MEPSAnd 1.7 user manual

Iñigo Marcos-Alcalde

November 2025

Contents

1 Introduction **3**

2 Availability and contact **4**

3 Citation **5**

4 Changelog **6**

5 Requirements and Installation **8**

6 The MEPSAnd algorithm **9**
6.1 MEPSA and MEPSAnd . . . 10
6.2 Surface Connectivity Network . . . 11
6.3 Global Network . . . 11
6.3.1 Minima detection . . . 12
6.3.2 Minima propagation and annotation . . . 14
6.3.3 Barrier detection . . . 14
6.3.4 Building of minima and barriers network: the Global Network . . . 16
6.4 Path calculations . . . 16
6.4.1 Path reductions . . . 18
6.4.2 Alternative paths detection . . . 19
6.5 Well sampling . . . 20

7 MEPSAnd user guide **21**
7.1 Loading and saving sessions . . . 21
7.2 Surface loading . . . 23
7.3 Defining surface connectivity . . . 24
7.3.1 Using MEPSAnd built-in connectivity detection methods . . . 24
7.3.2 Loading precalculated connectivity . . . 28
7.4 Saving minimum and barrier clusters . . . 29
7.5 Path-finding . . . 30
7.5.1 Defining origin and target points . . . 31
7.5.2 Minimum energy path calculation . . . 32
7.5.3 Alternative paths calculation . . . 32
7.5.4 Well sampling calculation . . . 33
7.5.5 Saving fragmentwise paths . . . 34
7.5.6 Saving simplified paths . . . 35
7.5.7 Saving well sampling . . . 35
7.6 Index based selections . . . 36
7.7 Data plotting . . . 37
7.7.1 Energy profiles . . . 37
7.7.2 Coordinate projections . . . 38
7.7.3 Network projections . . . 40
7.8 MEPSAnd python scripting . . . 52

7.9 Examples 55
7.9.1 3 minima surface: a step-by-step introduction to MEPSAnd general features 55
7.9.2 Geneva-Turin: alternative paths and network projections on a complex 3D surface 68
7.9.3 Displaced sine cubes: a 4D surface 74
7.9.4 5D sines: more than one million points over 5 dimensions 77
7.10 Good practices 79
7.10.1 Metadynamics and partially sampled datasets 79

Introduction

MEPSAnd (Minimum Energy Path Surface Analysis over n-dimensional space) is a GUI-based tool for the analysis of n-dimensional energy surfaces from a transition state theory perspective.

MEPSAnd can i) calculate the path that passes through the lowest energy barrier/s connecting two given points, ii) the region of the surface that has to be sampled from the origin and target points to reach those barriers and iii) a series of alternative sub-optimal paths. To facilitate the interpretation of this n-dimensional data, MEPSAnd also offers a range of plotting and projection options.

To follow this manual a series of test n-dimensional surfaces are provided in the MEPSAnd downloads page:

http://bioweb.cbm.uam.es/software/MEPSAnd/

Availability and contact

MEPSAnd can be downloaded from:

http://bioweb.cbm.uam.es/software/MEPSAnd/

License **GPLv3 (See the LICENSE file distributed with MEPSAnd)**

Contact: imarcos@cbm.csic.es; pagomez@cbm.csic.es

Please feel free to send us any kind of feedback, bug report or feature request. This is the only way we have to detect end user needs, as MEPSAnd can be used to analyze an vast array of datasets with varying requirements that are not easy to predict. Check the "changelog" (4) and "Good practices" (7.10) sections to see some of these contributions.

Citation

If you use MEPSAnd in your research, please cite as follows, as we depend on public funding to keep the program supported:

Marcos-Alcalde, I., Lopez-Viñas, E. & Gómez-Puertas, P. (2019). MEPSAnd: Minimum Energy Path Surface Analysis over n-dimensional surfaces. Bioinformatics 36, 956–95.
http://dx.doi.org/10.1093/bioinformatics/btz649

Changelog

```
1.7
- Implemented bug fixes following feedback from John J. Low.
- Introduced updated installation instructions with constrained package versions.
- Enhanced support for osx-arm64 architectures on the installation instructions following
    feedback from John J. Low.
- Fixed a bug introduced in connectivity logic in 1.6.
- Fixed some minor GUI bugs.
- New streamlined connectivity load/save logic implemented.
- Switched to f2_modified, which supports python>=3.9 and has implemented size-aware node
    repulsion.
- Implemented customizable node repulsion network layout calculations.
- The display of graph plots is now handled by pillow.
1.6
- New "Assume grid" connectivity prediction mode that significantly accelerates the
    neighboring calculations for homogeneously distributed datasets (the most common use
    case).
- Fixed subtle bug with global node ids annotation that could produce an error when
    selecting non-minimum origin or target points and could block network plots from
    working with networks containing disconnected regions.
- Added a check for the existence of disconnected regions that reports a warning message.
- Added a check for coordinate redundancy to prevent the use of artefactual surfaces.
- Thanks to Dr. Marcelo D. Polâto's feedback, an automatic data type reduction function to
     reduce memory usage was implemented.
- Thanks to Dr. Marcelo D. Polâto's feedback, the connectivity calculations were rewritten
     to reduce memory usage.
- Changed the way fragmentwise path is saved to avoid extensive memory use when extremely
    complex paths are calculated.
- Major overhaul of the "Energy profiles" and "Coordinate projections" point labeling
    logic. Now only minimum or barrier points are labeled, indicating the cluster number
    to which they belong.
- Made it possible to plot "Coordinate projections" without choosing any point.
- MEPSAnd no longer requires igraph, ForceAtlas2 or cairo to run. If any of these packages
     is not detected, "Network projections" will be disabled but everything else will work
     as usual.
- Replaced "dype=np.int" instances with "dtype=np.int_".
- Added terminal report messages indicating the number of neighbors found as a quick
    sanity check for connectivity calculations.
- Added terminal report messages indicating minima propagation progress.
- Removed instances of fig.gca(projection='3d') as it is deprecated matplotlib.
- Instances of "map_" in the source code have been replaced with "surface_" to unify
    terminology.
- Fixed well sampling calculations not identifying origin or target clusters correctly.
- Minor code cleaning.
- A "Good practices" section has been added to the manual, with a particular focus on
    metadynamics, based on Dr. Marcelo D. Polâto's reports.
```

1.5
- Withdrawn release.
1.41
- Added the correct citation reference to MEPSAnd.py, replacing the previous "XXXX..." placeholder.
- Updated the citation reference in the manual, as the paper is no longer "in press".
1.4
- Number formatting in "save_simplified_path_to_txt" method has been modified to improve the readability of the resulting files.
1.35
- BUGFIX: a bug introduced during network projection rewriting in 1.3 broke the ability to properly assign default plotting parameters in "barriers as vertices" graphs.
1.3
- Major fix: network projection edge selection was not working as intended and has been completely rewritten.
- Minor bugfixes to the GUI.
- Minor code cleaning.
1.2
- BUGFIX: As reported by Oriol Esquivias, the method to save minimum and barrier clusters was broken when clusters of more than one point were present. Fixed using an index array ("index_array") with the adequate shape in the "np.column_stack" calls of the "save_minbar_clusters_to_txt" method.
- Changed how the MEPSAnd version is printed on the terminal when it is launched to improve version readability.
1.1
- BUGFIX: As reported by Oriol Esquivias, instances of pandas compat.StringIO could lead to incompatibility issues with pandas 0.25.1. Thus, all instances of pd.compat.StringIO have been replaced with io.StringIO.

Requirements and Installation

To obtain MEPSAnd, download the latest zip file from: http://bioweb.cbm.uam.es/software/MEPSAnd/

Once downloaded, extract the contained folder and look for a file named "MEPSAnd.py".

To run **MEPSAnd**, we recommend using a conda package manager such as micromamba. We offer a range of strategies to build the environment required to run MEPSAnd:

1. **Install as much as possible from conda-forge:**

   ```
   micromamba create -f environment.yml
   ```

2. **Rely on pip**:

   ```
   micromamba create -n mepsand_env -c conda-forge python=3.11
   micromamba activate mepsand_env
   pip install -r requirements.txt
   ```

3. **Rely on pip for all packages except pycairo:**

   ```
   micromamba create -n mepsand_env -c conda-forge python=3.11 pycairo=1.26.1
   micromamba activate mepsand_env
   pip install -r requirements.txt
   ```

This will create the basic MEPSAnd environment (here called mepsand_env) that will allow MEPSAnd to perform all the pathfinding tasks but will disable the "Network projections" functionality. To install the dependencies of the "Network projections" code run:

- `conda activate mepsand_env`
- `conda install -c conda-forge pycairo python-igraph fa2`

For more advanced python users, note that all these dependencies can be installed using pip, without requiring a conda package manager at all, but some, such as pycairo, may require to install some external libraries first, as those are not python packages that can be distributed by pip.

To run the program, open a terminal window pointing to where MEPSAnd.py is located, activate the mepsand_env environment:

- `conda activate mepsand_env`

And call the conda python interpreter using the path to the "MEPSAnd.py" file as argument:

- `python MEPSAnd.py`

The MEPSAnd algorithm

In 2015 we published MEPSA (Marcos-Alcalde et al. 2015. Bioinformatics 31, 3853-3855), which offered a GUI based tool for the analysis of 3D energy surfaces. It used a method that somehow resembled Dijkstra's algorithm (Dijkstra 1959. Numerische Mathematik 1, 269–271). After origin and target points were defined, propagation from target to origin was performed, occupying the lowest energy neighbor available on each iteration. The step on which each point was occupied was annotated and later used to perform a trace-back from origin to target, minimizing the annotated step counts.

MEPSAnd generalizes the classic MEPSA abstraction offering many new features, such as:

- Work with surfaces with any number of dimensions.
- Handling of data without the requirement of grids as long as point neighborhood can be defined.
- Consideration of all equivalent paths at once instead.
- Accurate handling of plateaus.
- Automatic detection of alternative sub-optimal paths.
- Data projections to facilitate the interpretation of n-dimensional data.
- Significant performance improvements allowing the analysis of surfaces with more than a million points in regular PCs.

MEPSAnd was written from ground up as the treatment of n-dimensional data required a completely different (and significantly more convoluted) algorithm. Still some of the abstractions implemented are based on those classic MEPSA was based on. In this section the key aspects of the MEPSAnd algorithm are presented. In order to analyze energy surfaces with an arbitrary number of dimensions in an exhaustive fashion, MEPSAnd applies two sequential levels of abstraction to any surface that is loaded by the program (Fig. 1):

- A "Surface Connectivity Network" defined by the connectivity derived by point neighborhoods (see section 6.2).
- A "Global Network" abstraction that only consists on minima and barrier clusters (see section 6.3).

Although the main focus of MEPSAnd is to offer an energy surface analysis tool that can work with any number of dimensions, for clarity purposes a simple 3D (2 coordinates and energy) surface will be used to illustrate how the algorithm works in an intuitive way. Further ahead, in the MEPSAnd usage chapter, examples of the analysis of more complex 3D, 4D and 5D surfaces are provided (see section 7.9).

6.1 MEPSA and MEPSAnd

The source code of MEPSAnd has been written from scratch. Although the general abstraction of energy-guided propagations followed by trace-backs used in MEPSA is still present in MEPSAnd, profound differences exist between both approaches. The most remarkable one is that MEPSAnd applies these propagations and trace-backs over a range of networks that can be built independently of the number of dimensions of a given surface (see section 6.4). This enhancement offers several benefits, the most relevant of which is the native handling of surfaces with any number of dimensions, in contrast with classic MEPSA that was only able to handle uniformly sampled 3D surfaces. However, as both programs can analyze 3D surfaces, it might be worthy to comment out some the improvements that MEPSAnd offers over the classic MEPSA in this regard:

- One of the most notable differences for previous MEPSA users is that MEPSAnd works with realistic minima. Such change allows to natively handle a flat minimum region as a single cluster of points, instead of treating each point as a different minimum on its own. Also, due to this new abstraction, there is no need to choose a type of node definition anymore.
- Classic MEPSA did not show more than one equivalent path in a single run, while MEPSAnd can handle solutions composed by any number of equivalent paths, while still retaining the ability of simplifying such solution to a single path via optional path reductions (see section 4.4.1).
- Also, previous users might remember that the classic MEPSA most accurate path-finding method, "node-by-node", could be computationally intensive on complex surfaces. By contrast, due to its network-based approach, MEPSAnd does not need different types of path-finding, as it inherently samples every minima the basin of which is visited by the computed path (which was the purpose of the "node-by-node" analysis).
- Classic MEPSA offered a built-in surface editor to allow the user to force the sampling of alternative paths by blocking with artificial barriers previous solutions. While this approach can be transferred to MEPSAnd by exporting surfaces with classic MEPSA editor, MEPSAnd can automatically search for a finite number of alternative sub-optimal paths, forcing the sampling of increasingly unfavorable saddle points.
- Another benefit of MEPSAnd network-based algorithm is that it natively samples plateaus in an exhaustive fashion, thus providing a more accurate tool to analyze surfaces that contain this kind of topology.
- In addition to the classic MEPSA contour and profile plots, MEPSAnd offers network projections which can prove useful to understand the underlying structure of complex surfaces.
- MEPSAnd is significantly faster than classic MEPSA and supports scripting based pipelines, streamlining the analysis of large datasets.

6.2 Surface Connectivity Network

Instead of describing point connectivity through a grid, MEPSAnd considers any loaded surface as a network of points, taking only into consideration the points to which each point is connected. For each point of the surface the points to which it is connected are called "neighbors" in MEPSAnd. The user can either use one of the two the built-in neighbor determination methods (see section 7.3.1) or load a precalculated set of neighbors (see section 7.3.2) in order to determine the Surface Connectivity Network.

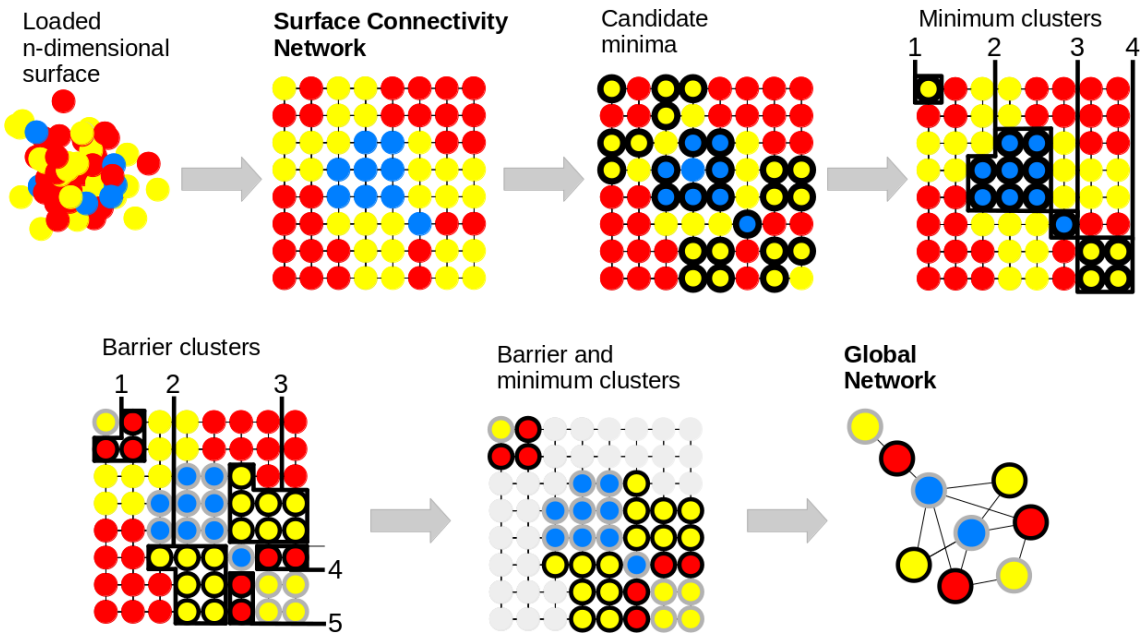

Figure 1: Description of the steps to generate the MEPSAnd Global Network. Colors indicate three discrete energy levels (blue: lowest, yellow: medium, red: highest). After the points of a surface with any number of dimensions are loaded, Surface Connectivity Network (see section 6.2) has to be defined. In this example a 3D (2 coordinates and energy) connectivity is shown for clarity, but note that all the steps after connectivity definition are independent of connectivity complexity. Once connectivity is defined, minimum clusters (see section 6.3.1) and barrier clusters (see section 6.3.3) are detected. First, a list of points with candidate minima compatible configurations is detected (Fig. 2). These points are propagated along plateau configurations (Fig. 2) to test whether they end up contacting minimum cluster incompatible configurations (Fig. 2) or not, obtaining the final list of minimum clusters. Minimum clusters are propagated to any higher neighbors available. The lowest overlapping points between the propagation of different minimum clusters are used to define barrier clusters. Finally, using this information, a network in which minimum and barrier clusters are the only vertices is built (see section 6.3.4).

6.3 Global Network

Once the Surface Connectivity Network has been defined, MEPSAnd abstracts the surface as network of minima connected by barriers, referred to as Global Network, allowing the description of any transition between two points of the surface as a succession of intermediate states (minima) and barriers, in a fast but exhaustive and accurate way. To build the Global Network, a series of steps are followed (Fig. 1):

- Minima detection (6.3.1).
- Minima propagation and annotation (6.3.2).
- Barrier detection (6.3.3).
- Building of minima and barriers network: the Global Network. (6.3.4).

6.3.1 Minima detection

First MEPSAnd searches for candidate minima. Any point with an energy value less or equal to those of its neighbors and lower than at least one of them is considered a candidate minima (candidate minima configurations in Fig. 2).

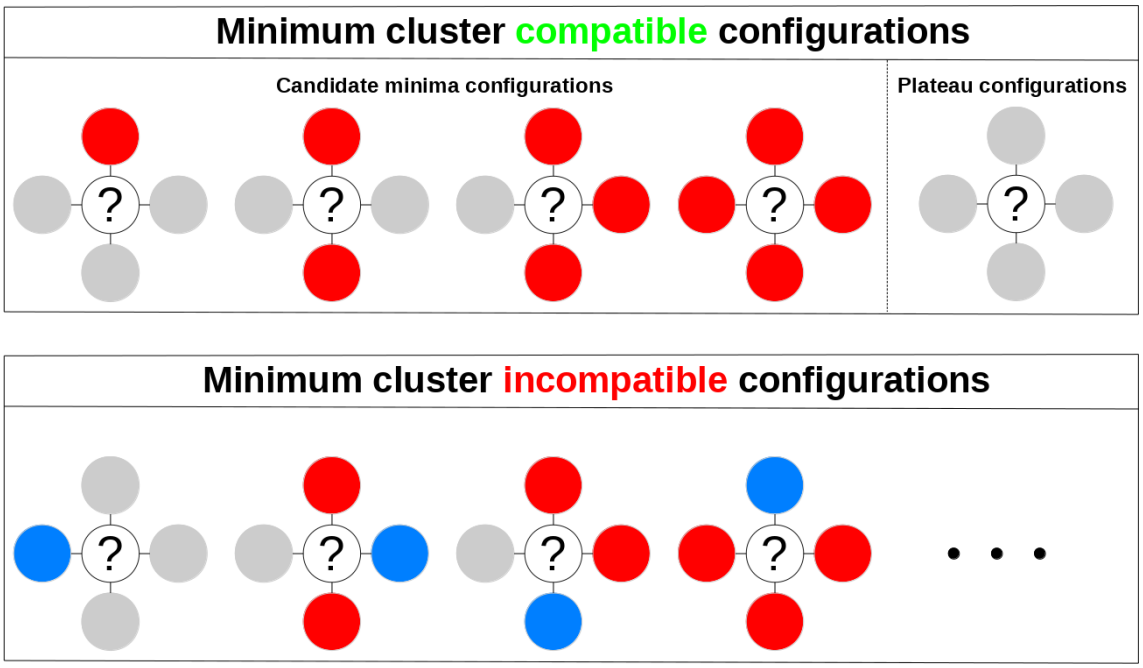

Figure 2: Description of the neighbor configurations that all the points of a minimum cluster can have. The point which is evaluated is denoted by a question mark, while the color code indicates the relative energy of each neighbor compared with this point. Red denotes a higher energy value, grey the same energy value and blue a lower one. Minimum cluster detection starts propagating from minima candidate configurations to both points with either minima candidate configurations or plateau configurations. If any of the points of the cluster presents an incompatible configuration, the whole cluster cannot be a minimum cluster any more. Note that, for illustrative purposes, in this figure only 4 neighbors per point are depicted, but the same principles would apply to any arbitrary number of neighbors per point.

Then MEPSAnd propagates every candidate minimum to those of its neighbors that present the same energy value (plateau configurations in Fig. 2). Such propagation continues on those visited neighbors until no more points with the same energy value can be reached. All the points visited during this propagation constitute a unique candidate minimum cluster. A candidate minimum cluster is accepted as a minimum cluster if none of its constituent points has a neighbor with lower energy (minimum cluster compatible configurations in Fig. 2). Any minimum that does not consists on a plateau will generate a single point minimum cluster (Fig. 3).

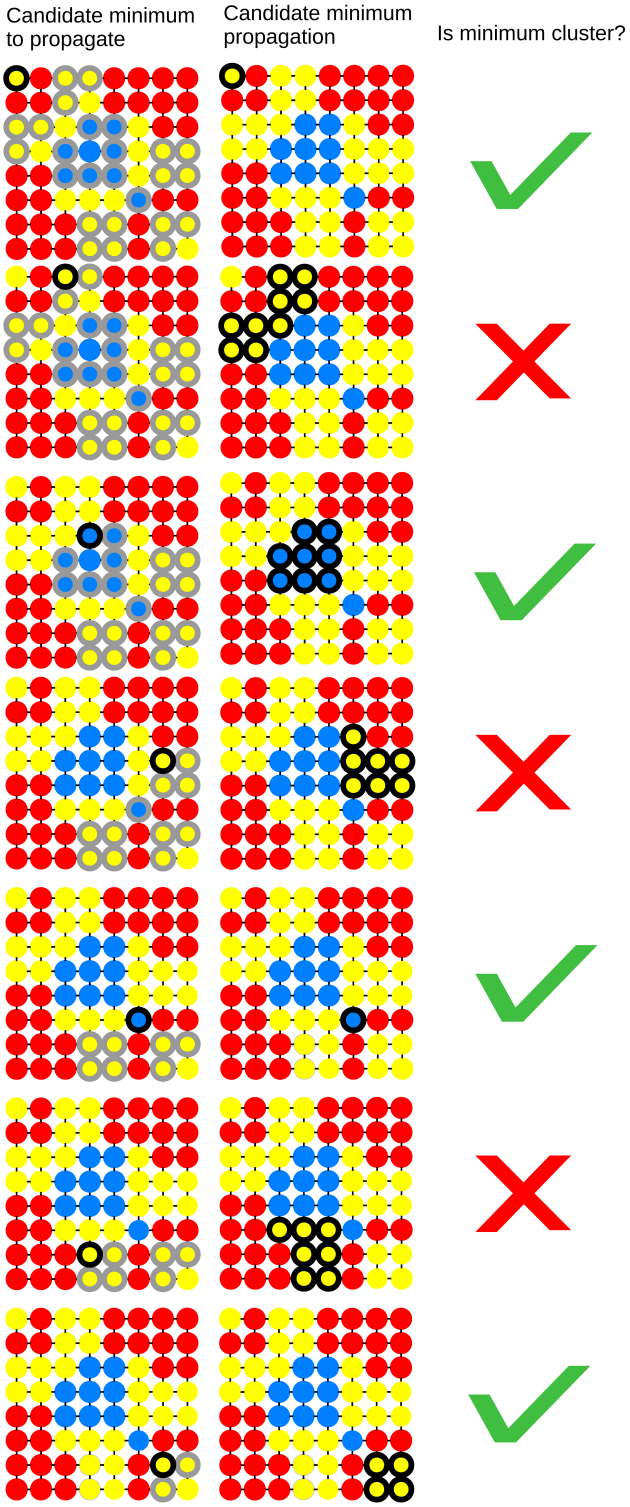

Figure 3: Description of the minimum cluster detection process over the surface shown in Fig. 1. Colors indicate three discrete energy levels (blue: lowest, yellow: medium, red: highest). In this case, to evaluate all the candidate minima detected, seven individual propagations are required. For each of these propagations three sets of information are shown: i) which candidate minimum is being propagated, ii) the points occupied during the propagation and iii) whether the resulting cluster can be considered a minimum cluster (green tick) or not (red cross).

6.3.2 Minima propagation and annotation

After identifying all the minimum clusters in the surface, these are propagated to those neighbors with higher or equal energy values. The same rule is applied to every occupied point, so that, for a given point, the neighbors with higher or equal energy are occupied as well. This propagation continues until no more points can be occupied (Fig. 4).

The annotation process consist on storing the points of the surface occupied during the propagation of each minimum cluster (Fig. 4).

6.3.3 Barrier detection

Using the information obtained during minima propagation MEPSAnd identifies those points occupied by the propagation of more than one minimum cluster. For each pair of minimum clusters that share points occupied during minimum clusters propagation, those shared points with the lowest energy are selected as barrier candidates. Similarly to what is done during minima detection, these barrier candidates are propagated to all those neighbors with equal energy values (Fig. 4). This way, all barrier candidates that are connected end up forming a single barrier cluster (Figs. 1 and 4).

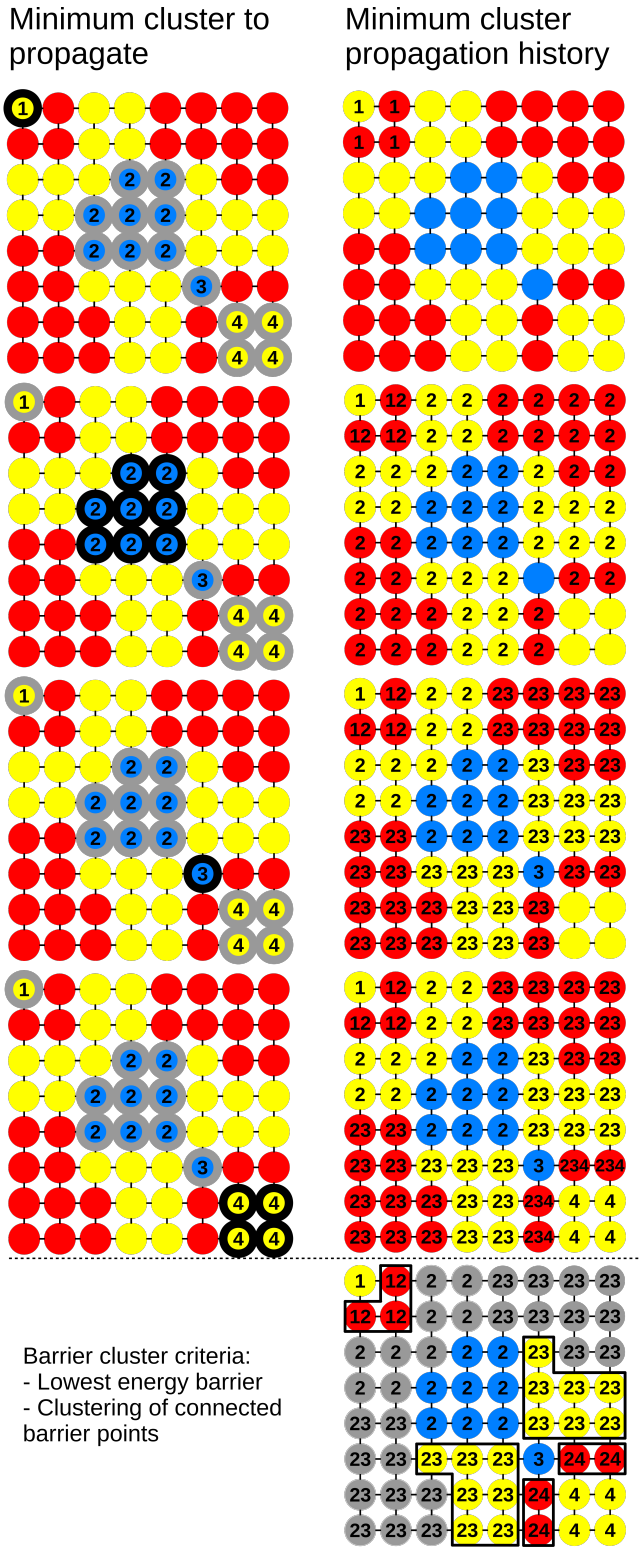

Figure 4: Description of the minima propagation and barrier detection processes over the surface shown in Fig. 1. Colors indicate three discrete energy levels (blue: lowest, yellow: medium, red: highest). As four minimum clusters were detected, four minimum cluster propagation have to be performed to detect the barrier clusters present in the surface. For each propagation two sets of information are shown. Left: The minimum cluster that is propagated (black circle) and the rest of minimum clusters (gray), also providing a different numeric index to each cluster (1-4); right: propagation annotation indicating the indices of the cluster from which each propagation started (i.e. 234 indicates a point visited by the propagation of minimum clusters 2, 3 and 4). The lowest panel below the dotted line indicates how barrier clusters are defined by interpreting the minimum cluster propagations.

6.3.4 Building of minima and barriers network: the Global Network

After the detection of all minimum and barrier clusters, a network in which each cluster is only described by a single vertex is built. In MEPSAnd this network is referred to as "Global Network". As expected, in the Global Network, vertices corresponding to minimum clusters are only connected to vertices that represent barrier clusters and vice versa (Fig. 5).

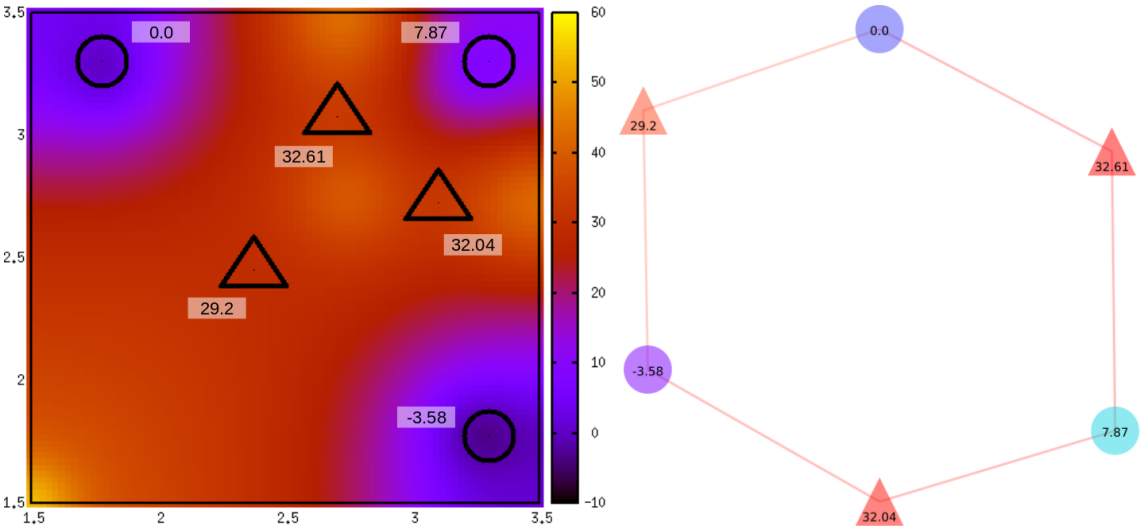

Figure 5: Example of Global Network. Left: an example 3D energy surface plotted via gnuplot. Circles indicate the location of the minimum clusters and triangles the location of the barrier clusters which where extracted via the "Save minimum and barrier clusters" button. Right: representation of the Global Network created by MEPSAnd using the built-in network projection system, representing minimum clusters as circles and barrier clusters as triangles.

6.4 Path calculations

In order to compute the minimum energy path between two given points MEPSAnd propagates over the Global Network in a similar way to classic MEPSA (Fig. 6). The most relevant difference is that two tracebacks are performed instead of one, as the energy guided propagation uses a blind descent approach. Therefore, the information used to select paths is only built during uphill propagations, enabling a strict detection of equivalent paths. Furthermore, due to how the Global Network is built (considering minimum and barrier plateaus as single vertices) minimum and barrier plateaus are exhaustively sampled in a native manner, providing a better description of topologically complex surfaces.

Once the Global Network path has been calculated, MEPSAnd computes paths over this network. The path along the Global Network returns a list of visited pairs of barrier and minimum clusters. The paths connecting each of this pairs are then computed over the full Surface Connectivity Network. Each of these real point paths is computed using a close rewrite of the classic MEPSA approximation, i.e. propagation to the lowest energy neighbors is performed from minimum to barrier annotating the loop step in which each point is occupied to later perform a steepest descent trace back from barrier to minimum minimizing the steps taken. Each of these paths is called a "path fragment". Path fragments are combined to reconstruct the complete path referred to as "fragmentwise path" (Fig. 7). MEPSAnd connects the ends of these fragments to build a "fragmentwise connectivity" that is used to sort the energy profiles as well as to perform some of the path reductions available (check section 6.4.1). After successful fragmentwise path calculations are done, a series of representations of the calculated path is enabled (Fig. 6), such as energy profiles (see chapter 7.7.1), coordinate projections (see chapter 7.7.2) and Global Network projections (see chapter 7.7.3).

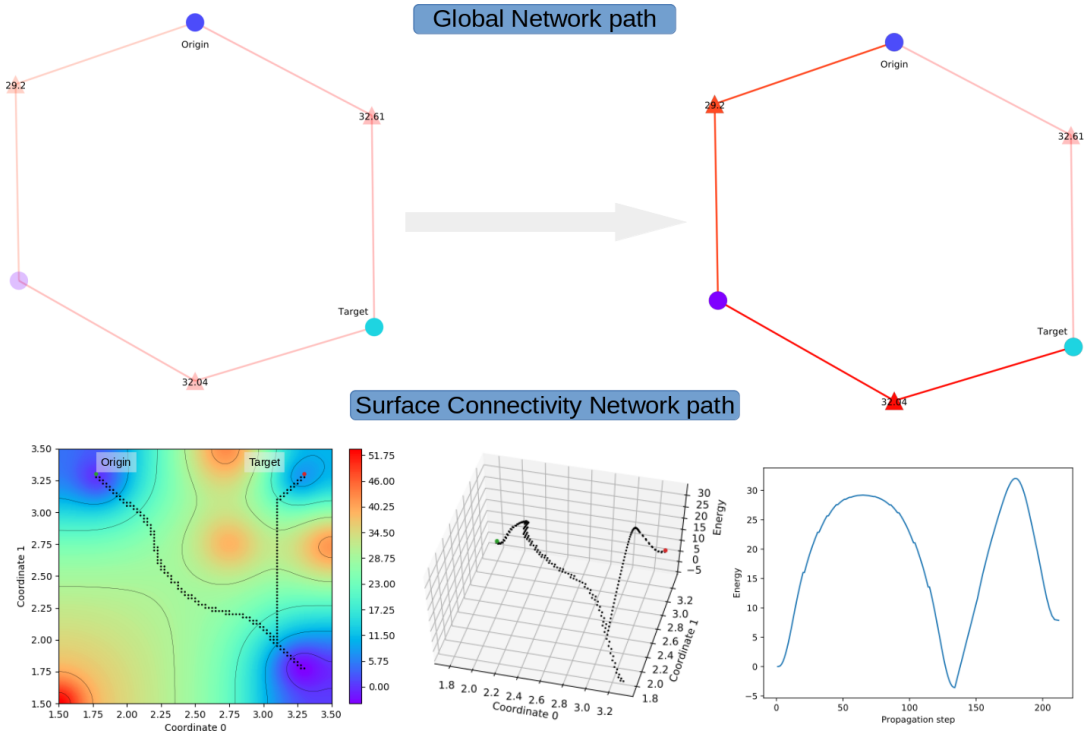

Figure 6: Schematic representation of the path-finding pipeline using MEPSAnd built-in representations. After selection of origin and target nodes, the path connecting them is first calculated over the Global Network and then rebuilt onto the Surface Connectivity Network (i.e. the real surface).

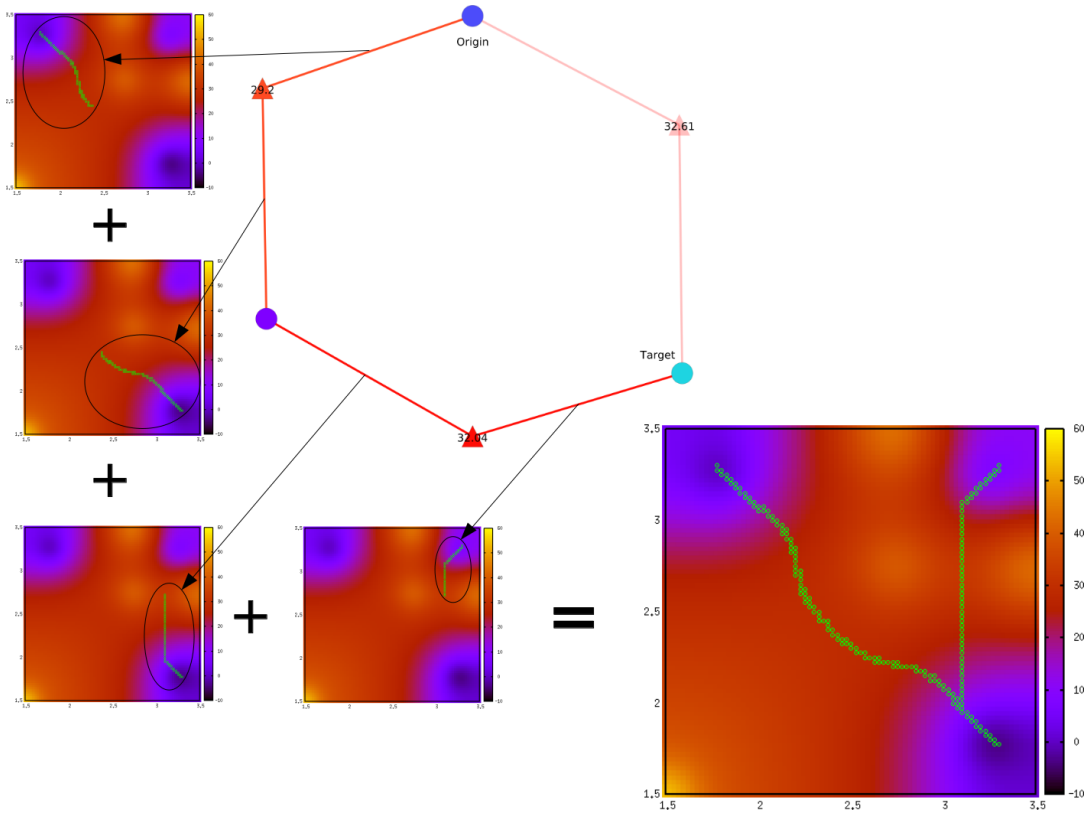

Figure 7: Schematic representation of the fragmentwise path reconstruction.

Summarizing, during path-finding, MEPSAnd first propagates over the Global Network (Fig. 6) to obtain the list of "minimum cluster-barrier cluster" pairs visited along the path. The paths connecting these pairs over the Surface Connectivity Network are computed to build a library of path fragments (Fig. 7). A temporary network connecting these fragments by their overlapping minimum and barrier points is then built ("fragmentwise connectivity"). By propagating over this fragmentwise connectivity, the points of the path fragment library can be sorted by the time in which they are visited, allowing the direct reconstruction of energy profiles describing the path progression. This propagation can also be performed using different heuristics, yielding different path reductions (see section 6.4.1).

6.4.1 Path reductions

Depending on the user needs, MEPSAnd offers a range of path reductions from the most detailed "raw path" to the most streamlined "simplified path". There are 5 levels of path reduction:

- Raw path: the full fragmentwise reconstruction of the Global Network path. When representing the energy profile of raw paths, there is an option to either explicitly represent plateaus ("Raw") or not ("Hide plateaus"). For more details check chapter 7.7.1.
- Level 1 reduction path ("Lvl1 reduction path"): raw path is reduced minimizing the euclidean distance to both origin and target. Reduction is done over the fragmentwise connectivity (see section 6.4).
- Level 2 reduction path ("Lvl2 reduction path"): Level 1 reduction path is further reduced minimizing the number of points between origin and target by using a Dijkstra-like propagation. However, when trace-back is performed, equivalent paths are kept. Reduction is done over the fragmentwise connectivity (see section 6.4).
- Level 3 reduction path ("Lvl3 reduction path"): Level 2 reduction path is further reduced by arbitrarily choosing one single path from any of those available. Reduction is done over the fragmentwise connectivity (see section 6.4).
- Simplified path: Level 3 reduction path is further reduced by performing a classic MEPSA propagation over the full Surface Connectivity Network. This remove any back and forth minima visits and typically yields a path closer to classic MEPSA's "global path".

As expected, depending on the surface that is being analyzed, reduction levels may show no difference when compared to previous ones.

In addition to this path reduction scheme, the use of diagonals (displacements in more than one dimension at once) can be combined with the presented path reductions. This option will take effect if the use of diagonals was allowed during connectivity definition (see section 7.3.1).

For examples of path reduction check chapters 7.7.1 and 7.7.2.

6.4.2 Alternative paths detection

MEPSAnd can automatically propose alternative paths by forcing the sampling of increasingly unfavorable barriers. To do this calculation, MEPSAnd will iteratively remove the connections to the highest and latest barrier visited on each path, forcing the sampling of alternative barriers on each iteration (Fig. 8). The program can iteratively calculate alternative paths until all links connecting the origin region with the target region of the Global Network are finally removed (see the last step of figure 8).

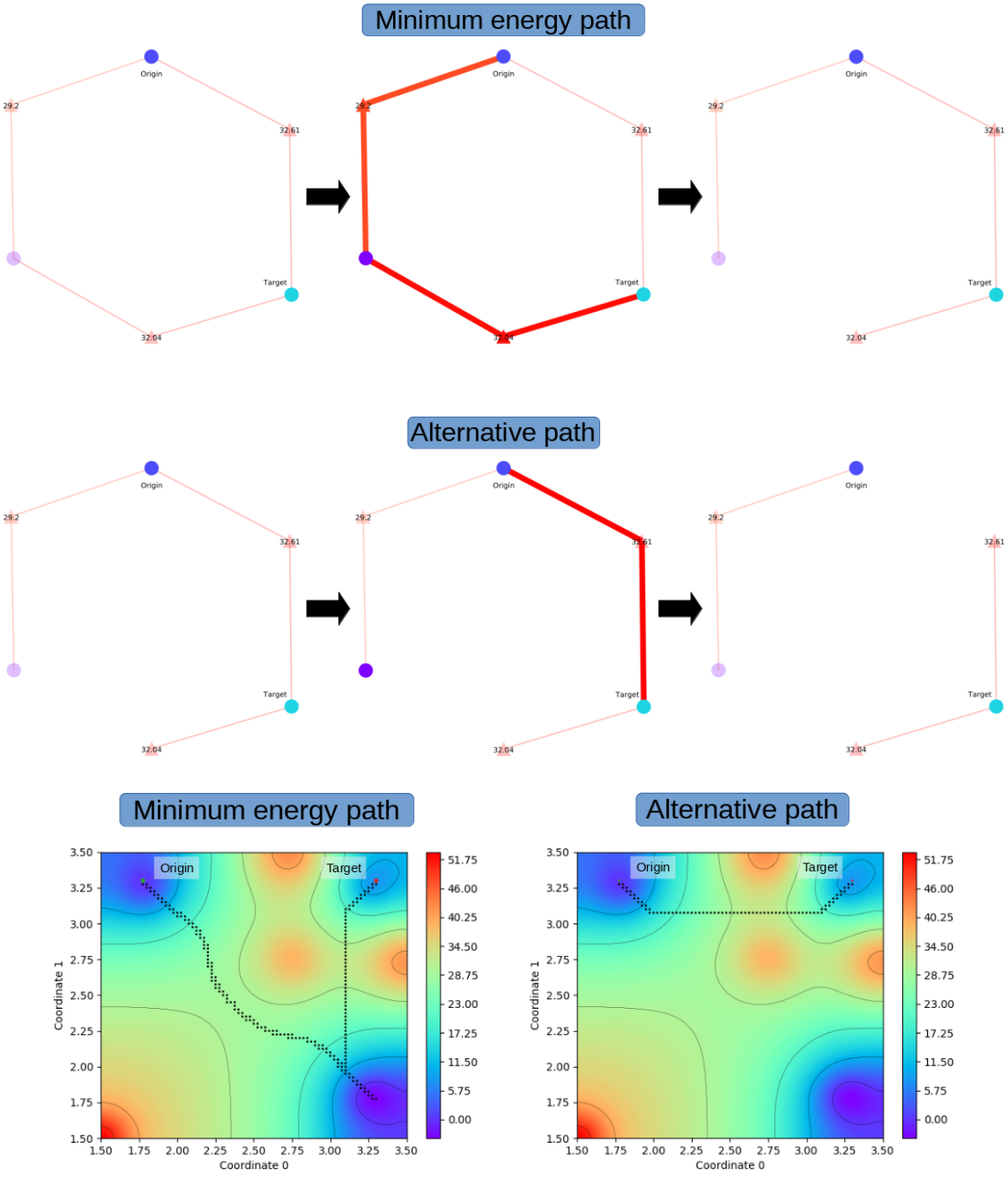

Figure 8: Schematic representation of the alternate path detection algorithm.

6.5 Well sampling

In MEPSAnd, as in classic MEPSA, the region that has to be sampled by the algorithm from origin and target to reach the highest saddled point can be obtained and is referred to as "well sampling" (Fig. 9).

Similarly to the path calculation, MEPSAnd first calculates well sampling over the Global Network to later reconstruct it over the full surface.

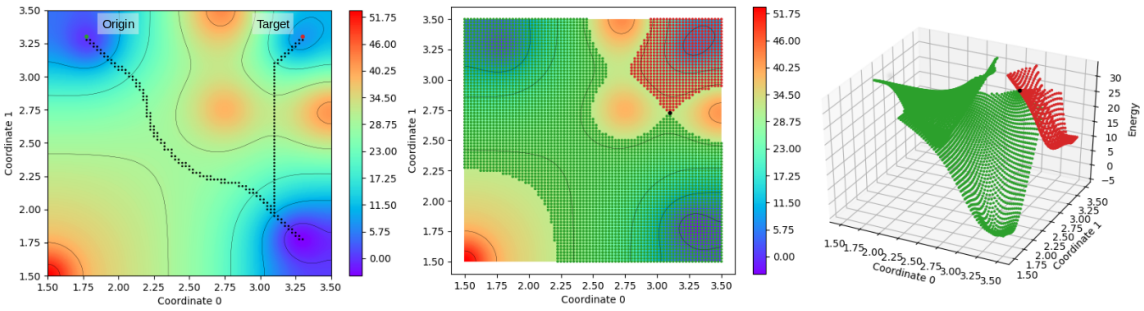

Figure 9: Demonstration of well sampling on a simple 3D surface. Colored points indicate the origin well region (green), target well region (red) and barrier/intermediate region (black).

Well sampling divides the surface in three regions:

- Origin well region: Region that has to be sampled from origin to reach the barrier/intermediate region (Fig. 9, green).
- Target well region: Region that has to be sampled from target to reach the barrier/intermediate region (Fig. 9, red).
- Barrier/intermediate region: the region defined by the highest point/s that have to be sampled to reach target from origin (Fig. 9, black). Typically it will be just formed by the highest barrier clusters visited by the minimum energy path, but if a set of lower energy points exists between two barriers of the exact same energy, an intermediate region is defined. This is why this region is referred to as barrier/intermediate region instead of just barrier region.

MEPSAnd user guide

7.1 Loading and saving sessions

MEPSAnd allows the user to save the current session from any window in the GUI (Fig. 10).

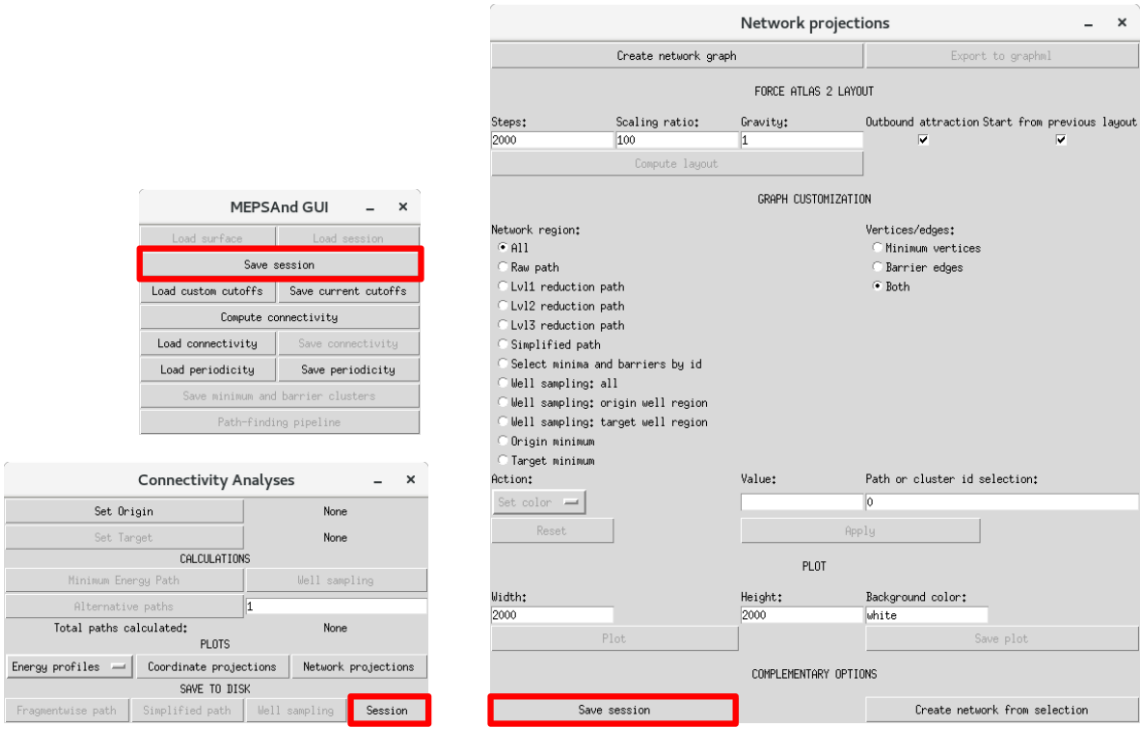

Figure 10: Buttons available to save session.

Depending on what the user have done to the point when a session is saved, the generated file incrementally contains:

- The loaded surface.
- Current cutoffs and/or periodicity and/or connectivity.
- Chosen origin and/or target points
- Computed paths and/or well sampling
- Computed network projections

This is due to how MEPSAnd is structured and what is actually being saved. MEPSAnd GUI interacts with an object called "map handler" that integrates the three objects that manage all

MEPSAnd tasks:

- Connectivity handler: manages all the tasks related with connectivity definition.
- Propagation handler: manages origin and target definition as well as path-finding and well sampling calculations.
- Graph plot handler: manages the network representations that can be used via the "Network projections" button available in the "Path-finding pipeline" window (see chapter 7.7.3).

When a session of MEPSAnd is saved, an instance of the "map handler" object is saved and, with it, instances of the objects it integrates (Fig. 11)

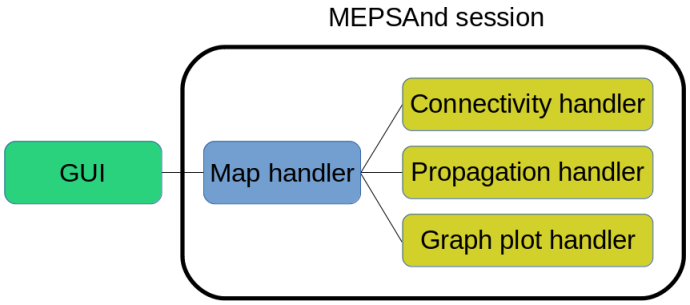

Figure 11: Scheme of MEPSAnd object connections. By saving an instance of the "map handler" object the existing instances of "connectivity handler", "propagation handler" and "graph plot handler" objects are saved as well.

To load a previously saved MEPSAnd session file use the "Load session" button available in the main window (Fig. 12).

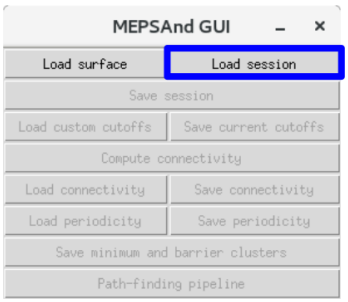

Figure 12: Load session button located in the main window.

7.2 Surface loading

MEPSAnd supports energy surfaces with 3 or more dimensions (including energy i.e. n-1 dimensions + energy).

Unlike classic MEPSA, periodic dimensions can be used (see section 7.3.1) and full sampling of every dimension is not a requirement anymore, as connectivity is no longer described by a grid.

Even data that is not uniformly distributed along one or more dimensions can be used as long as the user can provide a list of neighbors for every point (see section 7.3.2)

To load a surface file in the supported format use the load surface button in the main window (Fig. 13)

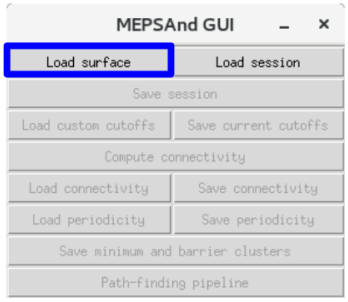

Figure 13: Load surface button located in the main window.

The supported format consists on n columns and m lines for a surface with n dimensions and m points. **Energy always has to be the last column.** Columns can be separated by space or tab symbols (including any combination of both).

As an example, a surface with 10 points (10 lines) and 4 dimensions (4 fields per line), i.e. 3 coordinates and energy, could be:

```
-10 -10 -10 16.3206333266811
-10 10.5 -8 8.16031666334055
10.5 -10 -10 16.3206333266811
10.5 10.5 -10 16.3206333266811
-10 -10 -9.5 11.5943578621746
-10 10.5 -7.5 5.79717893108729
10.5 -10 -9.5 11.5943578621746
10.5 10.5 -9.5 11.5943578621746
-10 -10 -9 14.5894885849632
-10 10.5 -7 7.2947442924816
```

For more surface file examples, please check:

http://bioweb.cbm.uam.es/software/MEPSAnd/

7.3 Defining surface connectivity

MEPSAnd, unlike classic MEPSA, does not rely on grids to define connectivity. Furthermore, in MEPSAnd, connectivity definition is completely detached from path-finding calculations, allowing the use of any set of points with a given energy value as long as the neighbors of each point can be accurately provided by the user.

There are two main strategies to define surface connectivity in MEPSAnd:

- Let MEPSAnd compute surface connectivity using distance cutoffs (see section 7.3.1).
- Provide a precalculated connectivity file (see section 7.3.2).

7.3.1 Using MEPSAnd built-in connectivity detection methods

Although MEPSAnd does not require data to be distributed on a grid as long as point neighborhood can be provided by the user, most of the users will be likely using uniformly distributed data, even if not all the dimensions are fully sampled. This is why MEPSAnd has two different built-in systems to infer network connectivity:

- Cutoff-based connectivity: this method was the only method available on MEPSAnd versions 1.41 or lower. It is based on a set of distance cutoffs to define which points are neighbors to each other. It is significantly slower but can handle networks that are not uniformly distributed over a grid-like structure (*i.e.* surfaces in which the distance between connected points is allowed to vary to some extent).
- Grid-like connectivity: as most MEPSAnd users analyze datasets that are distributed over a n-dimensional grid (even if regions of such grid are left unsampled) in version 1.6 a new connectivity calculation method was implemented and now is the default one. Points are distributed over an n-dimensional grid that matches the detected step size and that grid is used to compute the list of neighbors. This method is dramatically faster that the cutoff-based calculations.

To let MEPSAnd detect the connectivity of the surface, use the "compute connectivity" button on the main window (Fig. 14) and choose the calculation method and whether to accept diagonal neighbors in the options menu that will pop up.

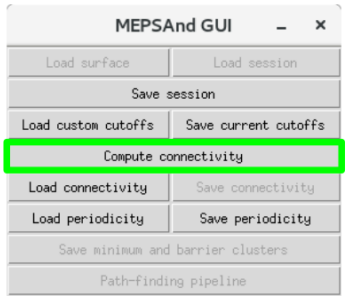

Figure 14: Compute connectivity button located in the main window.

Cutoff-based connectivity

This section describes the legacy method to compute network connectivity from a set of distance cutoffs. This method is still available for two reasons:

- As a fall-back alternative if the grid-like method fails with a dataset.
- To allow users to handle non-uniformly sampled datasets by providing a set of custom cutoffs (loaded from an external file).

As most of the users analyze uniformly sampled datasets, the default behavior of MEPSAnd cutoff-based connectivity calculations is to infer cutoffs from the smallest step detected on each dimension. To prevent errors in connectivity definition due to problems in float point interpretation, MEPSAnd defines two cutoffs per dimension (Fig. 15):

- Lower cutoff: distance between two points bellow which they are considered to be in the same grid step. If no tolerance is to be accepted, this value must be set to 0. The default value is the smallest coordinate difference detected multiplied by 0.5.
- Upper cutoff: distance between two points up to which they are considered to be one grid step away. If no tolerance is to be accepted, this value must be set to the exact grid step. The default value is the smallest coordinate difference detected multiplied by 1.5.

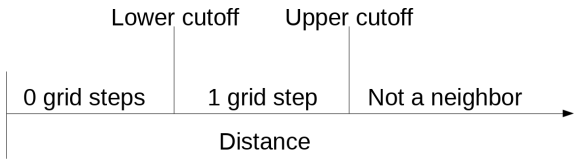

Figure 15: During neighbor detection, when the distance between two points is measured along a given coordinate, if such distance is bellow the lower cutoff both points are considered to be in the same grid step. If the measured distance is larger than the lower cutoff and lower or equal to the upper cutoff, both points are considered to be one grid step away. If distance is larger that upper cutoff, this two points cannot be neighbors.

When detecting the regular neighbors of a point, only those that are one grid step away in just one coordinate are accepted as neighbors (Fig. 16).

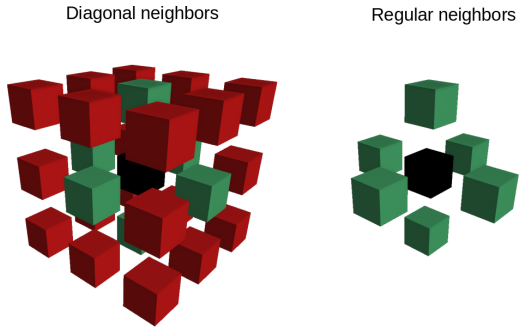

Figure 16: Schematic representation of neighbor detection on a 4D surface for a point (black), indicating the regular neighbors (green) and those only allowed with diagonal connectivity (red). Figure was created using VoxEdit (https://www.voxedit.io/).

Using custom cutoffs. To modify the distance cutoffs before computing connectivity, the "load custom cutoffs" button (Fig. 17) allows the user to load a cutoffs file, indicating the precise pair of cutoffs to be used for each dimension. For user convenience there is also a "save current cutoffs" button (Fig. 17), allowing to save the current cutoffs to the supported cutoff format, which may be modified and loaded again.

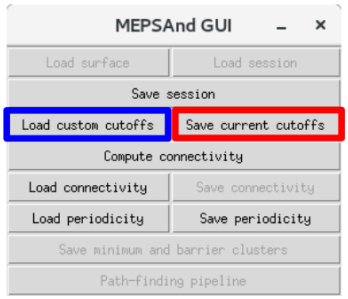

Figure 17: Location of the load custom cutoffs and save current cutoffs buttons in the main window.

As an example, a cutoff file for a 4D file could be:

```
0.0125  0.0375
0  1
0.0125  0.0375
```

Note that a 4D surface requires 3 pairs of cutoffs, as energy is the fourth dimension.

When using custom cutoffs take into account that using cutoffs with no tolerance ranges may induce artifacts in connectivity detection due to errors in float point interpretation.

Grid-like connectivity

Starting in version 1.6, there is a new default method for computing network connectivity for homogeneously sampled (steps size in each dimension must be constant) which implies distributing the points on a n-dimensional grid and inferring neighbor connectivity from that grid. Please take into account the sampling constraints required to use this connectivity method:

- Step sizes can be different between dimensions but must be constant on each one. In other words, dimension 1 can be sampled with a step size of 0.25 and dimension 2 can be sampled with a step size of 0.3, but dimension one cannot have points sampled every 0.25 units and points sampled every 0.1 units.
- Dimensions can be partially sampled. In methods like metadynamics one can obtain a n-dimensional grid that has been partially sampled. This is supported by MEPSAnd and the user **should not fill the non-sampled points with placeholder values**, as that will create a massive and unrealistic barrier cluster artifactually connecting lots of the minimum clusters of the surface, which has very detrimental effects on MEPSAnd performance (see section 7.10).

Diagonal connections

In the special case of the diagonal connectivity that can be used for path reduction, those points that are one grid step away in any number of coordinates (not just one) are accepted as neighbors (Fig. 16). To use diagonal connectivity during path reduction this has to be selected in a pop-up menu that is shown when "compute connectivity" button is used (Fig. 14).

Working with periodic dimensions

MEPSAnd cutoff based connectivity detection supports the use of periodic coordinates. The user can load a periodicity file using the "load periodicity" button (Fig. 18) on the main window as well as save current periodicity with the "save periodicity" button (Fig. 18).

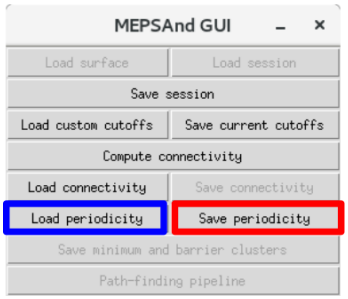

Figure 18: Location of the load periodicity and save periodicity buttons in the main window.

The periodicity file must contain as many lines as dimensions and one positive number indicating the period. A line with a 0 will indicate a non-periodic dimension.

As an example, the periodicity file for a 4D (3 coordinates and energy), in which coordinates 1 and 3 are not periodic and coordinate 2 has a period of 360, would be:

```
0
360
0
```

Note that for a 4D surface a 3 line periodicity file is provided, as energy would be the 4th dimension.

7.3.2 Loading precalculated connectivity

In addition to the cutoff based connectivity detection, MEPSAnd offers the alternative of loading a precalculated connectivity. To load a connectivity file use the "load connectivity" button (Fig. 19) in the main window. Furthermore, if connectivity has already been calculated, it can be saved to a file. This is particularly interesting if the user needs to analyze many large surfaces that share the same connectivity structure (Fig. 19) or if the user can create a connectivity file faster than MEPSAnd, which is not unlikely as MEPSAnd has to measure distances between all points (a computationally demanding task).

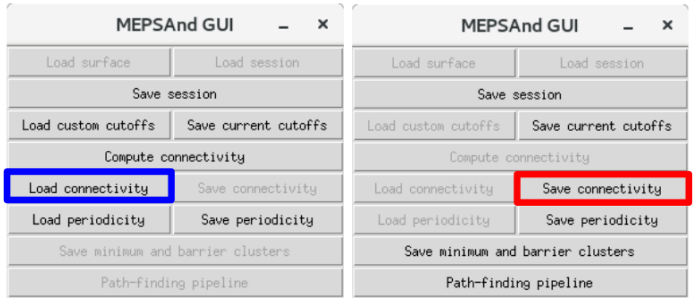

Figure 19: Location of the load periodicity and save connectivity buttons in the main window.

When loading a connectivity file a pop-up will ask whether to load a diagonal connectivity file (Fig. 20). This connectivity is only used for the path reductions (see section 6.4.1). If it is not provided, regular connectivity will be used instead, which is the default behavior when cutoff based connectivity detection is used.

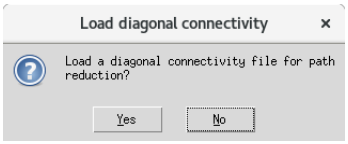

Figure 20: Pop-up window asking wether to load diagonal connectivity or not.

The connectivity file must contain as many lines as points in the surface, and one each line the indices of the neighbors to that point separated by spaces. Note that the point indices are those of the lines in the loaded surface file counting from 0, i.e. point in line 1 of the surface file has index 0, point in line 2 has index 1 and so on.

As an example, check following connectivity file:

```
1 2
0 3
0
1
```

This connectivity would imply that:

- Point 0 is connected to 1 and 2.
- Point 1 is connected to 0 and 3.
- Point 2 is connected to 0.
- Point 3 is connected to 1.

7.4 Saving minimum and barrier clusters

After connectivity is defined, Global Network is obtained (see section 6.3) which requires the detection of all the minimum and barrier clusters in the loaded surface given with such connectivity. MEPSAnd allows the user to save to a plain text file a list of all the minimum and barrier clusters detected using the "save minimum and barrier clusters" button (Fig. 21).

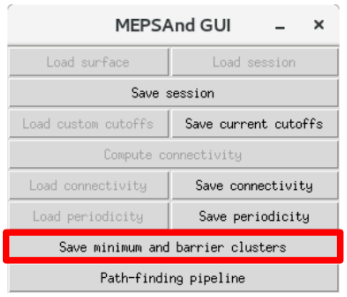

Figure 21: Location of the save minimum and barrier clusters button in the main window.

As a file format example, check the following lines:

```
#Minimum 0
0 3.3 1.775 -3.58
0 3.3 1.75 -3.58
#Minimum 1
1 1.775 3.3 0
#Minimum 2
2 3.3 3.3 7.87
#Barrier 3
3 2.375 2.45 29.2
#Barrier 4
4 3.1 2.725 32.04
#Barrier 5
5 2.7 3.075 32.61
```

This file contains information about 3 minimum cluster and 3 barrier clusters. All of them are single point clusters except minimum cluster 0, which contains 2 points. For each point the cluster index, coordinates and energy are shown. Note that the indices of the clusters are unique and sequential. Minima clusters always go first.

7.5 Path-finding

After loading a surface and defining its connectivity, the path-fining pipeline button becomes active (Fig. 22), that opens the path-finding pipeline window (Fig. 23).

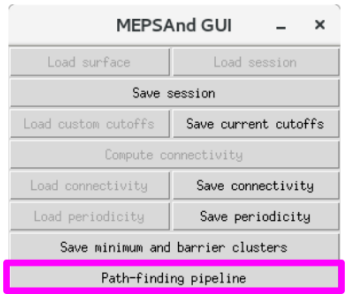

Figure 22: Loaction of the path-finding pipeline button on the main windown.

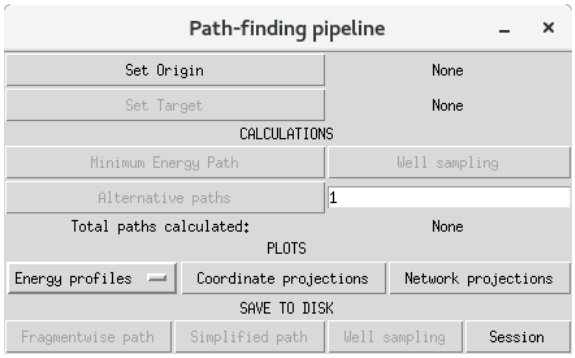

Figure 23: Path-finding pipeline window.

From this window the user can:

- Define the origin and target points for path-finding (7.5.1).
- Calculate the minimum energy path (7.5.2).
- Calculate a number of alternative paths with increasingly higher barriers (7.5.3).
- Calculate the minimum energy path well sampling (7.5.4).
- Save fragmentwise paths (7.5.5).
- Save simplified paths (7.5.6).
- Save the calculated well sampling (7.5.7).
- Save session (7.1).
- Invoke the path profile window (7.7.1).
- Invoke the coordinate projections window (7.7.2).
- Invoke the network projections window (7.7.3).

7.5.1 Defining origin and target points

To define the origin and target points, MEPSAnd offers a range of options via the "set origin" and "set target" buttons (Fig. 24).

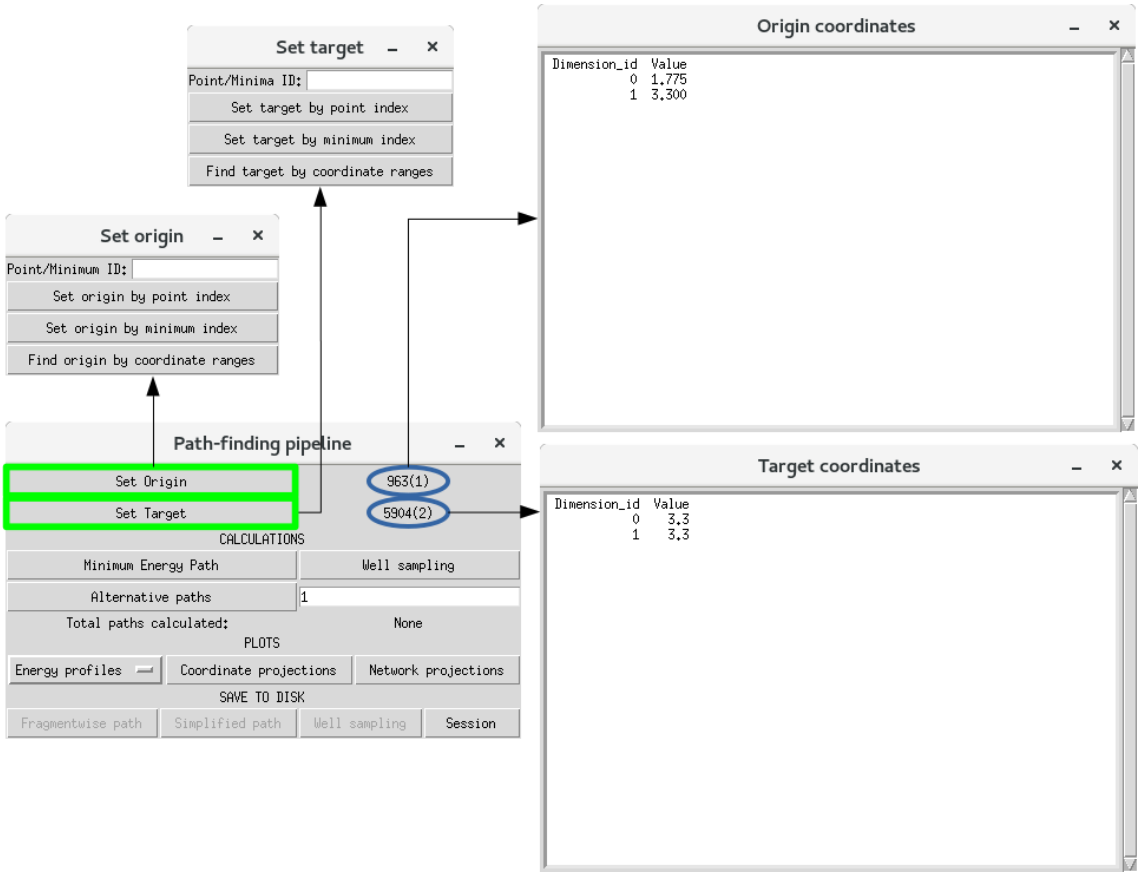

Figure 24: Location of the set origin and set target buttons (green squares). After successful origin or target definition the index of the point and the index of the minimum cluster (number in parenthesis) to which it has been assigned are shown (blue circles). If the user clicks on those numbers, the coordinates of the point are shown.

There are three ways to define an origin or target point:

- Set by point index: reads the index of the point. Note that points indices are assigned in the same order as points are read from the input surface file, counting from 0. Therefore, the first point on the file has index 0, the second index 1 and so on.
- Set by minimum index: the point closest the center of minimum cluster with the given index is selected. To identify minimum cluster indices user may:
  - Check the minimum and barrier clusters file (see 7.4).
  - Use a minimum clusters profile plot (see 7.7.1).
  - Use minimum and barrier clusters coordinate projections (see 7.7.2).
- Set by coordinate ranges (see 7.5.1).

Set origin or target by coordinate ranges

The find origin or target by range buttons open a window (Fig. 25) that let the user define upper and lower limit pairs for each coordinate either by editing the presented values or by loading a ranges file. When a file is loaded, the presented values are updated accordingly. To use the current

ranges (either loaded or modified by the user) the "set origin or target as the lowest minimum in the coordinate ranges" button must be used.

MEPSAnd will look for the origin or target point following these criteria:

- Find the lowest energy minimum cluster in range and select the closest point of that cluster to the center of mass of the provided ranges.
- If no minimum cluster is detected, select the lowest energy point available.

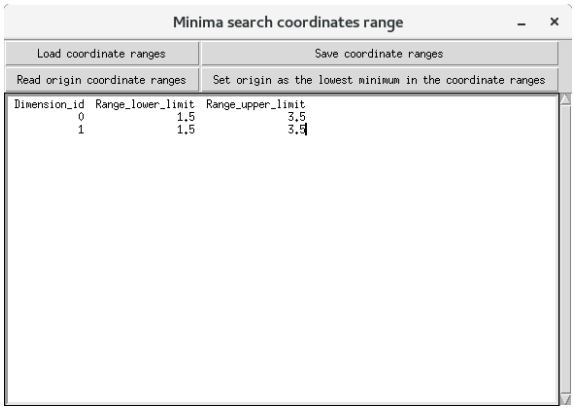

Figure 25: Window invoked by the find origin by coordinate ranges button. The target search equivalent is similar.

The ranges file format accepted by the origin and target search consists on a plain text file with one line per dimension (except energy) and two numeric fields per line, separated by either space or tab symbols, indicating the lower and upper ranges respectively.

As an example, a ranges file for a 4D surface (3 coordinates and energy) could be:

```
1.5 2
1.5 2.5
1 1.5
```

7.5.2 Minimum energy path calculation

Once origin and target points have been defined, if both points are assigned to different minima, the "minimum energy path" button becomes active (Fig. 26).

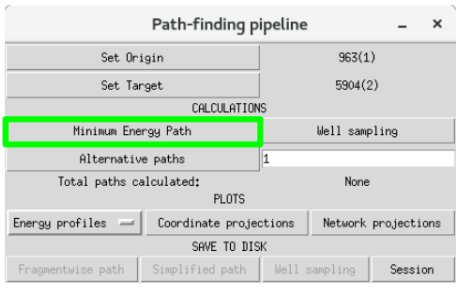

Figure 26: Location of the minimum energy path button on the path-finding pipeline window.

After a successful calculation, the counter of total paths calculated will indicate 1 and path saving buttons will become active (see 7.5.5 and 7.5.6) and path related options will become available in all the plotting systems (see 7.7.1, 7.7.2 and 7.7.3).

7.5.3 Alternative paths calculation

Once origin and target points have been defined, if both points are assigned to different minima, the "alternate paths" button becomes active (Fig. 27).

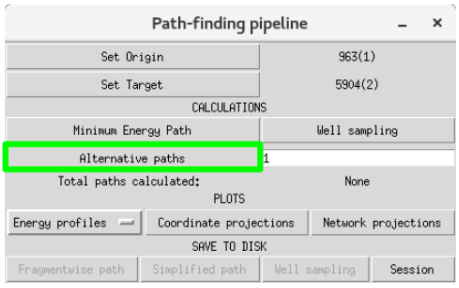

Figure 27: Location of the alternate paths button on the path-finding pipeline window.

The user may indicate the number of alternative paths MEPSAnd will try to find. When alternate paths have been calculated the user can select any combination of paths to represented in all MEPSAnd saving (see 7.5.5 and 7.5.6) and plotting functions (see 7.7.1, 7.7.2 and 7.7.3). The minimum energy path has always index 0 and the first alternative path has index 1, the second has index 2 and so on. If minimum energy path was not computed when the user clicks on the "alternate paths", MEPSAnd will first calculate the minimum energy path and, then, proceed to compute as many alternative paths as indicated by the user.

As soon as a single path is calculated the path saving buttons become active (see 7.5.5 and 7.5.6).

7.5.4 Well sampling calculation

Once origin and target points have been defined, if both points are assigned to different minima, the "well sampling" button becomes active (Fig. 28).

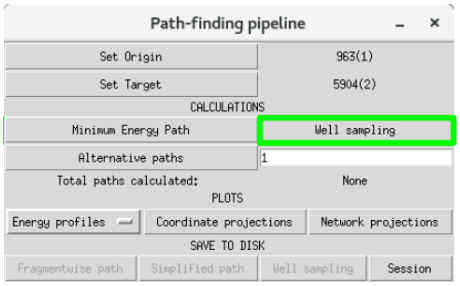

Figure 28: Location of the well sampling button on the path-finding pipeline window.

When well sampling is calculated the option of saving well sampling becomes available (see 7.5.7). Also, various well sampling related options become available for coordinate and network projections (7.7.2 and 7.7.3).

7.5.5 Saving fragmentwise paths

Calculated fragmentwise paths can be saved from the GUI (Fig. 29).

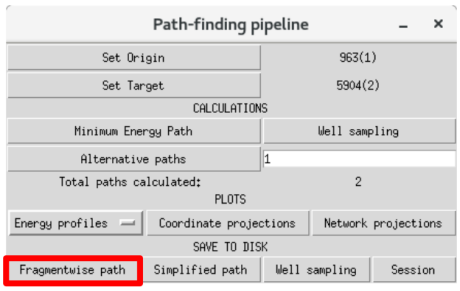

Figure 29: Location of the button to save fragmentwise paths in the path-finding pipeline window.

If many paths have been calculated, the user will be asked which paths to save at once. If a single path index is used the only one file will be generated with the full name the user has provided and the ".path" extension. On the other hand if many paths are selected to be saved at once, one file will be created per path and the input file name will be followed by "_" and the index of the path.

For example, if 4 paths have been calculated and when asked the user saves paths "1-3", giving the file name "a", the files generated would be:

```
a_1.path
a_2.path
a_3.path
```

See chapter 7.6 for a description of how index selection works in MEPSAnd.

The generated file format consists on path fragments preceded by a line starting with a "#" symbol, indicating which minimum and barrier clusters are connected. For each point of the fragment a line is written with the following fields:

- Index of the point in the full surface. With this the user can identify the point in the surface input file. Note that these indices begin counting on 0, so the first point in the input surface file has index 0, the second has index 1, and so on.
- As many fields as dimensions, indicating the coordinates and energy of a given point, being the energy always the last of these fields.
- Raw path trace: step of occupation of a given point in the assigning different occupation steps to plateau points so they can be seen as flat regions in an energy profile when plotted.
- Raw path trace hiding plateaus: step of occupation of a given point in the assigning the same occupation step to the points belonging to the same plateau, so they cannot be seen in an energy profile as flat regions when plotted.
- Level 1 reduction trace: step of occupation of a given point over the level 1 reduction path (see 6.4.1).
- Level 2 reduction trace: step of occupation of a given point over the level 2 reduction path (see 6.4.1).
- Level 3 reduction trace: step of occupation of a given point over the level 3 reduction path (see 6.4.1).

The purpose of the last 5 trace fields is to be used as sorting criteria for different profile representations. Note that a trace value of 0 implies that a point is not visited in a given path reduction.

7.5.6 Saving simplified paths

Calculated simplified paths can be saved from the GUI (Fig. 30).

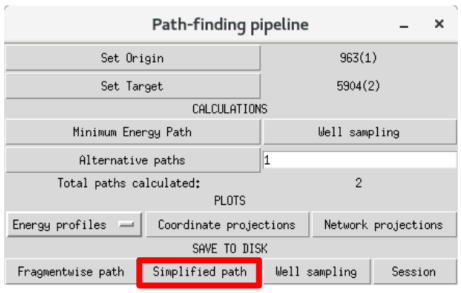

Figure 30: Location of the button to save simplified paths in the path-finding pipeline window.

If many paths have been calculated, the user will be asked which paths to save at once. If a single path index is used the only one file will be generated with the full name the user has provided and the ".spath" extension. On the other hand if many paths are selected to be saved at once, one file will be created per path and the input file name will be followed by "_" and the index of the path.

For example, if 4 paths have been calculated and when asked the user saves paths "1-3", giving the file name "a", the files generated would be:

```
a_1.spath
a_2.spath
a_3.spath
```

See chapter 7.6 for a description of how index selection works in MEPSAnd.

The generated file format writes one point per line. Each of these lines contains, in order, the following fields:

- Index of the point in the full surface. With this the user can identify the point in the surface input file. Note that these indices begin counting on 0, so the first point in the input surface file has index 0, the second has index 1, and so on.
- As many fields as dimensions, indicating the coordinates and energy of a given point, being the energy always the last of these fields.

7.5.7 Saving well sampling

Calculated well sampling can be saved from the GUI (Fig. 31).

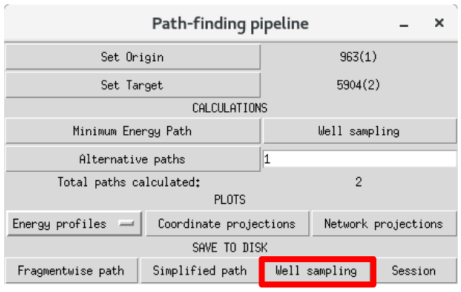

Figure 31: Location of the button to save well sampling in the path-finding pipeline window.

The generated file format consists on well sampling regions preceded by a line starting with a "#" symbol, indicating the type of well sampling region (see 6.5). For each point of a region a line is written with the following fields:

- Index of the point in the full surface. With this the user can identify the point in the surface input file. Note that this indices begin counting on 0, so the first point in the input surface file has index 0, the second has index 1, and so on.
- As many fields as dimensions, indicating the coordinates and energy of a given point, being the energy always the last of these fields.

7.6 Index based selections

There are many instances in which MEPSAnd will ask the user to select points, clusters or paths using their corresponding indices. The supported selection syntax allows:

- To use individual indices separated by space character, i.e. "1 3 5 7" would select indices 1, 3, 5 and 7.
- To indicate indices ranges using the "-" symbol to connect the lower and upper limits of the range, i.e. "3-7 15-17" would select indices 3, 4, 5, 6, 7, 15, 16 and 17. This can be arbitrarily combined with the individual indices syntax.
- To select all indices available using the keyword "all".

7.7 Data plotting

Working with n-dimensional data is the main focus of MEPSAnd. Easy to use plotting of relevant information can be a challenging task. MEPSAnd offers three different kinds of representations to facilitate the interpretation of results regardless of the number of dimensions of the surface being analyzed:

- Energy profiles: plots showing the energy level evolution of paths as well as the energy levels of different minimum and barrier clusters.
- Coordinate projections: 2D and 3D projections of points against sets of 2 or 3 coordinates chosen by the user.
- Network projections: projections of sets of points over the global network (see 6.3) MEPSAnd computations are based on.

For step-by-step examples see section 7.9.

7.7.1 Energy profiles

Energy profiles (Fig. 32) represent the energy value of a set of points over a certain metric (occupation step or cluster index). There are two types of energy profiles:

- Paths: there are 6 different path profile representations, all of them consisting on plotting the energy of each point vs the step in which it was occupied:
  - Raw path: full fragmentwise path profile assigning different occupation steps to plateau points so they can be seen as flat regions.
  - Hidden plateaus: full fragmentwise path profile assigning the same occupation step to the points belonging to the same plateau, so that each plateau is seen as a single point instead of a flat region.
  - Lvl1 reduction path: energy profile of the level 1 reduction path (see 6.4.1).
  - Lvl2 reduction path: energy profile of the level 2 reduction path (see 6.4.1).
  - Lvl3 reduction path: energy profile of the level 3 reduction path (see 6.4.1).
  - Simplified path: energy profile of the simplified path (see 6.4.1).
- Minimum and barrier clusters: the energy of the chosen minimum or barrier clusters is plotted against the cluster indices.

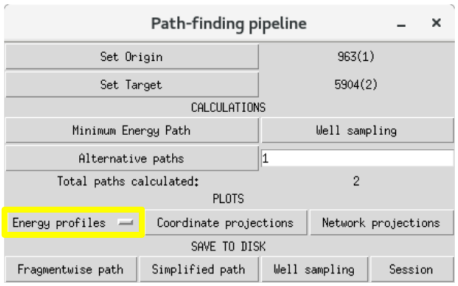

Figure 32: Location of the energy profiles button in the path-finding pipeline window.

For step-by-step examples see section 7.9.

7.7.2 Coordinate projections

Coordinate projections (Fig. 33) allow to plot different sets of points against 2 or 3 coordinates chosen by the user.

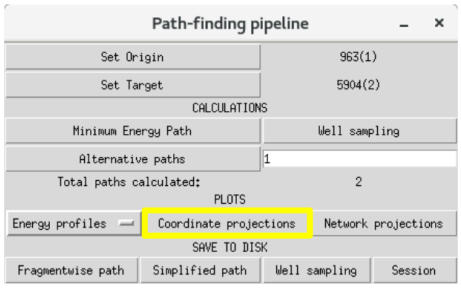

Figure 33: Location of the coordinate projections button in the path-finding pipeline window.

Depending on the number of dimensions of the loaded surface, the coordinate projections window will differ. If a 3D surface (2 coordinates + energy) is loaded, a contour plot button is present, allowing to perform 2D projections over a contour plot similar to the plots classic MEPSA offered (Fig. 34, top). For higher number of dimensions this option will not appear (Fig. 34, bottom).

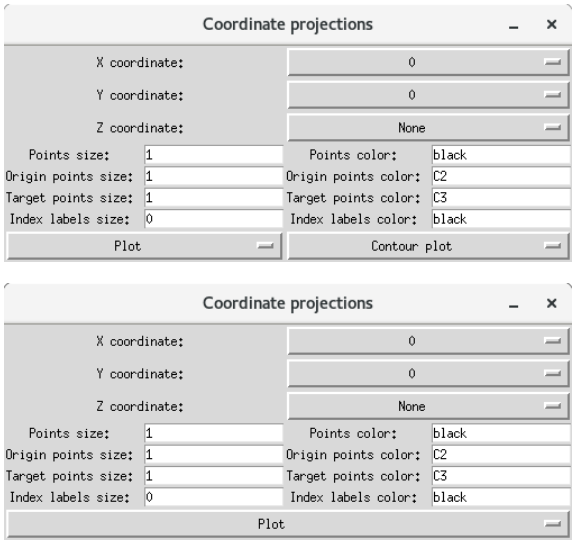

Figure 34: Coordinate projections window for surfaces with 3 dimensions (top) and 4 or more (bottom).

In this window the user may assign any surface dimension to each Cartesian coordinate. If Z coordinate box is left as "None" a 2D plot is generated. Otherwise a 3D plot is obtained.

There are 3 sets of options that control color and point size on the plots:

- "Origin points" options: affect points that are related to origin in the chosen plot.
- "Target points" options: affect points that are related to target in the chosen plot.
- "Points" options: affect points that are neither related to origin or target in the chosen plot.

Depending on the calculations done, a varying range of sets of points may become plottable:

- Raw path: full fragmentwise path. Origin point is plotted using the "origin points" options, target point using the "target points" options and the rest of the points using the "points" options.
- Lvl1 reduction path: level 1 reduction path (see 6.4.1). Origin point is plotted using the "origin points" options, target point using the "target points" options and the rest of the points using the "points" options.

- Lvl2 reduction path: level 2 reduction path (see 6.4.1). Origin point is plotted using the "origin points" options, target point using the "target points" options and the rest of the points using the "points" options.
- Lvl3 reduction path: level 3 reduction path (see 6.4.1). Origin point is plotted using the "origin points" options, target point using the "target points" options and the rest of the points using the "points" options.
- Simplified path: simplified path (see 6.4.1). Origin point is plotted using the "origin points" options, target point using the "target points" options and the rest of the points using the "points" options.
- Select minima and barriers by id: minimum and barrier clusters selected by index (see 7.6). It is plotted using the "points" options.
- "Well sampling: all": all three well sampling regions (see 6.5). Origin well region is plotted using the "origin points" options, target well region is plotted using the "target points" options and barrier/intermediate region is plotted using the "points" options.
- "Well sampling: origin well region": origin well region of the well sampling (see 6.5. It is plotted using the "origin points" options.
- "Well sampling: target well region": target well region of the well sampling (see 6.5. It is plotted using the "target points" options.
- "Well sampling: barrier/intermediate region": barrier/intermediate region of the well sampling (see 6.5. It is plotted using the "points" options.

For step-by-step examples see section 7.9.

7.7.3 Network projections

MEPSAnd takes advantage of the global network previously generated for path-finding calculations to offer a plotting system independent of the number of dimensions. This allows the user to visualize surface topology and the location of the obtained paths on such topology. To open the network projections window use the "network projections" button (Fig. 35) in the path-finding pipeline window.

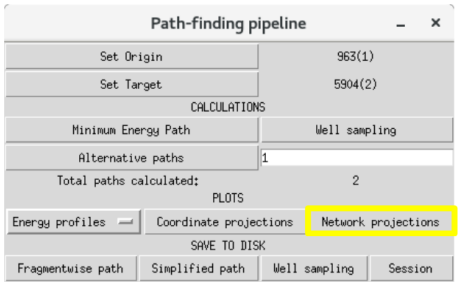

Figure 35: Location of the network projections button in the path-finding pipeline window.

From the network projections window (Fig. 36) the user can create and customize a graph plot describing the global network that MEPSAnd uses to abstract the loaded surface.

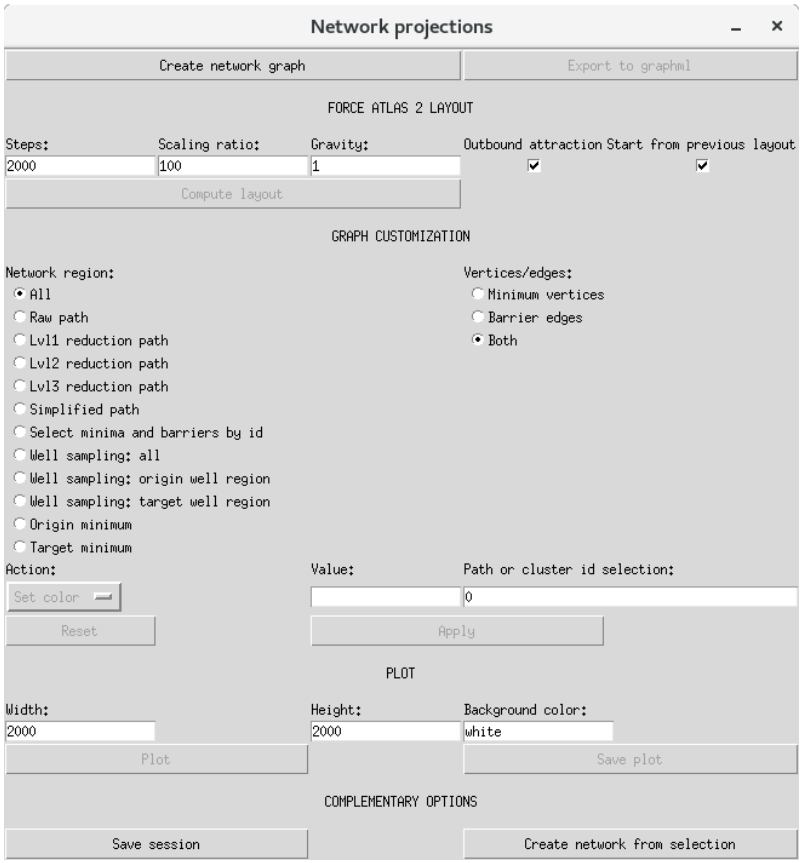

Figure 36: Network projections window.

Create network graph

The first step to use network projections is to create the graph via the "create network graph" button (Fig. 37). This will raise a window asking whether to explicitly represent barriers as vertices or not (Fig. 38).

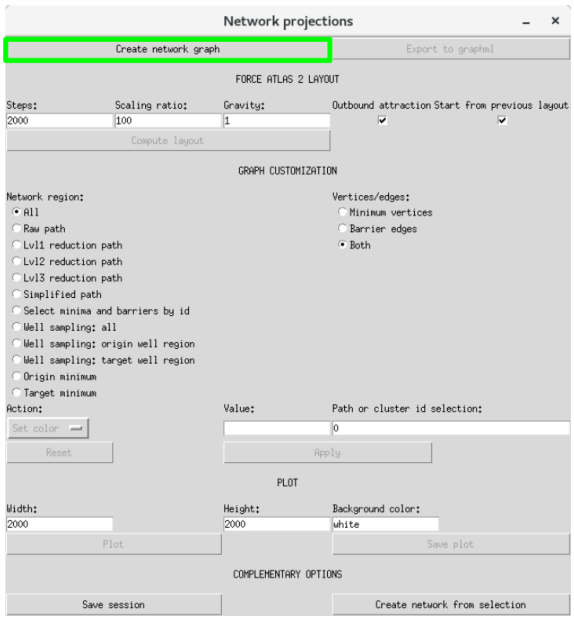

Figure 37: Location of the create network button in the network projections window.

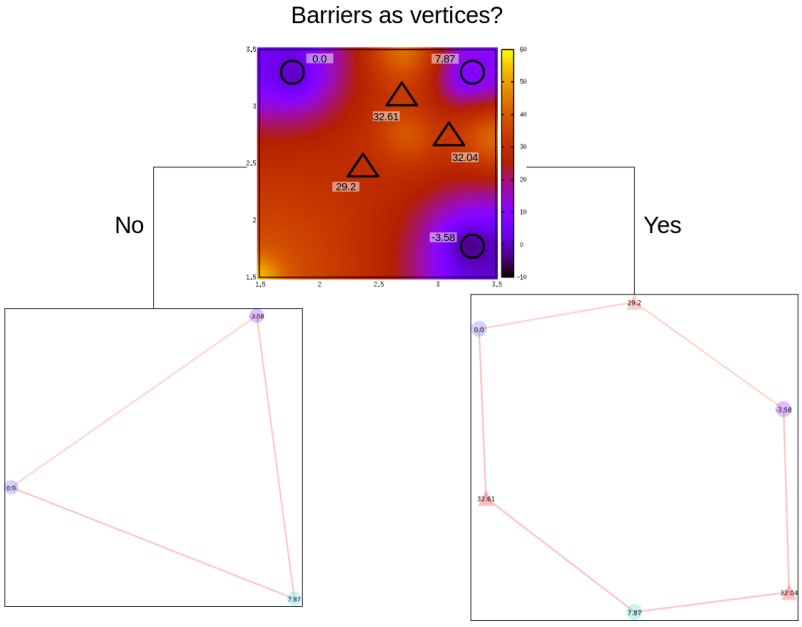

Figure 38: Schematic representation of the two different ways MEPSAnd can abstract the global network as a graph: representing barriers as vertices (right) or not (left).

Representing barriers as vertices allows the user to:

- Visualize if there are more than one equivalent barrier cluster connecting two minimum clusters.
- Visualize if there are more than two minimum clusters connected through the same barrier cluster.

- Add labels to the barrier vertices, such as cluster indices, energy values or arbitrary text strings.

On the other hand, it will create more complex surfaces which might be harder to interpreter depending on the user interests.

In both types of graph, when color maps are used, the edges energy attribute is the energy of the barrier cluster they are connecting, so energy based coloring is consistent among both representations.

Please note that this step is not reversible and saving session before is highly recommended (see section 10).

Force Atlas 2 layout

Once the graph has been created, in order to visualize it, its vertices must be distributed on a plane in the most informative way to describe the network topology. This distribution of vertices is called layout. MEPSAnd uses the port of Gephi's (https://gephi.org/) force atlas 2 algorithm (Jacomy et al. 2014. PLoS ONE 9, e98679) implemented in python by Bhargav Chippada (https://github.com/bhargavchippada/forceatlas2) and now forked, maintained and updated by Amin Alam (https://github.com/AminAlam/fa2_modified). This algorithm efficiently generates easy to interpreter layouts in manageable times even when working with dense and complex networks, a fundamental requirement for the tasks MEPSAnd is supposed to handle. It is a force based algorithm in which the network is iteratively relaxed over a certain amount of steps.

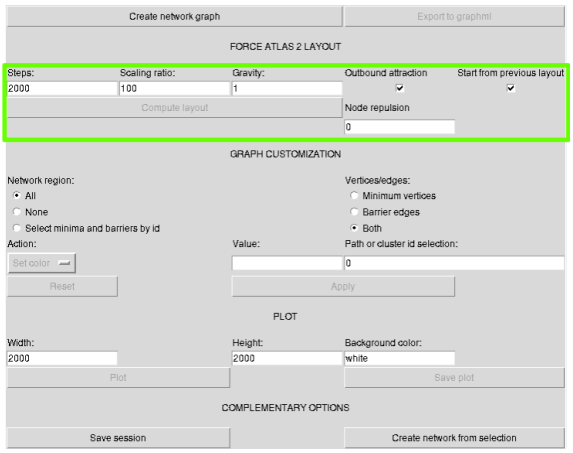

Figure 39: Location of force atlas 2 layout calculation button and options in the network projections window.

The parameters and options to configure the layout calculations are (Fig. 39):

- Steps: total amount of steps along which the network will be calculated. As the relaxation can be cumulative, if the performance of the network to be relaxed is still unknown, it is recommended to start the first layout relaxation with a low step count and gradually rise it.
- Scaling ratio: this parameter can be useful to regulate how cluttered the graph plot is. The higher the scaling ratio the sparser the resulting layout.
- Gravity: this parameter is relevant if there are completely unconnected regions on the network, as it creates a potential that prevents them to get completely separated.
- Outbound attraction: if active, attraction is distributed along outbound edges, dissuading hubs.
- Start from previous layout: the layout obtained in the previous calculation is used to define the initial positions of the vertices. In other words, this allows to continue the previous layout relaxation. If the user wants relax the layout from a random initial distribution, this option has to be unchecked.

- Node repulsion: controls the node size parameter. Increasing this number will separate the nodes further. Setting this to 0 or less disables node repulsion.

When all parameters and options are set, press the "compute layout" button to start the layout relaxation (Fig. 39).

Graph customization

When a layout has been calculated, the "plot" and "save plot" buttons become active. MEPSAnd offers a range of options to customize depending on i) the calculations performed in the path-finding window and ii) the type of graph generated (barriers as vertices or not).

Graph customization consists on selecting a range of vertices, edges or both (selections), and modifying their plotting attributes (actions). On this section selections and actions will be introduced separately, as the user may combine them at will.

In the network projections window a selection is defined combining the options: "network region" (Fig. 40), "vertices/edges" (Fig. 41) and, in some cases, "path or cluster id selection" (Fig. 40). In the description of the "network region" options bellow, it is indicated when the "path or cluster id selection" field is used.

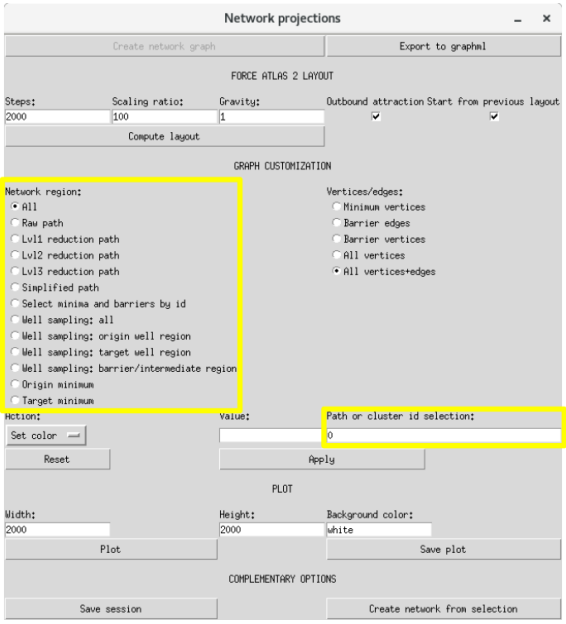

Figure 40: Location of the selection options regarding network regions in the network projections window.

Depending on the calculations performed in the path-finding pipeline window, the network region options can be (Fig. 40):

- All: selects the complete global network.
- Raw path: selects the elements of the global network visited by the complete fragmentwise path (see 6.4). If any alternative path has been computed, the paths selected by this option are defined reading the selection introduced in the "path or cluster id selection" field (see 7.6).
- Lvl1 reduction path: selects the elements of the global network visited by the level 1 reduction path (see 6.4.1). If any alternative path has been computed, the paths selected by this option are defined reading the selection introduced in the "path or cluster id selection" field (see 7.6).
- Lvl2 reduction path: selects the elements of the global network visited by the level 2 reduction path (see 6.4.1). If any alternative path has been computed, the paths selected by this option

are defined reading the selection introduced in the "path or cluster id selection" field (see 7.6).

- Lvl3 reduction path: selects the elements of the global network visited by the level 3 reduction path (see 6.4.1). If any alternative path has been computed, the paths selected by this option are defined reading the selection introduced in the "path or cluster id selection" field (see 7.6).
- Simplified path: selects the elements of the global network visited by the simplified path (see 6.4.1). If any alternative path has been computed, the paths selected by this option are defined reading the selection introduced in the "path or cluster id selection" field (see 7.6).
- Select minima and barriers by id: reads the "path or cluster id selection" field and selects the corresponding minimum and barrier clusters selected (see 7.6).
- "Well sampling: all": selects the elements of the global network visited by any point of the complete well sampling (see 6.5).
- "Well sampling: origin well region": selects the elements of the global network visited by the origin well region of the well sampling (see 6.5).
- "Well sampling: target well region": selects the elements of the global network visited by the target well region of the well sampling (see 6.5).
- "Well sampling: barrier/intermediate well region": selects the elements of the global network visited by the barrier/intermediate region of the well sampling (see 6.5).
- Origin minimum: selects the vertex of the global network corresponding to the minimum cluster assigned to the origin point.
- Target minimum: selects the vertex of the global network corresponding to the minimum cluster assigned to the target point.

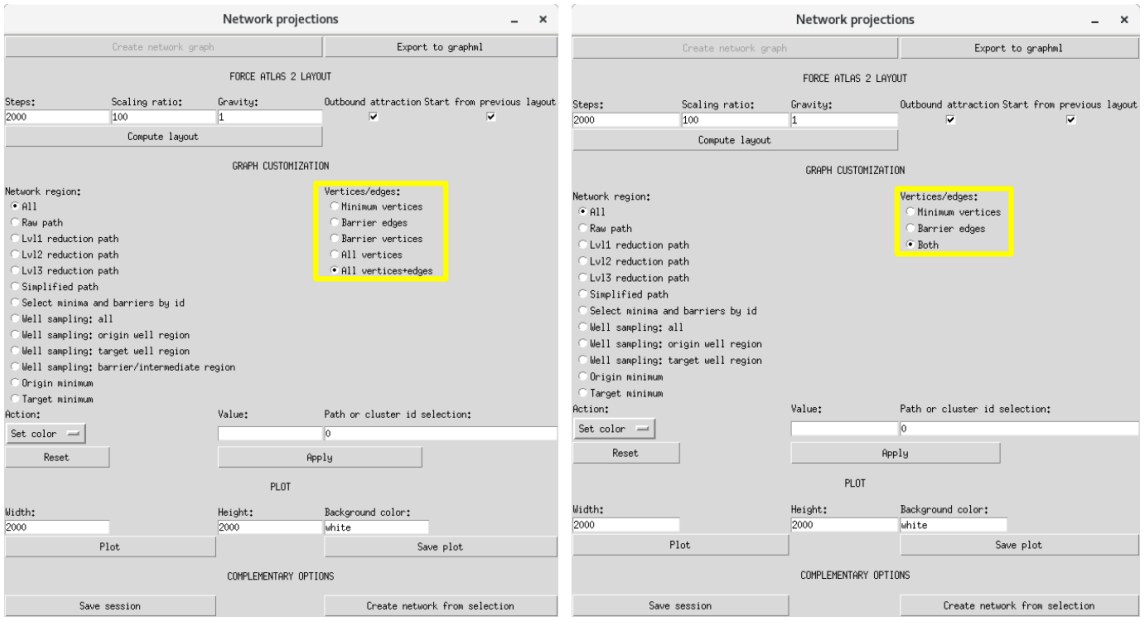

Figure 41: Location of the selection options regarding vertices and edges in the network projections window. These options are different if the graph has been created with barriers as vertices (left) or not (right).

The chosen "network selection" option (Fig. 40) is then modified by the "vertices/edges" chosen option. The "vertices/edges" options (Fig. 41) available will differ between the type of network create (a network with barrier clusters as vertices or not). The possible "vertices/edges" options are (Fig. 41):

- Minimum vertices: selects those vertices that represent minimum clusters.

- Barrier edges: selects those edges that represent barrier clusters.
- Both: selects the union of the sets defined by the "minimum vertices" and "barrier edges" options. This option is only shown for networks in which barriers are not represented by vertices.
- Barrier vertices: selects those vertices that represent barrier clusters.
- All vertices: selects the union of the sets defined by the "minimum vertices" and "barrier vertices" options. This option is only shown for networks in which barriers are represented by vertices.
- All vertices+edges: selects the union of the sets defined by the "all vertices" and "barrier edges" options. This option is only shown for networks in which barriers are represented by vertices.

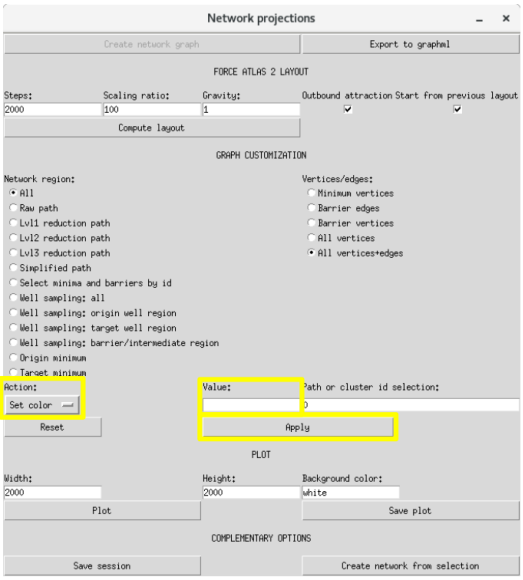

Figure 42: Location of the drop-down actions list and the apply button in the network projections window.

After selecting a set of elements of the network graph using the "network selection" and "vertices/edges" options, a list of actions becomes available to modify the plotting attributes of the selected elements (Fig. 42). Many of the actions available need to read information present in the "value" field (Fig. 42). The use of the "value" field is indicated in the actions descriptions bellow. There are three types of action lists depending on the selection done by the user:

- Actions over vertices:
  - Set color: defines the color of the selected vertices. Color is read from the "value" field. The color definition methods are:
    - Color name string (e.g. red, blue, yellow, etc).
    - Hex color code (e.g. #4286f4).
    - Gray scale. A number from 0 to 1, being 0 black and 1 white (e.g. 0.5 for a middle-point gray).
  - Assign color map: assigns a color map to represent the energy values. The name of the matplotlib color map (https://matplotlib.org/examples/color/colormaps_reference.html) is read from the "value" field. If the "value" field is left empty, "rainbow" color map is used. Note that, for flexibility reasons, the maximum and minimum points taken when assigning the color map depends on the user selection. Therefore if the user wants to assign a color map using the absolute maximum and

minimum energy values, the selection should be built with "all" and either "both" or "all vertices+edges" options.

- Set size: reads a positive number from the "value" field to define the size of the selected vertices. 0 is a valid size.
- Set alpha: reads a real number ranging from 0 to 1 from the "value" field to define the alpha (opacity) of the selection. 0 is completely transparent and 1 completely opaque.
- Set frame width: reads a positive number from the "value" field to define the width of the selected vertices frames. 0 is a valid width.
- Set frame color: reads a valid color definition (see "set color" description) from the "value" field to define the color of the selected vertices frames.
- Set frame alpha: reads a real number ranging from 0 to 1 from the "value" field to define the alpha (opacity) of the selected vertices frames. 0 is completely transparent and 1 completely opaque.
- Set shape: reads a valid shape definition from the "value" field to define the alpha (opacity) of the selected vertices frames. The supported shape definitions are:
  - rectangle
  - circle
  - triangle-up
  - triangle-down
- Set label: reads the "value" field to define the label of the selected vertices.
- Set cluster indices as labels: assigns the corresponding minimum or barrier cluster index as label to the selected vertices.
- Set cluster energy as labels: assigns the corresponding energy value of the selected vertices as their label.
- Set label distance: reads a number from the "value" field to define the distance from the center of each selected vertex at which its label will be drawn.
- Set label angle: reads an angle (in degrees) from the "value" field to define angle at which the label of the selected vertices will be drawn.
- Set label color: reads a valid color definition (see "set color" description) from the "value" field to define the color of the selected vertices labels.
- Set label size: reads a positive number from the "value" field to define the size of the selected vertices labels. 0 is a valid size.
- Bring to front: makes the selected vertices to be plotted on top of the others.
- Bring to back: makes the selected vertices to be plotted behind the others.
- Add tag: reads the "value" field to assign a tag to the selected vertices. This is only useful if the plot is to be exported, as it facilitates the selection of meaningful regions when using third party software.
- Remove tag: reads the "value" field to detect i) if a tag exists and ii) if any of the selected vertices has been assigned such tag. If the tag exists, it will be set to False for all the selected vertices.

- Actions over edges:
  - Set color: defines the color of the selected edges. Color is read from the "value" field. The color definition methods are:
    - Color name string (e.g. red, blue, yellow, etc).
    - Hex color code (e.g. #4286f4).

    - Gray scale. A number from 0 to 1, being 0 black and 1 white (e.g. 0.5 for a middle-point gray).
  - Assign color map: assigns a color map to represent the energy values. The name of the matplotlib color map (https://matplotlib.org/examples/color/colormaps_reference.html) is read from the "value" field. If the "value" field is left empty, "rainbow" color map is used. Note that, for flexibility reasons, the maximum and minimum points taken when assigning the color map depends on the user selection. Therefore if the user wants to assign a color map using the absolute maximum and minimum energy values, the selection should be built with "all" and either "both" or "all vertices+edges" options.
  - Set width: reads a positive number from the "value" field to define the width of the selected edges. 0 is a valid width.
  - Set alpha: reads a real number ranging from 0 to 1 from the "value" field to define the alpha (opacity) of the selection. 0 is completely transparent and 1 completely opaque.
  - Bring to front: makes the selected edges to be plotted on top of the others.
  - Bring to back: makes the selected edges to be plotted behind the others.
  - Add tag: reads the "value" field to assign a tag to the selected edges. This is only useful if the plot is to be exported, as it facilitates the selection of meaningful regions when using third party software.
  - Remove tag: reads the "value" field to detect i) if a tag exists and ii) if any of the selected edges has been assigned such tag. If the tag exists, it will be set to False for all the selected edges .
- Actions over vertices+edges:
  - Set color: defines the color of the selected vertices and edges. Color is read from the "value" field. The color definition methods are:
    - Color name string (e.g. red, blue, yellow, etc).
    - Hex color code (e.g. #4286f4).
    - Gray scale. A number from 0 to 1, being 0 black and 1 white (e.g. 0.5 for a middle-point gray).
  - Assign color map: assigns a color map to represent the energy values. The name of the matplotlib color map (https://matplotlib.org/examples/color/colormaps_reference.html) is read from the "value" field. If the "value" field is left empty, "rainbow" color map is used. Note that, for flexibility reasons, the maximum and minimum points taken when assigning the color map depends on the user selection. Therefore if the user wants to assign a color map using the absolute maximum and minimum energy values, the selection should be built with "all" and either "both" or "all vertices+edges" options.
  - Set alpha: reads a real number ranging from 0 to 1 from the "value" field to define the alpha (opacity) of the selection. 0 is completely transparent and 1 completely opaque.
  - Bring to front: makes the selected vertices and edges to be plotted on top of the others.
  - Bring to back: makes the selected vertices and edges to be plotted behind the others.
  - Add tag: reads the "value" field to assign a tag to the selected vertices and edges. This is only useful if the plot is to be exported, as it facilitates the selection of meaningful regions when using third party software.
  - Remove tag: reads the "value" field to detect i) if a tag exists and ii) if any of the selected vertices and edges has been assigned such tag. If the tag exists, it will be set to False for all the selected vertices and edges.

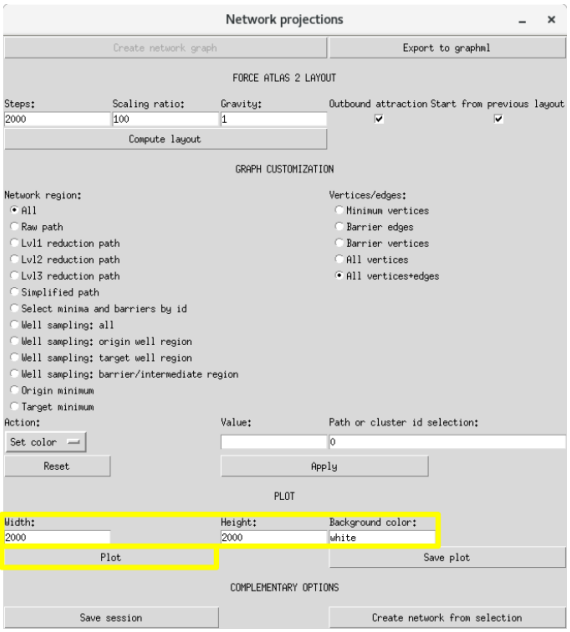

Figure 43: Location of the width, height and background color options as well as the plot button in the network projections window.

Use the "apply" button to apply the chosen action to the current selection, use the "reset" button to reset graph customization to its default values and use the "plot" button to see the current status of the graph (Fig. 43).

Width, height and background color can be set before plotting (Fig. 43). Note that "none" and "None" are valid background colors and will produce graphs with transparent background.

For step-by-step examples see section 7.9.

Saving to png and/or svg

The "save plot" button (Fig. 44) allows to save the graph plot to a file. The file format is inferred from the extension. In other words, to save a PNG file add the extension .png after the chosen file name and to save SVG file use the .svg extension instead.

Width, height and background color can be set before saving plots (Fig. 44). Note that "none" and "None" are valid background colors and will produce graphs with transparent background.

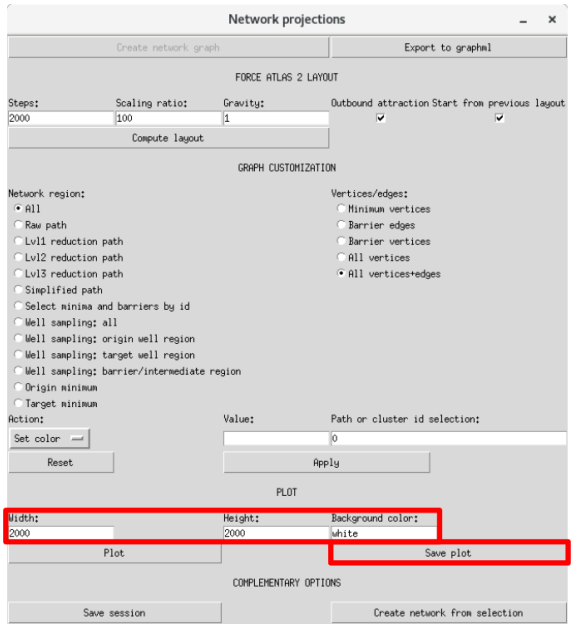

Figure 44: Location of the width, height and background color options as well as the save plot button in the network projections window.

Exporting to graphml

MEPSAnd supports exporting the created graphs to the graphml format, so that they can be handled with third party software, such as Gephi (https://gephi.org/). It is possible to add tags to set of points (see "add tag" and "remove tag" actions in section 7.7.3) that can be used externally to easily identify relevant regions of the graph. Also, every vertex and edge has an attribute "energy" containing the energy of the cluster it represents, which can also be used externally. Nevertheless, note that not all the attributes customized in MEPSAnd can be exported in the graphml format. The most notable aspect is the layout, that is not part of the information contained in this format. In fact, for this reason, MEPSAnd allows the user to export the graph right after its creation. However, it is worth mentioning that, for example, in gephi this layout can be easily computed in real time.

To export a graph to graphml format, use the "export to graphml" button (Fig. 45).

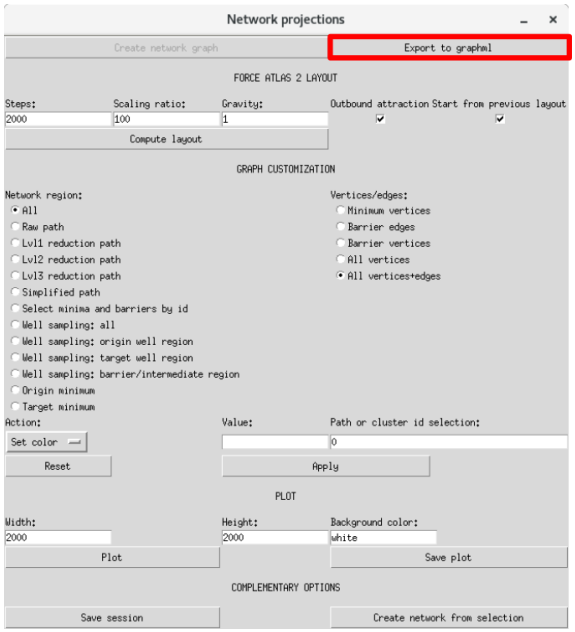

Figure 45: Location of the export to graphml button in the network projections window.

Creating a new network from selection

MEPSAnd allows the user to create a new network only containing a selection of vertices and edges. After selecting a range of vertices and edges (see how the selections system work in section 7.7.3) the "create network from selection" button will replace the current network with the new truncated one. Please note that this step is not reversible and saving session before is highly recommended (see section 10).

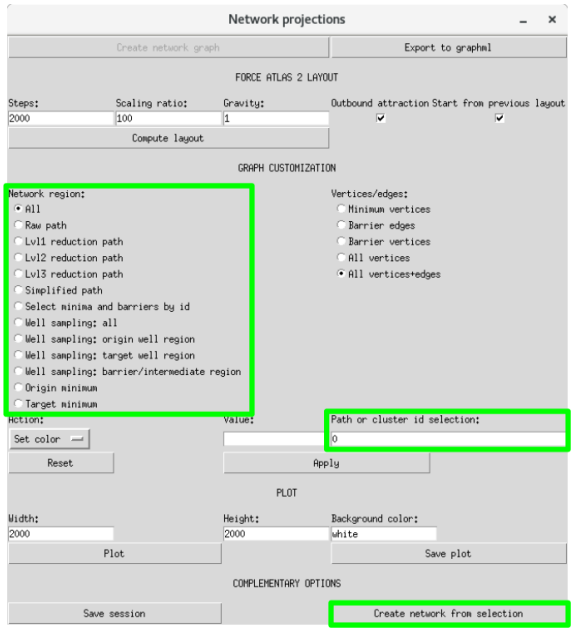

Figure 46: Location of the create network from selection button in the network projections window.

For step-by-step examples see section 7.9.

7.8 MEPSAnd python scripting

To support GUI-independent pipelines, MEPSAnd.py can be directly imported from a regular python 3 script. Once imported, the user may load a surface file to a "map_handler" object instance or restore a previously saved session.

As an example, this code would read a surface file and, if correctly loaded, compute the global network, minimum energy path and save session:

```
import MEPSAnd as mps

inputfile = "~/test_files/test_surface.dat" # just the path to a valid surface
    ↪ file

a = mps.map_handler(inputfile) #load a surface file to a map_handler instance

if a.good: #Check if the file has been sucessfully read
       a.get_global_network() #map_handler method to compute global network
       a.save_session("~/test_files/test_session.mepsand")
```

As an example, this code would load a previously saved session:

```
import MEPSAnd as mps

inputfile = "~/test_files/test_session.mepsand" # just the path to a valid
    ↪ session file

a = mps.load_object(inputfile) #load a session
```

The "map_handler" object offers a range of methods that can be called to perform all the basic calculations, such as connectivity computation and path-finding, etc. These methods are:

- set_diagonals(diagonals):
  - Description: sets the value of the diagonals boolean variable.
  - Arguments:
    - diagonals:True allows the use of diagonals for path reduction. Default(False).
- load_cutoffs(filepath):
  - Description: reads a set of cutoffs from a file.
  - Arguments:
    - filepath: string containing the path to the file to be read.
- save_cutoffs(filepath):
  - Description: saves current cutoffs to a file.
  - Arguments:
    - filepath: string containing the path to the file to be created.
- load_periodicity(filepath):
  - Description: reads a periodicity file.
  - Arguments:
    - filepath: string containing the path to the file to be read.
- save_periodicity(filepath):

  - Description: saves current periodicity to a file.
  - Arguments:
    * filepath: string containing the path to the file to be created.
- load_connectivity(filepath):
  - Description: reads a connectivity file.
  - Arguments:
    * filepath: string containing the path to the file to be read.
- save_connectivity(filepath):
  - Description: saves current connectivity to a file.
  - Arguments:
    * filepath: string containing the path to the file to be created.
- get_global_network():
  - Description: calculates the global network over which most of MEPSAnd calculations are performed. If not explicitly invoked, this calculation will take place the first time it is required by any MEPSAnd method.
  - Arguments: None.
- set_OT(OT):
  - Description: defines the points to be origin and target
  - Arguments:
    * OT: a list containing two integer values indicating the indices of the origin and target points. First element will be interpreted as the origin point index and second element as the target point index. To leave either origin or target unchanged, set the corresponding list element to None.
- get_OT():
  - Description: returns the current list of indices defining origin and target points. First element of the list represents the origin point index and the second element the target point index.
  - Arguments: None.
- select_origin_by_range(coord_range,minimum):
  - Description:
  - Arguments:
    * coord_range: array of coordinate range pairs with shape = (number_of_coords, 2).
    * minimum: boolean variable. If true, only minimum cluster points would be considered.
- select_target_by_range(coord_range,minimum):
  - Description:
  - Arguments:
    * coord_range: array of coordinate range pairs with shape = (number_of_coords, 2).
    * minimum: boolean variable. If true, only minimum cluster points would be considered.

- select_origin_by_minimum_id(index):
  - Description: Select as origin a point of the minimum cluster with the given index.
  - Arguments:
    - index:integer indicating the index of the minimum cluster from which to select origin point.
- select_target_by_minimum_id(index):
  - Description: Select as target a point of the minimum cluster with the given index.
  - Arguments:
    - index:integer indicating the index of the minimum cluster from which to select target point.
- get_npaths(n):
  - Description: computes the minimum energy path and sub-optimal alternative paths.
  - Arguments:
    - n: last path to calculate. Minimum energy path is path 0, so if 0 is passed, only the minimum energy path is calculated.
- get_well_sampling():
  - Description: computes the well sampling.
  - Arguments: None.
- save_minbar_clusters_to_txt(filepath):
  - Description: saves minimum and barrier clusters in plain text format.
  - Arguments:
    - filepath: path to the file to be created.
- save_simplified_path_to_txt(filepath,n):
  - Description: saves the simplified path in plain text format.
  - Arguments:
    - filepath: path to the file to be created.
    - n: index of the path to be saved. 0 is the index of the minimum energy path.
- save_fragmentwise_path_to_txt(filepath,n):
  - Description: saves the fragmentwise path in plain text format.
  - Arguments:
    - filepath: path to the file to be created.
    - n: index of the path to be saved. 0 is the index of the minimum energy path.
- save_well_sampling_to_txt(filepath):
  - Description: saves well sampling in plaint text format.
  - Arguments:
    - filepath: path to the file to be created.
- save_session(filepath):
  - Description: saves the current instance of the "map_handler" object to a session file.
  - Arguments:

* filepath: path to the file to be created.

7.9 Examples

In this section, a series of surfaces with different complexity and number of dimensions are analyzed with MEPSAnd step-by-step. All the surfaces analyzed in these examples can be downloaded from the MEPSAnd page (http://bioweb.cbm.uam.es/software/MEPSAnd/) so that new users can use this examples as quick start tutorials.

It is highly recommended to follow the first example (see section 7.9.1) before trying any other one, as it introduces the most basic mechanics of the program.

7.9.1 3 minima surface: a step-by-step introduction to MEPSAnd general features

- Go to http://bioweb.cbm.uam.es/software/MEPSAnd/.
- Download the file "examples.zip".
- Extract the "3_min_surf.dat" from "examples.zip".
- Run MEPSAnd.py following the instruction shown in chapter 5.
- Use the "load surface" button, find "3_min_surf.dat" and load it. It is a 3D surface in which the 2 coordinates are uniformly distributed along a grid with a step length of 0.25 arbitrary units.
- The automatically detected cutoffs for both coordinates should be:
  - Lower cutoff: 0.0125 (i.e. 0.25 + 0.5)
  - Upper cutoff: 0.0375 (i.e. 0.25 + 1.5)
- Use the "compute connectivity" button.
- Select "no" when asked if diagonals should be allowed for path reduction.
- Use the "save minimum and barrier clusters" button. The resulting file should look like this:

```
#Minimum 0
0 3.3 1.775 -3.58
#Minimum 1
1 1.775 3.3 0
#Minimum 2
2 3.3 3.3 7.87
#Barrier 3
3 2.375 2.45 29.2
#Barrier 4
4 3.1 2.725 32.04
#Barrier 5
5 2.7 3.075 32.61
```

  This file is telling us that all the minimum and barrier clusters obtained are only populated by a single point. This file is typically useful to select the origin and target regions by minimum cluster indices.
- Open the path-finding pipeline window.
- First lets choose an origin point, for example using coordinate ranges.
  - Use the "set origin" button.
  - In the set origin window use the "find origin by coordinate ranges".

  - Lets assume that you are interested in finding the lowest minimum cluster in the formed by points which are located between 1.5 and 2 for coordinate 0 and 2 and 3.5 for coordinate 1. Modify the values shown to look something like this:

```
 Dimension_id  Range_lower_limit  Range_upper_limit
            0                1.5                  2
            1                  2                3.5
```

  - To ensure that the parsing of this text is going to be correctly interpreted by MEPSAnd, use the "read origin coordinate ranges". It will reprint the coordinate ranges text using what MEPSAnd has interpreted. If you agree with the values that are printed back after this check, use the button "set origin as the lowest minimum in the coordinate ranges".

  - In the path-finding pipeline window, next to the "set origin button" the text "963(1)" should appear. It is indicating that the currently selected origin point is the point with index 963 that is assigned to the minimum cluster with index 1. If you click on this text, a window showing the coordinates of the chosen origin point will pop-up.

- Now lets assume that after seeing the minimum and barrier clusters file previously saved you decided that the minimum cluster with index 2 is an adequate target.

  - Use the "set target" button.

  - In the set target window write the chosen index (in this case 2) in the "Point/Minima ID:" field.

  - Use the "set target by minimum index" button.

  - In the path-finding pipeline window, next to the "set target button" the text "5904(2)" should appear. It is indicating that the currently selected origin point is the point with index 5904 that is assigned to the minimum cluster with index 2. If you click on this text, a window showing the coordinates of the chosen origin point will pop-up.

- After defining the origin and target points, if they are assigned to different minimum clusters, the buttons in the calculations section become active.

- Use the "minimum energy path" button to calculate the minimum energy path.

- Use the "well sampling" button to calculate the well sampling.

- Use the "alternative paths" button to calculate as many alternative paths as indicated in the field next to the button. Note that this is cumulative and that when no more paths can be found, the button will do nothing. In other words, if you are interested in calculate as many alternative paths as MEPSAnd can find, just use a very large number and the program will automatically stop when no more paths can be found. In this particular example only one alternative path can be found. Note, that, if minimum energy path has not been calculated, MEPSAnd will calculate it before calculating any alternative paths.

- After performing these calculations, the "total paths calculated" counter should be 2 and all the buttons in the "save to disk" section should be active.

- To plot the energy evolution of the calculated paths use the "energy profiles" button. There are many path reductions available. Lets start with the "raw path":
  - You can plot the minimum energy path by selecting index 0 (Fig. 47).

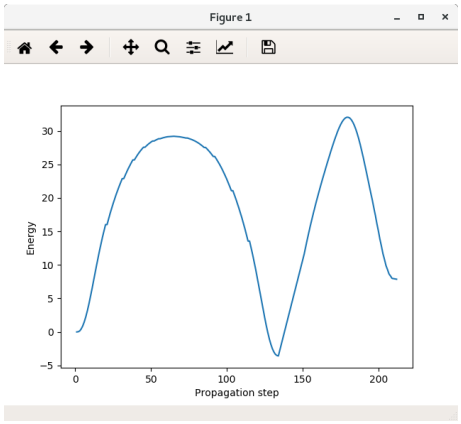

Figure 47: Minimum energy path raw path energy profile.

  - You can plot the alternative path by selecting index 1 (Fig. 48).

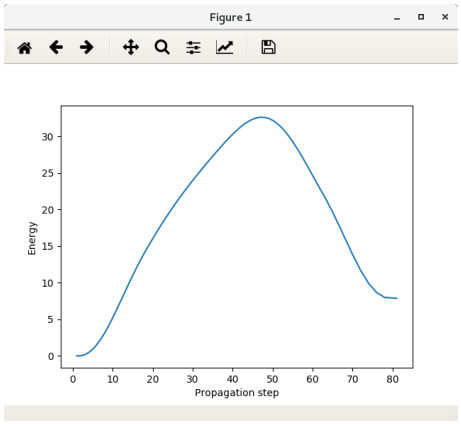

Figure 48: Alternative path 1 raw path energy profile.

  - You can plot both paths simultaneously by selecting 0-1 or all(Fig. 49).

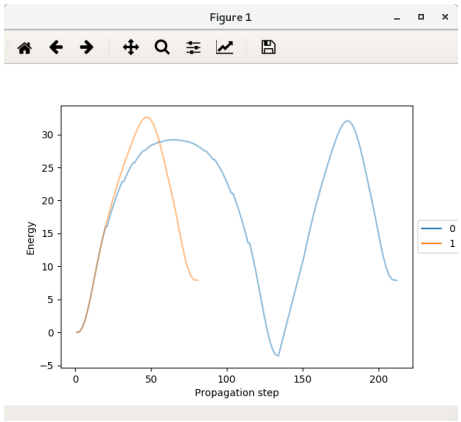

Figure 49: Minimum energy path (0) and alternative path 1 (1) raw path energy profiles.

- Repeat the same process with the different path reductions available.

- To project the calculated paths over a set of coordinates use the "coordinate projections" button, that will open the coordinate projections button. As in this example we are using a 3D surface, the contour plot is available.
- Set "Y coordinate" to 1 and, for example, change "origin points size" and "target points size" to 20, so that origin and target points are easier to visualize.
- Use the "contour plot" button, select raw path and path 0 (Fig. 50)

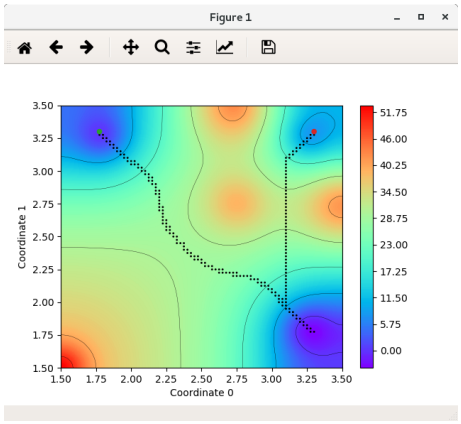

Figure 50: Minimum energy path projection over coordinates 0 and 1 using a contour background.

- Use the "plot" button following the same steps to obtain a similar plot without the contour background (Fig. 51)

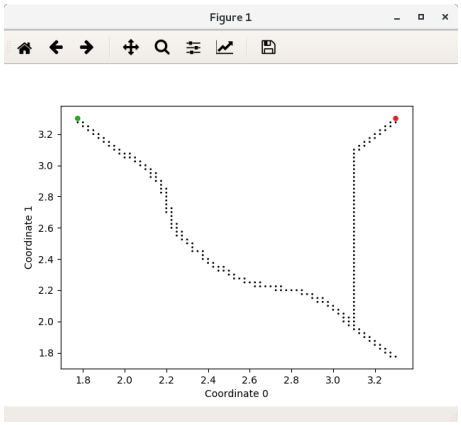

Figure 51: Minimum energy path projection over coordinates 0 and 1.

- Now we are going to project all paths at once. As this representation uses semi-transparent points to facilitate the sight of path overlaps we are going to increase the point size. Set the "points size" field to 10.

- Use the "contour plot" button using "all" paths to select paths in order to plot all the computed paths over a contour (Fig 52).

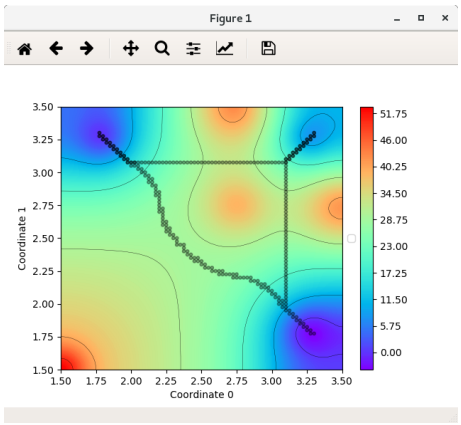

Figure 52: Projection of all the paths calculated over coordinates 0 and 1 using a contour background. The minimum energy path has index 0 and the only alternative path has index 1.

- Use the "plot" button using "all" paths to select paths in order to plot all the computed paths over a white background (Fig 53).

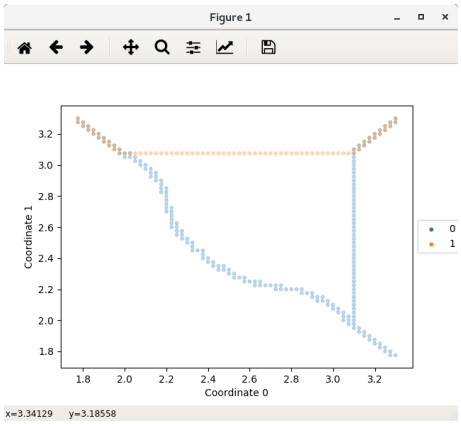

Figure 53: Projection of all the paths calculated over coordinates 0 and 1.

- Set "Z coordinate" to "energy" and use the "plot" button to see an example of a 3D projection (Fig. 54).

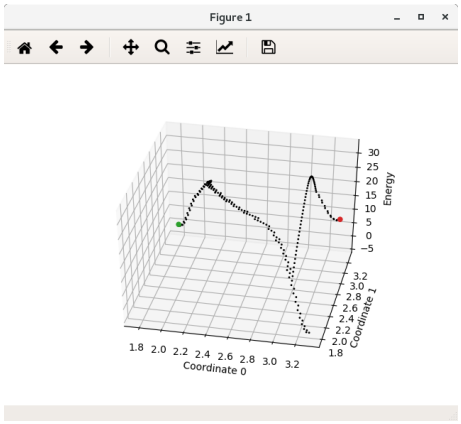

Figure 54: Minimum energy path projection over dimensions 0, 1 and energy.

- Set all point sizes fields to 1, set "Y coordinate" to 1 and set "Z coordinate" to "None".
- Use the "contour plot" button and select "well sampling: all" to project all the well sampling regions over dimensions 0 and 1 (Fig. 55).

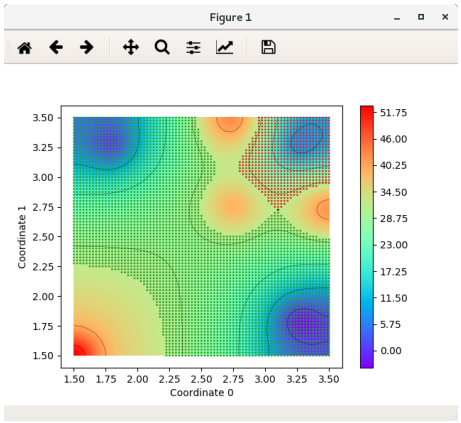

Figure 55: Well sampling regions projection over coordinates 0 and 1 using a contour background. Green points represent the origin well region, red points the target well region and black points the barrier/intermediate region.

- Set "Z coordinate" to "energy" and use the "plot" button to see an example of a 3D projection of the well sampling regions (Fig. 54).

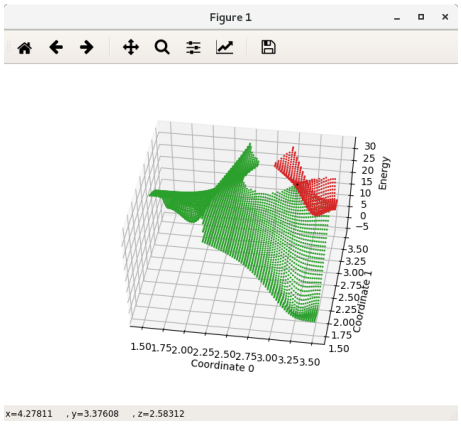

Figure 56: Well sampling regions projection over dimensions 0, 1 and energy. Green points represent the origin well region, red points the target well region and black points the barrier/intermediate region.

- Now lets comment path reductions a bit further.
- Set "Z coordinate" to "None", "points size" field to 1, "origin points size" field to 10 and "target points size" field to 10.
- Using the contour plot button, plot the raw path (Fig. 57), lvl1 reduction (Fig. 58), lvl2 reduction (Fig. 59), lvl3 reduction (Fig. 60) and simplified path (Fig. 61) versions of the minimum energy path (path index 0).
- Note that with a surface with such a simple topology, there is no difference between reduction levels 1 to 3. Also note how the simplified path is no longer forced to explicitly sample every minimum cluster along the path.
- If diagonal connectivity is allowed during connectivity definition, such connectivity is used for path reductions. All the reduced paths obtained up to this point have used regular connectivity on the reduction process. The next 4 path reduction projections (Fig. 62,63,64,65) have been obtained using diagonal connectivity. If you want to try to get them yourself open a new MEPSAnd instance, load the "3_min_surf.dat" file and, after pressing "compute

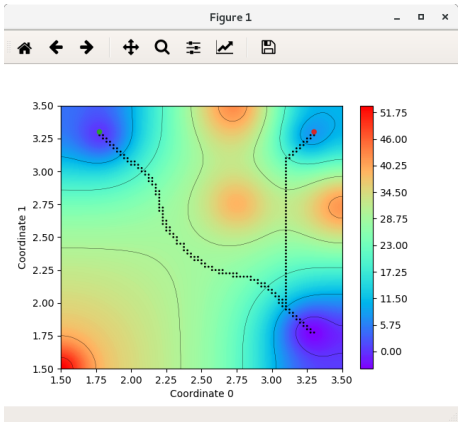

Figure 57: Minimum energy path (raw) projection over coordinates 0 and 1 using a contour background.

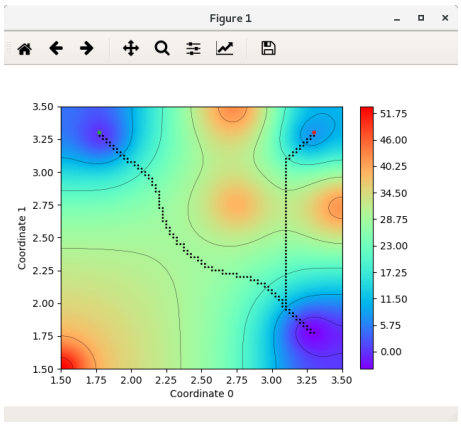

Figure 58: Minimum energy path (level 1 reduction) projection over coordinates 0 and 1 using a contour background.

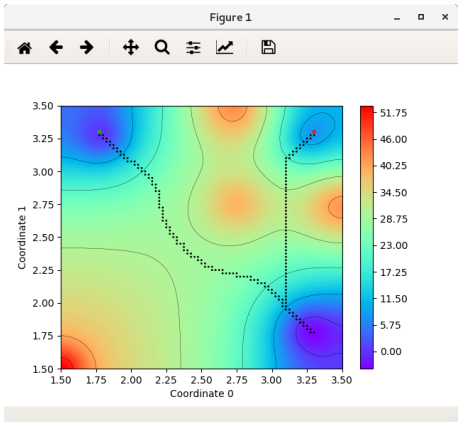

Figure 59: Minimum energy path (level 2 reduction) projection over coordinates 0 and 1 using a contour background.

connectivity" button, allow the use of diagonal connectivity in the pop-up window. Then reproduce all the other steps as indicated up to this point.

- We are now going to introduce the use of the network projections environment. Close the coordinate projections window and use the "network projections" button.
- Once the network projections window is open, save session, as we will now choose the type of network to generate and it is not a reversible choice.

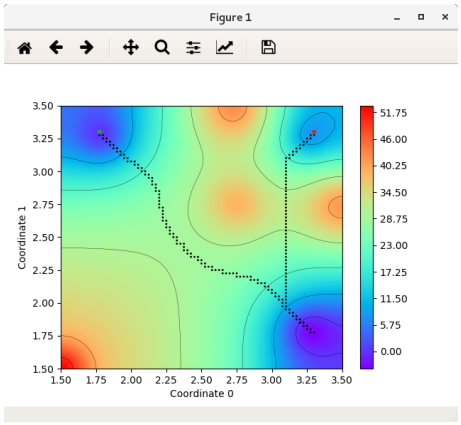

Figure 60: Minimum energy path (level 3 reduction) projection over coordinates 0 and 1 using a contour background.

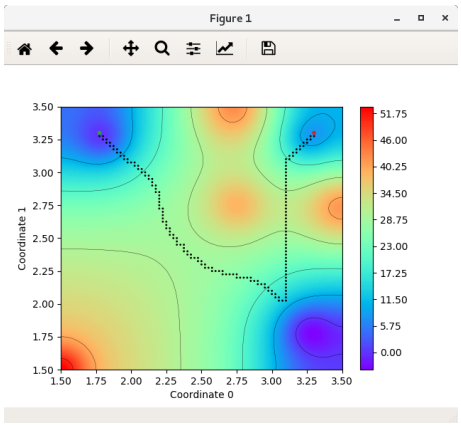

Figure 61: Minimum energy path (simplified) projection over coordinates 0 and 1 using a contour background.

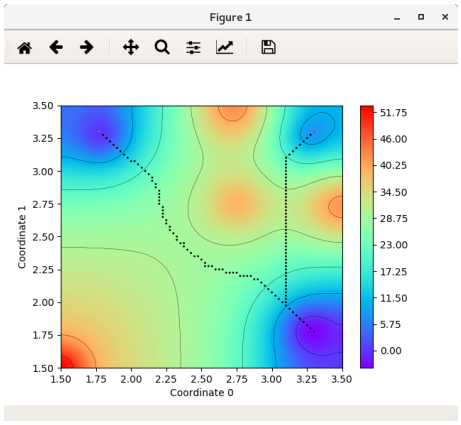

Figure 62: Minimum energy path (level 1 reduction with diagonal connectivity) projection over coordinates 0 and 1 using a contour background. Compare with figure 58.

- Use the create network graph and choose "yes" when asked whether to represent barriers as vertices.
- Use the default MEPSAnd values for the Force Atlas 2 algorithm and use the "compute layout" button.
- Use the "plot" button to see the graph plot with the default parameters (Fig. 67).

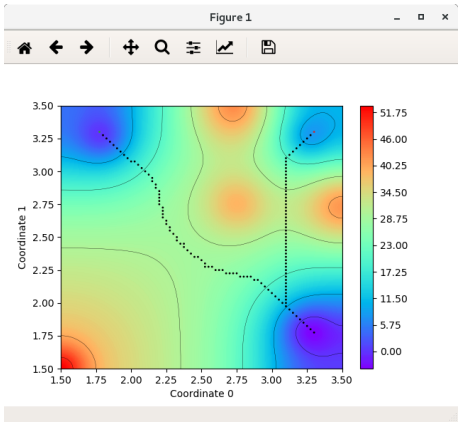

Figure 63: Minimum energy path (level 2 reduction with diagonal connectivity) projection over coordinates 0 and 1 using a contour background. Compare with figure 59. Also, this is a good example of actual path reduction versus the level 1.

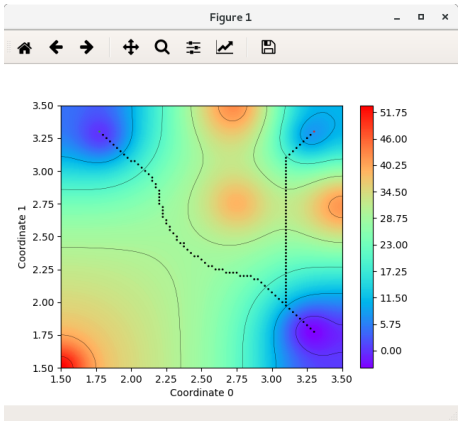

Figure 64: Minimum energy path (level 3 reduction with diagonal connectivity) projection over coordinates 0 and 1 using a contour background. Compare with figure 60.

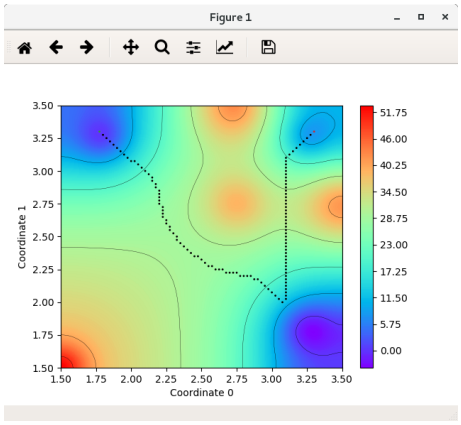

Figure 65: Minimum energy path (simplified with diagonal connectivity) projection over coordinates 0 and 1 using a contour background. Compare with figure 61.

- Lets increase the vertex size.
- Set the following:
  - Network region: all
  - Vertices/edges: all vertices

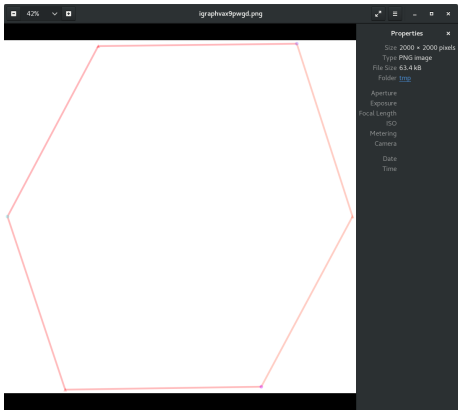

Figure 66: Outcome of the "plot" button on a Centos 7 linux distribution. Note that mininimum clusters vertices are represented by circles and barrier clusters vertices by triangles.

  - Action: set size
  - Value: 50
- Click on the "apply" button.
- Lets mark origin and target with thick and opaque frames.
- Set the following:
  - Network region: origin minimum
  - Vertices/edges: all vertices
  - Action: set frame width
  - Value: 10
- Click on the "apply" button.
- Set the following:
  - Network region: target minimum
  - Vertices/edges: all vertices
  - Action: set frame width
  - Value: 10
- Click on the "apply" button.
- Set the following:
  - Network region: origin minimum
  - Vertices/edges: all vertices
  - Action: set frame alpha
  - Value: 1
- Click on the "apply" button.
- Set the following:
  - Network region: target minimum
  - Vertices/edges: all vertices
  - Action: set frame alpha
  - Value: 1

- Click on the "apply" button.
- Lets annotate barrier vertices with their energy value.
- Set the following:
  - Network region: all
  - Vertices/edges: barrier vertices
  - Action: set cluster energy as labels
  - Value: it is ignored
- Click on the "apply" button.
- Set the following:
  - Network region: all
  - Vertices/edges: barrier vertices
  - Action: set label size
  - Value: 50
- Click on the "apply" button.
- Lets mark the minimum energy path as an opaque region with thicker edges.
- Set the following:
  - Network region: raw path
  - Vertices/edges: all vertices+edges
  - Action: set alpha
  - Value: 1
- Click on the "apply" button.
- Set the following:
  - Network region: raw path
  - Vertices/edges: barrier edges
  - Action: set width
  - Value: 20
- Click on the "apply" button.

- Use the "plot" button to see the outcome of these customizations (Fig. 67).

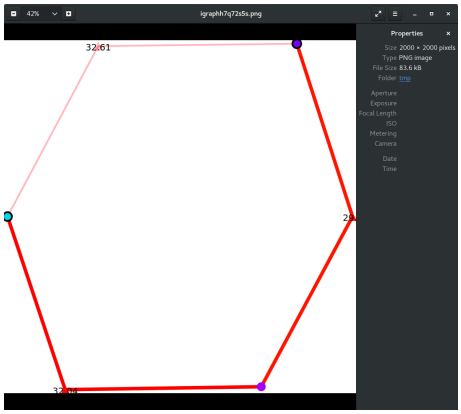

Figure 67: Outcome of the "plot" button showing the customized graph on a Centos 7 linux distribution.

- Sometimes, when working with small graphs, overlaps with the figure box edges may occur. A simple solution is to export the graph plot to an .svg file via the "save plot" button and edit with third party software (Fig. 68).

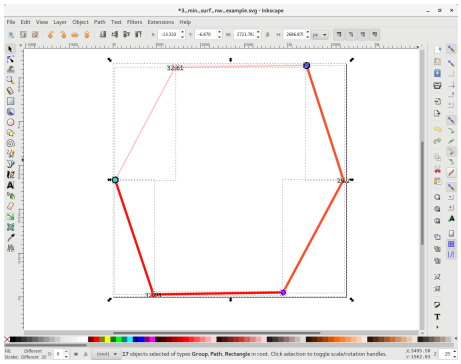

Figure 68: Using inkscape (`https://inkscape.org/`) to edit the resulting graph plot. Not that all the elements of the plot are individually editable.

- Open a new instance of MEPSAnd.
- Load the session saved before graph creation via the "load session" button.
- Open the path-finding pipeline window and, from it, the network projections window.
- Use the "create network" button but, this time, choose not to represent barriers as vertices.
- Use the "plot" button and compare the resulting plot with the obtained when representing barriers as vertices.

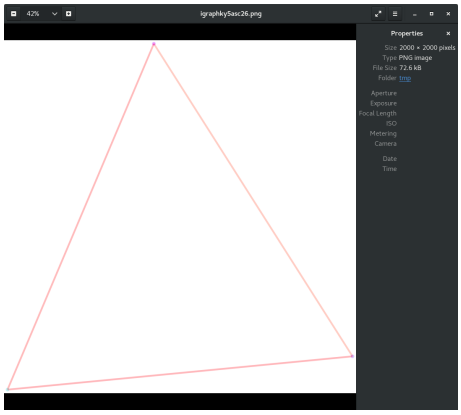

Figure 69: Outcome of the "plot" button showing the graph without barriers as vertices on a Centos 7 linux distribution.

7.9.2 Geneva-Turin: alternative paths and network projections on a complex 3D surface

It is highly recommended to follow the first step-by-step example (7.9.1) before checking this one, as the first example is closer to a basic tutorial, while the aim of the rest of the examples is to show different features of MEPSAnd in a graphic manner to suggest ideas that might be useful for users with a varied range of data sets to analyze.

In this example the surface file analyzed is "Genenva_Turin.dat" (Fig. 70) from the "examples.zip" file that can be downloaded from http://bioweb.cbm.uam.es/software/MEPSAnd/.

This case is a continuation of an example presented in the classic MEPSA manual making use of the MEPSAnd new features, which now is capable of fluently analyze this surface in a more informative fashion. The surface comes from a set of topographic data extracted from the SRTM (Shuttle Radar Topography Mission) 30 ARC-SECOND ELEVATION v2.1 data set through the ORNL DAAC OGC Spatial Data Access tool (http://webmap.ornl.gov/wcsdown/dataset.jsp?ds_id=10008).

The intention of using of this kind of data is to reinforce the idea that, no matter the origin, if a set of coordinates can be related to a valule that has to be minimized using a transtion state theory persepective, MEPSAnd can handle it. This is the last 3D example presented here as, in the next two examples, datasets with 4 and 5 dimensions are analyzed.

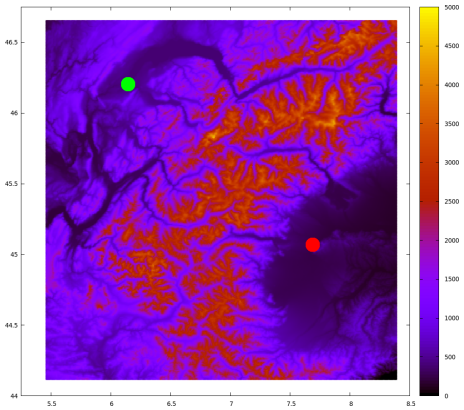

Figure 70: Geneva-Turin surface plotted with gnuplot. Google maps coordinates for Geneva (green) and Turin (red) are shown.

Using coordinate projections, the location of the minimum clusters can be visualized (Fig. 71), helping, for example, to choose meaningful origin and target regions. Note that lake Geneva is a large minimum cluster on its own, as the satelite data inidicates the same elevation along the whole lake.

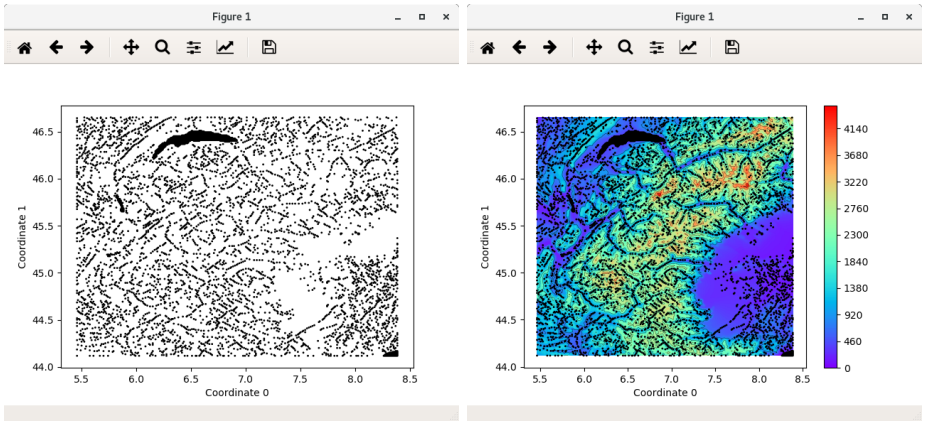

Figure 71: Minimum clusters locations over white and contour backgrounds.

In this example the origin (Geneva) point chosen was 18035 (6.152663,46.226224) and the target (Turin) was 65045 (7.769330,45.117891). Note that this indices are the ones used by MEPSAnd. Add +1 to identify the line in the input surface file. Using these settings the minimum energy path (Fig. 72) and 392 alternative paths (Fig. 73) can be obtained.

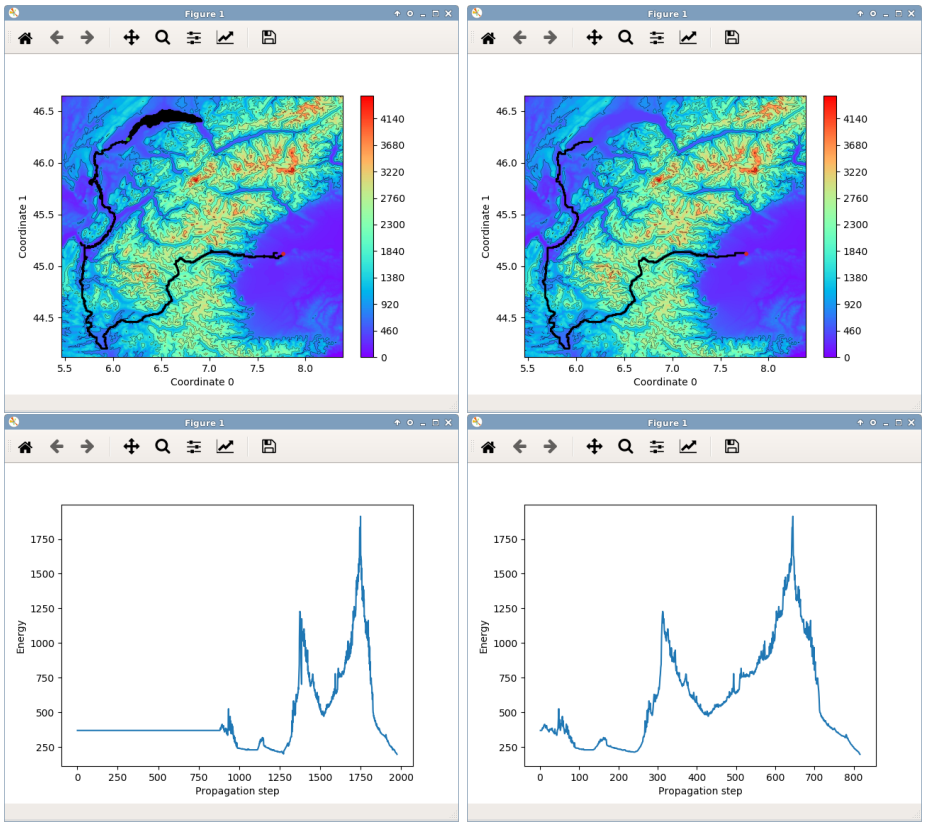

Figure 72: Minimum energy path. Upper pannels show the path projections over both coordinates using a contour background and lower pannels indicate the path profiles. Left pannels are showing the raw fragmentwise paths and right pannels the simplified paths.

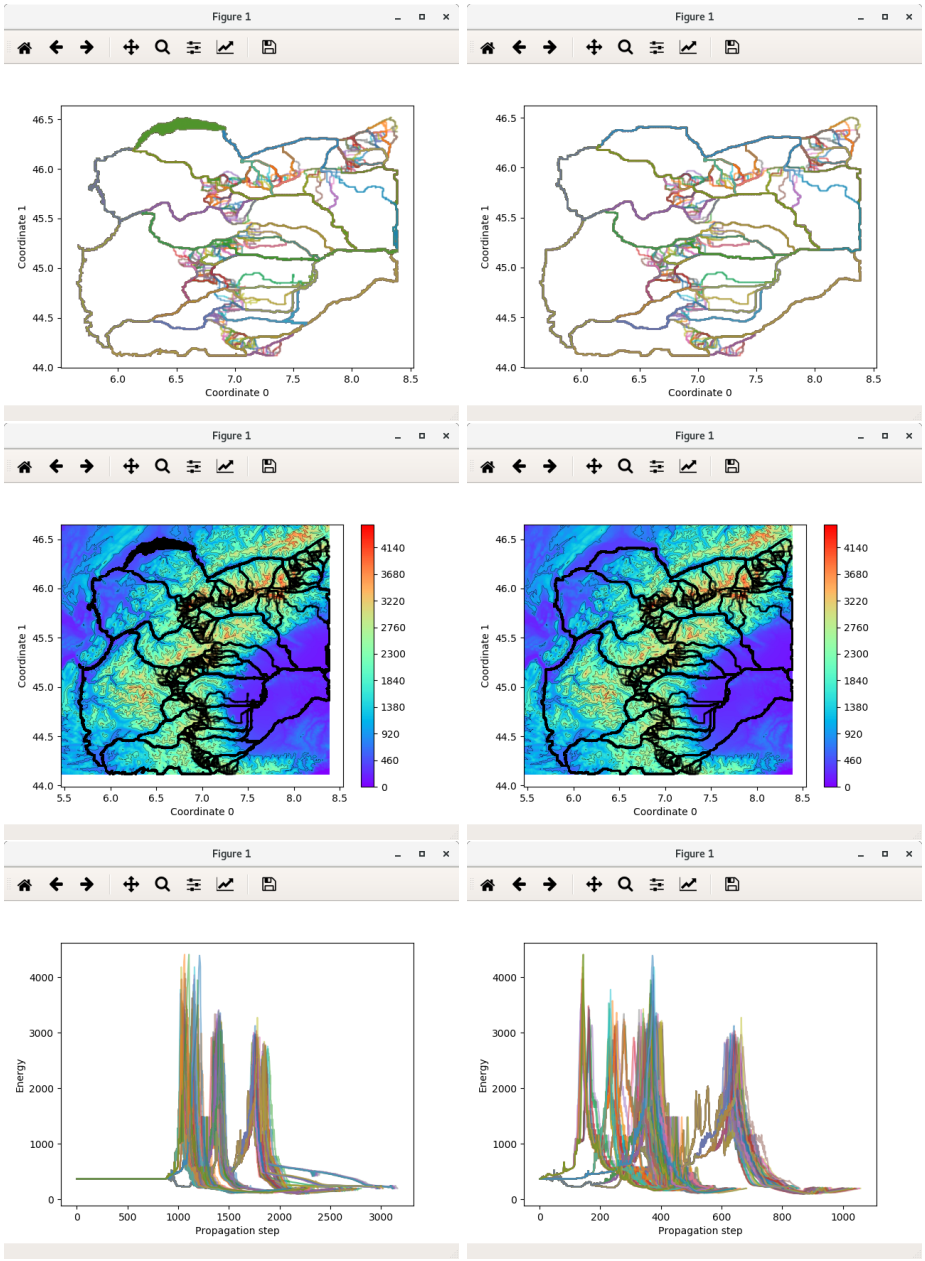

Figure 73: Minimum energy path and 392 alternative paths. Upper pannels show the path projections over both coordinates using a white background, middle pannels show the path projections over both coordinates using a contour background and lower pannels indicate the path profiles. Left pannels are showing the raw fragmentwise path and right pannels the simplified path.

After origin and target definition, well sampling can be calculated (Fig. 74).

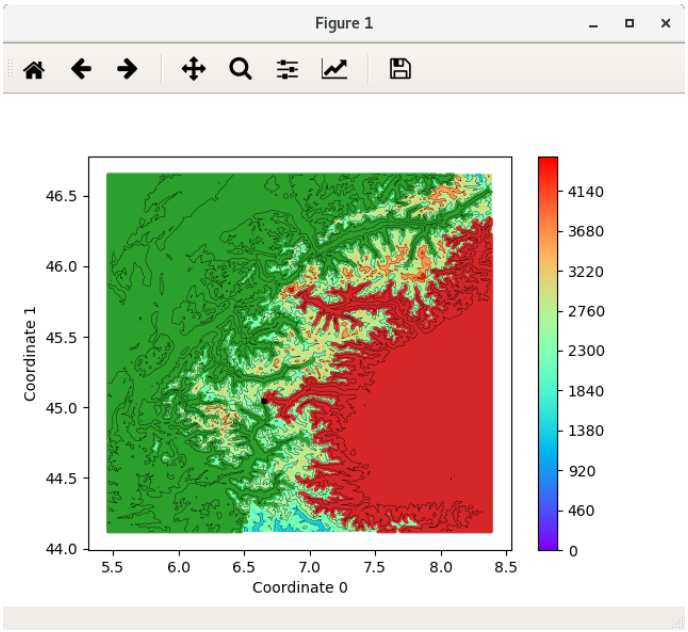

Figure 74: Well sampling. Green denotes the origin well region, red the target well region and black the barrier/intermediate region.

Given the complexity of the global network generated by this surface, the network projection was calculated representing barriers only as edges (Fig. 75).

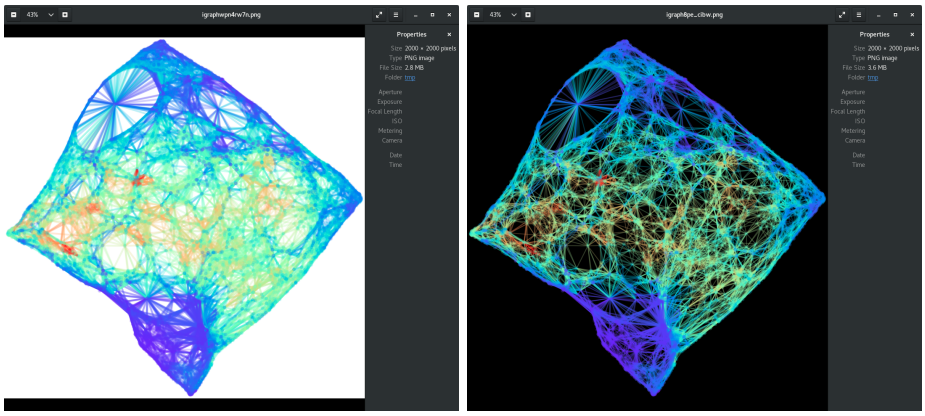

Figure 75: Network projection over white (left) and black (right) backgrounds.

Once the graph was generated, calculation results could be represented in a wide range of ways (e.g. Figs. 76, and 77).

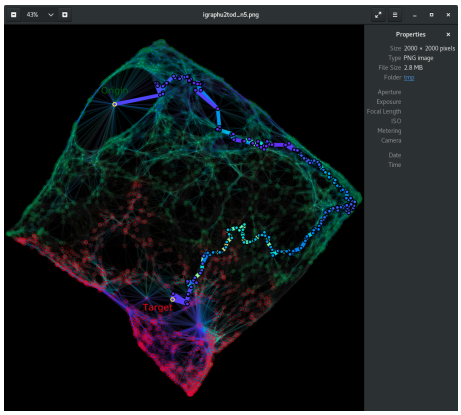

Figure 76: Network projection over black background. Points belonging to the origin well (green) and target well (red) regions of the well sampling are annotated by thin circles (frame width > 0). The minimum energy path is represented on top (bring to front), using opaque points and edges (alpha = 1) and colored using an energy-based colormap. Labels have been added to origin and target points.

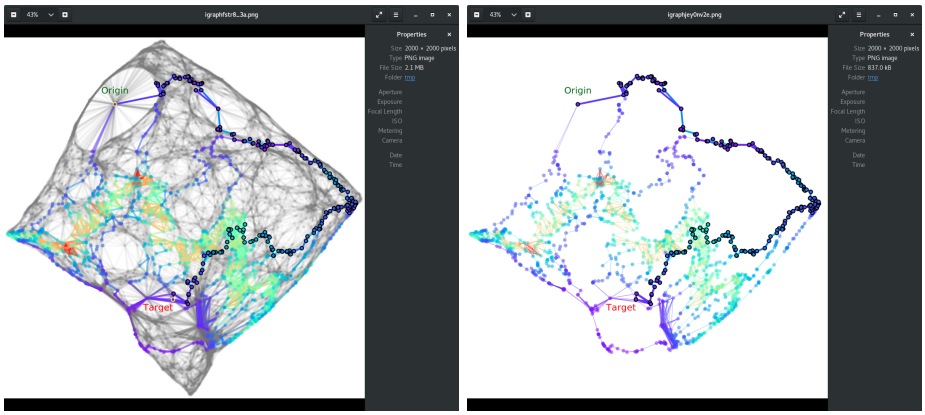

Figure 77: Network projections over white background. Points belonging to any of the 393 calculated path are colored according to their energy level. The minimum energy path points are marked by black frames. The region of the network that does not belong to any calculated path is colored gray (left) or not represented (right).

In addition to the full global network, truncated networks built from selections can be used to reduce the complexity of the data presented and focus the viewer attention on the most relevant features of the surface (e.g. Figs. 78 and 79).

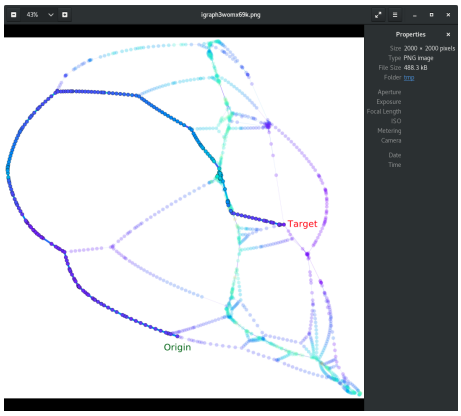

Figure 78: Graph plot of the network formed by points participating in any of the 393 calculated paths. The graph is colored according to the energy levels of its components and the minimum energy path is plotted on top, opaque and with thin black frames. Labels indicate origin and target points. Note that the plot shows a high-energy region (the Alps) that has to be crossed to reach target from origin.

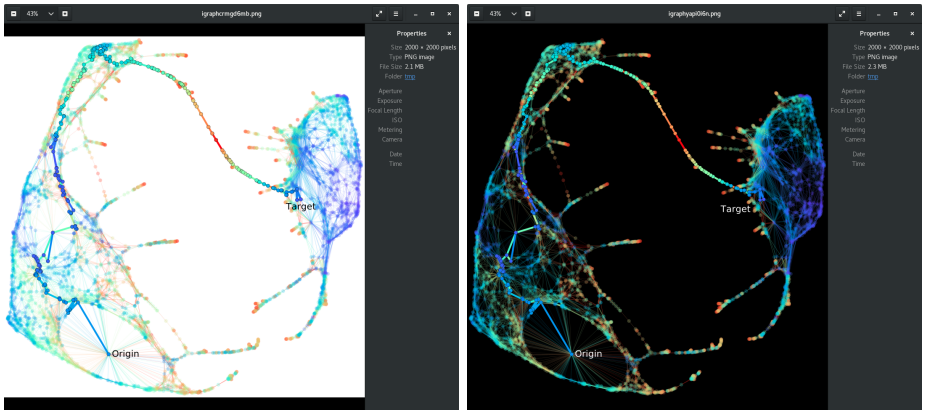

Figure 79: Graph plot of the network formed by points participating in the well sampling over white (left) and black (right) backgrounds. The graph is colored according to the energy levels of its components and the minimum energy path is plotted on top, opaque and with thin black frames. Labels indicate origin and target points. Note that minimum plateau regions, such as lakes in this case (e.g. the origin), form distinctive shapes in which a single point is connected to a high number of other vertices.

7.9.3 Displaced sine cubes: a 4D surface

It is highly recommended to follow the first step-by-step example (7.9.1) before checking this one, as the first example is closer to a basic tutorial, while the aim of the rest of the examples is to show different features of MEPSAnd in a graphic manner to suggest ideas that might be useful for users with a varied range of data sets to analyze.

In this example the surface file analyzed is "displaced_sine_cubes.dat" (Fig. 80) from the "examples.zip" file that can be downloaded from http://bioweb.cbm.uam.es/software/MEPSAnd/.

It is a an artificially created surface to illustrate some of the characteristics of MEPSAnd.

The Perl script used to create "displaced_sine_cubes.dat" is:

```
#!/usr/bin/perl
use warnings;
use strict;
use Math::Trig;

my $x;
my $y;
my $z;
my $energy;


for ($x = -10; $x <= 10; $x+=0.5)
{
        for ($y = -10; $y <= 10; $y+=0.5)
        {
                for ($z = -10; $z <= 10; $z+=0.5)
                {
                        $energy = abs($x) * abs(sin($x)) + abs($y) * abs(sin($y)) +
                            ↪  abs($z) * abs(sin($z));
                        print ("$x␣$y␣$z␣$energy\n");
                        print ( "$x␣" . ($y+20.5) . "␣" . ($z+5) . "␣" . ($energy
                            ↪ -10) ."\n");
                        print ( ($x+20.5) ."␣$y␣$z␣" . ($energy+10) ."\n");
                        print ( ($x+20.5) . "␣" . ($y+20.5) . "␣$z␣" . $energy . "\
                            ↪ n");
                }
        }
}
```

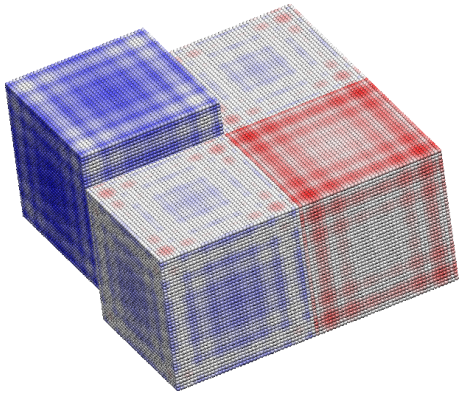

Figure 80: 3D plot of "displaced_sine_cubes.dat". Color scale relates to energy values (red: high; blue: low).

Using the fist (0) and last (275683) points of the surface as origin and target respectively, the raw fragmentwise and the simplified paths can be plotted in several ways: energy profiles (Fig. 81), coordinate projections (Fig. 82) and network projections (Fig. 83).

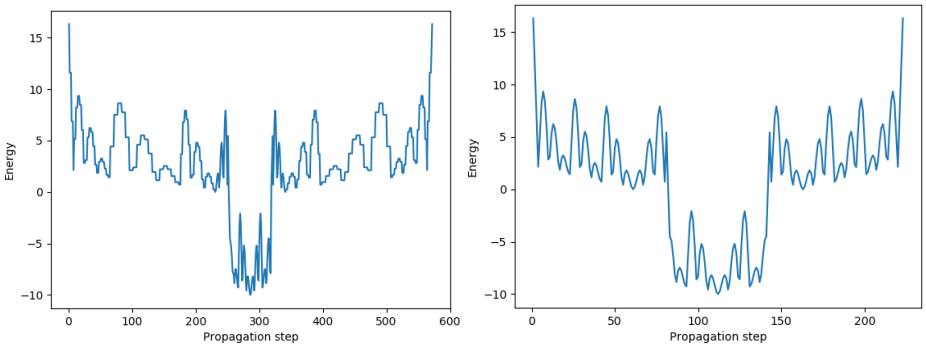

Figure 81: Energy profiles along the raw fragmentwise path (left) and the simplified path (right).

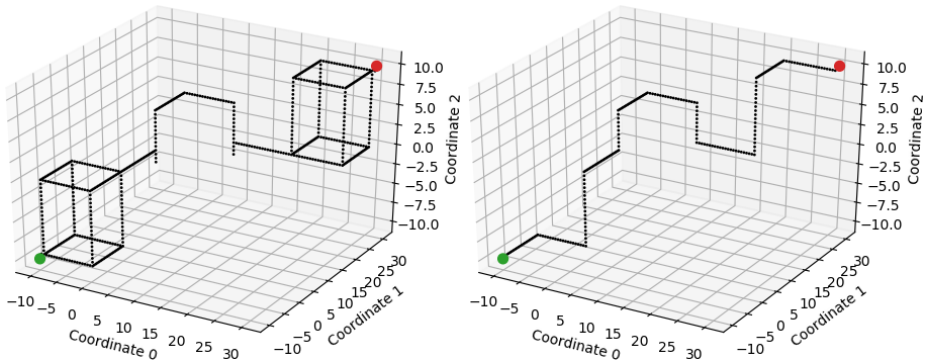

Figure 82: Raw fragmentwise path (left) and the simplified path (right) projection over coordinates 0, 1 and 2. Origin and target are indicated by the green and red points respectively.

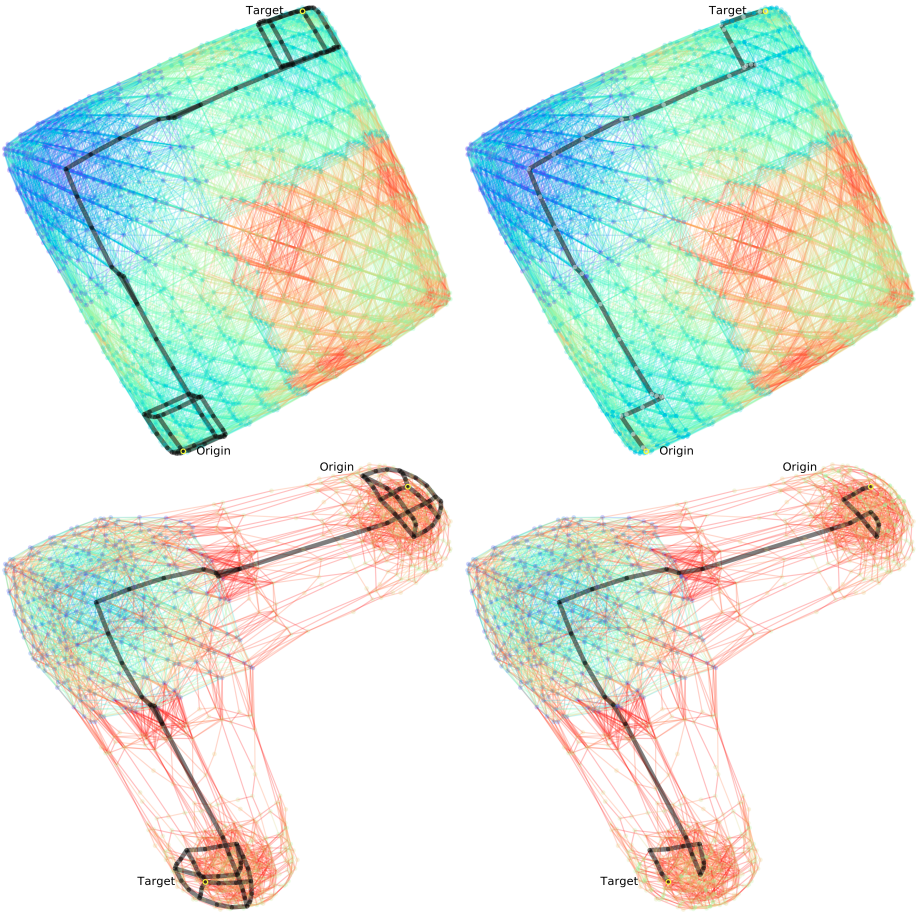

Figure 83: Raw fragmentwise path (left) and the simplified path (right) projection over the full global network (upper panels) and the nework formed by the well sampling points (lower panel). Origin and target points are indicated.

7.9.4 5D sines: more than one million points over 5 dimensions

It is highly recommended to follow the first step-by-step example (7.9.1) before checking this one, as the first example is closer to a basic tutorial, while the aim of the rest of the examples is to show different features of MEPSAnd in a graphic manner to suggest ideas that might be useful for users with a varied range of data sets to analyze.

In this example the surface file analyzed is "sine_5D.dat" from the "examples.zip" file that can be downloaded from http://bioweb.cbm.uam.es/software/MEPSAnd/.

It is a an artificially created surface with 1,022,028 points and 5 dimensions (4 coordinates and energy) to show that handling surfaces with no intuitive representation is still quite straightforward as long as you have some criteria to choose origin and target. As, in this example, there is no particular reason to select any particular point, the first (0) and last (1022027) points of the surface were used as origin and target respectively. If this is to be followed, please note that connectivity definition may take some time. On a laptop with an i7 processor, the whole process of connectivity definition and global network calculation took 102 minutes. Once that step is done, it is recommended to save session before proceeding with any other task.

The Perl script used to create "sine_5D.dat" is:

```
#!/usr/bin/perl
use warnings;
use strict;
use Math::Trig;

my $x;
my $y;
my $z;
my $w;
my $energy;
my $step = 1;

for ($x = -11; $x <= 11; $x+=$step)
{
	for ($y = -11; $y <= 11; $y+=$step)
	{
		for ($z = -11; $z <= 11; $z+=$step)
		{
			for ($w = -10; $w <= 10; $w+=$step)
			{
				$energy = abs($x) * abs(sin($x)) + abs($y) * abs(sin
				    ↪ ($y)) + abs($z) * abs(sin($z)) + abs($w) *
				    ↪ abs(sin($w));
				print ("$x␣$y␣$z␣$w␣$energy\n");
				print ( "$x␣" . ($y+22+$step) . "␣$z␣$w␣" . ($energy
				    ↪ -10) ."\n");
				print ( ($x+22+$step) ."␣$y␣$z␣$w␣" . ($energy+10) .
				    ↪ "\n");
				print ( ($x+22+$step) . "␣" . ($y+22+$step) . "␣$z␣
				    ↪ $w␣" . $energy . "\n");
			}
		}
	}
}
```

The raw fragmentwise path was plotted in several ways: energy profiles (Fig. 84), coordinate projections (Fig. 85) and network projections (Fig. 86).

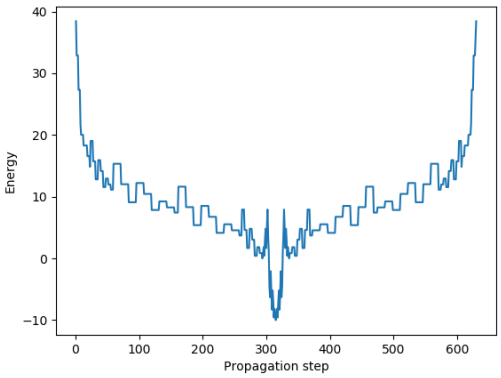

Figure 84: Energy profile along the raw fragmentwise path.

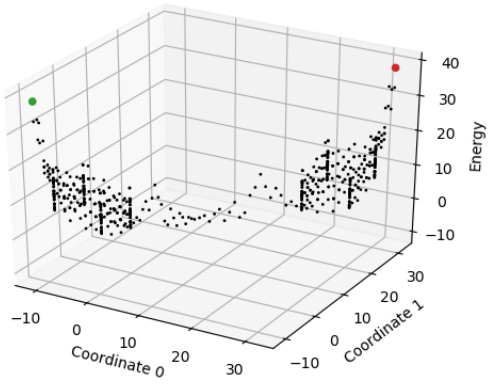

Figure 85: Raw fragmentwise path projection over coordinates 0, 1 and energy. Origin and target are indicated by the green and red points respectively.

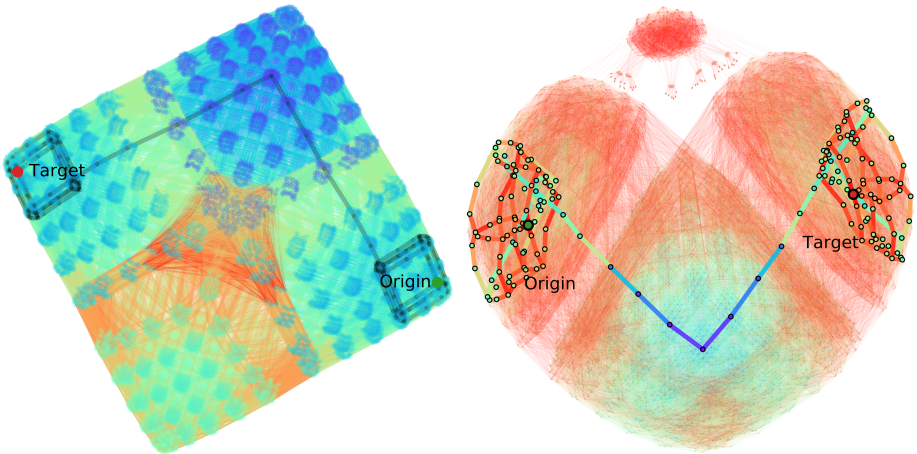

Figure 86: Raw fragmentwise path projection over the full global network (left) and the nework formed by the well sampling points (right).

7.10 Good practices

In this section we will add a series of tips provided by MEPSAnd users to handle specific types of datasets.

7.10.1 Metadynamics and partially sampled datasets

Thanks to the contributions of Dr. Marcelo D. Polêto we can indicate a series of tips to efficiently use MEPSAnd with partially sampled datasets and, in particular, those produced by metadynamics simulations:

- Do not fill unsampled regions with placeholder values (e.g. artificially high energy points). MEPSAnd does not need a complete sampling of all the dimensions to work and adding these artificial points will create a massive barrier cluster connecting a large portion of the minimum clusters to each other, which in turn creates a massive network of unrealistic connections that has a huge impact on both RAM usage and CPU times. In other words, **if a point is not sampled or yields NaN/inf energies, just remove that point from your dataset, do not use a placeholder value**.
- In his line of work, Dr. Polêto has observed that, when obtaining 3D energy surfaces (3 spatial coordinates and 1 energy coordinate) via metadynamics, using very small bin sizes (*e.g.* 10 bins/Ångstrom) produces large and noisy surfaces that are significantly slower and heavier to handle with MEPSAnd. On the other hand, using more realistic bin sizes (*e.g.* 2 bins/Ångstrom) fixes this problem.