SurfGraph Documentation

Release 1.0

Tristan Maxson, Siddharth Deshpande, Jeffrey Greeley

Dec 15, 2022

Contents:

CHAPTER 1

Installation Instructions

The first step will be to obtain the code either via pip or directly on gitlab.

1.1 Pip

```
pip install surfgraph
```

1.2 Gitlab

```
git clone git@gitlab.com:jgreeley-group/graph-theory-surfaces.git
```

1.3 Post-Installation

Next you should ensure the python module is importable and that you can find the executable scripts(example: compare-chem-env). If you used pip to install this should already be done, but if installing from source you will need to add the "surfgraph" directory to PYTHONPATH and the "bin" directory to PATH.

This is also a good time to ensure the tests pass correctly. The tests should be able to be run using the following command.

TODO: This is currently not implemented.

```
python -m surfgraph
```

CHAPTER 2

Surface Graphs: General Settings and Utilities

The general settings can be divided in the following categories:

- Graph generation settings
- Graph analysis settings
- Graph post-processing for different applications

2.1 Graph generation settings

These are settings that the user need to keep in mind, while attempting to generate a graph for a given system

- Neigborlist: To generate the graph, the package requires a neighborlist. We currently use the neighborlist generated using the Atomic Simulation Environment (https://wiki.fysik.dtu.dk/ase/ase/neighborlist.html). For the case of ase-neighborlist, the parameter of scaling the atomic radii and the skin factor are the required inputs.
- Atomic Radii Scaling: This number specifies how much the covalent radii of a given atom should be scaled. We have found a modest scaling of '1.1-1.2' to be sufficient for metal slabs.
- skin factor: This is an input used as an error factor for all the bonds. The default setting of 0.25 Angstroms is sufficient.
- Grid parameter: This parameter represents the number of periodic representations of the surface to make the surface graph. A grid setting of [1,1,0], will take '1' repetition of the unit cell in the +ve and -ve 'x' and 'y' directions respectively. The required grid size is dependent upon the size of the unit cell as well as on the chosen graph radius (described in the next bullet). Note however that large grids are undesirable as the scale poorly with the required computation, n representations in the grid increases the size of the graph by n^2.
- Radius: This parameter determines the number of nearest neighbor shells that are to be considered in a given chemical environment. As our algorithm is geared towards catalysis, this parameter for the case of

the graph constructed around a given adsorbate, will represent the number of neighboring surface shells that are desired by the user to capture the chemical environment of the adsorbate

- A radius of '0' represents only the adsorbate and the surface atoms that surround it. This radius, will help aid in finding the type of active site for the given adsorbate.
- Radius of '1' represents a graph with the atoms encompassing the active site as well as the nearest neighbors of the active sites as well.
- A 0.5 increament in the radii will capture any adsorbate atoms that are bound to surface nodes present in the graph.
- Node and Edge Properties: Graphs are highly versatile representations. An added benefit, which, makes them very useful for atomic scale simulations, are the node and edge attributes. For example, the distance between the surface atoms can be a useful edge property to store, or the coordination number of the surface site can be another important node property to store. These can be used to conviniently store important atomic information. A list of useful node and edge attributes is given below:
- Node Properties:
  - Coordination number
  - D-band center
  - Electronegativity
  - number of valence electrons
- Edge Properties:
  - Atom-atom bond distance
  - Type of bond: surface-surface, surface-ads, ads-ads, H–bond (some of this is captured in the "dist" parameter above)
  - bond energy of a given bond: this can be used to store entities like ads_energy between ads and surface atoms, vacancy formation energy between surface-surface atoms.

2.2 Graph analysis settings

After generating a full graph, and the subsequent chemical environments for all the adsorbate atoms, the generated graphs can be used in following ways:

- Unique chemical environments: For the case of complex adsorbate geometries such as high coverages of adsorbates and multi-dentate adsorbates, the graph representing the chemical environments of adsorbates can be used to systematically find unique chemical environments. Consider the case of 3 mono-dentate NO adsorbates on the surface. To estimate if this particular high coverage configuration is unique when compared to other 3 NO configurations, the chemical environments for each of the NO adsorbate in the case of interest, will be compared with the chemical environments of other cases having 3 NO. For the configuration to be unique, atleast one chemical environment of the three should be different for all the 3 NO cases in consideration. ** TODO: PUT A FIGURE TO EXPLAIN **
- Different settings for uniqueness: The most important setting for determining the uniqueness, with the above rule is the radius of the chemical environment considered. As explained in 'Graph generation settings' section above, the radius is the number of shells considered around the adsorbate. Therefore, larger the number of shells, the higher the number of interactions that are considered. Thus, the radius can in principle be large enough such as 4-5, such that for a given adsorbate, it captures all the atoms present on the surface model and can be used to estimate global minimum configurations, for that given unit cell. However, we have found that adsorption energies are not affected too much outside of 2-3 shells and that is the standard setting.

2.3 Graph post-processing for different applications

This section explains some of the settings that we have found useful for different applications.

- Evolutionary Algorithm: One way to systematically consider high coverages of intermediates is by using an evolutionary algorithm. The algorithm has a simple rule of only populating certain number of unique configurations for a particular coverage for the next high coverage configuration. The parameter of importance here, will be how many of the unique configurations are needed to be used to get a robust sampling to find global minimum for a given coverage. This is a case dependent parameter. In the case of strongly bound adsorbate, we found that considering configurations inside a small energy cutoff of 0.25 eV from the most stable case was sufficient. This selection was sufficient as the strong binding nature ensured that surface-adsorbate interaction was the dominant factor to consider, which helped screen next generation of coverages, using on a small subset of previous generation, which had all the relevant important interactions captured. However, in the case where there are competing interactions, such as surface-adsorbate interactions compete with adsorbate-adsorbate interactions, this choice of energy cutoff should be scrupoulously tested. One way to do this, is to plot energy histogram as a function of energy cutoff, this will aid in understanding the importance of certain kinds of interactions for a given system.
- Setting to consider while considering high coverage configurations: Presented here is a list of settings, that we have found useful to systematically find high coverage configurations for different kinds of surface models:
  - Ads-Ads distance: This setting is very system dependent and depends on the nearest distance of two adjacent adsorption site. For example on Pt3Sn(111), this distance is of the order of 2.4 Angstroms, while on a Pt(100) surface, which is more open, a cutoff of about 2 Angstrom is sufficient, as the sites are inherently farther apart.
  - Explicitely constraining non-surface layers on non-terrace models: On non-terrace models like steps and kinks, we have found that the algorithm most efficiently works if the non-surface atoms (starting from the second layer) are constrained. This helps as the normals method used to find surface atoms, can fail in defected surfaces, as some sub-surface atoms can have non-zero normal components, especially post-relaxation.

CHAPTER 3

Usage

After installation of surfgraph, you should be able to access two command line tools under surfgraph/bin/.

3.1 analyze_chem_env.py

This tool will analyze geometries for uniqueness or to simply visualize the resulting graphs. For example, to analyze a set of Pt111 surfaces with CO adsorbates you might use the following command.

```
analyze_chem_env.py -unique */OUTCAR
```

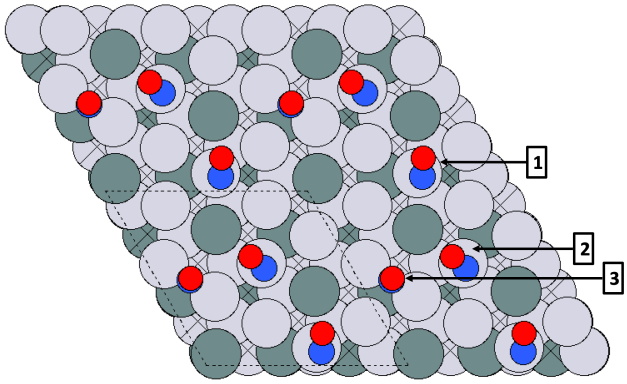
This will read all outcars in the folder using the defaults and comparing them for uniqueness. This will generally pick a reasonable radius and grid size for most calculations on an FCC surface. You can view the same graphs that are used by adding the "-view" argument, but this can be very confusing for radius > 2 unless you are working with a 2D material.

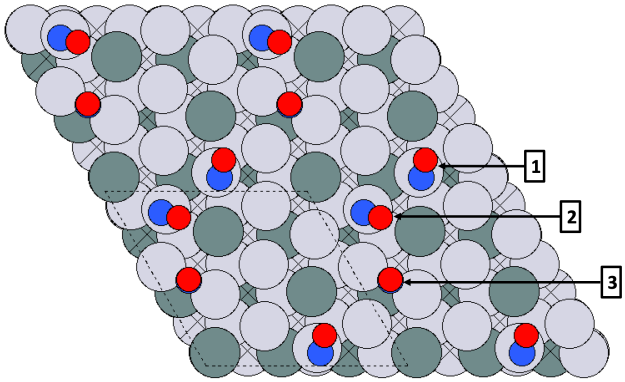
The following are options you should consider checking since the defaults may not be sufficent.

- "–radius 2": This corresponds to the graph radius for the chemical environments. This can be thought of as the number of coordination shells which should be considered. This can be tested for convergence by doing calculations on a large representative surface where one adsorbate is translated away from another and seeing how the binding energy changes. If a seperation of 3 shells is required before the energy stops changing, this is a good starting place for radius.
- "–grid '2,2,0'": This corresponds to a XYZ pair for the repetitions of the cell along periodic boundaries. This has large performance penalties if raised more than needed. As long as very small cells are avoided, a value of "2,2,0" works well for 2D periodic calculations. If sites which are clearly equivalent are not based on translational symmetry, then a grid problem is likely.
- "–adsorbate-atoms 'C,N,O,H'": This corresponds to which atoms can be considered an adsorbate. The default assumes that adsorbates will be formed from C, O, N, and H atoms. This will have to be changed when using surfaces which contain these atoms or when using adsorbates which don't include these atoms. More advanced ways of specifying which atoms belong to the surface can be implemented.

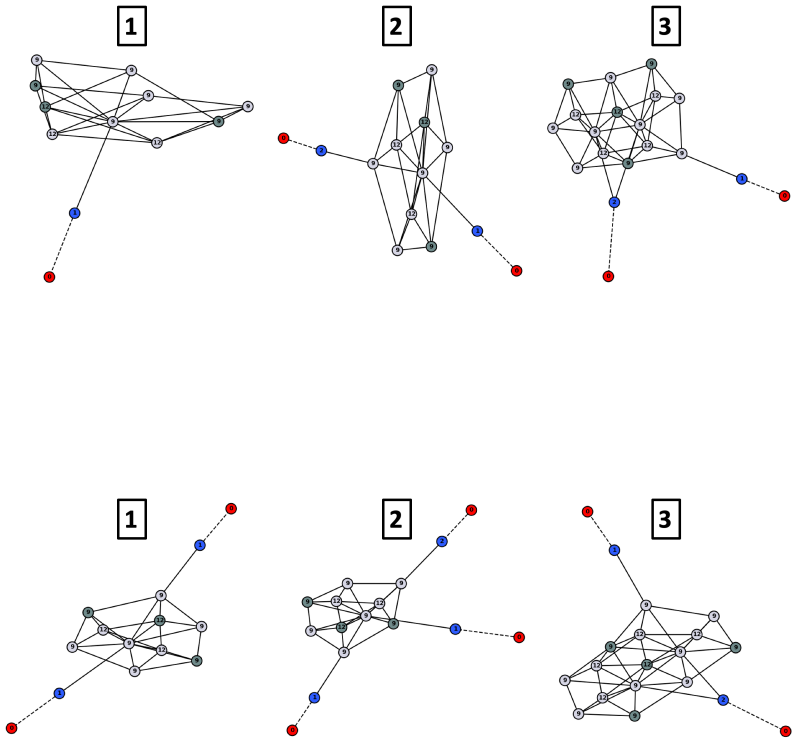
- "–mult 1.1": This corresponds to the covalent radius of the atoms. In general, this can be raised if bonds are not being found that are expected or lowered if bonds are being found that are not expected. This value is a scaling factor for the covalent radius
- "–skin 0.25": Similar to mult, this value allows for bonds to be extended by a fixed amount. This may be helpful as an additional tuning tool.
- "–view": Enabling this flag causes the code to visualize the graphs using matplotlib
- "–unique": Enabling this flags does a uniqueness analysis and outputs the duplicates along with their energies if available.

Shown below is an example, with 2 high coverage NO configurations:





Note that in both cases, the NO are adsorbed in a bridge and two top sites. The analyze_chem_env.py can be used to find that these two cases are unique, as in first case the two NO's are adsorbed on adjacent sites and they are on opposite sites in the second case. Further shown below are the graphs used to compare the chemical enivornment:



Note that in the first case, one NO molecule is isolated with no other NO adsorbed in adjacent sites. This makes the graph for this case have only 1 NO, while all the NO's for the second case have atleast 1 NO adjacent. These kinds of graphs are compared to each other to find unique chemical environments and subsequently find unique adsorbate configurations.

3.2 generate_sites.py

This tool will find unique sites, generate normal vectors for them, then adsorb an adsorabte into that site. The adsorbate is a seperate input geometry where the adsorbate is aligned along the Z axis. The atom to bind into the site should be found at (0, 0, 0) and the adsorbate will be rotated to align with the normal of the site.

```
generate_sites.py --view NO.POSCAR Pt111.POSCAR
```

This will adsorb a NO molecule (assuming NO.POSCAR exists) into all found sites on the Pt(111) surface. This can be heavily tuned with the following settings.

- "–radius 2": This corresponds to the graph radius for the chemical environments. This can be thought of as the number of coordination shells which should be considered. This can be tested for convergence by doing calculations on a large representative surface where one adsorbate is translated away from another and seeing how the binding energy changes. If a seperation of 3 shells is required before the energy stops changing, this is a good starting place for radius.
- "–grid '2,2,0'": This corresponds to a XYZ pair for the repetitions of the cell along periodic boundaries. This has large performance penalties if raised more than needed. As long as very small cells are avoided, a value of

"2,2,0" works well for 2D periodic calculations. If sites which are clearly equivalent are not based on translational symmetry, then a grid problem is likely.

- "–adsorbate-atoms 'C,N,O,H'": This corresponds to which atoms can be considered an adsorbate. The default assumes that adsorbates will be formed from C, O, N, and H atoms. This will have to be changed when using surfaces which contain these atoms or when using adsorbates which don't include these atoms. More advanced ways of specifying which atoms belong to the surface can be implemented.
- "–mult 1.1": This corresponds to the covalent radius of the atoms. In general, this can be raised if bonds are not being found that are expected or lowered if bonds are being found that are not expected. This value is a scaling factor for the covalent radius
- "–skin 0.25": Similar to mult, this value allows for bonds to be extended by a fixed amount. This may be helpful as an additional tuning tool.
- "–min-dist 2": This corresponds to the minimum distance allowed between adsorbates. If adsorbates would be placed closer than this, they will be rejected.
- "–no-adsorb '': This corresponds to a list of elements which cannot be adsorbed to. This is helpful when prior knowledge lets you know that a specific adsorbate cannot bind to a specific element effectively.
- "–coordination '1,2,3'": This corresponds to a list of coordinations which can be considered for absorption. Currently the code only works for top, bridge, and hollow sites but this will be expanded in the future.
- "–output POSCAR": This tells the code what file extension is requested for output of files. If this is omitted, then no files will be output.
- "–output-dir .": This tells the code what folder it should output its results into. If this is omitted, then files will be output in the current working directory.

For example, if we only wanted to adsorb molecules to the top sites of Ni atoms in a NiCu alloy, we could do the following.

```
generate_sites.py --output POSCAR --output-dir top-sites-Ni --coordination "1" --no-
↪adsorb "Cu" NO.POSCAR NiCu111.POSCAR
```

CHAPTER 4

Advanced Usage

While there is a goal of providing command line tools to perform this work, for more advanced tasks or custom properties our provided command line tools can serve as an example of how to script these tasks. The analyze_chem_env.py file serves to demonstrate the chemical_environment module and the generate_sites.py file serves to demonstrate the site_detection module. This may be useful when automation or high throughput calculations are required which can be optimized or run in parallel.

CHAPTER 5

Surface Graphs: Theory

Surface graphs are made to accomplish the following tasks:

- Unwrap the full connectivity for a surface type atomic object into a repeating non-periodic graph to simplify further analysis and make visualization easier.
- These graph representations can then identify unique locations on the graph as a function of the chemical environment of the given site. This can then be used:
- To identify unique starting positions for an adsorbate on a given graph.
- To identify unique adsorbate configurations amongst a set of configurations.

These two capabilities can be synergistically utilized to enumerate high coverage configurations, as well as, for finding unique configurations post - simulations.

5.1 Full Surface Graph

Initially, a surface graph must be created that encompasses the entire unit cell.

First, nodes are added for each atom within the cell as well as repetitions in a user specified grid (grid) in order to unwrap periodic boundary conditions. 'grid' dictates how many periodic repetitions are desired, though we have found that a large grid is not favorable from a computing stand point, and for cell sizes greater or equal to a 3x3, 1-2 repetitions are sufficient. Then edges are added for each bond ("bond") and the bonds are labelled in the following format with the elements in alphabetical order: AB (Example: PtSn). Edges are also assigned two distances for chemical environment analysis. The distance ("dist") is defined as 0, 1, or 2 based on how many surface atoms are involved as well as an extra distance ("ads_only") which is either 0 or 2.

- If the bond is between ads-ads, both distances are set to 0.
- If the bond is between surface-surface, both distances are set to 2.
- If the bond is between ads-surface, "dist" is set to 1 and "ads_only" is set to 2.

TODO: Review if "ads_only" is actually needed.

TODO: Tristan explain how the radius of the graph makes use of this parameter, and how 1.5 captures, the 1st shell surface atoms and then only the ads attached to them

5.2 Chemical Environments

The chemical environment is defined as the ego graph with a given radius which is representative of the number of surface coordination shells that should be captured. A shell radius of 0 will capture only the adsorbate and its direct connections to the surface. A shell radius of >1 will capture that many coordination shells on the surface (r=1, and dist=1 is allocated to surface - ads bonds). This is implemented by taking and ego graph from some atom of the adsorbate with a scaled radius. The graph radius is given as twice the shell radius plus one (r = 2*r_s +1). The user inputs 'r_s' as a variable and this results in the following behavior.

1. First the entire adsorbate is captured for free since the distance is given as 0.
2. The plus one of the radius will then capture the surface atoms bonded to the adsorbate (r=1, and dist=1). This should result in an even radius (r) leftover (as a result of r=2*r_s+1) for capturing shells on the surface. Therfore a radius (r_s) of '0' will provide the adsorbate atoms + the surface atoms surrounding the adsorbate.
3. Surface atoms will then be captured by decrementing the radius (r) by 2 each time (as 'dist' for surface-surface bond is set to 2). This should always result in an even number of leftover radius (r).
4. If another adsorbate is found, the leftover radius is decremented by 1, the entire adsorbate is captured for free, then it is decremeneted by 1 again as it captures all the surface atoms. This also results in an even number of leftover radius. To capture, just the surface atoms present in the active site, along with its neighbors and the attached adsorbates, a radius 'r_s' of 1.5 should be used. In this case, r = 4 (r=2*r_s+1), this will capture the ads-surface bonds first (dist=1), then the surface atoms surrounding the surface atoms in the active site (dist=2) and then finally all the adsorbate atoms attached to the neighbors identified in the previous step (dist=1). Therefore a total dist of '4' is captured.

Then a subgraph of the full graph is taken that contains all of the nodes found in the ego graph. This effectively ensures that all edges between all found nodes are captured, even if they were not traced out directly in the graph. This prevents some dangling bonds at the edges of the nodes and gives some additional information that we have found to be useful, otherwise an extra shell is required to find roughly the same results.

Knowing what an appropriate radius is depends on the system and should be tested. For example, we have found that in FCC metals that a radius of 2 is often sufficient and not much effect is seen by going above 3. Though this choice can be very system dependent, and some common heuristics can be found on the settings page.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search