MOMAP

Tutorial 11

ORCA Calculation with MOMAP

MOMAP
Molecular Material Property Prediction Package

Version 2021B

December, 2021

MOMAP Tutorial 11

Version 2021B edited by:

Dr. Deping Hu

Dr. Qikai Li

Released by Hongzhiwei Technology (Shanghai) Co., Ltd

and Z.G. Shuai Group

The information in this document applies to version 2021B of MOMAP

Table of Contents

ORCA Calculation with MOMAP

ORCA is a flexible, efficient and easy-to-use general purpose tool for quantum chemistry with specific emphasis on spectroscopic properties of open-shell molecules. It features a wide variety of standard quantum chemical methods ranging from semiempirical methods to DFT to single- and multireference correlated ab initio methods, and it is also actively evolving.

Details about the ORCA input file lines:

- Input files are pretty much free-format.
- Blank lines are allowed.
- Input is usually not case-sensitive.
- ! start a keyword line. In the simple input syntax, keywords are added in any order to the line beginning with "!". Multiple "!" lines are also allowed.
- # Comment lines can be added if the line begins with a #.
- % Start a block input. Advanced settings are often specified using the block input for different modules

Please refer to the ORCA Manual for details.

A typical way to perform an ORCA calculation from scratch is as follows:

1. Create a root directory and generate the `.xyz` file

Let us take `azulene` as an example. First create a root directory and enter into it,

```
$ mkdir orca-azulene
$ cd orca-azulene
```

Generate the atomic coordinates by any tool or program that you are familiar with, and save as the `.xyz` format which is a standard output format of all programs, e.g., `azulene.xyz`.

Next, we carry out the specific calculations in separate sub-directories.

2. Ground state geometry optimization and frequency calculation

Create a sub-directory `gs` for the ground state geometry optimization and frequency calculation, enter into that directory:

```
$ mkdir gs
$ cd gs
```

Create the ORCA input file `azulene-s0.inp` as shown below,

```
!B3LYP 6-31G* OPT FREQ
!PAL8
%MAXCORE 4000
```

```
* xyz 0 1
 C                  2.01378743    -1.48849852     0.00000000
 C                  2.28995141    -0.11795315     0.00000000
 C                  1.39185815     0.95357383     0.00000000
 C                  0.78413689    -2.15418449     0.00000000
 C                  0.00000000     0.93285810     0.00000000
 C                 -0.50398383    -1.61065958     0.00000000
 C                 -0.89316505    -0.27406276     0.00000000
 H                  2.88919252    -2.13621797     0.00000000
 H                  3.34387207     0.15083266     0.00000000
 H                  1.84191311     1.94635990     0.00000000
 H                  0.83658347    -3.24058384     0.00000000
 H                 -1.32037398    -2.33298523     0.00000000
 C                 -0.84567310     2.05536637     0.00000000
 H                 -0.51364908     3.08694089     0.00000000
 C                 -2.17758707     1.61062710     0.00000000
 H                 -3.04994479     2.25593917     0.00000000
 C                 -2.21339978     0.20656494     0.00000000
 H                 -3.10314368    -0.41207657     0.00000000
*
```

The coordinates are taken from the upper level `.xyz` file.

For more than eight processors (`!PAL8`), the explicit `%PAL` option has to be used:

```
%PAL NPROCS 16 END
```

Users can comment out the `!PAL8` line, as the `%PAL … END` line will be added with the actual used cores in the job script.

Use the job script shown later to launch the job, remember to change the variable `INPUT` in the job script to `azulene-s0`.

When the run is successful, the files may look like the following:

```
azulene-s0.densities azulene-s0.hess      azulene-s0.opt
azulene-s0.engrad    azulene-s0.inp       azulene-s0.out
azulene-s0.gbw       azulene-s0.log       azulene-s0.xyz
```

3. Excited state geometry optimization and frequency calculation

Create a sub-directory `es` for ground state geometry optimization and frequency calculation, enter into that directory:

```
$ mkdir es
$ cd es
```

Create the ORCA input file `azulene-s1.inp` as shown below,

```
!B3LYP 6-31G* OPT FREQ
%TDDFT NROOTS 3 END
!PAL8
%MAXCORE 4000
```

```
* xyz 0 1
 C                 2.01378700   -1.48849900    0.00000000
 C                 2.28995100   -0.11795300    0.00000000
 C                 1.39185800    0.95357400    0.00000000
 C                 0.78413700   -2.15418400    0.00000000
 C                 0.00000000    0.93285800    0.00000000
 C                -0.50398400   -1.61066000    0.00000000
 C                -0.89316500   -0.27406300    0.00000000
 H                 2.88919300   -2.13621800    0.00000000
 H                 3.34387200    0.15083300    0.00000000
 H                 1.84191300    1.94636000    0.00000000
 H                 0.83658300   -3.24058400    0.00000000
 H                -1.32037400   -2.33298500    0.00000000
 C                -0.84567300    2.05536600    0.00000000
 H                -0.51364900    3.08694100    0.00000000
 C                -2.17758700    1.61062700    0.00000000
 H                -3.04994500    2.25593900    0.00000000
 C                -2.21340000    0.20656500    0.00000000
 H                -3.10314400   -0.41207700    0.00000000
*
```

The coordinates are taken from the ground state optimized geometry.

Users can comment out this `!PAL8` line, as the `%PAL … END` line will be added with the actual used cores in the job script.

Use the job script shown later to launch the job, remember to change the variable `INPUT` in the job script to `azulene-s1`.

When the run is successful, the files may look like the following:

```
azulene-s1.cis       azulene-s1.gbw       azulene-s1.opt
azulene-s1.densities azulene-s1.hess      azulene-s1.out
azulene-s1.engrad    azulene-s1.inp       azulene-s1.xyz
```

4. Nonadiabatic coupling calculation

Create a sub-directory `nacme` for ground state geometry optimization, enter into that directory:

```
$ mkdir nacme
$ cd nacme
```

Create the ORCA input file `azulene-nacme.inp` as shown below,

```
! NumNACME B3LYP 6-31G*
%tddft
IROOT 1
nroots 3
end
!PAL8
%MAXCORE 4000

* xyz 0 1
  C       2.037282     -1.506296     -0.000000
  C       2.298173     -0.116341      0.000001
  C       1.419704      0.950186      0.000001
  C       0.784772     -2.163003     -0.000001
  C      -0.017204      0.877337      0.000001
  C      -0.492288     -1.635584     -0.000000
  C      -0.843698     -0.240494      0.000001
  H       2.911102     -2.152460     -0.000001
  H       3.354966      0.150126      0.000001
  H       1.850777      1.948488      0.000001
  H       0.839552     -3.251499     -0.000002
  H      -1.320544     -2.340172     -0.000001
  C      -0.886816      2.054171     -0.000002
  H      -0.525650      3.077097     -0.000004
  C      -2.224903      1.645101      0.000000
  H      -3.095269      2.288481     -0.000000
  C      -2.223705      0.245913      0.000002
  H      -3.095879     -0.399212      0.000002
*
```

The coordinates are taken from the excited state optimized geometry.

Users can comment out the `!PAL8` line, as the `%PAL … END` line will be added with the actual used cores in the job script.

Use the job script shown below to launch the job, remember to change the variable `INPUT` in the job script to `azulene-nacme.`

When the run is successful, the files may look like the following:

```
azulene-nacme.cis           azulene-nacme.gbw           azulene-nacme.out
azulene-nacme.densities     azulene-nacme.ges           azulene-nacme_property.txt
azulene-nacme.engrad        azulene-nacme.inp
```

Now the ORCA related calculations are done, we can continue to do the evc *etc*. calculations.

5. Perform evc calculation

Create a sub-directory `evc`, copy all the needed files from ORCA calculations to the directory:

```
$ mkdir evc
$ cd evc
$ cp ../gs/azulene-s0.hess ./
$ cp ../es/azulene-s1.hess ./
$ cp ../nacme/azulene-nacme.out ./
```

Create a MOMAP control file `momap.inp` with the following contents:

```
do_evc          = 1

&evc
  ffreq(1)    = "azulene-s0.hess"
  ffreq(2)    = "azulene-s1.hess"
  proj_nacme = 1
  fnacme      = "azulene-nacme.out"
  sort_mode   = 1
/
```

Finally, we can run the `momap.py` to perform the evc calculation as usual:

```
$ momap.py
```

Here we assume that you have set up the MOMAP environment properly.

The generated files should look like the following:

```
azulene-nacme.out evc.cart.dat      evc.dx.v.xyz      evc.vib1.xyz      momap.inp
azulene-s0.hess   evc.cart.nac      evc.dx.x.com      evc.vib10.xyz     nodefile
azulene-s1.hess   evc.dint.abs      evc.dx.x.xyz      evc.vib2.xyz
evc.cart.abs      evc.dint.dat      evc.out           evc.vib20.xyz
```

The users should check the contents carefully before performing the following tasks.

6. Job scripts

The scheduling system used in this tutorial is SLURM, a typical Slurm job script is shown as follows:

```
#!/bin/bash

#SBATCH --job-name=orca
#SBATCH --output=stdout.txt
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH --time=0-2:00:00

# Define variable "INPUT", change here!
INPUT=azulene-s0
#INPUT=azulene-s1
#INPUT=azulene-nacme

username=`whoami`

ulimit -s unlimited

# Define running command variable, need full path!
ORCADIR=/opt/orca/5_0_2_linux_x86-64_shared_openmpi411
EXEC=$ORCADIR/orca

# Define MPI
module purge
module load orca/5.0.2-openmpi411

# Make a scratch directory if it doesn't already exist.
SCRDIR=/scratch/${username}_$SLURM_JOB_ID
if [ ! -a $SCRDIR ]; then
   echo "Scratch directory $SCRDIR created."
   mkdir -p $SCRDIR
fi
export SCRDIR
echo "Using $SCRDIR for temporary ORCA files."

# Go to the work directory
cd $SLURM_SUBMIT_DIR
```

```
cp $SLURM_SUBMIT_DIR/* $SCRDIR
cd $SCRDIR

srun hostname -s | sort -n > hosts

echo "ORCA job start at" `date`
cat hosts

# edit .inp for parallel
echo "Numbers of Processors: $SLURM_NPROCS"
sed -i "1i end" ${INPUT}.inp
sed -i "1i %pal nprocs $SLURM_NPROCS" ${INPUT}.inp

# run program
time ${EXEC} ${INPUT}.inp > ${INPUT}.out

rm ${SCRDIR}/${INPUT}.inp
rm ${SCRDIR}/stdout.txt
mv ${SCRDIR}/* $SLURM_SUBMIT_DIR

echo "ORCA job finished at" `date`
echo "Work Directory is $SLURM_SUBMIT_DIR"

rm -rf $SCRDIR
```