

JKCS2.1 - MANUAL

Jammy Key
for
Configurational Sampling

Jakub Kubečka

February 21, 2020

Contents

Introduction	3
Installation	4
First test	5
0 - JKCS0_copy	6
The file input.txt	7
1 - JKCS1_prepare	8
Supercomputer cluster	9
2 - JKCS2_explore	10
3 - JKCS3_run	12
4 - JKCS4_collect	14
5 - JKCS5_filter	16
Questions and Tips	19

Introduction

JKCS is a systematic approach for configurational sampling of molecular clusters.

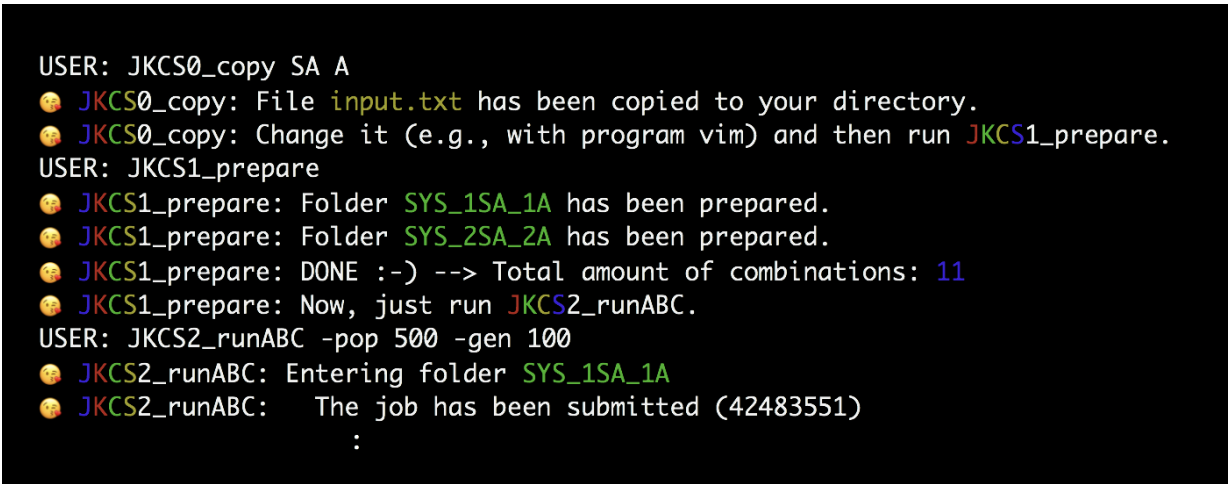
- collection of scripts which operates with large set of structures and files
- communication with 3rd-party computational programs such as Gaussian, Orca, ABCluster, XTB.
- automatized operating of jobs and submitting to supercomputer cluster

HOW TO CITE?

Upto now, there is just article about Configurational Sampling written by Kubečka *et al.* [1]. The article about the program itself is on the way.

JKCS2.1:

- most bugs are removed
- many parts are accelerated
- GitHub version (<https://github.com/kubeckaj/JKCS2.1.git>)
- almost proper manual :-D
- user-friendly coding (see example below)



```
USER: JKCS0_copy SA A
🤖 JKCS0_copy: File input.txt has been copied to your directory.
🤖 JKCS0_copy: Change it (e.g., with program vim) and then run JKCS1_prepare.
USER: JKCS1_prepare
🤖 JKCS1_prepare: Folder SYS_1SA_1A has been prepared.
🤖 JKCS1_prepare: Folder SYS_2SA_2A has been prepared.
🤖 JKCS1_prepare: DONE :-D --> Total amount of combinations: 11
🤖 JKCS1_prepare: Now, just run JKCS2_runABC.
USER: JKCS2_runABC -pop 500 -gen 100
🤖 JKCS2_runABC: Entering folder SYS_1SA_1A
🤖 JKCS2_runABC: The job has been submitted (42483551)
:
```

Figure 1: Example of the user-friendly coding.

CITATION:

- [1] KUBEČKA, J., BESEL, V., KURTÉN, T., MYLLYS, N., AND VEHKAMÄKI, H. Configurational sampling of noncovalent (atmospheric) molecular clusters: sulfuric acid and guanidine. *J. Phys. Chem. A* 123 (2019), 6022–6023.

Installation

HOW TO SETUP:

```
git clone https://github.com/kubeckaj/JKCS2.1.git #clone JKCS2.1
cd JKCS2.1
sh setup.sh                                     #runs setup (installation)
source ~/.bashrc                               #required just once
```

(sh setup.sh -r) rewrites old user setup (~/.JKCSusersetup.txt)

ADJUST USER SETUP:

(H.Vehkamäki group: Puhti users do not have to adjust it)

```
vim/nano/emacs ~/.JKCSusersetup.txt           #adjust user setup

#      CHANGE:
# ABCcluster: rigidmoloptimizer path
# XTB: xtb path
# PYTHON: please setup how to use python2.0 (python 3.0 is just for GoodVibes in JKCS4)
# Gaussian: G16 path and module
# Orca: Orca path and modules

sh test.sh                                     #test that all your paths
#                                               are set correctly
```

Each further sections shows in the end all required programs (*e.g.*, Python3.x is required just in JKCS4_collect when applying GoodVibes.py calculations.)

After writing "JK" and pressing tabulator [TAB] in command line, you should be able to see various JK & JKCS commands (see Figure 2).



JKattachstr.py	JKCS1_prepare	JKCS5_filter	JKg16.sh	JKname	JKremovefiles	JKsubg16
JKavg	JKCS2_explore	JKCS6_monster	JKgaussstat	JKorca	JKrevibPM7	JKxyz2com
JKcheck	JKCS2_runABC	JKfor	JKlog2com	JKorcaDLPNO	JKsacct	JKxyz2inp
JKchmod	JKCS3_run	JKformation	JKlog2xyz	JKqdelall	JKsend	
JKCS0_copy	JKCS4_collect	JKfreeslots.py	JKmovetoorigin.py	JKqmuch	JKsend8	

Figure 2: Auto-filling options for JK & JKCS commands.

First test

YOUR FIRST TEST:

The following shows how to use JKCS. Since the calculations are very fast, the command argument (-loc) is used in several cases to run jobs just on local computer.

```
cd $WRKDIR                #go to your working directory
mkdir TEST_JKCS           #create a new folder
cd TEST_JKCS              #and enter it
JKCS0_copy --help          #This fails if installation wasn't successful or
                           #you didn't source ~/.bashrc. Shows help
JKCS0_copy SA A            #creates input.txt for sulf. acid and ammonia
ls                         #list copied files
vim input.txt              #make changes to the input file //or nano,emacs
JKCS1_prepare --help       #shows help
JKCS1_prepare              #prepare folders and some files based on input.txt
ls                         #list files and folders
JKCS2_explore --help       #shows help
JKCS2_explore -gen 10 -lm 5 -pop 10 -loc
                           #This fails if you didn't setup ABCluster
                           #runs 10-bee colony for 10 generations and save 5
                           #best structures. (each subfolder is entered)
JKcheck                    #check that all calculations were finished
#optional: you can perform [JKCS4_collect ABC -loc] to collect ABC results
JKCS3_run --help           #shows help
JKCS3_run -loc             #This fails if you didn't setup xtb path
JKCS4_collect XTB -loc     #This fails if you didn't setup python2.x properly.
                           #Collects semi-empirically(xtb) optimized xyz
cd SYS_1SA_1A              #Enter folder with cluster 1sa1a
cat resultsXTB.dat         #see results [structure | gyration rad. | energy]
molden movieXTB.xyz        #visualize molecules
```

If everything worked, continue. Otherwise, contact Jakub Kubečka (jakub.kubecka@helsinki.fi). Add just the rest of file ~/.JKCSusersetup.txt: QUEING, QUANTUM CHEMISTRY PROGRAMS *etc.*

0 - JKCS0_copy

Configurational sampling starts everytime from here. Create a new directory (**WORKDIR**) where you plan to perform configurational sampling of desired system(s). Create the input file *input.txt*: either use JKCS0_copy command or copy your input file from previous projects. See below, how to use JKCS0_copy command. Moreover, also file *commands.txt* is copied to show typical sampling process.

COMMAND:

```
JKCS0_copy [OPTION(s)] [STRUCTURE(s)]
```

OPTIONS:

```
-help ..... print this help
-all ..... link all available structures
```

STRUCTURES:

W	water (H2O,OH-,H+)	CO2	carbon dioxide
AQ	water (H2O)	CH4	methane
		Ar	argone
HNO3	nitric acid	Ne	neone
SA	sulphuric acid	He	helium
MSA	methanesulfonic acid		
		H	proton (+)
A,AM	ammonia	Na	sodium (+)
GD	guanidine	Cl	chloride (-)
DMA	dimethyl ammine		
TMA	trimethyl ammine	HI03	iodic acid
urea	urea	HI02	iodous acid
		I205	iodine pentoxide

EXAMPLES:

```
JKCS0_copy SA AM #creates input file containing link to SA and AM structures
```

REQUIRED:

```
bash/sh
```

The file *input.txt*

The file gives the most import definition for all clusters in a current directory. If something is supposed to be different, thus, create just somewhere else a new folder for new configurational sampling. For each program (ABC,XTB,G16 ...), you should setup supercomputer parameters such as: how many jobs can run at maximum parallelly, how many cpu, nodes and memory can be used, what is the name of partition and how long is the requested walltime. Almost for all of them, you can use variables NoC and M to define the variable as a function of 'Number Of Combinations' or 'number of Molecules'. -loc program stays for computing on local computer. All these parameters could be rewritten by supercomputer cluster commands (see chapter Supercomputer cluster). Hopefully total charge a multiplicity are clear. The composition defines the amount of each monomer in the desired cluster. Based on the picture, the 1_1 composition stays for 1SA and 1A, because in structures it is defined first SA monomers and then A monomers. Program (JKCS1_prepare) will find all possible combinations of monomers with charge 'q' to form a desired cluster and still fulfil its total charge number. If more clusters is sampled, we can use some symbols to define the clusters: *e.g.*, 1-3_(4,5) stays for clusters = 1_4, 2_4, 3_4, 1_5, 2_5 and 3_5.

```
#####
## SUPERCOMPUTER PARAMETERS ##
## Number of Combinations - NoC ##
#####
## MAXTASKS CPU NODES REQ.TIME PARTITION MEMPERCPU ##
=====
ABC NoC 1 1 72:00:00 serial 4000
XTB NoC 1 1 72:00:00 serial 4000
G16 100 8 1 72:00:00 serial 4000
ORCA 100 8 1 330:00:00 longrun 4000
-loc 1 1 1 - - -
=====

#####
## SYSTEM CHARGE AND MULTIPLICITY ##
#####
TotalCharge 0
TotalMultiplicity 1

#####
## COMPOSITION: ##
## e.g.: 1_1_2 1_2_1 1_3 ##
## e.g.: 1_3-6 = 1_3 1_4 1_5 1_6 ##
## e.g.: (1,3,5)_1 = 1_1 3_5 5_1 ##
## e.g.: (2,4)_1-3_1 = 2_1_1 4_1_1 2_2_1 4_2_1 2_3_1 4_3_1 ##
## e.g.: 1_1_F2 = 1_1_0-2 #protons# to fulfill charge ##
#####
Composition 1_1 2_2

#####
## STRUCTURES OF BUILDING MONOMERS: ##
#####
# name | q | path
SA 0 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/ABC/h2so4.xyz
SA 0 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/JACOB/h2so4_cis.xyz
SA -1 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/ABC/hso4.xyz
SA -2 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/ABC/so4.xyz
A 0 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/ABC/nh3.xyz
A 1 /wrk/kubeckaj/DONOTREMOVE/Apps/JKCS2.0/JKCSx/./TOOLS/STRUCTURES/ABC/nh4.xyz
```

Figure 1: The file *input.txt* in its full beauty.

1 - JKCS1_prepare

Script JKCS1_prepare creates subfolders (SYS_{system}) in your "mother"-working directory based on the input file *input.txt* (see folder structures in Figure 1.1). In each directory, all possible combinations of molecular monomers are searched and written down. These combinations fulfil number of each monomer and total charge. JKCS1_prepare does not require any arguments.

COMMAND:

```
JKCS1_prepare [OPTION(s)]
```

OPTIONS:

```
-help ..... print this help
-s,-sample "X" ..... in the case of many combinations, picks just X
                      random combinations (useful when molecules with
                      a lot of isomers are used)
-o,-overwrite ..... overwrite parameters.txt also in existing folders
```

The variable X could be number (*e.g.*, 100) or function of number of Molecules (M) or number of all combinations (NoC) (*e.g.*, "20*M", "NoC/2", "2*NoC/M")

REQUIRED:

```
bash/sh
python = python2.x
```

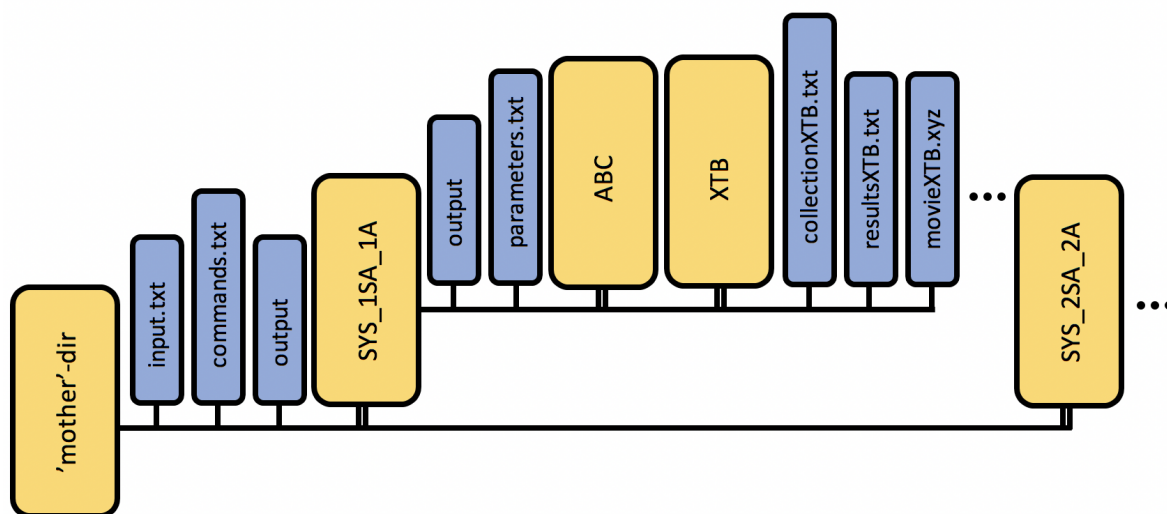


Figure 1.1: Scheme of folders and files formed during the configurational sampling.

Supercomputer cluster

When you are working on supercomputer cluster, all parameters for submitting jobs are loaded from file *parameters.txt*. This file is created based on the user input file *input.txt* (JKCS1_prepare creates the *parameters.txt*). (See JKCS1_prepare -overwrite if you need to rewrite already existing *parameters.txt* files.)

However, you can whenever overwrite those parameter by using following arguments in the specific command (see bellow). The order of arguments is not important. Basically, it make sense to use these arguments just for JKCS2_explore, JKCS3_run and JKCS4_collect.

ARGUMENTS:

```
-loc ..... run the calculation on local computer
-tasks,-maxtasks NUMBER .. max. number of jobs to be submitted (per subfolder)
-cpu NUMBER ..... amount of CPU(s) used per one job
-N,-nodes NUMBER ..... amount of nodes to be utilized [NOT TESTED]
-time TIME_FORMAT_NUMBER . requested walltime [e.g., 72:00:00]
-par,-partition NAME ..... partition name [e.g., short, hugemem, longrun ...]
-mem,-memory MEMORY ..... size of memory allocated per CPU [e.g., 4000mb]
```

EXAMPLES:

```
JKCS4_collect DFT -loc
JKCS3_run -program XTB -of ABC -mem 1GB -cpu 2 -par test -time 00:20:00
JKCS3_run -maxtasks 1
```

REQUIRED:

```
bash/sh
sbatch
```

2 - JKCS2_explore

You can run this script in each created sub-folder or directly from the "mother"-directory, and thus, the script will be applied immediately to all sub-folders. The exploration of potential energy surface is performed with the Artificial Bee Colony (ABC) algorithm, which was first time proposed by Karaboga [1] in 2008. The algorithm is implemented in the ABCcluster program [2, 3], which is also used by this script.

COMMAND:

```
JKCS2_explore [OPTION(s)] [STRUCTURE(s)]
```

DEFAULT COMMAND:

```
JKCS2_explore =  
JKCS2_explore -program ABC -lm 300*M/NoC -gen 100 -sc 4 -pop 300*M
```

OPTIONS:

```
-help ..... print this help and exit  
-l,-lm "X" ..... X local minima will saved [def=300*M/NoC]  
-g,-gen "X" ..... amount of ABC generations [def=100]  
-s,-sc "X" ..... amount of scout bees [def=4]  
-pop,-i,-init "X" . amount of initial guesses [def=300*M]  
  
OTHERS: -box [def=7*M]  
OTHERS: -wtb,-walltimebased {1000s,3h,1d ...} [TESTING]  
OTHERS: -wtc,-walltimecontrolled {1000s,3h,1d ...} [TESTING]
```

As you can see, some of the variables are by default defines as functions of some another variables. It helps if you are working with several clusters and you want do define that for smaller cluster does not have to be done so huge exploration like in the case of large clusters. You can use these symbols as well to define arguments.

SYMBOLS:

```
M .... number of molecules  
NoC .. number of combinations to form cluster  
use * or / or M or NoC and script will evaluate it
```

Some command arguments are now well tested yet but you can already try to use them. These are -wtb and -wtc:

WALL-TIME-BASED: -wtb XYh set number of generation to be equal 100 and bee population to be such that overall calculation take XYh to be finished by 1 CPU.

WALL-TIME-CONTROLLED: -wtc XYh works like -wtb XYh except that it overwrite population and generation if they are higher, i.e. the calculation would take more than XYh.

EXAMPLES:

WRONG exploration (BAD = DO NOT USE):

```
JKCS2_explore -pop 100 -lm 100 -gen 1      #too short (gen>=100)
JKCS2_explore -pop 1000 -gen 100 -lm 1      #few saved structures
JKCS2_explore -pop 100 -gen 100 -lm 300     #too small population
```

Correct exploration:

```
JKCS2_explore -pop 1000 -gen 100 -lm 3000
JKCS2_explore -pop 1000*M -gen 100 -lm 4000/NoC
JKCS2_explore -wtc 3h -pop 2000*M/NoC
```

REQUIRED: [2, 3]

```
bash/sh
ABCluster: rigidoptimizer
```

CITATION:

- [1] KARABOGA, D., AND BASTURK, B. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* 8 (2008), 687–697.
- [2] ZHANG, J., AND DOLG, M. ABCluster: the artificial bee colony algorithm for cluster global optimization. *Phys. Chem. Chem. Phys.* 17 (2015), 24173–24181.
- [3] ZHANG, J., AND DOLG, M. Global optimization of rigid molecular clusters by the artificial bee colony algorithm. *Phys. Chem. Chem. Phys.* 18 (2016), 3003–3010.

3 - JKCS3_run

Utilizes 3rd party programs to reoptimize or quantify clusters. In our research group, we use at the moment following steps[3]:

- semi-empirical re-optimization -> GFN2-*x*TB [2]
- optimization and vibration analysis -> Gaussian 16 [1]

The final DFT level is: ω B97X-D/6-31++g**

- electronic energy correction -> Orca [4]

See below how to use the script.

COMMAND:

```
JKCS3_run [OPTION(s)]
```

DEFAULT COMMAND:

```
JKCS3_run =  
JKCS3_run -program XTB -oldfolder ABC -newfolder XTB -method "-opt vtight"
```

OPTIONS:

```
-help ..... print this help  
-nf,-newfolder NAME ... name of the new (calc.) folder  
-of,-oldfolder NAME ... name of the old folder. On top of these structures,  
                        new calculation methods are applied  
                        (not compatible with -rf)  
-rf,-resultsfile NAME . name of the resultsNAME.dat file containing  
                        structures to be used for applying the comp. method  
                        (File resultsNAME_FILTERED.dat is used, if it exists!)  
                        (not compatible with -of)  
-m,-method "X" ..... method used by 3rd party quantum program  
-p,-program NAME ..... computational program to be used (XTB, G16, Orca ...)  
                        program_$program has to be in ~/.JKCSusersetup.txt
```

ADVANCED OPTIONS:

```
-bs,-add,-addbase X .. insert basis set for heavy atoms to the end of file  
                      (Br,I (BASIS SET ??) ... just for Gaussian)
```

EXAMPLES:

```
XTB:
  JKCS3_run
  JKCS3_run -p XTB -nf XTB_freq -rf XTB -m "-ohess"
G16:
  JKCS3_run -p G16 -rf XTB -nf DFT_sp -m "# HF 6-31+g*"
  JKCS3_run -p G16 -rf XTB -nf DFT_opt -m "# wb97xd 6-31++g** opt=verytight"
  JKCS3_run -p G16 -rf DFT_opt -nf DFT_freq -m "# wb97xd 6-31++g** freq"
ORCA:
  JKCS3_run -p ORCA -rf XTB -nf OPT -m "! PBE0 def2-TZVP TIGHTSCF Opt D3BJ"
ADVANCED:
  JKCS3_run -p G16 -rf XTB -nf DFT -bc I -m "#wb97xd GEN Pseudo=Read Opt
    Int=UltraFine Freq MaxDisk=32GB" -mem 12GB -cpu 16
  JKCS3_run -p ORCA -rf DFT_freq -nf DLPNO -m "! DLPNO-CCSD(T) aug-cc-pvtz
    aug-cc-pvtz/C GRID4 nofinalgrid TightPNO TightSCF NOPOP NOPRINTMOS"
```

REQUIRED:

```
bash/sh
3rd party program (XTB,Orca,Gaussian etc.)
```

CITATION:

- [1] FRISCH, M. J., TRUCKS, G. W., SCHLEGEL, H. B., SCUSERIA, G. E., ROBB, M. A., CHEESEMAN, J. R., SCALMANI, G., BARONE, V., PETERSSON, G. A., NAKATSUJI, H., LI, X., CARICATO, M., MARENICH, A. V., BLOINO, J., JANESKO, B. G., GOMPERTS, R., MENNUCCI, B., HRATCHIAN, H. P., ORTIZ, J. V., IZMAYLOV, A. F., SONNENBERG, J. L., WILLIAMS-YOUNG, D., DING, F., LIPPARINI, F., EGIDI, F., GOINGS, J., PENG, B., PETRONE, A., HENDERSON, T., RANASINGHE, D., ZAKRZEWSKI, V. G., GAO, J., REGA, N., ZHENG, G., LIANG, W., HADA, M., EHARA, M., TOYOTA, K., FUKUDA, R., HASEGAWA, J., ISHIDA, M., NAKAJIMA, T., HONDA, Y., KITAO, O., NAKAI, H., VREVEN, T., THROSSELL, K., MONTGOMERY, JR., J. A., PERALTA, J. E., OGLIARO, F., BEARPARK, M. J., HEYD, J. J., BROTHERS, E. N., KUDIN, K. N., STAROVEROV, V. N., KEITH, T. A., KOBAYASHI, R., NORMAND, J., RAGHAVACHARI, K., RENDELL, A. P., BURANT, J. C., IYENGAR, S. S., TOMASI, J., COSSI, M., MILLAM, J. M., KLENE, M., ADAMO, C., CAMMI, R., OCHTERSKI, J. W., MARTIN, R. L., MOROKUMA, K., FARKAS, O., FORESMAN, J. B., AND FOX, D. J. Gaussian 16 Revision A.03, 2016. Gaussian Inc. Wallingford CT.
- [2] GRIMME, S., BANNWARTH, C., AND SHUSKOV, P. A robust and accurate tight-binding quantum chemical method for structures, vibrational frequencies, and noncovalent interactions of large molecular systems parametrized for all spd-block elements ($z = 1-86$). *J. Chem. Theory Comput.* **13** (2017), 1989–2009.
- [3] MYLLYS, N., ELM, J., HALONEN, R., KURTÉN, T., AND VEHKAMÄKI, H. Coupled cluster evaluation of the stability of atmospheric acid–base clusters with up to 10 molecules. *J. Phys. Chem. A* **120** (2016), 621–630.
- [4] NEESE, F. The orca program system. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2** (2012), 73–78.

4 - JKCS4_collect

This is a script which has to be changed in future. Now it was basically copied from JKCS1.0, but, it seems to work. The program collect selected collective coordinates from specific folder. All the variables (such as file name, gyration radius, energy, dipole *etc.*) are collected to the file *collectionXXX.txt*, where *XXX* is the specific folder from which you want to collect data. Immediately, some smart uniqueness filtering is performed and redundant structures are removed. The remaining structures are printed to file *resultsXXX.txt*. The structures there are sorted with respect to energy (either electronic or Gibbs free energy). Moreover, all xyz structures are also printed to *movieXXX.xyz*, thus, user can visualize them all by using: molder movieXXX.xyz. Additionally, after using JKCS5_filter, new file *resultsXXX_FILTERED.txt* is formed. JKCS3_run prefer to select *resultsXXX_FILTERED.txt* (if it exists) than *resultsXXX.txt*. File FILTER.txt is not important for beginners (just shows history of all uniqueness/filter/selection). See description of command JKCS4_collect below. Some commands uses the GoodVibes program, [1] which has to again setup properly in /.JKCSusersetup.txt.

COMMAND:

```
JKCS4_collect [OPTION(s)] [FOLDER]
```

DEFAULT COMMAND:

```
JKCS4_collect =  
JKCS4_collect XTB
```

OPTIONS:

```
-help ..... print this help and exit  
  
New Collective Coordinates: (3rd column)  
-ncc ..... + something assuming hydrogen bond lengths  
-dip ..... + dipoles  
-g,-gibbs ..... + Gibbs free energy  
-gh,-gibbsh ..... + Gibbs free energy (GoodVibes)  
-b,-bonds [ATOM] [thresh] ..... ++ bonds with spec. atom  
-b2,-bonds2 [ATOM1] [ATOM2] [thresh] ..... ++ bonds with spec. atoms  
-b3,-bonds3 [ATOM1] [ATOM2] [ATOM3] [thresh] . ++ bonds with spec. atoms  
  
Some DLPNO corrected results [NOT TESTED]  
-dlpno ..... needs files from -dlpno1 & -dlpno2  
-dlpno1 "X" ..... set dlpno file [default: resultsDLPNO.dat]  
-dlpno2 "X" ..... set free energy file [default: resultsDFT_HIGH_freq.dat]
```

EXAMPLES:

```
JKCS4_collect DFT_HIGH_freq -gibbs -loc  
JKCS4_collect XTB -time 1:00:00  
JKCS4_collect DFT_HIGH -loc
```

REQUIRED:

```
bash/sh  
python2.x (for Rg, bonds or ncc calculation)  
python3.x (just for GoodVibes)  
GoodVibes (for some specific variables: -gh ...)
```

CITATION:

- [1] FUNES-ARDOIS, I., AND PATON, R. Goodvibes: Goodvibes v1.0.1. *DOI:*
<http://dx.doi.org/10.5281/zenodo.60811> (2016).

5 - JKCS5_filter

IN 2-DIMENSIONS:

1) You have to collect data:

```
JKCS4_collect XTB
```

File collectionXTB.txt is formed with 3 columns: XYZ-file-name | Rg | E

Rg = Radius of gyration, E = (electronic) energy

2) You can visualize the collected data:

```
gnuplot
```

```
plot "collectionXTB.txt" u 2:3
```

(zoom in by mouse might be required)

3) UNIQUENESS: (YOU CAN SKIP)

Redundant files are automatically removed (uniqueness filter), see file resultsXTB.dat. Nevertheless, what does the 'uniqueness filter' term means?

Two files with energy difference less than 0.001 hartree and Rg difference 0.01 are assumed to be the same -> one of the files is removed. If you are not satisfied with filtering thresholds, you can remake resultsXTB.dat by:

```
JKCS5_filter resultsXTB.dat -u -u1 2 -u2 4
```

//this mean thresholds for Rg - 0.01 and for El.E.- 0.0001

4) FILTERING:

If you need to remove out-lying structures use

```
JKCS5_filter resultsXTB.dat -dm 2.5
```

//this filter out structure with relative energy higher than 2.5*M kcal/mol

5) SAMPLING/SELECTION:

Still, a lot of structures will probably remain in resultsXTB_FILTERED.dat.

Thus, you can use uniform sampling from them to cover PES as best as possible with lower computational cost. (Oh, yes! You might loose global minimum structure, but you should not be so far from it). Join filtering and sampling:

```
JKCS5_filter resultsXTB.dat -dm 2.5 -s 100
```

6) Visualize the result:

```
gnuplot
```

```
p "resultsXTB.dat" u 2:3, "resultsXTB_FILTERED.dat" u 2:3 pt 5 ps 2
```

(zoom in by mouse might be required)

// green points indicate selected points

7) JKCS3_run will now take structures saved in resultsXTB_FILTERED.dat

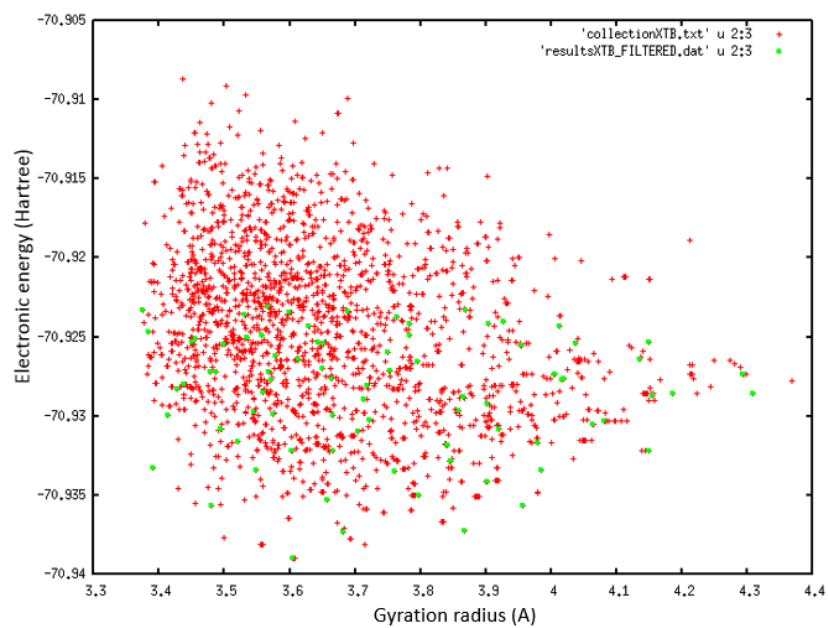


Figure 5.1: Example of 2D rough filtering and sampling.

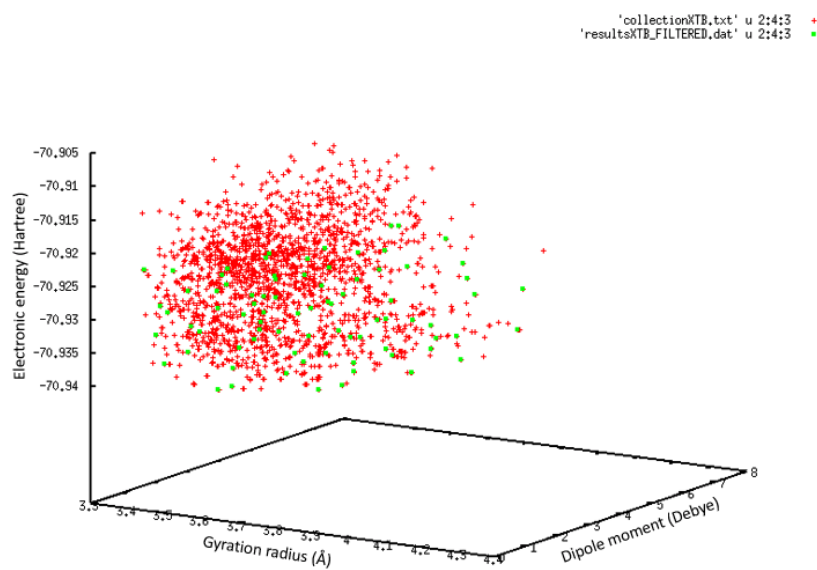


Figure 5.2: Example of 3D rough filtering and sampling.

IN 3-DIMENSIONS:

1) You have to collect data also with dipoles (or another collective coord.):

```
JKCS4_collect XTB -dip
```

File collectionXTB.txt is formed with 4 columns: XYZ-file-name | Rg | E | D

Rg = Radius of gyration, E = (electronic) energy, D = dipole

2) You can visualize the collected data:

```
gnuplot
```

```
splot "collectionXTB.txt" u 2:3:4
```

(zoom in might be required:set yrange [0,10].Green points indicate selection)

3) UNIQUENESS: (YOU CAN SKIP)

Redundant files are automatically removed (uniqueness filter), see file resultsXTB.dat. Two files with energy difference less than 0.001 hartree, Rg difference 0.01 and dipole difference 0.001 Debye are assumed to be the same -> one of the files is removed. We're not satisfied with filtering thresholds, so let us recreate resultsXTB.dat by:

```
JKCS5_filter resultsXTB.dat -u -u1 2 -u2 3 -u3 1
```

//this mean thresholds for Rg - 0.01 and for E- 0.001 and dip - 0.1

4) FILTERING:

If you need to remove out-lying structures use

```
JKCS5_filter resultsXTB.dat -dm 2.5
```

//this filter out structure with relative energy higher than 8 kcal/mol

5) SAMPLING/SELECTION:

Still, a lot of structures will probably remain in resultsXTB_FILTERED.dat. Thus, you can use uniform sampling from them to cover PES as best as possible with lower computational cost. (Oh, yes! You might loose global minimum structure, but you should not be so far from it). Join filtering and sampling:

```
JKCS5_filter resultsXTB.dat -dm 2.5 -c3 4 -s 100
```

//approx. 100 structures were selected

6) You can visualize the collected data:

```
gnuplot
```

```
splot "resultsXTB.dat" u 2:3:4, "resultsXTB_FILTERED.dat" u 2:3:4 pt 5 ps 2
```

(zoom in might be required:set yrange [0,10].Green points indicate selection)

7) JKCS3_run will now take structures saved in resultsXTB_FILTERED.dat

ALL TOGETHER IS ALSO POSSIBLE:

```
JKCS5_filter resultsXTB.dat -u -u3 1 -dm 2.5 -c3 4 -s 100
```

REQUIRED:

```
bash/sh
```

```
python2.x (for Rg, bonds or ncc calculation)
```

Questions and Tips

TIPS:

COLORS & SYMBOLS: in the printed output of JKCS commands can be turned of in 2 ways. Either change Qsymbol or Qcolours in ~/.JKCSusersetup.txt to "no" or:

COLORING TEXT: use -nocolors argument to have text without colors

KISSING SYMBOL: use -nosymbol to remove the symbol in the beggin. of output

PRINT: you can adjust amount of printed output by using command -print NUM:

-print 0 basically just error messages

-print 1 [DEFAULT] traditional output

-print 2 enlarged output, all algorithm steps are commented

-print 3 very very detailed output

PERFORMING JKCS COMMANDS: you can perform each command in specific subfolder ("SYS_"), but you can also stay in your 'mother directory' and perform the command from there [The algorithm enters to each directory and perform the command there].

If you wish to perform command from your 'mother directory' but just for some specific subfolders, you can use it as argument:

Examples: JKCS2_runABC SYS_3SA SYS_4SA -gen 100

JKCS2_runABC SYS_1SA_1-5AM -pop 2000 -gen 150

The order of arguments does not matter except following commands: JKgaussstat, JKCS4_collect

JKcheck is command which tells you if all calculations where finished or not.

JKCS6_monster is good to use just after consultation. (no help yet)

JKCS3_run can be used if on any file resultsXXX.dat containing list of files without being in JKCS folder.

QUESTIONS:

Can I be sure that I have found global minimum?

-> No, never. But you can use proper configurational sampling, and thus, reach at least energetically very low structures.