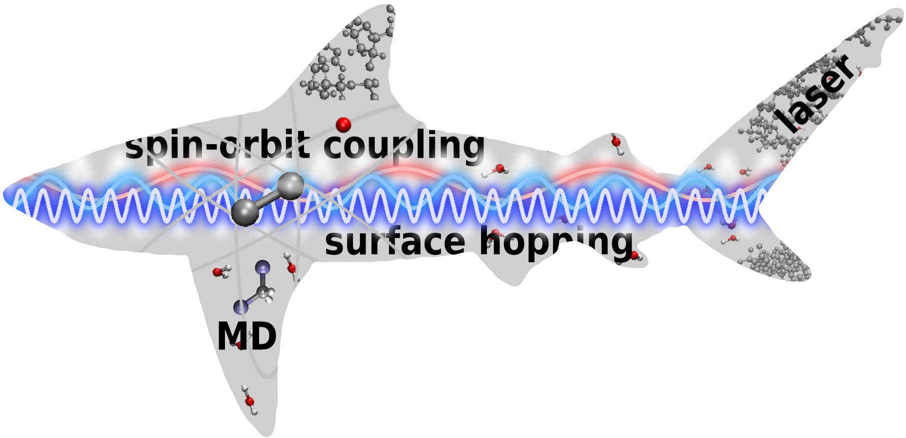


Wavefunction overlaps and Dyson orbitals by SHARC

Manual

AG González
Institute of Theoretical Chemistry
University of Vienna, Austria

Vienna, December 1, 2016

Contact:

AG González
Institute of Theoretical Chemistry, University of Vienna
Währinger Straße 17
1090 Vienna, Austria

Website: sharc-md.org
Email: sharc@univie.ac.at

Felix Plasser
Institute of Theoretical Chemistry, University of Vienna
Währinger Straße 17
1090 Vienna, Austria

Email: felix.plasser@univie.ac.at

Please, contact us if you have any questions or suggestions!

Contents

1 Introduction

1.1 Capabilities

The `wfoverlap.x` program provides an efficient and general algorithm [1] for the computation of wavefunction overlaps and Dyson orbitals. Wavefunction overlaps can be computed between any pair of wavefunctions given in a Slater determinant expansion as long as they contain the same number of α- and β-electrons. It is possible to vary the geometry, the basis set, the molecular orbitals, and the wavefunction expansion between the calculations.

The code provides flexible interfaces to various quantum chemistry programs that can be easily extended. Existing interfaces:

- Columbus 7: SA-MCSCF, MR-CISD
- Molcas 8: SA-RASSCF
- ADF: TD-DFT
- Turbomole: ADC(2), CC2

1.2 References

The following reference should be cited when using the wavefunction overlap code

- [1] F. Plasser, M. Ruckenbauer, S. Mai, M. Oppel, P. Marquetand, L. González: “Efficient and Flexible Computation of Many-Electron Wavefunction Overlaps”. *J. Chem. Theory Comput.*, **12**, 1207 (2016).

When computing Dyson orbitals, please cite

- [2] M. Ruckenbauer, S. Mai, P. Marquetand, L. González: “Revealing deactivation pathways hidden in time-resolved photoelectron spectra” (submitted for publication).

In the case of using the code to compare wavefunctions constructed at different levels of theory

- [3] F. Plasser, L. González: "Unambiguous Comparison of Many-Electron Wavefunctions Through Their Overlaps". *J. Chem. Phys.*, **in print** (2016).

1.3 Authors

The wavefunction overlap code was developed by Felix Plasser, Matthias Ruckenbauer, Sebastian Mai, Markus Oppel, Philipp Marquetand, and Leticia González at the Institute for Theoretical Chemistry, University of Vienna. The ADF interface was written by Andrew Atkins.

2 Installation

2.1 How to obtain

The overlap code is obtained from the SHARC homepage www.sharc-md.org in the same fashion as the main SHARC package. In the Download section, register with your e-mail address and affiliation. You will receive a download link to the stated e-mail adress. Clicking on the link in the email will download the archive file containing the overlap package. Note that the link is active only for 24 h and the number of downloads is limited.

Note that you must accept the "Terms of Use" in order to download the program. These are also found in the file COPYRIGHT of the distribution.

2.2 Getting started

Start by unpacking the downloaded file

```
tar -xf wfoverlap_1.0.tgz
```

For convenience, set the following two environment variables

```
export OVDIR=.../wfoverlap_1.0

export PATH=$PATH:$OVDIR/scripts:$OVDIR/bin
```

2.3 Precompiled executables

The overlap code is distributed with precompiled OpenMP parallel executables. If you wish to use those, no further steps are necessary and you can proceed to testing the implementation.

```
ovl_test.bash $OVDIR
```

If everything worked, you should see the following output

```
================================================

 *** All tests finished (number of errors: 0)

================================================
```

2.4 Manual installation

For the manual installation you need a working Fortran90 compatible compiler (Intel's ifort is recommended), some resonably fast BLAS/LAPACK libraries (Intel's MKL is recommended, although atlas is also fine).

Optionally, with a working Columbus Installation you can install the Columbus bindings, which will allow direct reading of SIFS integral files generated by DALTON. To use this option, it is necessary to use the `read_dalton.o` object file. Molcas/Seward integral files can be read by linking with the Columbus/Molcas interface. Link against `read_molcas.o` for this purpose.

To compile the source code, edit the Makefile to adjust it to your Fortran compiler and BLAS/LAPACK installation. The location of your Columbus installation has to be set via the enviroment variable `$COLUMBUS`.

Issuing the command:

```
make
```

will compile the source and create the binaries.

```
make test
```

will run a couple of tests to check if the program is working correctly. Tests requiring the Columbus and Molcas installations will only be run if the `$COLUMBUS` and `$MOLCAS` variables are set.

If you are unable to link against Columbus and/or Molcas, simply call

```
make  wfoverlap_ascii.x
```

to compile a minimal version of the CI Overlap program that only reads ASCII files.

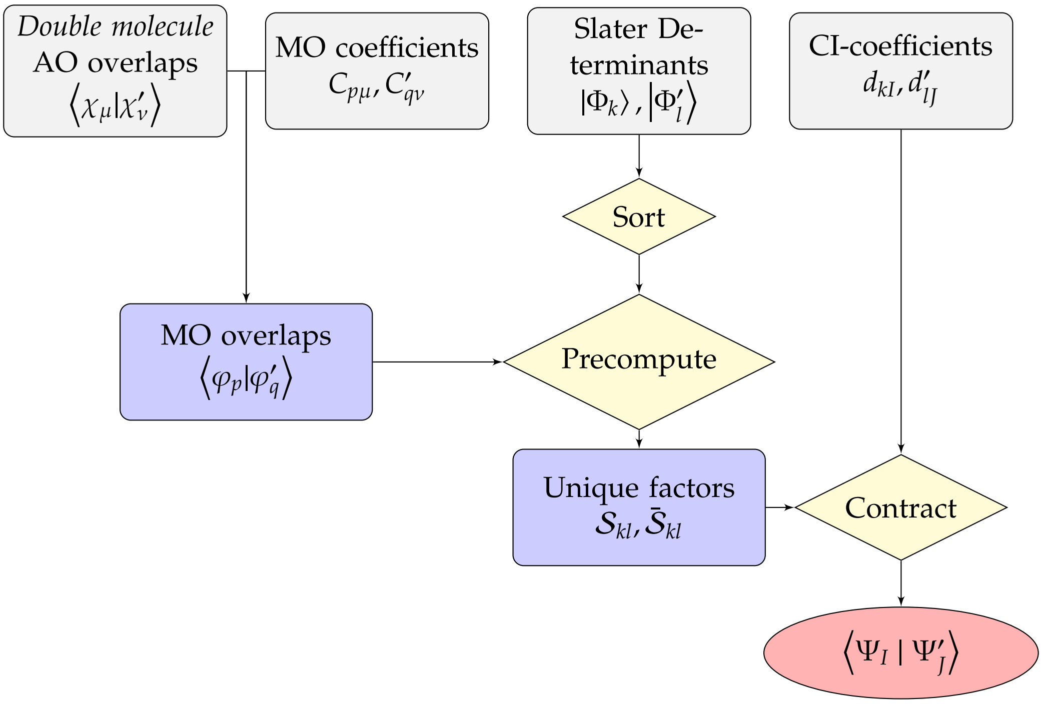
3 Execution

3.1 Workflow

The workflow of the overlap program is shown in Figure 3.1. Four pieces of input, as shown on top, have to be given:

- Overlaps between the two sets of AOs used to construct the bra and ket wavefunctions,
- MO coefficients of the bra and ket wavefunctions,
- information about the Slater determinants,
- the corresponding CI coefficients.

Two main intermediates are computed, the MO overlaps and the unique factors $\mathcal{S}_{kl}, \bar{\mathcal{S}}_{kl}$ where the latter may require significant amounts of memory to be stored.



Double molecule AO overlaps $\langle \chi_\mu | \chi'_\nu \rangle$
MO coefficients $C_{p\mu}, C'_{q\nu}$
Slater Determinants $|\Phi_k\rangle, |\Phi'_l\rangle$
CI-coefficients d_{kI}, d'_{lJ}
Sort
MO overlaps $\langle \varphi_p | \varphi'_q \rangle$
Precompute
Unique factors $\mathcal{S}_{kl}, \bar{\mathcal{S}}_{kl}$
Contract
$\langle \Psi_I \mid \Psi'_J \rangle$

Figure 3.1: Workflow of the wavefunction overlap program.

3.2 Calling the program

The main program is called in the following form

```
wfoverlap.x [-m <mem=1000>] [-f <input_file=cioverlap.input>]
```

with the command line options

- `-m` - amount of memory in MB
- `-f` - input file

Example:

```
wfoverlap.x -m 2000 -f wfov.in
```

A list of all input keywords is given in Table 3.1.

If the number of electrons in the bra and ket wavefunctions is the same, **overlaps** are computed. If they differ by 1 **Dyson orbitals** are computed.

The amount of **memory** given is a decisive factor for the performance of the code.

Depending on the amount of memory, one of three different modes is chosen:

1. All $\mathcal{S}_{kl}$ and $\bar{\mathcal{S}}_{kl}$ terms are kept in core

2. Only the $\mathcal{S}_{kl}$ factors are kept in core. This is seen by

   ```
   Allocation of Q block overlap matrix failed.
   - Using on-the-fly algorithm for Q determinants.
   ```

   This mode is generally as efficient as 1. but shows somewhat worse parallel scaling.

3. Not even all $\mathcal{S}_{kl}$ factors can be stored

   ```
   Only 437 out of 886 columns of the P_ovl matrix are stored in memory (3 MB)!
   Increase the amount of memory to improve the efficiency.
   ```

   This mode is significantly slower than 1. and 2. and should be avoided by increasing the amount of memory given.

Table 3.1: List of keywords given in the input file. The `a_mo, b_mo, a_det, b_det` keywords are mandatory, all others are optional.

Keyword	Default	Description
`a_mo`	-	MO coefficient file (bra)
`b_mo`	-	MO coefficient file (ket)
`a_mo_read`	0	Format for the MO coefficients (bra): 0 - Columbus, 1 - Molcas, 2 - Turbomole
`b_mo_read`	0	Format for the MO coefficients (ket)
`a_det`	-	Determinant file (bra)
`b_det`	-	Determinant file (ket)
`ncore`	0	Number of discarded core orbitals
`ndocc`	0	Number of doubly occupied orbitals (only for Dyson orbital calculations)
`mix_aoovl`	`S_mix/ONEINT/aoints` for `ao_read=0/1/2`	AO overlap file
`ao_read`	0	Format for overlap integrals: 0 - ASCII, 1 - Molcas, 2 - Columbus/SIFS, -1 - Compute by inversion of MO coefficient matrx
`same_aos`	.false.	If both calculations were performed with the same set of AOs (specify only for `ao_read=1/2`)
`nao_a`	*automatic*	Number of bra AOs (specify only if different from ket AOs)
`nao_b`	*automatic*	Number of ket AOs (specify only if different from bra AOs)
`moprint`	0	Print Dyson orbitals: 1 - coefficients, 2 - also as Jmol script
`force_direct_dets`	.false.	Compute $\mathcal{S}_{kl}$ terms directly (turn off "superblocks")
`force_noprecalc`	.false.	Do not precalculate the $\bar{\mathcal{S}}_{kl}$ factors

An example input file is shown below

```
a_mo=mocoef_a
b_mo=mocoef_b
a_det=dets_a
b_det=dets_b
ao_read=2
same_aos
```

3.3 Input data

Typically, three types of input are provided: AO overlaps, MO coefficients, and a combined file with determinant information and CI coefficients (cf. Figure 3.1). The file formats are explained here. Auxiliary scripts to create these files are explained in Section 4.1.

3.3.1 AO overlaps

The mixed AO overlaps $\langle \chi_\mu | \chi'_\nu \rangle$ between the AOs used to expand the bra and ket wavefunctions are required. They are in general created by a "double molecule" calculation, i.e. an AO integral calculation where every atom is found twice in the input file.

The native format (`ao_read=0`) is a simple ASCII file containing the relevant off-diagonal block of the mixed AO overlap matrix, e.g.

```
   7 7
     9.97108464676133E-001    2.36720699813181E-001   ...
     2.36720699813181E-001    9.99919192433940E-001   ...
     1.00147985713321E-002    6.52340422397770E-003   ...
 ...
```

In addition, Molcas (`ao_read=1`) and Columbus/SIFS (`ao_read=2`) files can be read in binary form.

If the same AOs are used for the bra and ket wavefunctions and the MO coefficient matrix is square, it is possible to reconstruct the overlaps by inversion of the MO coefficient matrix (`ao_read=-1`). In this case it is not necessary to supply a `mix_aoovl` file.

3.3.2 MO coefficients

MO coefficients of the bra and ket wavefunctions can usually be read in directly in the form written by the quantum chemistry program. The supported options for `a_mo_read` and `b_mo_read` are

0 - Columbus
1 - Molcas lumorb
2 - Turbomole

3.3.3 Slater determinants and CI coefficients

Slater determinants and CI coefficients are currently supplied by an ASCII file of the form

```
3 7 168
dddddee   0.979083342437    0.979083342437   -0.122637656388
ddddabe  -0.094807515471   -0.094807515471   -0.663224542162
ddddbae   0.094807515471    0.094807515471    0.663224542162
...
```

The first line specifies the number of states, the number of MOs, and the number of determinants in the file. Every subsequent line gives the determinant string and the corresponding CI coefficients for the different states. The following symbols are used in the determinant string:

d - doubly occupied
a - singly occupied (α)
b - singly occupied (β)
e - empty

3.4 Output

A typical output is shown below

```
 Overlap matrix <PsiA_i|PsiB_j>
                |PsiB  1>     |PsiB  2>
<PsiA  1|    0.5162656622 -0.2040109070
<PsiA  2|   -0.2167057391 -0.5266552021

 Renormalized overlap matrix <PsiA_i|PsiB_j>
                |PsiB  1>     |PsiB  2>
<PsiA  1|    0.5162656622 -0.2040109070
<PsiA  2|   -0.2167057391 -0.5266552021

 Performing Lowdin orthonormalization by SVD...

 Orthonormalized overlap matrix <PsiA_i|PsiB_j>
                |PsiB  1>     |PsiB  2>
<PsiA  1|    0.9273847015 -0.3741090956
<PsiA  2|   -0.3741090956 -0.9273847015
```

- `Overlap matrix` gives the raw overlap values

  $$\left\langle \Psi_I \middle| \Psi'_J \right\rangle$$

  of the wavefunctions supplied.
- `Renormalized overlap matrix` gives the renormalized overlap values

  $$\frac{\left\langle \Psi_I \middle| \Psi'_J \right\rangle}{\|\Psi_I\| \left\|\Psi_J\right\|'}$$

  relevant in the case of wavefunction truncation.
- The `Orthonormalized overlap matrix` is constructed according to a procedure described in more detail in Ref. [1].

4 Auxilliary Scripts

4.1 Interfaces

A number of auxiliary scripts providing interfaces to third party programs are available.

- `ADF_CIovlp.py`
  ADF: Parse the TAPE21 file

```
ADF_CIovlp.py <TAPE21>
```

- `AO_overl.py`
  ADF: Extract AO overlaps from the TAPE15 file

```
AO_overl.py <TAPE15>
```

- `dalton_double-mol.py`
  Columbus: Create double-molecule AO calculation with Dalton

```
dalton_double-mol.py <dir1> <dir2> [run]
```

- `read_civfl.py`
  Columbus: Read `civfl` file and convert to ASCII form.

```
read_civfl.py <nstate> [<maxsqnorm>]
   command line options: -debug, -ms <ms>, -m <mem (MB)>, -o <det_file>
```

- `read_rassi.py`
  Molcas: Read standard output of `rassi` program and convert to the format required by the wavefunction overlap program

```
read_rassi.py <molcas_log> <mult> [<det_file>]
```

- `seward_double-mol.py`
  Molcas: Create double-molecule AO calculation with `seward`

```
seward_double-mol.py <dir1> <dir2> [run]
```

4.2 Graphical output

The pie_chart.py script can be used to create pie charts of the squared overlap terms [3].

```
pie_chart.py <wfov_out>
  Main command line options:
  -o : format of the output file
  -f : font size
  -r : analyze renormalized overlap matrix
  -l : analyze Lowdin orthonormalized overlap matrix
  -c : colors
```

Example

```
pie_chart.py wfov.out -o eps -f 45 -r -c 'lightblue red green yellow'
```

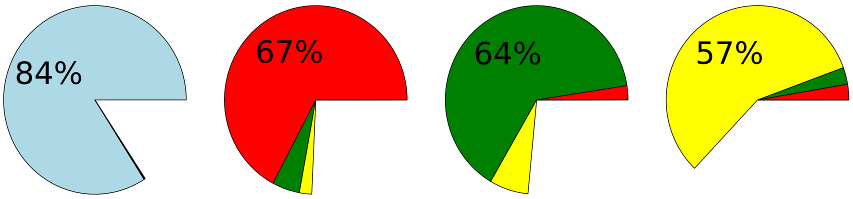


Figure 4.1: Examplary pie charts created by pie_chart.py.

5 Example Jobs

To find more examples of inputs for wavefunction overlap and Dyson orbital computations check the $OVDIR/test_jobs directory. Various input files are found in the respective IN_FILES subdirectories. Below, a list of all test directories and input files is given

- Adenine_fc
  Calculations on adenine using frozen core orbitals
  - ciov_all.in
    Overlap considering all MOs
  - ciov_fc.in
    Overlap using 10 discarded core MOs
  - ciov_fc_direct.in
    Overlap using 10 discarded core MOs and direct determinant computation
  - dyson.in
    Dyson orbitals using 10 discarded core MOs and 25 doubly occupied MOs

- CH2_CISD - ciov_bin.in
  Example of reading binary Molcas integral file

- CH2_dalton
  Example of reading binary Columbus/Dalton integral file
  - cioverlap.input
    Overlap
  - dyson.in
    Dyson orbitals

- CH2_doublet - cioverlap.input
  Overlap in the case of a charged system

- CH2_dyson
  Various Dyson orbital computations
  - dyson[1-4].in
    Check consistency for varying bra/ket and m_s quantum number
  - dyson4_inv.in
    Use MO matrix inversion (ao_read=-1)
  - dyson_quart_0.5.in
    Check that singlet/quartet Dyson orbitals vanish

- CH2_triplet
  Triplet overlaps for varying m_s quantum number

- `CH2_ricc2 - ciovl.in`
  Example for Turbomole `ricc2` using `a_mo_read=2`

- `water`
  Some simple applications on water
  - `ciovl.in`
    Run with default values
  - `ciovl.in.direct`
    Direct determinant computation
  - `ciovl.in.lumorb`
    Orbitals read in Molcas `lumorb` format (`a_mo_read=1`)

- `water_molcas`
  Example for creating Molcas double molecule files (`seward_double-mol.py`) and reading them (`ao_read=1`)
  - `ciov.in`
    Standard computation
  - `ciov_proj.in`
    Wavefunction projection between 6-31G* and 6-31G basis sets (application of the `nao_a, nao_b` keywords)

Bibliography

[1] F. Plasser, M. Ruckenbauer, S. Mai, M. Oppel, P. Marquetand, L. González: "Efficient and Flexible Computation of Many-Electron Wavefunction Overlaps". *J. Chem. Theory Comput.*, **12**, 1207 (2016).

[2] M. Ruckenbauer, S. Mai, P. Marquetand, L. González: "Revealing deactivation pathways hidden in time-resolved photoelectron spectra" (submitted for publication).

[3] F. Plasser, L. González: "Unambiguous Comparison of Many-Electron Wavefunctions Through Their Overlaps". *J. Chem. Phys.*, **in print** (2016).