

User's manual of ADPACK Ver. 2.1

Taisuke Ozaki

Japan Advanced Institute of Science and Technology (JAIST),
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan

Contributors

T. Ozaki (JAIST),
H. Kino (NIMS)

June 20, 2008

Contents

1	About ADPACK	2
2	Installation	3
3	Test calculation	3
4	Input file	6
5	All electron calculation	15
6	Making of pseudopotential	17
6.1	Example	17
6.2	Cutoff radius	19
6.3	Pseudopotentials for unbound states	20
6.4	Separable form	20
6.5	Logarithmic derivative of wave function	21
6.6	Ghost states	21
6.7	Partial core correction	22
6.8	Restart	23
7	Relativistic calculation	24
7.1	All electron calculation	24
7.2	Enhancement or depletion of a spin-orbit coupling	24
8	Making of pseudo-atomic orbitals	26
9	Virtual atom with fractional nuclear charge	27
10	Output files	28
11	Templates of the input files	28
12	Others	28

1 About ADPACK

ADPACK (Atomic Density functional program PACKage) is a program package for atomic density functional calculations, in which either Schrödinger or Dirac equation under a spherical atomic potential is numerically solved within a local density approximation (LDA) [1] or a generalized gradient approximation (GGA) [2] to the exchange-correlation energy. The distribution of this program package and the source codes follow the practice of the GNU General Public Licence (GPL) [17]. The program package can be freely downloadable from <http://www.openmx-square.org/>.

Current features of ADPACK are as follows:

- All electron calculation by the Schrödinger or Dirac equation
- LDA and GGA
- Pseudopotential generation by Troullier and Martine (TM) [4] and Bachelet, Hamann, and Schluter (BHS) [3] schemes
- Pseudopotential generation for unbound states by Hamann's scheme [7]
- Kleinman and Bylander (KB) separable pseudopotentials [5]
- Blöchl multiple projectors [6]
- Partial core correction to exchange-correlation energy [10]
- Logarithmic derivatives of wave functions [12]
- Detection of ghost states in separable pseudopotentials [13]
- Scalar relativistic treatment [14]
- Fully relativistic treatment with spin-orbit coupling [15, 3]
- Generation of pseudo-atomic orbitals under a confinement potential [11]
- Analysis of wave functions
- Analysis of electron density
- Database of pseudopotentials and pseudo-atomic orbitals

The norm-conserving pseudopotentials and pseudo-atomic orbitals generated by ADPACK could be input data to OpenMX which is a program package for molecules and solids. It is expected that ADPACK is available on a standard unix-like environment such as unix, linux, and cygwin [16], since the code is written in a standard C language. Database of pseudopotentials and pseudo-atomic orbitals is also found in the above website.

2 Installation

After downloading `adpack2.0.tar.gz`, decompress it as follows:

```
% tar zxvf adpack2.0.tar.gz
```

When it is completed, you can find three directories (`source`, `work`, `idensity`) under the directory, `adpack2.0`. Move to the directory, `source`, and then install as follows:

```
% make install
```

When the compile is completed normally, then you can find the executable file, `adpack`, in the directory, `work`. To make the execution of ADPACK efficient, you can change a compiler and compile options appropriate for your computational environment, which can generate an optimized executable file. Then, it might be made by specifying `CC` in the makefile which exists in directory, `source`. The default for the specification of `CC` is as follows:

```
CC = gcc -O3
```

However, it is highly recommended to use the gnu C compiler (`gcc`) for the numerical stability.

3 Test calculation

If the installation is completed normally, move to the directory '`work`', and then you can perform the program, `adpack`, using an input file, `C.inp` as follows:

```
% adpack C.inp
```

The test input file, `C.inp`, is for performing the SCF calculation of a carbon atom. The calculation is performed in only several seconds by a 2.4GHz Xeon machine, although it is dependent on a computer. When the calculation is completed normally, three files (`C0.aolog`, `C0.ao`, and `C0.aden`) are output to the directory, `work`. `C0.aolog` is the log file of the calculation which includes the contents of an input file, the convergence history in SCF steps, and the total energy decomposed to the contributions. A part of the file, `C0.aolog`, is shown below. It is found that the convergence is achieved by 8 SCF steps for the eigenvalues energy of a Kohn-Sham equation, `Eeigen`, and the norm of the difference between the input and output densities.

```
*****
```

```
SCF history in all electrons calculations
```

```
*****
```

```
SCF= 1 Eeigen=-21.1885511461333 (Hartree) NormRD= 0.0004422982883
SCF= 2 Eeigen=-21.3146245014854 (Hartree) NormRD= 0.0000372735309
SCF= 3 Eeigen=-21.3146503991147 (Hartree) NormRD= 0.0000371982068
SCF= 4 Eeigen=-21.3269108650127 (Hartree) NormRD= 0.0000068648722
```

```

SCF= 5 Eigen=-21.3258235985358 (Hartree) NormRD= 0.0000005715746
SCF= 6 Eigen=-21.3252551333262 (Hartree) NormRD= 0.0000000284064
SCF= 7 Eigen=-21.3251704880886 (Hartree) NormRD= 0.0000000011351
SCF= 8 Eigen=-21.3251315998899 (Hartree) NormRD= 0.0000000000345

```

The eigenvalues and the total energy, Etot, are also output in C0.alog.

```

*****
Eigenvalues (Hartree) in all electrons calculations
*****

```

```

n= 1 l= 0          -9.9608533995726
n= 2 l= 0          -0.5021796354376
n= 2 l= 1          -0.1995327649348

```

```

*****
Energies (Hartree) in all electrons calculations
*****

```

```

Eigen =      -21.3251315998899
Ekin   =       37.1494460559714
EHart  =       17.6374552935492
Exc    =       -4.7289869324025
Eec    =      -87.5230146517655
Etot   = Ekin + EHart + Exc + Eec
Etot   =      -37.4651002346473

```

The electron density $\rho(r)$ as a function of radius is output in a file, C0.aden. Figure 1(a) shows electron density of carbon atom stored in C0.aden. In the file, C0.aden, the first, second, third columns mean $\log(r)$, r , and the electron density, respectively. The order of data is also similar in the other files. The radial wave functions, shown in Fig. 1(b), are output in a file, C0.ao, in which they are listed in order of $\log(r)$, r , and the radial wave functions of $l=0$ for $n=1$. For $n=2$ or subsequent ones, radial wave functions are stored in the same order as that for $n=0$. However, note that the ingredients are output up to $l=n-1$ as follows:

```

n=1
log(r), r, l=0
.....
n=2
log(r), r, l=0, l=1
.....
n=3
log(r), r, l=0, l=1, l=2
.....

```

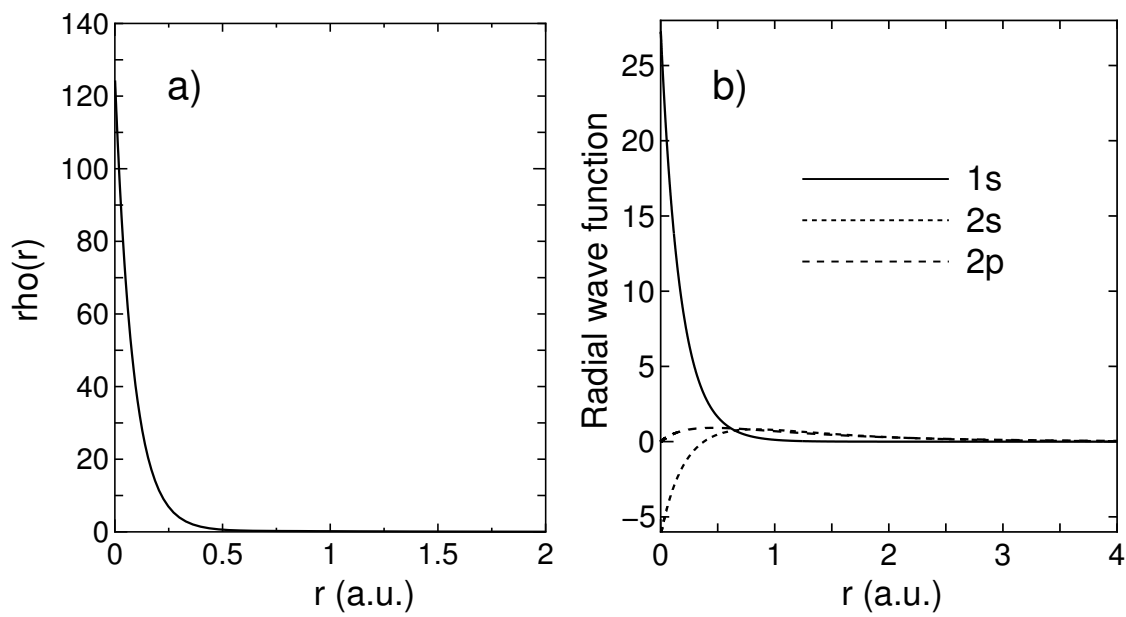


Figure 1: (a) Electron density of carbon atom, (b) Radial wave functions of carbon atom

4 Input file

An input file, C.inp, is shown below. This input file has a flexible data format (FDF), in which a parameter is given behind a keyword, the order of keywords is arbitrary, and a blank and a comment can also be described freely.

```
#
# File Name
#

System.CurrrentDir    ./          # default=./
System.Name           CO
Log.print             Off          # ON|OFF

System.UseRestartfile NO          # NO|YES, default=NO
System.Restartfile    CO_vps      # default=null

#
# Calculation type
#

eq.type               sch          # sch|sdirac|dirac
calc.type             ALL          # ALL|VPS|PAO
xc.type               LDA          # LDA|GGA

#
# Atom
#

AtomSpecies           6
max.ocupied.N         2
total.electron        6.0
valence.electron      4.0
<ocupied.electrons
 1  2.0
 2  2.0  2.0
ocupied.electrons>

#
# parameters for solving 1D-differential equations
#

grid.xmin             -7.0         # default=-7.0 rmin(a.u.)=exp(grid.xmin)
grid.xmax             2.5         # default= 2.5 rmax(a.u.)=exp(grid.xmax)
grid.num              6000        # default=4000
grid.num.output       2000        # default=2000

#
# SCF
#
```

```

scf.maxIter          40          # default=40
scf.Mixing.Type      simple      # Simple|GR-Pulay
scf.Init.Mixing.Weight 0.10      # default=0.300
scf.Min.Mixing.Weight 0.001     # default=0.001
scf.Max.Mixing.Weight 0.800     # default=0.800
scf.Mixing.History   7          # default=5
scf.Mixing.StartPulay 9         # default=6
scf.criterion        1.0e-10    # default=1.0e-9

#
# Pseudopotential, cutoff (A.U.)
#

vps.type             TM          # BHS|TM
number.vps           2
<pseudo.NandL
  0  2  0  1.50
  1  2  1  1.62
pseudo.NandL>
Blochl.projector.num 1          # default=1 which means KB-form
local.type            polynomial # Simple|Polynomial
local.part.vps        1          # default=0
local.cutoff          1.50       # default=smallest_cutoff_vps
local.origin.ratio    4.00       # default=3.0
log.deriv.RadF.calc   on         # ON|OFF
log.deriv.MinE        -3.0       # default=-3.0 (Hartree)
log.deriv.MaxE        2.0        # default= 2.0 (Hartree)
log.deriv.num         50         # default=50
<log.deriv.R
  0  2.2
  1  2.4
log.deriv.R>
ghost.check           off        # ON|OFF

#
# Core electron density for partial core correction
# pcc.ratio=rho_core/rho_V,
# pcc.ratio.origin = rho_core(origin)/rho_core(ip)
#

charge.pcc.calc       on         # ON|OFF
pcc.ratio              0.25      # default=1.0
pcc.ratio.origin       5.00      # default=6.0

#
# Pseudo atomic orbitals
#

maxL.pao               2         # default=2
num.pao                5         # default=7

```

<code>radial.cutoff.pao</code>	5.5	# default=5.0 (Bohr)
<code>height.of.wall</code>	20000.0	# default=4000.0 (Hartree)
<code>rising.edge</code>	0.2	# default=0.5(Bohr), r1=rc-rising.edge
<code>search.LowerE</code>	-3.000	# default=-3.000 (Hartree)
<code>search.UpperE</code>	20.000	# default=20.000 (Hartree)
<code>num.of.partition</code>	300	# default=300
<code>matching.point.ratio</code>	0.67	# default=0.67

The specification of each keyword is as follows.

Common keywords for `calc.type=ALL|VPS|PAO`

System.CurrentDir

The output directory of output files.

System.Name

The file name of output files.

Log.print

The informations in the middle of calculation are output to the standard output. Specify `Log.print=ON` when outputting, or `Log.print=OFF` when non-outputting. This keyword is used for developers.

System.UseRestartfile

For an atom with a large atomic number, all electron calculation requires a considerable computational time. So, it is needed to reduce the computational time when optimal cutoff radii of pseudopotentials are determined in a trial and error. If the keyword, `System.UseRestartfile`, is specified as `YES`, a restart file which contains informations of all electron calculation is used in order to skip all electron calculation. If there is no restart file, a restart file is generated in case of `System.UseRestartfile=YES`.

System.Restartfile

If `System.UseRestartfile=YES`, then the name specified by the keyword, `System.Restartfile`, is referred to as a restart file.

eq.type

The keyword, `eq.type`, specifies the type of equation. For the non-relativistic Kohn-Sham equation, please specify 'sch'. On the other hand, for the scalar and full relativistic Kohn-Sham equation, please specify 'sdirac' and 'dirac', respectively.

calc.type

The keyword specifies a calculation type. The SCF calculation for all electrons (`ALL`), the generation of pseudopotentials (`VPS`), or the generation of pseudo-atomic orbitals with a confinement potential (`PAO`) are available.

xc.type

Please choose the approximate method (`LDA` or `GGA`) used for an exchange correlation energy.

AtomSpecies

Give the atomic number.

max.ocupied.N

Give the maximum number of the principal quantum number, n, for occupied electrons.

total.electron

Give the total number of electrons in an atom. It is also possible to give the number of electrons corresponding to not only a neutral atom, but also a positive or negative charged atom. However, note that it becomes difficult to achieve the convergence in the SCF calculation for a negative atom (there are more electrons than atomic number), since wave functions tend to be expanded or unbound spatially.

valence.electron

Give the number of electrons of valence electrons.

ocupied.electrons

Give the number of electrons occupied in each orbital. As seen in C.inp, when 1s, 2s, and 2p orbitals of a carbon atom are occupied by two electrons in consideration of the spin degeneracy, respectively, they are specified as follows:

```
<ocupied.electrons
  1  2.0
  2  2.0  2.0
ocupied.electrons>
```

The beginning of the description must be <ocupied.electrons, and the last of the description must be ocupied.electrons>.

grid.xmin

The radial Kohn-Sham equation is solved numerically by the Hamming method [8] from both a radial point r_{\min} near the origin and a distant radial point r_{\max} (a.u.). Here, a radial point r_{\min} near the origin is specified by the keyword, grid.xmin. Note that there is a relation, $r_{\min}(\text{a.u.})=\exp(\text{grid.xmin})$.

grid.xmax

The keyword, grid.xmax, specifies a distant radial point r_{\max} (a.u.) which begins to solve a Kohn-Sham equation. As well as grid.xmin, note that $r_{\max}(\text{a.u.})=\exp(\text{grid.xmax})$. The selection of a suitable grid.xmax is dependent on an atom. For an atom with only localized electrons such as carbon and oxygen, the use of about 2.5 (a.u.) is recommended as grid.xmax. In case of an atom such as Na, Ti, Fe with delocalized electrons, the use of about 3.0 (a.u.) or more is recommended as grid.xmax. Moreover, it is required for grid.xmax to use a large value, when a atom is charged negatively.

grid.num

The radial direction r is divided to solve the radial Kohn-Sham equation by the Hamming method. The number of division is specified by grid.num. The actual mesh division is done for x ($=\log(r)$) as $dx=(\text{grid.xmax}-\text{grid.xmin})/\text{grid.num}$ rather than for r to cope with large variations near the origin of potential and wave functions.

grid.num.output

It is possible to change the number of grids for r in output files by the keyword, grid.num.output, although the actual calculation is performed using grid.num.

scf.maxIter

The maximum number of SCF iterations is specified by the keyword, `scf.maxIter`. The SCF loop is terminated at the number specified by `scf.maxIter` even if the convergence criterion is not satisfied.

scf.Mixing.Type

A mixing method for the electron density to generate an input electron density at the next SCF step is specified by keyword, `scf.Init.Mixing.Type`. You can choose either the simple mixing method (Simple) or GR-Pulay method (Guaranteed-Reduction Pulay method) [9]. The simple mixing method used here is modified to accelerate the convergence by referring to a convergence history. So, the use of the simple mixing method is recommended because of its robustness.

scf.Init.Mixing.Weight

The keyword, `scf.Mixing.Weight`, gives an initial mixing weight used by the simple mixing method and the GR-Pulay method. The effective range is $0 < \text{scf.Mixing.Weight} < 1$.

scf.Min.Mixing.Weight

The keyword, `scf.Init.Mixing.Weight`, gives the lower limit of a mixing weight in the simple mixing method.

scf.Max.Mixing.Weight

The keyword, `scf.Max.Mixing.Weight`, gives the upper limit of a mixing weight in the simple mixing method.

scf.Mixing.History

In the GR-Pulay method, the input electron density at the next SCF step is calculated by making use of the output electron densities in the several previous SCF steps. The keyword, `scf.Mixing.History`, specifies the number of previous SCF steps which are taken into account for the calculation. For example, `scf.Mixing.History` is specified to be 3, and the SCF step is 6th. Then, the output electron density at 5, 4, and 3 SCF steps are taken into account.

scf.Mixing.StartPulay

The SCF step which starts the GR-Pulay method is specified by the keyword, `scf.Mixing.StartPulay`. The simple mixing method is employed in SCF steps before starting GR-Pulay method.

scf.criterion

The keyword, `scf.criterion`, specifies a convergence criterion for the SCF calculation. The SCF iteration is terminated when a condition, $\text{NormRD} < \text{scf.criterion}$, is satisfied, where a norm of the deviation between the input and output electron densities, NormRD , is defined by $4\pi \int_{r_{\min}}^{r_{\max}} (\rho_{\text{inp}}(r) - \rho_{\text{out}}(r))^2 r^2 dr$.

Specific keywords fo `calc.type=VPS|PAO`**vps.type**

When VPS is chosen for the keyword, `calc.type`, the keyword, `vps.type`, specifies a generation method of pseudopotentials. Either BHS [3] or TM [4] scheme is available.

number.vps

Give the total number of pseudopotentials that you want to calculate.

pseudo.NandL

The keyword, `pseudo.NandL`, specifies a set of a principal quantum number, N , and an angular momentum quantum number, L , of pseudopotentials corresponding to the number of potentials specified by the keyword, `number.vps`. For example, if `number.vps` is chosen to be 2 for a carbon atom, and the pseudopotentials for 2s and 2p orbitals are generated, then specify in the following way:

```
<pseudo.NandL
  0  2  0  1.3
  1  2  1  1.3
pseudo.NandL>
```

The first number specifies a line number beginning from zero line, which is used in the specification of the keyword, `local.part.vps`. In the second or third columns, a principal number and an angular momentum quantum number are given. The fourth column provides a cutoff radius (a.u.) for the generation of pseudopotentials. Although an optimum cutoff radius is determined so that the generated pseudopotential has a smooth shape without distinct kinks and a lot of nodes, however, the choice is made in a somewhat empirical way. The beginning of the description must be `<pseudo.NandL`, and the last of the description must be `pseudo.NandL>`.

Blochl.projector.num

The keyword, `Blochl.projector.num`, specifies the number of projectors for each L-component in separable pseudopotentials. If you specify 1 for `Blochl.projector.num`, this means the Kleinman and Bylander (KB) separable pseudopotentials. As the number of `Blochl.projector.num` increases, the separable pseudopotential converges the semilocal non-separable pseudopotential. We recommend you to use 2 or 3 for `Blochl.projector.num` in order to increase the transferability of the separable pseudopotential. We guess that you might consider the increase of computational efforts due to the increasing projectors. However, the matrix elements for the non-local part are evaluated outside the SCF loop. Therefore, the computational demands for the larger projectors are quite small.

local.type

The keyword, `local.type`, specifies a way for generating the local part of pseudopotentials. 'Simple' means that a l-component of pseudopotential, specified by the keyword (`local.part.vps`), is used as the local part. 'Polynomial' means that the local part for the inside of a cutoff radius is generated using a polynomial and that the outer part is proportional to $-1/r$. At the cutoff radius the two parts are connected so that up to third derivatives are continuous.

local.part.vps

When 'Simple' for the keyword, `local.type`, is used, the keyword, `local.part.vps`, specifies the local potential used in the generation of KB factorized pseudopotentials. In this specification, please choose the number of the first column in the specification of the keyword, `pseudo.NandL`.

local.cutoff

When 'Polynomial' is used for the keyword, `local.type`, the cutoff radius, r_{lc} (a.u.), at which a polynomial local part is connected to $-Z/r$, is specified by the keyword, `local.cutoff`.

local.origin.ratio

When 'Polynomial' is used for the keyword, `local.type`. The keyword, `local.origin.ratio`, specifies the

value of the local potential at the origin. It should be noted to be $V_L(0) = \text{local.origin.ratio} \times V_L(r_{1c})$.

log.der.RadF.calc

In case of 'calc.type=VPS', if you want to calculate the logarithmic derivatives of radial wave functions for the all electron potential, semilocal pseudopotentials, and separable pseudopotentials, then, please specify ON for the keyword, log.der.RadF.calc. If not so, please specify OFF. The calculated logarithmic derivatives are output to the file, *.ld0,*.ld1,..., where * means 'System.Name' you specified, the number attached to the last of the file extension 'ld' is the angular momentum number L. In these files, the first column is energy, the second (D_0), third (D_1), and fourth (D_2) columns are the logarithmic derivatives of radial wave functions for the all electron potential, the semilocal non-separable pseudopotential, and the separable pseudopotential, respectively. In addition to the output of logarithmic derivatives to the files, an useful quantities, I_0 and I_1 , are evaluated in order to discriminate the transferability of the separable pseudopotentials by

$$I_0 = \int_{\text{log.der.MinE}}^{\text{log.der.MaxE}} (D_0 - D_2)^2 dE$$

$$I_1 = \int_{\text{log.der.MinE}}^{\text{log.der.MaxE}} (D_1 - D_2)^2 dE$$

Ideally, the maximum transferability can be obtained when I_0 and I_1 are zero. So, it is desirable to make pseudopotentials with small I_0 and I_1 as much as possible. I_0 and I_1 are output on the standard output (your display).

log.der.MinE

In case of 'calc.type=VPS' and 'log.der.RadF.calc=ON', the keyword, log.der.MinE, gives the lower bound of energy (Hartree) used in the calculation of logarithmic derivatives of radial wave functions.

log.der.MaxE

In case of 'calc.type=VPS' and 'log.der.RadF.calc=ON', the keyword, log.der.MaxE, gives the upper bound of energy (Hartree) used in the calculation of logarithmic derivatives of radial wave functions.

log.der.R

In case of 'calc.type=VPS' and 'log.der.RadF.calc=ON', the keyword, log.der.R, gives the radius (a.u.) at which the logarithmic derivatives of radial wave functions are evaluated. If eq.type=sch or eq.type=dirac, the keyword, log.der.R, is specified for each angular momentum number L as follows:

```
<log.der.R
  0  2.2
  1  2.4
log.der.R>
```

The beginning of the description must be <log.der.R, and the last of the description must be log.der.R>. The first column is the angular momentum number L, and the second column is the radius at which the logarithmic derivatives of radial wave functions are evaluated. If eq.type=dirac, the third column is needed as follows:

```
<log.der.R
```

```
0 2.0 1.9
1 2.0 2.1
log.deriv.R>
```

where the second and third column give the radii at which the logarithmic derivatives of radial wave functions of $j = l + 1/2$ and $j = l - 1/2$ are evaluated, respectively.

ghost.check

In case of 'calc.type=VPS', if you want to check whether there are ghost states for the generated separable pseudopotentials, please specify ON for the keyword, ghost.check. If not so, please specify OFF for the keyword. The calculation result appears on the standard output (your display).

charge.pcc.calc

In DFT calculations of a molecule and a solid using pseudopotentials, if you include a partial core correction to the exchange-correlation energy, you are requested to calculate a partial core electron density. If you calculate the partial core electron density, please specify charge.pcc.calc=ON, or charge.pcc.calc=OFF.

pcc.ratio

The keyword, pcc.ratio, is a parameter in the calculation of a partial core electron density. The core electron density is approximated using a fourth order polynomial below the cutoff radius r_{pcc} at which the ratio ρ_c/ρ_v between the core electron density ρ_c and the valence electron density ρ_v becomes pcc.ratio.

pcc.ratio.origin

The keyword, pcc.ratio.origin, is a parameter in the calculation of a partial core electron density. The core electron density is approximated using a fourth order polynomial, so that the core electron at the origin satisfies a relation, $\rho_c(0)=\text{pcc.ratio.origin}\times\rho_c(r_{\text{pcc}})$.

Specific keywords for calc.type=PAO

maxL.pao

The pseudo-atomic orbitals are generated up to an angular momentum quantum number, maxL.pao

num.pao

The number of pseudo-atomic orbitals generated with the same angular momentum quantum number.

radial.cutoff.pao

The keyword, radial.cutoff.pao, specifies a cutoff radius r_c (a.u.) for the pseudo-atomic orbitals.

height.of.wall

The keyword, height.of.wall, specifies a height (Hartree) of confinement wall

rising.edge

The keyword, rising.edge, controls a shape of rising edge of the confinement wall. Note that there is a relation $r_1=r_c-\text{rising.edge}$. See also the section, Making of pseudo-atomic orbitals.

search.LowerE

The keyword, `search.LowerE`, gives the lower bound of energy for searching the eigen energies of pseudo-atomic orbitals.

`search.UpperE`

The keyword, `search.UpperE`, gives the upper bound of energy for searching the eigen energies of pseudo-atomic orbitals.

`num.of.partition`

The keyword, `num.of.partition`, gives the number of energy partitioning, ranging from the `search.LowerE` to the `search.UpperE`. First, the eigenstates of pseudo-atomic orbitals are roughly explored for the energy ranges partitioned by the keyword, `num.of.partition`. Then, the eigenstates are refined in the energy range with a correct number of nodes.

`matching.point.ratio`

The keyword, `matching.point.ratio`, gives a matching point to connect two wave functions solved from the origin and the distant. It should be noted that the matching grid number is given by `matching.point.ratio` \times `grid.num`.

5 All electron calculation

In this section, keywords for the all electron calculation are explained. These keywords discussed here are important for all calculations, since both the generations of pseudopotentials and pseudo-atomic orbitals are based on the all electron calculation. The list of keywords and some comment for the all electron calculation are as follows:

1. **xc.type**

Choose GGA or LDA

2. **total.electron**

Give the total number of electrons. It is also possible to give the number of electrons corresponding to not only a neutral atom, but also a positive or negative charged atom.

3. **grid.xmin**

Set `grid.xmin` ($r_{\min}(\text{a.u.})=\exp(\text{grid.xmin})$), where r_{\min} is the minimum radius from which radial differential equations are solved to a distant. An appropriate value for `grid.xmin` is -7.0 from H to Kr, and -8.0 for heavier atoms.

4. **grid.xmax**

Set `grid.xmax` ($r_{\max}(\text{a.u.})=\exp(\text{grid.xmax})$), where r_{\max} is the maximum radius from which radial differential equations are solved to the origin. An appropriate value for `grid.xmax` is 2.5 to 4.0, but could depend on whether there are delocalized states.

5. **grid.num**

Set the number of grids to solve radial differential equations. A larger number of grids gives a higher degree of accuracy, while the computational time increases. An appropriate value for `grid.num` is 3000 to 12000. For heavier atoms, the use of a larger number of grids is better to achieve a reliable calculation.

6. **grid.num.output**

It is possible to change the number of grids for r in output files by the keyword, `grid.num.output`, although the actual calculation is performed using `grid.num`. Usually, we use 2000 for it.

7. **scf.maxIter**

Set the maximum number of SCF iteration.

8. **scf.Mixing.Type**

Choose a method for charge mixing. Either simple or GR-Pulay is available. In most cases, the simple mixing scheme is enough to achieve a sufficient convergence.

9. **scf.Min.Mixing.Weight**

Set the minimum mixing weight.

10. **scf.Max.Mixing.Weight**

Set the maximum mixing weight.

11. **scf.Mixing.History**

Set previous SCF steps for charge mixing in GR-Pulay method.

12. **scf.Mixing.StartPulay**

Set a SCF iteration number from which GR-Pulay starts.

13. **scf.criterion**

Set scf.criterion. At least 1.0e-10 for the keyword should be chosen for a convergent calculation.

6 Making of pseudopotential

6.1 Example

Generation of pseudopotentials is illustrated for the case of a carbon atom. Please set the keyword, `calc.type`, to `VPS` in the input file `C.inp`, and perform as follows:

```
% adpack C.inp
```

When the calculation is completed normally, the following seven files newly generate in the directory, `work`.

<code>C0.nsvps</code>	non-separable pseudopotentials
<code>C0.vps</code>	input file, results of the SCF calculation, and pseudopotentials in the KB or Blochl separable form, and partial core density for PCC
<code>C0.vpao</code>	radial parts of pseudo-atomic orbitals for pseudopotentials
<code>C0.vden</code>	valence electron density, the total electron density, core electron density, modified core electron density for PCC
<code>C0.loc</code>	local part of pseudopotentials
<code>C0.ld0</code>	logarithmic derivatives of wave functions(<code>l=0</code>).
<code>C0.ld1</code>	logarithmic derivatives of wave functions(<code>l=1</code>).

C0.nsvps

In a file, `C0.nsvps`, the pseudopotentials in a non-separable form are output, in which they are listed in order of $\log(r)$, r , the pseudopotential 0, and the pseudopotential 1, ..., where the number referred to specify the pseudopotential corresponds to the number given for the first column in the specification of the keyword, `pseudo.NandL`, in the input file. Figure 2 shows the pseudo potentials of a carbon atom stored in the file, `C0.nsvps`.

C0.vps

In a file, `C0.vps`, the pseudopotentials in a Blöchl separable form are output, in which they are listed in order of $\log(r)$, r , the local part of the pseudopotential, and the non-local part of the pseudopotential. Also, the input file and the results of the SCF calculation are added in this file for your adversaria. The file is output in the flexible data format, since the file `*.vps` is used for the input file to the program package, OpenMX. In Fig. 2(b) shows the Blöchl separable pseudopotentials of a carbon atom. In case of `charge.pcc.calc=ON`, then the file also includes the partial core density for PCC [10]. The format is the same as that of the pseudopotential, and they are listed in order of $\log(r)$, r , and the partial core density. The data of the partial core density is also used as the input date of OpenMX. In Fig. 3, the partial core density is shown together with the valence electron density stored in the file, `C0.vden`.

C0.vpao

The pseudo-atomic orbitals corresponding to the pseudopotentials are output in a file, `C0.vpao`. The format of the output is the same as that of `C0.nsvps`. Figure. 2(a) shows the pseudo-atomic orbitals and the pseudopotentials.

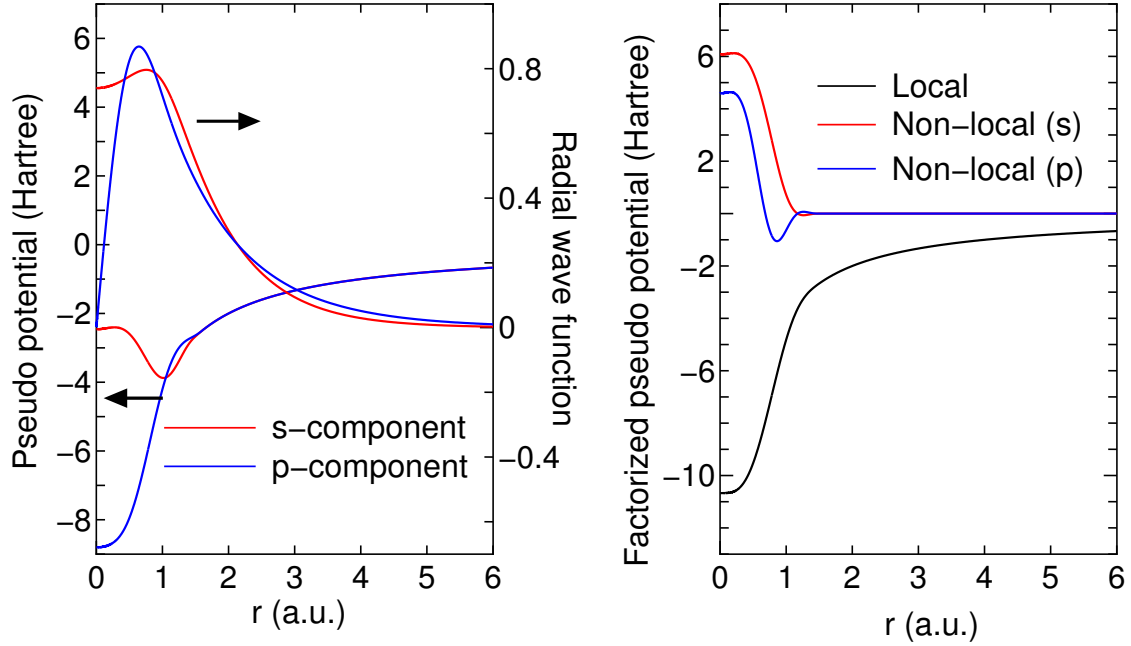


Figure 2: (a) Radial parts of the pseudo-atomic orbitals and the corresponding norm-conserving pseudopotentials, (b) Norm-conserving pseudopotentials in a Blöchl separable form

C0.vden

The electron density for the valence electron is stored in a file, C0.vden.

In case of charge.pcc.calc=OFF, the data are output in order of

$$\log(r), r, \rho_v, \rho_t, \rho_c, 4\pi r^2 \rho_v, 4\pi r^2 \rho_t, 4\pi r^2 \rho_c.$$

In case of charge.pcc.calc=ON, the data are output in order of

$$\log(r), r, \rho_v, \rho_t, \rho_c, \rho_{pcc}, 4\pi r^2 \rho_v, 4\pi r^2 \rho_t, 4\pi r^2 \rho_c, 4\pi r^2 \rho_{pcc}.$$

where

- ρ_v : the valence electron density,
- ρ_t : the total electron density,
- ρ_c : the core electron density,
- ρ_{pcc} : the modified core electron density for PCC.

C0.loc

The local part of separable pseudopotentials is output in the file, C0.loc, in order of $\log(r)$, r , and the local part. Figure. 2(b) shows the local part of the pseudopotentials.

C0.ld*

The logarithmic derivatives of radial wave functions are output in the file, C0.ld*, where * means the angular momentum quantum number. The data are stored in order of energy and the logarithmic derivatives of radial wave functions under the all electron potential, semi-local pseudopotential, and fully separable pseudopotential.

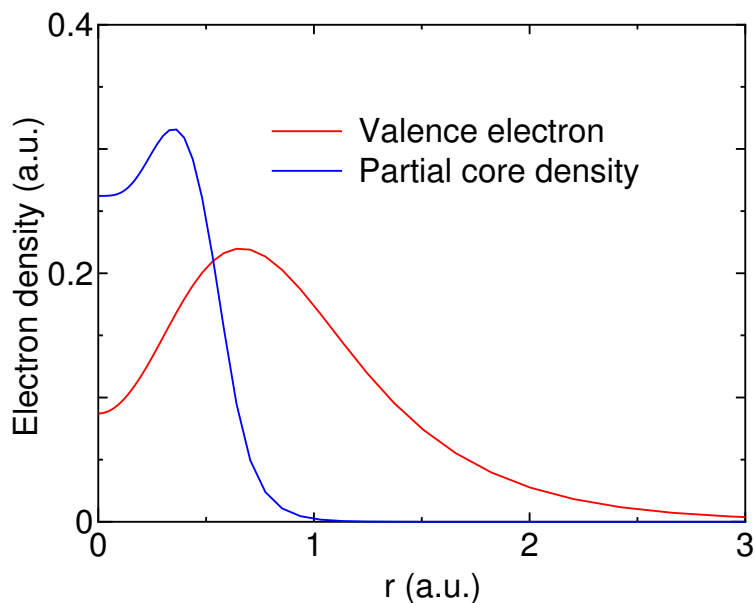


Figure 3: Valence electron and partial core densities of a carbon atom

In the generation of pseudopotentials, it is possible to choose either BHS type or TM type. In the template file, C.inp, TM type is chosen as the generation scheme. In practice, the choice of a suitable cutoff radius in the pseudopotential generation is made by a trial and error way so that the shape of the generated pseudopotentials is smooth. Also, it is required to carefully check whether appropriate results are obtained or not in physical quantities that you want to calculate when density functional calculations are performed for molecules and solids using the generated pseudopotentials. In addition to this, there is some ambiguity in the selection of states to generate pseudopotentials, while states, which can form the valence states, are selected in general.

6.2 Cutoff radius

Cutoff radii of pseudopotentials are specified by the following keyword:

```
<pseudo.NandL
 0 2 0 1.50
 1 2 1 1.62
pseudo.NandL>
```

The first number specifies a line number beginning from zero line, which is used in the specification of the keyword, local.part.vps. In the second or third columns, a principal number and an angular momentum quantum number are given. The fourth column provides a cutoff radius (a.u.) for the generation of pseudopotentials. Although an optimum cutoff radius is determined so that the generated pseudopotentials has a smooth shape without distinct kinks and a lot of nodes, however, the selection includes somewhat a empirical factor. It is also possible to take into account semicore states in the generation of pseudopotentials. For example, if you want to include 3s and 3p states as semicore states in a sodium atom, the specification is as follows:

```

<pseudo.NandL
  0  3  0  1.8
  1  3  1  2.3
  2  4  0  1.8
  3  4  1  2.3
pseudo.NandL>

```

In this case, a pseudopotential is generated for the lowest state in each angular momentum quantum number.

6.3 Pseudopotentials for unbound states

It is possible to generate pseudopotentials for unbound states with for any higher L-component by Hamann's scheme [7]. For example, although no electron is occupied for the 3d state in the input file 'C.inp', the cutoff radius for the 3d state can be specified as follows:

```

number.vps          3

<pseudo.NandL
  0  2  0  1.3
  1  2  1  1.3
  2  3  2  1.0
pseudo.NandL>

```

The pseudopotential generation of the 3d state will be generated with the cutoff radius, and then the reference energy is 0.0 (a.u.). A principal number and an angular momentum quantum number for the unbound state should be given as the state above occupied states but with the smallest principal number.

6.4 Separable form

Norm-conserving pseudopotentials generated by BHS and TM scheme are written in a semi-local form which is based on a projection by the spherical harmonic function. In the application of pseudopotentials to molecules and bulks, the semi-local form is rewritten by a fully separable form proposed by Kleinman and Bylander (KB) [5] or Blöchl [6] to reduce the computational effort. Then, the following keywords are important for transferability of the separable form.

- **Bloch1.projector.num**

The number of Blöchl projectors for each L-component in separable pseudopotentials. If you specify 1 for Blochl.projector.num, this means the Kleinman and Bylander (KB) separable pseudopotentials.

- **local.type**

'Simple' and 'Polynomial' are available.

- **local.part.vps**

Number of local potential in case of local.type=Simple

- **local.cutoff**

The cutoff radius of local part in case of local.type=Polynomial

- **local.origin.ratio**

Depth of local part at the origin in case of local.type=Polynomial

You can find details for these keyword in the section, Input file.

6.5 Logarithmic derivative of wave function

To check the transferability of generated pseudopotentials, a useful measure is to compare logarithmic derivatives of wave functions [12]. If the logarithmic derivative of pseudopotential is comparable to that by the all electron calculation through a wide range of energy, then the pseudopotential would possess a good transferability. In Fig. 4 shows the logarithmic derivatives in a carbon atom, indicating a good transferability of the pseudopotential. The keywords concerned to the calculations of the logarithmic derivative are as follows:

- **log.der.RadF.calc**

When the logarithmic derivatives are calculated, then ON, otherwise, OFF.

- **log.der.MinE**

The lower bound of energy (Hartree) used in the calculation of logarithmic derivatives of radial wave functions.

- **log.der.MaxE**

The upper bound of energy (Hartree) used in the calculation of logarithmic derivatives of radial wave functions.

- **log.der.R**

Radius at which the logarithmic derivatives of radial wave functions are evaluated.

You can find details for these keyword in the section, Input file. In case of log.der.RadF.calc=ON, calculated logarithmic derivatives are output in files *.ld#, where * is the file name that you specified by the keyword, System.Name, and # is the angular momentum number. If the full relativistic calculation is performed as 'eq.type=dirac', the file name is *.ld%_#, where % runs 0 to 1 corresponding to $j = l + 1/2$ and $j = l - 1/2$, respectively.

6.6 Ghost states

The fully separable form of pseudopotential would possess artificial ghost states [13], which is one of serious problems in the separable form, while multiple projectors proposed by Blöchl [6] is highly effective to avoid the existence of ghost states. To check it, a keyword, ghost.check, is provided. Although the keyword is useful to find the ghost states, however, it should be noted that a complete check to detect the ghost states is difficult.

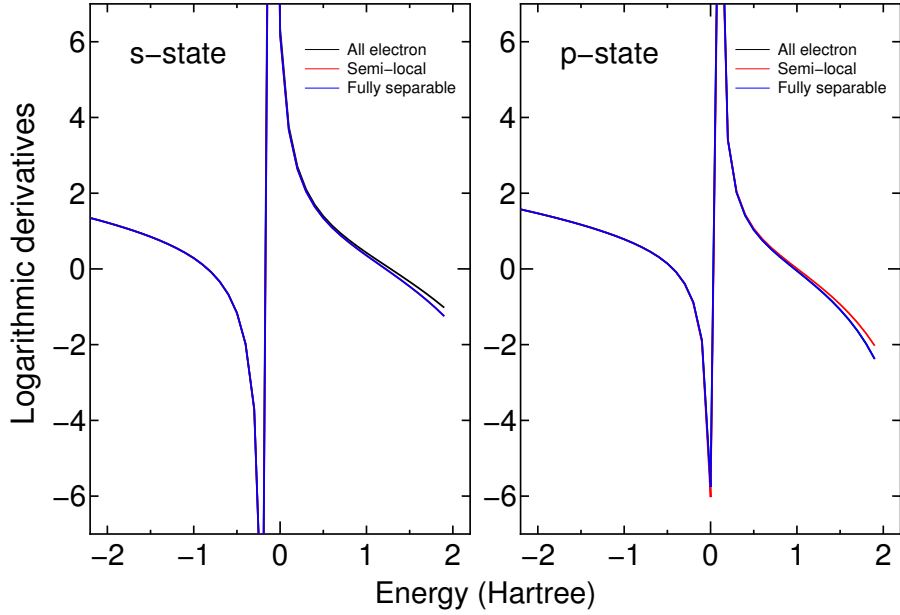


Figure 4: Logarithmic derivatives of radial wave functions under the all electron potential, semi-local pseudopotential, and fully separable pseudopotential of a carbon atoms

6.7 Partial core correction

The contribution to electron density from core electrons is ignored in the evaluation of exchange-correlation energy in the pseudopotential method, although there is a non-linearity of exchange-correlation energy with respect to electron density. Thus, a partial core correction would be important in order to take account of the non-linearity. A partial core charge for the partial core correction can be constructed by the following keywords:

- **charge.pcc.calc**

When a partial core charge is calculated, ON, otherwise OFF.

- **pcc.ratio**

The keyword, pcc.ratio, is a parameter in the calculation of a partial core electron density. The core electron density is approximated using a fourth order polynomial below the cutoff radius r_{pcc} at which the ratio ρ_c/ρ_v between the core electron density ρ_c and the valence electron density ρ_v becomes pcc.ratio.

- **pcc.ratio.origin**

The keyword, pcc.ratio.origin, is a parameter in the calculation of a partial core electron density. The core electron density is approximated using a fourth order polynomial so that the core electron at the origin satisfies a relation, $\rho_c(0) = \text{pcc.ratio.origin} \times \rho_c(r_{\text{pcc}})$.

Note that a precipitous partial core charge would cause numerical instabilities. Thus, a modest core charge is better from a numerical point of view.

6.8 Restart

As discussed above, a trial and error is needed to generate optimum pseudopotentials. However, all electron calculation prior to the pseudopotential generation requires a considerable computational time for an atom with a large atomic number. Therefore, it is desirable to reduce the computational time that results of the all electron calculation are stored in a file and skip the all electron calculation when we regenerate pseudopotentials in different parameters. To do this, two keywords, `System.UseRestartfile` and `System.Restartfile`, are available. The details are as follows:

- **System.UseRestartfile**

For an atom with a large atomic number, all electron calculation requires a considerable computational time. So, it is needed to reduce the computational time, when optimal cutoff radii of pseudopotentials is determined in a trial and error. If the keyword, `System.UseRestartfile`, is specified as YES, a restart file which contains informations of all electron calculation is used in order to skip all electron calculation. If there is no restart file, a restart file is generated in case of `System.UseRestartfile=YES`.

- **System.Restartfile**

If `System.UseRestartfile=YES`, then the name specified by the keyword, `System.Restartfile`, is referred to as a restart file.

7 Relativistic calculation

7.1 All electron calculation

Relativistic effects can be included in both the scalar relativistic [14] and the full relativistic treatment [3, 15]. To specify these, there are three options for the keyword, eq.type, as follows:

```
eq.type          sch          # sch|sdirac|dirac
```

where 'sch', 'sdirac', and 'dirac' mean the Schrödinger equation (no relativistic effect), a scalar relativistic treatment, and a full relativistic treatment of Dirac equation, respectively. In the scalar relativistic treatment, the coupled Dirac equations are averaged with a weight of j-degeneracy, and solved by taking account of both the majority and minority components of radial wave function. Thus, the scalar relativistic treatment includes explicitly kinematic relativistic effects (Darwin and mass velocity terms), and implicitly averaged spin-orbit coupling (no energy splitting). On the other hand, in the full relativistic treatment, j-dependent Dirac equations are solved including both the majority and minority components of radial wave function. Thus, energy splitting by spin-orbit coupling are also considered. In Table 1 shows eigenvalues of atomic platinum calculated by three different methods, where electron of 6s state is partially promoted to 6p state.

Table 1: Eigenvalues (Hartree) of atomic platinum calculated by the Schrödinger equation, a scalar relativistic treatment, and a full relativistic treatment of Dirac equation within GGA to DFT

state	sch	sdirac	dirac	
			j=l+1/2	j=l-1/2
1s	-2590.5589	-2830.0886	-2823.3059	
2s	-428.9975	-495.0056	-493.0913	
2p	-411.3962	-430.9421	-412.3151	-473.9681
3s	-101.5684	-118.0265	-117.5050	
3p	-93.5941	-99.3621	-95.2182	-108.6139
3d	-78.0400	-77.5940	-76.0344	-78.8582
4s	-20.8655	-24.9230	-24.7628	
4p	-17.5020	-18.8048	-17.7975	-21.0224
4d	-11.2052	-11.0328	-10.6848	-11.2849
4f	-3.1801	-2.7527	-2.6278	-2.7627
5s	-2.9300	-3.6769	-3.6427	
5p	-1.8835	-2.0614	-1.8949	-2.43625
5d	-0.2763	-0.2383	-0.2140	-0.26223
6s	-0.1648	-0.2173	-0.2177	

7.2 Enhancement or depletion of a spin-orbit coupling

To study the effect of a spin-orbit coupling, it is possible to generate a pseudopotential with a larger or smaller spin-orbit coupling compared to that in a real atom. The scaling factors can be specified to each angular momentum quantum number by the following keyword:

```
<SO.factor  
0 1.0  
1 0.5  
2 2.0  
SO.factor>
```

The beginning of the description must be <SO.factor, and the last of the description must be SO.factor>.

The number in the first column corresponds to that in the keyword 'pseudo.NandL', and a scaling factor is given for each pseudopotential by the second column, where '1.0' corresponds to the spin-orbit coupling in a real atom. One can control the strength of spin-orbit coupling by changing the scaling factor.

8 Making of pseudo-atomic orbitals

The pseudo-atomic orbitals are used in the program package, OpenMX, as the primitive basis orbitals. The pseudo-atomic orbitals are generated as follows: first, the SCF calculation is performed in consideration of all electrons under a confinement potential, second, the pseudopotentials are generated, and finally, the pseudo-atomic orbitals for the confinement pseudopotentials are evaluated numerically up to a required excited state. In this section, the generation of the pseudo-atomic orbitals is illustrated. In the file, C.inp, please set the keyword, calc.type, to PAO, and run the executable file, adpack, as follows:

```
% adpack C.inp
```

When the run is completed normally, then you find a file, C0.pao, in the directory, work. In this file, C0.pao, the valence electron density and the radial parts of the pseudo-atomic orbitals are output. For your adversaria, the contents of the input file and the results of all electron SCF calculation are also included. They are stored in order of $\log(r)$, r , and the valence electron density, and in order of $\log(r)$, r , and the radial part 1, the radial part 2,..., in the flexible date format, respectively. In Fig. 4, the confinement potential and the pseudo-atomic orbitals for the s-orbital are shown. From Fig. 4, we see that the pseudo-atomic orbitals are localized due to the confinement potential, and the number of nodes increases as the eigenvalue increases. The confinement potential is made by modifying the core potential as follows:

$$V_{\text{core}}(r) = \begin{cases} -\frac{Z}{r} & \text{for } r \leq r_1, \\ \sum_{n=0}^3 b_n r^n & \text{for } r_1 < r \leq r_c, \\ h & \text{for } r_c < r, \end{cases} \quad (1)$$

where b_0 , b_1 , b_2 , and b_3 are determined so that the values and the derivatives are continuous at both r_1 and r_c . Considering that there are relations, r_c =radial.cutoff.pao, r_1 = r_c -rising.edge, and h =height.of.wall, we find that the tunneling of wave function for the confinement wall becomes small as height.of.wall increases. Also, it is possible to control the shape of the rising edge around the wall by changing rising.edge. If you use a huge value for height.of.wall, then you might meet a case that the calculation is not completed normally, since the computational instability appears often. In such a case, the numerical instability may be avoided by enlarging the keywords, rising.edge and num.grid. As for the keyword, rising.edge, please refer the section, Input file. The file, *.pao, created here can be an input file of the program package, OpenMX.

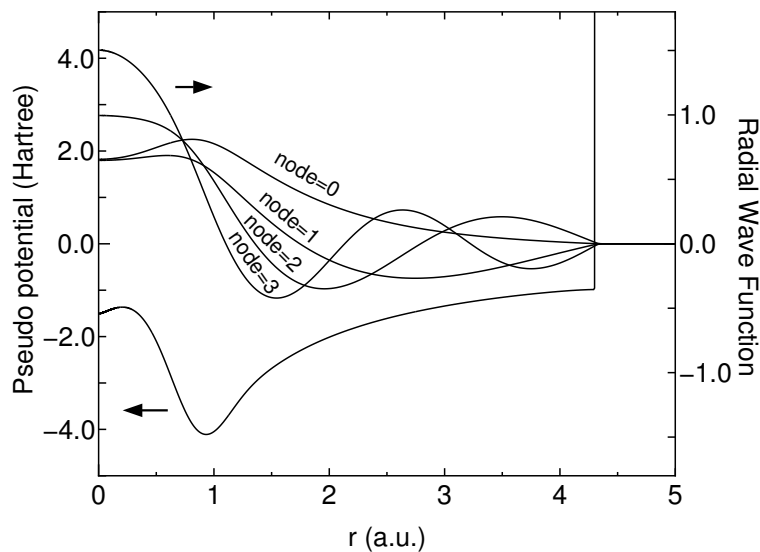


Figure 5: Confinement potential and radial parts of pseudo-atomic orbitals of a carbon atoms

9 Virtual atom with fractional nuclear charge

It is possible to generate pseudopotentials and basis functions for a virtual atom with fractional nuclear charge. The relevant keywords in ADPACK are given by

```
AtomSpecies          6.2
total.electron       6.2
valence.electron     4.2
<ocupied.electrons
 1  2.0
 2  2.0  2.2
ocupied.electrons>
```

The above example is for a virtual atom on the way of carbon and nitrogen atoms. By just controlling the above keywords, you can easily generate pseudopotentials and basis functions for virtual atoms. When you use those in OpenMX as input data, no specification by keywords is required. Please make sure that only OpenMX Ver. 3.4 or later accepts the pseudopotentials and the basis functions for the virtual atoms. Also, it is noted that basis functions for the pseudopotential of the virtual atom must be generated for the virtual atom with the same fractional nuclear charge, since the atomic charge density stored in *.pao is used to make the neutral atom potential in OpenMX.

10 Output files

The list of output files is listed below. The details of each file are described in each section (test calculation, making of pseudopotential, and making of pseudo-atomic orbitals).

`calc.type=ALL`

<code>C0.aalog</code>	input file and results of SCF calculations
<code>C0.ao</code>	radial wave functions in all electrons SCF calculations
<code>C0.aden</code>	electron density of all electrons.

`calc.type=VPS`

<code>C0.nsvps</code>	non-separable pseudopotentials
<code>C0.vps</code>	input file, results of the SCF calculation, and pseudopotentials in a KB separable form, and partial core density PCC
<code>C0.vpao</code>	radial parts of pseudo-atomic orbitals for pseudopotentials
<code>C0.vden</code>	valence electron density, the total electron density, core electron density, modified core electron density for PCC
<code>C0.loc</code>	local part of pseudopotentials
<code>C0.ld0</code>	logarithmic derivatives of wave functions(l=0).
<code>C0.ld1</code>	logarithmic derivatives of wave functions(l=1).

`calc.type=PAO`

<code>C0.pao</code>	radial parts of pseudo-atomic orbitals under a confinement potential
---------------------	--

In these output files, two files, `C0.vps` and `C0.pao`, could be the input files for OpenMX. When these two files are used in OpenMX, please copy `C0.vps` to directory, `openmx*/DFT_DATA/VPS`, and copy `C0.pao` to directory, `openmx*/DFT_DATA/PAO`, respectively.

11 Templates of the input files

There are templates of the input files of the atoms ($Z=1-103$), which can be used for your purpose.

12 Others

Program

The program package is written in the C language, including one makefile

makefile,

two header files

adpack.h,
Inputtools.h,

and 53 routines

```
addfunc.c adpack.c All_Electron.c BHS.c Calc_Vlocal.c Core.c  
Density.c Density_PCC.c Density_V.c DMF_Func.c Empty_VPS.c  
E_NL.c Find_LESP.c Frho_V.c Gauss_Legendre.c Gauss_LEQ.c  
Generate_VNL.c ghost.c GR_Pulay.c GVPS1.c GVPS2.c GVPS.c  
Hamming_I.c Hamming_O.c Hartree.c HokanF.c Initial_Density.c  
Init_VPS.c Inputtools.c Log_Deriv.c Make_EDPP2.c Make_EDPP3.c  
Make_EDPP4.c Make_EDPP.c MPAO_RadialF.c MR.c Multiple_PAO.c  
Output.c PAO_RadialF.c readfile.c Restart.c Set_Init.c  
Simple_Mixing.c TM.c Total_Energy.c VNLf.c VP.c  
XC4atom_PBE.c XC_CA.c XC_EX.c XC_PBE.c XC_PW91C.c XC_Xa.c
```

Copyright of the program package

The distribution of this program package follows the practice of the GNU General Public Licence [17]. Moreover, the author, Taisuke Ozaki, possesses the copyright of the original version of this program package. I cannot offer any guarantees in your use of this program package. However, when you report some program bugs, I will cooperate and work well as much as possible together in you to remove the problems.

Contributors

T.Ozaki (JAIST),
H.Kino (NIMS)

References

- [1] D. M. Ceperley and B. J. Alder, Phys. Rev. Lett., 45, 566 (1980); J. P. Perdew and A. Zunger, Phys. Rev. B **23**, 5048 (1981).
- [2] J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).
- [3] G. B. Bachelet, D. R. Hamann, and M. Schluter, Phys. Rev. B **26**, 4199 (1982).
- [4] N. Troullier and J. L. Martine, Phys. Rev. B **43**, 1993 (1991).
- [5] L. Kleinman and D. M. Bylander, Phys. Rev. Lett. **48**, 1425 (1982).
- [6] P. E. Blochl, Phys. Rev. B 41, 5414 (1990).

- [7] D. R. Hamann, Phys. Rev. B 40, 2980 (1989).
- [8] Density functional theory and its applications - electronic structures of molecule and cluster - A. Kanno, C. Satoko, and S. Oonishi, Koudansha scientific in Japanese.
- [9] D. R. Bowler and M. J. Gillan, Chem. Phys. Lett. 325, 475 (2000).
- [10] S. G. Louie, S. Froyen and M. L. Cohen, Phys. Rev. B 26, 1738 (1982)
- [11] T. Ozaki, Phys. Rev. B 67, 155108 (2003); T. Ozaki and H. Kino, appear to Phys. Rev. B.
- [12] X. Gonze et al., Phys. Rev. B 41, 12264 (1990).
- [13] D. M. Bylander and L. Kleinman, Phys. Rev. B 41, 907 (1990)
- [14] D. D. Koelling and B. N. Harmon, J. Phys. C: Solid State Phys. 10, 3107 (1977)
- [15] A. H. MacDonald and S. H. Vosko, J. Phys. C: Solid State Phys. 12, 2977 (1979).
- [16] <http://xfree86.cygwin.com/>
- [17] <http://www.gnu.org/home.html>
- [18] the web site on pseudopotential calculations by Dr. Kobayashi at NIMS
<http://www.geocities.co.jp/Technopolis/4765/INFO/guidej.html>