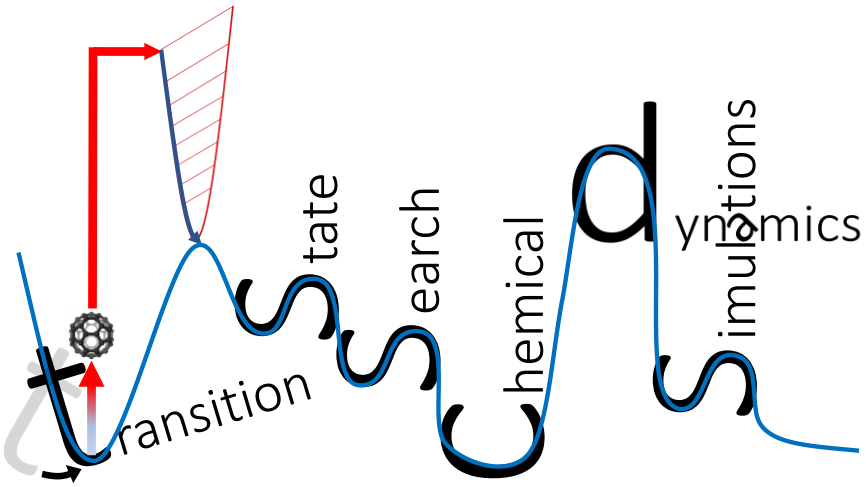


tsscds2018

transition state search using chemical dynamics simulations

Main developer:

Emilio Martínez-Núñez
Departamento de Química Física, Facultade de Química
Avda. das Ciencas s/n
15782 Santiago de Compostela, SPAIN
emilio.nunez@usc.es

With contributions from:

George L. Barnes, Aurelio Rodríguez, Roberto Rodríguez-Fernández, James J. P. Stewart and Saulo A. Vázquez

Contents

1. Introduction

The tsscds2018 program package has been designed to discover reaction mechanisms and solve the kinetics in an automated fashion, using chemical dynamics simulations. The basic idea behind this program is to obtain transition state (TS) guess structures from trajectory simulations performed at very high energies or temperatures. From the obtained TS structures, minima and product fragments are determined following the intrinsic reaction coordinate (IRC). Then, with all the stationary points, the reaction network is constructed. Finally, the kinetics is solved using the Kinetic Monte Carlo (KMC) method.

The program is interfaced with MOPAC2016 and Gaussian 09 (G09), but work is in progress to incorporate more electronic structure programs.

This tutorial is thought to guide you through the various steps necessary to predict reaction mechanisms and kinetics of unimolecular decompositions. To facilitate the presentation, we consider, as an example, the decomposition of formic acid (FA). The present version of the program can also be used to study homogeneous catalysis, but additional refinements are needed to make the code more general and user-friendly. This capability will be fully incorporated and described in the next released. Users are encouraged to read reference 1 before using the tsscds2018 package.

The present version has been tested on CentOS 7, Red Hat Enterprise Linux and Ubuntu 16.04.3 LTS. If you find a bug, please report it to the main developer (emilio.nunez@usc.es). Comments and suggestions are also welcome.

2. How to cite the program

Publications showing results obtained with the tsscds2018 package should include the following references:

1) Martinez-Nunez, E. *Phys. Chem. Chem. Phys.* 2015, *17*, 14912–14921; Martinez-Nunez, E. *J. Comput. Chem.* 2015, *36*, 222–234.
2) MOPAC2016, Version: 16.307, James J. P. Stewart, Stewart Computational Chemistry, web-site: HTTP://OpenMOPAC.net.

3. Installation

Untar and unzip the file tsscds-SOURCE-2018.tar.gz:

```
tar xvfz tsscds-SOURCE-2018.tar.gz
```

Before installing tsscds, be aware that the following packages are needed:

bc, **environment-modules**, **gawk**, **gcc**, **gfortran**, **parallel**, **python-numpy**, **python-scipy**, **sqlite3**, **zenity**

You can install the missing ones manually, or you can use install-required-packages-distro.sh (where distro=ubuntu-16.4lts, centos7 or sl7), which will do the work for you. The ubuntu-16.4lts script installs all dependencies, but for the RHEL derivatives (centos7 and sl7) you have to install parallel separately, and you have two choices:

a) install-gnu-parallel-from-source.sh. This script installs parallel latest version from source thanks to Ole Tange (the author). Also it can fallback to a user private installation into $HOME/bin if you have not administrator permisions to install it globally.
b) install-gnu-parallel-from-epel.sh. Enables the EPEL repository and installs parallel from it.

To use any of the above scripts, first go to the tsscds-SOURCE-2018 folder:

```
cd tsscds-SOURCE-2018
```

The program runs using two levels of theory: semiempirical (or Low-Level LL) and ab initio/DFT (or High-Level HL). So far, the only program interfaced with tsscds to perform the ab initio/DFT calculations is G09. Therefore, if you want to perform the HL calculations G09 should be installed and should run like in this example: g09<inputfile>outputfile

These packages might also be useful to analyze the results:

gnuplot, molden, sqlitebrowser

Once the above packages are installed, go to the tsscds-SOURCE-2018 folder (if you are not already there) to configure and install the package:

```
cd tsscds-SOURCE-2018
./configure
```

This will install tsscds2018 in $HOME/tsscds-2018 by default. If you want to install it in a different directory, type:

```
./configure --prefix=path_to_program
```

Finally, complete the installation:

```
make
make install
make clean
```

The last command (make clean) is only necessary if you want to remove from the src directory the object files and executables created in the compilation process.

For convenience, and once "**Environment Modules**" has been installed, you can add to your *.bashrc* file the following line to use the tsscds module:

```
module use path_to_program/modules
```

where path_to_program is the path where you installed tsscds (e.g., $HOME/tsscds-2018).

4. How to start using the program

To start using any of the scripts described below, you have to load the tsscds/2018 module:

```
module load tsscds/2018
```

5. Finding reaction mechanisms and solving the kinetics

The first step in our strategy for finding reaction mechanisms involves running classical trajectories, using the MOPAC2016 program,[2] which contains several semiempirical Hamiltonians. The trajectories sample the potential energy surface at the selected semiempirical level (the default is PM7), and the tsscds2018 program locates transition states by using the bond breaking/formation search (BBFS) algorithm described in the tsscds papers.[1] Then, reactants and products connected by the transition states (TSs) are obtained by intrinsic reaction coordinate (IRC) calculations. Finally, a reaction network is constructed with all the elementary reactions predicted by the program. To increase the efficacy of the tsscds2018 program, this process may be carried out in an iterative fashion as described in reference 1a. Once the reaction network has been predicted at the semiempirical level, the user can calculate rate constants for all the elementary reactions and run Kinetic Monte Carlo (KMC) calculations to predict the time evolution of all the chemical species involved in the global reaction mechanism and to calculate product ratios.

All the above steps can be run in an automatic fashion, using a single script as described below. However, the tsscds2018 package allows you the possibility to run the steps separatelly. This is important for checking purposes and, particularly, for the screening of structures, since you may need to adjust the screening parameters to your system (see below).

In a subsequent step, the collection of TSs located at the semiempirical level are reoptimized using a higher level of electronic structure theory. Notice that, depending on the selected level of theory, the total number of reoptimized TSs may differ from that obtained with the semiempirical Hamiltonian. For each reoptimized TS, IRC calculations are performed to obtain the associated minima (reactant and products). The reaction network is then constructed for the high level of theory. As for the low-level computations, the last step involves the calculation of rate constants and product ratios. As detailed below, all the high-level steps can be run separately, employing different scripts, or in an automatic way, using a single script. At present, all the high-level electronic structure calculations are performed with the G09 program.

To follow the guidelines of this tutorial, you can try the formic acid (FA) test case that comes with the distribution. Make a working directory and copy files *FA.dat* and *FA.xyz* from path_to_program/examples to your working directory. All the scripts described below (except select.sh) must be run in your working directory.

CAVEAT: use short names for the working directory and the input files. Otherwise, there may be crash problems.

a) Description of the input files

To run the tsscds2018 program, the user only needs two files:

i) ***molecule.xyz*** (*FA.xyz* in our example). Here *molecule* is the name of our system (FA in this case) This file contains the Cartesian coordinates of the system, usually the most stable conformer of the reactant molecule (for unimolecular decomposition).

ii) ***molecule.dat*** (*FA.dat* in our example). This file contains all parameters of the calculation and has different sections, which are explained as follows.

General section. In this section, the user provides keywords for all the electronic structure calculations. In our example, this section reads

```
--General section--
molecule FA
HighLevel b3lyp/6-31G(d,p)
HL_rxn_network complete
charge 0
mult 1
```

The following keywords can be used in this section:

molecule: refers to the name of the system. In our example, we used the name "FA". The Cartesian coordinates of the molecule must be specified in a file named *FA.xyz*, which must be located in the working directory. We notice that the tsscds2018 scripts use case-sensitive filenames.

LowLevel: is any of the semiempirical methods implemented in MOPAC2016. PM7 is the default method (you do not need to specify it). You can use a combination of MOPAC keywords. For instance, PM7 singlet excited state calculations can be run using:

```
LowLevel pm7 singlet cis c.i.=6 root=2 meci
```

Highlevel: indicates the level of theory employed in the high-level calculations described in section 7. You can employ a dual-level approach, which includes a higher level to refine the energy, as shown in the following example:

```
HighLevel ccsd(t)/6-311+G(2d,2p)//b3lyp/6-31G(d,p)
```

Supported methods are HF, MP2 and DFT for geometry optimizations and HF, MP2, DFT and CCSD(T) for single point energy calculations.

HL_rxn_network: is used to specify whether or not all the TSs located at the low level (e.g., PM7) will be reoptimized at the high level. The option *complete* indicates that all the TSs will be reoptimized. Alternatively, you may use the option *reduced*. This eliminates a series of TSs, as explained below.

charge: is the charge of the system.

mult: is the multiplicity of the system.

CAVEAT: all the keywords are case sensitive.

CDS (Chemical Dynamics Simulations) section. Here the user provides details of the accelerated dynamics simulations. In our example, we have

```
--CDS section--
sampling microcanonical
ntraj 10
```

The most important keywords available for this section are detailed as follows.

sampling: This keyword has four different options: *microcanonical*, *canonical*, *association* and *external***.** The options *microcanonical* and *canonical* refer to the type of initial conditions used to run classical trajectories. Both options are equally recommended. The canonical sampling allows the user to include partial constraints in the trajectories, which may be useful for large systems (see the "advanced users" section for more details). The *microcanonical* and *canonical* options have associated the following keywords:

ntraj: is the number of trajectories.

seed: can be employed to run a test trajectory (optional). This is the seed for the random number generator. If you plan to run more than one trajectory do not use this keyword, and every trajectory will have a different random number seed.

The sampling options *association* and *external* are explained in sections f and "advanced users", respectively.

BBFS (Bond Breaking/Formation Search) section. The BBFS algorithm selects TS guess structures monitoring changes in the adjacency matrix.[1b] The user can impose some constraints based on the imaginary frequency of the structures found:

```
--BBFS section--
freqmin 200
```

Here, the keyword ***freqmin*** refers to the minimum imaginary frequency (in absolute value and cm^{-1}) considered for the selection of TSs. This option can be used to avoid (or minimize) the selection of possible TSs of van der Waals complexes in which the imaginary frequency is associated with intermolecular degrees of freedom.

Structure screening section. The tsscds2018 program collects a series of structures associated with transition states of the potential energy surface of the system. Some of these structures might correspond

to the same transition state. Furthermore, some of the structures may correspond to transition states of van der Waals complexes formed upon fragmentation of the reactant molecule. To avoid or minimize repeated structures and van der Waals complexes, the tsscds2018 package includes a screening tool, which is based on the values of the following features calculated for each structure: energy, SPRINT coordinates,[3] degrees of each vertex and eigenvalues of the Laplacian matrix.[1a] Comparing these values for two structures, the mean absolute percentage error (MAPE) and the biggest absolute percentage error (BAPE) are obtained. The keywords ***avgerr*** and ***bigerr*** set the maximum values for MAPE and BAPE, respectively, which are used for screening. If both the MAPE and BAPE values calculated for two structures are below the ***avgerr*** and ***bigerr*** values, respectively, the structures are considered to represent the same transition state, and therefore only one of these structures is included in the TSs list. The values used in the FA example are:

```
--Screening of the structures section--
avgerr 0.008
bigerr 2.5
thdiss 0.1
```

The last keyword, called ***thdiss***, refers to the eigenvalues of the Laplacian (EL). This keyword gives the threshold for an EL to be considered 0. Therefore, in our example, if an EL<0.1, then this EL is set to 0. The number of zero ELs provides the number of fragments in the system. This criterion is used to identify van der Waals complexes that are formed by unimolecular fragmentation.

Kinetics section. This part is employed to provide details for the kinetics calculations at the (experimental) conditions you want to simulate. An example is given as follows.

```
--Kinetics section--
Rate  microcanonical
EKMC  150
```

The most relevant keywords of this section are the following.

Rate: can either be *canonical* or *microcanonical*, which means that the rate constants will be calculated according to Transition State Theory (TST) or Rice-Ramsperger-Kassel-Marcus (RRKM) theory, respectively. ***EKMC*:** If rate is microcanonical, this is the energy (in kcal/mol) for which microcanonical rate coefficients will be calculated.

***TKMC*:** If rate is canonical, this is the temperature (in K) for which thermal rate coefficients will be calculated. At present, temperatures below 100 K are not allowed.

b) Running the dynamics in a single processor

Canonical and microcanonical sampling methods provide initial coordinates and momenta to run accelerated dynamics simulations. Select the number of trajectories with *ntraj* and remember to avoid the *seed* keyword if the number of trajectories is greater than 1. In you want to run 10 trajectories, your CDS section should look like (remember that the tsscds/2018 module must be loaded):

```
--CDS section--
sampling microcanonical
ntraj 10
```

The dynamics can be run either in a single processor or in parallel. To run trajectories in a single processor use the tsscds.sh script:

```
tsscds.sh FA.dat >tsscds.log &
```

The ouput file *tsscds.log* provides information about the calculations. In addition, a directory called tsdirLL_FA is created, which contains information that may be useful for checking purposes. We notice that the program creates a symbolic link to the *FA.dat* file, named *tsscds.dat*, which is used internally by several tsscds2018 scripts. At any time, you can check the transition states that have been found using:

```
tsll_view.sh
```

The output of this script will be something like this:

```
  ts #  MOPAC file name  w_imag    Energy     w1     w2     w3     w4 traj #   Folder
  ----  ---------------  ------    ------   ----   ----   ----   ---- ------   ------
     1           ts1_FA 1587.3i    -35.71  204.3  438.3  461.3  726.8      1       FA
     2           ts2_FA 2009.6i    -17.61  327.2  472.7  522.7 1078.6      2       FA
     3           ts3_FA 2930.8i    -20.17  450.6  586.9  908.6  997.2      7       FA
```

where the first column is the label of each TS, the second is the filename of the MOPAC output (located in the tsdirLL_FA directory), the third is the imaginary frequency (in cm^{-1}), the fourth one is the energy in kcal/mol (actually, the heat of formation calculated by MOPAC2016) and the next four numbers are the four lowest vibrational frequencies (in cm^{-1}). Finally, the last two columns are the trajectory number and the name of the folder where the accelerated dynamics were run.

CAVEAT: since the dynamics employ random number seeds, the above results may differ from those obtained in your computer.

As already mentioned, the MOPAC2016 output files of the optimized TSs are stored in tsdirLL_FA. You can use a visualization program (e.g., Molden) to analyze your results. Try, for instance:

```
molden tsdirLL_FA/ts1_FA.out
```

You can also watch the animation of trajectories, which are stored in the coordir folder inside the working directory:

```
molden coordir/FA_dyn1.xyz
```

We notice that the coordir folder is temporary. It is removed during the execution of a subsequent script.

c) Running the dynamics in multiple processors

If you have access to several processors and want to run the dynamics in parallel, you can use the script tsscds_parallel.sh, which is executed interactively (a Zenity progress bar will appear on the screen). For instance, to submit 50 trajectories split in 5 different tasks (10 trajectories each) you should use:

```
tsscds_parallel.sh FA.dat 5
```

This will create temporary directories batch1, batch2, batch3, batch4 and batch5 that will be removed when the IRCs are calculated. Each of these folders includes a coordir directory, which contains the individual trajectories. The TSs found in each individual task will be copied in the same folder, tsdirLL_FA, and, as indicated above, using the tsll_view.sh script you can monitor the progress of the calculations. Notice that the total number of trajectories is given by *ntraj* (value specified in the CDS section) multiplied by the number of tasks. We recommend running the tsscds_parallel.sh script interactively only for checking purposes, and particularly to carry out the screening. To run many trajectories for production, we recommend using the llcalcs.sh script, which is described below.

If the Slurm Workload Manager is installed on your computer, you can submit the jobs to Slurm using:

```
sbatch [options] tsscds_parallel.sh FA.dat ntasks
```

where *ntasks* is the number of tasks. If no options are specified, *sbatch* employs the following default values:

```
#SBATCH --output=tsscds_parallel-%j.log
#SBATCH --time=04:00:00
#SBATCH -c 1 --mem-per-cpu=2048
#SBATCH -n 8
```

These values can be changed when you submit the job with *options*.

CAVEAT: if you use Slurm Workload Manage for the tsscds_parallel.sh script, you will have to wait until all tasks are completed before going on.

d) Analyzing the dynamics results

1) The tsscds package includes the irc.sh script, which performs intrinsic reaction coordinate calculations for all the located TSs. This script also allows one to perform an initial screening of the TS structures before running the IRC calculations:

```
irc.sh screening
```

This will do the screening and stop. The process involves the use of tools from Spectral Graph Theory and utilizes the three threshold values indicated above: *avgerr*, *bigerr* and *thdiss*. The redundant and fragmented structures are printed on screen as well as in the file *screening.log*. The MOPAC2016 ouput files are gathered in tsdirLL_FA, and use filenames initiated by "REPEAT" and "DISCNT", which refer to repeated and disconnected (i.e., fragmented) structures, respectively. Please check these structures and, if needed, change the above parameters. Should you change some of the above parameters (*avgerr, bigerr, thdiss*), you need to redo the screening with the new parameters:

```
redo_screening.sh
```

You can repeat the above process until you are happy with the "screening".

Once you are confident with the threshold values, you can submit many trajectories to carry out a thorough exploration of the potential energy surface. Subsequently, you can proceed with the IRC calculations.

2) Obtaining the IRCs:

```
(sbatch [options]) irc.sh
```

3) Optimizing the minima:

```
(sbatch [options]) min.sh
```

4) Creating the reaction network:

```
rxn_network.sh
```

Once you have created the reaction network, you can grow your TS list by running more trajectories (with tsscds_parallel.sh or tsscds.sh). Now the trajectories will start from the newly generated minima as well as from the main structure, specified in the *molecule.xyz* file. It is important to notice that, in general, trajectories run in separate batches (i.e., performed in several tasks) may be initialized from different minima and will have different energies. In this regard, the efficiency of the code may increase if the calculations are submitted using a large number for the *ntasks* parameter.

Convergence in the total number of TSs can be checked doing:

```
track_view.sh
```

When you are happy with the obtained TSs or you achieve convergence, you can proceed with the next steps.

5) Solving the kinetics using KMC with the parameters given in the kinetics section:

```
kmc.sh
```

6) Gathering all relevant information in folder FINAL_LL_FA:

```
final.sh
```

This folder will gather all the relevant information data, which are described below.

e) Running all low-level calculations using a single script

All the above steps can be done automatically using a single script, called llcalcs.sh. To run this script on a workstation with the GNU Parallel tool, type

```
nohup llcalcs.sh molecule.dat ntasks niter runningtasks >llcalcs.log 2>&1 &
```

where *ntasks* is the number of tasks for tsscds_parallel.sh, *niter* is the number of tsscds iterations, and *runningtasks* is the number of simultaneous tasks (useful if the workstation is shared by several users). The script can be run without the arguments (i.e., *molecule.dat*, *ntasks*, niter and *runningtasks*), and two pop-up windows will help you enter the arguments.

Finally, if your computer system has the Slurm job scheduler, you can submit the calculations as follows:

```
sbatch llcalcs.sh molecule.dat ntasks niter
```

CAVEAT: the use of llcalcs.sh is recommended once you have verified that the screening process works fine for your system.

During the execution of the llcalcs.sh script, the user may know the total number of trajectories completed at a given time using the script ntraj.sh. This also works for executions with the tsscds_parallel.sh script.

f) Running the high-level calculations

Once the the low-level calculations have been completed, the user can perform the high-level computations, which use the G09 program. These include the optimization of TSs, IRC calculations, optimization of minima and products, construction of the reaction network, calculation of rate coefficients and evaluation of the time evolution of the chemical species involved in the global reaction mechanism. All these steps can be performed in an automatic fashion using the hlcalcs.sh script, employing the following sentence (for the FA example):

```
nohup hlcalcs.sh FA.dat runningtasks >hlcalcs.log 2>&1 &
```

As for the low-level calculations, the argument *runningtasks* is the maximum number of tasks that can be run simultaneously in your computer. If your computer system has the Slurm job scheduler, the calculations can be submitted in the following way:

```
sbatch hlcalcs.sh FA.dat
```

Although we recommend using the automatic procedure for the simulation of reaction mechanism and kinetics at the high level, it is possible to perform the calculations step by step, as described next:

1. From your working directory (FA in the example), run:

```
(sbatch [options]) TS.sh FA.dat
```

In this case, the default values for a job submitted to Slurm are:

```
#SBATCH --time=04:00:00
#SBATCH -n 4
#SBATCH --output=TS-%j.log
#SBATCH --ntasks-per-node=2
#SBATCH -c 12
```

2. The scripts needed to build the reaction network and solve the kinetics are the same as those described above for the *LL* calculations. Namely:

```
(sbatch [options]) IRC.sh
(sbatch [options]) MIN.sh
RXN_NETWORK.sh
KMC.sh
```

Remember that the use of Slurm involves checking that every script has finished before proceeding with the next one.

3. The product fragments are optimized using

```
(sbatch [options]) PRODs.sh
```

CAVEAT: Step 3 is mandatory before proceeding to step 4. Run step 3 only when you are sure the first two steps have been successfully completed and you do not need to add more transition states.

4. To make a summary of the calculations in folder FINAL_HL_FA:

```
FINAL.sh
```

We notice that the high-level calculations also generate the directory tsdirHL_FA, which is the counterpart of the tsdirLL_FA folder. Finally, remember that you can use the kinetics.sh script to calculate rate coefficients and product branching rations for an energy or temperature different from that specified in the kinetics section of the *molecule.dat* file (*FA.dat* in our example).

g) Aborting tsscds calculations

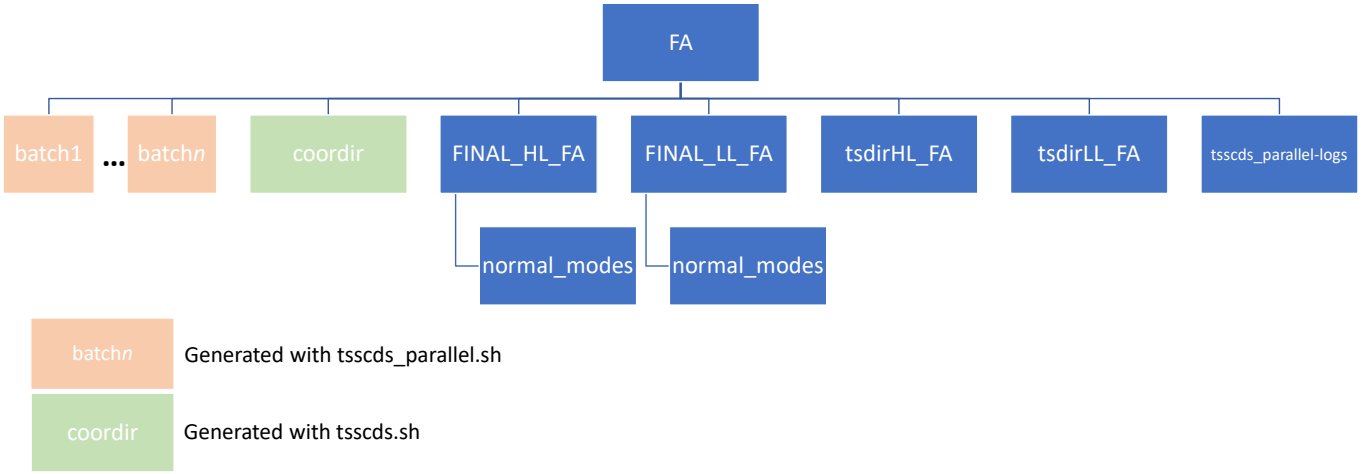
If, for any reason, you want to kill all the calculations, execute the following script from the working directory:

```
abort.sh
```

This script kills the processes whose PID are specified in these hidden files: *.parallel.pid* and *.script.pid*. We notice that, if G09 jobs are killed, the read-write files (Gau-#####) generated in the Gaussian scratch directory are not removed. The user should do it manually.

h) Directory tree structure of the working directory

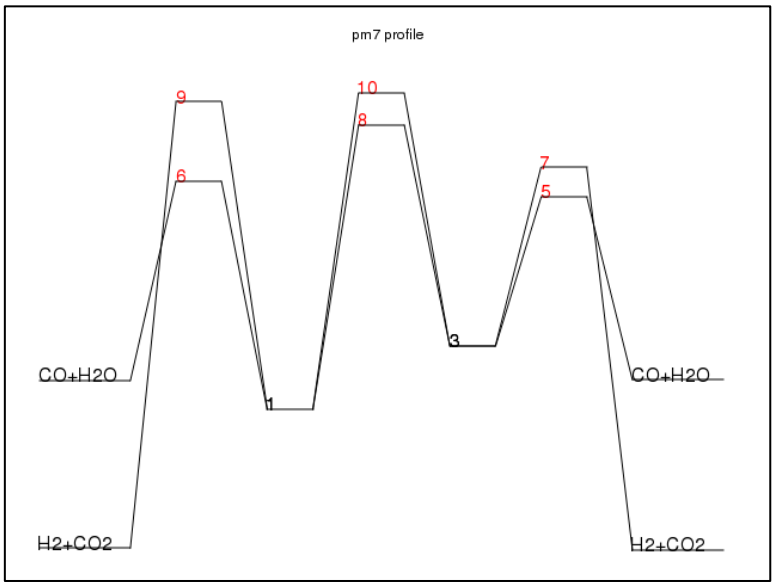
The figure below shows the main folders that are generated in the working directory. Folders batch1, batch2, and so on, include a coordir directory, which contains the individual trajectories computed in the associated task. The directories shown in blue will remain at the end of the calculations, while the other ones are temporary. The tsscds_parallel-logs directory contains a series of files that give information on CPU time consumption for the different calculation steps when they were executed with GNU Parallel. The most important files and the information they contain are described in the next section.



i) Relevant information

As already mentioned, the scripts final.sh and FINAL.sh collect all the relevant information in folders FINAL_LL_FA and FINAL_HL_FA, respectively (for our example in which *molecule* is *FA*). These folders contain some files as well as a subdirectory called normal_modes, which includes, for each structure, a file (in MOLDEN format) with which you can visualize the corresponding normal modes. The files included in FINAL_XL_FA (XL = LL or HL) are the following.

Energy_profile.gnu: is a gnuplot data file with which you can plot an energy diagram with the **relevant paths**. If you change the value of *ImpPaths* in the kinetics section of the input data (*FA.dat* in our case), you will incorporate/remove some pathways. In our example, the energy diagram is the following:



MINinfo: contains information of the minima:

```
MIN #    DE(kcal/mol)
    1       -8.340
    2        0.000
    3        5.283
    4        6.710
    5       15.338
Conformational isomers are listed in the same line:
1 2
3 4 5
```

TSinfo: contains information of the TSs:

```
TS  #    DE(kcal/mol)
    1        1.873
    2        9.625
    3       25.137
    4       32.852
    5       37.596
    6       40.962
    7       43.960
    8       53.165
    9       58.155
   10       60.011
   11       90.312
Conformational isomers are listed in the same line:
8 10
```

In the above files, DE is the energy relative to that of the main structure specified in the *FA.dat* file (optimized with the semiempirical Hamiltonian). The integers are used to identify, independently, minima and transition states. Notice that, in this example, MIN 2 corresponds to the structure specified in *FA.xyz*.

table.db: with *table* being *min*, *prod* or *ts*. These are SQLite3 tables containing the geometries, energies and frequencies of minima, products and TSs, respectively. The different properties can be obtained using the select.sh script, which should be run in the FINAL_LL_FA (or FINAL_HL_FA) folder:

```
select.sh property table label
```

where *property* can be: *natom*, *name*, *energy*, *zpe*, *g*, *geom*, *freq*, *formula* (only for prod) or *all*, and label is one of the numbers shown in RXNet (see below), which are employed to label each structure. At the semiempirical level, the energy values correspond to heats of formation. For high-level calculations, the tables collect the electronic energies. As an example, to obtain the geometry of the first transition state, you should use:

```
select.sh geom ts 1
```

RXNet: contains information of the complete reaction network, that is all the elementary reactions found by the tsscds-2018 program.

```
TS #    DE(kcal/mol)    -------Path info--------
    1        1.873        MIN    1 <-->  MIN    2
    2        9.625        MIN    3 <-->  MIN    4
```

```
    3       25.137        MIN    1 <-->  MIN    1
    4       32.852       PROD    1 <--> PROD    2
    5       37.596        MIN    4 <--> PROD    2
    6       40.962        MIN    1 <--> PROD    2
    7       43.960        MIN    3 <--> PROD    1
    8       53.165        MIN    1 <-->  MIN    4
    9       58.155        MIN    2 <--> PROD    1
   10       60.011        MIN    2 <-->  MIN    5
   11       90.312       PROD    2 <--> PROD    7
PROD 1 H2 + CO2
PROD 2 CO + H2O
PROD 7 H2 + CO2
```

As can be seen, for each transition state, this file specifies the associated minima and/or products and their corresponding identification numbers. Notice that TSs, minima (MIN) and products (PROD) have independent identification numbers. If you use the option *complete* for the keyword *HL_rxn_network* (in the General section of the input data), all the TSs will be reoptimized in the high-level calculations. You may reduce significantly the number of TSs to be reoptimized in the HL calculations, and therefore the reaction network, if you use the option *reduced*. If it is employed without an argument, TSs associated to PROD ↔ PROD steps (i.e., bimolecular reactions) and to interconversion between optical isomers (e.g., TS 3) will not be reoptimized in the HL calculations. You may include a number as an argument of this option:

```
HL_rxn_network reduced 55
```

In this case, besides the above TSs, all TSs having relative energies larger than 55 kcal/mol will not be considered for HL reoptimizations, that is, they will not be included in the HL reaction network. We notice that the last argument must be an integer.

RXNet.cg: By default (see below) the KMC calculations are "coarse-grained", that is, conformational isomers form a single state, which is taken as the lowest energy isomer. Such reaction network, which also removes bimolecular channels, is the following:

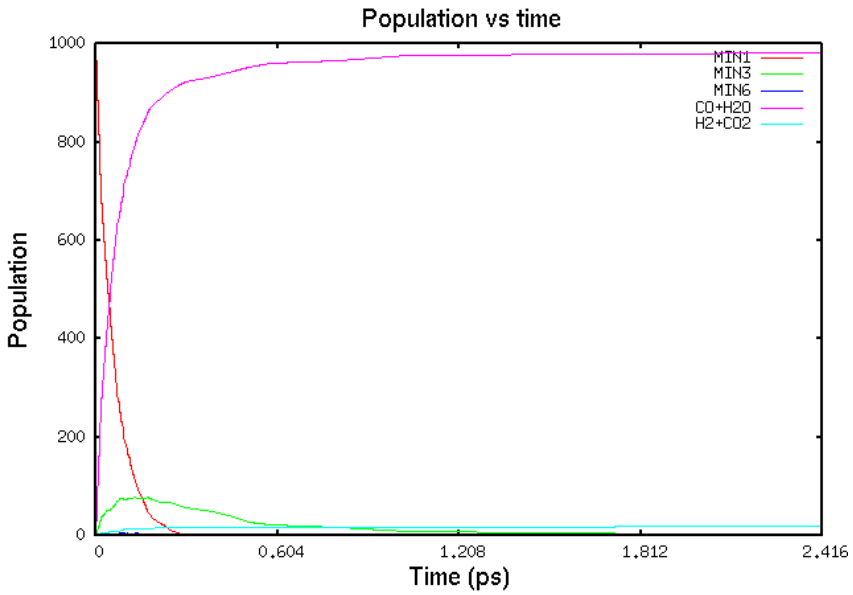
```
TS #    DE(kcal/mol)    -------Path info--------
    5       37.596        MIN    3 <--> PROD    2    CONN
    6       40.962        MIN    1 <--> PROD    2    CONN
    7       43.960        MIN    3 <--> PROD    1    CONN
    8       53.165        MIN    1 <-->  MIN    3    CONN
    9       58.155        MIN    1 <--> PROD    1    CONN
   10       60.011        MIN    1 <-->  MIN    3    CONN
PROD 1 H2 + CO2
PROD 2 CO + H2O
PROD 7 H2 + CO2
```

The last column with the flag "CONN" or "DISCONN" indicates whether the given process is connected with the others (CONN) or whether it is isolated (DISCONN). This flag is useful when you choose a starting intermediate for the KMC simulations, because that intermediate should be connected with the others. If

you want to include all conformational isomers explicitly in the KMC simulations, you need to construct the reaction network by using the *allstates* option, as described in the next section.

RXNet.rel: This file is similar to *RXNet.cg*, but only specifies the relevant paths, that is, those included in the *Energy_profile.gnu* file.

kineticsFvalue: This file contains the kinetics results, namely, the final branching ratios and the population of every species as a function of time. In the name of the file, F is either "T" or "E" for temperature or energy, and "value" is the corresponding value. For instance, the kinetics results for a canonical calculation at 298 K would be printed in a file called *kineticsT298*. A gnuplot file called *populationFvalue.gnu* is also available. It is a plot with the population of each species as a function of time. The following figure shows an example of such a plot obtained for the decomposition of FA using the PM7 stationary points.



j) Details of the kinetics simulations

By default, for the KMC simulations the different conformational isomers form a single state, which speeds up the calculations. If you prefer to treat each conformational isomer as a single state in the KMC calculations, you should run the rxn_network.sh script again (or RXN_NETWORK.sh for the high level), using the argument *allstates*, and solve the kinetics again. The following three scripts should be run to take all low-level conformational isomers into account in the KMC simulations:

```
rxn_network.sh allstates
kmc.sh
final.sh
```

The corresponding calculation for the high-level reaction network would be:

```
RXN_NETWORK.sh allstates
KMC.sh
PRODs.sh
FINAL.sh
```

When the calculations seek to simulate a thermal experiment (and therefore *rate canonical* is specified in the input file), the kinetics calculations can be rerun for a temperature different from that specified in the input file (using the *TKMC* keyword). This can be easily done using the kinetics.sh script with the following arguments:

```
kinetics.sh temp calc (allstates)
```

where temp is the new temperature of the system (in K), and calc is either ll (low-level) or hl (high-level). At this point, you should employ ll, but hl is available when you complete the hl calculations (vide infra). Finally, with no other options, the conformational isomers will form a single state (as above), and using allstates as the last argument, the calculations will regard every conformational isomer as a different state.

6. Other capabilities

a) Association complexes

The tsscds2018 package includes an option to predict association complexes. The input file for this type of calculation differs slightly from that used for unimolecular decompositions. Here the basic idea is to perform a series of full optimizations starting from separated molecules or fragments A and B. An example of such input file can be found in path_to_program/examples/assoc.dat. Two more additional files are also needed for this example, *cat.xyz* and *co.xyz*, which are also available in the same folder. The *assoc.dat* file contains the following data:

```
--General section--
charge 0
mult 1

--CDS section--
sampling association
A= cat
B= co
rotate 2 com 2.0 1.0

--Screening of the structures section--
avgerr 0.0001
bigerr 5
thdiss 0.1
```

This type of sampling only needs three sections: General, CDS and Screening. The CDS section only needs three keywords:

A: is the name of fragment A. A file with the Cartesian coordinates *fragA.xyz* (*cat.xyz* in this example) must be present in the working directory.

B: is the name of fragment B. A file with the Cartesian coordinates *fragB.xyz* (*co.xyz* in our example) must be present as well.

rotate**:** refers to the method employed to optimize the complexes. The calculation is carried out by taking 100 relative structures of both fragments, obtained via random rotations. Thus, the next two fields "**2 com**" indicate the pivot positions of the rotations: atom 2 of fragment A and the center of mass (com) of fragment B. The last two numbers "**2.0**" and "**1.0**" are distances, in Å, that indicate the distance between both pivots and the minimum intermolecular distance between any two atoms of both fragments, respectively.

With this sampling, you do not need the BBFS and kinetics sections. However, you still need to provide the parameters for the screening (*vide supra*).

To run the calculations, type:

```
tsscds.sh assoc.dat
```

This job will submit 100 independent optimizations to find the structures. After the jobs finished, the script will automatically remove duplicates and select the best association "complex".

You can check the optimized structures in folder assoc_cat_co. The program will also select the "best" structure according to the minimum number of structural changes between the complex and the individual fragments and its energy. The structure selected will be called *cat_co.xyz*. For fragments containing metals (like in this example), the selection is also based on the valence of the metal center. The file *assoclist_sorted* (in the assoc_cat_co folder) collects a summary of the structures and their energies, as well as the MOPAC2016 output files of each of them, which are called *assocN.out*, where N is a number from 1 to 100.

b) Advanced options

The following are keywords that can be useful for experienced users.

General section

iop: can be employed to specify an IOp in G09. Example:

```
HighLevel mpwb95/6-31+G(d,p)
iop iop(3/76=0560004400)
```

CDS section

atoms: is analogous to *modes* (explained below) but for a canonical ensemble. It indicates the atoms that are excited when a canonical ensemble is employed (the default is *all*).

etraj: can be employed along with *microcanonical* sampling and is the energy (in kcal/mol) of the accelerated dynamics simulations. This can be a single value (200) or a range, in which case the energy is randomly selected in the given energy range (200-300 for instance).

If *etraj* is not specified, the program automatically employs the following range of energies: $[16.25 \times (s-1) - 46.25 \times (s-1)]$ kcal/mol, where *s* is the number of vibrational degrees of freedom of the system. The values 16.25 and 46.25 have been determined from the formic acid results and making use of RRK theory. Those are the initial values of *etraj*, but the program automatically adjusts the range to obtain at least 60% reactivity at the boundaries.

factorflipv: using the default options, trajectories are terminated after 500 fs (see the *fs* keyword) or when there is one interatomic distance, r_{ij}, that reaches 5 times its value at time equals zero (i.e., at the initial conditions of the trajectory). Using this option, the trajectories are propagated during 500 fs (or during the

simulation time specified using the keyword *fs*). In addition, a change in the atomic velocities is applied when the following relationship is fulfilled:

$$r_{ij} \geq factorflipv \times r_{ij}^{0}$$

where r_{ij}^{0} is the distance between atoms i and j at time = 0. Specifically, the program modifies the atomic velocities according to the following criteria:

$$\vec{v}_k = \begin{cases} -\vec{v}_k & \text{if } k = i \text{ or } j \\ -0.9 \times \vec{v}_k & \text{if } k \neq i \text{ or } j \end{cases}$$

This way, the trajectory continues exploring points in configuration space that may be close to possible transition states. This option may increase the efficiency of the program. We recommend a value of 3.0 or larger for *factorflipv*.

fs: is the simulation time (in fs). The default is 500.

modes: can be employed together with *microcanonical* sampling. The default is *all*, which means that all modes are excited. If you want to excite just the three lowest normal modes, you must specify:

```
modes 3 1,2,3
```

Notice that the labels of the normal modes must be comma separated.

temp: can be employed with *canonical* sampling and is the temperature, in K, of the accelerated dynamics. As for *etraj*, it can be a single number or a range.

In the absence of the *temp* keyword, the program automatically defines the following range of temperatures: $[5452.04 \times (s-1)/natom - 15517.34 \times (s-1)/natom]$ K, which has been optimized for formic acid. However, as for *etraj*, the boundaries are adjusted "on the fly" to obtain a minimum reactivity of 60%.

thmass: can be employed, together with *canonical*, to specify the required minimum mass (in a.u.) of an atom to be initially excited.

BBFS section

fastmode: using this keyword, only one point in the vicinity of a possible TS is selected for TS optimization, which is conducted with the Eigenvector Following algorithm, as implemented in MOPAC. This is the default.

slowmode: using this keyword, up to 3 points are picked in the vicinity of a possible TS and the Hessian is updated every 10 steps. Obviously, this option is slower than *fastmode*, but it might be tested when the BBFS algorithm is not able to find TSs with efficacy.

Kinetics section

***imin*:** this keyword is used to specify the starting minimum for the KMC simulations. The argument is an integer, which identifies the desired structure. The default is the starting reference structure. All the minima are listed in *MINinfo* file and the user must examine *RXNet.cg* file to check that the minimum is indeed connected with the other ones (last column of each pathway indicates this fact).

nmol: specifies the number of molecules for the KMC simulations. The default is 1000.

Stepsize: indicates that the population of all the species in a KMC run is printed every *Stepsize* reactions. The default is 10.

MaxEn: is the maximum allowed energy for a TS to be included in the reaction network. The default is 100 kcal/mol when *rate* is *canonical* and it equals EKMC when *rate* is *microcanonical*.

PathInfo: can be used to run the KMC simulations only for the relevant paths. In this case, you have to run the final.sh script and then you must specify the following keywords in the kinetics section before running kmc.sh again:

```
PathInfo Relevant
```

ImpPaths: is the minimum percentage of processes occurring through a particular pathway (in the KMC simulation) that has to be achieved in order to be considered relevant and finally included in the *Energy_profile.gnu* file. The default is 0.1; therefore, pathways that contribute less than 0.1% to product formation are not included in this file. If you want to include them all use 0. Notice that these pathways may refer to the "coarse-grained" mechanism (default option) or to the complete mechanism that includes conformational isomers (obtained by using the *allstates* option as described above).

c) Biased dynamics

The tsscds2018 package includes several methods to bias the dynamics towards specific reaction pathways. So far, these are the available options:

1) The first option uses the AXD algorithm described in Ref [4], with which selected bond lengths are not allowed to stretch more than 30% with respect to their initial values. This can be useful to prevent the breakage of certain bonds. This option can be used adding the following keyword in the CDS section:

```
nbondsfrozen 2
1 13
2 8
```

This would "freeze" two bond distances connecting atoms 1-13 and 2-8, respectively. The labels of the atoms must follow the line starting with ***nbondsfrozen***.

2) The second algorithm bias the dynamics towards a particular reaction mechanism. An example of this option is provided in file path_to_program/examples/FA_biasH2.dat (you also need *FA.xyz*), which illustrates a way to search for H_2 elimination transition states from formic acid. In this example, the keywords added to the CDS section are:

```
nbondsbreak 2
3 5
1 4
nbondsform 1
4 5
Kapparep  100
Kappa     100
rmin      0.5
iexp        1
irange     10
```

In this case, the reaction coordinate is composed of bond distances: 3-5, 1-4 and 4-5. While the first two bonds have to break, the last one has to be formed during the elimination of molecular hydrogen.

Keywords ***nbondsbreak*** and ***nbondsform*** follow the same syntax as *nbondsfrozen* above. The other keywords are explained as follows. A bias potential is added to the potential energy obtained using the semiempirical Hamiltonian. The bias potential has the following simple form:

$$V = \sum_{i=1}^{nbondsbreak} V_{rep}(r_i) + \sum_{j=1}^{nbondsform} V_{attrac}(r_j)$$

$$V_{rep}(r_i) = \frac{\kappa_{rep}}{r_i}$$

$$V_{attrac}(r_j) = \frac{r_{min}^{iexp}}{2} \times \frac{\kappa}{r_j^{2\times iexp}} - \frac{\kappa}{r_j^{iexp}}$$

where κ_{rep}, κ , r_{min} and $iexp$ are parameters corresponding to the keywords ***Kapparep***, ***Kappa***, ***rmin*** and ***iexp***, and their units are such that the potential energy is in kcal/mol and the distances in Å. Additionally, the keyword (parameter) ***irange*** corresponds to the time window (in fs) employed by BBFS.[1b] This parameter takes a default value of 20 fs when nonbiased simulations are performed.

A similar test can be performed on the same molecule to get the TS for H_2O elimination. The corresponding input file, *FA_biasH2O.dat*, is also available in directory path_to_program/examples. Additionally, a Diels-Alder reaction has also been tested (ethylene+1,3-butadiene→cyclohexene), using the input files *diels_bias.dat* and *diels.xyz* provided in the tsscds2018 distribution.

The following table shows the parameters employed for the above examples:

Diels-Alder rxn	H_2 elimination from FA	H_2O elimination from FA

κ_{rep}	200	100	100
κ	200	100	100
r_{min}	0.5	0.5	0.5
$iexp$	1	1	1
irange	10	10	10

The units are such that the potential energy is in kcal/mol and the distances in Å.

The above examples can be tested using the tsscds.sh script:

```
tsscds.sh inputfile
```

Should you try this second option, you have to optimize the parameters for your own system, perhaps starting from those collected in the above table.

References

1. (a) Martínez-Núñez, E., An automated transition state search using classical trajectories initialized at multiple minima. *Phys. Chem. Chem. Phys.* **2015,** *17*, 14912-14921; (b) Martínez-Núñez, E., An automated method to find transition states using chemical dynamics simulations. *J. Comput. Chem.* **2015,** *36*, 222-234.
2. Stewart, J. J. P. *MOPAC2016*, Stewart Computational Chemistry: Colorado Springs, CO, USA, HTTP://OpenMOPAC.net, 2016.
3. Pietrucci, F.; Andreoni, W., *Phys. Rev. Lett.* **2011,** *107*, 085504.
4. Martinez-Nunez, E.; Shalashilin, D. V., Acceleration of classical mechanics by phase space constraints. *J. Chem. Theor. Comput.* **2006,** *2* (4), 912-919.