Kylin: Ab-Initio DMRG Package
User Guide

Version 1.3.0

The Group of Prof. Haibo Ma

January 26, 2025

Contents

1 Introduction

1.1 Overview

Kylin is an *ab-initio* quantum chemistry software package for evaluating electronic structures of molecular systems with high precision and efficiency. The *ab-initio* methods are fundamental and powerful tools in quantum chemistry researches. Because their dependence on artificial assumptions and parameter fittings is less than many other popular methods such as the density functional theory (DFT), the *ab-initio* methods are usually more reliable, but more expensive at the same time, in the study of electronic structures of complex systems such as radicals, transition metal compounds, and large conjugated molecules. Energies, properties, and many interesting chemical reactions of these systems are strongly influenced by their near-degenerate electronic configurations, as well as the fast movements and interactions of electrons. Kylin can be used for precisely measuring electronic correlations in these systems, and predicting energies, properties and reactions with extensive treatment.

The heart of Kylin is an efficient implementation of the density matrix renormalisation group (DMRG) method, which is an accurate approximation of the classical full configuration interaction (Full-CI) method but can be applied on large active spaces with more than 100 active orbitals. Besides, Kylin also features extensive capabilities in pre- and post-DMRG treatments, including Hartree-Fock self-consistent field (HF-SCF), complete active space SCF (CASSCF), single-reference and multi-reference CI, single-reference and multi-reference perturbation theories (PT), as well as DMRG wave function analysis and geometry optimization. Therefore, Kylin can also be used as an independent quantum chemical software, with no necessity to involve additional packages.

1.2 Features

The developers of Kylin aim to provide an efficient implementation of the *ab-initio* quantum chemistry DMRG method. Beyond that, the developers also want to create a complete package for general *ab-initio* quantum chemistry researches, which includes all necessary procedures in a typical wave-function-based calculation. The following methods are available (also Shown in Fig. 1):

- Hartree-Fock SCF, including the restricted approach for closed-shell systems and the unrestricted one for open-shell systems;

- Second-order Møller-Plesset perturbation theory;

- Full-CI, CASCI and CISD with the graphical unitary group approach (GUGA);

- DMRG based on the matrix product states (MPS);
- CASSCF and DMRG-SCF, based on the second-order Werner-Meyer-Knowles method;
- Multi-reference CI, with CASSCF/DMRG-SCF/sCI reference wave function;
- Multi-reference Epstein-Nesbet perturbation theory, with CASSCF/DMRG-SCF/sCI reference wave function.
- geometry optimization in Hartree-Fock, CASSCF and DMRG-SCF calculations.

More methods and functions will be introduced in further releases.

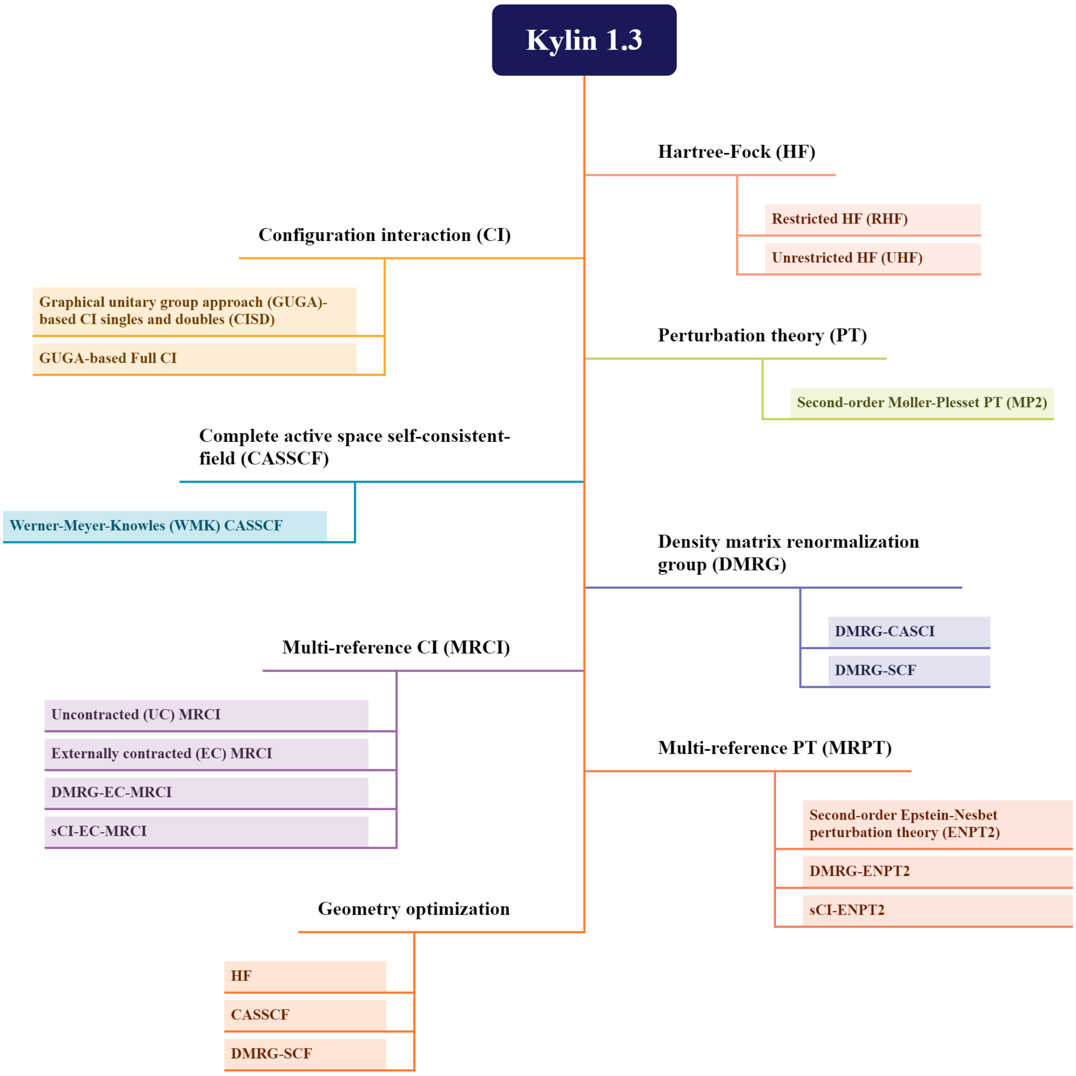

Figure 1: Available methods in Kylin 1.3.

1.3 Copyright and Citation

The Kylin package is developed by the research group of Prof. Dr. Haibo Ma at Nanjing University. The copyright has been certified by the National Copyright Administration of the People's Republic of China.

Any publication resulting from use of this program must acknowledge the following article:

> Zhaoxuan Xie, Yinxuan Song, Fangwen Peng, Jianhao Li, Yifan Cheng, Lingzhi Zhang, Yingjin Ma, Yingqi Tian, Zhen Luo, Haibo Ma, *Kylin 1.0: An Ab-Initio Density Matrix Renormalization Group Quantum Chemistry Program*. **Journal of Computational Chemistry**. 2023, 44(13), 1316. `https://doi.org/10.1002/jcc.27085`

2 Release Notes

2.1 Version 1.3

- A new Hartree-Fock module `[scf]`:
  - Advanced DIIS techniques.
  - Several initial guess options.
  - Supporting direct SCF methods.
  - Supporting scalar relativistic corrections (via sfx2c1e).
- `[dmrg]` module:
  - Less memory cost via storing some temporary tensors to disk (keyword `store_ENV`).
  - More convenient to sample important configurations from DMRG wavefunction (see Section 9.2).
- `[dmrgscf]` module:
  - Fixed a bug that makes convergence difficult.
- Deprecated modules:
  - The old Hartree-Fock module [hf].
  - The entanglement driven genetic algorithm (EDGA) module `[edga]`.

3 Installation and Running

3.1 Installation

The latest version of Kylin is distributed as a pre-compiled binary. After your download request is approved, a download link will be sent to your email address. See the official website[1] for more information.

It is easy to install Kylin on your computer: firstly you need to download the zip file, then extract this file and set the required environment variables. After that, the Kylin program is ready for work. For example, you can extract the zipped file as

```
unzip kylin-latest.zip
```

Then a new directory `kylin` appears, which containing two sub-directories `bin` and `share`. Note the two sub-directories must be always placed in the same directory. You can put the `kylin` directory wherever you want (e.g. `Applications/kylin` in your `home` directory, as shown in the following examples).

The binary is compiled on an intel platform running Ubuntu 20.04 and the AVX2 instructions must be enabled.

3.2 Environment

The only environment required by Kylin is `KYLIN`, which is the absolute path of Kylin directory. The environment can be automatically initialized by adding the following two lines into your `.bashrc` file.

```
export KYLIN=$HOME/Applications/kylin
export PATH=$KYLIN/bin:$PATH
```

Note that `$HOME/Applications/kylin` is the directory of Kylin. Besides, to run Kylin parallely, the `OMP_NUM_THREADS` must be declared. The value of this environment variable should be the number of CPU cores.

```
export OMP_NUM_THREADS=24
```

3.3 Running

Once the environment variable `KYLIN` is successfully loaded, and the `kylin` command could be found in your `PATH`, the Kylin program can be used as

---

[1] `https://kylin-qc.com`

```
kylin INPUT_FILE
```

Then all output will be printed on the screen, which can be redirected to a file, as

```
kylin INPUT_FILE > OUTPUT_FILE
```

Since all temporary files, such as the molecular orbital integrals, will be stored in the `tmp` folder in the current working directory, we recommend to place your working directory on a disk with fast I/O, for example, on a solid state drive.

4 Input and Output Files

4.1 Input File

The syntax of Kylin's input file is simple. Currently there are only 4 kinds of entries in the input file, which are

1. module, surrounded by a pair of square brackets "`[`" and "`]`";
2. keyword and its value(s), which should be separated by at least one separator (space or "="), for example, `keyword = value`. Input keywords do not have case or underscore sensitivity, which means that `InputKeyword` and `input_keyword` are regarded as exactly the same.
3. comment, which is similar to python. The single-line comment starts with hashtag symbol (#), while the multi-line comment is surrounded by triple quotes (”’).
4. line break. If there are too many values which can not be put in one line, the backslash character (\) can be put at the end of the line.

Note that the calculation of a certain method (eg. `METHOD`) can only be initialized with the `[METHOD]` label, and all the necessary keywords used in the `METHOD` must be put under its label.

Here is an example input file.

```
[info]
    # coordination file
    coord = example.xyz

    # multiple basis sets
    basis = 6-31g C cc-pvtz N cc-pvdz \
                  O S cc-pvqz

    pg = c1 # not use point group symmetry

'''
multi-line comment
'''

[integral]
```

```
[scf]
```

The case of the module names and keywords are ignored. However, the case of some values, such as the value of the `coord` in the above example, must be specified clearly, because it denotes the name of another file, and file names in Linux are case-sensitive.

The detailed keywords in each module are listed in the following sections.

4.2 Output File(s)

When running a calculation, a temporary folder `tmp` will be created in the working directory, which stores atomic orbital integrals and some other scratch files. Besides, some useful output will also be stored in this folder, including the molecular orbital file and the related `molden` file.

> To visualize molecular orbitals with `molden` file, we recommend to use the Multiwfn software.

5 Geometry and Basis Set

As we can see in the input example in the last section, a calculation should always start with the `[info]` module, which defines the basic information of the system, including the geometry, basis set and point group of the molecule. Kylin can also read such information from a MOLDEN file, but point group symmetry will not be used when using a MOLDEN file as input.

Keyword	**Description**
Basis	(case-insensitive string) Basis sets for the calculation
Coord	(case-sensitive string) The path to an XYZ file.
InputType	(case-insensitive string) default: `normal` The type of the input file. This keyword should be set as `normal` or `molden`.
MoldenFile	(case-sensitive string) The path to an Molden file. This keyword is incompatible with keywords "Coord" "Basis" "RotCoord" and "PG", because only C1 molden are allowed and all information are already in molden file.
PG	(case-insensitive string) The symmetry of the molecular system. When set as `C1`, point group symmetry will not be used.
RotCoord	(bool) default: `true` Whether to rotate molecular coordinates when using point group symmetry. If the coordinates has been rotated, they can be used directly to do calculations under point group symmetry by setting it to `false`.
SymThresh	(double) default: `1e-5`

Keyword	Description
	Threshold for symmetry point group symmetry recognition. It can be loosen to 1e-2 at most, or the accuracy of Single Point Calculation maybe affected.

Table 1: Keywords in Info.

5.1 Geometry

Molecular geometry in Kylin is set by the keyword `coord`, as

```
coord = XYZ_FILE
```

where `XYZ_FILE` is the relative path to a standard XYZ file defining the cartesian coordinates (default in Ångstrom units). Usually we put the XYZ file in the same directory of our input file, in this case `XYZ_FILE` is the name of this XYZ file. For example, the following is a XYZ file defining the structure of a naphthalene molecule.

```
 18
unit or comment
 C                0.00000000  -0.70070000   0.00000000
 C               -1.21364800  -1.40140000   0.00000000
 C                1.21364800  -1.40140000   0.00000000
 C               -2.42729600  -0.70070000   0.00000000
 H               -1.21364800  -2.47140000   0.00000000
 C               -0.00000000   0.70070000   0.00000000
 C                2.42729600  -0.70070000   0.00000000
 H                1.21364800  -2.47140000   0.00000000
 C               -2.42729600   0.70070000   0.00000000
 H               -3.35394318  -1.23570000   0.00000000
 C               -1.21364800   1.40140000   0.00000000
 C                1.21364800   1.40140000   0.00000000
 C                2.42729600   0.70070000   0.00000000
 H                3.35394318  -1.23570000   0.00000000
 H               -3.35394318   1.23570000   0.00000000
```

```
H                -1.21364800   2.47140000   0.00000000
H                 1.21364800   2.47140000   0.00000000
H                 3.35394318   1.23570000   0.00000000
```

In the first line the number of atoms in this molecule is declared. If atomic unit is used, "`a.u.`" or "`bohr`" should be put at the beginning of second line. Other strings in the second line are recognized as comment and ignored.

5.2 Basis Set

There are four ways to specify basis set with `basis` keyword.

1. single basis set

```
    basis = 6-31g
```

2. multiple basis set

```
    basis = 6-31g C O cc-pvtz N cc-pvqz
```

   (C and O with cc-pVTZ, N with cc-pVQZ, other atoms with 6-31G)

3. multiple basis set

```
    basis = C O cc-pvtz N cc-pvqz
```

   (C and O with cc-pVTZ, N with cc-pVQZ)

4. multiple basis set

```
    basis = specified
```

   Then the basis sets are specified in coordination file, for example

```
    3

    o      1.8926600      0.6524290      0.0     cc-pvqz
    h      2.8385790      0.3622590      0.0     cc-pvtz
    h      2.0138280      1.6344200      0.0     cc-pvdz
```

The available basis sets are listed in the file `$KYLIN/share/basis/basis_list`. This file actually provides indices of the basis set files in the same directory.

If the basis name contains space character, the space character should be replaced by underscore character in input file. For example, if using "Ahlrichs pVDZ" basis set, the `basis` keyword should be set as

```
basis = Ahlrichs_pVDZ
```

You can add new basis sets into this directory if necessary, by putting a Molcas-style basis set file into this directory and adding its name to the `basis_list` file. Please visit the Basis Set Exchange[2] website for more basis sets.

> The ECP basis sets are not supported currently.

> Only spherical basis functions are used in Kylin currently.

5.3 Point Group Symmetry

The point group symmetry will be automatically determined, unless the keyword `pg` is set to `C1`. The supported point groups and the numbering of their irreducible representations (irreps) are listed in Table 2. Note that currently point group symmetries are not supported in some modules.

---

[2] `http://www.basissetexchange.org`

Point group	**irrep**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
C_1	A							
C_i	A_g	A_u						
C_s	A'	A''						
C_2	A	B						
C_{2v}	A_1	B_2	B_1	A_2				
C_{2h} (main C_2 axis is $C_2(z)$)	A_g	A_u	B_u	B_g				
C_{2h} (main C_2 axis is not $C_2(z)$)	A_g	B_u	A_u	B_g				
D_2	A	B_1	B_2	B_3				
D_{2h}	A_g	B_{3u}	B_{2u}	B_{1g}	B_{1u}	B_{2g}	B_{3g}	A_u

Table 2: Supported point groups and the numbering of their irreps.

6 Hartree-Fock SCF

If there is no specific reason, it is recommended to use the `[scf]` module instead of `[hf]` for Hartree-Fock calculations.

The “new” Hartree-Fock module, `[scf]`, in Kylin provides support for both restricted and unrestricted Hartree-Fock methods. Additionally, it includes advanced DIIS techniques (CDIIS, EDIIS, ADIIS and MIXED(ADIIS+CDIIS)), offers an improved Superposition of Atomic Densities (SAD) initial guess for better SCF convergence, and supports Direct-SCF methods as well as scalar relativistic corrections (via sfx2c1e).

For pure Hartree-Fock calculations, Direct-SCF is recommended as it is computationally more efficient and does not require the `[integral]` module. The corresponding molden file is saved as `tmp/hf.molden`, which can be visualized with the Multiwfn software.However, if you plan to perform post-SCF computations using orbitals obtained from `[scf]`, Direct-SCF should be avoided, and it is advised to include the `[integral]` module. Scalar relativistic corrections is compatible with post-Hartree-Fock methods.

The keywords in the `[scf]` module are listed in Table 3.

Keyword	**Description**
2S	(integer) default: 0 The value of spin multiplicity $2S$.
Charge	(integer) default: 0 Charge of the system.
SCFType	(case-insensitive string) default: `rhf` options: `rhf`, `uhf` This option must be set manually if UHF is require.
AccConv	(case-insensitive string) default: `mixed` options: `cdiis`, `adiis`, `ediis`, `mixed`, `off` Acceleration technique to be used in the self-consistent field procedure.
DIISMaxVec	(integer) default: 12 Size of the DIIS space. This option is valid only if a DIIS-type accelerator is enabled.

Keyword	**Description**
DIISStart	(integer) default: 2 Represents the number of SCF iteration cycles with DIIS enabled. This option is valid only if a DIIS-type accelerator is enabled.
Guess	(case-insensitive string) default: `atom` options: `core`, `minsad`, `atom` Select the type of initial guess.
MaxIter	(integer) default: 100 Maximum number of Hartree-Fock iterations.
ToleranceEnergy	(double) default: 1.0e-10 Energy convergence tolerance of the Hartree-Fock interations.
ToleranceGrad	(double) default: 1.0e-5 Gradient convergence tolerance of the Hartree-Fock interations.
Direct	(bool) default : `false` Whether to use Direct-SCF. If post-Hartree-Fock is required, set it to false.
SFX2C1E	(bool) default : `false` Whether to use scalar relativistic calculations.

Table 3: Keywords in SCF.

7 Configuration Interaction

The configuration interaction (CI) calculations are performed by the graphical unitary group approach (GUGA). [1–6] The full CI (FCI) method are implemented in module `[GUGACI]`, while the CI singles-and-doubles (CISD) method will be implemented in next version. The CISD wavefunction includes HF configuration as well as its single and double excitation configurations, while the FCI wavefunction further includes all other higher excitation configurations. The keywords are listed in Table 4.

Keyword	**Description**
CISD	Perform CISD calculation (will be implemented in next version). Default is `false` and FCI calculation will be performed.
2S	Spin ($2S$), default 0 for singlet.
docc	Number of doubly-occupied orbitals in each irrep.
active	Number of active orbitals in each irrep.
charge	Charge, default 0.
irrep	Irreducible representation of target state(s) (start from 1), default 1.
nstate	Number of states to calculate, default 1.
eigensolver	Algorithm of eigensolver. `lanczos`, `davidson` and `jacdav` for Lanczos, Davidson and Jacobi-Davidson algorithms respectively. Default is `davidson`.
dump	Print active MO integrals to file FCIDUMP and not perform CI calculation, default `false`.
dumpAll	Print all of MO integrals to file FCIDUMP and not perform CI calculation, default `false`.

Table 4: Keywords in GUGACI.

7.1 Start with Integral File (FCIDUMP)

The `[GUGACI]` module in Kylin could be used as a stand-alone program if the molecular orbital integrals are provided. Some extra required parameters are listed in Table 5.

Keyword	**Description**
independent	Perform stand-alone GUGACI calculation, default `false`.
integral	FCIDUMP file name. In this file, the irreps and orbital indices both start from 1.
nelec	Number of total electrons. If FCIDUMP file does not contain doubly-occupied orbitals, `nelec` = number of active electrons and `docc` should be 0 for each irrep.
pg	Point group symmetry of the system, default `c1`.

Table 5: Extra required keywords for stand-alone DMRG.

Here is an example of N_2 with CAS(6e, 6o). The FCIDUMP file contains active orbital integrals only.

```
[gugaci]
independent
integral = FCIDUMP
2S = 0
active = 6
nelec = 6
pg = c1
irrep = 1
```

8 Møller-Plesset Perturbation Theory

The MP2 method in Kylin requests a restricted Hartree-Fock calculation, which is followed by a second-order Møller-Plesset correction on energy. [7] Unrestricted MP2 calculation will be supported in the future.

The MP2 module can be initialized with `[mp2]`. Keywords of the `[mp2]` module are listed in Table. 6.

Keyword	**Description**
frozen	Total number of frozen orbitals.
frozen_irrep	Number of frozen orbitals in each irreducible representation.
full	Whether all of occupied orbitals are taken into consideration. If `true`, no orbitals are frozen. The default value is `false`.

Table 6: Keywords in MP2.

The MP2 module regrads non-valence orbitals as frozen by default. Keywords `frozen`, `frozen_irrep` and `full` can be used to specify the number of frozen orbitals. To avoid conflict, only one of them can be valid. The following is an example of MP2 calculation of C_6H_6 benzene with point group symmetry.

```
[info]
    coord = benzene.xyz
    basis = 6-31g

[integral]

[hf]

[mp2]
    # frozen = 6
    frozen_irrep = 2 0 2 0 1 0 1 0
    # full
```

9 DMRG

In Kylin package, the implemented DMRG method is based on the matrix-product state (MPS) ansatz. The symmetry-protected DMRG method is implemented as the `[DMRG]` module. The `[DMRG]` module support the $U(1)$ and $SU(2)$ spin symmetry and the point group symmetry listed in Table 2. The `[DMRG]` module can run in double-precision and mixed-precision. The mixed-precision DMRG is suggested when the truncation dimension M is larger than 1500. The keywords of the `[DMRG]` module are listed in Table 7.

Keyword	**Description**
2S	Spin, default 0 for singlet.
active	Number of active orbitals in each irrep.
charge	Charge, default 0.
docc	Number of doubly-occupied orbitals in each irrep.
dump	Print active MO integrals to file FCIDUMP and not perform CI calculation, default `false`.
dumpAll	Print all of MO integrals to file FCIDUMP and not perform CI calculation, default `false`.
eigensolver	Algorithm of eigensolver. `lanczos`, `davidson` and `jacdav` for Lanczos, Davidson and Jacobi-Davidson algorithms respectively. Default is `davidson`.
ENV_GT_MEM	Set if the size of the two largest ENVs lager than the computer memory, default `false`.
MPS_dir	The path to read or write MPS files, when reading MPS files as the initial guess (`init_MPS_from_file=true`) or storing MPS (`store_MPS=true`), default `tmp/mps`.
MPS_only	Only read MPS files and not perform DMRG calculation, default `false`.
InitialConfig	Initial guess configuration. For detailed usage, see Section 14.2.

Keyword	**Description**
irrep	Irreducible representation of target state(s) (start from 1), default 1.
M	Number of reserved states. Need three input integer as final reserved states, step increment, starting number of reserved states, delimited with space.
maxCycle	Max number of diagonalization iteration, default 100.
maxSpace	Max space of diagonalization, default 10.
mixed_double	Using mixed-precision diagonalization in the double-precision sweeps, default `true`.
nstate	Number of states to calculate, default 1.
nsweeps	Maximal number of DMRG sweeps.
nsweeps_d	Maximal number of double-precision DMRG sweeps. Active when using mixed precision DMRG.
nsweeps_f	Maximal number of single-precision DMRG sweeps. Active when using mixed precision DMRG.
nsweeps_screen	Maximal number of screen sweeps.
reorder_method	Orbital reorder method, default 1. Available inputs are: 0 (no reorder) and 1 (Fiedler orbital reordering).
store_ENV	Store ENV to disk, default `true`.
store_MPO	Store MPO to disk, default `false`.
store_MPS	Store MPS to disk, default `true`.
sweep_precision	The precision for the DMRG sweep. Available input are: `double`, `mix` and `float`. The `float` perform full single-precision DMRG, may generate unreliable results. For feasible single-precision DMRG, run sweep_precision=mix, nsweeps_d=0.
symm	Spin symmetry. Available input are: `2u1` and `su2u1` for $U(1)$ and $SU(2)$ spin symmetry respectively.

Keyword	Description
tol_davidson	Tolerance for diagonalization. Need two input as the starting tolerance and final tolerance.
tol_E	Tolerance for DMRG, default 1e-8.
tol_svd	Tolerance for SVD. Need two input as the starting tolerance and final tolerance.
use_cadmrg	Use DMRG with continuous address storage in memory, default `true`.
MPO_first	Contract MPO first rather than MPS for calculating $\hat{H}\|\Psi\rangle$ when `use_cadmrg` is `true`, default `true`.
twoSite	Perform two-site algorithm DMRG, default `true`. If `false`, perform one-site algorithm DMRG.
2sitePlus1site	Perform one-site algorithm DMRG after two-site algorithm DMRG finished, default `true`.
sample_thrs	default `-1`. If `sample_thrs` $\geq$ `0`, sample all configurations with coefficients larger than `sample_thrs` when DMRG calculation is finished.

Table 7: Keywords in DMRG.

9.1 Start with Integral File (FCIDUMP)

The DMRG module in Kylin could be used as a stand-alone program if the molecular orbital integrals are provided. Some extra required parameters are listed in Table 8.

Keyword	Description
independent	Perform stand-alone DMRG calculation, default `false`.
integral	FCIDUMP file name. In this file, the irreps and orbital indices both start from 1.

nelec	Number of total electrons. If FCIDUMP file does not contain doubly-occupied orbitals, `nelec` = number of active electrons and `docc` should be 0 for each irrep.
pg	Point group symmetry of the system, default `c1`.

Table 8: Extra required keywords for stand-alone DMRG.

Here is an example of hexacene with CAS(26e, 26o). The FCIDUMP file contains active orbital integrals only.

```
[dmrg]
independent
integral = FCIDUMP
2S = 0
active = 26
nelec = 26
pg = c1
irrep = 1
use_cadmrg
mpo_first
store_env
```

9.2 Sample Configurations

The parameter `sample_thrs` could be used to sample important configurations in DMRG wavefunction (via the method from the appendix of Ref. [8]). If `sample_thrs` $\geq$ `0`, the sampling will be performed when DMRG calculation is finished. All of the configurations with (absolute) coefficients larger than `sample_thrs` will be written in file `tmp/sample`*n*`.out` (including their coefficients), here n represents the nth state.

Some notes:

1. If the keyword `symm = 2u1` in DMRG calculation, the sampled configurations are Slater determinants. If `symm = su2u1`, the sampled configurations are CSFs.

2. If Fiedler order is used in DMRG calculation (i.e. `reorder_method = 1`), the MOs of sampled configurations will be in that order.
3. The `irrep` and `2S` keywords in DMRG calculation determine the final irrep and spin multiplicity of the sampled configurations.

> If a previous DMRG calculation is done and the MPS files are saved on disk, one could use the keyword `MPS_only` to skip DMRG calculation and perform sampling only.

Here is an example. Assume that a DMRG calculation is performed first using the input file in Section 9.1 and the MPS files are saved in directory `./tmp/mps`.

```
[dmrg]
independent
integral = FCIDUMP
2S = 0
active = 26
nelec = 26
pg = c1
irrep = 1
use_cadmrg
mpo_first
store_env
mps_dir = ./tmp/mps
mps_only
sample_thrs = 1e-3
```

10 CASSCF and DMRG-SCF

The CASSCF and DMRG-SCF modules in Kylin are based on the second-order Werner-Meyer-Knowles method. [9–14] In these two modules, both CI coefficients and molecular orbital coefficients are optimized simultaneously. These two methods are widely applied to evaluate static electronic correlations in molecular systems, and their wave functions are essential for multi-reference dynamic electronic correlation calculations.

The CASSCF method applies to small active spaces with less than 16 active orbitals. For larger active spaces, we recommend using the DMRG-SCF method.

> Note that the code with point group symmetry for MO coefficients optimization is not robust currently, thus the point group symmetry is not supported for CASSCF/DMRG-SCF. However, if perform CASCI/DMRG-CASCI only, the point group symmetry is supported.

10.1 CASSCF

The CASSCF module can be initialized with either `[mcscf]` or `[casscf]`. Before initializing the `[casscf]` module, a set of initial molecular orbitals and integrals must be accessible, which are usually generated by Hartree-Fock calculations. Therefore, in most cases the `[info]`, `[integral]` and `[scf]` modules should be finished before using `[casscf]`. The keywords in CASSCF calculations are listed in Table.9.

Keyword	**Description**
2S	The value of spin multiplicity $2S$, default `0` (singlet).
pg	Point group symmetry. If running standalone MCSCF with keyword `independent`, `pg` is required, default `c1`.
irrep	Irrep index of the target state, default `1`.
frozen	Number of frozen orbitals in each irreducible representation of the point group symmetry (irrep).
docc	Number of doubly-occupied orbitals in each irrep.
active	Number of active orbitals in each irrep.

Keyword	**Description**
nactel	Number of active electrons (all electrons minus twice the number of inactive and frozen orbitals). This keyword mustn't be omitted if running standalone MCSCF with keyword `independent`.
charge	Charge of the system, default 0. It is incompatible with keywords `nactel` and `independent`. Because you can destroy the system's neutrality by setting `nactel` directly. Default `0`.
rotate	The indices of molecular orbitals to be swapped. This option is helpful if the indices of active orbitals are discontinuous. For example, `rotate = 1 3 2 4` indicates exchanging the orbital 1 with 3, and 2 with 4. Default `None`. Note: only orbitals with the same irreducible representation are allowed for exchange.
nstate	Number of state(s) to be solved, default `1`.
weights	Weights of different states from state 1 to state n in the average energy. For example, if `nstate = 3` and `weights = 1 1 3`, the average is taken over three states from state 1 to state 3 with weights 20%, 20% and 60%, respectively. NOTE: if you set `nstate` without specifying weights, all states's weights will be set to 1/n automatically. Default `None`.
max_macro	Maximal number of the macro-iterations, default `100`.
max_micro	Maximal number of the micro-iterations in each macro-iteration, default `50`.
tol_macro	Convergence threshold of macro-iterations, default `1.0E-6`.
tol_micro	Convergence threshold of micro-iterations, default `1.0E-6`.
tol_davidson	Convergence threshold in matrix diagonalization, default `1.0E-6`.
independent	Run a standalone `[MCSCF]` calculation with a complete integral file(not only active orbitals) specified by keyword `integral`. Default `false`. NOTE: it is incompatible with `[geoopt]` module.
integral	A complete integral file name to run a standalone MCSCF calculation. Default `None`.

Keyword	Description
dump_all	Generate full integrals when MCSCF is converged, default `false`. NOTE: the meaning of this keyword is a little different with that in modules `gugaci` and `dmrg`.
fileorb	External orbital file (currently the OpenMolcas orbital file format is used). It can be used to resume MCSCF from latest MCSCF orbitals. Default `None`.
rlx_state	Specify which state is to be relaxed in geometry optimization if `nstate > 1`, default `1`.
CI_only	Perform CASCI/DMRG-CASCI only, default `false`.

Table 9: Keywords in CASSCF.

10.2 DMRG-SCF

The DMRG-SCF module is initialized with `[dmrgscf]`. As the CASSCF and DMRG-SCF methods share the similar orbital optimization procedures, all keywords in Table 9 are also valid in DMRG-SCF module. Besides, the DMRG-SCF module also inherits all DMRG keywords in Table 7.

10.3 Start with Orbital File

The MCSCF calculation can be performed using keyword `fileorb`.

Since the MCSCF orbital is saved in a file named `MCSCFOrb` under `tmp/MCSCF/` directory, this file could be used to resume the MCSCF calculation.

Assume that we first perform a MCSCF calculation (or using some other ways) for benzene with CAS(6e, 6o) to generate an orbital file.

```
[info]
basis = cc-pvdz
coord = benzene.xyz
pg = c1
```

```
[integral]

[scf]

[casscf]
frozen = 0
docc = 18
active = 6
rotate = 17 19 24 30
2S = 0
```

Then we use the file `tmp/MCSCF/MCSCFOrb` to perform the MCSCF calculation.

```

[info]
basis = cc-pvdz
coord = benzene.xyz
pg = c1

[casscf]
fileorb = tmp/MCSCF/MCSCFOrb
frozen = 0
docc = 18
active = 6
2S = 0
```

> The active orbitals are consecutive now, thus the keyword `rotate` is not required. Moreover, `[casscf]` could invoke the `[integral]` module to calculate the orbital integrals, thus the `[integral]` module is unnecessary here.

10.4 Start with Integral File (FCIDUMP)

The CASSCF and DMRG-SCF modules in Kylin can be used as stand-alone MCSCF solvers, as long as the initial molecular orbitals integrals are provided. In this case, the initial molecular orbitals and integrals are directly loaded from external files, while the `[info]`, `[integral]` and `[scf]` modules are not necessary. The standalone MCSCF solvers do not directly update the input molecular orbitals, instead the update matrix

$\mathbf{U}$ will be printed. The MCSCF orbitals $\mathbf{C}$ can be simply obtained by

$$\mathbf{C} = \mathbf{C}_0\mathbf{U},$$

in which $\mathbf{C}_0$ is the initial molecular orbital coefficient matrix, which is not required in the standalone MCSCF calculation.

To start a MCSCF calculation independently, the option `independent` must be enabled, and the initial molecular orbital integrals must be provided by setting the `integral` keyword to a FCIDUMP file. The FCIDUMP is a standard format to store one- and two-electron integrals, and can be generated by many quantum chemistry softwares such as Molpro, Molcas as well as Kylin (in modules `[gugaci]`, `[dmrg]` and `[mcscf]`, all orbital integrals will be written to running path by setting keyword `dumpAll` to `true`).

> For the standalone MCSCF calculation in Kylin:
>
> - The input FCIDUMP file must contain integrals of all the molecular orbitals (not only active but also inactive orbitals).
> - Incompatible with geometry optimization, because geometry optimization requires orbitals.

Here is an example for benzene with CAS(6e, 6o). The FCIDUMP file contains all MO integrals.

```
[casscf]
independent
integral = FCIDUMP
frozen = 0
docc = 18
active = 6
nactel = 6
2S = 0
```

11 MRCI

The multi-reference configuration interaction (MRCI) module in Kylin supports the uncontracted (UC) and the externally contracted (EC) MRCI methods. This module can be initialized with the label `[mrci]`.

Since the second order Epstein-Nesbet perturbation theory (ENPT2) energy could be obtained during the EC-MRCI calculation, this module could also perform ENPT2 calculation.

Currently, the configuration state function (CSF) which is the eigenfunction of operator $\hat{S}^2$ is used as wavefunction basis. Moreover, the electron excitations from doubly-occupied (or core) orbitals to other orbitals are not considered.

The keywords in the MRCI module are listed in Table 10.

Keyword	**Description**
method	Which method to be performed, should be `uc`, `ec` or `enpt2`, default `ec`. Note that if `method = ec`, the ENPT2 energy could also be obtained.
ref	Reference configurations, default `all` for all CAS configurations. If reference configurations are selected, these selected configurations ("2", "u", "d" and "0" represent double, positively spin-coupled, negatively spin-coupled and zero occupation respectively) should be written in a file (one configuration on each line) and the file name should be passed to this keyword.
2S	The value of spin multiplicity $2S$, default `0` (singlet).
frozen	Number of frozen orbitals in each irreducible representation of the point group symmetry (irrep). The frozen orbitals are doubly-occupied and the electron excitations from frozen orbitals to other orbitals are not allowed.
active	Number of active orbitals in each irrep.
external	Number of external orbitals in each irrep, default all of external orbitals in each irrep.
charge	Charge of the system, default `0`.
irrep	Irrep index of the target state(s), default `1`.

nstate	Number of state(s) to be solved, default `1`.
directCI	Perform direct CI calculation, default `false`.
eigensolver_cas	Eigensolver for reference space CI calculation before MRCI, should be `lanczos`, `davidson` or `jacdav` for Lanczos, Davidson and Jacob-Davidson algorithms respectively, default `lanczos`.
eigensolver	Eigensolver for MRCI calculation, should be `lanczos`, `davidson` or `jacdav` for Lanczos, Davidson and Jacobi-Davidson algorithms respectively, default `davidson`.
refGuess	User specified initial guess for reference Hamiltonian diagonalization, default empty. For example, a singlet system using CAS(2e, 2o) as reference space, if `refGuess = 20 02` and assuming that index(20) = 0 (C-style, starting from 0) and index(02) = 2, then [1, 0, 1] will be used as initial guess.
initH0	Initial guess for reference Hamiltonian (H0, not $\hat{H}_0$ of ENPT2) diagonalization (only valid when refGuess is empty), should be `0` for [[1, 0, 0, ...], [0, 1, 0, ...]], `1` for [1, 1, 1, ...], `2` for [diagElem(H0)] or `3` for [random numbers and, 1000 * thisvector[maxIndex(diagElem(H0))]], default `3`.
tol_H	Threshold value of saving Hamiltonian matrix elements when `directCI = false`, default `1.0E-8`.
tol_E	Threshold value of diagonalization convergence, default `1.0E-10`. If `directCI = true`, `1.0E-7` is recommended for balanced precision and cost.
tol_C	Threshold value of contraction coefficients for EC-MRCI, default `1.0E-6`.
tol_O	Threshold value of orthogonalization for EC-MRCI when `nstate` > 1, default `1.0E-10`.
actOrbOrder	Order of active orbitals when `ref` $\neq$ `all`, default empty. For example, if reference configurations are obtained from DMRG wavefunction with Fiedler order, actOrbOrder should be specified with Fiedler order.

actExc	Allow electron excitations in active space when `ref` $\neq$ `all`, default `false`.
verbose	Print level, default `1`.
printWfn	Print CI coefficients to file MRCICoef-*n* (*n* is the state index), default `false`. If `verbose` ≥ 3, `printWfn` will be set to `true` automatically.

Table 10: Keywords in MRCI.

Here is an example of N_2.

```
[info]
coord = n2.xyz
basis = cc-pvdz
pg = c1

[integral]

[scf]

[mrci]
method = ec
2S = 0
frozen = 4
active = 6
```

11.1 Start with Integral File (FCIDUMP)

The MRCI module in Kylin could be used as a stand-alone program if the molecular orbital integrals are provided. Some extra required parameters are listed in Table 11.

Keyword	**Description**
independent	Perform stand-alone MRCI calculation, default `false`.
integral	FCIDUMP file name. In this file, the irreps and orbital indices both start from 1.
nelec	Number of total electrons. If FCIDUMP file does not contain doubly-occupied orbitals, `nelec` = number of active electrons and `frozen` should be 0 for each irrep.
pg	Point group symmetry of the system, default `c1`.

Table 11: Extra required keywords for stand-alone MRCI.

Here is an example of hexacene. The FCIDUMP file contains active and external orbital integrals. The CollectedCFGs.txt contains the selected reference CSFs.

```
[mrci]
independent
method = ec
integral = FCIDUMP
ref = CollectedCFGs.txt
2S = 0
active = 26
nelec = 26
pg = c1
irrep = 1
```

The `ref` file (CollectedCFGs.txt) supports two formats.

First, one CSF and the corresponding coefficient a line (this format is the same as the output file of EDGA module)

```
22222222222220000000000000 0.93
```

Second, one CSF a line

```
22222222222220000000000000
```

12 MRPT

Currently, Kylin only supports the second order Epstein-Nesbet perturbation theory (ENPT2) using CASCI/CASSCF/sCI/DMRG2sCI wavefunction as reference. Since the ENPT2 energy could be obtained during the EC-MRCI calculation, the ENPT2 calculation is implemented in MRCI module (see Section 11).

13 Geometry Optimization

Geometry Optimization module [geoopt] in Kylin supports the localization of equilibrium structure in redundant internal coordinates. Currently, only the following methods are available:

(1) restricted Hartree-Fock (only supports `[hf]` module currently);

(2) state-specific CASSCF without froen orbitals;

(3) state-specific DMRGSCF without frozen orbitals.

This version has several limitations:

(1) only support Abelian point group, non-Abelian point group can't be sustained with 100% confidence;

(2) can't deal with the degradation of structural symmetry.

Ways to smash above limitations, Z_mat format, constrained optimization and other functions will be supported in the future.

Besides, attention must be paid to the memory requirement in MCSCF geometry optimization: Twice the memory of two-electron atomic integrals is required by MCSCF while four times the memory of two-electron atomic integrals is required by MCSCF geometry optimization, so you'd better select a suitable machine to do MCSCF geometry optimization.

The Geometry Optimization module can be started with `[geoopt]`. The keywords in this module are listed in Table 12.

Keyword	**Description**
method	The method used to do single point calculation, default `hf`. Available values: `hf`, `casscf` and `dmrgscf`.
optimizer	Optimization method used to locate suitable structures, `rfo`, `gdiis` and `gediis` for Rational Functional optimization, geometry-based and energy-based direct inversion in the iterative subspace algorithms respectively. Default is `rfo`.
hess_type	Option to specify the type of guess hessian which will be updated by Broyden–Fletcher–Goldfarb–Shanno formula, including `schlegel` and `model`. Default `schlegel`.
max_opt_iter	Max number of optimization iterations, default `50`.
max_opt_step	Max step length of each optimization iteration, default `0.3`(Angstrom). If an atom's step overshoots this value, actual optimization step length will be scaled.

Keyword	Description
max_force	Threshold for maximum force in redundant internal coordinates of an optimal structure, default `0.00045`(hartree/bohr or hartree/rad).
rms_force	Threshold for root-mean-square deviation(RMSD) of force in Cartesian coordinates of an optimal structure, default `0.0003`(hartree/bohr).
max_displace	Threshold for maximum displacement in Cartesian coordinates of an optimal structure, default `0.0018`(Angstrom).
rms_displace	Threshold for root-mean-square deviation(RMSD) of displacement in Cartesian coordinates of an optimal structure, default `0.0012`(Angstrom).
diis_size	Size of the GDIIS/GEDIIS space, default `3`.
numeric_hess	Option to calculate numeric hessian matrix of optimal structure, default `false`. Note: this is only for point group = C1

Table 12: Keywords in Geometry Optimization.

13.1 Output Norm of Geometry Optimization

(1) During geometry optimization, energy, max_force, rms_force, max_displace and rms_displace of each structure is updated in a text file named "GeooptList.txt" in the current working directory, so you can check the progress of geometry optimization directly.If max_opt_iter is too small, the end of GeooptList.txt won't tell if geometry optimization converged or not.

(2) All structures generated in geometry optimization are saved to a folder with the same name of ***.xyz or ***.molden you specify in `[info]`. This folder is created under `tmp` path.

(3) Cartesian coordinates, Cartesian gradient and steps are all written in x.xyz file(0.xyz is the initial guess structure specified in `[info]`, 1.xyz is the first structure produced by geometry optimization) and saved to the above folder. If abortion occurs during MCSCF geometry optimziation, you can use x.xyz and corresponding MCSCFOrb with keyword "continue = true" to resume geometry optimization directly.

(4) For visualization, each structure has a copy of x.gjf file generated at the same time.

(5) If numeric hessian is calculated, hessian matrix will be updated in text file numHess.txt in time. And all frequencies as well as zero point energy will be output at the tail of your log file.

13.2 Start MCSCF Geometry Optimization from [HF] Module

(1) If a molecular's active space has been picked out, you can run MCSCF geometry optimization from [hf] module, as following input file shows:

```
[info]
basis = sto-3g
coord = c18h18.xyz

[integral]

[hf]
guess = true
moint = false
acc_conv = cdiis
tol = 5e-5

[dmrgscf]
irrep = 1
2S = 0
frozen = 0 0 0 0
docc = 28 26 0 0
active = 0 0 9 9
M = 500 300 50
print_mo_indices
tol_macro = 1e-5
nstate = 1
InitialConfig = 222220000222200000

[geoopt]
method = dmrgscf
optimizer = gediis
```

```
hess_type = schlegel
```

The geometry of $C_{18}H_{18}$ polyacetylene is defined in the file `c8h18.xyz` in chapter Examples.

(2) Start MCSCF geometry optimization from saved HFOrbForMCSCF file, if you have generated Hartree-Fork orbitals:

```
[info]
basis = sto-3g
coord = c18h18.xyz

[dmrgscf]
fileorb = tmp/HFOrbForMCSCF
irrep = 1
2S = 0
frozen = 0 0 0 0
docc = 28 26 0 0
active = 0 0 9 9
M = 400 200 50
#reorder_method=0
print_mo_indices
tol_macro = 1e-5
nstate = 1
InitialConfig = 222220000222200000

[geoopt]
method = dmrgscf
optimizer = gdiis
hess_type = schlegel
```

13.3 Start MCSCF Geometry Optimization from MCSCF Orbitals

(1) You can resume MCSCF geometry optimization from any external molden if point group = C1. Take benzene's geometry optimization with point group = C1 as example:

```
[info]
inputtype = molden
moldenfile = tmp/MCSCF/MCSCFOrb.molden

[integral]

[dmrgscf]
irrep = 1
2S = 0
frozen = 0
docc = 18
active = 6
’’’
#because 17 has exchanged with 19 in former mcscf calculation
#so remove keyword "rotate"
#rotate = 17 19
’’’
print_mo_indices

[geoopt]
method = dmrgscf
optimizer = gediis
hess_type = schlegel
```

(2) You can resume MCSCF geometry optimization from MCSCF exported orbitals of initial guess structure:

```
[info]
basis = sto-3g
coord = t-Bu-DHP.xyz

[dmrgscf]
continue = true
fileorb = tmp/MCSCF/MCSCFOrb
irrep = 2
2S = 0
frozen = 0 0
```

```
docc = 44 43
active = 7 7
M = 400 100 50
reorder_method = 0
print_mo_indices
'''
#because active space has been made consecutive
#in former mcscf calculation
#so remove keyword "rotate"
#rotate = 43 45 44 46
'''
nstate = 1
InitialConfig = 222u000222d000
[geoopt]
method = dmrgscf
optimizer = gediis
hess_type = schlegel
```

(3) You can resume MCSCF geometry optimization from MCSCF exported orbitals of latest generated structure if abortion occurs:

```

[info]
basis = sto-3g
coord = tmp/c8h18/3.xyz
#because coordinates has been rotated in single point calculation
#we have to maintain the consistency of orbitals and coordinates
rotcoord = false

[dmrgscf]
continue = true
fileorb = tmp/MCSCF/MCSCSFOrb
irrep = 1
2S = 0
frozen = 0 0 0 0
docc = 28 26 0 0
active = 0 0 9 9
M = 400 200 50
```

```
#reorder_method=0
print_mo_indices
tol_macro = 1e-5
nstate = 1
InitialConfig = 222220000222200000

[geoopt]
method = dmrgscf
optimizer = gdiis
hess_type = schlegel
```

14 Examples

In this section, we give some practical examples to show how to use Kylin in quantum chemistry researches.

14.1 CASSCF and DMRG-SCF

The following is an example of CASSCF calculation of $C_{12}H_{14}$ polyacetylene.

```
[info]
    coord = c12h14.xyz
    basis = 6-31g
    pg = c1

[integral]

[scf]

[casscf]
    2S = 0
    frozen = 0
    docc = 37
    active = 12
    tol_davidson = 1e-7
    nstate = 1
    nactel = 12
    rotate = 47 53 48 54
```

The geometry of $C_{12}H_{14}$ polyacetylene is definded in the file `c12h14.xyz`, as

```
 26
Angstrom
 C                 -0.08389600   4.32582500   0.00000000
 H                 -1.17875300   4.39409800   0.00000000
 C                  0.49694800   3.09580500   0.00000000
 H                  1.59237900   3.03292300   0.00000000
 C                 -0.22577000   1.85266800   0.00000000
```

```
H                 -1.32108200   1.91302100   0.00000000
C                  0.36008900   0.62116400   0.00000000
H                  1.45554400   0.56169500   0.00000000
C                 -0.36008900  -0.62116400   0.00000000
H                 -1.45554400  -0.56169500   0.00000000
C                  0.22577000  -1.85266800   0.00000000
H                  1.32108200  -1.91302100   0.00000000
C                 -0.49694800  -3.09580500   0.00000000
H                 -1.59237900  -3.03292300   0.00000000
C                  0.08389600  -4.32582500   0.00000000
H                  1.17875300  -4.39409800   0.00000000
C                 -0.65094200  -5.57286900   0.00000000
H                 -1.74484800  -5.49307100   0.00000000
C                 -0.08389600  -6.79496400   0.00000000
H                  1.00336100  -6.91703400   0.00000000
H                 -0.68811500  -7.70416700   0.00000000
C                  0.65094200   5.57286900   0.00000000
H                  1.74484800   5.49307100   0.00000000
C                  0.08389600   6.79496400   0.00000000
H                 -1.00336100   6.91703400   0.00000000
H                  0.68811500   7.70416700   0.00000000
```

The active space used in the CASSCF calculation consists of 12 π orbitals and 12 π electrons. However, the indices of the π orbitals are not consecutive. We use the `rotation` option to exchange the 47-th molecular orbital with the 53, and 48 with 54.

We can also use DMRG-SCF method to perform MCSCF calculation. However, if there are less than 16 active orbitals, the classical CASSCF method are usuaslly much cheaper, faster than the DMRG-SCF method.

14.2 Initial Guess of DMRG

Currently, there is only one initial guess MPS option for DMRG in Kylin. The initial guess is generated by performing $\hat{H}|\Psi\rangle$ where $|\Psi\rangle$ is a single configuration MPS. Sometimes, the default configuration (HF configuration if `pg = c1`) is not suitable especially considering point group symmetry.

The keyword `InitialConfig` could be used to specify a more suitable configuration (only active space) in the form like "2ud0". Here "2" and "0" represent double and zero occupation respectively. For CSF (`symm = su2`), "u" and "d" represent positively and negatively spin-coupled occupation respectively. For Slater determinant(`symm = 2u1`), "u" and "d" represent spin-up and spin-down occupation respectively.

A suitable initial configuration MUST meet TWO requirements, Otherwise, an assertion related to quantum number error may occur like "system_qn == Q0".

1. The direct product of irreducible representations of all active electrons MUST EQUAL to the value of keyword `irrep` you specified.
2. The spin number(2S) of active space MUST EQUAL to the value of keyword `2S` you specified.

Here are two examples to illustrate how to specify keyword `InitialConfig` in `[dmrg]`:

(1) Ground state of benzene with point group D_{2h}:

```
[info]
basis = sto-3g
coord = benzene.xyz

[integral]

[scf]

[dmrg]
#ground state
irrep = 1
2S = 0
frozen = 0 0 0 0 0 0 0 0
docc = 6 5 4 3 0 0 0 0
active = 0 0 0 0 2 2 1 1
M = 1000
#In tmp/hf.molden, six active PI orbitals’
#irreducible representation as well as
#occupation numbers are
#B1u B1u B2g B2g B3g Au
# 2  0   2  0   2  0
#According to direct product rules
#of electronic configuration and 2S = 0
#202020 is suitable
initialconfig = 202020
```

(2) 1^1B_{2u} state of benzene with point group D_{2h}:

```
[info]
basis = sto-3g
coord = benzene.xyz

[integral]

[scf]

[dmrg]
#B2u's first singlet state
irrep = 3
2S = 0
frozen = 0 0 0 0 0 0 0 0
docc = 6 5 4 3 0 0 0 0
active = 0 0 0 0 2 2 1 1
M = 1000
#In tmp/hf.molden, six active PI orbitals'
#irreducible representation as well as
#occupation numbers are
#B1u B1u B2g B2g B3g Au
# 2   0   2   0   2  0
#According to direct product rules
#of electronic configuration
#B2u = B1u * B3g
#as well as 2S = 0
#2u20d0 is suitable
InitialConfig = 2u20d0
```

14.3 Mixed-Precision DMRG

The following is an example of mixed-precision DMRG calculation of benzene molecule. Note that do not use sweep_precision = single to run single-precision sweeps in DMRG calculations because they may lead to unreliable results. To run reliable single-precision, use sweep_precision = mix and nsweep_d = 0.

```
[info]
basis = 6-31g
coord = benzene.xyz
pg = c1

[integral]

[hf]
2S = 0
guess = true

[dmrg]
2S = 0
docc = 11
active = 20
M = 2000 400 400
tol_davidson = 1e-3 1e-5
tol_svd = 1e-50 1e-7
sweep_precision = mix
mixed_double = true
symm = su2u1
```

The geometry of benzene molecule is definded in the file `benzene.xyz`, as

```
12

6                 1.21364800   0.70070000   0.00000000
6                 0.00000000   1.40140000   0.00000000
6                -1.21364800   0.70070000   0.00000000
6                -1.21364800  -0.70070000   0.00000000
6                 0.00000000  -1.40140000   0.00000000
6                 1.21364800  -0.70070000   0.00000000
1                 2.14029518   1.23570000   0.00000000
1                 0.00000000   2.47140000   0.00000000
1                -2.14029518   1.23570000   0.00000000
1                -2.14029518  -1.23570000   0.00000000
1                 0.00000000  -2.47140000   0.00000000
```

```
1                2.14029518  -1.23570000   0.00000000
```

14.4 HF geometry optimization

The following is an example of Hartree-Fock geometry optimization of $C_{18}H_{18}$ polyacetylene.

```
[info]
basis = sto-3g
coord = c18h18.xyz

[integral]

[hf]
guess = true
moint = false #molecular integral is not required in hf geoopt
acc_conv = cdiis
tol = 1e-5

[geoopt]
method = hf
optimizer = rfo
hess_type = schlegel
max_opt_iter = 20
```

The geometry of $C_{18}H_{18}$ polyacetylene is definded in the file `c18h18.xyz`, as

```
36

C                0.00000000   0.00000000   0.00000000
C                0.00000000   0.00000000   3.03674316
C                0.13922104   0.00000000  -1.30314036
C                0.13922104   0.00000000   4.33988352
C                1.41586478   0.00000000  -2.04606030
C                1.41586478   0.00000000   5.08280346
C                2.61089088   0.00000000  -1.50951328
```

```
C                  2.61089088    0.00000000    4.54625644
C                  3.94052985   -0.00000000   -2.12894389
C                  3.94052985   -0.00000000    5.16568705
C                  5.09745163    0.00000000   -1.45518376
C                  5.09745163    0.00000000    4.49192692
C                  5.23384423    0.00000000    0.00391420
C                  5.23384423    0.00000000    3.03282896
C                  6.28517534    0.00000000    0.78068273
C                  6.28517534    0.00000000    2.25606043
C                 -1.19423177    0.00000000    2.18675605
C                 -1.19423177    0.00000000    0.84998712
H                  0.91974131    0.00000000    0.54928970
H                  0.91974131    0.00000000    2.48745346
H                 -0.75703156    0.00000000   -1.91246154
H                 -0.75703156    0.00000000    4.94920470
H                  1.33618480   -0.00000000   -3.12702166
H                  1.33618480   -0.00000000    6.16376482
H                  2.62506861    0.00000000   -0.43885687
H                  2.62506861    0.00000000    3.47560003
H                  3.98182577   -0.00000000   -3.21228061
H                  3.98182577   -0.00000000    6.24902377
H                  6.01512073    0.00000000   -2.03216572
H                  6.01512073    0.00000000    5.06890888
H                  4.29879649    0.00000000    0.52406896
H                  4.29879649    0.00000000    2.51267420
H                  7.26231500    0.00000000    0.31190553
H                  7.26231500    0.00000000    2.72483763
H                 -2.15318908    0.00000000    0.34445811
H                 -2.15318908    0.00000000    2.69228505
```

14.5 Start with Molden File

The basis set and molecular orbital information could be saved in molden file. Thus the molden file could be used to perform HF, CASCI, DMRG, MCSCF and MRCI calculations in Kylin. In module `[info]`, the keyword `inputType` should be set to `molden` and the keyword `moldenFile` should be specified.

> The point group symmetry is not supported if using molden file to perform these calculations.

Here we take benzene as an example. Assume that we use another software to perform an HF calculation with cc-pVDZ basis set and output the molden file (named `benzene.molden`) in which the 6 π orbitals are consecutive. The examples below show how to perform DMRG, MCSCF and MRCI calculations.

- DMRG.

```
[info]
inputType = molden
moldenFile = benzene.molden

[integral]

[dmrg]
docc = 18
active = 6
2S = 0
M = 1000
```

- MCSCF.

```
[info]
inputType = molden
moldenFile = benzene.molden

[integral]
# Actually, [integral] is unnecessary
# if basis set and MO information are provided
# [casscf] could invoke [integral]

[casscf]
frozen = 0
docc = 18
active = 6
```

```
2S = 0
```

- MRCI.

```
[info]
inputType = molden
moldenFile = benzene.molden

[integral]

[mrci]
method = ec
frozen = 18
active = 6
2S = 0
```

15 Deprecated Modules

15.1 [HF]

The Hartree-Fock module `[hf]` in Kylin supports both the restricted and the unrestricted Hartree-Fock methods with acceleration techniques like direct inversion of the iterative subspace (DIIS). When a Hartree-Fock calculation is successfully converged, a set of canonical molecular orbitals is saved to the `tmp/hf.ScfOrb` file, which is a standard Molcas-style orbital file and can be used independently. The corresponding molden file is saved as `tmp/hf.molden`, which can be visualized with the Multiwfn software. Besides, population analysis will be automatically performed at the end of Hartree-Fock calculations.

The keywords in the Hartree-Fock module are listed in Table 13. Note that the point group symmetry of the molecule is directly taken from the `[info]` module and can not be changed in the Hartree-Fock module.

Keyword	**Description**
2S	(integer) default: 0 The value of spin multiplicity $2S$.
AccConv	(case-insensitive string) default: `cdiis` options: `cdiis`, `c2diis`, `ediis`, `mixed`, `off` Acceleration technique to be used in the self-consistent field procedure.
CalMOIntegrals	(bool) default: `true` aliases: CalMOInt, CalMOIntegral, MOInt, MOIntegral, MOIntegrals Whether to generate molecular orbital integrals at the end of the Hartree-Fock calculation.
Charge	(integer) default: 0 Charge of the system.
DIISSpace	(integer) default: 10

Keyword	Description
	Size of the DIIS space. This option is valid only if a DIIS-type accelerator is enabled.
Guess	(bool) default: true
	Whether to use an optimized matrix for initial guess.
MaxIter	(integer) default: 500
	Maximum number of Hartree-Fock iterations.
PopulationAnalysis	(case-insensitive string) default: `mulliken`
	alias: PopAnalysis
	options: `cspa`, `mulliken`
	Population analysis method.
Tol	(double) default: `1.0e-6`
	Energy convergence tolerance of the Hartree-Fock interations.
UHF	(bool) default: `false`
	Whether use the unrestricted Hartree-Fock method. This option must be set to `true` manually if UHF is required.

Table 13: Keywords in Hartree-Fock.

15.2 [EDGA]

The entanglement-driven genetic algorithm (EDGA) module could be used to sample important configurations (i.e. with large coefficients) in DMRG wavefunction. This module can be initialized with the label `[edga]`.

Some notes:

1. Currently, only genetic algorithm is supported and the entanglement-driven one will be implemented in the future.
2. If the keyword `symm = 2u1` in DMRG calculation, the sampled configurations are Slater determinants. If `symm = su2u1`, the sampled configurations are CSFs.
3. If uses Fiedler order in DMRG calculation (i.e. `reorder_method = 1`), the MOs of sampled configurations will be in that order.
4. The `irrep` and `2S` keywords in DMRG calculation determine the final irrep and spin multiplicity of the sampled configurations.

The keywords in the `[edga]` module are listed in Table 14.

Keyword	**Description**
seed_file	A file containing initial guess configuration(s), one configuration a line. Incompatible with `result_file`. If using Fiedler order in DMRG calculation, the MOs of these configurations should also be in that order.
result_file	A file containing configurations and their coefficients, used to continue sampling with the sampled configurations last time. Incompatible with `seed_file`.
output_file	A file to save sampled configurations and their coefficients, default `CollectedCFGs.txt`.
loop	Maximum number of loop cycles, default `100000`.
target_thresh	Target completeness (i.e. sum of coefficients square), default `0.995`.
pop_size	Population size during genetic algorithm, default `2000`.
cross_rate	Cross rate during genetic algorithm, default `0.5`.

mutate_rate	Mutation rate during genetic algorithm, default `0.5`.
coeff_thresh	Threshold value of configuration coefficients, default `1.0E-6`.
ncyc_chk_list	When to check list and print some information, default every `50` cycles.
ncyc_save_rslt	When to save sampled configurations, default every `2000` cycles.
ncfgs_show	How many important configurations to show when printing information, default the top `20`.
verbose	Print level, default `1`.

Table 14: Keywords in EDGA.

The `[edga]` module requires the MPO and MPS generated by `[dmrg]` module. Consequently, the `[edga]` module could also run with a FCIDUMP file and the MPS files. Considering that one may run `[edga]` module a few times to sample important configurations as many as possible, we recommend to use the workflow listed below.

1. Perform DMRG calculation (`[dmrg]` or `[dmrgscf]`) and save MPS to files using the keyword `store_MPS`. It's better to save active orbital integrals to a FCIDUMP file at the same time.

2. Write a new input file that only contains `[dmrg]` and `[edga]` modules.

   The `[dmrg]` requires some keywords:

   (a) `independent`: To run as a stand-alone program.

   (b) `integral`: To load MO integrals which are used to construct MPO.

   (c) `MPS_dir`: To specify the MPS files directory.

   (d) `MPS_only`: To only load MPS files and not perform DMRG calculation.

   (e) Other required parameters for `[dmrg]` (e.g. `nelec`, `2S`, `active`, `pg`, `irrep`, `symm`, `reorder_method`, ...) must be identical to the input file in the first step.

   The important keywords in `[edga]`:

   (a) `seed_file`: To give initial guess configuration(s). Incompatible with `result_file`.

(b) `result_file`: To continue sampling with the sampled configurations last time. Incompatible with `seed_file`.

(c) `target_thresh`: Target completeness (i.e. sum of coefficients square).

(d) `loop`: Maximum number of loop cycles.

Here is an example. Assume that it's a hexacene with CAS(26e, 26o) in C_1 point group symmetry. The MPS files of previous DMRG calculation is saved in ./tmp/mps directory. The file FCIDUMP only stores the corresponding active orbital integrals. The initial guess configuration is written in file guess.txt (e.g. 22222222222222000000000000).

```
[dmrg]
independent
integral = FCIDUMP
2S = 0
active = 26
nelec = 26
symm = su2u1
pg = c1
irrep = 1
M = 1000
reorder_method = 1
MPS_only
init_MPS_path = ./tmp/mps
store_MPS = false

[edga]
seed_file = guess.txt
target_thresh = 0.999
loop = 1000000
```

References

[1] Isaiah Shavitt. Matrix element evaluation in the unitary group approach to the electron correlation problem. *International Journal of Quantum Chemistry*, 14(S12):5–32, 1978.

[2] Isaiah Shavitt. The utilization of abelian point group symmetry in the graphical unitary group approach to the calculation of correlated electronic wavefunctions. *Chemical Physics Letters*, 63(3):421–427, 1979.

[3] Bernard R. Brooks and Henry F. Schaefer. The graphical unitary group approach to the electron correlation problem. methods and preliminary applications. *The Journal of Chemical Physics*, 70(11):5092–5106, 1979.

[4] Bernard R Brooks, William D Laidig, Paul Saxe, Nicholas C Handy, and Henry F Schaefer. The loop-driven graphical unitary group approach: A powerful method for the variational description of electron correlation. *Physica Scripta*, 21(3-4):312–322, jan 1980.

[5] Josef Paldus. Unitary group approach to many-electron correlation problem. In Jürgen Hinze, editor, *The Unitary Group for the Evaluation of Electronic Energy Matrix Elements*, pages 1–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1981.

[6] Isaiah Shavitt. The graphical unitary group approach and its application to direct configuration interaction calculations. In Jürgen Hinze, editor, *The Unitary Group for the Evaluation of Electronic Energy Matrix Elements*, pages 51–99. Springer Berlin Heidelberg, Berlin, Heidelberg, 1981.

[7] Christian Møller and Milton S. Plesset. Note on an approximation treatment for many-electron systems. *Physical Review*, 46(7):618, 1934.

[8] Seunghoon Lee, Huanchen Zhai, Sandeep Sharma, C. J. Umrigar, and Garnet Kin-Lic Chan. Externally corrected CCSD with renormalized perturbative triples (R-ecCCSD(T)) and the density matrix renormalization group and selected configuration interaction external sources. *J. Chem. Theory Comput.*, 17(6):3414–3425, 2021.

[9] Hans-Joachim Werner and Wilfried Meyer. A quadratically convergent multiconfiguration–self-consistent field method with simultaneous optimization of orbitals and ci coefficients. *The Journal of Chemical Physics*, 73(5):2342–2356, 1980.

[10] Hans-Joachim Werner and Peter J Knowles. A second order multiconfiguration scf procedure with optimum convergence. *The Journal of Chemical Physics*, 82(11):5053–5063, 1985.

[11] Peter J Knowles and Hans-Joachim Werner. An efficient second-order mcscf method for long configuration expansions. *Chemical Physics Letters*, 115(3):259–267, 1985.

[12] Hans-Joachim Werner. Matrix-formulated direct multiconfiguration self-consistent field and multiconfiguration reference configuration-interaction methods. *Advances in Chemical Physics*, 69:1–62, 1987.

[13] Yingjin Ma, Stefan Knecht, Sebastian Keller, and Markus Reiher. Second-order self-consistent-field density-matrix renormalization group. *Journal of Chemical Theory and Computation*, 13(6):2533–2549, 2017.

[14] David A Kreplin, Peter J Knowles, and Hans-Joachim Werner. Second-order mcscf optimization revisited. i. improved algorithms for fast and robust second-order casscf convergence. *The Journal of Chemical Physics*, 150(19):194106, 2019.