molSimplifyAutomaticDesign (mAD) is a collection of open-source routines that make use of the inorganic molecular design toolbox molSimplify to conduct targeted molecular inorganic design. This is accomplished through either surrogate models (i.e. our neural network) or using DFT calculations. mAD includes functions for defining design spaces, generating required simulation files, submitting them to the automated queuing systems, post-processing the results and then selecting which complexes to test next!

Getting mAD:

Before getting mAD, you will need an installation of the latest build of molSimplify. This is easily available from Conda or Git. Installation instructions can be found at mAD has no additional dependencies on top of molSimplify.

mAD is available from our github repo () and you can clone it from there. From a unix/MAC terminal, run:

git clone

Make sure you are in the Conda environment with molSimplify installed. This is development software and use of a contained Conda environment is STRONGLY recommended. Then navigate into the molSimplifyAutomaticDesign folder and run

python setup.py install

alternatively, you can run python setup.py develop if you want to edit the code with reinstalling it. That’s all there is to it!

Quick start

In order to run a simple neural network guided design run to search for spin crossover complexes with the default options (see below), simply enter:

mad –new

this will create a default run named “GA\_run”. This can be launched with

mad –resume GA\_run –reps 21

this will perform 21 generations of evolution using the neural network to evaluate fitness, while control for distance from training data and pool diversity. This process should take 5-7 minutes on a standard workstation.

Updates will be logged to GA\_run/statespace/log.txt. See the section of interpreting results below for more information.

How to use mAD:

mAD is intended to be autonomous and easy to use and has only two commands: **new**, and **resume**. The steps to use mAD are:

* Create an optimization run using mAD using **new**
* Check compatibility of the queue submission script (if using DFT)
* run with **resume**.

Defining a run:

A new mAD optimization can be created using the following syntax:

mad –new input.txt

All information that mAD needs to work is provided in the form of a plain text input file. This file completely defines the behavior of the run. If you don’t provide an input file or you don’t specify and option, the **default option**s, indicated in bold, will be assumed. It is possible to omit the input file direct, which will result in a neural network based run with default ligands

|  |  |  |
| --- | --- | --- |
| Keyword | Options (**default**) |  |
| General setup | | |
| DFT | True | **False** | Toggle if DFT single point calculations (using TeraChem) or the neural network model will be used to assess the properties of new complexes |
| optimize | True | **False** | Using geometry optimization instead of single points |
| runtype | redox | **split** | Run in spin splitting or redox modes |
| symclass | **strong** | weak | Determine if axial ligands should match (strong) or not (weak) |
| use\_singlets | True | **False** | Force low spin singlet spins (DFT only) |
| solvent/thermo | True | **False** | Run solvent and thermos corrections after DFT calculation completes (for reference, ignored by GA) |
| rundir | Name of run directory | **Current dir + “/GA\_run/”** | Set the run directory for the mAD run. If the directory already exists, the path will be extended by adding “\_X” where X is the smallest integer that creates a new directory. |
| liglist | **(default ligands, see next section)** | path to a file containing a list of ligands as per the next section | The list of ligands define the design space for the run |
| queue (dft only) | **SGE |** slurm | Which queue manager to use |
| GA configuration | | |
| npool | **20** |  |
| ncross | **5** |  |
| pmut | **0.15** |  |
| maxgen | **20** |  |
| split\_parameter | **15** |  |
| monitor\_diversity | **True** | False |  |
| monitor\_distance | **True |** False |  |
| distance\_paramter | **1.0** |  |
| octahedral | **True** | False | Whether structures are supposed to be octahedral. |
| HFXsample | **True** | False | Whether to expand the HFX region |
| maxjobs | **20** | Maximum number of simultaneous DFT jobs to allowed in the queue |

Running with DFT:

In order to use DFT single point energy evaluation, mAD needs to be able to submit jobs to the queue of your compute environment. Because environments differ, you will need to check the generated jobscript, called launch\_script.sh, is compatible with your configuration (i.e. account numbers and allocations, as well as any system-specific info your environment requires).

Resuming a run:

Once you have created a run, mAD can be launched from the command line using:

mad –resume PATH\_TO\_RUN –reps X –wait Y

where PATH\_TO\_RUN is the folder containing the run created before. Each call to resume will activate the code once, meaning that any outstanding calculations will be submitted and any finished calculations will be processed. Using the neural network, the calculations will be finished instantly, so each repetition will advance the generation by 1 and there is no need to use the wait flag.

In the DFT case, the single point calculations take a variable amount of time and the generation cannot be advanced until they are complete, so each call to resume will not necessarily advance the generation. The –wait flag is used to delay repetitions to allow for DFT jobs to complete, and takes a number in seconds. For example, -wait 1800 would cause mAD to check on the job every 30 minutes. mAD is interruption safe so if the process is killed or runs out of repetitions before the requisite number of generations, simply call –resume again and pick up where the run left off.