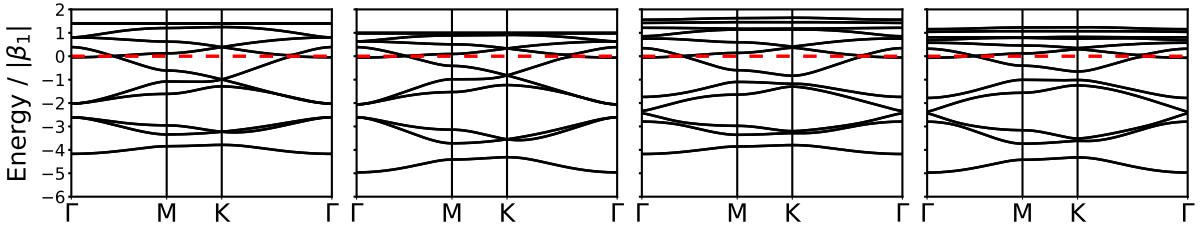
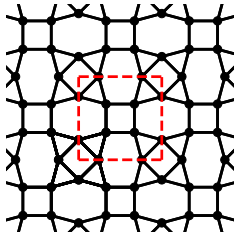
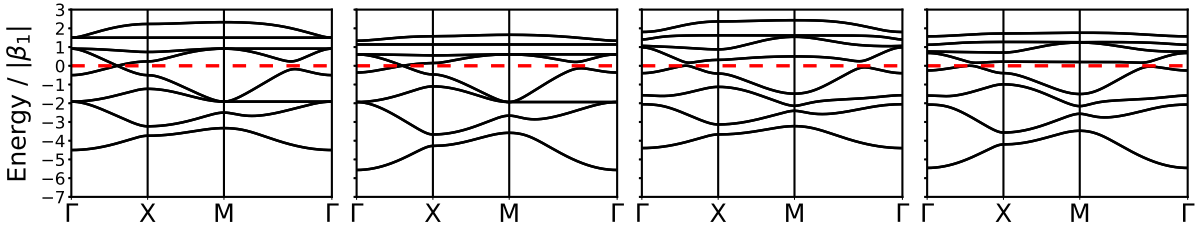
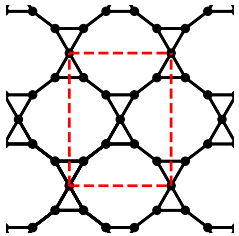
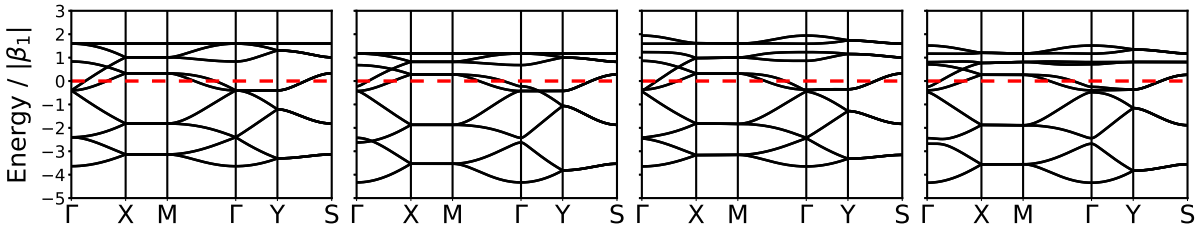
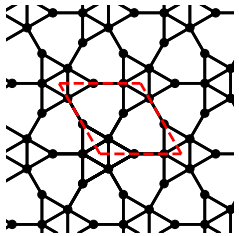
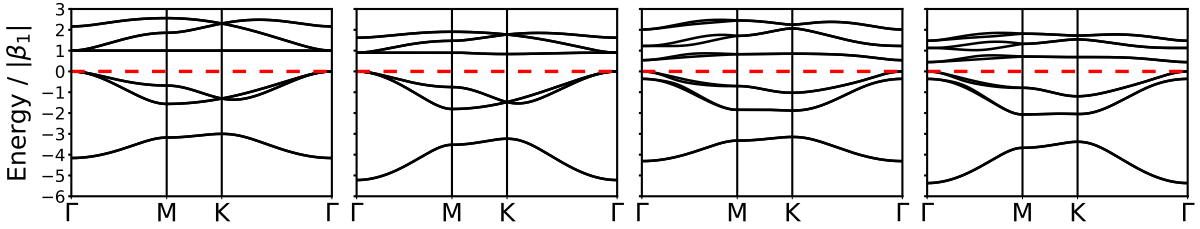
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
usm					

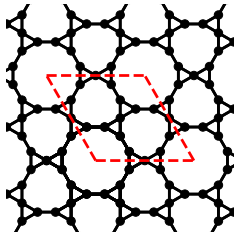
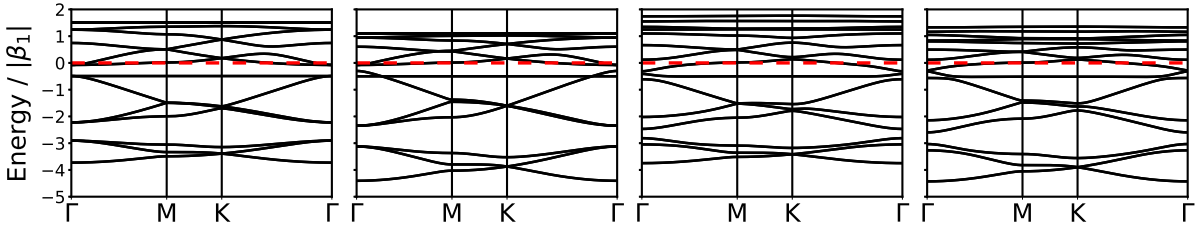
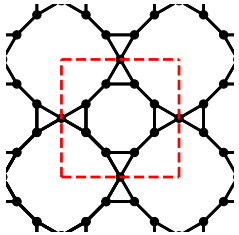
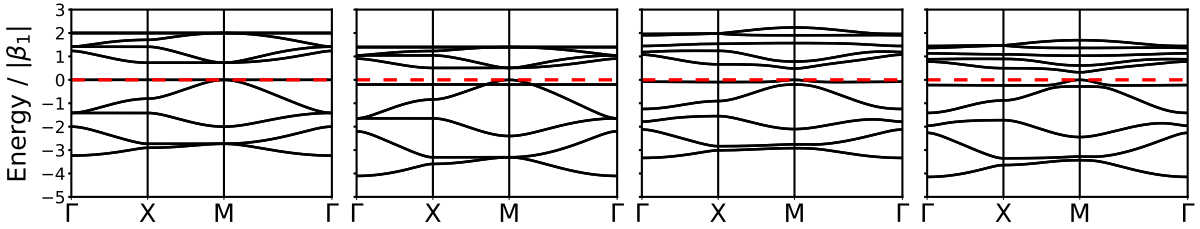
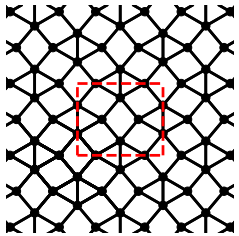
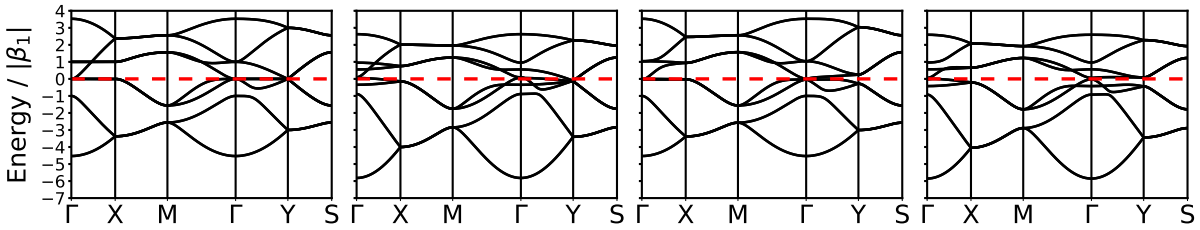
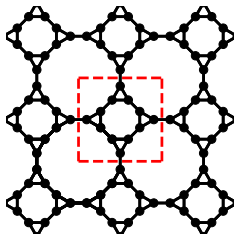
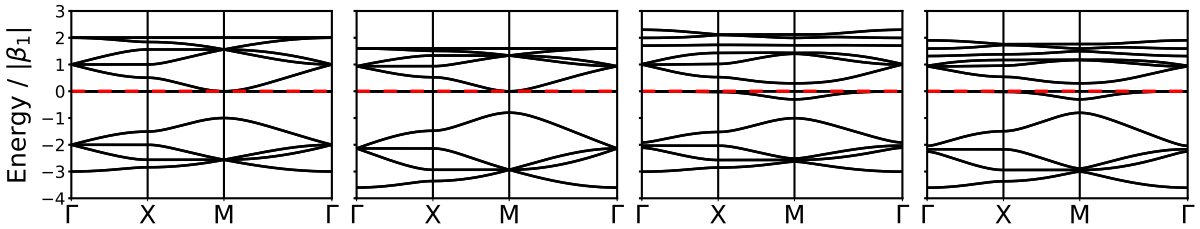
Table S4: Sketches and tight-binding band structures for additional nets.

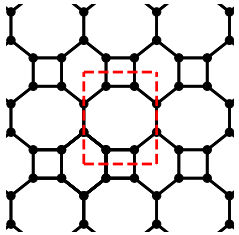
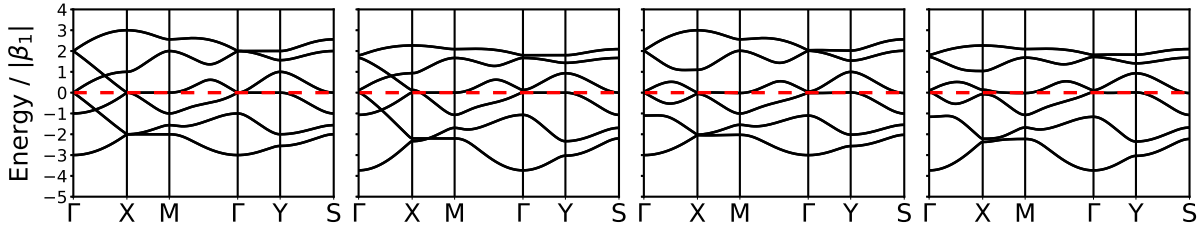
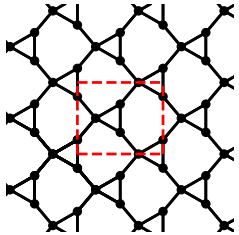
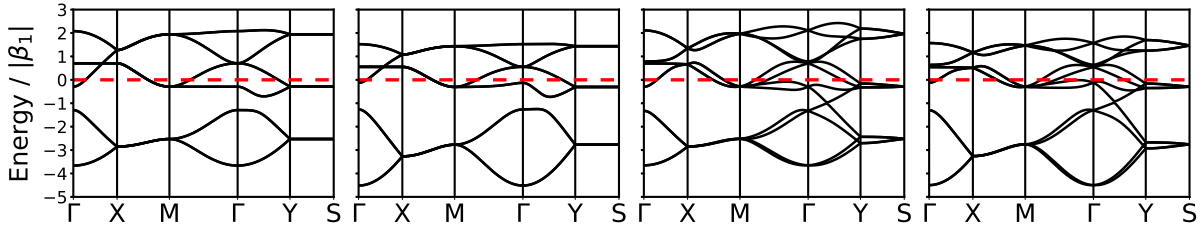
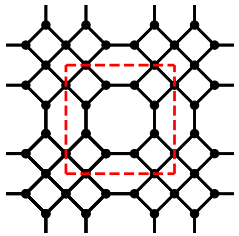
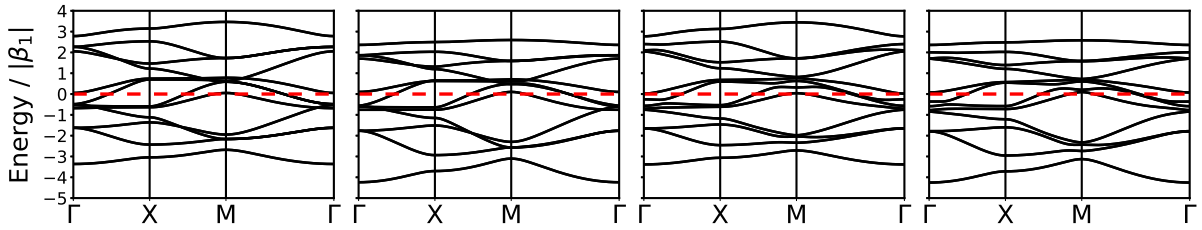
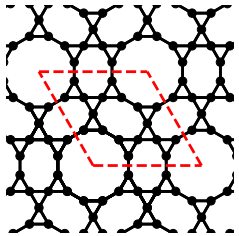
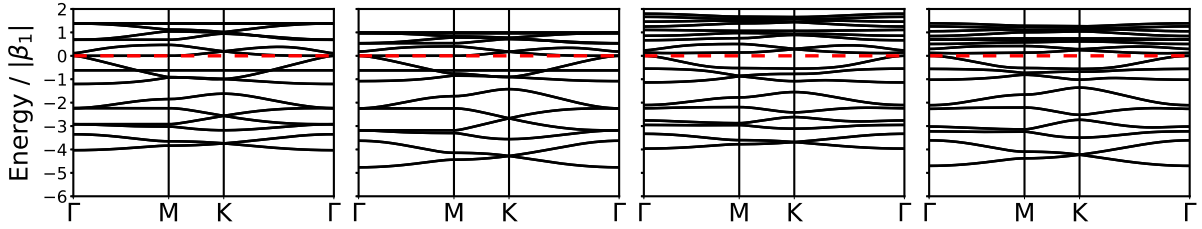
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
bex					
bey					
bhd					

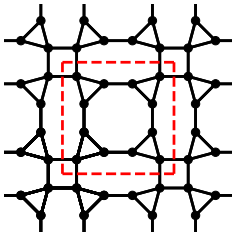
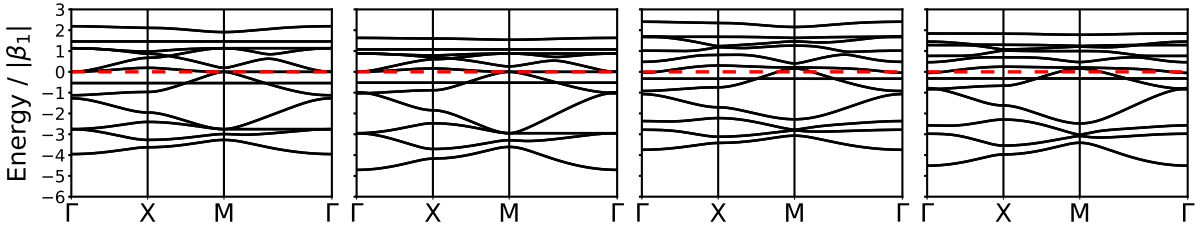
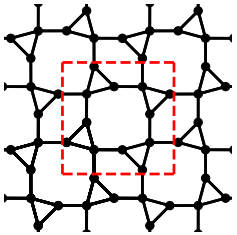
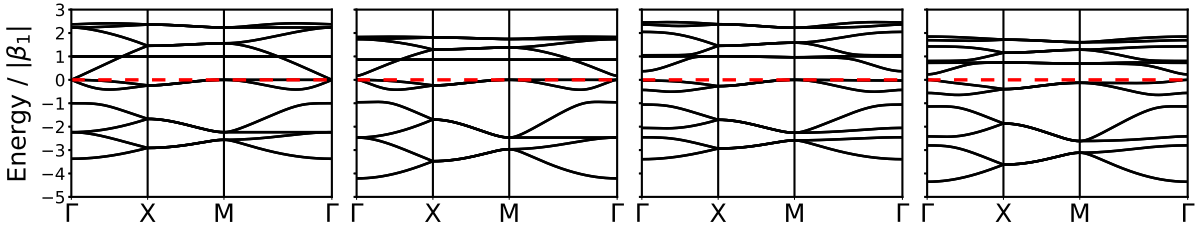
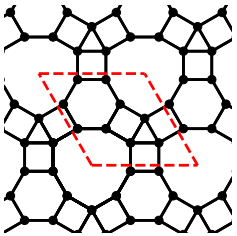
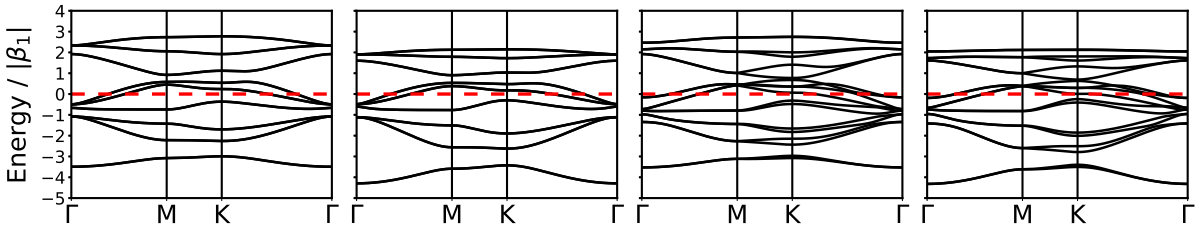
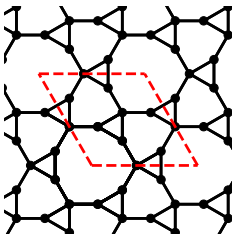
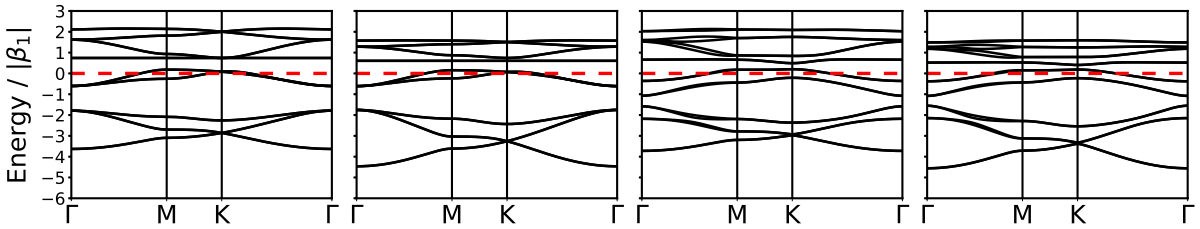
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
car					
cem-d					
cpa					
cpb					

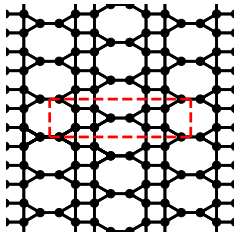
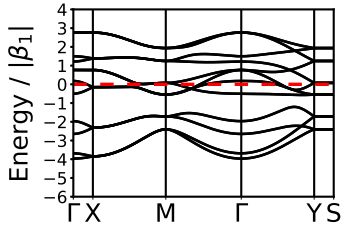
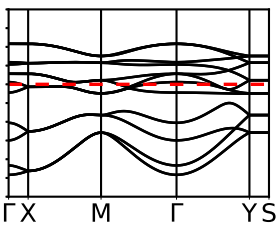
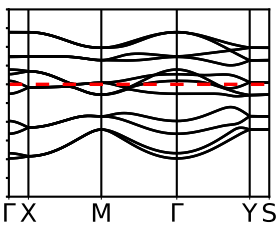
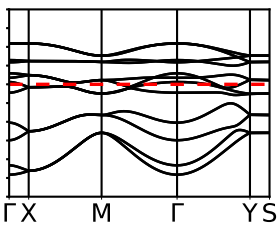
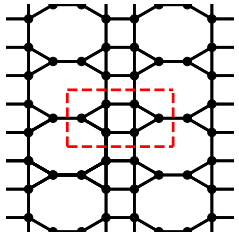
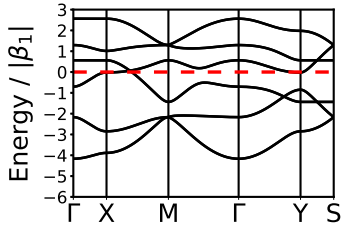
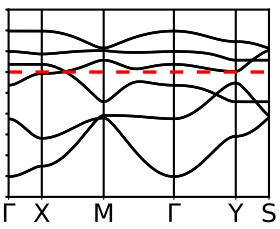
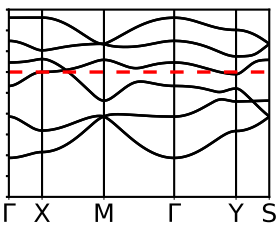
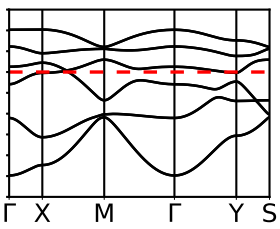
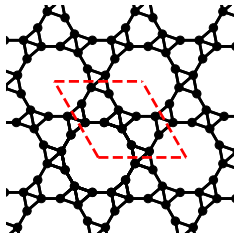
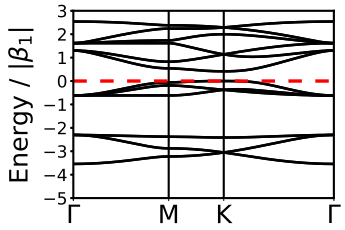
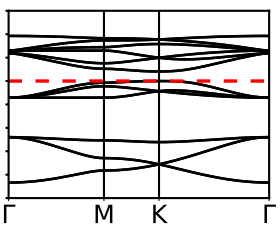
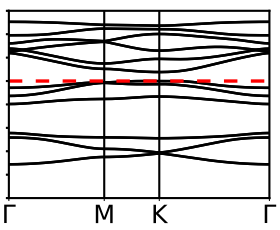
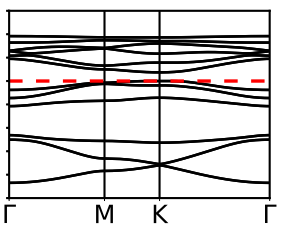
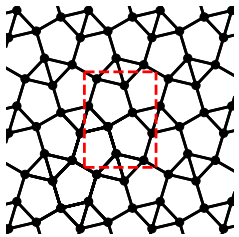
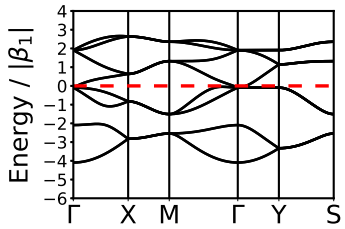
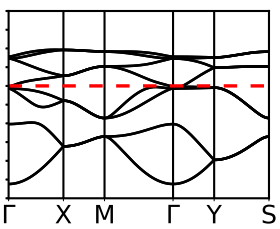
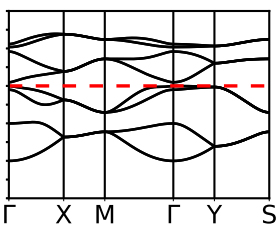
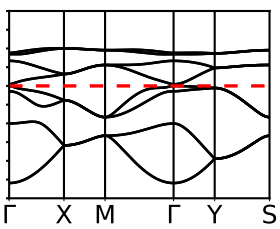
Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
cpc					
cpd					
cpe					
cpf					

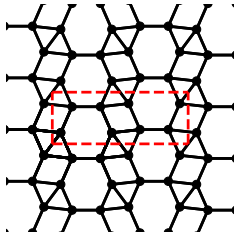
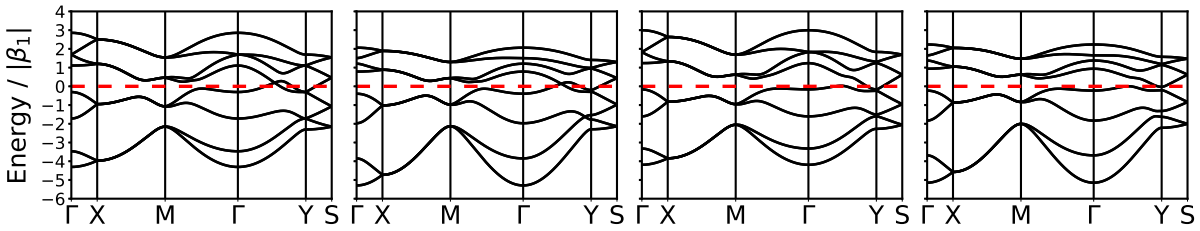
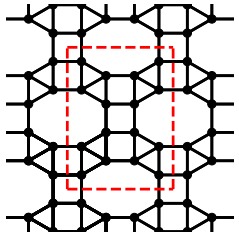
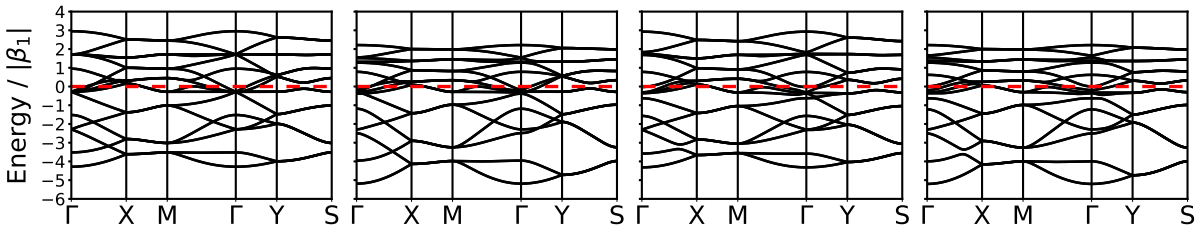
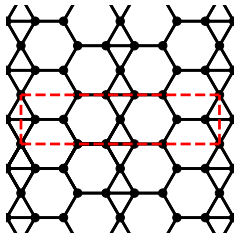
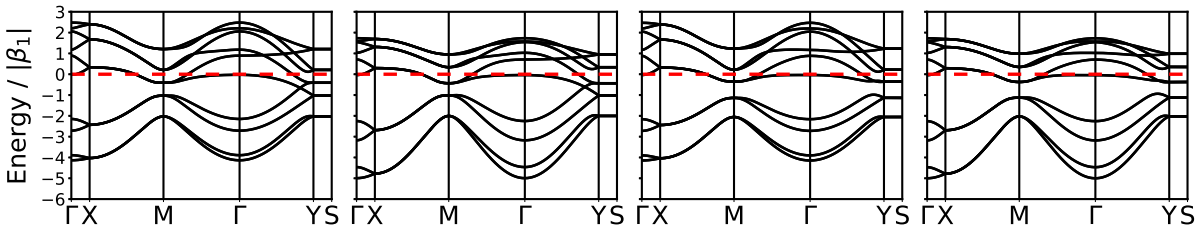
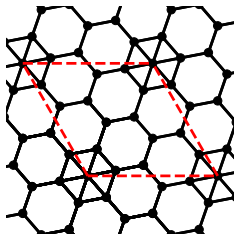
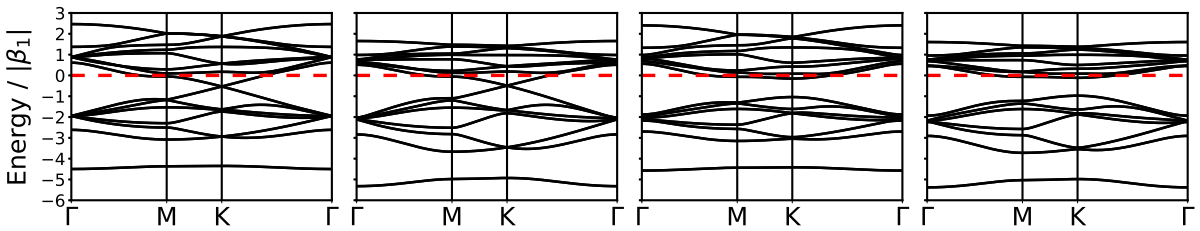
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cpg					
cpj					
cpk					
cpl					

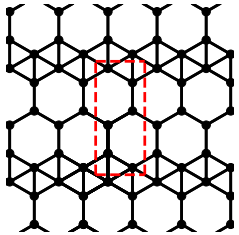
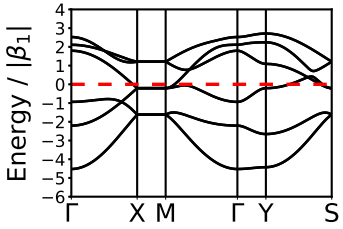
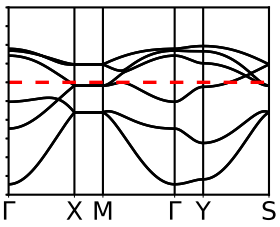
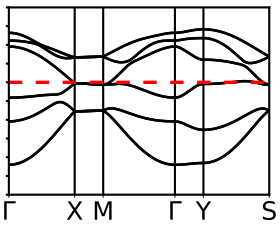
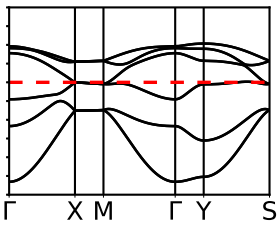
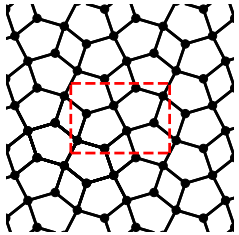
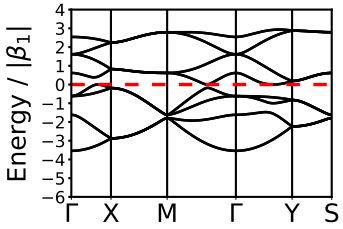
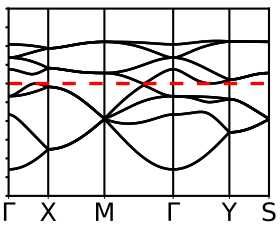
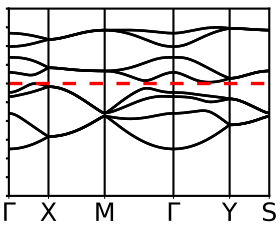
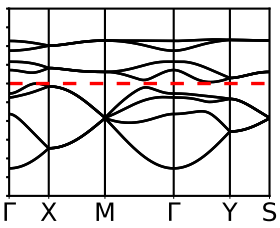
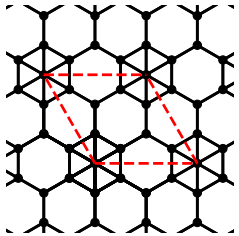
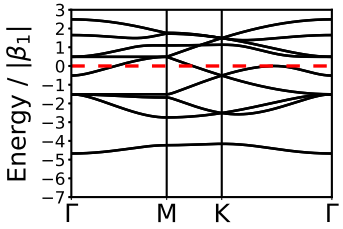
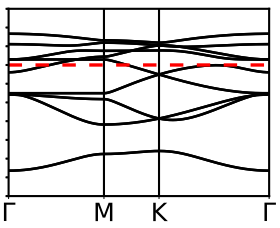
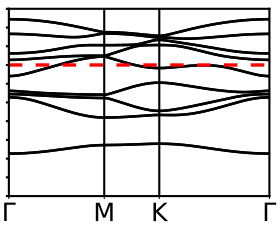
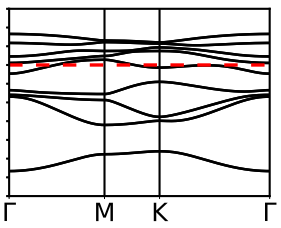
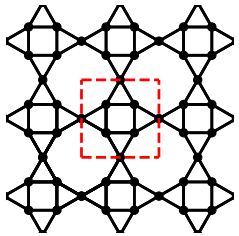
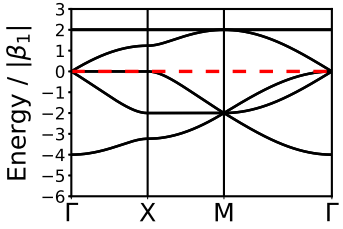
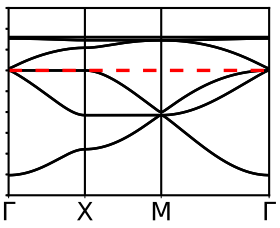
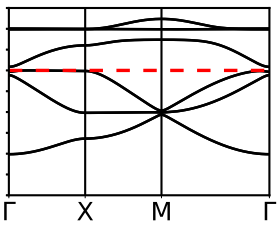
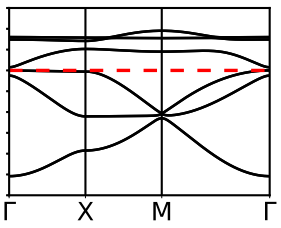
Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
cpm					
cpn					
cpo					
cpp					

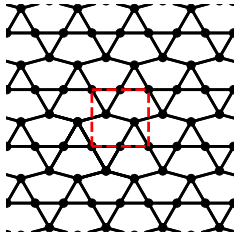
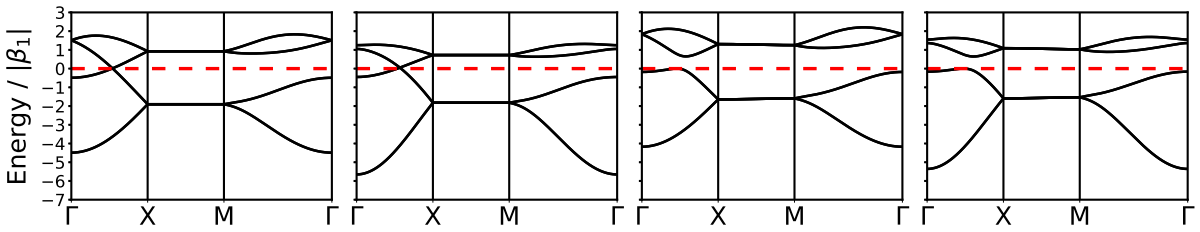
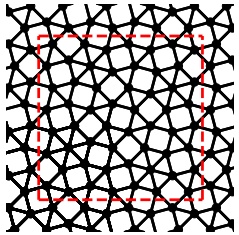
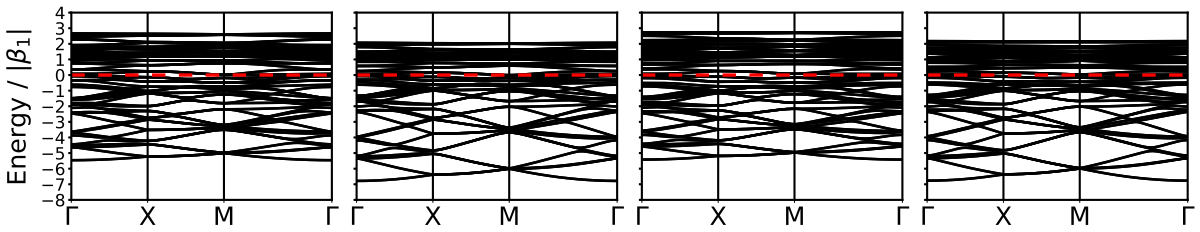
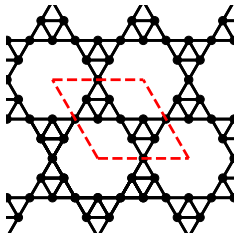
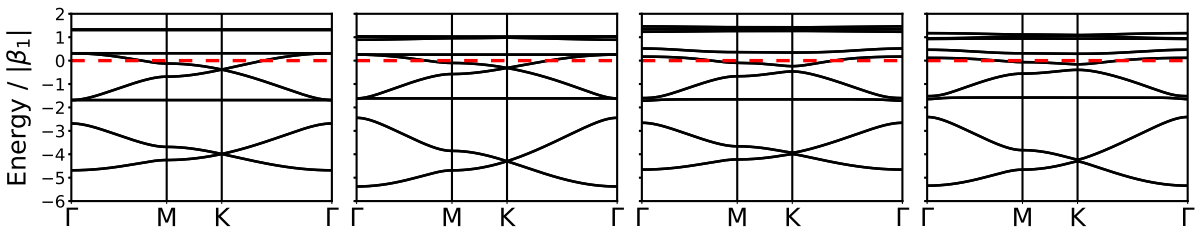
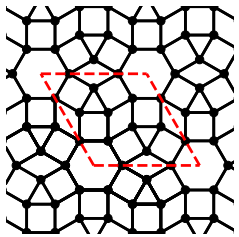
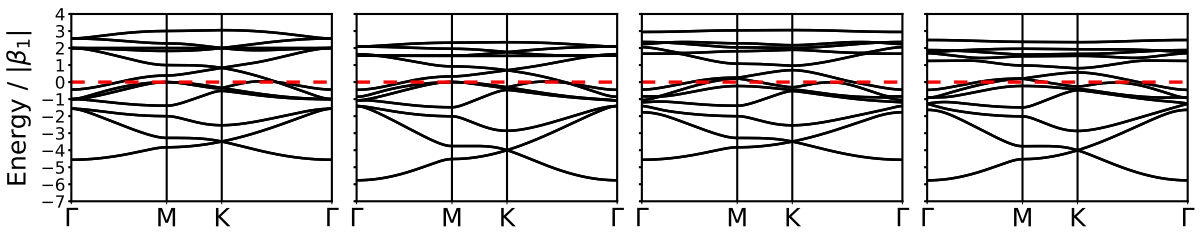
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cpq					
cpr					
cps					
cpt					

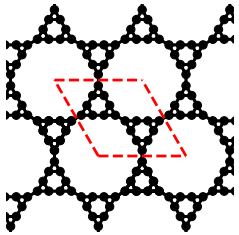
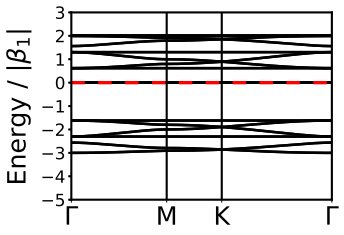
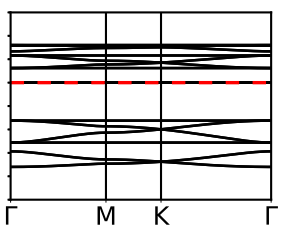
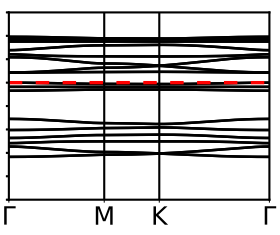
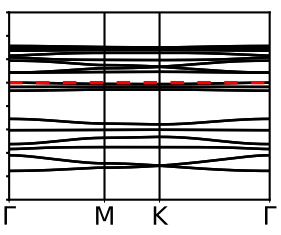
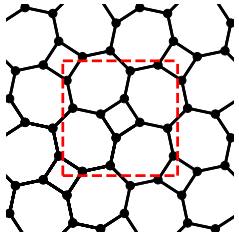
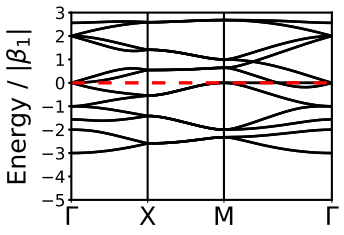
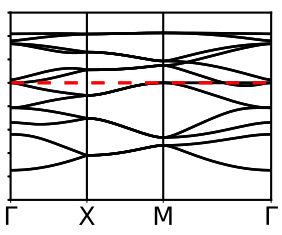
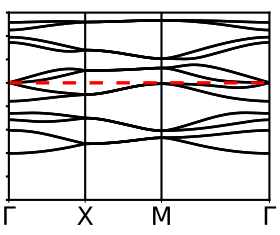
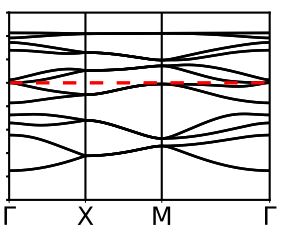
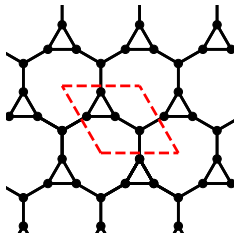
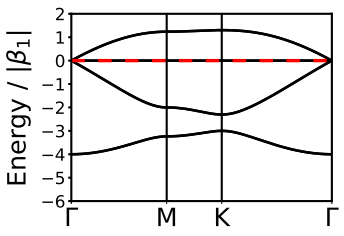
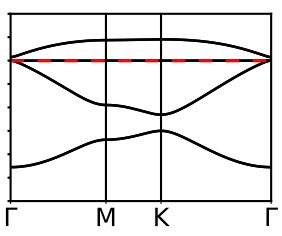
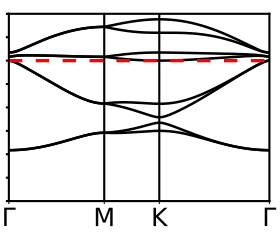
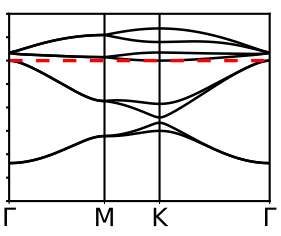
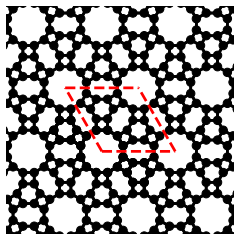
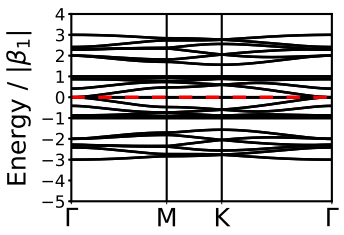
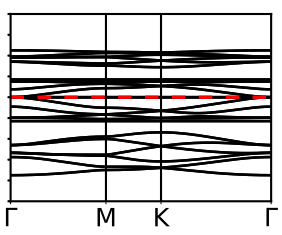
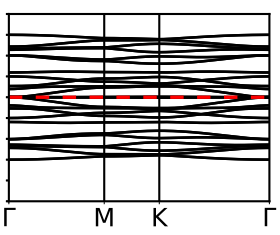
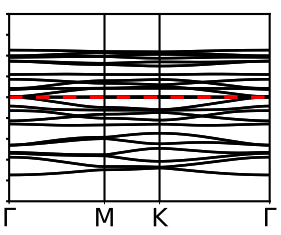
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cpu					
cpv					
cpw					
cpx					

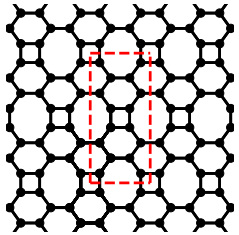
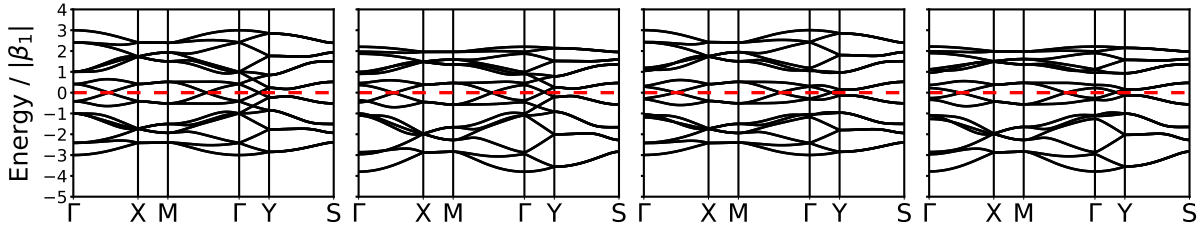
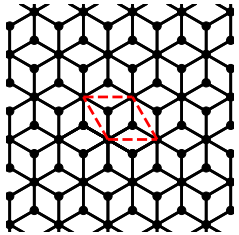
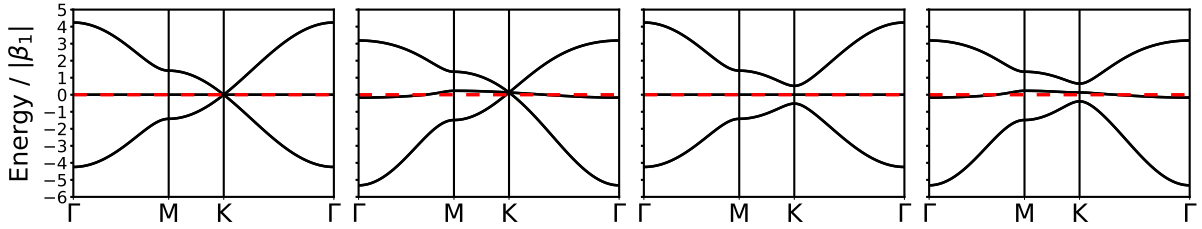
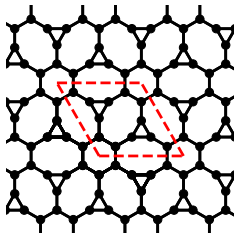
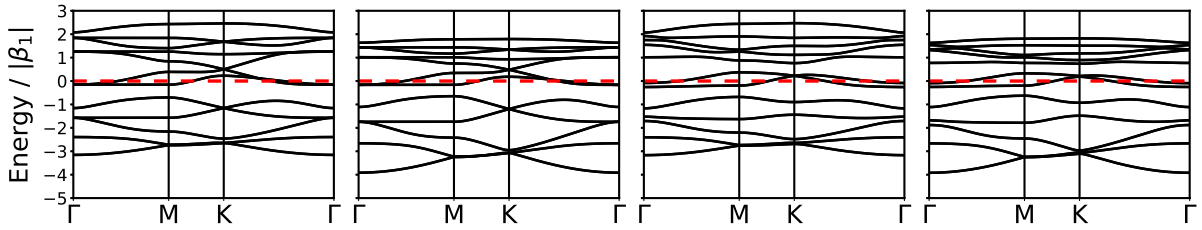
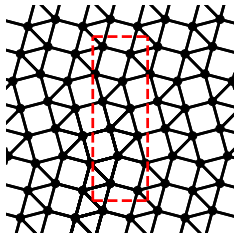
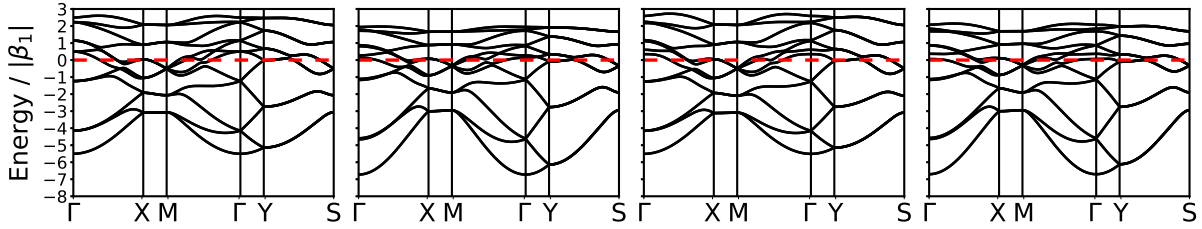
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cpy					
cpz					
cqa					
cqb					

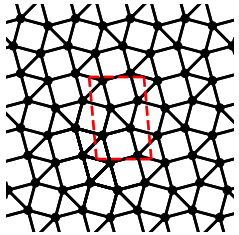
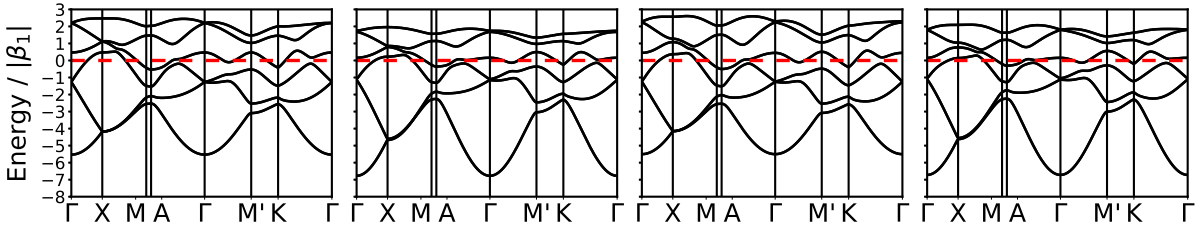
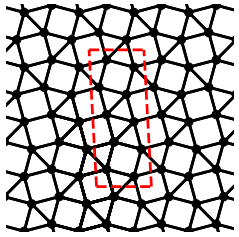
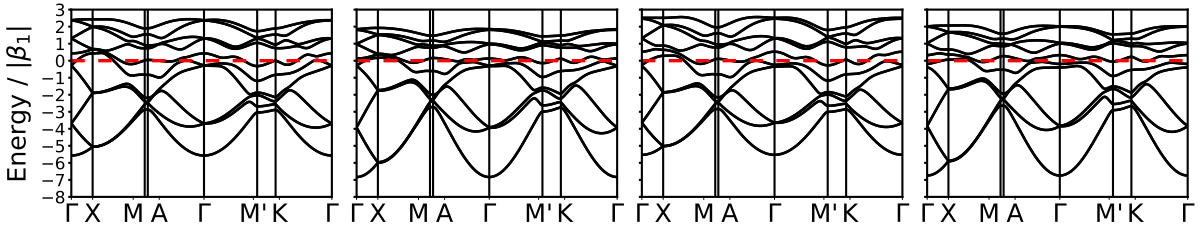
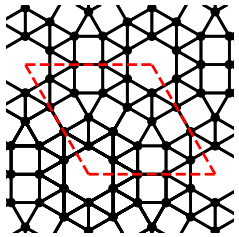
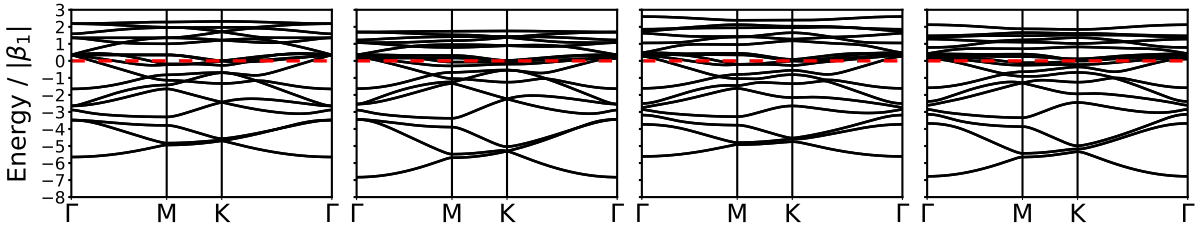
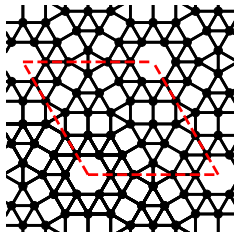
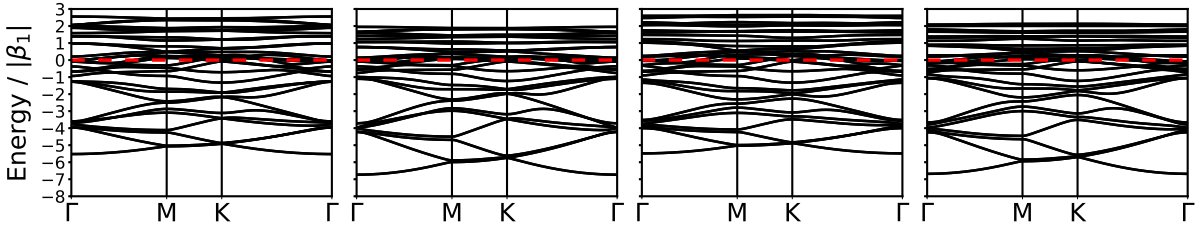
Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
cqc					
cqd					
cqg					
cqp					

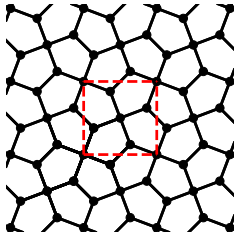
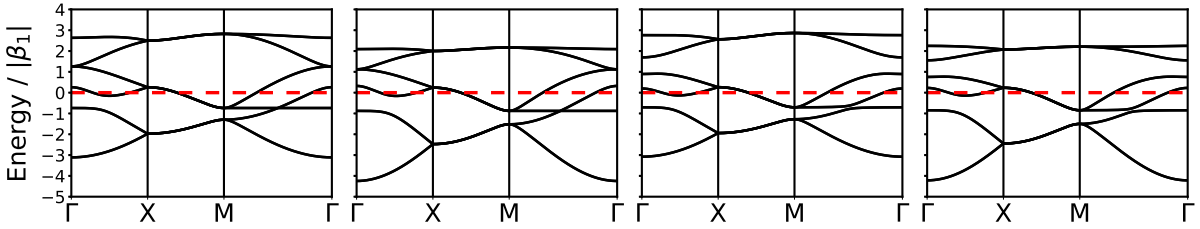
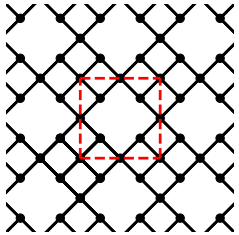
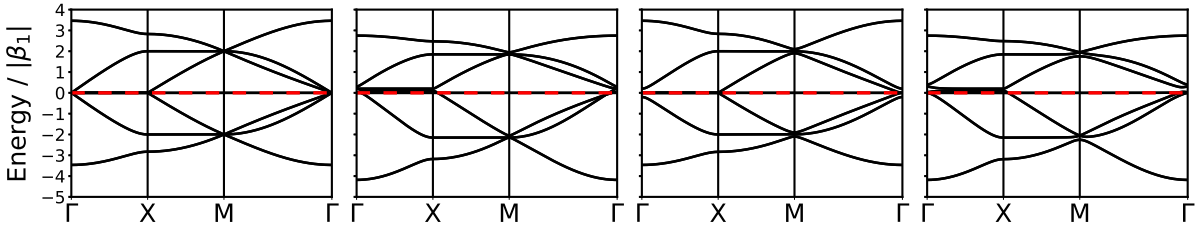
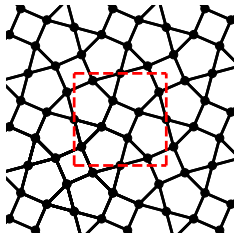
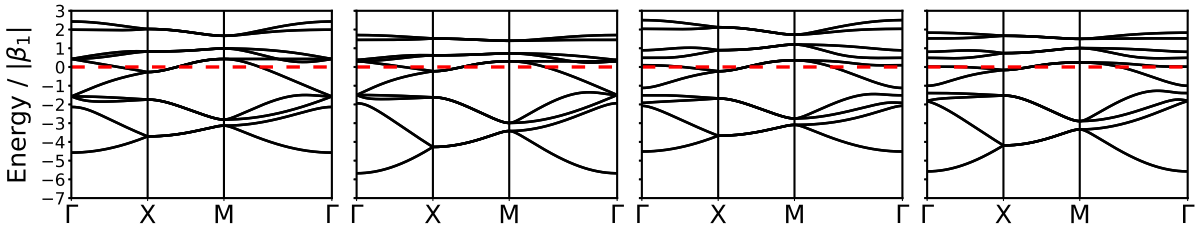
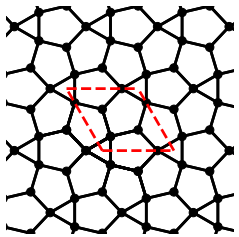
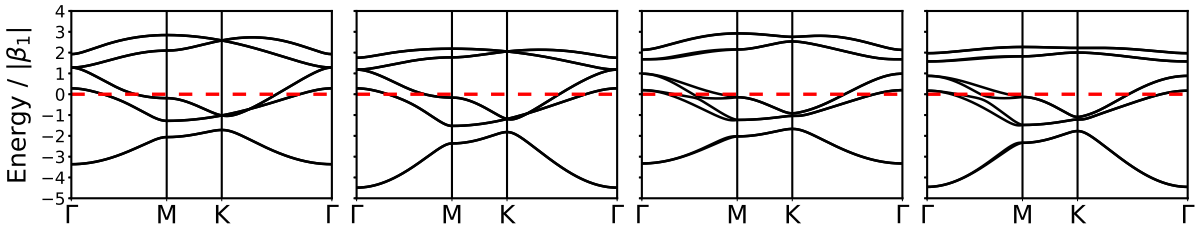
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
cqr					
cru					
crz					
dha					

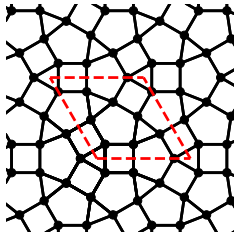
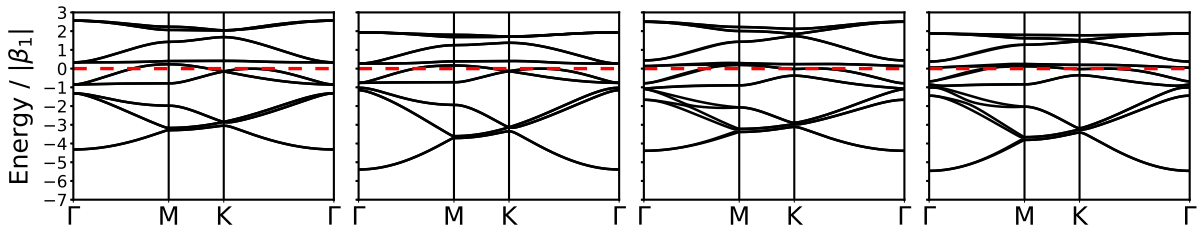
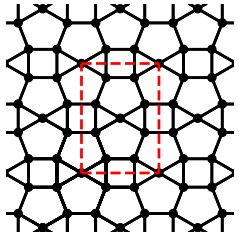
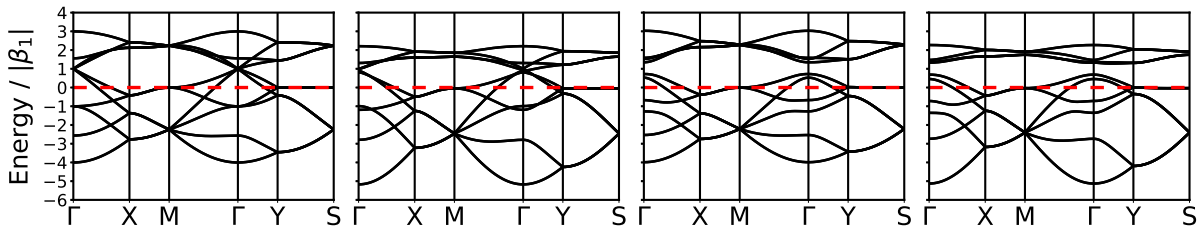
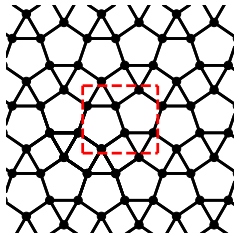
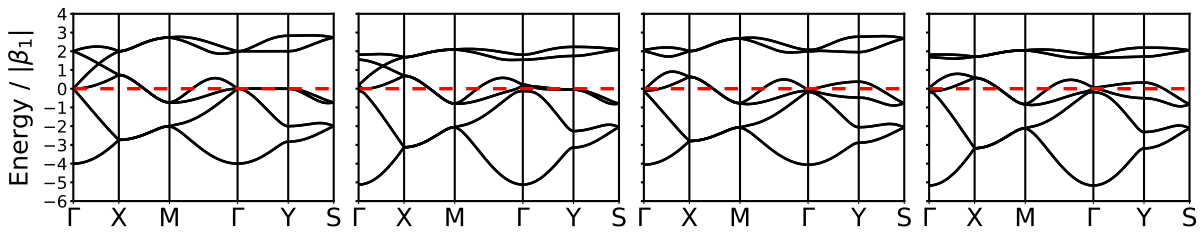
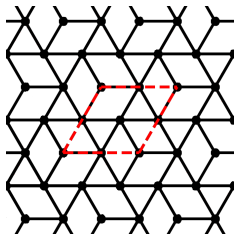
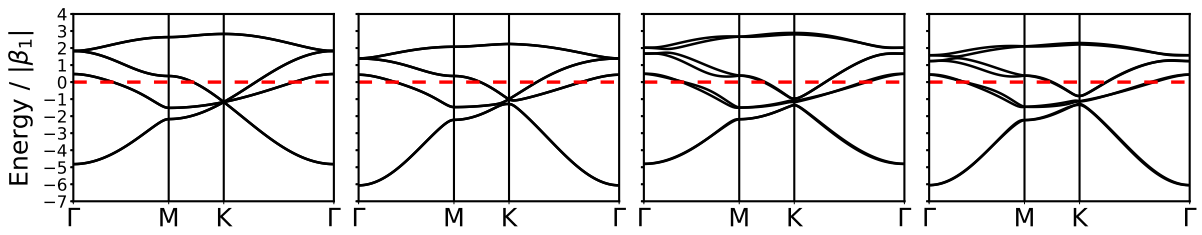
Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
dhb					
esq					
fwb					
fwe					

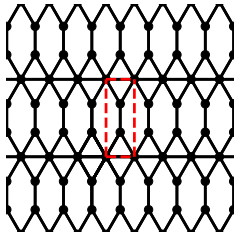
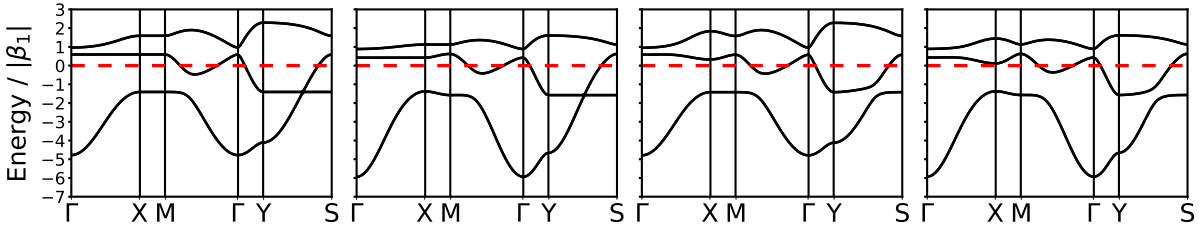
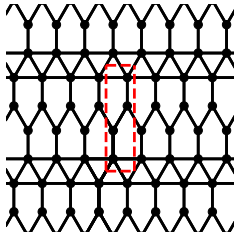
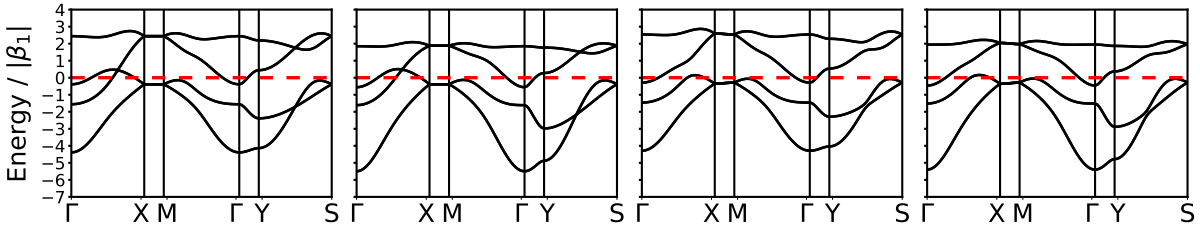
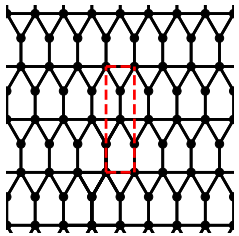
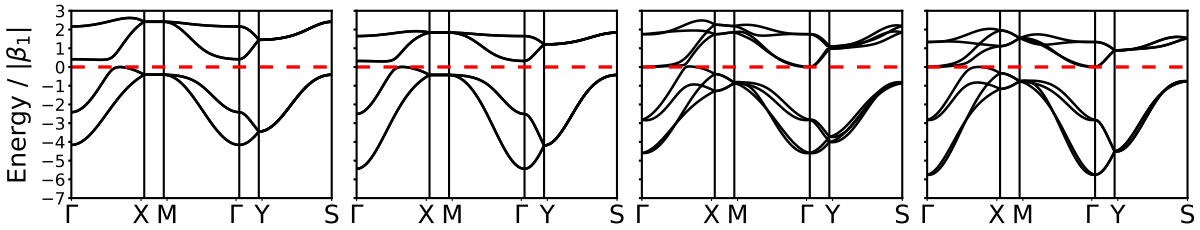
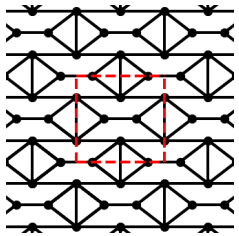
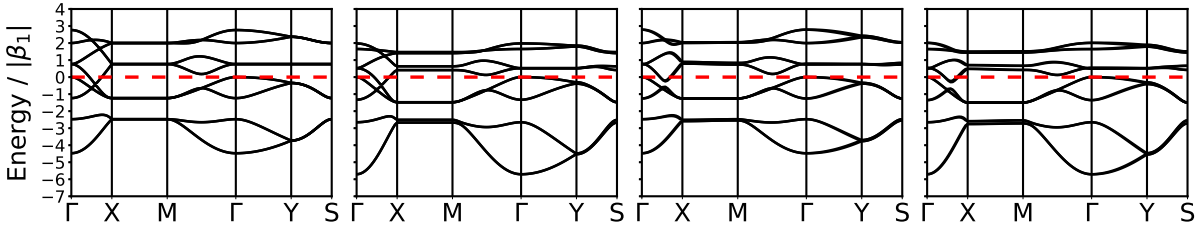
Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
hca-a					
hna					
hnb					
htb-a					

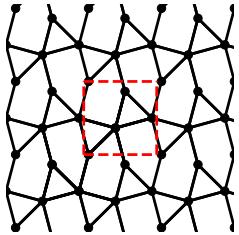
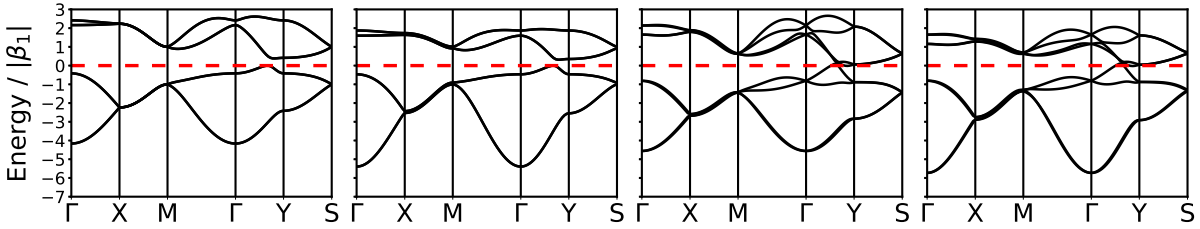
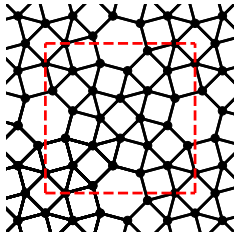
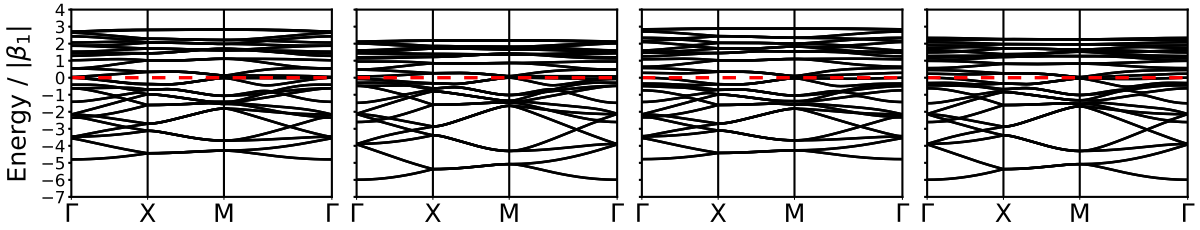
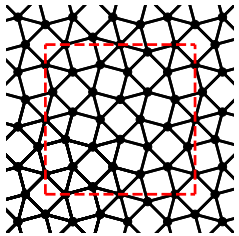
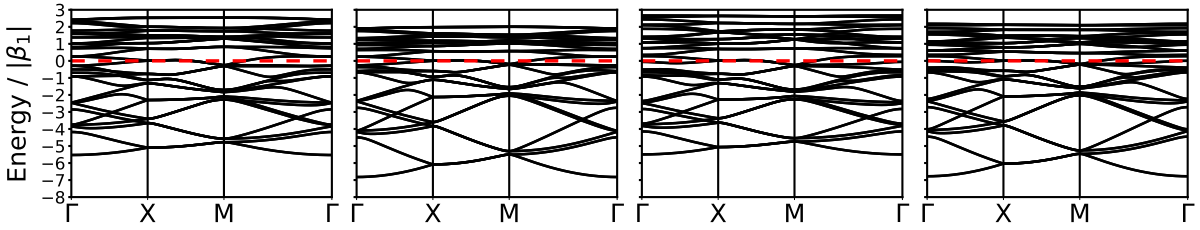
Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
jvh					
kgd					
kgd-a					
kru					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
krv					
krw					
krx					
kry					

Topology	Sketch	1st Neighbour	1st + 2nd Neighbour	1st + SOC	1st + 2nd + SOC
mcm					
mtf					
pnb					
pnd					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
pne					
pnf					
png					
pnh					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
sdb					
sdc					
sdd					
sde					

Topology	Sketch	1^{st} Neighbour	1^{st} + 2^{nd} Neighbour	1^{st} + SOC	1^{st} + 2^{nd} + SOC
sdf					
stz					
suz					

2 Technical tutorial for tight-binding calculations

In this review, we have used the `pythTB` package, which can be downloaded from http://www.physics.rutgers.edu/pythtb/. Here, we explain the elementary steps to perform a tight-binding (TB) calculation with the example of the honeycomb (**hcb**) lattice (this tutorial is written for `python` 3.6 and `pythTB` 1.7.2). For setting up the calculations, a python script with the following content is needed:

1. Load necessary modules:

   ```
   from __future__ import print_function
   from pythTB import *
   import numpy as np
   ```

2. Define lattice`lat` and fractional coordinates of sites `orb`:

   ```
   lat = [[1, 0], [0.5, np.sqrt(3)/2]]
   orb = [[1/3, 1/3], [2/3, 2/3]]
   ```

3. Initialise the `pythTB` class:

   ```
   my_model = tb_model(dim_k, dim_r, lat, orb, nspin=2)
   ```

   Here, `dim_k` and `dim_r` define the dimensionality in reciprocal and real space, respectively. For the **hcb** lattice, `dim_k = 2` and `dim_r = 2` are correct settings corresponding to both a two-dimensional real space and a two-dimensional reciprocal space. `nspin = 2` enables spin-polarised calculations, while the default `nspin = 1` is the setting for spin-restricted (diamagnetic) systems.

4. Setting on-site energies α for each site as a list with the same ordering as in `orb`:

   ```
   my_model.set_onsite([alpha, alpha])
   ```

   For different chemical potentials, different values for α are possible. Herein, we use $\alpha = 0$.

5. Setting hopping integrals β:

```
my_model.set_hop(beta, i, j, [A, B])
```

Hopping integrals need to be specified one by one. In this function, β is the hopping integral for interaction between vertex `i` in the first cell and vertex `j` in the cell replicated by the translation vector `R[A, B]`. Where `A` and `B` are in units of the respective lattice vector, moving from the first cell to the replicated cell. For first-neighbour hoppings within the first cell, `A` and `B` can be set to `0`. In `pythTB`, symmetric hoppings are set automatically. `i -> j + R[A, B]` and `i -> j - R[A, B]` are set simultaneosly, and cannot be specified twice in the script. For example, if we have

```
my_model.set_hop(beta, 1, 0, [0, 1])
```

then there should be no `my_model.set_hop(beta, 0, 1, [0, -1])`, since this is the same translation vector in opposite direction. In addition, individual hopping elements, e.g. β', can be specified afterwards by setting the mode `add`:

```
my_model.set_hop(beta', 1, 0, [0, 1], mode='add')
```

In this case, the final hopping from site 0 to site 1 in the cell with translation vector $\vec{R} = (0, 1)$ would be $\beta + \beta'$. In the case of **hcb**, three hoppings have to be set:

```
my_model.set_hop(beta, 0, 1, [0, 0])
my_model.set_hop(beta, 1, 0, [0, 1])
my_model.set_hop(beta, 1, 0, [1, 0])
```

6. Setting the path in k-space which should be used for the band structure calculation:

```
(k_vec, k_dist, k_node) = my_model.k_path(path, nk)
```

With `path` being a list with the fractional coordinates of high-symmetry points in the Brillouin zone and `nk` the total number of calculated eigenvalues for this path. Returns `k_vec`, a list of the k-points, for which the eigenvalue problem should be solved. `k_dist` is the accumulated distance of each k-point to the first k-point. `k_node` is the distance of high-symmetry points to the first k-point.

These are, therefore, selected values from `k_dist`. For hexagonal cells, the setting

```
path = [[0.0, 0.0], [0.5, 0.5], [2/3, 1/3], [0.0, 0.0]]
```

will return a band structure along Γ-M-K-Γ. Additionally, the labels for these high-symmetry points can be given:

```
label = [r'$\Gamma$', 'M', 'K', r'$\Gamma$']
```

7. Start the calculation and return eigenvalues for each k-point in `k_vec`:

```
evals = my_model.solve_all(k_vec)
```

8. Plotting can be done by, e.g., using `matplotlib`, which requires to load the package *via* `import matplotlib.pyplot as plt`. Band structure plots can be created by the following:

```
[plt.plot(k_vec, eigenvalue, color='black') for eigenvalue in evals]
plt.xticks(k_nodes, labels)
plt.ylabel('Energy')
plt.show()
```

9. Additional notes:

   a) In order to use the formalism given by Kane and Mele[2] to take spin-orbit coupling (SOC) into account, one needs to use the spin-polarised Hamiltonian (thus, `nspin = 2`) and to manually define the Pauli matrix σ_z:

   ```
   sigma_z = np.array([0, 0, 0, 1])
   tsoc = -1.j*Lambda*sigma_z
   ```

   With a SOC strength of λ. Note that for the Haldane model[3] is possible for a spin-restricted calculation:

   ```
   tsoc = Lambda*np.exp((1.j)*np.pi/2)
   ```

   Note that if there are two equivalent paths electrons could use to travel from a site to the second neighbour, the effect of SOC cancels out, e.g., two ver-

tices laying diagonally in a square show no spin-orbit contribution. Since the second neighbour hopping includes both the normal hopping β_2 and the SOC contribution, two lines per second neighbour and the mode `add` can be used to assign the hopping and SOC:

```
my_model.set_hop(beta_2, 0, 0, [1, 0])
my_model.set_hop(tsoc, 0, 0, [1, 0], mode='add')
```

b) Calculation details can be printed by `my_model.display()`.

c) In order to estimate the Fermi level, it is necessary to integrate over the whole Brillouin zone. In a k-grid of N_G points, each grid point contributes $1/N_G$ electrons ($2/N_G$ for spin-restricted calculations). The bands are occupied from the lowest to higher bands, until the number of electrons matches the number of electrons in the cell (see Section 3.1 and the example input file).

d) Obtaining the Chern number and Z_2 invariant:
The `pythtb.wf_array` class calculates the Berry phase and related properties. After setting up the Hamiltonian, this class can be initialised with

```
my_array = wf_array(my_model, [nk, nk])
```

with `nk` being the number of grid points per axis.

```
my_array.solve_on_grid([x, y])
```

solves the model in a regular grid with the origin being at the point `[x, y]`. The Chern number equals the Berry phase

```
berry = [my_array.berry_flux([i]) for i in range(len(my_array))]
```

divided by 2π (with `i` being the band index). It is worth noting that for calculating Chern number/Berry phase, a spin-restricted calculation is required. In order to get the Z_2 invariant, the Wannier Charge Centres (WCC) have to be calculated. An example for this is given on the `pythTB` website.[4]

References

1. http://www.physics.rutgers.edu/pythtb/, accessed 11 February 2020.

2. C. L. Kane and E. J. Mele, *Phys. Rev. Lett.*, 2005, **95**, 226801.

3. F. D. M. Haldane, *Phys. Rev. Lett.*, 1988, **61**, 2015.

4. http://www.physics.rutgers.edu/pythtb/examples.html#kane-mele-model-using-spinor-features, accessed 11 February 2020.