LOMAP – some thoughts

Antonia Mey

Abstract

Overview

This short document will discuss the current state of LOMAP and ideas where it can go to be used in automated workflow generation for computing relative free energies of binding using alchemical perturbation methods. LOMAP tries to fulfil the following goals when automatically planning networks for , as laid out in the original paper [2]:

- Goal 1: Compounds should be as similar as possible
- Goal 2: Do not allow ring breaking
- Goal 3: The net charge of ligands should be preserved
- Goal 4: Multi ring system can only partially deleted if they are planar
- Goal 5: Every molecule must be part of a closed thermodynamic cycle
- Goal 6: The set of planned calculations should be spanned by few calculations.

These above goals are very useful and should definitely be fulfilled in a more fine grained version. However, if I think about generating automated workflows I have more coarse grained criteria. To be precise for use of LOMAP in an automated fashion I would really like the following three things fulfilled:

1. Have a reliable measure that can assess whether a proposed perturbation is likely to work.
2. Construct a network that has a high fault tolerance, while minimizing the number of edges (i.e. free energy computations)
3. Identify situations where the dataset cannot fulfil the above two criteria and propose the insertion of intermediates.

A closer look at some of the issues

Let us look at the above points in a bit more detail: Attached to this document is a LOMAP output for the CatS dataset of GC4 (D3R2018.pdf). The green arrows are simulations I have deemed as successful in an FEP protocol and red arrows are simulations that I have deemed most likely to fail. In fact some of the simulations failed none the less. An example for this is the perturbation between 4 and 9 as seen in figure 1 (A). An intermediate had to be proposed to allow for a successful simulation with SOMD, see figure 1(B). This leads me to think that with challenging datasets where the perturbation in congeneric series are much larger than halogen substitutions, the similarity score as it stands is not sufficient to generated perturbations that are reliable, or rather, it seems that the loose condition may be too loose. Since machine learning is so en vogue it would seem that probably a ML algorithm could be use to learn this measure, particularly using the source of successful perturbation networks that are available in the literature.

Regardless of finding an idealised measure of how likely a proposed perturbation will work, the current version of LOMAP using the 2018.01 version of RDKit will sometimes face time outs for the MCS matching which can result in poor prediction of the MCS overlap and in this way a badly proposed mapping. This may be the first thing to look at and build in some more challenging test cases as part of the unit testing suite.

Assuming we now have this measure and can classify proposed perturbations to likely to fail or be successful, I would probably propose to use ideas from percolation theory of how to actually construct the network. Cycle closure is one way of assessing robustness in a network, but the same applies for computing forward and backward perturbations, or even using multiple paths between to compounds.

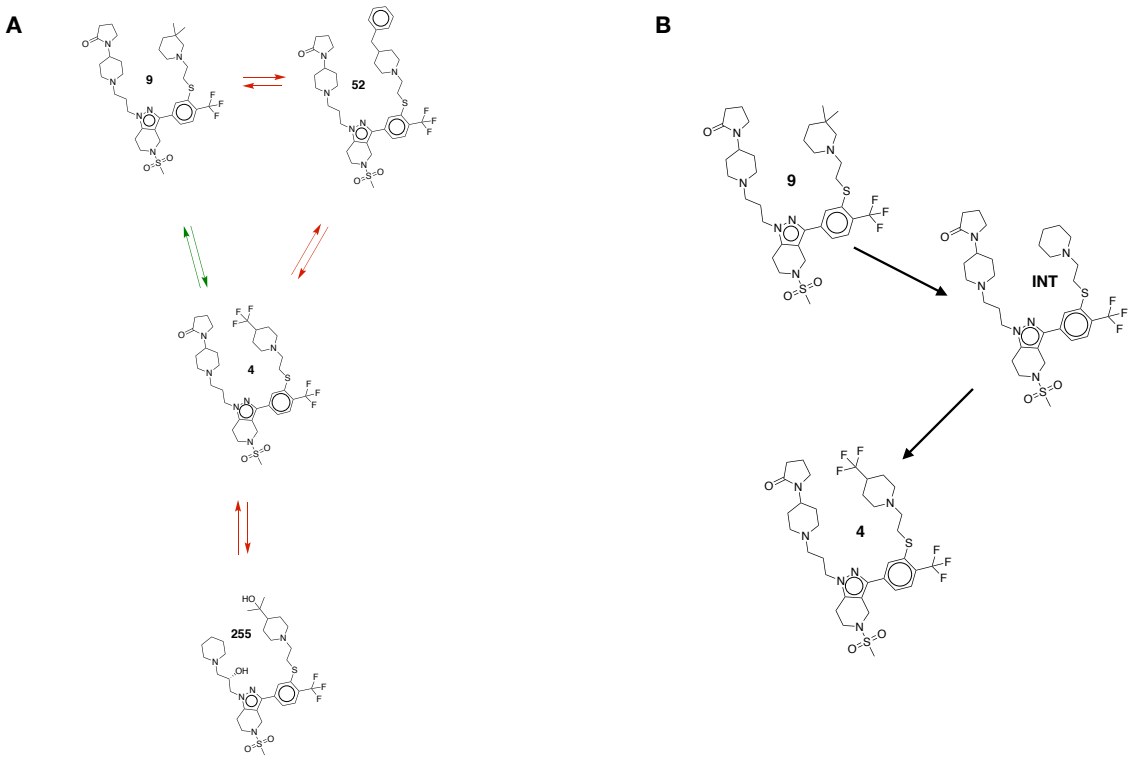

Figure 1. A: partial excerpt of a LOMAP proposed mapping for D3R GC4, with red arrows indicating likely fails and green arrows deemed as ok. Green arrows also failed after an attempted simulation. B: proposed intermediate between compound 9 and 4 that will fail based on attempted simulations.

The relative free energy of binding between compound A and B should be the same no matter which path along the network is taken. Therefore another assessment of how good some of the free energy estimates in a network are, is how consistent is the computed free energy value taken different paths between two nodes, or also known and shortest simple paths, a term from graph theory. So rather than having every compound in a thermodynamic cycle, which may be large would be to have them involved in multiple routes that are as short as possible, while still keeping the number of edges to a minimum. So for example for the GC4 dataset generated with LOMAP, there are 48 unique paths between node 'CatS_178' and 'CatS_4', the reference compound. However, only one route between the reference compound and node 'CatS_255'. In fact adding the cycle condition does add multiple paths and randomly human constructed network for the GC4 perturbations is much worse in terms of nodes having many possible paths between them.

Ideas for an improved algorithm

I would like to see if constructing a network in a slightly different way help improve robustness of free energy calculations and would be happy to receive feedback about the idea and feasibility of the construction of the network. The proposed algorithm should look like this:

1. compute measure that will identify n nearest neighbours that could potentially connect compounds for perturbations
2. Construct a small world network with connecting up to m nearest neighbours.
3. Rewire the x% of the network by randomly picking edges provided the rewiring will still ensure that the perturbation is allowed
4. Remove edges such that edges are only picked from the pool of non-rewired edges.
5. Test the robustness of network (see notes below)

How can we test the robustness [1]? This is still quite hand-wavy and would be matter of actually looking at some network literature (percolation thresholds etc), but for sure one of the criteria should be that the number of simple shortest paths between all nodes should be similar and not result in some nodes having many connections and others only very few. Maybe goals 1-6 stated above already fulfil this. But mostly the idea is born from a poorly generated manual network with a single bottleneck edge, whose large error can introduce a very large error in the overall network, when using error propagation to compute errors over multiple paths needed for the computation of a relative free energy of binding.

Smaller wish list of features

Mostly the ideal wish-list consists of features that will improve the graphical display and programming side.

- Have an interactive network that let's you manually adjust connection via drag and drop.
- Have nodes displayed as balls that can be expanded in order to reveal the underlying molecular structure.
- Automatic suggestion of adding intermediates.
- Consistent naming of variables classes and function (e.g. PEP 8, or model RDKit convention)
- Adhere to minimum standards of a well maintained repository

Comments on current status of the repository

The feature branch incorporating pytest can be merged into devel and a release from it can be generated. This stable version supports python 3.+ using networkx2, a relatively recent version of RDkit and no longer a dependency on pyQt. It has been tested on OSX and Linux. I can be in charge of this and then happily pass on to a different maintainer.

References

[1] Li, D., Zhang, Q., Zio, E., Havlin, S., and Kang, R. (2015). Network reliability analysis based on percolation theory. *Reliability Engineering & System Safety*, 142:556 – 562.

[2] Liu, S., Wu, Y., Lin, T., Abel, R., Redmann, J. P., Summa, C. M., Jaber, V. R., Lim, N. M., and Mobley, D. L. (2013). Lead optimization mapper: automating free energy calculations for lead optimization. *Journal of Computer-Aided Molecular Design*, 27(9):755–770.