Overview of Molcas

Taken from Valera Veryazov, as hacked by: Per Åke Malmqvist

Department of Theoretical Chemistry
Chemical Center, University of Lund
Sweden

NOTE: Most of the following slides have been taken directly from the Molcas introduction by Valera Veryazov, at the Molcas workshop in València, Spain, 2006-04-20 – 26.

Contents:

- installation of Molcas
- maintenance
- howto run Molcas
- howto get help
- howto create input
- howto use GUI

A code called MOLCAS

- source code (Fortran, C) + scripts (Perl, bash)
- can be configured for
  - different platforms: Unix, MS Windows, MacOS
  - serial, parallel (SMP or cluster), grid
  - 32- and 64- bit
  - all major compilers: G77, gfortran, Intel, PGI, PathScale, NAG, Absoft
- a compiled code includes:
  - set of executables to solve a QCh problem
  - driver scripts to parse input, and run executables
  - tools and interface programs
  - databases (basis set library)
- user interface includes:
  - input file
  - environment variables
  - intermediate files

Features and methods

- Wave Functions, Energies, Properties
  SCF, DFT, CASSCF/RASSCF, CASPT2, MRCI, RASSI, CCSD(T)
  ..
- Molecular Structures, Vibrational Frequencies, Thermodynamics
- Excited States and Electronic Spectra
- Environmental Effects

Typical applications with MOLCAS

- Accurate Potential Energy Curves
- Molecular spectroscopy
- Reaction mechanisms
- Biochemistry
- Heavy element chemistry

Installation1

Prerequisite software:

- Fortran77 and C compilers, Perl interpreter
- GNU make, uuencode
- GUI related software/libraries: OpenGL, glut, Python
- PATH variable

Configuration: from easy to hard

- *./configure*
- *./configure -setup*
- *./configure -compiler intel -speed fast*
- Well, we have also a nice installation guide..

Installation 2

Installation:

- *make*
- *make doc*

Reconfiguring

- *make distclean; ./configure [flags]; make*
- edit *Symbols* file

Maintenance

- molcas command
- Making a copy: *molcas copy /destination*
- updating molcas
  - Service Pack
  - manual: *molcas getpatch*
  - manual (without Internet connection): *molcas getpatch -f*
- Packages
- Verification: *molcas verify*

HowTo run Molcas

- Multiple installations of Molcas
  - case 1: Current directory is MOLCAS
  - case 2: $MOLCAS$ environment is set
  - case 3: latest molcas installation
- Environment variables:
  - $WorkDir$ - scratch directory
  - $Project$ - Project name
  - $MOLCAS\_PRINT$ - print levels
- Running Molcas
  - $molcas\ input\_file > log\ 2 > \&1$
  - $molcas\ seward\ seward\_input\_file > log\ 2 > \&1$
  - parallel run: $CPUS = 2;\ export\ CPUS;\ molcas\ input\_file$

Running molcas, example

```
Project=my
WorkDir=/tmp/$Project
mkdir $WorkDir
molcas $Project.input >out 2>err
```

Help!!

- on-line documentation
- molcas help command
  - *molcas help*
  - *molcas help scf*
  - *molcas help scf aufbau*
  - *molcas help -t b3lyp*
- Solving problems
  - Molcas user's billboard
  - Bug report system
  - direct mail: *molcas@teokem.lu.se*

Understanding molcas input

- Native molcas input
- Enhanced Molcas Input Language (EMIL)
  - Old molcas input is a subset of EMIL
  - Comment lines: $*$ at the beginning of line; /* */ multiline comments.
  - $\&MODULE$ is a call of module $MODULE$ (old style $\&MOLCAS\&END$ is also valid)
  - No need to end input for a module (old style $EndofInput$ is allowed)
  - a line started from ! is a UNIX command (Note an exception: $!export$ is illegal)
  - strings $\$VAR$ replaced by the value of VAR
  - Short notation: ; and $=$ replaced by a new line
  - EMIL commands started from $>$ sign ($>>>>>>>>>>> COMMAND <<<<<<<<<<<<<<$)

EMIL commands

- loop

```
>>>> Do While <<<<
....
>>>> EndDo    <<<<
```

- iteration counter

```
>>>> If ( Iter = 1 ) <<<<
....
>>>> EndIf    <<<<
```

- Export

```
>>>>Export VAR=VALUE
```

- include file

```
>>>>Include filename
```

coord command

```
>>>>coord file.xyz basis basis_type [nosym]
```

- file.xyz - is a standard xyz file:
  first line: Number of atoms
  second line: comments [angstrom, bohr]
  next lines: Atom name, 3 cartesian coords
- Basis type: STO-3G, ANO-S-MB, ANO-L-VDZ, etc.
  It is possible to specify: H=STO-3G O=6-31G
- nosym: special flag meaning- No Symmetry.
  the default - use maximum possible symmetry (D_{2h} subgroup)

EMIL: input example

```
/* this is an example of geometry optimization of
   water molecule using DFT */
>>>>Do While
&SEWARD
>> coord water.xyz basis ANO-S-MB
&SCF
KSDFT=B3LYP
&ALASKA; &SLAPAF
>>>> EndDo
```

The SEWARD input:

In input to SEWARD, the keywords Douglas-Kroll, AMFI, and Finite Nucleus are used for relativistic calculations. In addition, the ANO-RCC basis set should be used.

(Exception: Effective core basis sets.)

```
 &SEWARD  &END
Title
 Iodine atom
Symmetry
  XYZ
Basis set
I.ano-rcc...7s6p4d2f1g.
 I   0.00  0.00  0.00
End of basis
AMFI
Douglas-Kroll
End of Input
```

Calculations involving spin-orbit interactions should be done with low symmetry, preferably none at all or C_i. The ANO-RCC basis set is designed for such calculations, and are available for all the periodic table. The AMFI integrals are available for most common basis sets, including many ECPs. The Douglas-Kroll integrals are computed using, internally in SEWARD, the given basis set, uncontracted. This is useless for e.g. STO-3G basis sets.

The RASSCF input:

This RASSCF input is for computing the ground state of the iodine atom, which is a degenerate state ($^2P^o$) with three components. The calculation was done using $C_i$symmetry, so these states belong to irrep nr 2 (odd).

```
 &RASSCF &END
Title
 Iodine atom
Spin
 2
Symmetry
 2
Nactel
 5 0 0
Inactive
 15  9
Ras2
  0  3
CiRoot
 3 3 1
End of Input
```

'Spin 2' means this is a doublet, ' Symmetry 2' for irrep nr 2.
The Nactel keyword means 'Nr of active electrons, and three numbers are given: Nr of active electrons, Max nr of holes in RAS1, and Max nr of electrons in RAS3 orbitals, thus: 5 0 0. In this calculation, only the 6p orbitals are active.
Finally, the CIRoots keyword has in this case an extra number '1' at the end: 3 3 1 then means that we are taking a straight average of the three lowest states, and orbitals are optimized for that average energy.

The CASPT2 input:

For subsequent use in SO-RASSI, the RASSCF wavefunction 'interphase file' JOBIPH is used. The SO-RASSI will compute the Hamiltonian interaction matrix elements for the RASSCF wave functions, and include the spin-orbit terms. However, for better accuracy, the Hamiltonian may be 'dressed' with the contributions from neglected dynamic correlation. This is done by letting CASPT2 produce intermediate 'JOBMIX' files.

```
 &CASPT2 &END
Title
 Iodine atom
Frozen
  3  6
MultiState
  3
  1  2  3
End of Input
```

As usual, we wish to avoid complications that arise from attempts to correlate the core, so a number of orbitals are frozen. Molcas can automatically select a 'decent' core, but we may choose to do it manually.

The multistate input: Number of RASSCF states to dress by CASPT2 (3), and the serial number of these states (The three lowest: 1,2,3).

The CASPT2 program will automatically produce a new interphase file, called JOBMIX. Several such files may be used by the RASSI program.

The RASSI input:

The following RASSI input is actually from another calculation, showing the input for several JOBIPH or JOBMIX files:

```
!ln -fs T.Job JOB001
!ln -fs S.Job JOB002
 &RASSI &END
Nr of JobIph
 2  3  3
 1 2 3
 1 2 3
Spin
EJob
Omega
End of Input
```

The JOBMIX files are linked using soft links named JOB001, etc.
The Nr of JobIphs keyword is followed by: The number of JOBMIX files (2), The number of states to pick from each of them (3,3), and the the serial numbers of these states for the first file (The lowest three, 1,2,3), then those from the second file, etc.
EJob implies that energies are taken from the JOBMIX files instead of being recomputed.
Omega implies that spin-orbit states are annotated with their Omega quantum numbers, appropriate for linear molecules.

GUI

- molden
  - Z-matrix editor
  - molecular orbitals ($\$Project.ScfOrb.molden$)
  - geometry optimization ($\$Project.geo.molden$)
  - vibrations ($\$Project.freq.molden$)
  - problems: external program; X-windows only; slow
- Molcas GUI (yattagawa)
  - under development.
- Gabedit
  http://lasim.univ-lyon1.fr/allouche/gabedit/

Molcas Grid Viewer (gv)

- OpenGL/glut based code.
- integrated with yattagawa
- visualization of coordinates:
  molcas gv -z file.xyz
- simple editing of coordinates
- visualization of densities and orbitals
  (module GRID_IT produces $\$Project.M2Msi$ file)
- visualization of geometry optimization and vibrations is under construction