Manual for

Ab Initio Research Services LTD.

PACKAGES

August 2019

MANUAL

August 2019

AB INITIO RESEARCH SERVICES Ltd.

Preface

Ab Initio Research Services was founded in the September of 2017. Contact us at: ????. Or visit our website:
`http://abinires.com/?v=eb78d28dfa79`

License

Contents

1 Overview

This document contains a detailed description of all the Ab Initio Research Services packages. These packages are:

AbiView This package provides an easy way to create job files and other small programs to evaluate results obtained from the other Ab Initio Research Services LTD. packages. The module and its source code are available, free of charge.

ADMAPack Provides the means to compute the density matrices of large molecules. The package contains the RhoCalc, with this module the electron density, electrostatic potential and energy of the molecule can be calculated, using the obtained density matrix.
Note: to use the ADMAPack you must have direct access to Gaussian software or Gamess software software. Field Adapted- ADMA calculations are currently only available for Gamess software.

ShapeGPack A package for the computation of shape maps of a molecules electron density. Comparing these shape gives a measure of similarity between molecules, from which you can asses chemical properties of the compared molecules. The package contains the RhoCalc.

ServerPack This package is designed to manage your computations and distribute jobs among nodes of a server. Currently it is only available for Linux machines.

All of the afore mentioned packages can be downloaded from our website (`http://abinires.com/?v=eb78d28dfa79`). The website also offers the chance to try out these packages by running them on our servers. For more details on how to use this service visit the website.

Explanation of symbols used in the document are given in section 9

2 ADMAPack

2.1 Overview/Recommendations

The Adjustable Density Matrix Assembler method provides easy means to calculate the *ab initio* quality density matrix of a large molecule. The procedure consists of the division of large molecules into smaller fragments, for which quantum chemical calculations can be performed easily. This fragmentation procedure is based on the Additive Fuzzy Density Fragmentation method. These fragments, called nuclear families, are formed from target molecules and their surroundings. The target molecules are mutually exclusive atom groups, in our program these are the residues of the molecule. The surroundings of each target molecule are chosen based on the selection radius parameter. If any atom of another residue is situated within this distance from the target molecule, then the residue is considered to be part of the nuclear family. Single point calculations are performed for all the nuclear families, after which the elements corresponding to the respective target molecule are reassembled to form the full molecules density matrix, as described by the Adjustable Density Matrix Assembler method.

The Field Adapted- ADMA is an iteration of the Adjustable Density Matrix Assembler. In each iterative step the Mulliken charges of atoms outside the fragment, obtained in the previous step, are used to recompute the fragments density matrix. By including these partial charges the precision of the computations are increased, but computational time is raised because of the iteration steps. It is recommended to use the results obtained in the second step of the iteration, as in most cases this provides the most accurate value.
Notice: Field Adapted- ADMA calculations are currently only available for Gaussian software.

The selection radius affects the size of the fragments, thus increasing its value will result in an increase of precision but it will also raise the computation times. It is recommended to use at least a 3Åselection radius to preserve the sulfur-sulfur bonds within fragments.

There are two assembly schemes available: Mezey-Mulliken scheme and the block-ADMA scheme. These only differ in the assembly step of the procedure. In the case of the Mezey-Mulliken scheme, the elements for which at least on orbital is centered on an orbital of the target molecule are selected from the density and Fock matrix. The block-ADMA scheme only selects the elements which have both orbitals centered on the target molecule. In general the block-ADMA scheme results a block-diagonal matrix which is more sparse then the one obtained from the Mezey-Mulliken scheme. The fewer non-zero elements of the block-diagonal matrix means an increase in computational speed, but also a loss in precision. It is recommended to use the block-ADMA approach to compute electron density and electrostatic potential of molecules.

The properties which can be calculated from the density matrix are: electron density on grid, electrostatic potential on a grid and energy. These can be obtained using the RhoCalc which is included in the package. For details on editing the basic settings of the program or to redefine the constants used please see: 8.

2.2 Preparing input '.pdb' files

The '.pdb' files which will be used as input must first cleaned of water residues and any atoms in surplus. The atoms and residues must be ascending order of their indices, the N and C atom names are reserved for the N and C terminal atoms of each residue. The structure must also have hydrogen all atoms. The specification of residue and terminal charges of the '.pdb' can be done using the PDBPrep found in the AbiView. For more details see section 5.2.

2.3 Running calculations

There are multiple options on how to run ADMA calculations, these are:

Via direct input To run calculations with direct input open the program in the terminal or the command line. You will be prompted to introduce the input files and parameters used by the module.

Via job file For doing calculations by job files, run the abiParse in the terminal or command line with the job file as the system argument. For example in Linux:
./path/to/ADMAPack/abiParse path/to/jop/file/job_file.job

Via the ServerPack To run the ADMA computations on a server, make sure to have the ServerPackinstalled, then copy the job file into the input directory of the ServerPack.

2.4 Job parameters [keyword:ADMA]

To create the job file you may use the ADMA module of the AbiView, or write it manually using any text editor. For more details on structure of job files see: section 7.

The keyword of ADMA calculations is **ADMA**. The parameters and their respective meanings:

PDB file The input '.pdb' file containing the structure

PDB dir Absolute path to directory where, the input '.pdb' file is located.

scheme (Integer) ADMA scheme to be used, when assembling fragments. (**1** - Mezey-Mulliken scheme, **2** - block-ADMA scheme)

selection radius (Floating point value) Selection radius that will be used when creating parent molecules.

method Method/DFT method that will be used in single point calculations.

basis set Basis set that will be used to do single point calculations.

sp program (Integer) Specify program for executing single point calculations. (1-Gaussian, 2-GAMESS)

nproc (Integer) Number of processors which will run single point calculations. (Only for Gaussian)

memory (Floating point value) GBs of memory allocated for single point calculations. (Only for Gaussian)

Fragment dir Directory to write nuclear family data. (Debugging)

Save file File name for results. The program outputs a '.mol' containing basic description of molecule and some matrix files.

Save dir Directory where the output files will be saved, if unspecified, the programs writes data into the scratch directory.

FA ADMA (Integer) Wether to perform FA-ADMA calculation or just ADMA calculation. (1 - do FA-ADMA, 0 - do NOT do FA-ADMA) (default=0)

chargeconv (Floating point value) Value of convergence of Mulliken charges in FA-ADMA calculations. If *maxcycles* specified, this value is omitted and FA-ADMA calculation is terminated, when it reaches numebr of maximum cycles. (default=1e-9)

maxcycles (Integer) Maximum number of iterative cycles allowed in FA-ADMA calculation. (default=None, if unspecified, iterations will run untill the convergence value is reached)

charge factor (Floating point value) Scaling factor for Mulliken charges used in FA-ADMA calculations. (default=1.0, no scaling is applied)

3 ShapeGPack

3.1 Overview/Recommendations

The module is used to generate a similarity index between molecules or fragments of molecules based on the shape and topology of their electron density. Isosurfaces of the electron density are partitioned into domains, based on their curvature properties relative to multiple reference values. The domains are: convex, concave and saddle-type. For each electron density isosurface value and reference curvature pairing the isosurface is truncated. In this truncation all convex domains are removed and the remaining form is encoded into an integer. The encoding is performed based on the Betti numbers of the shape. The program computes these encoded values for 41 isosurface values and 21 reference curvature values, which results in a 41×21 integer matrix, which is called a shape map. By direct comparison of the values of shape maps the similarity index is obtained.

In the first step the electron density is computed in a grid enclosing the molecule. Smaller resolution values of this grid, create a finer electron density surface, but also increase computation time. Resolution values above 0.5 Bohr are great to obtain rapid but rough comparisons, but for more detailed studies it is recommended to use 0.1 Bohr resolutions. Next Hessian matrices are computed for each voxel, the eigen values of these matrices are saved in a '.ev.npy' binary file. From these grids the shape map is created and written into a '.map' file.

Parallel computation options are available for the module. The electron density and shape map computation can be spread out on multiple processors, by setting the *nproc* value to the number of processors that you wish to use. GPU support is also available for CUDA capable devices, this requires that you have the NVIDIA driver and CUDA toolkit installed. Notice: only the electron density computation is performed on GPU.

Comparison of the shape maps can be done easily using the CompMap of the AbiView. The package also includes the RhoCalc module.

3.2 Preparing input molecule files

The program requires '.mol' files as input for calculations. Using the ToMol of the AbiView output files from other programs can be converted into the '.mol' format. Currently supported file types are: '.log' obtained from Gaussian software or Gamess software calculations and '.fchk' files obtained from Gaussian software.

3.3 Running calculations

There are multiple options on how to run Shape Group Analysis calculations, these are:

Via direct input To run calculations with direct input open the program in the terminal or the command line. You will be prompted to introduce the input files and parameters used by the module.

Via job file For doing calculations by job files, run the abiParse in the terminal or command line with the job file as the system argument. For example in Linux:
./path/to/ShapeGPack/abiParse path/to/job/file/job_file.job

Via the ServerPack To run the Shape Group Analysis computations on a server, make sure to have the ServerPackinstalled, then copy the job file into the input directory of the ServerPack.

3.4 Job parameters [keyword:ShapeGroup]

The keyword for Shape Group Analysis calculations is **ShapeGroup**. The parameters are:

molecule file Input file used fo calculations.

molecule dir Direcotry of the input file.

resolution (Floating point value) Resolution of grid on which molecular properties are calculated. (values: 0.25, 0.5, 1.0)

GPU (Integer) Use GPU in electron density calculation. (1-use GPU, 0-do NOT use GPU) (default=0)

nthreads (Integer) Number of thread/block to be used on GPU (CUDA capable), it is recommended tha you use multiple 32(WARP size). Only specify is GPU is used for electron density calculations. (default=512)

fragment (Comma separated integers) Indices of atoms which are studied, use *0* if full molecule is used. (default=0)

grid file File name which will contain grid data.

grid dir Directory in which grid data will be written. If unspecified the grid data is saved to scratch directory.

map file File name of the map file.

map dir Directory where the map file will be saved. If unspecified map file is saved to the scratch directory.

4 RhoCalc module

4.1 Overview

The module computes molecular properties: electron density, electrostatic potential and energy based on data read from molecule files. The electron density and electrostatic potential are calculated on a grid. For each voxel of the grid the electron density value is given by:

$$\rho_{i_x i_y i_z} = \sum_i \sum_j \Psi_i(i_x, i_y, i_z) P_{ij} \Psi_j(i_x, i_y, i_z) \tag{1}$$

The electrostatic potential is calculated based on the condensed Mulliken charges of the molecule:

$$EP_{i_x i_y i_z} = \sum_a \frac{Q_a^M}{\boldsymbol{r}_{av}} \tag{2}$$

The energy of a molecule:

$$E_{HF} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} ((F_{ij} + H_{ij}^{core}) \cdot P_{ij}) + V_{NN} \tag{3}$$

The computation of the grids can be done in parallel on multiple processors by setting the *nproc* or executed on GPU. GPU support is only available for CUDA capable devices and requires NVIDIA driver as well as CUDA toolkit to be installed.

The grid is created to include all atoms of the molecule and 5 a.u. margin, from the outermost atoms. To further decrease computation time in each voxel, the effect of atoms at greater distance than 15 Bohr are neglected.

The RhoCalc module is included with the ADMAPack and the ShapeGPack.

4.2 Running calculations

There are multiple options on how to run RhoCalc calculations, these are:

Via job file For doing calculations by job files, run the abiParse in the terminal or command line with the job file as the system argument. For example in Linux:
./path/to/ADMAPack/abiParse path/to/job/file/job_file.job

Via the ServerPack To run the RhoCalc computations on a server, make sure to have the ServerPackinstalled, then copy the job file into the input directory of the ServerPack.

4.3 Job parameters [keyword:Rhocalc]

To create a RhoCalc job file use the AbiView or use any text editor, following the guidelines in section 7. The keyword for RhoCalc calculations is **Rhocalc**. The parameters for these jobs are:

molecule file Name of input molecule file.

molecule dir Directory of molecule file.

resolution Resolution of grid, in a.u.

GPU (Integer) Use CUDA capable GPU device for calculations. (1 - use GPU, 0 - do not use GPU; defaut=0)

nthreads (Integer) The number of threads/block to use in the case of GPU computations (it is recommended that you use a multiple a 32, as this is the WARP size).

nproc (Integer) Number of processors to use in case of non-GPU computations. (default= edit in systemSettings file)

ed (Integer) Perform electron density calculation. (1 - yes, 2 - no)

ep (Integer) Perform electrostatic potential calculation. (1 - yes, 2 - no)

fragment (Comma separated integers) Comma separated list of atom indices, which will be included in the calculation, write 0 to use the whole molecule. (default=0)

grid file Name of output grid info file.

grid dir Directory of output grid file, if unspecified scratch directory will be used (edit scratch directory location in systemSettings file).

5 AbiView

The GUI package provides an easy way to create job files, prepare input files and evaluate results from Ab Initio calculations. The modules are presented in more detail in the following subsections. Note: the default save directory of each module is the scratch directory, which can be set in the systemSettings file.

5.1 Job file creators

These modules are designed to help the process of creating job files for the ADMAPack, ShapeGPack and the RhoCalc module. Easy to understand graphic user interfaces are provided for each module, in which the user can select all parameters of the jobs. For a detailed explanation of the parameters read the section of the respective module (sections: 2, 3 and 4), or read the help menu of the user interface.

5.2 PDBPrep

The PDBPrep module is used to create input '.pdb' files for Adjustable Density Matrix Assembler calculations, by adding remarks which contain the information about the charges of the molecule. The structures that you input must contain hydrogen atoms, have water residues and any other surplus of atoms or molecules removed. The atoms and residues must be numbered by their physical placing, from the N terminal to the C terminal. It is also recommended that the structures be optimized for faster processing and more precise computations. **Notice**: the atom names *N* and *C* atom labels are reserved for the residues N and C terminal atom. Labeling any other atom with *N* or *C* will result in errors in computation.

The provided *GuessTer* and *GuessRes* functions automatically detect charges based on the resCharges.txt file found in the settings directory.

5.3 ToMol

The module converts files into '.mol' format. Supported file types are: '.log' and '.fchk' files from Gaussian software calculations and '.log' files from Gamess software calculations.

5.4 CubeGen

The module converts grid files into cube files, for this just input the grid file and select which of the grids available you wish to convert. The grid types are:

ED Electron density grid.

EP Electrostatic potential grid.

EV Curvature eigen values grid. Pair this cube file with the electron density to view the partitioning of curvature domains of the isosurfaces for the selected reference curvature value.

5.5 CompMap

Use this module to compare map files obtained from the Shape Group Analysis calculations. The resulting tables comparison values can be viewed in text format or LATEX format.

There are two methods available to perform this comparison: a simple element-by-element comparison, and a method based on the Manhattan distance (taxicab geometry). The first method returns the percentage of matching elements of the shape maps. In this method the neighbors of each element are taken into account, with a factor of 0.5, if is not a direct match. The method based on a Manhattan distance takes compares each element to all others, factoring by the inverse of the Manhattan distance of the closest matching element. The algorithm goes over all the elements of the first map, checking for matching values in the second map. The Manhattan distances are calculated for each of the matching valued elements, and the shortest Manhattan distance is selected. This value is used to weigh the comparison value.

$$d_{Manhattan}(ShapeMap_{ij}, ShapeMap'_{i'j'}) = |i - i'| + |j - j'| \tag{4}$$

$$D_{ij} = \{d_{Manhattan}(ShapeMap_{ij}, ShapeMap'_{i'j'}), ShapeMap_{ij} = ShapeMap'_{i'j'}\} \tag{5}$$

$$C_{ij} = \frac{1}{1 + min(D_{ij})} \tag{6}$$

$$Comp = \frac{\sum_{i,j} C_{ij}}{N_i N_j} \tag{7}$$

5.6 CarboIndex

The module is used to compare electron density grids of the same size, shape, position and resolution, using the Carbo index similarity measure. A discrete version of the original method was used, which fits grid comparison.

$$R_C = \frac{\sum_{i_x} \sum_{i_y} \sum_{i_z} \rho^A_{i_x i_y i_z} \rho^B_{i_x i_y i_z}}{[\sum_{i_x} \sum_{i_y} \sum_{i_z} (\rho^A_{i_x i_y i_z})^2]^{\frac{1}{2}} [\sum_{i_x} \sum_{i_y} \sum_{i_z} (\rho^B_{i_x i_y i_z})^2]^{\frac{1}{2}}} \tag{8}$$

Notice: The comparison values are only valid if the grid are the same.

6 ServerPack

The package is designed to automate job runs and to distribute jobs among the nodes of a server. It contains two main modules: server_root and server_branch. A single server_root module can communicate with multiple server_branch modules.
Notice: The package is only compatible with Linux servers.

6.1 server_root

The module is designed to run in the background and checks for new input files in the input directory (this directory can be specified in the serverSettings.txt file). The server root queues the jobs and distributes them among the running server_branch modules with which it is paired. This pairing is done by editing the serverSettings.txt file and adding the IP address as well as the Port number of the branches. The details on how to edit this file can be found in section 8.

6.2 server_branch(es)

The module is also designed to run in the background, waiting for jobs from the server_root module. If the necessary modules to run the calculation are not present in the package, then the job will be skipped. When starting the module specify the port number of the node with which the communication shall be performed. This number must coincide with the port number specified in the setting file of the server_root module. Make sure that the port number that you are allocating to the program is not reserved by any other system processes! In most cases any number above 8000 is usable. The results of each calculation will be saved in the output directory. Change the location of this directory by editing the serverSettings.txt file. Set the input directory in the serverSettings.txt file to match the input directory specified for the server_root.

6.3 Running the ServerPack

Copy the contents of the package onto the head node of the server, edit the serverSettings.txt file and start the server_root script via the terminal:

./path/to/abinitio/package/server_root

Copy the server_branch script alongside the serverSettings.txt into a working directory of the Ab Initio Research Services LTD. packages. Edit the newly created serverSettings.txt file and run the program:

./path/to/abinitio/package/server_branch -port:port_number -loglevel:log_level

Copy the job files to the input directory, and wait for the calculations to finish. The results to appear in the output directory. The log_level parameter determines the information printed by the program, the levels are:

0 No-printing (default)

1 Data-print – information about the studied system (data about loaded molecules, grid sizes, shape map values etc.)

2 Debug-print – debug information ab

3 Time-print – computational times of the processes.

4 Data-print + Time-print (1 and 3)

5 Debug-print + Time-print (2 and 3)

7 Abinitio Job Files

Job files can be directly submitted to abiParse, which will sort calculations and run the needed modules or copied into the input directory of the ServerPack. These files can be written manually (always use the '.job' extension) or creating using the respective modules of the AbiView.

7.1 Format

A job file may contain multiple tasks. A task is a computation by any of the Ab Initio modules. Each task has three parts: a start line, parameter lines and an end line. The start line consists of the start keyword, followed by the keyword of the module. The module keywords are:

ADMA Adjustable Density Matrix Assembler calculations. (Section 2)

ShapeGroup Shape Group Analysis computations. (Section 3)

Rhocalc Calculate molecular properties. (Section 4)

The start line is followed by the parameter lines. These lines hold the variables, file names and locations needed to run the computation. Each parameter must be written in a new line. Such a line must contain, in this order: the parameter name (a detailed list of these names are given in the section of the respective module); the separator character (=) and the value of the parameter. The enumeration is finished by end line, which only contains the **end** keyword. All lines starting with # character are considered as comments and are omitted by the programs. The programs will skip any computation which requires a module that is unlicensed. Examples of job files can be found in the following subsection (7.2).

7.2 Examples

ADMA calculation followed by Rhocalc calculation

```
# This is the start line:
start ADMA
# The following lines are the parameter lines:
pdb file= pdb_file_name.pdb
pdb path = C:\path\to\pdb
# For Linux systems use: /path/to/pdb
scheme =1
# Mezey-Mulliken scheme
selection radius = 5.0
method= RHF
basis set = 3-21G
# Level of theory: RHF/3-21G
sp program = 1
# Use Gaussian software
nproc =4
memory=2.5
# Allocate 4 processors and 2.5 GBs memory to single point calculations
```

fragment dir=C:\scratch\path\to\fragment\dir
# For Linux: /scratch/path/to/fragment/dir
chargeconv=1e-3
# Terminate calculations if Mulliken charges reach convergence
# smaller than 10^{-3}
save name=output_molecule_file_of_adma.mol
save path = C:\path\to\output\molecule\dir
For Linux: /path/to/output/molecule/dir # This is the end line:
end
start rhocalc
Compute electron density of the previous molecule with RhoCalc
molecule file = output_molecule_file_of_adma.mol
molecule dir=C:\path\to\output\molecule\dir
# Use output file of ADMA calculation as input for RhoCalc
resolution=0.5
# Grid of 0.5 a.u. resolution
GPU = 1
nthreads=512
# Use CUDA capable device to perform computation with 512 threads/block.
ed=1
ep=0
# Compute only electron density of molecule.
fragment = 0
# Perform the computation for the full molecule.
grid file = output_grid_info_file_of_rhocalc.gr
Unspecified *grid dir*, the results will be written to the scratch directory.
end

ShapeGroup calculation

start ShapeGroup
Example of Shape Group Analysiscomputation
molecule file = molecule_file_name.mol
molecule dir = C:\path\to\pdb
# Linux systems: /path/to/pdb
resolution = 0.25
GPU = 1
nthreads = 256
# Perform the calculation of the electron density on the GPU, with 256 threads/block
nproc = 6
# Allocate 6 processors for to the rest of the algorithm
fragment = 1, 9, 6
# Only use atoms: 1,6,9 for calculations
grid file = output_grid_info_file_of_shapegroup.gr
grid dir = C:\path\to\grid
map file = output_map_file_of_shapegroup.gr
map dir = C:\path\to\map
For Linux systems: /path/to/grid/ and /path/to/map end

8 Settings files

The settings files are found in the settings directory of the main directory of the program. These are simple text files, that must be edited with caution. The settings files are:

systemSettings.txt Contains system specifications.

constans.txt Contains values of the constants used in the computations.

admaSettings.txt Contains basic default settings for ADMA calcualtions.

resCharges.txt Contains a list of default residue charges, used in by the AbiView when guessing charges for ADMA job files.

serverSettings.txt Contains settings for the ServerPack.

In the settings files each parameter must appear in a new line. Each line contains the parameters name, followed by the separator character (=) and then the value of the parameter. All parameters are described in detail in the following subsections. A line starting with ### are treated as comments and are omitted by the programs.

8.1 System settings

This file contains system specifications used for all the calculations by the package. The parameters of this file are:

scratch_dir An absolute path to the scratch directory. All files which have unspecified save directories will be written here.

nproc (Integer) The number of processors which will perform calculations in parallel.

nthreads (Integer) The number of threads/block that are allocated in case of GPU computations. The GPU computations are only available for CUDA capable devices. (It is recommended to use a multiple of 32, which is the WARP size of the GPU)

mem (Floating point value) The GBs of memory, which will be allocated by default to calculations.

gaussian_command The command with which the Gaussian software is run from command line or terminal.

formchk_command The command with which the formchk module of the Gaussian software is run. The formchk is used to convert checkpoint files into formatted checkpoint files.

unfchk_command The command with which the unfchk module of the Gaussian software is run. The unfchk is used to convert formatted checkpoint files into checkpoint files.

rungms_command The command with which the Gamess software is started from the terminal or command line.

rungms_version The version number of Gamess software, which was given when compiling the program.

gamess_scratch The scratch directory of the Gamess software.

8.2 Constants

The file contains the values of constants used in the calculations, these are:

Bohr_to_angstrom The conversion factor for Bohr to Angstrom conversion.

Bohr_to_meter The conversion factor for Bohr to meter conversion.

Cm_to_Debye The conversion factor for Coulomb meter to Debye conversion.

electron_charge The charge of an electron, given in Coulombs.

Coulomb_constant The value of the Coulomb constant, given in Nm^2C^{-2}

volts_to_au The conversion factor for Volt to a.u. conversion.

angstrom_to_Bohr The conversion factor for Angstrom to Bohr conversion.

8.3 ADMA settings

The file contains specifications and default values for ADMA calculations. The parameters are:

nTerCharge (Integer) The default value of the N terminal charges.

cTerCharge (Integer) The default value of the C terminal charges.

selRadius (Floating point value) The default value of the selection radius.

NH_bond_length (Floating point value) The value of the nitrogen-hydrogen bond length, given in Angstroms.

CH_bond_length (Floating point value) The value of the carbon-hydrogen bond length, given in Angstroms.

SH_bond_length (Floating point value) The value of the sulfur-hydrogen bond length, given in Angstroms.

8.4 Residue charges

This file contains contains the default values of the residue charges used when preparing '.pdb' files with the PDBPrep module. The residues are specified by their three character code and an integer value for the charge of the residue.

8.5 Server settings

The file contains specifications host-client run of the programs. These files need to be edited both in the server branch and server root directories. The parameters contained in the file are:

input_dir The directory where the server branch will search for input files.

output_dir The directory where the server branch will save the output files.

sleep_time (Floating point value) The number of seconds that the algorithm will wait before rechecking the input directory for new files.

queue_max (Integer) The maximum number of files that will be queued by the server branch.

servers (Comma separated list) The IP addresses and port numbers of available server branches. The server branch addresses are listed as [IP address:Port number], these are listed in comma separated format.

BUFF_SIZE (Integer) The buffer size of server branch - server root communications, given in bytes.

9 Symbols

Symbols and their explanation used in the document:

$\boldsymbol{\Psi}$ wave function

i, j, k, l indices used for denoting atomic orbitals

a index for denoting atoms

i_x, i_y, i_z indices for denoting voxels

Q^M Mulliken charge

$\boldsymbol{P}$ density matrix

$\boldsymbol{\rho}$ electron density grid

$\boldsymbol{E}P$ electrostatic potential grid

$\boldsymbol{H}^{core}$ core Hamiltonian matrix

$\boldsymbol{F}$ Fock matrix

V_{NN} nuclear repulsion energy

E_{HF} Hartree-Fock energy

R_C Carbo index

If the summations have no limits, then they are done over all possible values.