KST48 Tutorial

**In this tutorial, I will provide you with several examples to show the usage and typical procedure for some of the most important areas. Always be sure to download the latest version before using KST48.*

Overview

KST48 contains only one .py file. After installing Python3 and Numpy, you will be able to run it anywhere. In this tutorial, each task shares the same directory structure: in the directory named after the task name, there exists *kst48.py* as the program, *inp* as the input file, and *JOBS/* as the working directory. Some may contain *2.sub*, which is the script for submitting the job to the author's HPC cluster.

Output files: *kst48.out* as the output file; Files like *XXX_A.gjf* as the converged geometry.

The following examples are included in the tutorial:

Singlet-triplet MECP for transition metal complex

S1-T1 MECP of a TADF molecule

Gaussian ONIOM Calculation: S1/T1 MECP of Styrene in Explicit DCM

ORCA CASSCF: S0/S1 crossing point of CH2

ORCA SF-TDDFT: S-T crossing point for CH2

Singlet-triplet MECP for transition metal complex

In this example, we are going to locate the S-T MECP for a Pincer-Co complex at PBE0-D3BJ/def2-SV(P) level. For transition metals, especially when its geometry is close to a crossing point, the close-shell KS-wavefunction may be unstable. Therefore, we will run KST48 in **stable** mode, in which a stable UKS wavefunction will be first located and used for further MECP optimization.

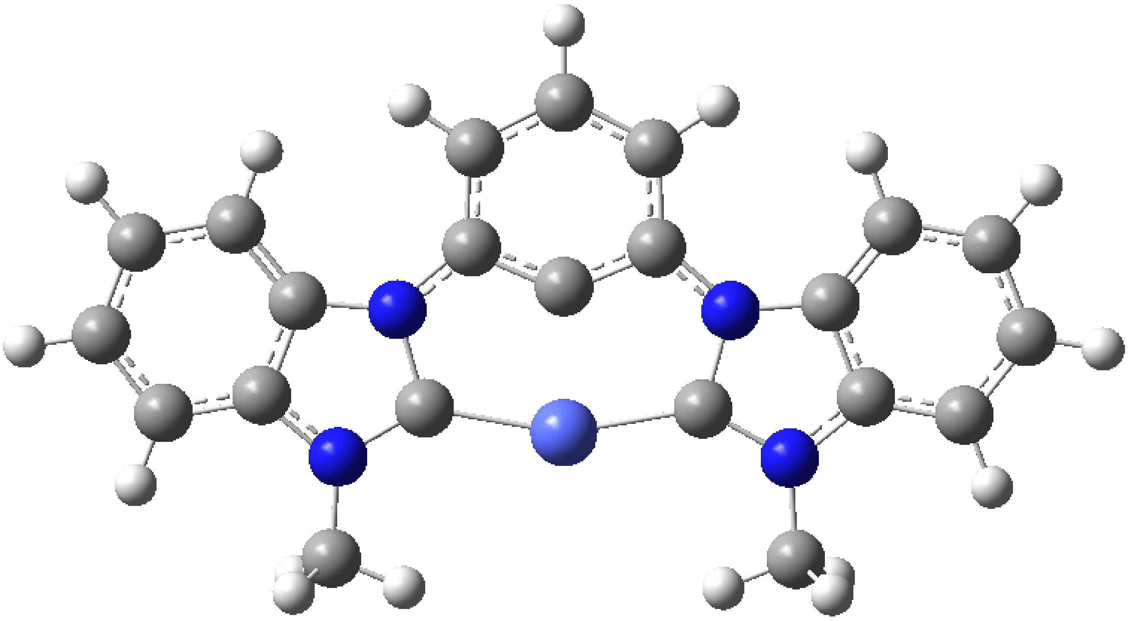
The triplet geometry was first optimized through *opt.gjf*, and the optimized geometry (or the latest geometry if the opt is not completed) will be read by KST48 by setting *geom @geom.log *:

```
nprocs = 56
mem = 96GB # change this into the maxcore value for orca
method = pbe1pbe em=gd3bj genecp g09default scf=(xqc,maxcycle=64)
td1 =  # keywords for td-dft (only for gaussian; please write it in the tail part for orca)
td2 =
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 1
mult2 = 3
mode = stable #normal; stable; read; inter_read; noread

program = gaussian  #gaussian, orca, xtb
gau_comm = 'g16'
orca_comm = '/opt/orca5/orca'
xtb_comm = 'xtb'

*geom
@opt.log
*
```

```
*tail1
-C -H -O -N 0
6-31g(d)
****
-Co 0
SDD
****

-Co 0
SDD


*
*tail2
-C -H -O -N 0
6-31g(d)
****
-Co 0
SDD
****

-Co 0
SDD


*
```

In this task, KST48 will first run stable=opt as pre_A.gjf and pre_B.gjf, and store the stable wavefunction as a.chk and b.chk. In this example, the singlet state is still close-shell even near the MECP, but it is not always the case for other complexes. Therefore it is always recommended to run the program under stable mode, or manually prepare stable wavefunction files and run in read mode.

The convergence is generally rapid. In this case, it converges within 41 iterations, and 41_A.gjf, 41_B.gjf, 41_A.log and 41_B.log are copied to the current directory.



S1-T1 MECP of a TADF molecule

2CzPN is a common TADF molecule, with a low energy gap for the T1 state to be re-converted to S1 state. In order to locate the S1-T1 MECP, we can set TD-DFT calculation for the state A, and simply set the multiplicity of state B to be 3 and run a ground-state calculation.

Here is the input file:

```
nprocs = 28
mem = 48GB # change this into the maxcore value for orca
method = 6-31g(d) m062x g09default scf=(xqc,maxcycle=64)
td1 = td # keywords for td-dft (only for gaussian; please write it in the tail
part for orca)
td2 =
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 1
mult2 = 3
mode = normal #normal; stable; read; inter_read; noread

program = gaussian  #gaussian, orca, xtb
gau_comm = 'g16'
orca_comm = '/opt/orca5/orca'
xtb_comm = 'xtb'
*geom
@opt.log
*

*tail1
*
*tail2
*
```

By setting *td1 = td*, a TD-DFT calculation will be performed for state A (multiplicity = 1). Of course you can write any arbitrary keywords, such as *td1 = td(nstates=5,root=2)* (then you will be able to locate an MECP involving S2 state). Crossing points such as S1-S2 can be located similarly by setting both *td1* and *td2*. Furthermore, as *td1* and *td2* allows for different keyword line for the two states, it may be useful for other applications other than TD-DFT.

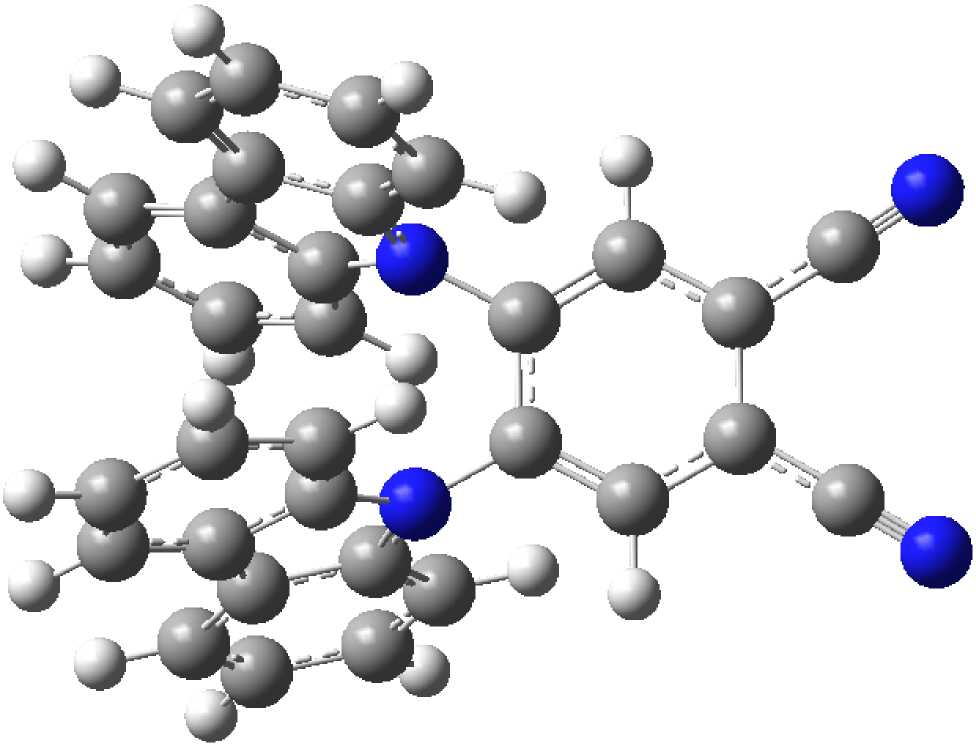
I read the optimization output file for the T1 state as the initial geometry. After 100 iterations, it is close but not yet converged. Therefore the latest .log file in the JOBS/ directory is copied to be cont.log, and the following inp file is used to continue optimizing:

```
nprocs = 28
mem = 48GB # change this into the maxcore value for orca
method = 6-31g(d) m062x g09default scf=(xqc,maxcycle=64)
td1 = td # keywords for td-dft (only for gaussian; please write it in the tail part for orca)
td2 =
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 1
mult2 = 3
mode = read #normal; stable; read; inter_read; noread

dE_thresh = 0.000050
rms_thresh = 0.0025 # 0.1*default
max_dis_thresh = 0.004 # 0.1*default
max_g_thresh = 0.0007
rms_g_thresh = 0.0005
max_steps = 100
max_step_size = 0.1
reduced_factor = 1 # the gdiis stepsize will be reduced by this factor when rms_gradient is close to converge

program = gaussian  #gaussian, orca, xtb
gau_comm = 'g16'
orca_comm = '/opt/orca5/orca'
xtb_comm = 'xtb'
*geom
@cont.log
*

*tail1
*
*tail2
*
```

There is no difference but the geometry file and the running mode. This time the *read* mode is used because I have already had proper a.chk and b.chk from the previous calculation.

Finally it is converged within 20 iterations.

```
Now Entering GDIIS Step 19
E1 = -1448.58247683
E2 = -1448.58243497
deltaE                    0.000042     0.000050     YES
RMS Gradient              0.000239     0.000500     YES
Maximium Gradient         0.000496     0.000700     YES
RMS Displacement          0.000785     0.002500     YES
Maximium Displacement     0.002919     0.004000     YES
****Congrats! MECP has converged****
```



In the inp file attached, I forgot to delete the genecp information in the tail part. It makes no difference to the calculation as it is never read by Gaussian.

Gaussian ONIOM Calculation: S1/T1 MECP of Styrene in Explicit DCM

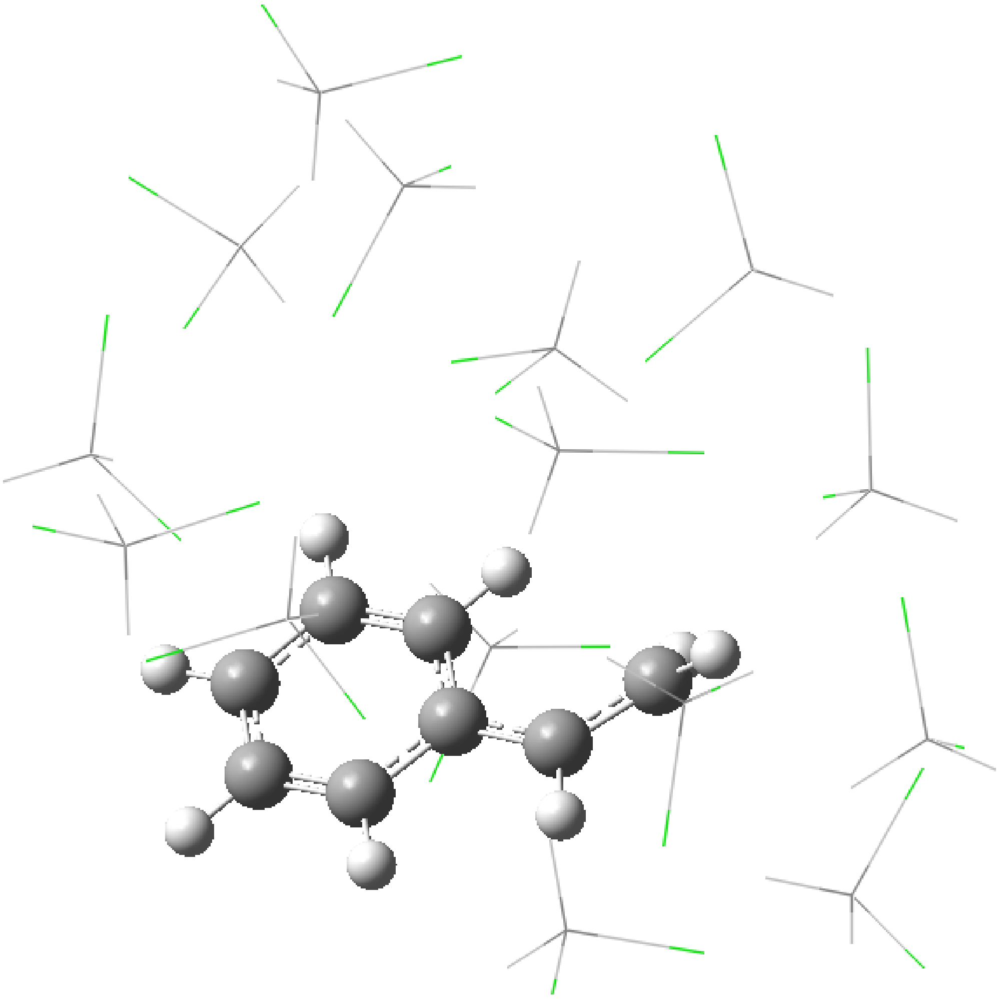
ONIOM calculation in Gaussian is supported by KST48 since 2023 Oct version. In this example, a styrene molecule is placed within 15 DCM molecules, and we are going to locate the S1/T1 MECP. Styrene is the high level with B3LYP-D3BJ/6-31G(d) method, and the solvent molecules are treated by PM6 method.

In a serious research, you may build the initial cluster geometry using the *packmol* program. In this tutorial, I simply made up an initial geometry by hand and optimized it using *xtb* program. The optimized geometry is stored as xtbopt.xyz. However, external geometry file link is not supported by ONIOM calculation. Therefore we have to set up proper .gjf format and copy the geometry part to *geom. Note that geometry generated by GaussView may contain a column ' 0 ', which must be removed. The keywords, charge and spin multiplicity is set by *isONIOM*, *chargeAndMultForONIOM1* and *chargeAndMultForONIOM2*.

```
nprocs = 28
```

```
mem = 48gb # change this into the maxcore value for orca
method = scf=(xqc,maxcycle=64)
td1 = oniom(td/b3lyp/em=gd3bj/6-31G(d):pm6)  # keywords for td-dft (only for gaussian; please write it in the tail part for orca)
td2 = oniom(b3lyp/em=gd3bj/6-31G(d):pm6)
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 1
mult2 = 3
mode = normal #normal; stable; read; inter_read; noread

program = gaussian  #gaussian, orca, xtb
gau_comm = g16
orca_comm = /opt/orca5/orca
xtb_comm = xtb

isONIOM = true # if you are using the ONIOM feature of Gaussian, set this true.
chargeAndMultForONIOM1 = 0 1 0 1 0 1  #only useful for ONIOM calculation
chargeAndMultForONIOM2 = 0 1 0 3 0 3 #only useful for ONIOM calculation

*geom
 C               0.54930783   -0.17331140   -0.17401924 H
...
*

*tail1
*
*tail2
*
```

The initial run did not converge within 100 iterations, so it is continued by copying the latest geometry to the *inp* file. This time, it is converged within 13 iterations.



ORCA CASSCF: S0/S1 crossing point of CH2

KST48 supports ORCA and BAGEL for multi-reference calculations. For the optimization of S0/S1 crossing point of CH2 at CASSCF(2,2) level, the following input file is used:

```
nprocs = 28
mem = 1000 # change this into the maxcore value for orca
method =  hf 3-21g
td1 =  # keywords for td-dft (only for gaussian; please write it in the tail part for orca)
td2 =
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 1
mult2 = 1
mode = normal #normal; stable; read; inter_read; noread

program = orca  #gaussian, orca, xtb
gau_comm = g16
orca_comm = /opt/orca5/orca
xtb_comm = xtb

*geom
 C                  1.40905202    0.37147736    0.00000000
 H                  1.76570645   -0.63733264    0.00000000
 H                  0.33905202    0.37149055    0.00000000
*

*tail1
% casscf nel 2 norb 2 nroots 2 weights[0] = 1,0
maxiter 200
OrbStep SuperCI_PT # or any other from the list above
SwitchStep DIIS # or any other from the list above
ShiftUp 2.0 # static up-shift the virtual orbitals
ShiftDn 2.0 # static down-shift the internal orbitals
MinShift 0.6 # minimum separation subspaces
end
*
*tail2
% casscf nel 2 norb 2 nroots 2 weights[0] = 0,1
maxiter 200
OrbStep SuperCI_PT # or any other from the list above
SwitchStep DIIS # or any other from the list above
ShiftUp 2.0 # static up-shift the virtual orbitals
ShiftDn 2.0 # static down-shift the internal orbitals
MinShift 0.6 # minimum separation subspaces
end
*
*constr
*
```

The CASSCF convergence is often problematic. As a tutorial, the 3-21G basis set is employed to achieve fast convergence. In the *tail1* and *tail2* part, we can write any keywords controlling the two ORCA tasks. In this example, CASSCF(2,2) calculation is performed to generate 2 roots and the state for the force computation is controlled by their weight.

The computation converges within 22 iterations. KST48 will copy the latest .inp, .out and .xyz file for the converged geometry to the current directory.

ORCA SF-TDDFT: S-T crossing point for CH2

There is nothing special for setting an SF-TDDFT calculation in KST48. **In their article "XMECP: Reaching State-of-the-Art MECP Optimization in Multiscale Complex Systems" (*J. Chem. Theory Comput.* 2024, 20, 9, 3590–3600), Minyi Zhang, Minyi Zhang and other authors deliberately ignored the existing functionalities of KST48 and viciousnessly claimed that KST48 does not support ONIOM and SF-TDDFT. In their response to JCTC Editor, they said that 'no one knows how to' use KST48. As a response, I will give a very straightforward example herein.**

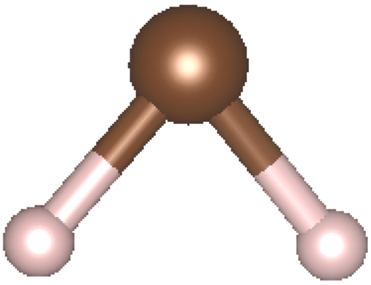
SF-TDDFT is believed to be superior for describing the potential energy surface near the crossing point. The following input file is employed to locate the S-T crossing point of CH2.

```
nprocs = 28
mem = 1000 # change this into the maxcore value for orca
method = b3lyp 6-31g(d)
```

```
td1 =  # keywords for td-dft (only for gaussian; please write it in the tail part
for orca)
td2 =
mp2 = false #set true for mp2 or doubly hybrid calculation in gaussian
charge = 0
mult1 = 3
mult2 = 3
mode = normal #normal; stable; read; inter_read; noread

program = orca  #gaussian, orca, xtb
gau_comm = g16
orca_comm = /opt/orca5/orca
xtb_comm = xtb

*geom
 C                  1.40905202    0.37147736    0.00000000
 H                  1.76570645   -0.63733264    0.00000000
 H                  0.33905202    0.37149055    0.00000000
*

*tail1
%TDDFT SF TRUE
NROOTS 2
IROOT 0
END
*
*tail2
%TDDFT SF TRUE
NROOTS 2
IROOT 1
END
*
```

SF-TDDFT requires the spin multiplicity be set to be the high-spin state. In this case, both states have a multiplicity of 3. The %TDDFT keywords in *tail1* and *tail2* controls ORCA to generate forces for the 0th and 1th states (it is numbered from 0 in ORCA).

The optimization is converged within 13 iterations.



An ACET Reaction

In an addition-coupled electron transfer (ACET) reaction (10.26434/chemrxiv-2021-qlh3r-v4, 10.1002/slct.202202354, 10.1021/acs.joc.3c01956), an electron transfer event occurs in a coupled manner with a nucleophilic addition. In many cases it occurs through a crossing point instead of a transition state. In this example, we will locate the crossing point for the following reaction:

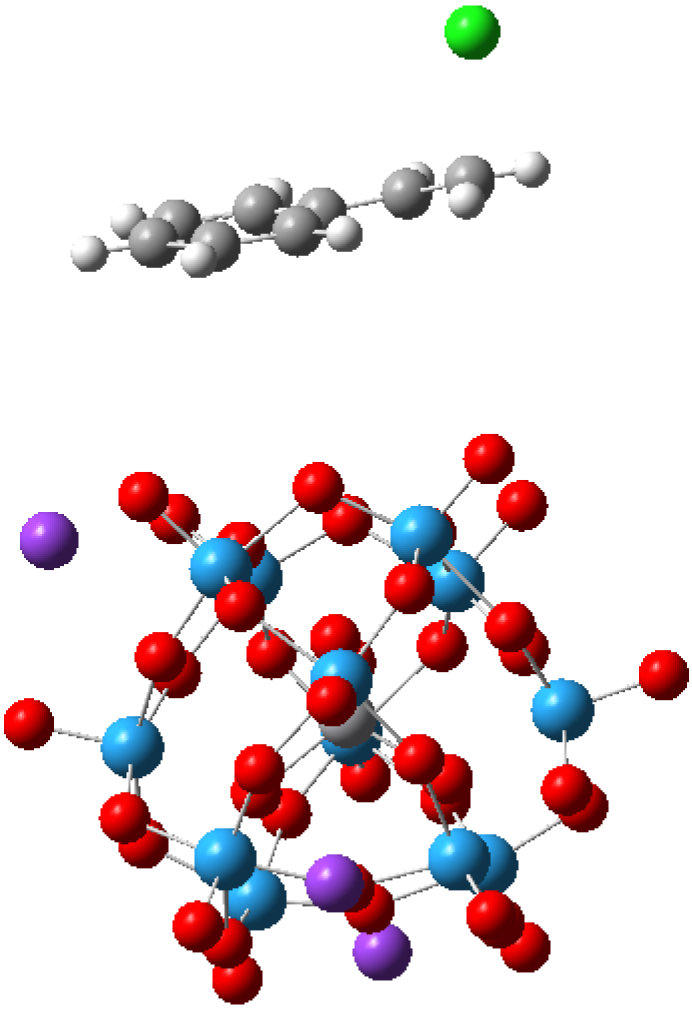
[Ox] + Cl- + styrene → PhCH·CH2Cl + [ox]-

It is the crossing point between the following two electronic states:

A: [Ox] + (Cl- + styrene)

B: [Ox]- + (Cl + styrene)·

In this case, [Ox] is a polyacid.



Both the two states represent a stable wavefunction. They can be obtained by guess=fragment keywords of Gaussian. In this example, *a.gjf* and *b.gjf* generates the two initial wavefunctions, and *c.gjf* reads *b.chk* and optimizes the state B to provide an initial geometry of the crossing point. Then the remaining thing is straightforward: let KST48 read the geometry of c.log and the two .chk files in *read* mode, and it converges within 77 iteractions.