1.3 Graphical User Interfaces (GUI)

Currently, there is no native graphical user interface (GUI) developed by the ORCA developers. However, there are several programs that provide functionalities to create ORCA inputs and read, and visualize ORCA output files. Some useful programs are outlined in Table 1.1 and some applications can be found at the ORCA tutorials website.

Warning

None of the programs mentioned are developed by the ORCA team. We can therefore not guarantee their functionality. Please contact the developers of the respective program in case of problems and feature requests.

Table 1.1: Selected programs that may be used as GUI to ORCA.

Program	Platform	Comment
Avogadro 2	Windows, Linux, MacOS	3D molecule builder, creation of ORCA input, visualization of ORCA outputs.
ChimeraX	Windows, Linux, MacOS	Visualization of ORCA outputs. Very powerful visualization suite. The SEQCROW plugin is very useful. It can be downloaded from the `Tools/More→Tools` menu.
Chemcraft	Windows, Linux	3D molecule builder, creation of ORCA input, visualization of ORCA outputs. For Linux, we recommend to use the Windows version via a virtual machine.

Note

The ORCA enhanced version of Avogadro 1 was not further developed or tested for ORCA 6 and beyond! We recommend to try Avogadro 2 instead.

1.4 Running ORCA

Under Windows the program is called from the command prompt! (Make sure that the PATH variable is set such that the orca executables are visible)

```
orca orca.inp > orca.out
```

Under UNIX based operating systems the following call is convenient (here also: make sure that the PATH variable is set to the directory where the orca executables reside):

```
orca orca.inp >& orca.out &
```

The program writes to *stdout* and *stderr*. Therefore the output must be redirected to the file *orca.out* in this example. *orca.inp* is a free format ASCII file that contains the input description. The program will produce a number of files *orca.x.tmp* and the file *orca.gbw*. The "*.gbw" file contains a binary summary of the calculation. GBW stands for "Geometry-Basis-Wavefunction". Basically this together with the calculation flags is what is stored in this file. You need this file for restarting SCF calculations or starting other calculations with the orbitals from this calculation as input. The "*.tmp" files are temporary files that contain integrals, Fock matrices etc. that are used as intermediates in the calculation. If the program exits normally all of these files are deleted. If it happens to crash you have to remove the files manually (`rm orca*.tmp` under Unix or `del orca*.tmp` under Windows). In case you want to monitor the output file while it is written, you can use the command (under Unix):

```
tail -f orca.out
```

to follow (option -f) the progress of the calculation. Under Windows you have to either open another command shell and use:

```
type orca.out
type orca.out |more
```

or you have to copy the output file to another file and then use any text editor to look at it.

```
copy orca.out temp.out
edit temp.out
```

you cannot use `edit orca.out` because this would result in a sharing violation.

1.4.1 A simple example

A great strength of ORCA is the simple structure of its input file. ORCA input files can be easily generated by any simple text editor. Simply create a file with an `.inp` extension, e.g. `orca.inp`, and add the input information to it. The most basic input file looks like this:

```
! METHOD

*XYZ CHARGE MULTIPLICITY
Atom1   X1  Y1  Y1
Atom2   X2  Y2  Y2
*
```

Here, we define our quantum mechanical method of choice via the so called *simple input line* that starts with an exclamation mark. Afterwards, we define the structure of our molecule, its net charge, and its multiplicity (M = 2S+1). In this example, we chose cartesian coordinates in XYZ format given in Ångström. The coordinates block must be closed by an `*`.

An example, using Hartee-Fock theory and a def2-SVP basis set for the neutral, singlet dihydrogen molecule looks like this:

```
! HF DEF2-SVP

*XYZ 0 1
H   0   0   0
H   0.74   0   0
*
```

More details on the ORCA input file syntax can be found in the section *General Structure of the Input File* and the section *Hello water! Your first ORCA calculation* will guide you through your first steps with ORCA.

1.5 Hello water! Your first ORCA calculation

After *setting up ORCA correctly*, you are ready to perform your first calculation with ORCA! Here, we will exemplarily compute the energy of a water molecule using *Hartree-Fock theory* and the def2-SVP *basis set*. We will use the following `hellowater.inp` ORCA input file:

```
! HF DEF2-SVP

* XYZ 0 1
O   0.0000   0.0000   0.0626
H  -0.7920   0.0000  -0.4973
H   0.7920   0.0000  -0.4973
*
```

Hint

The input usually has the `.inp` extension, but `.txt` is also ok.

You can now run ORCA.

```
orca hellowater.inp > hellowater.out
```

The output is then printed in the file `hellowater.out` in working directory.

```
                                 *****************
                                 * O   R   C   A *
                                 *****************

                                            #,
                                            ###
                                            ####
                                            #####
                                            ######
                                           ########,
                                     ,,################,,,,,
                               ,,#################################,,
                          ,,##########################################,,
                       ,#########################################, ''#####,
                    ,#############################################,,   '####,
                  ,##################################################,,,,####,
                ,###########''''           ''''###############################
              ,#####''   ,,,,##########,,,,          '''####'''          '####
            ,##' ,,,,###########################,,,                        '##
           ' ,,###''''                  '''############,,,
         ,,##''                                '''############,,,,        ,,,,,,###
↪''
      ,#''                                            '''#######################'''
     '                                                          ''''####''''
             ,#######,   #######,   ,#######,      ##
            ,#'     '#,  ##    ##  ,#'     '#,    #''#        ,####,    ,#,
            ##       ##  ##   ,#'  ##            #'  '#       #'        ,# #
            ##       ##  #######   ##           ,######,      #####,      #
            '#,     ,#'  ##    ##  '#,     ,#' ,#      #,     #,   #      #
             '#######'   ##     ##  '#######'  #'      '#     '####' #    #



                ##########################################################
                #                        -***-                           #
                #          Department of theory and spectroscopy         #
                #                                                        #
                #                      Frank Neese                       #
                #                                                        #
                #     Directorship, Architecture, Infrastructure         #
                #                    SHARK, DRIVERS                      #
                #        Core code/Algorithms in most modules            #
                #                                                        #
                #        Max Planck Institute fuer Kohlenforschung       #
                #                Kaiser Wilhelm Platz 1                  #
                #                 D-45470 Muelheim/Ruhr                  #
                #                      Germany                           #
                #                                                        #
                #                  All rights reserved                   #
                #                        -***-                           #
                ##########################################################
```

(continues on next page)

(continued from previous page)

```
[...]
```

You can then open the output file to see the results of your calculation (make sure not to overwrite it while the calculation is running). If the calculation ended normally, a respective message is printed to the end of the output file.

```
                             ****ORCA TERMINATED NORMALLY****
```

The energy of the water molecule can now be found in the output file:

```
-------------------------   --------------------
FINAL SINGLE POINT ENERGY       -75.959340303174
-------------------------   --------------------
```

Congratulations, you successfully ran your first quantum chemical calculation with ORCA!

1.6 General Recommendations

Warning

There are certainly differing opinions among scientists on what method to use when and why. However, we want to give some subjective advise in the context of ORCA based on the developers experience.

ORCA provides a comprehensive and powerful toolkit for quantum chemical calculations. However, the choice of a suitable method for the chemical challenge at hand is often difficult, yet essential to obtain reasonable results. Nevertheless, some general subjective recommendations can be made to navigate through the sheer mass of available method combinations and computational approaches. We further recommend to check for educational review articles that cover the basics of computational chemistry applications.

Educative Reviews on Computational Chemistry

1. Bursch, Markus; Mewes, Jan-Michael; Hansen, Andreas; Grimme, Stefan. Best-Practice DFT Protocols for Basic Molecular Computational Chemistry. *Angew. Chem. Int. Ed.*, **2022**, 61 (42), e202205735. DOI: 10.1002/anie.202205735.

1.6.1 Know Your Methods

Even though the theoretical background of some methods can be complicated, one should know the strengths and limitations of the chosen method. Accordingly, it is recommended to get a fundamental idea of the theoretical basis of any method before using it. Some prominent examples are:

- Typical DFT functionals cannot recover London dispersion effects by theory. Therefore employing a suitable *dispersion correction* is mandatory, especially for large systems.
- Second order Møller-Plesset perturbation theory relies on an energy difference of occupied and unoccupied orbitals in the denominator of its energy expression. Thus, it may break down for systems with small orbital gaps or near-degenerate electronic states (e.g. in many 3d transition metal complexes)
- Single-reference methods like conventional DFT and *single-reference Coupled-Cluster may not be useful for multi-reference cases*. When to go to multireference methods is a more complicated question. Typically, this will be the case if multiplets are desired, pure spin functions for systems with several unpaired electrons, in bond breaking situations or for certain classes of excited states (loosely speaking: whenever there are weakly interacting electrons in the system). However, whenever you decide to do so, please be aware that this require substantial insight into the physics and chemistry of the problem at hand. An uneducated use of *CASSCF* or

MRCI/MRPT method likely yields numbers that are nonsensical and that at tremendous computational cost. Here, there is no substitute for experience (and patience).

💡 **Tip**

Know the strengths and weaknesses of your method of choice. After all, the accuracy of any approximate method is limited and whether its perfectly suited to compute a desired property can only be answered by experimentation and comparison to experimental results or high-level *ab initio* calculations. In this respect, we recommend to make use of various very useful benchmark studies that provide useful data and method recommendations for various properties!

1.6.2 Cost vs. Accuracy

Time and computer resources are not endless. Accordingly, it is important to know if the method of choice is capable of yielding sufficient accuracy at a given timeframe. Further, the fact that you may be able to compute one or the other number a little more accurate doesn't mean that this helps understanding the physics and chemistry of a target system any better. The danger of getting locked into technicalities and miss the desired insight is real! The most relevant aspects that determine the cost-accuracy-ratio are (a) the method used, (b) the basis set used and (c) the cutoffs and tolerances used. Some example cases are:

- Even high-level electronic structure methods will yield bad or at least unreliable results when used with minimal or small split-valence basis sets like STO-3G or 3-21G. However, they may be useful for very large molecules or in screening and exploratory approaches, if purely qualitative results are required. Basis sets of triple-ζ quality and beyond are recommended to obtain reliable results. Some examples are the Karlsruhe def2-TZVPP and the Dunning cc-pVTZ *basis sets*.
- Using very loose optimization thresholds may speed up the optimization but can prevent your system to reach the real minimum at the given potential energy surface. This is specifically the case for flexible molecules with relatively flat potential energy surfaces.
- Insufficiently tight *numerical integration* grids can speed up the calculation but can also limit the accuracy of otherwise highly accurate method/basis set combinations. Even though, the defaults in ORCA are chosen to give reasonable results in most cases, you should increase the numerical integration grid if you are aiming for high accuracy.
- Some approximations harm more than others. For example, the *RI approximation* combined with sufficiently large auxiliary basis sets
  drastically reduces the computation times without at almost no loss in accuracy. Therefore, we generally recommend the usage of RI techniques in line with ORCA's defaults.

💡 **Tip**

Be clear about the level of accuracy you want to achieve. If qualitative results are sufficient, it may make sense to reduce the base set size or thresholds. Nevertheless, there are limits beyond which any results are to be considered questionable.

1.6.3 Creating the Right Model

Reproducing properties of a system at experimental conditions, e.g. at elevated temperatures or in solution, is one of the holy grails of quantum chemistry. However, even the best electronic structure method will not help if the model for your system is incomplete or wrong. Some examples are:

- If a molecule is flexible and has a flat potential energy surface, its molecular properties under experimental conditions may not be described by only one conformer. In such cases, multi-structure approaches should be used to describe properties via Boltzmann-weighted conformer ensembles that may be generated with *GOAT*.
- If your experiment is performed in solution, gas-phase simulations may not give reasonably accurate results. Inclusion of solvation effects either by *implicit solvation* or explicit solvent molecules that can be added with the *SOLVATOR* tool may be necessary.
- Creating a smaller model system may reduce the computation times but bears the risk of missing subtile effects that can cause significant differences compared to the experiment.

Tip

Create your model wisely. Always consider inclusion of environmental effects in your calculations and make sure that your structural model can reproduce the desired property.

1.6.4 Matching Method and Property

A method that performs well for one property does not necessarily have to do so for another. This applies in particular to more approximate methods like *DFT*. Therefore, we recommend to consult benchmark studies to find accurate and robust density functional approximations and method combinations for a given property. Some prominent cases are:

- *(meta-)GGA functionals* like PBE or r^{2}SCAN yield good geometries even when combined with relatively small basis sets. However, for energies at least *hybrid* or even *double-hybrid functionals* are required to reach sufficiently high accuracies.

  Warning

  Some researchers like to adjust the amount of Hartree-Fock exchange according to their needs or what they think is “better” than the standard. This increases the semiempirical character of the calculations and may represent fixes that only work for a given class of compounds and/or properties while worsening the results for others. With this caveat in mind it is one of the things that you are free to try if you like it. However, we do not recommend it since it will deteriorate the comparability of your results with those of other workers the vast majority of which use standard functionals. An alternative to changing the amount of HF exchange could be to simply construct a linear regression for a number of known cases and then use the linear regression.

- Functionals that give mediocre energies like TPSS or TPSSh can be very useful in the context of computing spectroscopic properties like NMR shielding constants or EPR g-Tensors and hyperfine couplings.
- Double-hybrid functionals yield high accuracies for most organic systems but should be avoided for systems with small HOMO-LUMO gaps (e.g. 3d transition metal complexes).

Tip

Check your method of choice. We generally recommend to check for detailed benchmark studies on different properties. If none is available, it may be worth investing the time to benchmark promising methods for your system class and the desired property to compute. If that is not feasible, choosing less empirical higher-level *ab-initio* methods may be the better choice to increase the reliability of the results.

1.6.5 Some Method-specific Recommendations

- DH's The perturbatively corrected functionals (B2PLYP) may also be a very good choice for many problems (at comparable cost to MP2; note that even for large molecules with more than 1000 basis functions the MP2 correction only takes about 10-20% of the time required for the preceding SCF calculation if the RI approximation is invoked. For even larger molecules one has the option of speeding up the MP2 part even further by the DLPNO approximation).

- DLPNO-CCSD(T) Beyond DFT and (SCS-)MP2 there are coupled-cluster methods and their implementation in ORCA is efficient. With the local pair natural orbital methods you can even study molecules of substantial size and with appealing turnaround times.

- **def2-TZVP** is different from the old TZVP. It has been realized that if one invests into an accurate triple-zeta description of the valence region it makes limited sense to only employ a single polarization function. The accuracy is then limited by the polarization set and is not much better than what one gets from SV(P). Hence, def2-TZVP contains a single p-set for hydrogens but is otherwise very similar to the old TZVPP basis set, e.g. it contains 2d1f polarization for main group elements and much more extensive polarization sets for transition metals. The highest polarization function (f for main group) does add substantially to the computational effort. Hence, we often use def2-TZVP without the f polarization function. In order to do that one can use the keyword def2-TZVP(-f). Together with RI or RIJCOSX this is still computationally economic enough for most studies.

- **def2-TZVPP** is a fully consistent triple-zeta basis set that provides excellent accuracy for SCF calculations (HF and DFT) and is still pretty good for correlated calculations. It is a good basis set to provide final single point energies.

- **def2-QZVPP** is a high accuracy basis set for all kinds of calculations. It provides SCF energies near the basis set limit and correlation energies that are also excellent. It is computationally expensive but with RI and RIJCOSX in conjunction with parallelization it can often still be applied for final single-point energy calculations. In conjunction with such large basis sets one should also increase the accuracy of the integration grids in DFT and RIJCOSX — it would be a shame to limit the accuracy of otherwise very accurate calculations by numerical noise due to the grid.

- **Correlation consistent basis sets** provide good correlation energies but poor to very poor SCF energies. For the same size, the ano-pVDZ basis sets are much more accurate but are also computationally more expensive. Except for systematic basis set extrapolation we see little reason to use the cc bases.

- **Pople basis sets** are somewhat old fashioned and also much less consistent across the periodic table than the basis from the Karlsruhe group. Hence, we generally prefer the latter.

- For **scalar relativistic calculations** (X2C,ZORA and DKH) we strongly recommend to use *basis sets that are recontracted for the respective relativistic correction*.

- **Effective core potentials (ECPs)** lead to some savings (but not necessarily spectacular ones) compared to all-electron relativistic calculations. For accurate results, small core *ECPs* should be used. They are generally available for the def2 Karlsruhe type basis sets for elements past krypton. In general we prefer Stuttgart–Dresden ECPs over LANL ones. For the first transition row, the choices are more meager. Here Karlsruhe basis sets do not exist in conjunction with ECPs and you are bound to either SDD or LANL of which we recommend the former. Geometries and energies are usually good from ECPs, but for property calculations we strongly recommend to switch to all electron scalar relativistic calculations using ZORA (magnetic properties) or DKH (electric properties).

- You can take advantage of a built-in basis set (printed using `!PrintBasis` or `orca_exportbasis`) and then modify it by uncontracting primitives, adding steeper functions etc. (fully uncontracted bases are generated via `uncontract` in `%basis`) Alternatively, some basis sets exist that are of at least double-zeta quality in the core region including the DZP and Dunning basis sets. For higher accuracy you may want to consider the `aug-` series of basis sets. See section *Basis Sets* for more about basis set input.

- Likewise, if you are doing calculations on anions in the gas phase it is advisable to include diffuse functions in the basis set. Having these diffuse functions, however, makes things much more difficult as the locality of the basis set is significantly reduced. If these functions are included it is advisable to choose a small value for `Thresh` (10^{-12} or lower). This is automatically done if the smallest eigenvalue of the overlap matrix is below DiffSThresh (which is 1e-6 by default). Also, diffuse functions tend to introduce basis set linear dependency issues, which can be solved by setting `Sthresh` to a larger value than the default 10^{-7} (see

Section *Linear Dependence*). Any value of `Sthresh` beyond 1e-6 has to be used carefully, specially if one is running geometry optimizations, were different basis might be cut off during different geometry steps, or when comparing different conformers since there could be some discontinuity on the final basis set.

- The **integration grids** used in DFT should be viewed together with the basis set. If large basis set calculations are converged to high accuracy it is advisable to also use large DFT integration grids (like `! DEFGRID3`). For "unlimited" accuracy (i.e. benchmark calculations) it is probably best to use product grids (`Grid=0`) with a large value for `IntAcc` (perhaps around 6.0). The default grids have been chosen such that they provide adequate accuracy at the lowest possible computational cost, but for all-electron calculations on heavy elements in conjunction with scalar relativistic Hamiltonians you should examine the grid dependency very carefully and adjust these parameters accordingly to minimize errors. You should be aware that for large molecules the exchange-correlation integration is usually *not* the dominating factor (not even in combination with RI-J).
- Similarly important is the value of `Thresh` that will largely determine the tunaround time for direct SCF calculations. It may be possible to go to values of 10^{-6}–10^{-8} which will result in large speed-ups. However, the error in the final energy may then be 3 orders of magnitude larger than the cutoff or, sometimes, your calculation will fail to converge, due to the limited integral accuracy. In general it will not be possible to converge a direct SCF calculation to better than `Thresh` (the program will also not allow this). For higher accuracy values of maybe 10^{-10}–10^{-12} may be used with larger molecules requiring smaller cutoffs. In cases where the SCF is almost converged but then fails to finally converge (which is very annoying) decreasing `Thresh` and switch to `TRAH` SCF is recommended. In general, `TCut` should be around `0.01` ×`Thresh` in order to be on the safe side.
- DFT calculations have many good features and in many cases they produce reliable results. In particular if you study organic molecules it is nevertheless a good idea to check on your DFT results using MP2. MP2 in the form of RI-MP2 is usually affordable and produces reliable results (in particular for weaker interactions where DFT is less accurate). In case of a large mismatch between the MP2 and DFT results the alarm rings — in many such cases MP2 is the better choice, but in others (e.g. for redox processes or transition metal systems) it is not. Remember that SCS-MP2 (RI-SCS-MP2) and double hybrid functionals will usually produce more accurate results than MP2 itself.
- Coupled-cluster calculations become more and more feasible and should be used whenever possible. The DLPNO-CCSD and DLPNO-CCSD(T) calculations are available for single-point calculations and provide accurate results. However, a coupled-cluster study does require careful study of basis set effects because convergence to the basis set limit is very slow. The established basis set extrapolation schemes may be very helpful here. For open-shell molecules and in particular for transition metals one cannot be careful enough with the reference. You have to carefully check that the Hartree-Fock calculation converged to the desired state in order to get coupled-cluster results that are meaningful. Orbital optimized MP2, CASSCF or DFT orbitals may help but we have often encountered convergence difficulties in the coupled-cluster equations with such choices.
- Generally speaking, CEPA is often better than CCSD and approaches the quality of CCSD(T). It is, however, also a little less robust than CC methods because of the less rigorous treatment of the single excitations in relation to electronic relaxation.

1.7 Troubleshooting Problems in ORCA

When starting out with ORCA, trying out features for the first time, or when delving deeper into computational studies, one may run into issues that need troubleshooting. In this section, we summarize typical sources of errors and the error messages encountered to help with troubleshooting common problems.

Note

This section is to be a growing library of examples of common issues. If you are not able to find a solution to your problem here, please post in the ORCA Forum with a general description of your issue and—if possible—the input and/or output files that showcase the issue. If a type of problem comes up often enough, it will be included in this section for future releases!

1.7.1 ORCA Does Not Run...

If ORCA does not seem to be running at all, make sure that the path variables are set correctly (refer to the *Installation* section) and that there are no other programs on your system that interfere with the use of ORCA. One example is if there is another program named ORCA. To check if this is the case, check that `which orca` or `where orca` (or an equivalent thereof) outputs the correct ORCA executable. If it does not, try using the full path to the ORCA executable when trying to run a job. When in doubt, turn to the section on *Running ORCA*.

If ORCA runs for you in serial, but not in parallel, first make sure that you are not running the main ORCA executable with mpirun. The main ORCA driver needs to be run in serial and will call the different modules in parallel if requested in the input file. If you try to run using mpirun, you will typically get the output

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!                        FATAL ERROR ENCOUNTERED                !!!
!!!                        -----------------------                !!!
!!!                          I/O OPERATION FAILED                 !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

and you may also have an error from prterun/mpirun. If any output is generated, this may be duplicated multiple times.

To run ORCA in parallel, you must also provide the complete path to the ORCA executable. Simply using `orca` will result in the error:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!     ERROR (ORCA_MAIN): For parallel runs                      !!!
!!!     ORCA has to be called with full pathname                  !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

See *Hints on the Use of Parallel ORCA* for more tips.

1.7.2 Where is my ORCA Output?

If you ran ORCA, see that some files were generated, but cannot find the output file itself, you are likely running ORCA without redirecting the output to a file. See *Running ORCA* for instructions.

If, however, you are missing other files that should be generated by ORCA and you are using a submission script, make sure that the script is copying all relevant files from where the job is actually being run.

1.7.3 Sudden Termination of ORCA

The general advice when ORCA terminates unexpectedly is to always look at the last few lines of output—ORCA typically tells you what is going on! For example,

```
            !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                                 INPUT ERROR
            UNRECOGNIZED OR DUPLICATED KEYWORD(S) IN SIMPLE INPUT LINE
                       PAL12
            !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[file orca_main/main_input_keywordline.cpp, line 13941]:
```

tells us that the keyword `PAL12` is not recognized. This is because there are only simple keywords for the more common number of processors. At this point, referencing the manual (searching for "parallel" or "processors", for example) tells us that only "PAL2 to PAL8 and PAL16, PAL32, PAL64" are recognized simple inputs. That same part of the manual shows how to specify any number of processors using the `nprocs` keyword in the `%pal` input block.

However, there are instances where no clear error message is printed. This may look something like this:

```
ORCA finished by error termination in <ORCA MODULE>
Calling Command: mpirun -np 8 /path/to/orca/module myjob.moduleinp.tmp myjob
[file orca_tools/qcmsg.cpp, line 394]:
.... aborting the run".
```

At this point, looking at the warnings issued by ORCA at startup can be very helpful. This is located in the output file section titled `WARNINGS`—right after the list of ORCA contributors and before the copy of the input file:

```
===========================================================================
                                   WARNINGS
                  Please study these warnings very carefully!
===========================================================================
WARNING: Geometry Optimization
  ===> : Switching off AutoStart
         For restart on a previous wavefunction, please use MOREAD
```

This is especially helpful if ORCA found something odd, but decided not to terminate and instead assumed that this was intended by the user.

Other warnings may be issued after the startup warnings. Search the output for other instances of `WARNING` to see what ORCA flagged as possible issues during the calculation. This could be—but is not limited to—non-converging (CAS)SCF, CPSCF, or CC iterations, linear dependencies in the basis set, and memory problems. Typical trademarks of poorly converging geometry and/or SCF runs include, for example:

```
                                   *** Resetting DIIS ***
 WARNING: the maximum gradient error descreased on average only by a factor   0.9
          during the last 20 iterations
                      *** Initiating the TRAH-SCF procedure ***
```

or

```
  0    -4604.304089906417    2.255713e+02                    0.400 (TRAH MAcro)  ␣
↪No
WARNING : negative HOMO - LUMO gap : -60.865582
  0   dE    -8.949363e+01    1.445336e+01    -5.5474e+02     0.400 (TRAH MIcro)
```

Next, check to see if the issue is maybe from outside of ORCA. For example, if you exceed time, disk, or memory limits of the queuing system you use to submit ORCA jobs, the system (or sometimes even an administrator) will terminate your job. In this case, there may be no error message or an error message from your queuing system or system shell. The form of error message you see will depend on your operating system and how the queuing system is set up. The queuing system may also gather the errors in a separate file.

1.7.4 Incorrect Input File Syntax

The following subsections explore common errors with input file syntax. It may be good in general to refer to the sections *General Structure of the Input File* and *Input of Coordinates*.

Unknown Simple Input Keyword

As shown above, the typical error message associated with an unrecognized simple input keyword is

```
            !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                                 INPUT ERROR
            UNRECOGNIZED OR DUPLICATED KEYWORD(S) IN SIMPLE INPUT LINE
                       <KEYWORD>
            !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Unknown Input Block Keyword

Another issue with the input file syntax may be an unrecognized keyword in an input block. The error message for this is typically

```
         Unknown identifier in <BLOCK_NAME> block line <LINE_NUMBER>  :
         Last token : <KEYWORD>
```

If you are sure that the keyword exists, check to make sure that you have closed all input blocks with an `end` statement. Additionally, there may be some subblocks in the input blocks that require an `end` statement.

Required Input is Missing

It is best practice that all ORCA input files specify at least three things: (1) a geometry, (2) a method to run, and (3) a basis set. (1) can be set in the `%coords` block or using the simple input syntax. It may also be read from an external file. (2) and (3) can be set using simple input keywords or their respective input blocks. Refer to the sections *General Structure of the Input File*, *Input of Coordinates*, *Basis Sets*, and the part of the manual for the desired method.

If a method or basis set is not specified, there is often no error message and a default is set. In these cases, it is important to check the output carefully! However, a missing geometry and some auxiliary basis sets do not have defaults and will result in an error. Following are examples of the errors encountered from these issues.

Missing geometry:

```
You must have a [COORDS] ... [END] block in your input
```

Missing auxiliary basis (caught in the main check):

```
ERROR: RI-MP2 needs an AuxC basis but none was defined!
```

Missing auxiliary basis (caught by SHARK):

```
Error (TMOIntegrals): Aux/C requested but not available in SHARK
ORCA finished by error termination in <MODULE>
```

If you are unsure of which auxiliary basis sets to use, a convenient functionality is `!AutoAux`. Note here that these automatically generated auxiliary basis sets may lead to linear dependence issues.

Additionally, if an atom `Xx` is not defined in the provided basis set, it will give the error:

```
               *****************************************************************
               ** There are no main   basis functions on atom number   0 (Xx) **
               *****************************************************************
    The basis set was either not assigned or not available for this element -␣
↪Aborting the run
```

This can be fixed by (1) requesting a different basis set in which that element is defined or (2) using the requested basis set for all other elements and assigning a different basis set just for that element. This can be achieved by using `NewGTO` (see *Assigning Basis Functions to an Element*).

Simply requesting `RI` when the chosen method uses exchange integrals will also lead to ORCA aborting, as it does not know how to handle these integrals. The error message gives the user the options which may be specified:

```
Error: RI is on but the HF exchange must be handled somehow
Use one of the keywords
     RIJDX  : treat the HF exchange exactly (equals RIJONX)
     RIJCOSX: treat the HF exchange by chain-of-spheres
     RIJK   : treat Coulomb and exchange both by RI
```

Depending on the method run and what other keywords are specified, other inputs may be required. In most of these cases, ORCA outputs an error that should be self-explanatory. This error may, for example, look like:

```
         Error: Missing keyword in <BLOCK_NAME> block
         Missing keyword : <KEYWORD>
```

Missing `end` Statement

If an `end` statement is missing to close out an input block, a couple errors may occur. Often, one encounters the `Unrecognized symbol` error, where the next symbol is interpreted as being a part of the previous input block. This error would be of the form:

```
         Unrecognized symbol in <BLOCK_NAME> block line <LINE_NUMBER>   :
         Last token : %
```

One may also get a more obvious error message:

```
         'END' was expected at the end of the <BLOCK_NAME> block line <LINE_NUMBER>
↪   :
         Last token : <PREVIOUS_KEYWORD>
```

Missing Symbols

Just as errors occur with not enough `end` statements, it is also the case with too many `end` statements. For example, some inputs like `%base`, `%moinp`, and `%MaxCore` are global variables and not input blocks. They therefore do not expect `end` statements. In this case, the error may look something like this:

```
ERROR: expect a '$', '!', '%', '*' or '[' in the input
       Line <LINE_NUMBER> of <INPUT_FILE> (END)
```

Here, ORCA was not expecting another keyword and was instead looking for a symbol that would begin a new input block, geometry block, etc.

If instead you provide a keyword outside of an input block, forget the `!` at the beginning of a simple input line or the `*` for the geometry block, or write a comment without a `#`, you will get the same error message as above, just with the first keyword it runs into after the missing symbol in place of `END`.

Trying to read in external files typically requires the file to be provided in quotation marks. If these are missing for the `%moinp` keyword, for example, the error message is:

```
Error: Quotes expected after scanning %%moinp
```

An exception to this is that external coordinate files should not be specified using quotation marks in the simple geometry input section (for example via `*xyzfile`).

Unrecognized Symbols

Another issue is if there are extra symbols than what ORCA is expecting. A common example of this is the simple coordinates input, which begins with an asterisk and ends with an asterisk on a new line if you provide the coordinates in the input file. However, reading coordinates from an external file—using the `*xyzfile` syntax, for example—while also providing a second asterisk (on a new line) gives the error:

```
ERROR: found a coordinate definition line (* ctyp charge mult)
   but could not read ctyp, charge and multiplicity
```

Using `%%` to open an input block instead of `%` or having a stray `%` in the input file results in the error:

```
Error: expected an identifier after '%%' in the input
```

However, using `!!` instead of `!` for the simple input keywords gives an unrecognized keyword error as it interprets the second exclamation mark as its own keyword. Other symbols that ORCA is not expecting will give the error:

```
ERROR: expect a '$', '!', '%', '*' or '[' in the input
```

Cannot Open Files

If you need ORCA to read from another file, you will encounter an error if the file specified does not exist. For example, using `*xyzfile 0 1 geom.xyz` will give the error

```
ORCA_ReadXYZFile::Error
There was a problem reading file: geom.xyz
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!                        FATAL ERROR ENCOUNTERED               !!!
!!!                        -----------------------               !!!
!!!                           CANNOT OPEN FILE                   !!!
!!!               Filename: geom.xyz                                   !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

if `geom.xyz` does not exist. This same error may be seen if the file exists, but quotation marks are given around the file name. In that case, the error message will include the quotation marks in the output of the file name.

Additionally, you may encounter this error if the coordinates are specified in the input file via the simple geometry input block, but the keyword used suggests that ORCA should read from an external file. For example, using the `*xyz` coordinate syntax with the keyword `*xyzfile` gives the error:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!                        FATAL ERROR ENCOUNTERED               !!!
!!!                        -----------------------               !!!
!!!                           CANNOT OPEN FILE                   !!!
!!!               Filename:                                    !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Characteristic here is the absence of a filename, as `*xyzfile` was used, but no external file was provided.

Other file types may have different error messages. For example, if a GBW file is read in with `%moinp MOs.gbw`, but `MOs.gbw` does not exist, it will give the error:

```
Cannot open GBW file: MOs.gbw
```

End-of-file (EOF) Issues on Windows Machines

If you are running ORCA on a Windows machine, you may run into issues with your input file if it does not end with a blank line. A common case is that the last line tries to read coordinates from an external file, resulting in an error message claiming that the file is missing. If adding a blank line at the end of the input file does not help, also try adding a blank line at the end of (user-created) external files that ORCA is to read from.

Non-ASCII Characters

You may also get an error if you try to use a non-ASCII character in the input file. In older versions of ORCA, non-ASCII characters were not checked for, which could lead to seemingly valid keywords being flagged as unrecognized. Since ORCA 6.0.1, a helpful error message is printed to aid in finding these characters:

```
ERROR: non-ASCII character(s) found in words on line <LINE_NUMBER>:
<KEYWORD W/ NON-ASCII> Please remove all non-ASCII characters from your input file
```

Especially difficult to spot are non-ASCII characters like non-breaking spaces, en- and em-dashes, and the various types of quotation marks as these look very similar to their ASCII counterparts.

1.7.5 Out of Memory (OOM)

A typical error message that occurs if `%MaxCore` is not given or if insufficient memory is available for the calculation to proceed is `Please increase MaxCore`. This will typically be the case if ORCA tried to estimate the memory needed and found that `MaxCore` is not sufficient.

> **Note**
>
> `MaxCore` is given in MB and is the amount of maximum memory to allocate **PER PROCESSING CORE**! ORCA may also overshoot `MaxCore` during the calculation and is not necessarily set for all methods. As such, it is generally advised to set `MaxCore` such that `MaxCore * nprocs <= 0.75(Available Memory).`

For some methods, ORCA tries to estimate the memory needed early on during the calculation. If this estimation is larger than `MaxCore`, but smaller than 2*`MaxCore`, ORCA will issue a warning and proceed. However, if the estimation is larger than 2*`MaxCore`, ORCA will abort. You will then have to increase `MaxCore` in the input file and run the job again.

In some cases, this rough estimation of the memory required is not enough and ORCA realizes during the calculation that it does not have enough memory to proceed. Such an error may look like this:

```
[file orca_tools/qcmem.cpp, line 949, Process 7]:  OUT OF MEMORY ERROR!

--------------------------------------------------------------------------
Primary job  terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
--------------------------------------------------------------------------
--------------------------------------------------------------------------
mpirun noticed that process rank 0 with PID <PID> on node <NODE> exited on signal 11 (Segmentation fault).
--------------------------------------------------------------------------
4 total processes killed (some possibly by mpirun during cleanup)

ORCA finished by error termination in SCF
Calling Command: mpirun -np 8  /path/to/orca_scf_mpi myjob.gbw b myjob
[file orca_tools/qcmsg.cpp, line 458]:
  .... aborting the run
```

Note that the `OUT OF MEMORY ERROR` message may not be in the output, depending on the method run.

1.7.6 Disk Full

Some ORCA calculations require a lot of data to be written to disk. This can either be temporary files used during the calculation and deleted afterwards or files that are stored even after the calculation is finished. If the disk becomes full at some point during the calculation, ORCA will abort. This may happen silently (without a clear error message) or with an error message that typically mentions not being able to write a matrix or set a pointer. For example:

```
Error TMatrixContainers::AddMatrix - Failed to add <INDICES> to <TMP_FILE>
```

or

```
Error TMatrixContainers::AddMatrix - Failed to set FilePointer
```

1.7.7 Basis Set Linear Dependence

If some functions in the basis set used are (nearly) linearly dependent, this may lead to numerical issues in the calculation. This is more often the case for basis sets with diffuse functions and auxiliary basis sets generated by `!AutoAux`. ORCA and SHARK try to catch these issues early. For example, you may see

```
WARNING! Potentially linear dependencies in the auxiliary basis
===>>    At the moment, SHARK is not equipped to deal with this
  ... sorry, have to bail out
```

in the output. However, this is sometimes not caught and could lead to numerical instabilities or even ORCA aborting down the line. For example, one may encounter the issue:

```
Diagonalization failed with error code: -5 !
```

One part of the output to check in this case is the diagonalization of the overlap matrix. Very small eigenvalues could point to linear dependence issues. This output may look something like:

```
Diagonalization of the overlap matrix:
Smallest eigenvalue                        ... 3.064e-08
Time for diagonalization                   ... 3.223 sec
Threshold for overlap eigenvalues          ... 1.000e-07
Number of eigenvalues below threshold      ... 3
Smallest eigenvalue above threshold        ... 1.002e-07
```

There are options to help mitigate linear dependence issues, which are explained in more detail in the section on *Linear Dependence*.

1.7.8 Symmetry of Molecule Not as Expected

If you run ORCA with symmetry (via `!UseSym`), be sure to check that the `Auto-detected point group` matches what would be expected. The input geometry may not be perfectly symmetric due to numerical noise. In that case, loosening the threshold for point group identification (for example, with `%sym SymThresh 0.01 end`) should lead to the expected point group and ORCA will use a symmetry-perfected geometry for the calculation. For more information, see the section *ORCA and Symmetry*.

1.7.9 Using Old ORCA Inputs

If you try using an old ORCA input file with a newer version of ORCA, you may find that it does not run. Often when this is the case, there will be an error message issued during the startup of the calculation. This message should help you troubleshoot the issue.

Here, the issue typically arises due to keywords and defaults which may have changed from one version to another. In some cases, a keyword may have even been deprecated. When updating to a new version of ORCA, it is often helpful to refer to the *Detailed change log*, as many of these changes are noted.

If you are trying to read in a GBW file from another version of ORCA, you may find that you run into an error such as:

```
WARNING!!!!!!!
Your GBWFile is either corrupt or from a different ORCA version!
Please be VERY careful with your calculation results!!!

Data_size, sizeof( TOrcaInfo ): 974880, 980832

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!                        FATAL ERROR ENCOUNTERED               !!!
!!!                        -----------------------               !!!
```

(continues on next page)

(continued from previous page)

```
!!!                          I/O OPERATION FAILED                !!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

This is especially common when using `!MORead` or when using ORCA utility programs, such as `orca_plot`. In the case of reading in starting orbitals for a subsequent calculation, the additional keyword `!Rescue` may help. Note that with this, fewer checks are performed, so it is important to make sure that the GBW file represents a system that is compatible with the current input file.

For the case of plotting orbitals and certain densities with `orca_plot`, you can either use the same ORCA version as the one that produced the GBW file or you can run a calculation with the newer ORCA version by using `!Rescue MORead NoIter` in the input file (and reading in the GBW file via the `%moinp` keyword). This calculation should not take long and the new GBW file produced should be readable by `orca_plot`. Note that in this case, all other settings should be specified as they were for the calculation that produced the older GBW file.

If you are unsure about an input, please first consult the relevant parts of this ORCA Manual. It is provided by the ORCA developers and should contain all information about the functionality implemented in this version of ORCA. It is also important to reference the ORCA Manual for the version of ORCA that you are using. If you are viewing this manual online and are not using the most recent release version of ORCA, please update your ORCA version or find the ORCA Manual for the version you are using. If you are still unable to get an input file to work, the ORCA Forum is a great place to turn to for getting answers to your questions!

1.7.10 Reproducing Results of Other Program Packages or Older ORCA Versions

Reproducibility of results is fundamental in science, so it is good practice to compare results when switching from one program to another or from one ORCA version to another. If you are trying to do this and observe deviations that are larger than expected, here are some hints and tips.

Newer versions of ORCA might include changes to the default settings, which may affect the numerics. For example, ORCA 6 has improvements to the numerical integration grids. In most cases, these changes should lead to better results and are most likely the reason for differences between the latest ORCA release and previous versions. It is usually possible to manually set these values to their old defaults for comparison purposes. For such an inspection, compare the information in the output to see which values have changed. Reference the relevant sections of this manual to determine how to change these values.

In trying to reproduce results from another program in ORCA (or vice versa), it is important to check that the methods being used are in fact equivalent and that both programs compute the value(s) in question in the same manner. This may mean using some settings that you would not otherwise use so that you can make sure that the two should be comparable. For example, ORCA uses pure d- and f-functions (i.e. 5 d-functions and 7 f-functions instead of the Cartesian 6 d-functions and 10 f-functions) for all basis sets, so this should be taken into account when comparing to programs that do not.

If you still run into issues with reproducibility, open a post in the ORCA Forum and give enough details on what is being compared so that other users and developers can be of assistance. It is often helpful to include the input(s) and/or output(s).

1.7.11 Orbitals Not Found

If you encounter this error:

```
No orbitals were found in the gbw file
```

it is most likely that you are trying to use `!MORead` to read in a GBW file with the same basename as the current input file. This is done automatically with ORCA's AutoStart feature (on by default), so you should be able to simply remove the `!MORead` and `%moinp` parts of the input file. If you want to save the previous orbitals (for later access) and read them in, copy the GBW file to something with a different basename and then read that file in as usual.

1.7.12 Trouble Restarting Input File

Another AutoStart issue is when you make a significant change to your input file and try to rerun it without deleting the old ORCA outputs. This often happens if there is a significant mismatch in the geometry—like when atoms have been reordered or removed. In this case, the error encountered is:

```
Error: Input geometry does not match current geometry
```

Typically, the basis set can be changed without issue, though, as ORCA will perform a projection of the old orbitals onto the new basis set if needed.

If there is an issue with restarting a file, you can always try to run the job from scratch by adding `!NoAutoStart` to the input file or by deleting all other files with the same basename as the input file before rerunning.

1.7.13 SCF Convergence Problems

A typical problem that occurs for systems with complex electronic structures like transition metal complexes or systems with many orbitals in a narrow energy range (such as metal clusters) is a poorly-converging SCF. While ORCA offers keywords like `!SlowConv`, this might not be the best option. Specifically, `!SlowConv` may converge to a local minimum solution that is closer to that of the initial guess. We have compiled a list of tips and tricks in the section *Converging SCF Calculations*.

Example: Converging DFT for Open-Shell Transition Metals

Since DFT is frequently applied to open-shell transition metals, we show here a (more or less trivial) example of a Cu(II) complex treated with DFT:

```
*xyz -2 2
  Cu   0      0      0
  Cl   2.25   0      0
  Cl  -2.25   0      0
  Cl   0      2.25   0
  Cl   0     -2.25   0
*

%compound
NewStep
  !BP86 SV SlowConv
StepEnd

ReadMOs(1);
NewStep
  ! B3LYP NoRI TZVP TightSCF
  %scf GuessMode CMatrix end
StepEnd
EndRun
```

Although it would not have been necessary for this particular example, this shows a possible approach to converging such calculations. We first perform a less accurate (but quicker) job using the RI approximation, a GGA functional, and a small basis set without polarization functions. Note that a larger damping factor has been used in order to guide the calculation (`!SlowConv`).

The second job takes the orbitals of the first as input and performs a more accurate hybrid DFT calculation. A subtle point in this calculation on a dianion in the gas phase is the input specification `GuessMode CMatrix`, which requests the use of the corresponding orbital projection. This matches the orbitals of the smaller basis set calculation to those of the larger basis set. This is always required when the orbital energies of the small basis set calculation are positive, as will be the case for anions.

1.7.14 Troublesome Geometry Optimizations

While some issues during optimization jobs give errors that have been covered above, there are problems that may arise which do not produce error messages or terminate ORCA. In some cases, you may get a normal termination, but the results you obtain are not valid. One must often use chemical intuition to determine if the geometry optimization yields reasonable structures and frequencies.

Imaginary Frequencies after Optimization

If a geometry optimization terminates normally, the algorithm has decided that it is sufficiently close to a stationary point on the potential energy surface. To test whether this stationary point is a minimum, maximum, or transition state or to check how close the structure is to the stationary point, a frequency calculation must be run. This might yield imaginary frequencies (which arise due to negative force constants as the frequencies are proportional to the square root of them), which indicates that we are not at a minimum.

```
-----------------------
VIBRATIONAL FREQUENCIES
-----------------------

Scaling factor for frequencies =  1.000000000  (already applied!)

   0:          0.00 cm**-1
   1:          0.00 cm**-1
   2:          0.00 cm**-1
   3:          0.00 cm**-1
   4:          0.00 cm**-1
   5:          0.00 cm**-1
   6:         -5.66 cm**-1 ***imaginary mode***
   7:         14.88 cm**-1
   8:         21.54 cm**-1
```

There are several approaches to deal with this issue. Typically, using tighter optimization criteria, tighter DFT grids, and/or tighter SCF convergence criteria helps, especially for small imaginary frequencies. Another approach can be to identify the normal modes associated with imaginary frequencies, distort the molecule along these vectors, and re-optimize the structure. Note that the compound script *Iterative optimization* does this exact process in an automated fashion.

Getting Unwanted Local Minima

If the optimized geometry you are getting is not the one you were expecting, you may be running into a nearby local minimum on the potential energy surface. This is often encountered when optimizing a molecule that has many rotamers.

First, check that the input geometry you have provided is reasonably close to the conformation you are aiming for. It may help to preoptimize the structure with an inexpensive method. If aiming for the global minimum, one may make use of *GOAT* to preoptimize the structure to a reasonable guess for the global minimum.

If there is symmetry breaking during the optimization, but you would like to enforce the original point group, look at the options in *Geometry optimizations using symmetry*.

Issues with Optimization Convergence

There are a few options you have when the calculation simply does not meet the optimization convergence criteria by the set number of maximum iterations. Inspect the last single point geometry along the optimization trajectory. If it seems intuitively close to a stationary point, try to rerun the optimization from this geometry. This is equivalent to rerunning the original input file, but increasing the number of iterations in the optimization.

Another approach is to first optimize with a less expensive method and/or smaller basis set, then use that geometry for the optimization at the level of theory wanted. The optimization can also first be done with looser convergence thresholds before tightening them as needed.

If none of these tips help, one may identify parts of the structure that are giving issues by analyzing the trajectory of the optimization. The optimization may have gotten stuck at the end in a region where the potential energy surface is nearly flat or has many shallow minima—such as for structures with many rotamers or for systems with multiple molecules. In that case, some reasonable geometry constraints may be needed. It may also be wise to change the trust radius or the maximum step in internal coordinates. In some cases, cartesian optimization via `!COPT` may even be needed.

For more details, see the section *Geometry Optimizations*, especially for the *Geometry Optimization Keywords*.

1.7.15 "Strange" Results for Energies and Properties

If results for energies or properties seem off, first make sure that the geometry, spin multiplicity, and charge used are reasonable for the system at hand. Make sure that the input is a good approximation of the real system you are trying to model. To this end, consider if a solvent model is appropriate. This is especially important for charged systems. For mean-field methods, also consider using a dispersion correction if not already included. If heavy elements are included in the system, scalar relativistic methods may be appropriate, and possibly picture change and finite nucleus corrections.

Next, make sure that the SCF has converged well. For properties and post-HF energies and properties, the SCF should be converged tightly enough that the subsequent calculations do not suffer from the numerical noise of the underlying SCF results. A loose convergence threshold will ultimately affect the properties that follow from the results of the SCF.

For unrestricted methods, it is advisable to look at the calculated spin contamination in the output. If this is significant, results for the given system with that method may be unreliable. One should be careful in the analysis of the output file in this case. In some cases, the underlying representation of the system is flawed or incomplete, but in other cases, you may consider turning to methods that give pure spin state solutions, such as ROHF, ROKS, or multireference methods.

In general when using single reference methods, one should question if the system may exhibit multireference character. In such a regime, results from single reference methods may not be reliable. To help determine this, there are a couple diagnosistics that can be computed. When running a coupled cluster calculation, the T_1 diagnostic is commonly used to estimate the importance of static correlation in the description of the system. The computed value is printed when running a coupled cluster calculation and if greater than 0.02, the system **may** have multireference character. More reliable is the newer S-diagnostic, which can be requested by `%mdci DoSDiagnostic true end`. See the *Coupled Cluster S-Diagnostic* section for more information.

When running a multireference calculation, it is important to make sure that the final active space is reasonable and includes all important orbitals that contribute to the static correlation. This may require changes to `nel` and `norb`. Good starting orbitals are important here (see the section on *Choice of Inital Guess*). Starting orbitals may also be generated from other calculations—for example using MP2 natural orbitals or QROs from UKS—which often represent a more physically relevant starting guess. *AVAS* is helpful for building the active space from these starting guesses, provided that you can intuitively determine the AOs that contribute to the MOs you want to include in the active space. Note that although an active space may seem reasonable, there may be convergence issues in CASSCF or in the post-CASSCF calculations when some of the active orbitals are very highly occupied/unoccupied. Finally, one should also consider if a state-specific or state-averaged solution is most appropriate for the system. Keywords which may need to be changed here are `nroots`, `mult`, `weights`, and `irrep`.

Finally, it should be checked that the methods used are appropriate for the system at hand. For example, MP2—and CCSD(T) to a lesser extent—has issues with orbital degeneracies. For properties, linear response and sum-over-states results are only reliable for systems without low-lying excited states. If the system is outside of the regime where the requested method is reliable, another method should be considered.

If everything seems reasonable, but there still seems to be an issue, post the case in the ORCA Forum. It could be that there is a technical reason for the values to be how they are or there may be a bug that should be fixed.