Miroslav Kohout

DGrid 4.5

– User’s Guide –

Springer

Preface

Like in all the previous versions the package experienced again some more or less cosmetic improvements in the program structure and few rather small changes in the format structure of the output files. DGrid 4.5 includes new routines covering two important topics – the search for the critical points and the evaluation of overlap integrals.

The search for critical points is based on approach very different from the one used in the 4.4 version. Now the whole grid field is searched and the basins are no more necessary for the choice of the starting points (but still used for convenient descriptors). Besides the attractors and saddle points also the ring critical points as well as the minima are identified and evaluated. The routines work reliable for the electron density. For steep properties, like ELI-D, the program searches only outside (predefined) atomic core regions to avoid numerical instabilities.

New feature in DGrid is the analytical calculation of overlap integral which can be evaluated when Gauss-type basis is used within grid boxes oriented parallel with the Cartesian axes. The overlap integrals enables the calculation of fluctuation and delocalization indices over basins.

Additionally, the source code and the description of the implementation and usage of the interface to the visualization program Avizo is a part of the DGrid package. Avizo is relatively expensive (but excellent) program and thus will probably not suit everybody's potentiality. With the interface to the free visualization program OpenDX another possibility is offered by the package. The interface to OpenDX as well as the corresponding part of the manual was written by Dr. Alexey I. Baranov. He also made lot of testing of DGrid and found many of the incompatibilities occurring during the development process.

The manual was written with Latex using the SVMono style. The permission of Springer is acknowledged.

Rudimentary help for certain parts of DGrid is given by calling:

```
dgrid -h
```

which prints the following information:

```
--------------------------------------------------
dgrid  -p                          print property keywords
dgrid  -s                          print spin keywords
dgrid  -t                          print type keywords
dgrid  -u                          utilities
dgrid  -v                          print DGrid version
--------------------------------------------------
```

Radebeul, September 2009 *M. K.*

Contents

List of Tables

List of Figures

Acronyms

AO	atomic orbital
CP	critical point
DAFH	domain-averaged Fermi-hole
DFT	density functional theory
ELI	electron localizability indicator
ELI-D	electron localizability indicator based on D-restricted partitioning
HF	Hartree-Fock
ICL	interconnection line
MO	molecular orbital

Chapter 1
Property Grids

1.1 Disclaimer

The program DGrid is distributed ‘as is’ without warranties of any kind. No responsibility of any kind is assumed from the use of DGrid. Use the following citation for the program DGrid:

> M. Kohout, DGrid, version 4.5, Radebeul, 2009

1.2 Distribution

DGrid source code is distributed free of charge. The code remains in the ownership of the author. The copyright must not be changed when adapting the code. DGrid can be freely redistributed. However, all changes and corrections implemented after the latest distribution will be sent only to registered users.

1.3 Installation

The program DGrid is written in C++, i.e., an appropriate compiler must be available. The installation on a Unix system and on a Windows PC (using MS Visual Studio) is described.

1.3.1 Linux

Unpack the distribution tar file dgrid-4.5.tar.gz using the commands:

```
gunzip  dgrid-4.5.tar.gz
tar xf  dgbas-4.5.tar
```

This creates new directory *dgrid-4.5* (the file *dgrid.sln* is needed only for Windows installation). Change to the directory *dgrid-4.5*. Besides the directory *doc* with the documentation and examples there should be the directory *src* with all the source code files and a makefile (the file *dgrid-4.5.vcproj* is needed only for Windows installation). View the makefile - you may have to change the name of the compiler (for instance xlC on the IBM-AIX) and the compiler options. DGrid will be created with the command:

```
make
```

The executable `dgrid-4.5` will be written to the directory $HOME/bin (this can be changed in the makefile). For convenience, link this executable with the name dgrid:

```
ln -s  dgrid-4.5  dgrid
```

Run the DGrid calculation with the command:

```
dgrid  controlfilename
```

The *control* file name is given as a parameter (i.e., without '<'). The output goes to the console, unless the *output* file name is given in the *control* file, cf. the Sec. 1.6, 'Control file', page 6).

1.3.2 Windows

In this section the compilation of DGrid with Visual Studio 8 (VS8) is described. Change to the directory where you wish to save the Visual Studio Project, for instance *C:\Mydir*. Create an empty directory named *dgrid* in which the distribution tar file dgrid-4.5.tar.gz has to be unpacked with an appropriate Windows tool (like

Wintar or 7-zip). Change into the directory *dgrid*. You should find there the project file *dgrid.sln* and the directory *dgrid-4.5*. The directory *dgrid-4.5* contains, besides the directory *doc* with the documentation and examples, the directory *src* with all the source code files and the file *dgrid-4.5.vcproj*.

Start the VS8 and open the project *dgrid* using the project file *dgrid.sln*. Build the project *dgrid*. This compiles all the files and creates the executable *dgrid-4.5.exe* which can be finally found in the directory *C:\Mydir\dgrid\Release*.
Run the DGrid calculation with the command:

```
C:\Mydir\dgrid\Release\dgrid-4.5.exe  controlfilename
```

The output goes to the console, unless the *output* file name is given in the *control* file, cf. the Sec. 1.6, ‘Control file’, page 6).

1.4 Introduction

DGrid is a program for the generation of property values on an equidistant grid (respectively at a single point only) and the analysis of those gridded fields, inclusive the basin search. For the calculation of property values two input files are essential:

1. *basis* file with the information about the basis set and the molecular orbital representation (for CI calculation additionally also the density matrices) for the evaluated atom or molecule. The atomic or molecular data files supplied by one of the quantum chemical packages GAUSSIAN [13], MOLPRO [34], MOLCAS [4], TURBOMOLE, and ADF [1], respectively, need to be converted into the DGrid *basis* file format. This conversion is described in the next section.
2. *control* file (written by the user) with keywords controlling the property value calculation.

Both *basis* and *control* file must be written in a special format. The information is read in by the parsing routine `rdcard` that obeys few simple rules:

- Empty lines and lines starting with a colon are skipped.
- Strings and numbers terminated by a colon are skipped.
- Strings and numbers must be separated by one or more blanks.
- Two colons at the beginning mark the line as a text input – the line is read in without parsing (e.g., as a title).
- String followed by ‘=’ is a variable. The string or number following the equal sign is assigned to this variable.

1.5 Basis file

Each of the quantum chemical packages GAUSSIAN, MOLPRO, MOLCAS, TURBOMOLE, and ADF provide an option to write the information about (among others) the molecular geometry, basis set, i.e., the atomic orbitals (AO), and the resulting molecular orbitals (MO) to a separate output file. Those output files are written in different formats. Additionally, ADF is based on Slater-type orbitals, whereas the other programs use Gauss-type orbitals.

GAUSSIAN calculations supply the formatted *name*.wfn file as well as the binary check-point file *name*.chk, where the latter need first to be converted into the formatted check-point file *name*.fchk using the GAUSSIAN utility `formchk`. Similarly, the density functional program ADF writes the required information to a binary file TAPE21, that needs to be converted into a formatted file using the ADF utility `dmpkf`. For MOLPRO, MOLCAS, and TURBOMOLE the necessary information need to be written to a file in *molden* format. In case of TURBOMOLE include in the *molden* file as the second line the string:

```
[TURBOMOLE]
```

otherwise the normalization of d and f functions will be wrong.

DGrid reads the necessary information from the *basis* file written in a special format. Thus, the quantum chemical packages output files have to be converted into this format by the conversion routines included in the DGrid package:

```
dgrid  formatted-qm-file
```

The resulting DGrid *basis* file includes all orbitals from the quantum chemical calculation (occupied as well as the virtual ones). Of course, in the DGrid *basis* file the unoccupied orbitals have occupations equal zero. To include these orbitals into the property calculation change the occupations manually in the DGrid *basis* file (or in the *control* file using the keyword **occupation**). However, virtual orbitals cannot be used for partial ELI-D.

Find in the *examples* directory of the DGrid package the formatted GAUSSIAN03 checkpoint file C3H3NO_HF_6-31G.fchk for the oxazole molecule (geometry from Ref. [29] re-optimized with GAUSSIAN03). Using the command:

```
dgrid  C3H3NO_HF_6-31G.fchk
```

yields the DGrid *basis* file file C3H3NO_HF_6-31G.g03 with 51 orbitals (6-31G basis set). Of course, only 18 orbitals are (doubly) occupied.

Table 1.1 Basis file name extender subject to QM package used

Program descr.	Source		Extender
GAUSSIAN94	Checkpoint file	↦	g94
GAUSSIAN98	Checkpoint file	↦	g98
GAUSSIAN03	Checkpoint file	↦	g03
GAUSSIAN	WFN file	↦	gwf
MOLPRO	molden file	↦	mp
MOLCAS	molden file	↦	mc
MOLDEN	molden file	↦	md
ADF	TAPE21	↦	adf
ADF	TAPE21 (loc. orbitals)	↦	adf_loc
CLEMENTI-ROETTI	Tables [10]	↦	CR

The first line of the *basis* file starts with the keyword 'BASISFILE'. Additionally, the version of DGrid creating the data is included. The *basis* file contains the information about the package used for the quantum chemical calculation as well as the date, time and the title of the calculation, followed by the coordinates of the involved atoms and the assignment of the basis sets to the atoms. After the header 'MO DATA' the descriptor and the atomic orbital expansion for each symmetry is given (sym: NONE means orbital output without symmetry descriptors). In the AO expansion the number and the descriptor point to the atom and the corresponding AO, respectively, in the basis set list. For example '3 pz_a' is the p_z AO on the atom number 3 (nitrogen), with the exponents and coefficients of the second *p* AO on this atom (the first one is without the character appended). The AO expansion descriptors are followed by the energy and the MO coefficients for each orbital in the symmetry. The *basis* file C3H3NO_HF_6-31G.g98 from the above example reads:

```
BASISFILE   created by DGrid version 4.5   07-08-2009

program: GAUSSIAN03        date: Fri Aug  7 15:58:46 2009

:title
::C3H3NO HF 6-31G

:atom  No.         x          y          z          charge
:-----------------------------------------------------------
  O     1:      -2.0618     0.5748     0.0000         8.00
  C     2:       0.0000     2.1211    -0.0000         6.00
  N     3:       2.0869     0.8955    -0.0000         7.00
  C     4:       1.4135    -1.6748    -0.0000         6.00
  C     5:      -1.1101    -1.8596     0.0000         6.00
  H     6:      -0.3077     4.1025    -0.0000         1.00
  H     7:       2.8049    -3.1219    -0.0000         1.00
  H     8:      -2.4310    -3.3680     0.0000         1.00
:-----------------------------------------------------------


:                      +-------------------------------+
:                      :   calculation:   RESTRICTED   :
:                      +-------------------------------+

Energy=   -244.50273 Hartree              Electrons=   18 alpha  +    18 beta
```

```
:                          +-------------------------+
                           :  basis:  CARTESIAN  GTO  :
:                          +-------------------------+

: atom  No.       type           exponents and coefficients
:-------------------------------------------------------------------------------
   O     1          s    exp:     5.48467166e+03    8.25234946e+02    1.88046958e+02
                         coe:     1.83107443e-03    1.39501722e-02    6.84450781e-02
         ...
   H     6 -   8    s    exp:     1.61277759e-01
:-------------------------------------------------------------------------------

:                                +------------+
:                                |  MO  DATA  |
:                                +------------+


sym: NONE              1  s           ...       1  py          1  pz
                          ...          ...       ...            ...

:orb       energy     coefficients
:-------------------------------------------------------------------------
   1     -20.65207    0.9958683260      ...    -0.0005791664    0.0000000000
                         ...            ...            ...       ...
...
  51       2.16106    0.0133842933      ...     0.0285089560    0.0000000000
                         ...            ...            ...       ...
:-------------------------------------------------------------------------
:end of MO data

:                        +------------+
                         | OCCUPATION |
:                        +------------+

:-------------------------------------------------------------
:  #  symmetry         orb          ALPHA           BETA
:-------------------------------------------------------------
   1  NONE              1      1.00000000000    1.00000000000
      ...
  51  NONE             51      0.00000000000    0.00000000000
:-------------------------------------------------------------
```

Notice that the molecular orbital number 51 has positive orbital energy (like all other virtual orbitals in the *basis* file). The MO expansion coefficients are followed by the occupation section. Only the first 18 orbitals are occupied. For an actual DGrid calculation the *basis* file can be renamed to any name. The name of the particular *basis* file is known by the DGrid program through the assignment '**basis**=*basisfilename*' in the *control* file described in next section.

1.6 Control file

For the *control* file any name can be chosen. The *control* file supplies all the information concerning the calculation. The data must comply with the rules described in the introduction. Following is the example *rho.inp* for the *control* file (see the *examples* directory):

```
:TITLE
:-----------------------------------------------|
::C3H3NO   HF    6-31G
:-----------------------------------------------|

:KEYWORDS
:-----------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 output=C3H3NO_run

:CHOOSE THE DESIRED PROPERTIES
:---------------------------------------
 compute=rho
 compute=rho      laplacian
:---------------------------------------

 mesh=0.05   4.0

END
```

The first readable information in the *control* file must be the title (remember that all empty lines as well as lines beginning with a single colon will be skipped). The title line starts with two colons, i.e., the line will not be parsed. The two colons are obligatory, whereas the title text can be omitted (i.e., empty title). If some title text is found in this line, it will be written into the header of each *property* file.

The keyword **basis** is followed by the name of the *basis* file (described in the previous section). This can also be a UNIX path to this file.

The keyword **output** is followed by the generic name of the *output* file. To this name the extension '.dg' will be appended (for the above example the output will be written to the file *C3H3NO_run.dg*). Using the assignment '**output=.**' the name of the *output* file would be generated from the *basis* file name (i.e., the file name *C3H3NO_HF_6-31G.g03.dg* would be generated from the *basis* file *C3H3NO_HF_6-31G.g03.g98*). Omitting the **output** keyword will send the output to the console. In the *output* file some setup information and the calculation progress is shown. For a single point calculation (see the keyword '**point**=' in Sec. 1.8) all property data will be printed to the given output file.

At least one **compute** assignment, choosing the property to be computed, is obligatory for a grid calculation. With the keyword **compute** the desired property to be computed is chosen. Each property must have its own '**compute**=property' assignment. In the above example the electron density and its Laplacian will be computed. Let as have a closer look on the **compute** assignments. The general form is (case insensitive):

compute=property <type> <spin> <pair>

Depending on the combination of the descriptors the *result* file name will be affected. As already shown above for the density Laplacian, in the *result* file name the string 'lap_' will precede the property name. With the descriptors <spin> and <pair> additional information will be appended to the *result* file name. In Tables 1.2 to 1.4 the possible descriptors are given together with the strings (extenders) that will be appended to the *result* file name (exemplary for the property *prop* and the generic name *filename* in coordinate space, i.e., with '_r' appended).

Table 1.2 Grid file name extender subject to property type

Property type		File name[a]
	↦	*filename.prop_r*
gradient-mag	↦	*filename*.**grad-mag**_*prop_r*
gradient-vec	↦	*filename*.**grad**_*prop_r*
ln-derivative	↦	*filename*.**ln_deriv**_*prop_r*
hessian	↦	*filename*.**hess**_*prop_r*
laplacian	↦	*filename*.**lap**_*prop_r*
relative-laplacian	↦	*filename*.**rel-lap**_*prop_r*

[a] Exemplary for the generic name *filename* and property *prop* in real space

If <type> is omitted, cf. the first line in Tab. 1.2 then the property value will be computed (and the *result* file named *filename.prop_r*). The <type> descriptors can be listed with the command '`dgrid -t`'.

Table 1.3 Grid file name extender subject to property spin

Property spin		File name[a]
alpha	↦	*filename.prop_r*_**a**
beta	↦	*filename.prop_r*_**b**
both	↦	*filename.prop_r*
singlet	↦	*filename.prop_r*_**s**
triplet	↦	*filename.prop_r*_**t**

[a] Exemplary for the generic name *filename* and property *prop* in real space

If <spin> is omitted, then total spin will be used for 1-particle property (i.e., the default value is `both`, cf. Tab. 1.3). For exceptions from this rule, as well as the choice in case of 2-particle property see remark 1.1 below. The <spin> descriptors can be listed with the command '`dgrid -s`'.
If <pair> is omitted, then the default value is either `unknown` (in which case the file name is created as for `spinless`, cf. Tab. 1.4) or the value is spin dependent, see remark 1.1 below. The <pair> descriptors can be listed with the command '`dgrid -s`'.

Remark 1.1. The descriptors can be omitted when calling the **compute** command. Then the default values for <type>, <spin>, <pair> will apply. In case of <type>

Table 1.4 Grid file name extender subject to property pair spin

Property pair spin		File name[a]
alpha-alpha	↦	*filename.prop_r_***aa**
beta-beta	↦	*filename.prop_r_***bb**
alpha-beta	↦	*filename.prop_r_***ab**
beta-alpha	↦	*filename.prop_r_***ba**
singlet-pair	↦	*filename.prop_r_***sg**
triplet-pair	↦	*filename.prop_r_***tr**
spinless	↦	*filename.prop_r*

[a] Exemplary for the generic name *filename* and property *prop* in real space

the property value will be computed. For <spin> and <pair> the default values depend on the property. For a 1-particle property (like the electron density, kinetic energy density, local source, etc.) the <spin> descriptor is set to `both`, except for '**compute**=phi' and overlap integrals, in which case the majority spin `alpha` is chosen. For 2-particle property the <pair> descriptor is set in accordance with the <spin> descriptor, i.e., `alpha-alpha` pair spin if `alpha` spin is given, etc. (if <spin> given, otherwise `spinless` is used). If only <pair> descriptor is given, then the <spin> descriptor is set accordingly (for the <pair> descriptor `spinless` the <spin> descriptor `both` is used).

The *property* file names are generated from the generic file name (which is either the *basis* file name or the file name assigned by the keyword **result**, see below). For each property a unique file extension, given in Table 1.5, will be appended to the generic file name. Additionally, for properties computed in coordinate space the descriptor '_r' will be appended (for calculations performed in momentum space – keyword '**space**=momentum' – the descriptor '_p' would be appended). In the above *control* file the properties are given by the assignments '**compute**=rho' and '**compute**=rho laplacian', i.e., the total electron density (summing up both spins) and the total density Laplacian is chosen. The two *property* files will be named *C3H3NO_HF_6-31G.g03.rho_r* and *C3H3NO_HF_6-31G.g03.lap_rho_r*, respectively (because of the assignment **basis**=*C3H3NO_HF_6-31G.g03*).

As already mentioned, the generic file name can be changed using the keyword **result**. For instance, with the assignment '**result**=*C3H3NO*' the two *property* files would be named *C3H3NO.rho_r* and *C3H3NO.lap_rho_r*, respectively.

In any case, files written by DGrid will not overwrite already existing files. If a file exists, DGrid will create new file name by appending '_1' to it (and successively higher numbers). The *property* files will be written in the DGrid format (special header and 5 values per line). This can be changed with keyword **format** (cf. Sec. 1.8).

Per default all properties are computed in the real (coordinate) space. With the assignment **space**=momentum it is possible to perform the calculation of the desired properties in the momentum space. At the time the transformation into the momentum space is done only for Gauss-type orbitals.

The assignment '**mesh**=0.05 4.0' defines the grid region. The keyword '**mesh**=' is followed by the mesh-size, i.e., the distance between neighboring grid points (the units from the *basis* file are used which is usually set to bohr). Additionally, a border around the initial box can be given. In the example a grid with the distance between neighboring points of 0.05 bohr and a border of 4.0 bohr will be created around the C_3H_3NO molecule. The program works as follows (cf. Fig. 1.1 on page 25). From the atomic positions the average coordinate is computed. The 'initial box' (the smallest box around the molecule enclosing all atoms) is created. The box sides are parallel to principal axes of the molecular rotational ellipsoid (however, as given by the example, not necessarily parallel with the coordinate axes – to achieve this add the keyword **parallel**). The 'initial box' is enlarged in all three directions by the border of 4.0 bohr (the grid region dimensions being multiple of the mesh-size of 0.05 bohr). This results in a grid of $266 \times 309 \times 161$ points (with the dimensions of 13.25 bohr $\times$ 15.4 bohr $\times$ 8.0 bohr, cf. the lower diagram in Fig. 1.1).

Following example for the *control* file showing another possibility to define the grid region is given by the file *elid_vec.inp* (see the *examples* directory):

```
:TITLE
:-----------------------------------------------|
::C3H3NO   HF    6-31G
:-----------------------------------------------|

:KEYWORDS
:-----------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 output=.


:CHOOSE THE DESIRED PROPERTIES
:---------------------------------------
 compute=ELI-D    triplet-pair
:---------------------------------------



 GRID_DEFINITION:   vectors
:----------------------------
:             X     Y     Z
:----------------------------
 origin:    -6.0  -7.0  -4.0
                                INTERVALS:
:-------------------------------------
 i-vector:  12.0   0.0   0.0     120
 j-vector:   0.0  14.0   0.0     140
 k-vector:   0.0   0.0   8.0      80

END
```

Here, the grid region is explicitly given by three vectors. The definition starts with the keyword **vectors**. It is followed by a line with three floating point numbers

for the coordinates of the region origin. Next three lines contain the data for three vectors spanning the chosen volume (i.e., the shift with respect to the region origin, NOT the endpoints of the vectors) as well as the number of intervals along each vector. In the example an orthogonal grid (parallel to the Cartesian axes) of 121 $\times$ 141 $\times$ 81 points in a box of 12 bohr $\times$ 14 bohr $\times$ 8 bohr (i.e., 0.1 bohr mesh) with lower left vertex at the coordinates [-5.0, -5.0, -8.0] will be computed.

The assignment '**output=.**' in the above example directs the program to write the output data into the file *C3H3NO_HF_6-31G.g03.dg* as determined by the *basis* file name. The '**compute**=ELI-D triplet-pair' assignment yields a data grid written to the *result* file named *C3H6_test.elid_r_t_tr*, showing that in this case ELI-D is based on the charge of triplet-coupled electrons ('_t' part) that is needed to form a fixed fraction of triplet pair ('_tr' part) [23]. Additionally, it can be seen that in this case the <spin> descriptor is set automatically to 'triplet' (similarly, for the assignment '**compute**=ELI-D triplet' the <pair> descriptor would be set to 'triplet-pair').

The properties are computed at every grid point from all occupied orbitals. The occupations are given in the *basis* file for each orbital of particular symmetry. To compute a property for specified orbitals (for instance the electron density based on certain orbitals) the occupations must be specified after the keyword **occupation**. The data must be written in 1 line for each chosen orbital. First the name of the symmetry must be given, followed by the number of the desired orbital in this symmetry (the number is accessible from the *basis* file) together with the α-spin and β-spin occupations. In the *control* file *rho_orb.inp*, shown below, the orbitals number 17 and 18 of the symmetry 'NONE' (GAUSSIAN output without symmetry), both double occupied, are selected for the calculation. If the occupation input is not the last command in the *control* file then the occupation block must be closed with the **occupation_end** command on a separate line. To compute the amplitude (i.e., not the orbital density) for 1 single orbital, use '**compute**=phi'. In this case only 1 orbital is allowed in the occupation section.

```
:TITLE
:-------------------------------------------------|
::C3H3NO   HF    6-31G
:-------------------------------------------------|

:KEYWORDS
:-------------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 output=.


:CHOOSE THE DESIRED PROPERTIES
:----------------------------------------
 compute=rho
:----------------------------------------

 mesh=0.1  4.0

 USERS_OCCUPATION_INPUT:  occupation
```

```
:-----------------------------------
: SYM      #        ALPHA  BETA
:-----------------------------
 NONE     17        1.0    1.0
 NONE     18        1.0    1.0

END
```

Table 1.5 summarizes the properties that can be calculated (the conditional properties are given in Table 1.6 on page 14). For almost all properties the derivatives can be computed using the <type> descriptors for the it compute keyword, cf. page 8. ELI-D in Table 1.5 is written as $\Upsilon_D = \rho\tilde{V}_D$, i.e., the electron density ρ multiplied by the pair volume function $\tilde{V}_D$. As described in Ref. [18, 32] ELI-D can be exactly decomposed into orbital contributions, the same way as the electron density is given by the sum over orbital contributions.

The calculation of the ELI-D contributions is performed with the property keyword 'pELI-D', for instance with '**compute**=pELI-D alpha', cf. the *control* file *pelid.inp* in the *example* directory. In this case the Υ_D^α, i.e., ELI-D for the α-spin pairs is considered. The contributions of the two highest occupied orbitals to Υ_D^α are chosen with the keyword **occupation**. The pair-volume function $\tilde{V}_D$ for pELI-D is of course computed from the total pair density. The occupations of the chosen orbitals are highlighted in the *output* file. Observe, that although the occupations are given for both spins, only the α-spin part is taken into account. Summing up all the contributions of all orbitals recovers the total ELI-D.

Remark 1.2. pELI-D contributions are possible only for occupied orbitals. DGrid does not compute contributions of virtual orbitals.

Remark 1.3. Using the assignment '**compute**=eli-d alpha' together with the keyword **occupation**, instead of '**compute**=peli-d alpha', yields another quantity, because in this case also the pair-volume function $\tilde{V}_D$ is calculated from the chosen orbitals. This quantity will NOT add up to total ELI-D.

With the assignment '**point**= *x y z*' the calculation is performed at the Cartesian position [*x y z*] only. The desired property value, its gradient, Hessian, eigenvectors and curvatures are printed into the *output* file, resp. to the console. In this case the grid definition given by the **vectors**, respectively **mesh** keyword are not used, cf. the *control* file *cond.inp* used below (with no **output** assignment, i.e., results printed to console):

```
:TITLE
:-------------------------------------------------|
::C3H3NO   HF    6-31G
:-------------------------------------------------|

:KEYWORDS
:-------------------------------------------------
 basis=C3H3NO_HF_6-31G.g03
```

Table 1.5 Available properties

Keyword	Property	Description	Ref.	File ext.
compute=ELF	$\left[1+\chi^2\right]^{-1}$	ELF: 'spin-polarized'	[9, 26, 21]	elf
compute=ELF-cs	$\left[1+\chi_{cs}^2\right]^{-1}$	ELF: 'closed-shell'	[9, 26, 21]	elf_cs
compute=ELF-Kirshnitz	$\left[1+\chi_{kirsh}^2\right]^{-1}$	ELF: Tsirelson	[31]	elf_kirsh
compute=ELI-D	$\rho\tilde{V_D}$	ELI-D	[17, 19]	elid
compute=ELI-q	$\rho_2^{(s)}\tilde{V_q}^2$	ELI-q	[23]	eliq
compute=ELIA	$\rho_2^{\alpha\beta}/(\rho_\alpha\rho_\beta)$	ELIA	[20]	elia
compute=hole-curvature	$\nabla^2\rho_2^{\alpha\alpha}$	Fermi-hole curvature		hole_curv
compute=LOL	$\frac{\tau^{LSDA}}{\tau+\tau^{LSDA}}$	localized orbital locator	[28]	lol
compute=LS	$-\frac{1}{4\pi}\frac{\nabla^2\rho(r)}{\lvert r-r'\rvert}$	local source	[6, 14]	ls
compute=OEP	$\frac{\nabla^2\sqrt{\rho}}{2\sqrt{\rho}}$	one-electron potential	[15]	oep
compute=pair-volume-function	$\tilde{V_D}$	pair volume function	[18, 32]	pair_vol
compute=pELI-D	$\rho_{orb}\tilde{V_D}$	orbital resolved ELI-D	[18, 32]	p_elid
compute=phi	ϕ_i	orbital amplitude		phi
compute=charge-volume-function	$\tilde{V_q}$	charge volume function	[23]	q_vol
compute=rho	ρ	electron density		rho
compute=on-top-density	$\rho_2(r,r)$	on-top density		rho_ontop
compute=rho-spin	ρ_s	spin density		rho_spn
compute=tau	τ	kin. energy density		tau
compute=tp	$\tau - t_w$	Pauli kin. energy dens.		tp
compute=tw	$\frac{1}{8}\frac{(\nabla\rho)^2}{\rho}$	Weizsäcker term		tw

$\rho_s = \rho_\alpha - \rho_\beta$

$g^\sigma = \sum_{i<j}^{\sigma}\sum_{k<l}^{\sigma} P_{ij,kl}\,[\phi_i\nabla\phi_j - \phi_j\nabla\phi_i].[\phi_k^*\nabla\phi_l^* - \phi_l^*\nabla\phi_k^*]$ Fermi hole curvature

$\tilde{V_D} = [12/g^\sigma]^{3/8}$

$\tilde{V_q} = 1/\rho^{(s)};$ $\rho^{(s)}$ density of singlet coupled electrons

$c_F = \frac{3}{10}(3\pi^2)^{2/3}$ Fermi constant

$\tau = \frac{1}{2}\sum_i |\nabla\phi_i|^2$ kinetic energy density of non-interacting system

$\chi = \left[\tau - \frac{1}{8}\frac{(\nabla\rho_\alpha)^2}{\rho_\alpha} - \frac{1}{8}\frac{(\nabla\rho_\beta)^2}{\rho_\beta}\right] / \left[2^{2/3}c_F\,(\rho_\alpha^{5/3} + \rho_\beta^{5/3})\right]$

$\chi_{cs} = \left[\tau - \frac{1}{8}\frac{(\nabla\rho)^2}{\rho}\right] / \left[c_F\,\rho^{5/3}\right]$

$\chi_{kirsh} = \left[c_F\,\rho^{5/3} - \frac{1}{9}\frac{(\nabla\rho)^2}{\rho} + \frac{1}{6}\nabla^2\rho\right] / \left[c_F\,\rho^{5/3}\right]$

$\tau^{LSDA} = 2^{2/3}c_F\,\rho^{5/3}$ originally defined for single spin channel, i.e., ρ_σ [28]

Table 1.6 Conditional properties

Keyword	Property	Description	File ext.
compute=cond-pair-density	$P_{cond}(1,2)$	conditional pair density	pd_cond
compute=DAFH	$f_{\rm av}^{\sigma\sigma}(1)$	domain-averaged Fermi-hole	dafh
compute=hole-density	$\rho_{ex}^{\sigma}(1)$	hole density	hole_density
compute=pair-density	$\rho_2(1,2)$	pair density	pd

$\rho_2^{\sigma\sigma}(1,2) = \frac{1}{2}\sum_{i<j}^{\sigma}\sum_{k<l}^{\sigma} P_{ij,kl}\ \left|\phi_i(1)\phi_j(2)\right|\ \left|\phi_k^*(1)\phi_l^*(2)\right|$

$f_{\rm av}^{\sigma\sigma}(1) = \int_\Omega \rho_2^{\sigma\sigma}(1,2)\mathrm{d}2$

$P_{cond}^{\sigma\sigma}(1,2) = \rho_2^{\sigma\sigma}(1,2)/\rho_\sigma(2)$

$\rho_2^{\alpha\beta}(1,2) = \frac{1}{2}\sum_{i,j}^{\alpha}\sum_{k,l}^{\beta} P_{ij,kl}\ \left|\phi_i(1)\phi_j(2)\right|\ \left|\phi_k^*(1)\phi_l^*(2)\right|$

$P_{cond}^{\sigma\sigma'}(1,2) = \rho_2^{\sigma\sigma'}(1,2)/\rho_{\sigma'}(2)$

$\rho_2(1,2) = \rho_2^{\alpha\alpha}(1,2) + \rho_2^{\beta\beta}(1,2) + \rho_2^{\alpha\beta}(1,2) + \rho_2^{\beta\alpha}(1,2) = \rho_2^{(s)}(1,2) + \rho_2^{(t)}(1,2)$

$P_{cond}(1,2) = \rho_2(1,2)/\rho(2)$

$\rho_{ex}^{\sigma}(1) = 2P_{cond}^{\sigma\sigma}(1,2) - \rho_\sigma(1)$

```
:CHOOSE THE DESIRED PROPERTIES
:----------------------------------------
 compute=rho                 alpha
 compute=cond-pair-density   alpha-alpha
:----------------------------------------

 ref_point= 0.0  0.0  0.5
 point    = 0.0  0.0  0.0

 mesh=0.1  4.0

END
```

With the assignment '**ref_point**= *x y z*' the position of the reference electron is fixed at Cartesian coordinates [*x y z*]. This is necessary for the calculation of the electron pair density (6 dimensional property) and some related properties (for instance the conditional properties), cf. Table 1.6. Then, fixing the position of 1 electron yields a 3 dimensional property. The calculation using the *control* file *cond.input* yields the electron density (together with the gradient and Hessian) at the position [0.0, 0.0, 0.0] (for the electron density itself the reference point has no meaning). Observe, that the (practically) zero gradient and the signature of the curvatures mark this position as a ring critical point. Additionally, the conditional $\alpha\alpha$-pair density at [0.0, 0.0, 0.0] (together with the corresponding gradient and Hessian) for the reference electron fixed at the position [0.0, 0.0, 0.5] is computed.

Similar background follows the calculation of the domain-averaged Fermi-hole, DAFH [24], in which case one of the pair density coordinates is confined to given

region Ω (for instance a basin). To compute DAHF overlap integrals over the confinement region are needed, cf. Sec. 2.7 and the example on page 61.

The last information in the *control* file is the keyword **end** (can be omitted if no data lines are following). Data lines following the keyword **end** are not read. All keywords are read case insensitive. In many cases the keywords can be written in any order often there is no need to write each keyword in a separate line. But it is better to put every keyword separately in a line, because sometimes additional data are assumed per default. Some data even need to be written in rigorous sequence.All available keywords are described in detail in Sec. 1.8 'Keywords').

1.7 Property files

Each property is written to separate file distinguished by a descriptor according to the **compute** assignment, see Tables 1.5 and 1.6. Every *property* file has a header followed by the property values. Following is an example of the *property* file header for the orbital density computed from data supplied by the GAUSSIAN03 package:

```
GRIDFILE    created by DGrid version 4.5   19-08-2009

:job was executed on:         Wed Aug 19 10:39:58 2009

basis_from_program:      GAUSSIAN03              basis=C3H3NO_HF_6-31G.g03

property:   rho[r]                               spin=both     pair=unknown

: property computed from orbitals
:--------------------------------------
:  symmetry         No.    occupation
:--------------------------------------
  NONE             17     1.000   1.000
  NONE             18     1.000   1.000
:--------------------------------------

::C3H3NO   HF    6-31G

:atom       x          y          z
:-----------------------------------
 O      -2.0618     0.5748     0.0000
 C       0.0000     2.1211    -0.0000
 N       2.0869     0.8955    -0.0000
 C       1.4135    -1.6748    -0.0000
 C      -1.1101    -1.8596     0.0000
 H      -0.3077     4.1025    -0.0000
 H       2.8049    -3.1219    -0.0000
 H      -2.4310    -3.3680     0.0000
:-----------------------------------

Energy=    -244.50273 Hartree           Electrons=   18 alpha  +    18 beta

:             lattice vectors
:-------------------------------------
:         a           b           c
:-------------------------------------
x:    1.000000    0.000000    0.000000
y:    0.000000    1.000000    0.000000
z:    0.000000    0.000000    1.000000
:-------------------------------------
```

```
:        origin           v_i         v_j         v_k
:-------------------------------------------------------
x:      -6.2431       13.278611   -0.873034    0.000000
y:      -7.6599        0.753984   15.375234    0.000000
z:      -4.0000        0.000000    0.000000    8.000000

points:                   134         155          81
:-------------------------------------------------------

: start of data
:-------------------------------------------------------------------------------
 3.2267759e-11   4.1766113e-11   5.3713663e-11   6.8636402e-11   8.7144003e-11
...
```

The header starts with the keyword ‘GRIDFILE’ followed by the information about the DGrid version. Next lines contain the date and time of the calculation, the name of the quantum chemical package that produced the *basis* file and the name of the *basis* file.

Remark 1.4. In certain cases DGrid needs the access to the *basis* file. Thus, do not rename this file (otherwise change the name in the respective grid files as well). If DGrid is not able to find the *basis* file it will possibly follow another evaluation route (the corresponding information will be written into the *output* file).

The next line contains the name of the calculated property followed by the chosen spin and pair spin. This information is again necessary for DGrid to perform certain evaluations.

Remark 1.5. In *grid* files generated with old DGrid versions, this information is missing. Then some new features of the program will not be used! For instance, in previous version ‘ELI’ was used instead of ‘ELI-D’. The programs do not known the property ‘ELI’ in the 4.4 version. Thus, e.g., DGrid will not be able to find the exact attractor positions, or to evaluate the curvatures at attractor positions.

If the calculation was performed for selected orbitals, like in the above example, then the chosen orbitals as well as the occupations are written in lines following the property.

The title (from the DGrid *control* file) is followed by the atomic coordinates. Next line shows the energy of the system and the number of electrons. The remaining part of the header contains the information about the grid region, which are the lattice vectors (meaningful only for solid state calculations) and the description of the computed region with the coordinates of the grid origin together with the three vectors spanning the region and the number of points in each direction. In the case of scalar field this is also the last line of the header (see below for vector fields).

The values of the computed property are written in the format 14.7e. The grid points run in the x direction first (in contrast to the LMTO format where z runs first).

If the gradient vector of the property *prop* is computed with the assignment ‘**compute**=*prop* gradient-vec’ then all 3 vector components will be written to the *result* file. The descriptor ‘$vec = 3$’ indicates that the data are not single valued. Instead, 3 values per point are written in each line, as exemplary shown below:

```
:        origin           v_i         v_j         v_k
:-------------------------------------------------------
x:      -4.7000        9.400000    0.000000    0.000000
y:      -5.4000        0.000000   10.800000    0.000000
z:      -4.5000        0.000000    0.000000   10.200000

points:                   95         109         103
:-------------------------------------------------------

vec=3

: start of data
:-------------------------------------------------------------------------
:     vx              vy              vz
:-------------------------------------------------------------------------
 1.1930383e-06   1.1807959e-06   1.1716233e-06
 1.4118954e-06   1.4427352e-06   1.4308966e-06
...
```

1.8 Keywords

In this section all keywords (for exceptions see below) available for the DGrid *control* file are described in alphabetical order. The keywords are read case insensitive. The DGrid program is now inclusive the Basin part, i.e., the calculation and examination of basins is performed by DGrid. However, there is still a separate *input* file needed for such analysis. Therefore, the keywords associated with basin evaluation are described in Sec. 2.8.

The keywords are given either as variables, like e.g. '**compute**=', or as standalone expressions, like '**end**'. Some of them are followed by 1 or more numbers, e.g., '**mesh**=0.05 4.0'. If float number should be given then one must really input float numbers, otherwise the input routine will throw an error message. Thus, the command '**point**= 0.5 1.0 3' would not be valid, because '3' is an integer number.

Often more keywords can be written in single line. But it is better to put every keyword in separate line, because sometimes additional data are assumed per default. Some data even need to be written in rigorous sequence, e.g., the definition of a grid using the **vectors** keyword or the choice of occupied orbitals. Table 1.7 summarizes all the keywords available for the grid and single point evaluation, respectively.

1.8.1 Basis

The keyword **basis** is followed by the name of the *basis* file (described in Sec. 1.5). This can also be a UNIX path to this file:

Table 1.7 Keywords and parameters available for the grid calculation

Keyword	Description	Value
basis=	basis set	basis set filename
close-atom_distance=	distance to nucleus	float number (default 0.1)
component=	write vector component	x, y, z, xx, yy, zz, xy, xz, yz
compute=	choose property	properties given in Table 1.5
contributions=	orbital contributions	property (rho, ELI-D), x, y, z
cp=	search for critical point	(max, saddle, ring), x, y, z (float)
curv_decomp=	decomposition of ELI-D curvature	3 Cartesian coordinates (float)
end	end of input	
ev_projection=	gradient projection	property, x, y, z (float)
format=	result file (grid) format	cube, `dgrid`, grace, lmto
mesh=	define grid by mesh size	step and box border
occupation	choose orbital occupations	list of orbitals
output=	specify output file	output filename
overlap_matrix=	specify overlap matrix file	overlap matrix, basin number
point=	properties at given point	3 Cartesian coordinates (float)
reduced_step=	step for trajectory	float number (default 0.04)
ref_point=	reference point	3 cart. coordinates (float)
result=	specify result filename	generic name of the property file
space=	space representation	momentum, `position`
values=	number of values in a row	integer number (default 5)
vectors	define grid by 3 vectors	origin, 3 vectors, and intervals
wf_part=	select wave function part	`both`, imaginary, real

Default values are given in `Typewriter` font

basis=/home/test/molecule.adf

If the name of the *result* file is not explicitly given then it will be generated from the name of the *basis* file. Consequently, in this case the *result* files will be written into the same directory where the *basis* file is located. Computing for the above *basis* file the electron density yields the *result* file named */home/test/molecule.adf.rho_r*. If you wish to avoid this, use the '**result**=*newname*' assignment, yielding the file *newname.rho_r* that will be written into the local directory where the calculation is performed. The same applies to the *output* file (where the file */home/test/molecule.adf.dg* would be generated).

1.8.2 Close-atom_distance

The keyword **close-atom_distance** is followed by the distance <dist>. At the distance <dist> from the nucleus the search for a critical point stops. The default value is 0.01 bohr.

close-atom_distance=<dist>

The reason is to avoid discontinuities or very large gradients close to the nucleus.

1.8.3 Component

The keyword **component** is followed by the desired component <prop_i> of a property-vector array.

component=<prop_i>

If a gradient or Hessian of a property is computed, all values of the vector are written in 1 line in the *output* file, cf. page 16. With the above assignment a single component can be chosen. The allowed values for the <prop_i> descriptor are X, Y, Z for the gradient and XX, YY, ZZ, XY, XZ, YZ for the Hessian.

1.8.4 Compute

The keyword **compute** is followed by the desired property <prop>. All available properties are given in Tables 1.5 and 1.6 on pages 13 and 14, respectively.

compute=<prop> <type> <spin> <pair>

The descriptors <type>, <spin>, and <pair>, in detail given in the Sec. 1.6 on page 8, can be written in any order. Each property must have its own **compute** assignment on separate line. Bear in mind that for some properties calculation based on set of chosen orbitals (see keyword **occupation**) may not be reasonable (for instance for one-electron potential or ELF).

1.8.5 Contributions

The keyword **contributions** is followed by the property <prop> together with the descriptors <type>, <spin> and <pair> and the coordinates for the position.

contributions=<prop> <type> <spin> <pair> <x> <y> <z>

For the property only one of ‘rho’, ‘rho laplacian’, or ‘ELI-D’, respectively, is allowed by now. DGrid computes the property and its orbital contributions at the position given by the Cartesian coordinates [*x y z*] (3 float numbers). Using the *control* file *elid_contrib.inp* yields the *output* file *C3H3NO_elid_contrib.dg*. Below is the result of the analysis:

```
-------------------------------------------------------------------------------
contributions at point           [  1.050510,    1.500850,    0.000000]
-------------------------------------------------------------------------------

                    spin :        alpha
                    pair :        alpha-alpha
                                 ----------
                ELI-D[r] :       1.73735

--------------------
      MO          %
--------------------
    7    7      37.6
   14   14      18.6
    9    9      11.0
    8    8       9.3
   17   17       6.7
    6    6       5.9
   10   10       4.4
   15   15       4.3
   11   11       1.3
--------------------
               99.0
```

The orbital contributions are given at the position of the ELI-D bond attractor between the nitrogen and carbon (found using the keyword **cp**, see below). The analysis shows that the ELI-D value 1.73735, which is proportional to the population of α-spin electrons needed to form a fixed fraction of an $\alpha\alpha$-spin pair [23, 32], is a sum of mainly six contributions. The two highest contributions originate from the canonical orbitals $\phi_7\phi_7^*$ (37.6%) and $\phi_{14}\phi_{14}^*$ (18.6%). The orbital indexes correspond to the numbering in the *basis* file *C3H3NO_HF_6-31G.g03* set in the *control* file. In the above case the sum of the orbital contributions does not yield 100% because contributions smaller than 1% are not listed.

If the keyword **contributions** is used, any **compute** assignments will be ignored (actually, in this case the **compute** assignment can be omitted). Also, any grid definitions will be skipped, i.e., only the given position will be evaluated, and no *result* file will be written.

1.8.6 Cp

The keyword **cp** is followed by the descriptor <crit> (case insensitive) for the critical point:

cp=<crit> <x> <y> <z>

Table 1.8 Descriptors for critical point search

<crit>		description	signature
min	↦	minimum	(3, +3)
max	↦	attractor	(3, -3)
saddle	↦	saddle point	(3, -1)
ring	↦	ring critical point	(3, +1)

The search for the critical point starts from the position given by the Cartesian coordinates [*x y z*] (3 float numbers). The property for which the critical point will be searched must be given in a **compute** assignment. The position of the critical point (if found) is printed to the *output* file, together with the property value at this point, the gradient (which will be close to zero) and the Laplacian. Additionally, the eigenvectors and curvatures are determined from the Hessian. If you see the information 'Using Taylor' then the Hessian needed for the search was not computed analytically.

The search path is saved in the file *search.path.str*. This file is written in the STR format (for more details see Sec. C.6 in the Appendix C). In the *control* file *elid_cp.inp* (from the *examples* directory) for the search of an ELI-D attractor the starting point [1.5, 2.0, 0.0] was chosen. The resulting attractor (C-C bond descriptor) position is [1.05051, 1.50085, 0.00000], see the *output* file *C3H3NO_elid_cp.dg*. This position was used in the above example for the ELI-D contributions.

1.8.7 Curv_decomp

The keyword **curv_decomp** is followed by the property <prop> together with the descriptors <type>, <spin> and <pair> and the coordinates for the position.

curv_decomp=<prop> <type> <spin> <pair> <x> <y> <z>

Only ELI-D is allowed at the time for <prop>. At the position given by the Cartesian coordinates [*x y z*] (3 float numbers) the value, gradient, Laplacian and Hessian, together with the curvature eigenvalues and eigenvectors are computed and printed to the console. Along the principal ELI-D curvatures the ELI-D Laplacian is decomposed into the electron density and pair-volume function dependent terms [33]. The decomposed values are printed:

```
------------------------------------------------------------------------------
ELI-D decomposition at point          [  1.050510,    1.500850,    0.000000]
------------------------------------------------------------------------------

                     spin :        alpha
                     pair :        alpha-alpha
                                  ----------
                  ELI-D[r] :       1.73735

                  gradient :     9.086e-06
                components :  [ -4.216e-06, -8.049e-06,    0.00000 ]

                 Laplacian :      -7.69085

                   Hessian :  |   -4.47877,    1.27618,    0.00000 |
                              |    1.27618,   -2.79458,    0.00000 |
                              |    0.00000,    0.00000,   -0.41750 |

                                       Eigenvectors                curvature
                                  -------------------------------    -----------
                     1. ev :  [  0.88056, -0.47394,  0.00000 ]      -5.16565
                     2. ev :  [  0.47394,  0.88056,  0.00000 ]      -2.10770
                     3. ev :  [  0.00000,  0.00000,  1.00000 ]      -0.41750
                                    x         y         z

------------------------------------------------------------------------------

Data projected onto the ELI-D eigenvectors
------------------------------------------------------------------------------

             Y''/Y           rho''/rho             V''/V      2rho'/rho*V'/V
------------------------------------------------------------------------------
 11 |      -2.97329            1.79670           -4.52209           -0.24790
 22 |      -1.21317           -2.26056            1.05507           -0.00768
 33 |      -0.24031           -1.95647            1.71617            0.00000
------------------------------------------------------------------------------
sum |      -4.42677           -2.42033           -1.75085           -0.25558
------------------------------------------------------------------------------
```

Here the 11, 22, and 33 rows contain data for curvatures along the corresponding principal curvatures, e.g., 11 for the curvature along the first eingenvector (‘1. ev’ in the list above). ELI-D is given by the symbol Y (meaning Y_D) and the pair-volume function by the symbol V.

1.8.8 End

The keyword **end** is the last keyword in the *control* file that will be parsed. Any information after this keyword will be ignored.

1.8.9 Ev_projection

The keyword **ev_projection** is followed by the property <prop> together with the descriptors <type>, <spin> and <pair> and the coordinates for the position.

ev_projection=<prop> <type> <spin> <pair> <x> <y> <z>

For the property <prop> the value, gradient, Laplacian and Hessian, together with the curvature eigenvalues and eigenvectors are computed at the position given by the Cartesian coordinates [*x y z*] (3 float numbers) and printed to the console. The gradients of properties given by **compute** assignments are projected onto the principal <prop> curvatures. Using the example *evproj.inp*:

```
-------------------------------------------------------------------------------
eigenvectors at point           [  0.773035,    1.624593,    0.000000]
-------------------------------------------------------------------------------

                     spin :        alpha
                     pair :         --
                                 ----------
                   rho[r] :       0.18699

                 gradient :     1.631e-07
               components :  [  8.461e-08,  1.394e-07,    0.00000 ]

                Laplacian :      -0.61740

                  Hessian :  |    0.03440,   -0.23782,    0.00000 |
                             |   -0.23782,   -0.30100,    0.00000 |
                             |    0.00000,    0.00000,   -0.35080 |

                                      Eigenvectors                curvature
                                 -------------------------------    -----------
                    1. ev :  [  0.88777, -0.46028,  0.00000 ]       0.15770
                    2. ev :  [  0.46028,  0.88777,  0.00000 ]      -0.42430
                    3. ev :  [  0.00000,  0.00000,  1.00000 ]      -0.35080
                                   x         y         z

-------------------------------------------------------------------------------
```

shows the value, the Laplacian, and the principal curvatures of α-spin electron density of C_3H_3NO at the position [0.773035, 1.624593, 0.000000] which is the bond critical point between the nitrogen and carbon. This is followed by the projection of the ELI-D gradient in the direction of the mentioned eigenvectors:

```
gradient projections onto the eigenvectors:
-------------------------------------------------------------------------------

                     spin :        alpha
                     pair :        alpha-alpha
                                 ----------
                 ELI-D[r] :       1.43897

                 gradient :       2.07429
               components :  [    1.76076,   -1.09655,    0.00000 ]

    component projections :  [    2.06787,   -0.16304,    0.00000 ]

                Laplacian :      -4.93182

                  Hessian :  |   -3.96054,    3.13798,    0.00000 |
                             |    3.13798,   -2.17949,    0.00000 |
                             |    0.00000,    0.00000,    1.20821 |
```

```
          rotated Hessian :  |   -6.14773,    1.08057,    0.00000 |
                             |    1.08057,    0.00770,    0.00000 |
                             |    0.00000,    0.00000,    1.20821 |
 ------------------------------------------------------------------------------
```

First the ELI-D value and the gradient components along the Cartesian axes are given, followed by the gradient components projected onto the principal density curvatures (of course, the gradient vector remains unchanged). It can be seen that the ELI-D gradient vector is oriented mainly along one of the principal density curvatures. Similar projection is performed for the ELI-D Hessian components.

1.8.10 Format

The keyword **format** is followed by the descriptor <fmt> (case insensitive) for the format of the *result* file:

format=<fmt>

Table 1.9 Format descriptors

<fmt>		Format
cube	↦	CUBE [2]
dgrid	↦	DGrid *result* file
grace	↦	Grace [3]
lmto	↦	TB-LMTO-ASA [16]

Remark 1.6. In both the Cube and Grace formats the property values are given by float numbers. Routines which expect grids of integer values (like the basin evaluation) could fail.

1.8.11 Mesh

The computed grid region can be conveniently defined with the keyword **mesh**. It is followed by the mesh-size, i.e., the distance between neighboring grid points. Optionally, a border around the molecule can be given. To orient the grid along the Cartesian axes append the keyword `parallel`.

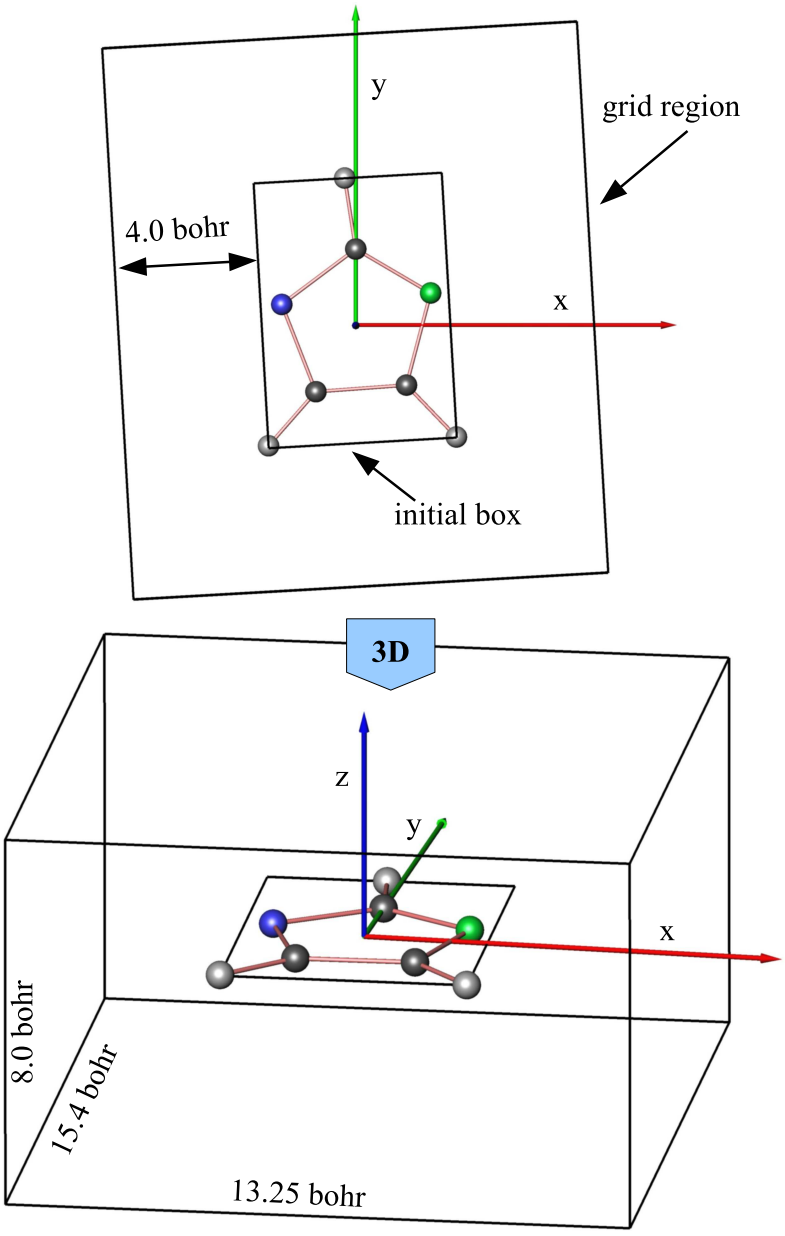

Fig. 1.1 Grid region for oxazole C_3H_3NO (C black, H gray, N green, O blue) created by the assignment '**mesh**=0.05 4.0'

mesh=<size> <border> `parallel`

The mesh-size must be given by float number. The units for both the mesh-size and the border are identical with the units for the atomic positions given in the *basis* file (usually bohr).

The determination of the grid region can be shown with the example *rho.inp* from the *example* directory (cf. example on page 7) where the assignment '**mesh**=0.05 4.0' is used. In the example a grid with the distance between neighboring points of 0.05 bohr and a border of 4.0 bohr is created around the C_3H_3NO molecule. The routine works as follows (cf. Fig. 1.1). From the atomic positions the average coordinate is computed. The 'initial box' (the smallest box around the molecule enclosing all atoms) is created. The box sides are parallel to the principal axes of the molecular rotational ellipsoid. As shown in the example, where the box is slightly rotated in the xy plane, the box need not be parallel to the Cartesian axes. To force a layout with grid box parallel with the Cartesian axes the keyword `parallel` must be appended. The 'initial box' is enlarged in all three directions by the border of 4.0 bohr (the grid region dimensions being multiple of the mesh-size of 0.05 bohr). This results in a grid of $266 \times 309 \times 161$ points (with the dimensions of 13.25 bohr $\times$ 15.4 bohr $\times$ 8.0 bohr, cf. the lower diagram in Fig. 1.1).

Remark 1.7. For the calculation of overlap integrals form Gauss-type orbitals, used for the fluctuation and delocalization indexes (see Sec. 2.7), it is necessary that the grid is oriented parallel to the Cartesian axes.

1.8.12 Occupation

The keyword **occupation** marks a section in the *control* file, where orbitals can be chosen for the evaluation. It must be the only keyword in the line:

occupation

The next readable information after this line describes the orbital choice using the following syntax:

<label> <nr> <occa> <occb>

Where <label> is the symmetry descriptor from the *basis* file (cf. page 5, for instance 'NONE' for the Gaussian calculations, see also the example *control* file *rho_orb.inp*). The symmetry label is followed by the orbital number, which must

comply with the numbering in the *basis* file (for ADF calculations for each symmetry the orbital numbering starts again with 1). Next two (float) numbers are the orbital occupations for each spin channel. Other possibility is:

```
<label>  all
```

Here all orbitals with given symmetry label will be chosen (for instance all sigma orbitals). The occupation input is closed by the following line:

```
occupation_end
```

1.8.13 Output

The keyword **output** is followed by the generic name of the *output* file. This can also be a UNIX path to this file.

```
output=<XY_test>
```

The *output* file name will be generated by appending the string ‘’.dg’ to the generic name (i.e., ‘XY_test.dg’ for the above example). If the name of the *output* file is not explicitly given by the **output** assignment then the output will be written to the console. Using the assignment:

```
output=.
```

generates the *output* file name from the name of the *basis* file by appending the string ‘.dg’ to it. If the file is already present, then the string ‘_1’, ‘_2’, etc. will be appended to avoid overwriting of data.

1.8.14 Overlap_matrix

The keyword **overlap_matrix** is followed by the name of the file overlap-matrix. This can also be a UNIX path to this file.

overlap_matrix=<name.sij> <basnr>

The overlap matrix for all molecular orbitals over the chosen basin is set up in DGrid. The data are needed for instance to compute the domain-averaged Fermi-hole.

1.8.15 Point

The keyword **point** is followed by the position given by the Cartesian coordinates [*x y z*] (float numbers) at which the properties will be evaluated:

point= <x> <y> <z>

The properties must be given by separate **compute** assignments. For each property the value at the position [*x y z*] is printed to the *output* file (respectively console), together with the property gradient and the property Laplacian. Additionally, the eigenvectors and curvatures are determined from the Hessian. If you see the information 'using Taylor' then the data were not computed analytically.

If the keyword **point** is used any grid definitions will be ignored, i.e., only the given position will be evaluated, and no *result* file will be written.

An additional feature is the information about indexes connected with local virial [5]. The feature is invoked with the keyword *virial* following the [*x y z*] coordinates:

point= <x> <y> <z> virial

For all properties given by the **compute** assignments the corresponding data at the [*x y z*] position, like in the previous case. However, at the end of the output the the following line will be included (data for C_3H_3NO at the position [0.77304, 1.62459, 0.00000], the N-C bond critical point in electron density):

```
-------------------------------------------------------------------
Virial indexes:    L         V         G       |V|/G       H
-------------------------------------------------------------------

                -1.2348   -0.8647    0.2780    3.1104   -0.5867
-------------------------------------------------------------------
```

where L is the total electron density Laplacian, $V = \frac{1}{4}\nabla^2\rho - 2\tau$ is the local potential energy density, G is the positive definite kinetic energy density and $H = G + V$ is the energy density. $|V|/G$ is the corresponding local virial ratio at the chosen position [*x y z*].

1.8.16 Reduced_step

With the keyword **reduced_step** the search for critical points can be done with reduced maximal step (to avoid jumps between regions with different Hessian form):

reduced_step=<step>

The default value for <step> is 0.02 bohr.

1.8.17 Ref_point

The keyword **ref_point** is followed by the position given by the Cartesian coordinates [*x y z*] (float numbers) for the reference electron:

ref_point= <x> <y> <z>

The assignment is needed only for 2-particle properties, see Table 1.6 on page 14. In this case the coordinate of one electron is fixed at the position [*x y z*], whereas the position of the other electron runs through the grid (as defined by the **vectors** or **mesh** assignments). Additionally, it is used for the reference position when calculating the local source (with '**compute**=LS').

1.8.18 Result

The keyword **result** is followed by the generic name of the *result* file. This can also be a UNIX path to this file.

result=<XY_test>

The *result* file names for each computed property are generated from this generic name appending the corresponding strings (see Tables 1.5 and 1.6 on page 13 and 14, respectively). If the name of the *result* file is not explicitly given by the **result** assignment then DGrid generates the *result* file name from the name of the *basis* file by appending the strings corresponding to the respective properties.

1.8.19 Space

The keyword **space** is followed by the descriptor <spc> determining in which space representation the properties will be computed:

space=<spc>

Table 1.10 Space representation descriptors

<spc>		Representation
position	$\mapsto$	coordinate space [$x\ y\ z$]
momentum	$\mapsto$	momentum space [$p_x\ p_y\ p_z$]

The default is the evaluation in position space representation. In case of momentum space representation the grid definition is using the momentum coordinates.

Remark 1.8. DGrid computes the properties in momentum space using the same formulas as in the coordinate space. The only difference is that it utilizes Fourier-transformed orbitals (momentals). Thus, the charge given by the modulus of squared orbitals has in momentum space similar meaning as in the real space. However, not the kinetic energy density, because this property is computed by DGrid from squared orbital gradients (of course, in momentum space the kinetic energy is proportional to the expectation value of p^2).

1.8.20 Values

The keyword **values** is followed by the number <num> of values per line in the *result* file:

values=<num>

<num> must be given by an integer number.

1.8.21 Vectors

The keyword **vectors** marks a section in the *control* file, where the grid region is defined. It must be the only readable information in the line:

vectors

The next readable information after this line defines the origin of the grid by the Cartesian coordinates (float numbers):

```
<x>  <y>  <z>
```

The strings ‘GRID_DEFINITION:’, ‘origin:’, ‘i-vector:’, etc. in the *control* file are not parsed due to the colon at the end of the strings – they used just for clarity:

```
 GRID_DEFINITION:   vectors
:            X     Y     Z
:---------------------------
 origin:     0.0  -4.0   0.0
:                            INTERVALS
:-------------------------------------
 i-vector:   3.0   4.0   0.0     100
 j-vector:  -4.0   3.0   0.0     100
 k-vector:   0.0   0.0   5.0     100
```

The next 3 lines after the origin define the vectors spanning the grid mesh. Each line contains 3 float numbers for the vector length along the *x*, *y*, *z* axes, and an integer number for the number of intervals the vector will be divided in. Thus, the above assignment will generate a grid mesh starting at the position [0, −4, 0]. The 3 vectors (*i*,*j*,*k*) are 5 bohr long. The direction of vector *k* coincides with *z* axis. The vectors *i, j* are perpendicular to each other. The endpoint of vector *i* is at [3, 0, 0] (this information is not explicitly given in the input, but can easily be deducted from the data – from the origin position [0, −4 ,0] one moves 3 bohr in *x* direction and 4 bohr in *y* direction). The grid mesh has points spaced by 0.05 bohr along each vector (i.e., $101 \times 101 \times 101$ points in total). To avoid numerical instabilities the mesh point distance should not exceed 0.1 bohr.

1.8.22 Wf_part

The keyword **wf_part** is followed by the descriptor <img> determining which part of the wave-function is used for the calculation of the properties:

wf_part=<img>

The default value is `both`.

Table 1.11 Wave-function part descriptors

<img>		Representation
real	↦	use real part only
imaginary	↦	use imaginary part only
both	↦	use both parts

Chapter 2
Basin evaluation

2.1 Introduction

On an equidistant grid of property values regions surrounded by the surfaces of zero-flux of the property gradient (i.e., basins [5]) are determined. Alternatively, localization domains [30], i.e., regions surrounded by an isosurface, resp. separate regions surrounded by two isosurfaces can be determined.

DGrid computes the volumes of the basins, resp. the localization domains. If there is a second grid for another property (with the same grid parameters as the first one) the program integrates this property over each basin. An output file can be created, where each grid point is assigned to its basin (a *basin* file).

If the *basis* file is present, the attractors, minima, saddle points and ring points are determined, together with the corresponding curvatures and Hessian eigenvectors. Additionally, the interconnection graphs can be produced (written in the STR format, see the Appendix C). The interconnection graphs are given by all the attractors and saddle points, respectively ring points and minima, connected by gradient field lines. In case of the electron density such diagram for the attractors and saddle points is called the molecular graph.

In case of Gauss-type basis the MO overlap integrals over the basins can be computed. Those overlap integrals are the prerequisite for the calculation of the fluctuation and delocalization indexes.

The evaluation is performed by DGrid using a *control* file with keywords specific for the chosen analysis. The control file must comply with the rules given in the introduction to the DGrid program (cf. page 3).

2.2 Control file

For the *control* file any name can be chosen. The *control* file supplies all the information concerning the evaluation. The data must comply with the rules described in the

introduction to chapter 1, page 3. A simple basin calculation can be performed with very few commands. Following is the *control* file *rho_basins.inp* for the calculation of electron density basins for the C_3H_6 molecule (see the *examples* directory):

```
:TITLE
:-------------------------------------------------------------|
::C3H3NO cropped density basins
:-------------------------------------------------------------|

:KEYWORDS
:-------------------------------------------------------------
 property =C3H3NO_HF_6-31G.g03.rho_r
 crop     =C3H3NO_HF_6-31G.g03.rho_r       0.001
 integrate=C3H3NO_HF_6-31G.g03.rho_r

 output=.

end
```

The first readable information in the *control* file must be the title (whereby all empty lines as well as lines beginning with a single colon will be skipped). The title line starts with two colons, i.e., the line will not be parsed. The two colons are obligatory, whereas the title text can be omitted (i.e., empty title). If some title text is found in this line, it will be written into the header of the *basin* file , see Sec. 2.3 below.

The keyword **property** is obligatory for the basin calculation. It is followed by the name of the *property* file (i.e., the file with the scalar grid values for the desired property). This can also be a UNIX path to this file. The *property* grid file *C3H3NO_HF_6-31G.g03.rho_r* must be computed in advance by a separate DGrid job (cf. the *control* file *rho.inp* from the *example* directory). The program determines the basins, i.e., regions enclosing the manifold of all property gradient trajectories terminating at given attractor.

Remark 2.1. Alternatively, a *basin* file (see below) with basin data from previous run can be read in instead of the *property* file. In this case the basins will not be recalculated (and written on disc again), thus saving time and disc space.

For a molecule the outer basins extend into infinity. Of course, in an actual calculation those basins extend just till the border of the chosen grid region. Obviously, it is hard to compare such basin volumes in some reasonable way. It seems to be convenient first to crop the region around the molecule by certain isodensity surface, a kind of envelope. The electron density isovalues used are 10^{-3}–10^{-5} bohr^{-3}. The cropping is invoked by the keyword **crop** followed by the name of the grid file utilized for the isosurface generation and the cropping value. With this, all grid points outside the cropped region will not be searched for basins. Thus, it avoids the time consuming basin search in regions of low gradients as well. The result for the above *control* file is shown in Fig. 2.1.

If the keyword **integrate** is given, then the property from the file named after the keyword will be integrated over each basin. In case of electron density use a grid with the grid point distance less than 0.1 bohr (better 0.05 bohr) to get reasonable charges (and basins). Alternatively, the file from a DGrid refinement can be used.

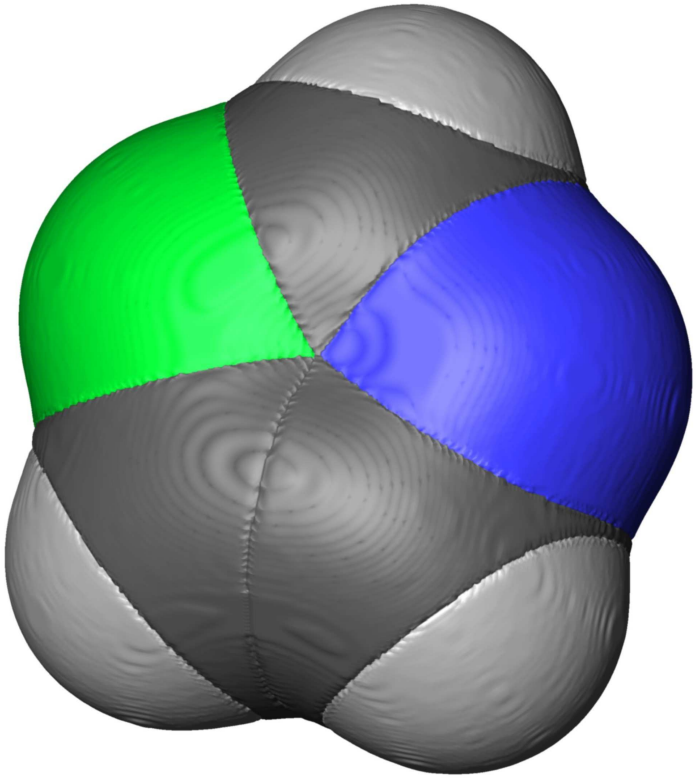

Fig. 2.1 Electron density basins for C_3H_3NO, cropped by 0.001 bohr^{-3} electron density isosurface. Blue: oxygen basin; green: nitrogen basin; black: carbon basins; gray: hydrogen basins

Remark 2.2. For electron density based on Gauss-type orbitals the overlap integrals over basins can be calculated (keyword **overlap**, cf. Sec. 2.7.1). In the subsequent evaluation of the fluctuation also the basin populations are computed analytically. The remaining error in the basin population is due to the form of the basin borders derived from a discrete grid (i.e., small cubes).

The keyword **output** is followed either by the name of the *output* file or by the character '.' (as in the above example). In this case the *output* file name will be generated from the *property* file name. In both cases the string '.bas' will be appended to the generic file name. Thus, for the above example the output will be written to the file *C3H3NO_HF_6-31G.g03.rho_.bas*.

After the basins are determined a grid file will be created with each grid point assigned to its basin by an integer number. The name of the resulting *basin* file will be generated from the *property* file name. Alternatively, the name for the *basin* file can be chosen using the keyword **result** followed the generic file name. In both cases the string '.bsn' will be appended to the generic file name. The *basin* file is

written in the DGrid format (**x** coordinate runs first). The format can be changed with the keyword **format**.

All available keywords are summarized in Sec. 2.8.

2.2.1 Attractors

The attractors, the prerequisite for a basin, are searched on the discrete grid. This often yields much more discrete local maxima than actually given by the attractors of the vector field. There are two possibilities for the DGrid program to decide how to proceed, depending on the presence of the *basis* file.

Remark 2.3. The location of the *basis* file is given in the *property* file – bear in mind that the *basis* file must be present at the given location!

2.2.1.1 Case: unknown *basis* file

This is usually the case for solid state calculations or if the *basis* file was not found by DGrid (in which case a warning will be printed into the output. Using the keyword **attractors** prints the actual list of discrete local maxima on the grid (the list length depends on the additional parameters, see keyword **attractors** in Sec. 2.8). It supplies the following information:

```
                    ******************************************
                    *    D I S C R E T E    M A X I M A      *
                    ******************************************

-------------------------------------------------------------------------------------------------------
Loc.max  basin    i   j   k       value     min.dist   value diff.       x       y       z      atom    dist
-------------------------------------------------------------------------------------------------------
    1      1     92 160  80     232.7108     2.57        129.5149     -2.054   0.591   0.000     O1     0.02
    2      2    175 161  80     155.6372     2.41         52.4413      2.086   0.876   0.000     N3     0.02
    3      3    108 110  80     117.4279     1.98        117.0010     -1.114  -1.860   0.000     C5     0.00
    4      4    135 188  80     103.1959     1.96        102.7652      0.013   2.111   0.000     C2     0.02
    5      5    159 111  80     100.9773     1.95        100.5468      1.429  -1.665   0.000     C4     0.02
    6      6    131 227  80       0.4307     1.96       -102.7652     -0.297   4.046   0.000     H6     0.06
    7      7    184  81  80       0.4305     1.95       -100.5468      2.763  -3.092   0.000     H7     0.05
    8      8     80  82  80       0.4269     1.98       -117.0010     -2.432  -3.337   0.000     H8     0.03
-------------------------------------------------------------------------------------------------------

Local maxima written to the file locmax.str
```

The above result is for the *control* file *rho_basins.inp* including the keyword **attractors** and the *basis* file removed form the directory (thus unknown for DGrid). *Loc.max* is the running number of the local maximum. The value *basin* indicates to which basin the local maximum belongs.

Remark 2.4. After the trajectory search the basins will be renumbered in ascending order of the volume size.

The indexes *i j k* are the grid coordinates of the local maximum, whereas *x y z* are the corresponding Cartesian coordinates. *value* shows the function value (here the electron density) at the grid position. The local maxima are given in descending

order with respect to the function values. *min.dist* is the distance to the closest local maximum (given in the same units as the grid parameters). *value diff.* is the value difference between the local maximum and the closest local maximum. *atom* shows the symbol of the closest atom (at the distance *dist*).

Using the keyword **attractors** creates the file *locmax.str* with the coordinates of the local maxima written in the STR format (for visualization, cf. the Appendix C). In this case, no search for the basins is performed. For grids from solid state calculations, where the wave function information is not given, it is a good idea first to start with the keyword **attractors** and check the total number of basins found on the discrete grid. Then, with the keyword **attractors** still active, the number of basin can be reduced (without actually performing the time consuming trajectory search – for the basin search the keyword **attractors** needs to be commented out). The local maxima will be written to the output as well as into the file *locmax.str*. The number of attractors written out can be changed appending additional parameters to the keyword **attractors**, cf. the Sec. 2.8.

Remark 2.5. To start the basin search the keyword **attractors** must not be active!

2.2.1.2 Case: using *basis* file

In this case all local maxima on the discrete *property* grid will be tested by DGrid using the Hessian search and assigned to the corresponding attractor. Using the keyword **attractors** disables the basin search. Instead, the topology section appears in the output (here corresponding to the local maxima on previous page, now for a run with the *basis* file included):

```
PROPERTY:    rho[r]                  spin = both          pair = unknown

* * *  A T T R A C T O R S    I N    R E G I O N  * * *

-------------------------------------------------------------------------------------------------------
 Nr.  Basin       x          y          z         value     Laplacian               curvature
-------------------------------------------------------------------------------------------------------
  1      1    -2.0618     0.5748     0.0000     291.3237        -                      O1
  2      2     2.0869     0.8955    -0.0000     192.1030        -                      N3
  3      3    -1.1101    -1.8596     0.0000     118.3147        -                      C5
  4      4     0.0000     2.1211    -0.0000     118.3135        -                      C2
  5      5     1.4135    -1.6748    -0.0000     118.3130        -                      C4
  6      6    -0.3026     4.0665     0.0000       0.4320     -19.1959     -6.6123    -5.9828    -6.6008
  7      7     2.7803    -3.0966     0.0000       0.4315     -19.1470     -6.5836    -5.9880    -6.5753
  8      8    -2.4073    -3.3406     0.0000       0.4289     -18.9880     -6.5374    -5.9233    -6.5272
-------------------------------------------------------------------------------------------------------
```

The string ‘rho[r]’ together with the spin-pair information show that the attractors were searched in the total electron density field. For each attractor the Cartesian coordinates are printed, followed by the property value at the attractor position. In the above example the value of the property Laplacian is not always given, because some attractors correspond to positions closer then 0.01 bohr to a nucleus (cf. keyword **close-atom_distance**, Sec. 1.8.2). Instead of the three curvatures only the atomic symbols are printed.

Remark 2.6. To start the basin search the keyword **attractors** must not be active!

In both cases, with or without the presence of the *basis* file, the number of basins can be larger then requested (for instance large number of core ELI-D basins for heavier elements). Then, the total number of basins can be reduced in three ways (also simultaneously):

- using the keyword '**compactify** *dist vdiff*':

  local maxima closer then the distance *dist* and differing in the function values by less then *vdiff* will be reduced to single attractor. If omitted *vdiff* is set to the property value range.

- using the keyword '**top**=*iso*':

  local maxima inside *iso*-localization domains will be merged into single attractor.

- using the keyword **eli_core**:

  local maxima inside the respective atomic core defined by ELI-D (internal table) will be merged into single basin.

In DGrid the integration of a property is done numerically on an equidistant grid. Thus, the precision depends on the mesh size and can be subject to large errors. In case of electron density – the standard charge determination – especially in the atomic core regions. There are 3 possibilities to compensate for the errors:

- using the keyword **core_charge**:

  in a small sphere around each atom the electronic charge is replaced by a value from an internal table (data computed from the basis sets of Clementi and Roetti for the atoms N - Xe, as well as from relativistic ADF [1] calculations for the atoms Cs-Lu). Of course, the atomic coordinates need to be specified in this case (either by the **pic_file** directive or from the property file in the DGrid format.

- using the refinement procedure (see Sec. 3.5). The file name from the refinement procedure is given after the keyword **integrate** (see the help using `dgrid -u`).

- For Gauss-type orbitals see remark 2.2.

The grid from a solid state calculation often runs over the full unit cell. Then, using the assignment '**coordinates**=relative' gives all positions in the output relative to the grid parameters. Otherwise the positions are given in units used in the *property* file (i.e., absolute positions). If translation or mirror symmetry is given at the borders of the grid, then using the assignment:

symmetry=translation `i j k`

or

symmetry=mirror `i j k`

utilizes the respective symmetry in the `i`, `j`, and `k` direction (i.e., the first, second and third dimension). At least 1 symbol for the direction must be present. After the basin search all symmetry equivalent basins are added together and shown as single basin.

Remark 2.7. The last command in the *control* file is the keyword **end**. All data after this keyword will be ignored. Without the **end** keyword the data will be read till end of file.

2.3 Basin file

After the basin search each grid point is assigned to its basin by an integer number. The data are written on separate *basin* file. It has a header followed by the basin values. The format of the header is similar to the one for the *property* file. Following is the example file *C3H3NO_HF_6-31G.g03.rho_r.bsn* created by the *control* file *rho_basins.inp* mentioned in the introduction:

```
BASINFILE   created by DGrid version 4.5   19-08-2009

:job was executed on:         Wed Aug 19 15:06:44 2009

property_grid=C3H3NO_HF_6-31G.g03.rho_r

basis_from_program:    GAUSSIAN03    basis=C3H3NO_HF_6-31G.g03

basins_for: rho[r]                   spin=both    pair=unknown

Property basins cropped by 0.001000 C3H3NO_HF_6-31G.g03.rho_r isosurface

::C3H3NO   HF    6-31G

:atom       x          y          z
:-----------------------------------
 O      -2.0618     0.5748     0.0000
 C       0.0000     2.1211    -0.0000
 N       2.0869     0.8955    -0.0000
 C       1.4135    -1.6748    -0.0000
 C      -1.1101    -1.8596     0.0000
 H      -0.3077     4.1025    -0.0000
 H       2.8049    -3.1219    -0.0000
 H      -2.4310    -3.3680     0.0000
:-----------------------------------

Energy= -244.50273 Hartree     Electrons= 18 alpha  +  18 beta
```

```
:             lattice vectors
:-------------------------------------
:         a           b           c
:-------------------------------------
x:    1.000000    0.000000    0.000000
y:    0.000000    1.000000    0.000000
z:    0.000000    0.000000    1.000000
:-------------------------------------

:        origin           v_i         v_j         v_k
:-----------------------------------------------------
x:      -6.1931       13.228691   -0.873034    0.000000
y:      -7.6570        0.751150   15.375234    0.000000
z:      -4.0000        0.000000    0.000000    8.000000

points:                   266         309         161
:-----------------------------------------------------
```

The first readable information is a line beginning with the string ‘BASINFILE’ and the DGrid version. Next lines contain the date and time of the calculation, followed by the name of the *property* file used to determine the basins, the name of the quantum chemical package that produced the *basis* file followed by the name of the *basis* file. In the next line is the name of the calculated property and the chosen spin and pair-spin. This is followed by a line with the information about the cropping of the grid region. Bear in mind that DGrid reads this grid for cropping. Thus, it should be available at the given address.

The title (line starting with double colon) is followed by the atomic coordinates as well as well as the energy of the system and number of electrons. The next part contains the information about the grid region. The lattice vectors are meaningful only for solid state calculation. The lattice data are followed by the coordinates of the grid origin together with the three vectors describing the computed region and the number of points in each direction.

The last part of the header gives some information about the basins. It shows the volumes and the maximal grid value in the respective basins as well as a descriptor indicating to which atom the respective basin can be attributed.

```
:basin  volume   maximum     x       y       z       i    j    k descriptor
:-------------------------------------------------------------------------
    1   40.542    0.4307  -0.297   4.046   0.000   131  227   80         H6
    2   41.243    0.4269  -2.432  -3.337   0.000    80   82   80         H8
    3   42.715    0.4305   2.763  -3.092   0.000   184   81   80         H7
    4   62.999  103.1959   0.013   2.111   0.000   135  188   80         C2
    5   78.960  100.9773   1.429  -1.665   0.000   159  111   80         C4
    6   79.533  117.4279  -1.114  -1.860   0.000   108  110   80         C5
    7   99.673  232.7108  -2.054   0.591   0.000    92  160   80         O1
    8  116.216  155.6372   2.086   0.876   0.000   175  161   80         N3

:-------------------------------------------------------------------------
```

Remark 2.8. The basin descriptors were not given in the previous versions.

This section is followed by integer values according to the basin the given point corresponds to. The grid points run in the *x* direction first. The data line below shows

that there is a ‘basin’ number zero. This corresponds to the grid points outside the molecular envelope.

```
: start of data
:------------------------------------------------------------------------
   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

It is always a good idea to visualize the basins, cf. Fig. 2.1, to see whether there are some unusual artifacts. Special visualization techniques should be utilized (this creates surfaces around blocks of constant data values, cf. Appendix A.2).

Remark 2.9. If the only possibility to visualize the basins is to create isosurfaces of basin values then bear in mind that, for instance, the isosurface with (basin) value 3 is not only found around the basin 3, but also between the basins 1 and 4, 2 and 4, etc.

2.4 Output file

The output of the basin evaluation includes the information about the files read in, the calculation progress and some statistics. The progress during the basin search is shown by a scale, giving the number of points which are tested. This check is repeated till the zero-flux surfaces do not move anymore. After the basins are determined the integration section starts. If the **integrate** keyword was not given, then just the volumes of each basin will be determined.

If there is a property file assigned by the **integrate** command, like the electron density grid *C3H3NO_HF_6-31G.g03.rho_r* in the *control* file *rho_basins.inp*, the corresponding property will be integrated over each basin. In the *output* file first the structure data are given, followed by the information about replaced charge if the **core_charge** keyword was used. For each basin its volume, the integral of the property over this volume (usually the charge), the property value at the attractor position (maximum of the scalar property that was used to define the basins) as well as the attractor position itself are given:

```
                rho[r]     rho[r]
BASIN  VOLUME  INTEGRAL    MAXIMUM     <X>    <Y>    <Z>    ATOMS DIST ECCNT
------------------------------------------------------------------------------
   1   40.542    0.8673     0.4320   -0.303  4.067  0.000   H6    0.04   -
   2   41.243    0.8795     0.4289   -2.407 -3.341  0.000   H8    0.04   -
   3   42.715    0.9063     0.4315    2.780 -3.097  0.000   H7    0.04   -
   4   62.999    4.9641   118.3135    0.000  2.121 -0.000   C2    0.00   -
   5   78.960    5.6659   118.3130    1.413 -1.675 -0.000   C4    0.00   -
   6   79.533    5.6153   118.3147   -1.110 -1.860  0.000   C5    0.00   -
   7   99.673    9.0038   291.3237   -2.062  0.575  0.000   O1    0.00   -
   8  116.216    7.9439   192.1030    2.087  0.895 -0.000   N3    0.00   -
------------------------------------------------------------------------------
TOT   561.881   35.8460

Volume difference:  -1070.519

Basin data written on file C3H3NO_HF_6-31G.g03.rho_r.bsn
```

The above output shows that the QTAIM basins for the hydrogens are slightly positive (around +0.1 each), whereas the carbon atoms are markedly positive (especially C2, cf. basin 4). The oxygen and nitrogen basins are both single negative charged).

If the *basis* file is not present then the attractor positions and property values correspond to the closest grid point. For each attractor the closest atom (or atoms, see below) as well as the distance to this atom are given (in the above example the distance is always zero for atoms other then hydrogen, because in this case the attractors of electron density coincide with the atomic positions).

Remark 2.10. When the basins are cropped by an isosurface (keyword **crop**) then the sum of the basin volumes is less then the total grid region volume. In that case the line 'Volume difference:' will be appended.

Following is the output example for the ELI-D basins of the C_3H_3NO molecule (cropped by the 0.001 bohr^{-3} density isosurface):

```
                rho[r] ELI-D[r]
BASIN VOLUME INTEGRAL  MAXIMUM    <X>    <Y>    <Z>    ATOMS  DIST      ECCNT
-------------------------------------------------------------------------------
   1   0.268   2.1288  13.1296  -2.062  0.575  0.000   O1     0.00        -
   2   0.453   2.1099  14.9864   2.087  0.895 -0.000   N3     0.00        -
   3   0.813   2.0952  17.1006   1.413 -1.675 -0.000   C4     0.00        -
   4   0.813   2.0920  17.2159   0.000  2.121 -0.000   C2     0.00        -
   5   0.818   2.0942  17.2365  -1.110 -1.860  0.000   C5     0.00        -
   6   5.462   1.1927   1.5295  -1.633 -0.561  0.000   O1-C5  1.21-1.40 0.014
   7   5.802   1.2585   1.5247  -1.101  1.296  0.000   O1-C2  1.20-1.38 0.001
   8  10.919   1.6965   1.6748   1.797 -0.386  0.000   N3-C4  1.31-1.34 0.044
   9  40.696   2.8040   1.6768   1.051  1.501  0.000   N3-C2  1.20-1.22 0.003
  10  71.845   2.2711   7.6437  -2.501 -3.442  0.000   H8     0.10        -
  11  73.889   2.2487   7.5734   2.883 -3.195  0.000   H7     0.11        -
  12  78.033   2.3838   7.9725  -0.329  4.216  0.000   H6     0.12        -
  13  85.723   4.7544   1.6749  -3.091  0.870  0.000   O1     1.07        -
  14  91.854   3.0218   1.9272   3.314  1.377  0.000   N3     1.32        -
  15  94.491   3.6944   1.7583   0.145 -1.876  0.000   C5-C4  1.26-1.28 0.108
-------------------------------------------------------------------------------
TOT  561.881  35.8460

Volume difference:  -1070.519

Basin data written on file C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn
```

Now there are much more basins then in the previous example using the electron density field (where each ρ-basin can be attributed to one of the 8 atoms). The first 5 basins show the atomic cores of oxygen, nitrogen, and carbon, with ELI-D attractors at the corresponding atoms, i.e., *DIST=0.00*, cf. also the colored basins in Fig. 2.2. The cores are populated roughly by 2.1 electrons.

The basins 6–9 correspond to the O-C and N-C bonds, respectively, with ELI-D attractors each time located between two atomic nuclei. *DIST* shows the distance to the particular neighbors. The attractor number 6 is located 1.21 bohr from oxygen and 1.40 bohr from carbon. It is not located exactly at the interconnection line between the atoms, because the eccentricity (perpendicular distance to the O1-C5 line) *ECCNT=0.014* bohr (the attractor number 7 is already located practically at the interconnection line).

The basins number 10-12 are attributed to the C-H bonds, because of the absence of hydrogen cores in the ELI-D representation (thus, from a simple ELI-D analysis

one cannot distinguish between the part that could be attributed separately to the hydrogen atom).

Similarly to the basins number 10-12 the ELI-D attractors number 13 and 14 are located around single nucleus. A closer inspection of the data reveals that the corresponding basins describe lone-pairs, cf. the ELI-D diagram in Fig. 2.2. Interestingly, there is just single oxygen lone-pair basin populated by 4.75 electrons (not split into two separate parts, like in water molecule).

The distance of the property maximum to closest atoms together with the eccentricity is used to set up the basin descriptors (respectively the 'ATOMS' column in the output). It works in the following way. If the basin property maximum (i.e., the basin attractor) lies within the tabulated ELI-D core radius, then the corresponding atomic symbol is assigned to the respective basin. If the basin attractor is positioned

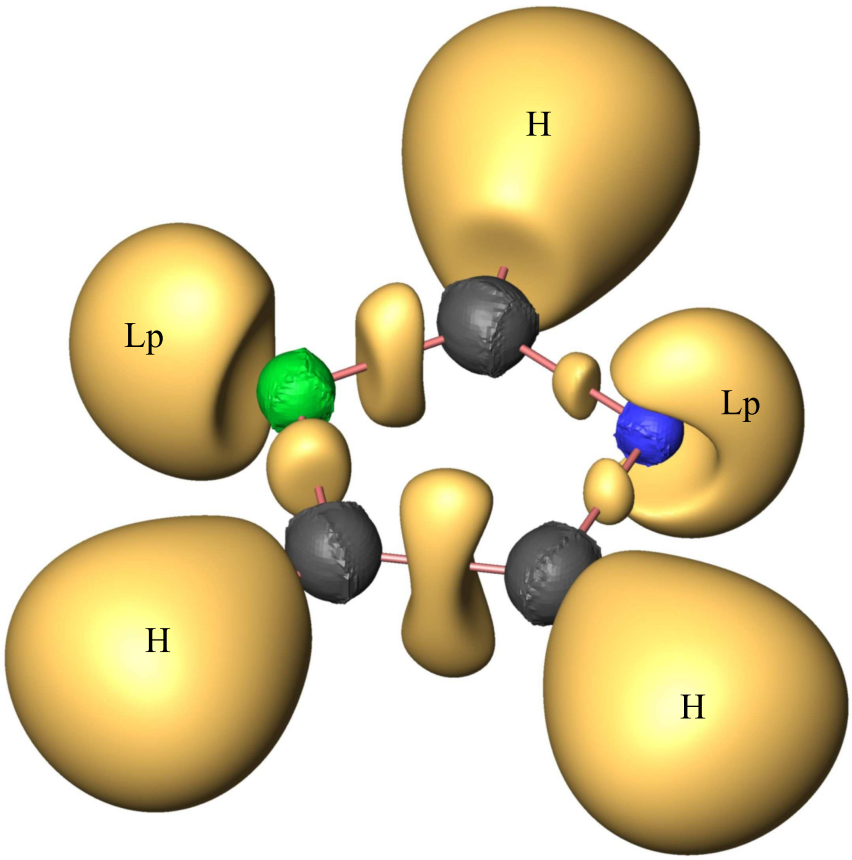

Fig. 2.2 ELI-D for C_3H_3NO. Gold colored (irreducible) 1.45-localization domains ('H' and 'Lp' mark the hydrogen and lone-pair domains, respectively. Blue: oxygen core basin; green: nitrogen core basin; black: carbon core basins

between two atoms and the angle between the atom-atom interconnection line and the attractor-atom line is less than 40° then the basin descriptors is formed by the corresponding atomic symbols with a period in between. Otherwise, in the *basin* file, the symbol 'LP' (for lone-pair) is used.

Remark 2.11. The basin descriptors should serve as a hint or help. It is not intended to be used as a rigid classification.

2.5 Search for critical points

The integration section in the output is closed with the information about the file to which the basin data are written. This *basin* file can be used for further evaluations. An important application is the search for the critical points followed by the evaluation of some quantities at those points found. In the *control* file below (modified *rho_basins.inp* file) the density *basin* file is used with the keyword **property** and the command **topology**:

```
:TITLE
:-------------------------------------------------------------|
::C3H3NO density basins
:-------------------------------------------------------------|

:KEYWORDS
:-----------------------------------------------------------
 property =C3H3NO_HF_6-31G.g03.rho_r.bsn

 output=.

 topology
 icl_graph=full

end
```

Now, the basins are not recomputed by DGrid (what would be otherwise the case for the density grid file, i.e., it is possible to run the basin search immediately followed by the critical point search in single job). Instead, they are directly read from the *basin* file *C3H6_HF_3NO-31G.g03.rho_r.bsn* created in previous run. The headers in the *output* file will be the same as already discussed (with a 'Volume' section included). Additionally, the 'Topology' section will appear in the *output* file. The first part of this section shows all electron density attractors:

```
* * * A T T R A C T O R S   I N   R E G I O N * * *

-------------------------------------------------------------------------------------------------------
 Nr.  Basin       x          y          z          value      Laplacian                curvature
-------------------------------------------------------------------------------------------------------
  1      7    -2.0618     0.5748     0.0000      291.3237         -                       O1
  2      8     2.0869     0.8955    -0.0000      192.1030         -                       N3
  3      6    -1.1101    -1.8596     0.0000      118.3147         -                       C5
  4      4     0.0000     2.1211    -0.0000      118.3135         -                       C2
  5      5     1.4135    -1.6748    -0.0000      118.3130         -                       C4
  6      1    -0.3026     4.0665     0.0000        0.4320      -19.1959      -6.6123    -5.9828    -6.6008
```

```
  7     3     2.7803    -3.0966     0.0000        0.4315      -19.1470      -6.5836    -5.9880    -6.5753
  8     2    -2.4073    -3.3406     0.0000        0.4289      -18.9880      -6.5374    -5.9233    -6.5272
-------------------------------------------------------------------------------------------------------------
```

This part is followed by the information about the saddle points (points with 2 negative and 1 positive principal curvatures):

```
* * * (3,-1)  S A D D L E   P O I N T S   I N   R E G I O N * * *

-------------------------------------------------------------------------------------------------------------
 Nr. Basins      x       y       z     value  Laplacian           curvature         ellip  |V|/G      H
-------------------------------------------------------------------------------------------------------------
  1    8- 4   0.7730  1.6246  0.0000   0.3740   -1.2348    0.3154  -0.8486  -0.7016  0.21  3.1104  -0.5867
  2    6- 5   0.2027 -1.7737  0.0000   0.3307   -0.8849    0.3332  -0.6832  -0.5349  0.28  3.7726  -0.3460
  3    4- 1  -0.2059  3.4012  0.0000   0.2858   -1.0035   -0.7896   0.5492  -0.7631  0.03  8.8533  -0.2875
  4    6- 2  -1.9621 -2.8230  0.0000   0.2824   -0.9697   -0.7670   0.5333  -0.7360  0.04  8.2561  -0.2812
  5    8- 5   1.6778 -0.6139  0.0000   0.2802   -0.7099   -0.5397   0.3406  -0.5108  0.06  3.3400  -0.3099
  6    5- 3   2.2983 -2.5931  0.0000   0.2798   -0.9333   -0.7398   0.5286  -0.7221  0.02  7.5816  -0.2751
  7    7- 4  -0.6802  1.5630  0.0000   0.2619   -0.1784    0.6952  -0.4217  -0.4519  0.07  2.1419  -0.3589
  8    7- 6  -1.4000 -1.0226  0.0000   0.2446   -0.0688   -0.3527   0.6643  -0.3803  0.08  2.0563  -0.3226
-------------------------------------------------------------------------------------------------------------
```

The column ‘Basins’ shows between which basins the saddle point is located (the numbering from the ‘Volume’ section is used). The Cartesian position of the saddle point as well as the property value (here the electron density) and its Laplacian at this position together with the corresponding principal curvatures are given in next columns. From the above data it can be seen that the density Laplacian is negative at all bond critical points of the oxazole molecule.

The ellipticity at the saddle point is given as $c_1/c_2 - 1$, with the two (negative) curvatures c_1 and c_2 perpendicular to the interaction path, whereby $|c_1| \geq |c_2|$. The virial ratio $|V|/G$ is computed from the positive definite kinetic energy density $G = \nabla\nabla' \Gamma(r,r')$ and the local potential energy density is set to $V = \frac{1}{4}\nabla^2\rho - 2G$ [11]. The energy density at the saddle point is given by $H = G + V$.

Similar table is given for the ring critical points (points with 1 negative and 2 positive principal curvatures). For the oxazole molecule there is 1 ring critical point:

```
* * * (3,+1)  R I N G   P O I N T S   I N   R E G I O N * * *

-------------------------------------------------------------------------------------------------------------
 Nr. Basins      x       y       z     value  Laplacian           curvature         ellip  |V|/G      H
-------------------------------------------------------------------------------------------------------------
  1    8- 7   0.0141  0.0039  0.0000   0.0514    0.4363    0.2378   0.2551  -0.0567  0.07  0.9104   0.0090
-------------------------------------------------------------------------------------------------------------
```

The column ‘Basins’ shows only two of the basins touching at the ring point (instead of all five). The ellipticity is computed similar to the saddle point ellipticity, but using the two positive curvatures.

The last table includes data for the minima (no data for C_3H_3NO – the electron density minimum is at infinity) in the region. The closing information in the ‘Topology’ section shows the Euler characteristic inside the cropped region:

```
Euler characteristics (inside cropped grid):
              ->      8 (Attr)  -  8 (Bcp)  +  1 (Rcp)  -  0 (Min)  =  1
```

showing that the Poincare-Hopf sum is fulfilled. This need not be the case if just part of the system was calculated. As mentioned above, the sum applies only to the computed (cropped) region.

The command ‘**icl_graph**=full’ direct the program to create the interconnection lines (ICLs) between the saddle points and the attractors as well as between the ring

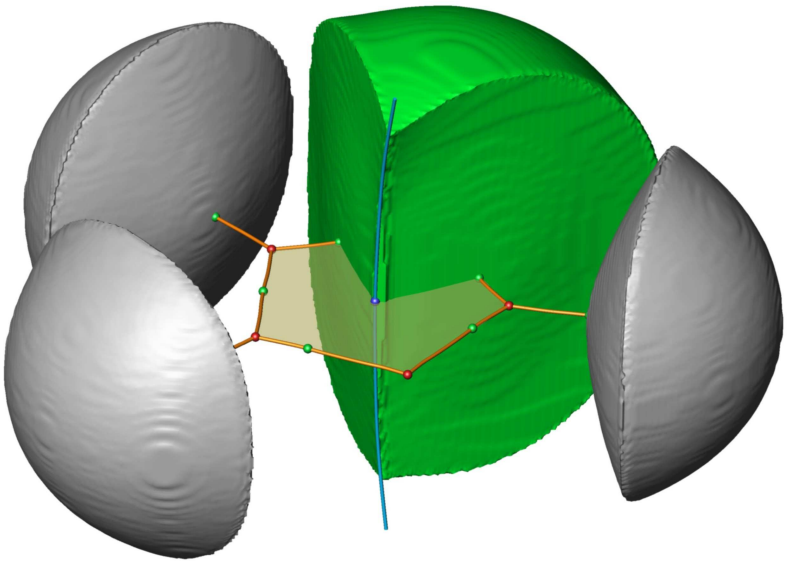

Fig. 2.3 ρ-basins (cropped by the 0.001 bohr^{-3} isosurface of electron density) and the molecular graph for C_3H_3NO. Hydrogen basins are in gray, the nitrogen basin is green colored (the oxygen and carbon basins are not shown). Red spheres: attractors; green spheres: saddle points; blue sphere: ring critical point

critical points and the minima (parameter 'full', cf. Sec. 2.8.10, page 65). The resulting data are written to the file *C3H3NO_HF_6-31G.g03.rho_r.bsn.graph.str* in the STR format. In case of electron density the diagram formed by the lines from saddles to the attractors is called molecular graph [5]. Fig. 2.3 shows (program Avizo) the molecular graph for oxazole, together with the nitrogen and hydrogen basins (cropped with 0.001 bohr^{-3} isodensity surface). It can be seen that in the saddle points (small green spheres) are located in the zero-flux surfaces (basin borders). The ring critical point (blue sphere) is located within the molecular ring. The blue line follows the gradient path to the minimum in the direction of the negative principal curvature (only the part till the grid border is drawn as it is a long way into the infinity). It can nicely be seen that the trajectory runs along the edge of the nitrogen basin.

Instead of computing the ICLs directly after the topology evaluation it is possible to perform this task with an utility, cf. Sec. 3.6). Commenting out the '**icl_graph**=full' command would result in a topology run with all the data described above, but without the ICLs. The critical points would be saved in a file named *C3H3NO_HF_6-31G.g03.rho_r.bsn.cp.str*, i.e., without the data for the lines. With this file the interconnection graph can be computed using:

```
dgrid  C3H3NO_HF_6-31G.g03.rho_r.bsn.cp.str  full
```

In case of the density evaluation of the oxazole molecule the topology and ICL graph is a short calculation. However, for larger systems and more oscillating functions with high gradients, like $\nabla^2\rho$ or ELI-D, with large number of critical points it can turn into a time consuming task. Then it is better first to perform the search for the critical points, check whether the Euler characteristics is the desired one (depending on the region or periodicity) and then create the ICL graphs (which can take lot of time as well).

In case of ELI-D there are much more basins then in case of electron density. Especially if heavy elements are involved one encounters the problem of resolving the large number of inner shell basins connected with huge number of critical points.

Remark 2.12. DGrid is working on an equidistant grid of data. Thus, it cannot resolve details hidden between the grid points. Strong oscillations in atomic regions can be taken into account only by appropriately fine mesh. At least 0.05 bohr mesh should be used to properly resolve the core-valence separation for heavier elements. Even then, due to the very high gradients and curvatures, the determination of critical points can fail.

To avoid the complicated as well as time consuming search for critical points in the core region of heavier atoms it is possible to exclude this region from the search. In case of ELI-D tabulated shell radii are utilized with the keyword **ELI_core**. Then, the procedure will not search for critical points within the core region, cf. Table 2.1.

Table 2.1 Core regions with ELI_core keyword

Atoms		Core shells
Li–Ne	$\mapsto$	1^{st}
Na–Ar	$\mapsto$	$1^{st}+2^{nd}$
K	$\mapsto$	$1^{st}+2^{nd}+3^{rd}$
Ca–Cu	$\mapsto$	$1^{st}+2^{nd}$
Zn–Kr	$\mapsto$	$1^{st}+2^{nd}+3^{rd}$
Rb	$\mapsto$	$1^{st}+2^{nd}+3^{rd}+4^{th}$
Sr–Ag	$\mapsto$	$1^{st}+2^{nd}+3^{rd}$
Cd–Xe	$\mapsto$	$1^{st}+2^{nd}+3^{rd}+4^{th}$
Cs–La	$\mapsto$	$1^{st}+2^{nd}+3^{rd}+4^{th}$
Ce–Lu	$\mapsto$	$1^{st}+2^{nd}+3^{rd}$
Hf–Au	$\mapsto$	$1^{st}+2^{nd}+3^{rd}+4^{th}$
Hg–Rn	$\mapsto$	$1^{st}+2^{nd}+3^{rd}+4^{th}+5^{th}$

The radii can be found in the constructor of the class Per_Sys, respectively in Ref. [21]

The reason for the choice of the 1^{st} and 2^{nd} shell as the core region for the series Ca–Cu is the observation that the structurization of the penultimate atomic shell seems

to be important [22] (and similarly for other series). The critical points within the core-valence separation will be searched, whereas each core region will be replaced by a single unit, called ‘core’ and treated as an attractor in the determination of Euler characteristic.

As an example let us utilize the **ELI_core** keyword in the evaluation of ELI-D of the oxazole molecule. Of course, in this case the core region replace really just a single attractor (thus, the same result would be given for the evaluation without **ELI_core**). The following *control* file:

```
:TITLE
:-------------------------------------------------------------|
::C3H3NO ELI-D basins
:-------------------------------------------------------------|

:KEYWORDS
:-------------------------------------------------------------
 property =C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn

 output=.

 ELI_core
 topology

end
```

yields a ‘Topology’ section similar to the one for the evaluation of the electron density shown above. However, the table for the attractors is now followed by a table with the data for the core region:

```
* * *  A T O M I C    C O R E S    I N    R E G I O N  * * *

-------------------------------------------------------------------------------
 Nr.  Atom  Basin       x          y          z         value        radius
-------------------------------------------------------------------------------
  1    C5      5    -1.1101    -1.8596     0.0000      17.2365        0.4350
  2    C2      4     0.0000     2.1211    -0.0000      17.2159        0.4350
  3    C4      3     1.4135    -1.6748    -0.0000      17.1006        0.4350
  4    N3      2     2.0869     0.8955    -0.0000      14.9864        0.3600
  5    O1      1    -2.0618     0.5748     0.0000      13.1296        0.3000
-------------------------------------------------------------------------------
```

For the 5 atomic cores the corresponding atomic symbols and basin numbers are given, followed by the position of the atom with the ELI-D value as well as the radius used. For ELI-D there are also 11 minima found in the computed region:

```
* * *  M I N I M A   I N   R E G I O N  * * *

-------------------------------------------------------------------------------------------------------
 Nr.     x        y        z      value     Laplacian            curvature              |V|/G     H
-------------------------------------------------------------------------------------------------------
  1  -1.9212   0.9616   0.0000    0.6680     33.7741      0.8159   32.1033    0.8549     1.0338  -0.7313
  2  -2.1364   0.1696   0.0000    0.6674     33.6895      0.8103   32.0236    0.8556     1.0330  -0.7088
  3   2.0331   0.8449   0.4841    0.6409     24.5150      0.8422    0.1686   23.5042     0.9834   0.1377
  4   2.0331   0.8449  -0.4841    0.6409     24.5150      0.8422    0.1686   23.5042     0.9834   0.1377
  5   0.4958   2.4271  -0.0000    0.6290     17.6058     16.9168    0.6421    0.0470     0.9498   0.1645
  6   1.4679  -1.6105   0.5859    0.6281     16.8136      0.3210    0.1148   16.3778     0.9455   0.1603
  7   1.4679  -1.6105  -0.5859    0.6281     16.8136      0.3210    0.1148   16.3778     0.9455   0.1603
  8  -0.8764  -2.3950   0.0000    0.6230     17.6142      0.6679   16.7976    0.1486     0.9456   0.1755
  9  -0.3759   1.9230  -0.4179    0.6201     16.8560     16.4229    0.3485    0.0846     0.9330   0.1893
 10  -0.3759   1.9230   0.4179    0.6201     16.8560     16.4229    0.3485    0.0846     0.9330   0.1893
 11  -1.3781  -1.3290  -0.0000    0.6118     18.3150      0.2749   18.0045    0.0357     0.9115   0.2545
-------------------------------------------------------------------------------------------------------
```

```
Euler characteristics (inside cropped grid):  10 (Attr) - 29 (Bcp) + 26 (Rcp) - 11 (Min) + 5 (Core) = 1
Topology data written to the file C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn.cp.str
```

The addition of the 5 cores yields the proper Euler characteristic. The data for the critical points are written to the file *C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn.cp.str* which can be used to create the ICL graphs for the oxazole molecule:

```
dgrid   C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn.cp.str   full
```

The resulting ELI-D ICL-graphs are not very complicated thus, both ICL graphs (from saddles to attractors and from rings to minima) can be shown at once, cf. Fig. 2.4. The core spheres in the figure have the radius given in the above table of atomic core regions.

Fig. 2.5 shows the same ICL graphs like Fig. 2.4. Additionally, 3 ELI-D basins are shown (corresponding to the C-O and C-N bonds). The borders of the basins (the ELI-D zero-flux surfaces) are colored according to the ELI-D value at the given position. It can nicely be seen, that the ELI-D maxima (2-dimensional; rank 2, signature -2) in the zero-flux surfaces correspond to ELI-D saddle points (3,-1), whereas

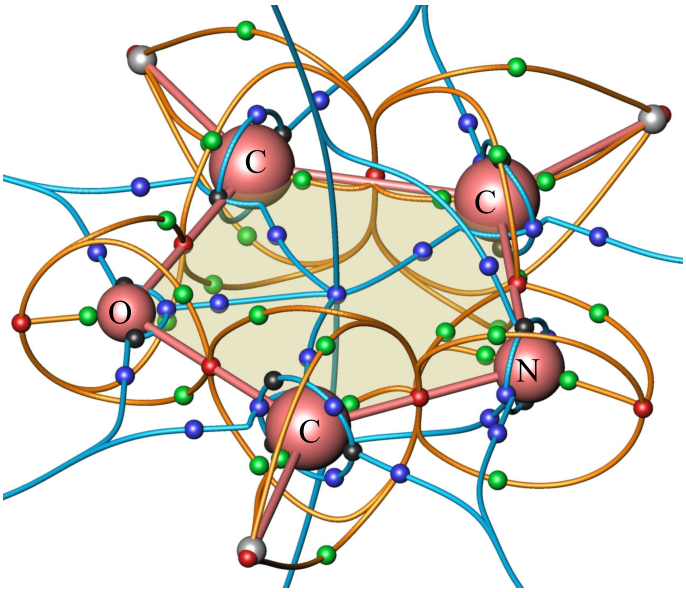

Fig. 2.4 ELI-D ICL graphs for C_3H_3NO. The small colored spheres have following encoding: red - attractors; green - saddle points; blue - ring critical points; black - minima. The hydrogen positions are marked by gray spheres. The gold colored ICLs run toward the attractors, whereas the blue ones run toward the ELI-D minima. The large rose-colored spheres are the respective core regions. The rose-colored stick are used to highlight the molecular structure

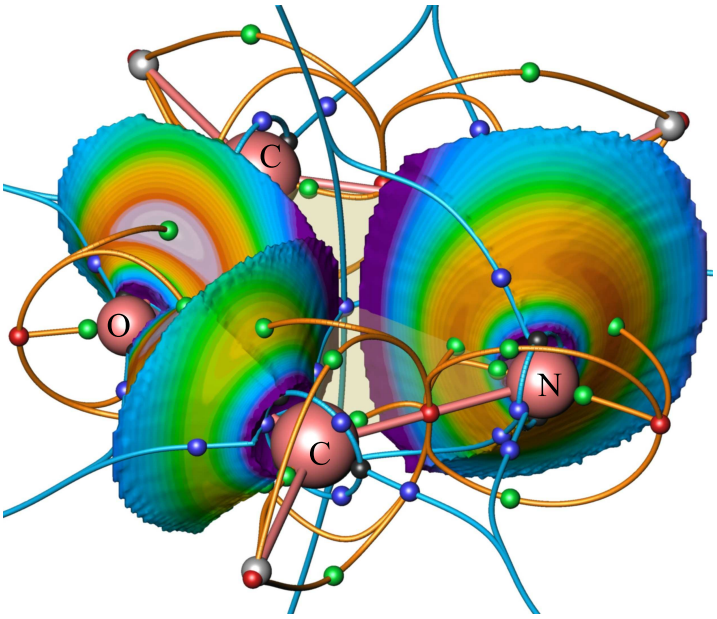

Fig. 2.5 ELI-D basins and ICL graphs for C_3H_3NO. The small colored spheres have following encoding: red - attractors; green - saddle points; blue - ring critical points; black - minima. The hydrogen positions are marked by gray spheres. The gold colored ICLs run toward the attractors, whereas the blue ones run toward the ELI-D minima. The large rose-colored spheres are the respective core regions. The rose-colored stick are used to highlight the molecular structure. The 3 ELI-D basins (corresponding to the C-O and C-N bonds) are colored according the the ELI-D value at the zero-flux surface

the ELI-D saddle points in the zero-flux surface (again 2-dimensional; rank 2 signature 0) correspond to the ring critical points (3,+1).

Remark 2.13. The search for critical points is using certain thresholds for gradient, curvatures, etc. in DGrid. If the critical points are in regions below the thresholds, then they cannot be detected. For instance in almost spherical regions around atoms in a molecule, which of course cannot be perfectly spherical, however the changes in gradient and curvatures are close to numerical instabilities. Even if the Euler characteristics is the proper one, there still could be some critical points missing (bear in mind that if an attractor *and* a bond critical point was not found the Euler characteristics remain unchanged).

2.6 Superbasins

In some cases it is desirable to unify basins into single (connected or disconnected) object, so called superbasin. Be it all the core basins of an atom, some basins in the valence region or even basins of different chemical groups.

First DGrid must identify all the basins in the computed region and write the basin data into the *basin* file, cf. previous section. Then, for instance, the *C3H3NO_HF_6-31G.g03.rho_r.bsn* file can be read with the **property** keyword and the integration of electron density over basins performed. But in contrast to the example at page 41, now with each C and H basins unified into a superbasin:

```
:TITLE
:-------------------------------------------------------------|
::C3H3NO density superbasins
:-------------------------------------------------------------|

:KEYWORDS
:-------------------------------------------------------
 property =C3H3NO_HF_6-31G.g03.rho_r.bsn
 integrate=C3H3NO_HF_6-31G.g03.rho_r

 output=.

 superbasins
   C2-H6 = 4 1
   C4-H7 = 5 3
   C5-H8 = 6 2
 superbasins_end

end
```

The definition of the superbasins starts with the keyword **superbasins**. It must be the only keyword on the line. The block must be closed with the keyword **superbasins_end**. Within the block each superbasins is given a descriptor, e.g., 'C2-H6' for the first C-H group followed by the equal sign and the basin numbers of the constituting basins. Any basins can be chosen to form a superbasins.

Remark 2.14. In contrast to a superbasin, so called basin set [22] is given by special choice of basins. Basin set is join of sets of interconnected basins where the basin interconnection points are above or equal to a specified value (the basin interconnection point is the position and property value corresponding to the basins connecting saddle point). Thus, the basin set is always a connected object. Not every superbasins is a basin set. In the left diagram of Fig. 2.6 the 0.3-localization domains are shown (isosurfaces for the density value 0.3 bohr^{-3}). It can be seen that none of the C and H density basins, as chosen in the above *control* file, can form separate basin set (by lowering the density to connect the C-H one cannot avoid that C-N is already connected as well). However, the N and C as well as the two carbon basins form (2 membered) basin sets to the interconnection value $\rho_{\mathrm{ic}} = 0.3$, whereas the hydrogen and oxygen basins remain separated. Lowering the interconnection value

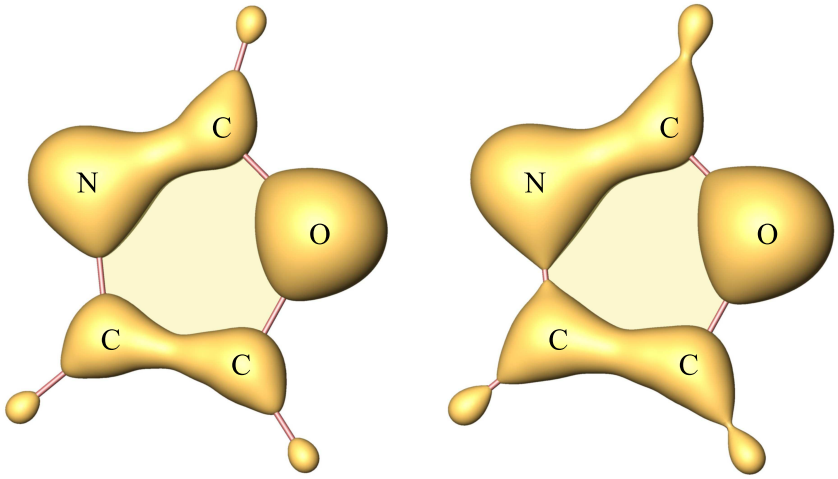

Fig. 2.6 Electron density isosurfaces for C_3H_3NO. Left: 0.3-localization domains; the N-C and C-C groups are within separate localization domain. Right: 0.281-localization domains; now the N-C-H and C-C-H groups are within separate localization domains

to $\rho_{ic} = 0.281$ gives two 3-membered basin sets (NCH and CCH) and two separate basins (H and O). For the corresponding basins cf. Fig. 2.1.

In the output an additional section preceding the basin information will appear:

```
Creating  3  superbasins:

SB   Descriptor        #   basins
-----------------------------------------------------
1    C2-H6             1   4
2    C4-H7             3   5
3    C5-H8             2   6
-----------------------------------------------------
```

showing that 3 superbasins were created, with corresponding descriptors. The numbers for the constituting basins are also repeated in the output. In the basin data following this section the correspondence between the basins and superbasins is highlighted:

```
                    rho[r]     rho[r]
BASIN    VOLUME   INTEGRAL    MAXIMUM      <X>    <Y>    <Z>    ATOMS DIST ECCNT
-------------------------------------------------------------------------------
   1 S1 103.540    5.8314     0.4320   -0.303  4.067  0.000   H6    0.04   -
   2 S3 120.776    6.4948     0.4289   -2.407 -3.341  0.000   H8    0.04   -
   3 S2 121.675    6.5722     0.4315    2.780 -3.097  0.000   H7    0.04   -
   4 S1     -        -      118.3135    0.000  2.121 -0.000   C2    0.00   -
   5 S2     -        -      118.3130    1.413 -1.675 -0.000   C4    0.00   -
   6 S3     -        -      118.3147   -1.110 -1.860  0.000   C5    0.00   -
   7     99.673    9.0038   291.3237   -2.062  0.575  0.000   O1    0.00   -
   8    116.216    7.9439   192.1030    2.087  0.895 -0.000   N3    0.00   -
-------------------------------------------------------------------------------
```

```
TOT     561.881   35.8460

Volume difference:  -1070.519

Basin data written on file C3H3NO_HF_6-31G.g03.rho_r.bsn.super
```

Here, for instance, S1 means basin belonging to the superbasin number 1, i.e. superbasin ‘C2-H6’. Observe that the total volume (as well the total integral) of the superbasin number 1 is given for the volume of the first basin member of the superbasin. For the remaining members, in this case basin number 4, there is just a period given instead of the volume and integral.

A new *basin* file, including the superbasins, with *.super* appended to the original *basin* file name is created. In this file the correspondence between the basins and superbasins is given in the basin descriptor section. Fig. 2.7 visualizes the resulting superbasin data file. Now there are only 5 basins instead of 8, cf. Fig. 2.1, because each of the previously separate C and H basins form C-H groups – the superbasins.

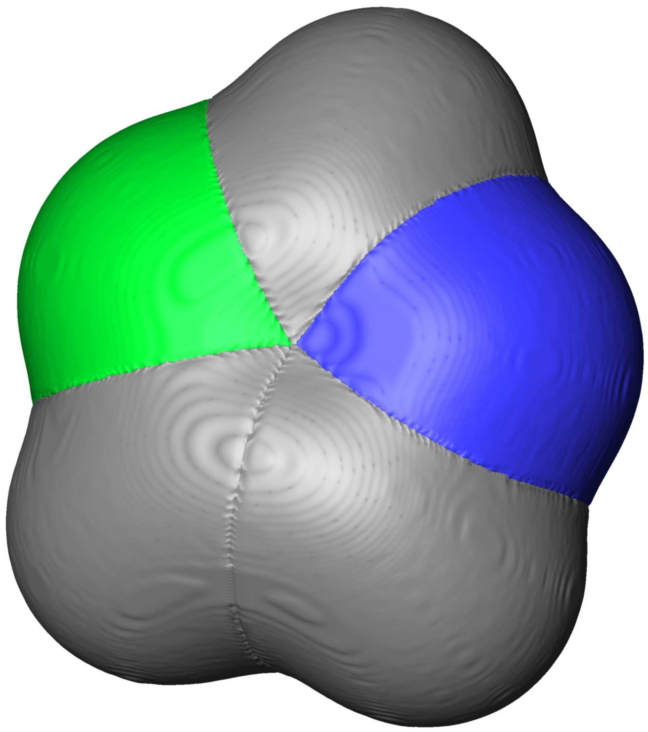

Fig. 2.7 Density superbasins for C_3H_3NO, cropped by 0.001 bohr^{-3} electron density isosurface. Blue: oxygen basin; green: nitrogen basin; dark gray: three C-H superbasins;

2.7 Overlap integrals

The integration of electron density over basins yields the basin populations (the average number of electrons within the region), which provides information about the atomic charges, respectively a hint when considering the bond order (for ELI-D basins). For the examination of chemical bonding it is of interest to analyze the electron pair density as well. In Sec. 1.6 it was already mentioned that it is possible to compute some pair-density properties, cf. Table 1.6, however there the focus was on the conditional properties. The reason was that the pair density itself is a 6-dimensional function and one need to reduce 3 dimensions to get a 3D-property grid.

In this section we will deal with integrals of electron pair density. For molecules DGrid is using Slater-type or Gauss-type orbitals approximating the wave function by one or more determinants build up from the orbitals. From this Ansatz the 1-matrix and 2-matrix is formed, providing the pivot for all the property evaluations.

Remark 2.15. In DGrid the 2-matrix is created formally by the reduction of the determinantal Absatz. It is not guaranteed that the 2-matrix properly describe the pair interaction. For instance, starting from a DFT result yielding wavefunction represented by Kohn-Sham orbitals DGrid creates the 2-matrix the same way as for a HF calculation, even if the DFT wavefunction is not meant to properly describe the local pair interactions, but 'just' the (correlated) electron density.

The electron pair density $\rho_2(\mathbf{r}_1,\mathbf{r}_2)$ is the diagonal part of the reduced 2-matrix $\rho_2(\mathbf{r}'_1\mathbf{r}'_2,\mathbf{r}_1\mathbf{r}_2)$. The 2-matrix can be decomposed into same-spin and opposite-spin contributions, which in the orbital representation can be written as:

$$\rho_2^{\sigma\sigma}(\mathbf{r}'_1\mathbf{r}'_2,\mathbf{r}_1\mathbf{r}_2) = \frac{1}{2}\sum_{i<j}^{n}\sum_{k<l}^{n} P_{ij,kl}^{\sigma\sigma}\,|\phi_i(\mathbf{r}'_1)\phi_j(\mathbf{r}'_2)|\,|\phi_k^*(\mathbf{r}_1)\phi_l^*(\mathbf{r}_2)| \tag{2.1}$$

and

$$\rho_2^{\sigma\sigma'}(\mathbf{r}'_1\mathbf{r}'_2,\mathbf{r}_1\mathbf{r}_2) = \frac{1}{2}\sum_{i<j}^{n}\sum_{k<l}^{n} P_{ij,kl}^{\sigma\sigma'}\,\phi_i(\mathbf{r}'_1)\phi_j(\mathbf{r}'_2)\phi_k^*(\mathbf{r}_1)\phi_l^*(\mathbf{r}_2). \tag{2.2}$$

Thus, the determination of the integrals of electron pair density over basins are reduced to the calculation of the overlap integrals $S_{ik} = \int_\Omega \phi_i(\mathbf{r})\phi_k(\mathbf{r})\mathrm{d}V$ of the molecular orbitals over the basins. For this the factorization of the Gauss functions can be utilized.

Remark 2.16. This factorization is necessary to achieve the speed up for the evaluation. For this reasons the calculation of the overlap integrals is performed in DGrid only for Gauss-type orbitals and only if the grid is oriented parallel with the Cartesian axes.

First, let us compute for the oxazole molecule a electron density grid parallel with the Cartesian axes:

```
:TITLE
:-------------------------------------------------|
::C3H3NO   HF    6-31G      parallel mesh
:-------------------------------------------------|

:KEYWORDS
:-------------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 result=C3H3NO_HF_6-31G_parallel

:CHOOSE THE DESIRED PROPERTIES
:----------------------------------------
 compute=rho
:----------------------------------------

 mesh=0.05   4.0   parallel

END
```

The resulting density grid is written to the file *C3H3NO_HF_6-31G_parallel.rho_r*. The grid-box is parallel with the Cartesian axes which can be seen already in the output where the definition of the grid region is written:

```
REGION :  origin = [ -6.450,  -7.350,  -4.000]

i = [ 13.250,  0.000, 0.000]   |i| =  13.250 by 266 points   d|i| = 5.000e-02
j = [  0.000, 15.450, 0.000]   |j| =  15.450 by 310 points   d|j| = 5.000e-02
k = [  0.000,  0.000, 8.000]   |k| =   8.000 by 161 points   d|k| = 5.000e-02
```

All three grid vectors i, j, k have only one nonzero x, y, z component. To compute the basins for the resulting density grid basins, cf. Sec. 2.2 (conveniently with the 0.001 cropping). For the density basins the overlap integrals are computed using the following *control* file with the keyword **overlap**:

```
:TITLE
:-------------------------------------------------------------|
::C3H3NO overlap over density basins
:-------------------------------------------------------------|

:KEYWORDS
:-------------------------------------------------------------
 property =C3H3NO_HF_6-31G_parallel.rho_r.bsn
 integrate=C3H3NO_HF_6-31G_parallel.rho_r

 output=.

 overlap

end
```

The command '**property**=C3H3NO_HF_6-31G_parallel.rho_r.bsn' tells DGrid where to find the file with basins over which to compute the overlap integrals. With the assignment '**integrate**=C3H3NO_HF_6-31G_parallel.rho_r' the basis set is chosen (namely the one used to compute the density grid).

Remark 2.17. With this combination it is possible to compute the basins with chosen basis set and then determine the overlap integrals for a different basis set over those basins. One could even create a *basin* file with artificial basins and then compute the desired overlap integrals.

2.7.1 Fluctuation and delocalization indices

The output from the overlap run is written to a file with the extension '*.ovl*' (instead of '*.bas*'). There the 'Integration' section is followed by a 'Pair density analysis' section. The progress for the calculation of the overlap integrals of the primitive functions (S_munu) and of the MOs (S_ij) is shown by running arrows. After the calculation of the overlap integrals is done the name of the file where the overlap tables are saved is given.

The resulting overlap integrals are used to immediately evaluate some indexes, because the time demand for this is usually negligible compared to the evaluation of the integrals itself. First the table for the localization indices is given

```
              +----------------------+
              | Localization indices |
              +----------------------+

Basin Descriptor   Q       Daa      Dbb      Dab    sigma2   lambda     LI
-------------------------------------------------------------------------------
   1        H6   0.865    0.006    0.006    0.187    0.514    0.595    0.350
   2        H8   0.879    0.007    0.007    0.193    0.520    0.591    0.360
   3        H7   0.908    0.008    0.008    0.206    0.527    0.580    0.381
   4        C2   4.968    2.285    2.285    6.171    1.767    0.356    3.201
   5        C4   5.662    3.084    3.084    8.015    1.968    0.348    3.695
   6        C5   5.616    3.010    3.010    7.886    1.883    0.335    3.733
   7        O1   9.001    8.170    8.170   20.256    1.169    0.130    7.832
   8        N3   7.943    6.292    6.292   15.772    1.569    0.198    6.374
-------------------------------------------------------------------------------
                 35.843
```

For each basin the descriptor and the electron population (Q) is given. The population is computed analytically and can be compared to the numerical results in the preceding 'Integration' section.

Remark 2.18. The density integrals are computed analytically in this section. However, the regions (basins) over which the integrations are performed are still the same ones formed from small cubes given by the grid mesh. Especially for small basins are coarse mesh one should not mistake the populations with 'exact' basin populations for which also an analytical determination of the zero-flux surfaces would be necessary (but the errors are very small).

In next columns the numbers of same-spin electron pairs (Daa and Dbb) as well as opposite-spin pairs (Dab) are given. The numbers are given by the integral of the corresponding pair densities, cf. Eqs. 2.1 and 2.2, with both coordinates confined to the given basin region.

The last 3 columns 'sigma2', 'lambda', and 'LI' shows the fluctuation (variance) in the average population of the respective region, the relative fluctuation, and the localization index. Bear in mind that in literature sometimes differing symbols can be found (Fradera [12] used also λ for the localization index LI, respectively σ^2 for the fluctuation, whereas Bader [7, 8] used $F(\Omega,\Omega)$ for LI, Λ for the fluctuation, and λ for the relative fluctuation). Table 2.2 should clarify the meaning of the symbols as used by DGrid.

Remark 2.19. The indices 'sigma2', 'lambda', and 'LI' are given as total values summed up over the spin components.

Table 2.2 Localization and delocalization indices

Index	Description	Equation
$N(\Omega)$	average population in Ω	$\int_\Omega \rho(\mathbf{r})\mathrm{d}\mathbf{r}$
$D^{\alpha\alpha}(\Omega)$	$\alpha\alpha$-pairs in Ω	$\iint_\Omega \rho_2^{\alpha\alpha}(\mathbf{r}_1,\mathbf{r}_2)\mathrm{d}\mathbf{r}_1\mathrm{d}\mathbf{r}_2$
$D^{\alpha\beta}(\Omega)$	$\alpha\beta$-pairs in Ω	$\iint_\Omega \rho_2^{\alpha\beta}(\mathbf{r}_1,\mathbf{r}_2)\mathrm{d}\mathbf{r}_1\mathrm{d}\mathbf{r}_2$
$D^{\beta\alpha}(\Omega)$	$\beta\alpha$-pairs in Ω	$\iint_\Omega \rho_2^{\beta\alpha}(\mathbf{r}_1,\mathbf{r}_2)\mathrm{d}\mathbf{r}_1\mathrm{d}\mathbf{r}_2$
$D(\Omega)$	average number of pairs in Ω	$D^{\alpha\alpha}(\Omega)+D^{\beta\beta}(\Omega)+D^{\alpha\beta}(\Omega)+D^{\beta\alpha}(\Omega)$
$F(\Omega)$		$N^2(\Omega)-2D(\Omega)^a$
$\sigma^2(\Omega)$	fluctuation in population in Ω	$\langle N^2\rangle_\Omega-\langle N\rangle_\Omega^2=N(\Omega)-F(\Omega)$
$D(\Omega,\Omega')$	$\mathbf{r}_1$ in Ω, $\mathbf{r}_2$ in Ω'	$\int_\Omega \mathrm{d}\mathbf{r}_1\int_{\Omega'}\rho_2(\mathbf{r}_1,\mathbf{r}_2)\mathrm{d}\mathbf{r}_2$
$F(\Omega,\Omega')$	$\mathbf{r}_1$ in Ω, $\mathbf{r}_2$ in Ω'	$N(\Omega)N(\Omega')-2[D(\Omega,\Omega')$
Q	charge	$N(\Omega)$
Daa	$\alpha\alpha$-pairs	$D^{\alpha\alpha}(\Omega)$
Dbb	$\beta\beta$-pairs	$D^{\beta\beta}(\Omega)$
Dab	opposite-spin pairs	$D^{\alpha\beta}(\Omega)+D^{\beta\alpha}(\Omega)$
sigma2	fluctuation	$\sigma^2(\Omega)$
lambda	relative fluctuation	$\sigma^2(\Omega)/N(\Omega)$
LI	localization index	$F(\Omega)$
$\delta(A,B)$	delocalization index	$F(\Omega,\Omega')^b$
$D(A,B)$	pairs between regions A and B	$D(\Omega,\Omega')$

[a] Number of pairs normalized to $N(N-1)/2$. In Ω: $D(\Omega)=\frac{1}{2}\left[N^2(\Omega)-F(\Omega)\right]$
[b] often defined as $F(\Omega,\Omega')+F(\Omega',\Omega)$

The information about localization is followed by table of delocalization indices $\delta(A,B)$, summed up over the spin components. The delocalization indices can be connected with the number of electrons shared between the basins:

```
             +------------------------+
             | Delocalization indices |
             +------------------------+

Basin     1        2        3        4        5        6        7        8
:-------------------------------------------------------------------------------
   1      x      0.000    0.000    0.444    0.005    0.005    0.023    0.027
   2    0.000      x      0.001    0.005    0.019    0.459    0.021    0.005
   3    0.000    0.001      x      0.003    0.464    0.022    0.006    0.020
   4    0.444    0.005    0.003      x      0.026    0.054    0.455    0.760
   5    0.005    0.019    0.464    0.026      x      0.795    0.071    0.562
   6    0.005    0.459    0.022    0.054    0.795      x      0.464    0.058
   7    0.023    0.021    0.006    0.455    0.071    0.464      x      0.107
   8    0.027    0.005    0.020    0.760    0.562    0.058    0.107      x
:-------------------------------------------------------------------------------
```

The delocalization indices $\delta(A,B)$ between the basins are given only for $A \neq B$. The diagonal values in the table ($A = B$) are the localization indices LI. Following sum rule is valid $\sigma^2(A)\sum_{A\neq B}\delta(A,B)$, respectively half of the sum depending on the definition of the delocalization index, see Table 2.2. Summing up the delocalization indices for basin 8 yields 1.539, which is less then $\sigma^2(8) = 1.569$ in the table of the localization indices. This discrepancy is due to the cropping of the basins. The missing part is the delocalization between the basin 8 and the region outside the 0.001-envelope (which contains 0.157 electrons, cf. the total charge in the localization table).

The last two tables in the output show the number of same-spin and opposite-spin pairs formed between the basins.

```
             +-----------------+
             | Same-spin pairs |
             +-----------------+

Basin     1        2        3        4        5        6        7        8
:-------------------------------------------------------------------------------
   1    0.012    0.190    0.196    0.852    1.222    1.212    1.934    1.704
   2    0.190    0.014    0.199    1.090    1.235    1.006    1.968    1.744
   3    0.196    0.199    0.016    1.126    1.053    1.263    2.040    1.792
   4    0.852    1.090    1.126    4.570    7.019    6.949   10.953    9.485
   5    1.222    1.235    1.053    7.019    6.168    7.553   12.706   10.962
   6    1.212    1.006    1.263    6.949    7.553    6.020   12.407   11.124
   7    1.934    1.968    2.040   10.953   12.706   12.407   16.340   17.820
   8    1.704    1.744    1.792    9.485   10.962   11.124   17.820   12.585
:-------------------------------------------------------------------------------
      303.334 same-spin pairs
```

For a closed-shell calculation the number of same-spin electron pairs within a basin (the diagonal part of the table) is twice the 'Daa' value in the table of localization indices. The total number of same-spin pairs for oxazole is $2(18\times 17)/2 = 306$. The difference between this value and the one found is again due to the cropping of the basins. The same applies to the number of opposite-spin pair and the total number of electron pairs.

2.7.2 Overlap over superbasins

The same procedure for the calculation of the overlap integrals with the successive evaluation of the fluctuation and delocalization indices can be applied to the superbasins, cf. Sec. 2.6. There are two possibilities to perform such calculation.

- Either create the *basin* file with the desired superbasin in advance by a separate job, see for example the *control* file on page 51. Then the overlap integrals can be computed with the *control* file from page 55 using the *basin* file *name.bsn.super*.

- The original *basin* file is used with the **property** keyword, but the 'superbasin' input section, cf. example on page 51, is included into the *control* file.

In both cases the data in the 'Pair density analysis' section will be given only for the superbasins, i.e., for all basins accumulated in particular superbasin only a period will appear instead of data:

```
              +----------------------+
              | Localization indices |
              +----------------------+

Basin  Descriptor    Q      Daa     Dbb      Dab    sigma2   lambda     LI
-------------------------------------------------------------------------------
   1 S1      H6    5.833   3.143   3.143    8.506    1.394    0.239    4.439
   2 S3      H8    6.496   4.022   4.022   10.549    1.486    0.229    5.010
   3 S2      H7    6.570   4.145   4.145   10.791    1.567    0.238    5.003
   4 S1      C2     -       -       -        -        -        -        -
   5 S2      C4     -       -       -        -        -        -        -
   6 S3      C5     -       -       -        -        -        -        -
   7         O1    9.001   8.170   8.170   20.256    1.169    0.130    7.832
   8         N3    7.943   6.292   6.292   15.772    1.569    0.198    6.374
-------------------------------------------------------------------------------
                  35.843
```

The correspondence between the basins and superbasin descriptors is similar to the one for the 'Integration' section as described in Sec. 2.6. The same applies to the table of delocalization indices:

```
              +------------------------+
              | Delocalization indices |
              +------------------------+

Basin     1        2        3        4        5        6        7        8
:------------------------------------------------------------------------------
   1      x      0.064    0.035      -        -        -      0.478    0.786
   2    0.064      x      0.837      -        -        -      0.486    0.063
   3    0.035    0.837      x        -        -        -      0.077    0.582
   4      -        -        -        x        -        -        -        -
   5      -        -        -        -        x        -        -        -
   6      -        -        -        -        -        x        -        -
   7    0.478    0.486    0.077      -        -        -        x      0.107
   8    0.786    0.063    0.582      -        -        -      0.107      x
:------------------------------------------------------------------------------
```

Observe that the localization index (LI) for the superbasin S1 (formed by the basins number 1 and 4) is not just the sum $3.551 = 0.350 + 3.201$ of the LI(1) and LI(4), cf. table on page 56. One needs to add the delocalization indices $\delta(1,4) = \delta(4,1)$, cf. table on page 58 as well, $3.551 + 2 \times 0.4444 = 4.439$. Thus, it is possible to

compute the superbasin localization indices from the ones for the original basins, however with inconvenience increasing with the number of basins participating on the superbasin.

The overlap matrix file contains only the overlap matrices for the superbasins (and the unchanged basins).

2.7.3 Domain-averaged Fermi-hole

Running the *control* file given on page 55 creates a new input file (with '*.sij_a*' appended, i.e., *C3H3NO_HF_6-31G_parallel.rho_r.bsn.sij_a*) containing tables with the overlap integrals over the basins:

```
OVERLAP  computed by DGrid version 4.5   28-08-2009

:job was executed on:         Fri Aug 28 13:44:41 2009

for_basin_grid=C3H3NO_HF_6-31G_parallel.rho_r.bsn

basis_from_program:      GAUSSIAN03              basis=C3H3NO_HF_6-31G.g03

property:   Sij[r]                               spin=alpha       pair=unknown

:            +------------------+
             | OVERLAP MATRICES |
:            +------------------+

:-------------------------------------------------------------
For_basins_No.:    1    2    3    4    5    6    7    8
:-------------------------------------------------------------

Precision =   0.0000000

Basin:   1

  MO        1         2         3         4         5         6         7         8         9        10
:-----------------------------------------------------------------------------------------------------------
   1    0.000000
   2   -0.000000  0.000000
   3    0.000000  0.000000  0.000006
   4   -0.000000  0.000000  0.000000  0.000000
   5    0.000000  0.000000  0.000000  0.000000  0.000000
   6    0.000001 -0.000001  0.000066 -0.000001 -0.000001  0.001610
   7   -0.000003  0.000002  0.000055 -0.000000 -0.000000  0.001256  0.001540
   8   -0.000000  0.000002 -0.000245  0.000003  0.000002 -0.005893 -0.005172  0.022260
   9    0.000007 -0.000008  0.000256 -0.000003 -0.000001  0.006494  0.004948 -0.023900  0.026959
  10   -0.000002 -0.000002  0.000513 -0.000004 -0.000002  0.012676  0.011566 -0.048502  0.051805  0.106563
  11   -0.000000  0.000003 -0.000342  0.000004  0.000002 -0.008621 -0.007742  0.032845 -0.035287 -0.072040
  12    0.000000  0.000000  0.000000  0.000000  0.000000 -0.000000 -0.000000  0.000000 -0.000000 -0.000000
  13   -0.000004 -0.000000  0.000519 -0.000003 -0.000001  0.013335  0.012486 -0.051462  0.054821  0.113780
  14    0.000007 -0.000010  0.000374 -0.000003 -0.000001  0.009936  0.008217 -0.037325  0.041454  0.081804
  15    0.000001 -0.000001  0.000195 -0.000000  0.000001  0.005066  0.004368 -0.019266  0.021154  0.042552
  16    0.000000  0.000000  0.000000  0.000000  0.000000 -0.000000 -0.000000  0.000000 -0.000000 -0.000000
  17   -0.000004  0.000005 -0.000142  0.000000 -0.000001 -0.003938 -0.003489  0.015147 -0.016774 -0.033600
  18   -0.000000 -0.000000 -0.000000 -0.000000 -0.000000  0.000000  0.000000 -0.000000  0.000000  0.000000
:-----------------------------------------------------------------------------------------------------------
```

The header start with the descriptor OVERLAP. In next lines the file name of the *basin* file as well as the information about the basis set and spin is given (here only one spin-channel is given because of closed-shell data). List of basin number for which the overlap were computed is given. The 'Precision" information has meaning only for numerical calculations. For each of the selected basins a table of overlap integrals follow. In the above example there are 18 MOs. The overlap data can be further utilized, for instance, to compute the domain-averaged Fermi-hole (DAFH) [24], cf. Table 1.6 on page 14. To compute DAFH the overlap integrals as well as

the basin to which one of the pair density coordinates is confined are introduced to DGrid with the keyword **overlap_matrix** in the *control* file:

```
:TITLE
:-------------------------------------------------|
::C3H3NO   HF    6-31G
:-------------------------------------------------|

:KEYWORDS
:-------------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 overlap_matrix=C3H3NO_HF_6-31G_parallel.rho_r.bsn.sij_a   7

:CHOOSE THE DESIRED PROPERTIES
:----------------------------------------
 compute=DAFH      alpha
:----------------------------------------

 mesh=0.1  4.0

END
```

This would compute data grid for DAFH where one electron is confined to the oxygen density basin (number 7). The DAFH can be also averaged over a superbasins (using the correspond overlap matrix file for the superbasins, see previous section).

2.8 Keywords

In this section the keywords for the DGrid *control* file concerning the basin evaluation are described in alphabetical order. The keywords are read in case insensitive. The DGrid program is now inclusive the Basin part, i.e., the calculation and examination of basins is performed by DGrid. However, there is still a separate *control* file needed for the analysis.

The keywords are given either as variables, like e.g., '**property**=', or as standalone expressions, like '**end**'. Some of them are followed by 1 or more numbers, e.g., '**top**=0.8'. If float number should be given then one must really input float numbers, otherwise the input routine will throw an error message. Thus, the command '**top**=1' would not be valid, because '1' is an integer number.

Often more keywords can be input in single line. But it is better to put every keyword in separate line, because sometimes additional data are assumed per default. Some data even need to be written in rigorous sequence. For instance the symmetry operation during the basin search using the **symmetry** keyword. Table 2.3 summarizes the available keywords.

Table 2.3 Keywords and parameters available for the basin evaluation

Keyword	Description	Value
attractors	write local maxima to file	first, last (integers, default 1-50)
close-atom_distance=	distance to nucleus	float number (default 0.1)
compactify	unify local maxima	distance, value-diff (floats)
coordinates		relative, `cell`
core_charge	include core charge	
crop	crop basins with isosurface	file name for the property field
eli_core	reduce basins inside core	
end	end of input	
extrapolate_border	extrapolate border points	
format=	result file format (basins)	cube, `dgrid`, grace, lmto
gravity	basin center of gravity	
icl_graph=	create ICL graphs	bcp, rcp, full
integrate=	integrate property	grid file name
localization_domains	search for localiz. domains	two isovalues (floats)
output=	specify output file	output file name
overlap	overlap integrals over basins	
property=	scalar field defining basins	grid file name
reduced_step=	step for trajectories	step (float), default 0.04 bohr
result=	specify result file name	generic name of the property file
structure=	specify coordinates file name	file name with atomic coordinates
superbasins	set up superbasins	list terminated by superbasin_end
symmetry=	using symmetry	mirror, translation
top=	first cut	field value (`maximum`) (float)
topology	analyze critical points	

Default values are given in `Typewriter` font

2.8.1 Attractors

With the keyword **attractors** the search for the basins is disabled. Instead, only the attractor positions will be searched and printed to the output when the *basis* file is accessible. If the *basis* file is unknown to DGrid, cf. page 36, then a separate file named *locmax.str* will be created, containing the positions of the discrete local maxima written in STR format (cf. Appendix C). Only first 50 local maxima will be printed (this is true also for the output into the *locmax.str* file). To output another number of local maxima use the command:

attractors <from> <to>

where <from> and <to> are integers giving the first and last maximum to be printed.

2.8.2 Close-atom_distance

The keyword **close-atom_distance** is followed by the distance <dist>. At the distance <dist> from the nucleus the search for a critical point stops. The default value is 0.01 bohr.

close-atom_distance=<dist>

The reason is to avoid discontinuities or very large gradients close to the nucleus.

2.8.3 Compactify

With the keyword **compactify** all attractors (respectively the local maxima) closer than <dist> (in units used for the grid definition) and differing in property value by less than <vdiff> will be assigned to the same basin, cf. page 38:

compactify <dist> <vdiff>

Especially if the wave function information is not present, as is usually the case for solid state calculations, there can be large number of discrete local maxima. The keyword **compactify** enables a grouping of otherwise large number of (possibly artificial) basins. The default value for <dist> is twice the grid point distance. The default value for <vdiff> is the property range.

2.8.4 Core_charge

The keyword **core_charge** is used to invoke the inclusion of precomputed core charge into the integration procedure, see also page 38. The achieved integration precision is less then using the grid refinement (however, it is the only possibility in DGrid to enhance the integration precision if the *basis* file is not present, like for solid state calculations).

Remark 2.20. For Gauss-type orbitals and grids parallel with the Cartesian axes the analytical integration can be performed using the keyword **overlap**.

2.8.5 Crop

The keyword **crop** is followed by the name of the *property* which is used to crop the basins. This can also be a UNIX path to this file.

crop=<XY_prop> <iso>

The basins created by DGrid using the field given by the **property** keyword will be cropped by the isosurface based on the field from <XY_prop> file with the isovalue <iso> (default is 0.001). The basins must be created during the run (not read in). As an example cf. the ρ-basins in Fig. 2.1 on page 35 cropped by the 0.001 bohr^{-3} isosurface of the oxazole electron density.

Remark 2.21. If the cropping procedure is not used then the outer basins for a molecular system extend to the borders of the computed region. Already computed basins can be conveniently cropped after the basin search using the DGrid utility 'crop', see Sec. 3.9.

Of course, with the cropping procedure part of the volume (outside the cropping isosurface) will not be considered for the integration. Consequently, the sum of the basin volumes will not recover the total volume of the computed region, cf. the line 'Volume difference' for a run with the *control* file *rho_basins.inp* (from the *example* directory, see also page 34) in the corresponding output, cf. Sec. 2.4:

```
TOT   561.881   35.8460

Volume difference:  -1070.519
```

which shows the missing volume (the sum of the the total basin volume and the volume difference yields the total grid-box volume of 1632.4 bohr^3). The charge in the excluded region (1070.519 bohr^3) will not be considered by the integration routine.

If a large grid-box around the molecule is used and electron density basins are searched, then due to the very low density (and density gradient) the search takes much more time because of corrections. In this case it is better to cut off the regions of low density with the **crop** command. If the default value removes more volume than demanded, use lower isovalue, for instance 1e-5.

2.8.6 Eli_core

The keyword **eli_core** is used to force a unification of core basins with attractors closer to the nucleus than the tabulated radius (given by the ELI-D radius of the inner core atomic shell, cf. page 38). Additionally, DGrid does not search for critical points in the core region defined as 0.75*ELI-D radius, cf. page 47.

2.8.7 End

The keyword **end** is the last keyword in the *control* file that will be parsed. Any information after this keyword will be ignored.

2.8.8 Format

The keyword **format** is followed by the descriptor <fmt> for the format of the *result* file:

format=<fmt>

Table 2.4 Format descriptors

<fmt>		Format
cube	↦	CUBE [2]
dgrid	↦	DGrid *basin* file
grace	↦	Grace [3]
lmto	↦	TB-LMTO-ASA [16]

Remark 2.22. In both the Cube and Grace formats the property values are given by float numbers. Routines which expect grids of integer values (like the basin evaluation) could fail.

2.8.9 Gravity

Using the keyword **gravity** changes the output section, where the volumes of basins are given, cf. page 41. Then, the property values are given for the center of gravity of the corresponding basin.

2.8.10 Icl_graph

The keyword **gravity** can be used in a ‘topology’ run, cf. Sec. 2.5. After the critical points are found the routine searches for the interconnection lines (ICLs):

icl_graph=<icl>

Table 2.5 ICL descriptors

<icl>		Trajectories
bcp	↦	from saddle to attractor
rcp	↦	from ring to minimum
full	↦	like bcp and rcp

The routine creates the desired ICL graphs and writes the coordinates each 0.04 bohr to an output file *name.bsn.graph.str* using the STR format. Because the search for the critical points can be heavily time demanding (and the calculation of the trajectories as well) it is possible to perform the creation of the ICL graphs using a utility, cf. Sec. 3.6.

2.8.11 Integrate

The keyword **integrate** is followed by the name of the *property* file with a grid of values to be integrated over the basins. This can also be a UNIX path to this file.

integrate=<XY_prop>

The data in the *property* file must be written in one of the formats known to DGrid (dgrid, lmto, cube). If the **integrate** keyword is omitted, then only the basin volumes are determined and printed out. If the **integrate** keyword is given, but the **property** keyword is omitted then only the total integral within the grid-box region will be determined.

2.8.12 Localization_domains

With the keyword **localization_domains** the search for basins is disabled. Instead, regions enclosed by isosurfaces (of field given by the **property** keyword) will be determined. It has the following syntax:

localization_domains <iso1> <iso2>

If the value <iso2> is not given, then the command yields isosurfaces enclosing separate regions of values higher or equal <iso1>, i.e., the <iso1>-localization domains will be determined. Setting both values <iso1> and <iso2> restricts the (separate) regions to function values between <iso1> and <iso2>.

2.8.13 Output

The keyword **output** is followed by the generic name of the *output* file. This can also be a UNIX path to this file.

output=<XY_test>

The *output* file name will be generated by appending the string '.bas' to the generic name (i.e., 'XY_test.bas' for the above example). If the name of the *output* file is not explicitly given by the **output** assignment then the output will be written to the console. Using the assignment:

output=.

generates the *output* file name from the name of the *property* file by appending the string '.bas' to it.

2.8.14 Overlap

The keyword **overlap** can be used only for Gauss-type basis set. Additionally, the grid-box must be oriented parallel with the Cartesian axes (cf. keyword **mesh**, Sec. 1.8.11):

overlap

For each basin the overlap integrals $S_{ij} = \int_\Omega \phi_i \phi_j^* \mathrm{d}V$ will be computed and the tables written to a file (with string '.sij' appended) for possible evaluation by other programs. Additionally, the localization and delocalization indices will be computed and written to the output, cf. Sec. 2.7.1.

2.8.15 Property

The keyword **property** is followed by the name of the *property* file with a grid of scalar data. The corresponding gradient field defines the basins. The name can also be a UNIX path to this file.

property=<XY_prop>

The data in the *property* file must be written in one of the formats known to DGrid (dgrid, lmto, cube). If the **property** keyword is omitted then only the total integral of the integrated property (given by the **integrate** keyword) within the grid-box region will be determined.

If the **property** keyword is followed by the name of a *basin* file then the basin search will be disabled. Instead, the basins will be directly read from the *basin* file (thus saving time).

2.8.16 Reduced_step

With the keyword **reduced_step** the search for critical points (and the interaction paths) can be done with reduced maximal step, avoiding jumps between regions:

reduced_step=<step>

The default value for <step> is 0.02 bohr.

2.8.17 Result

The keyword **result** is followed by the generic name of the *basin* file. This can also be a UNIX path to this file.

result=<XY_test>

The *basin* file name will be generated by appending the string '.bsn' to the generic name (i.e., 'XY_test.bsn' for the above example). If the name of the *basin* file is not explicitly given by the **result** assignment then the *basin* file will be generated from the name of the *property* file by appending the string '.bsn' to it.

2.8.18 Structure

The keyword **structure** is followed by the name of the *structure* file. This can also be a UNIX path to this file.

structure=<XY_struct>

Using the *dgrid* format (and *cube* format as well) the atomic positions are already included in the *property* file. However, using the *lmto* format this information is missing. Then, the keyword **structure** reads this information from external file. This can also be an LMTO-PIC file (atomic symbol, atomic radius and the xyz coordinates in 1 line for each atom). Otherwise the data must be given in the STR format (described in Sec. C.6).

2.8.19 Superbasins

The keywords **superbasins** and **superbasins_end** (on separate lines) mark a block containing the definition of superbasins, one definition per line:

superbasins
<sup_i>= <nr1> <nr2> ...
...
<sup_j>= <nr3> <nr4> ...
superbasins_end

Each superbasin is given by a descriptor <sup_i> (any string). The equal sign is followed by a list of basin numbers <nr1> (known from previous basin job) forming the superbasins. The superbasins must not overlap. The superbasin descriptor and the correspondence between the basins and superbasins will be written into the new *basin* file (with '.super' appended).

2.8.20 Symmetry

The keyword **symmetry** is followed by the descriptor <sym> for the symmetry operation and the direction in which the operation is acting:

symmetry=<sym> *i* *j* *k*

Table 2.6 Symmetry descriptors

<sym>		Format
translation	$\mapsto$	translate in given direction
mirror	$\mapsto$	mirror at the start and end of the grid region

There must be at least one of the *ijk* descriptors for the directions. The vectors corresponding to the directions are defined in the *property* file).

2.8.21 Top

The keyword **top** is used to merge basins into larger sets:

top=<val>

Each attractor has its corresponding basin. However, if the *basis* file is absent (e.g. for solid state calculation) often larger number of discrete local maxima are found around the attractor position. With the *top* value <val> DGrid 'cuts' out all grid points with property values higher than the *top* value and assign a basin to each separate closed region. Thus, choosing the value $V1$ in the Fig. 2.8 will mark all values above the $V1$ line and assign the points to the first basin. After that the basin search proceed the usual way to determine the remaining basins. Choosing as the top value $V2$ would first mark two separate regions and assign basins to it, thus avoiding possible spurious basins around the second attractor (if there were no spurious basins then the result would be the same as for the *top* value $V1$).

Sometimes it is desirable to merge few basins into a single one (creating a basin set, when the basins have common scalar field isosurface, for superbasins see Sec. 2.6). With $V3$ as the top value the third basin would form a separate region. However, the first and second basin would merge together (the values above the $V3$ line yield a single region). In this way it is possible conveniently merge together lot of basins and analyze such sets of basin as autonomous parts.

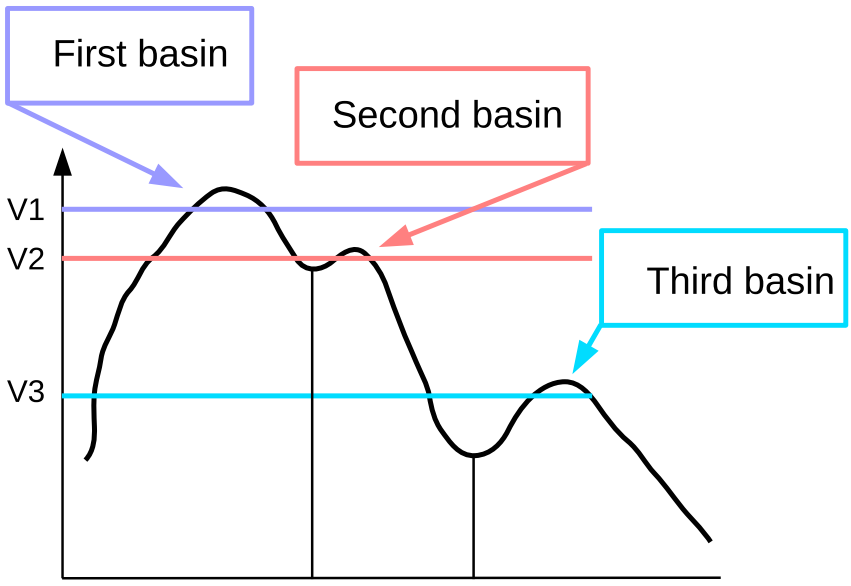

Fig. 2.8 Usage of the **top** keyword. The vertical lines mark the basin borders (saddle points)

2.8.22 Topology

Using the keyword **topology** invokes the topological analysis:

topology

It can be included in the *control* file either at the stage of the basin search or after the search, in which case the keyword **property** is followed by a *basin* file from previous run.

To perform the topological analysis the program needs to access the basins. The reason in just in the structure of the program, i.e., to invoke the search for the attractors as well as to get access to basin descriptors. For a detailed description of the results see Sec. 2.5. The topology section creates either a file (with the ending 'cp.str') in which the critical points are saved in the STR format or, if the search for the ICL graph was invoked with the **icl_grap** keyword, a file (with the ending 'graph.str') with the positions of the critical points as well as the ICL paths (in case of electron density – bond paths). The connection graph can be visualized with suitable interface (for the program Avizo see Sec. A.2). In case of the electron density evaluation the corresponding connection graph is called a molecular graph, cf. Fig 2.3.

Chapter 3
Utilities

The DGrid utilities are called from the command line with the following general syntax:

```
dgrid  filename  <kwr>  <par>
```

The possible choices of keywords <kwr> and one or more parameters <par> depend on the input file (given by the *filename*), which can be an output file from quantum chemical program, a *basis* file, *grid* file, or *basin* file, respectively. The DGrid utilities work without the *control* file. Quick reference to the syntax is given by the command:

```
dgrid  -u
```

3.1 Conversion of QM package output

As described in previous sections, the DGrid program needs a *basis* file to evaluate the chosen properties. The *basis* file is written in a specific format, cf. section 1.5, and must be created in advance from data supplied by the quantum mechanical packages.

3.1.1 ADF

In case of the program ADF [1] the formatted TAPE21 *filename.kf* (using the 'dmpkf' utility of ADF) must be convert into the *basis* file using the DGrid command:

```
dgrid  filename.kf
```

yielding the *basis* file named *filename.adf*. For localized orbitals computed with ADF the command must be extended as follows:

```
dgrid  filename.kf  locorb
```

In this case the *basis* file will be named *filename.adf_loc*.

3.1.2 Gaussian

In case of Gaussian [13] the formatted Checkpoint file *filename.fchk* (using the 'formchk' utility of Gaussian) must be convert into the *basis* file using the DGrid command:

```
dgrid  filename.fchk
```

yielding the *basis* file named *filename.g98*, respectively *filename.g03*. Another possibility is to use the information in the Gaussian WFN file:

```
dgrid  filename.wfn
```

yielding the *basis* file named *filename.gwf*.

Remark 3.1. In contrast to the *basis* file created from the Checkpoint file the *filename.gwf* does not contain virtual orbitals (as those are not present in the WFN file).

Remark 3.2. Even for a Gaussian calculation yielding at once both the Checkpoint and the WFN files the resulting *basis* files will be different! The reason is that in the WFN file the basis is already decontracted (and so will be the data in the corresponding *basis* file). In contrast, the *basis* file created from the Checkpoint file has

contracted basis and the decontraction is performed internally during a DGrid job. However, the property values from a DGrid calculation will be the same, regardless which route is chosen.

Remark 3.3. IMPORTANT!
In case of ROHF (restricted open-shell) calculation Gaussian03 does not write properly the basis set data to the WFN file. In the WFN file from such calculation all orbitals are doubly occupied, i.e., the information about the spin polarization is not given. One need to change after the conversion manually the occupation for the single occupied orbitals. The data in the WFN file from an UHF calculation are fine.

3.1.3 Molpro, Molcas, Turbomole

For the programs Molpro [34], Molcas [4], and Turbomole the basis data output must be written in the Molden [27] format. Such output file *filename.molden* in Molden format must be convert into the *basis* file using the DGrid command:

```
dgrid   filename.molden
```

producing the *basis* file named *filename.md*, respectively *filename.mp*, *filename.mc* and *filename.tm* for the recognized programs. In case of the program Turbomole include in the Molden output file manually as the second line the string:

```
[TURBOMOLE]
```

otherwise the normalization will not be the proper one.

Remark 3.4. For natural orbitals from spin-polarized calculation the occupation is declared only as the α-spin channel. DGrid distributes occupations larger then 2 equally between the spins. Occupations smaller or equal 1 is assigned to the majority spin.

3.2 Basis file conversion

An older *basis* file can be converted into the new format using the DGrid command:

```
dgrid  basisfilename  wrt
```

This yields a copy of the *basis* file named *basisfilename_1* written in new format. If the system energy or the number of electron were not present in the older file then the corresponding data will be set to zero.

3.3 Structure file

With the DGrid command:

```
dgrid  basisfilename  str
```

a structure file will be created from the *basis* file. It contains the information about the atomic positions, cf. section C.6 in the Appendix C (however, without the color code and connections part). The resulting file named *basisfilename.str* can be used by an external program to visualize the molecular structure.

3.4 AO contributions

The atomic orbital expansions of the molecular orbitals are printed by the DGrid command:

```
dgrid  basisfilename  <thres>  <occ>
```

where <thres> (float number) is the threshold for the AO coefficients (the default value for the threshold is 0.1) and <occ> the threshold for the MO occupation (with the default value 0.0). For instance:

```
dgrid  C3H3NO_HF_6-31G.g03  0.05
```

prints out each molecular orbital from the *basis* file C3H3NO_HF_6-31G.g03 as a linear combination of atomic orbitals with coefficients of absolute value larger or equal 0.05 and MO occupation larger than 0.0 (which is the default value for <occ>):

```
*****************************************************************************
AO expansion coefficients above the threshold of  0.0500    (occup >  0.0000)
*****************************************************************************

input from  GAUSSIAN03      basis file  C3H3NO_HF_6-31G.g03

entitled:   C3H3NO HF 6-31G

Total energy =    -244.5027 hartree


-------------------------
CLOSED SHELL DATA
-------------------------


NONE  #  1:          occup = 2.00000000       Energy =     -20.6521

 0.9959  O  1 ( s       )


NONE  #  2:          occup = 2.00000000       Energy =     -15.5905

 0.9962  N  3 ( s       )

...
...

NONE  # 18:          occup = 2.00000000       Energy =      -0.3677

-0.0768  O  1 ( pz  )     -0.0722  O  1 ( pz_a )     -0.3026  C  2 ( pz  )     -0.2363  C  2 ( pz_a )
-0.1675  N  3 ( pz  )     -0.1454  N  3 ( pz_a )      0.3077  C  4 ( pz  )      0.2474  C  4 ( pz_a )
 0.3764  C  5 ( pz  )      0.3033  C  5 ( pz_a )
```

The virtual orbitals will not be printed for the default <occ> value. The printing of virtual orbitals can forced by any negative value for <occ> (for instance -1.0).

3.5 Grid refinement

The integration done by the DGrid program is performed numerically on the property grid from a previous run. The precision of the integration crucially depends on the grid mesh size, which should not exceed 0.1 bohr, better is to use a 0.05 bohr mesh. Of course, the fine the mesh the better the integration result will be. However, decreasing the grid point distance by 1/2 increases the total number of grid points by the factor 8.

Higher mesh resolution is needed only in regions of highly non-linear behavior of the integrated property. DGrid can perform a refinement of the grid mesh by computing additional points in such regions. This is done by the DGrid command:

```
dgrid  gridfilename  refine  <prec>
```

where <prec> is a float number for the precision of the integration. It is connected with the maximal change for the integral contribution in a box (determined by the mesh distances) around each of points chosen by the program.

For instance, the integration of the electron density grid (0.1 bohr mesh) saved in the file *C3H3NO_HF_6-31G.g03.rho_r* (created in a DGrid job using the *control* file *rho.inp*) yields the total charge of 35.9901 electrons in the grid region (the total number of electrons for C_3H_3NO is 36). The result for this light molecule (and

0.05 bohr mesh) is close to the expected number, but the number of significant figures is certainly larger then the precision of the integration. Now, the electron density grid can be refined using the DGrid command:

```
dgrid  C3H3NO_HF_6-31G.g03.rho_r  refine  0.1
```

which performs the refinement with the precision of 0.1 electrons. With the above command the corresponding output will be written to the console. Following is the information concerning the refinement:

```
Refine grid for inhomogeneity >    0.100000

Sum of absolute inhomog. =   0.09300

0                                              414
|----*----|----*----|----*----|----*----|----*----|
|>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>|

5796 values computed. Integrals written to file  C3H3NO_HF_6-31G.g03.rho_r.rfn

Sum of absolute changes =             0.0652327
```

It can be seen, that the DGrid refined the cells around 414 points computing 5796 additional values. The refinement data (refined integrals for grid points chosen by program) are saved in the file *C3H3NO_HF_6-31G.g03.rho_r.rfn* (i.e., the original file will not be changed). The subsequent integration using a *control* file with:

```
integrate = C3H3NO_HF_6-31G.g03.rho_r.rfn
```

yields the total charge of 35.9846 electrons in the grid region. This is a small decrease by 0.0055 electron with respect to the original grid mesh.

Performing a deeper refinement using the precision parameter 0.01 results in much higher number of additional points to be computed. In this case the cells around 65 173 points are refined by additional 1 201 286 points. Of course, the time demand increases adequately. The subsequent integration using the refined data gives 35.9807 electrons, i.i., lowering by 0.0039 electron with respect to the 0.1 refinement.

The impact of the refinement procedure is much higher for functions with highly non-linear behavior, for instance for the electron density of heavier atoms or for the density Laplacian. With an appropriate *control* file the density Laplacian can be computed yielding the file *C3H3NO_HF_6-31G.g03.lap_rho_r* with data grid of 0.1 bohr mesh size (and 4 bohr border). The integral of density Laplacian over the whole space equals zero. The integral over the computed region (ca. 1639 bohr^3 around the C_3H_3NO molecule) should only slightly deviate from zero. However, the numerical integration yields -123.0404 bohr^{-5}. The refinement of the density Laplacian grid with the precision parameter 1.0 decreases the integral to -0.0917 bohr^{-5} (computing additional 1 062 752 points).

Choosing the precision parameter 0.1 changes the total integral to the value of -0.0865 bohr^{-5} (computing 6 580 850 points). The change with respect to the 1.0 refinement is already very small ($\Delta = 0.0052$). Decreasing the precision parameter further to 0.01 yields the total integral of the density Laplacian of -0.0875 bohr^{-5} (i.e., lowering by 0.001) over the computed region (at the expense of computing the huge amount of additional 23 914 342 points). The exact value for the integral of the density Laplacian over the box region is of course not zero, because of the remaining small positive contribution outside the box.

It is clear that the refinement procedure can be used this way with reasonable evaluation time for higher precision only for molecules composed of light atoms. Another possibility is to preform the refinement only over separate basins, thus saving evaluation time. This will be shown in next section on the example of integrals of the local source function.

Remark 3.5. For electron density based on Gauss-type orbitals the overlap integrals over basins can be calculated (keyword **overlap**, cf. section 2.7.1). In the subsequent evaluation of the fluctuation also the electron population in the basins is computed analytically. The remaining error in the basin population is due only to the form of the basin borders derived from a discrete grid (i.e., small cubes).

3.5.1 Evaluation of the Source Function

The source function contribution $S(r,\Omega)$ is defined as the integral of the local source $LS(r,r')$ [6, 14]

$$LS(r,r') = -\frac{1}{4\pi}\,\frac{\nabla^2\rho(r')}{|r-r'|} \tag{3.1}$$

over given region Ω

$$S(r,\Omega) = -\frac{1}{4\pi}\int_{\Omega}\frac{\nabla^2\rho(r')}{|r-r'|}\,dr' \tag{3.2}$$

It shows, how the region Ω participate on the reconstruction of the electron density at the chosen reference position r. Positive values of $S(r,\Omega)$ mark the region Ω as a source, negative values as a sink.

To compute the local source field, the reference position needs to be specified in the *control* file. The specification is done with the keyword **ref_point** followed by the Cartesian position. Following is an example for the oxazole molecule:

```
:------------------------------------------------|
::C3H3NO   HF    6-31G
:------------------------------------------------|

:-------------------------------------------------
 basis=C3H3NO_HF_6-31G.g03
```

```
 output=.

:CHOOSE THE DESIRED PROPERTIES
:--------------------------------------
 compute=LS
:--------------------------------------

 ref_point= -1.4000    -1.0226     0.0000

 mesh=0.05   4.0

END
```

The reference position is usually a saddle point in the electron density. The above position is the electron density saddle point between the atoms O1-C5, cf. the saddle point Nr.8 in the output of the 'Topology' run on page 45, in Sec. 2.5. The mesh size is set to 0.05 bohr. This is the recommended value to get better basin borders (zero-flux surfaces). In the resulting *grid* file for the local source contains the following lines:

```
property:   Local-source[r]          spin=both       pair=unknown

Ref_point= -1.4000 -1.0226 0.0000   ref_density= 0.24458
```

Thus, the *grid* file bears not only the information about the reference position, but also the density at that position. The (exact) integral of the local source over the whole space should recover this value. As the computed region is just a part of total space and the integration is done numerically, the integral will deviate from the density at the reference position. As proposed by Gatti [6, 14], the source function contribution $S(r,\Omega)$, i.e., the local source integrals over the basins, are given by DGrid as percentage source contribution to the density at the reference point.

Integrating the above local source field from the file *C3H3NO_HF_6-31G.g03.ls_r* yields value much larger than the density $\rho(r_b) = 0.24458$ bohr^{-3} at the saddle point. It gives 1531% (!) of the expected value. Refining the local source grid by the precision parameter 0.1 (computing additional 632 876 points) yields by integration 102.13% of the reference density. Increasing the precision with the refinement parameter 0.01 (evaluating 3 795 276 additional points) recovers by integration 100.54% of the reference density. Further increase of the precision (refinement parameter 0.001) does not significantly change the recovering of the density at the reference position (100.51% – at large distances there is a charge depletion, i.e., positive electron density Laplacian; those remaining contributions from the missing volume will reduce the recovering value to 100%).

The above procedure just shows a test of the quality of the source function. In the analysis of the source function contribution one is interested, how particular basins contribute to the density at the reference (saddle) point. One possibility is to refine the total local source grid and then perform an integration of that grid over the basins. The basins can be created by a separate run with the *C3H3NO_HF_6-31G.g03.rho_r* grid , cf. page 34 (0.05 bohr mesh size, cropped by 0.001 bohr^{-3}

density isosurface), yielding the *basin* file *C3H3NO_HF_6-31G.g03.rho_r.bsn*. The *control* file for the source function contributions with the refined local source grid (refinement parameter 0.01) and the mentioned *basin* file reads as follows:

```
:-------------------------------------------------------------|
::C3H3NO source function contribution over density basins
:-------------------------------------------------------------|

:-------------------------------------------------------------
 property =C3H3NO_HF_6-31G.g03.rho_r.bsn
 integrate=C3H3NO_HF_6-31G.g03.ls_r.rfn

 output=.

END
```

DGrid reads the basin data directly from the *basin* file without the time consuming basin search. The *C3H3NO_HF_6-31G.g03.ls_r.rfn* file contains not only the refinement data but also the name of the local-source grid file, which will be read in by DGrid. Following is a part of the *output* file:

```
          Local-source[r]    rho[r]
BASIN    VOLUME  INTEGRAL   MAXIMUM    <X>    <Y>    <Z>    ATOMS DIST ECCNT
-------------------------------------------------------------------------------
   1 R   40.542    1.4633    0.4320  -0.303  4.067  0.000   H6    0.04   -
   2 R   41.243    5.2369    0.4289  -2.407 -3.341  0.000   H8    0.04   -
   3 R   42.715    1.7967    0.4315   2.780 -3.097  0.000   H7    0.04   -
   4 R   62.999    2.2427  118.3135   0.000  2.121 -0.000   C2    0.00   -
   5 R   78.960    6.5979  118.3130   1.413 -1.675 -0.000   C4    0.00   -
   6 R   79.533   42.6020  118.3147  -1.110 -1.860  0.000   C5    0.00   -
   7 R   99.673   41.2473  291.3237  -2.062  0.575  0.000   O1    0.00   -
   8 R  116.216    3.7839  192.1030   2.087  0.895 -0.000   N3    0.00   -
-------------------------------------------------------------------------------
TOT     561.881  104.9707

Volume difference:  -1070.519
```

Each basin number is followed by an ‘R’ showing that the integrated data were refined for the basin. The volume difference at the end is a hint that the basins were cropped. Thus, part of the volume of the box-region was not considered. The impact of the cropping is that the recovering of the electron density at the reference position amounts to ca. 105%.

It can be seen that all the atomic basins are contributing as a source of the density at the saddle point between the oxygen and carbon nr. 5, with main contributions from the the neighboring basins (basins nr. 6 and 7).

In the above example the evaluation of the source function was not too expensive. With this advantage it was possible to refine the whole local source grid. This is not the case when heavy atoms and large basis sets are involved. Such refinement could take hours! Then it is possible to perform the refinement for local source in the desired basins only. In this case the command for the refinement includes the *basin* filename and reads as follows:

```
dgrid  gridfile  refine  <prec>  basinfile  <bas1> <bas2> ...
```

where <bas1> <bas2> etc. is a sequence of integer numbers for the chosen basins inside which the refinement will be performed.

As an example let us take the same data as before, but perform the source function evaluation for the basins number 7 and 8 (the oxygen and nitrogen atomic basins):

```
dgrid C3H3NO_HF_6-31G.g03.ls_r  refine 0.01
          ↪  C3H3NO_HF_6-31G.g03.rho_r.bsn  7  8
```

The information about the basins is included in the resulting refinement file. For the above refinement 826 616 additional points were computed (cf. the 3 795 276 points for the refinement with the same precision parameter computed previously in the total box-region). Taking closer look on the participation of basin number 8 (nitrogen) on the recovering of the saddle point electron density shows a change from 3.78% to 3.83%:

```
          Local-source[r]    rho[r]
BASIN    VOLUME  INTEGRAL   MAXIMUM    <X>    <Y>    <Z>    ATOMS DIST ECCNT
-------------------------------------------------------------------------------
   1     40.542    1.4646    0.4320  -0.303  4.067  0.000   H6    0.04   -
   2     41.243    5.2395    0.4289  -2.407 -3.341  0.000   H8    0.04   -
   3     42.715    1.7981    0.4315   2.780 -3.097  0.000   H7    0.04   -
   4     62.999   50.6511  118.3135   0.000  2.121 -0.000   C2    0.00   -
   5     78.960   28.8798  118.3130   1.413 -1.675 -0.000   C4    0.00   -
   6     79.533 1416.7646  118.3147  -1.110 -1.860  0.000   C5    0.00   -
   7 R   99.673   41.2407  291.3237  -2.062  0.575  0.000   O1    0.00   -
   8 R  116.216    3.8260  192.1030   2.087  0.895 -0.000   N3    0.00   -
-------------------------------------------------------------------------------
TOT     561.881 1549.8644
```

The reason for this change is that now the precision parameter is valid for the integral of just two basins, i.e., the relative precision is lower. For the integral of the whole box-region the precision is roughly 0.01 of 0.24 (saddle point density), i.e., ca. 4% error, whereas for the chosen basins (recovering around 45% of the saddle point density) it corresponds to 0.01 of ca. 0.11, i.e., around 9% error. The results are safe by using the precision parameter of 0.001 for both routes. Of course, this increases the amount of computed points.

The stability of the integration should be checked with increased precision of the refinement. Especially, when heavier atoms are involved.

3.6 ICL graph creation

Performing the search for critical points without the keyword **icl_graph**, cf. Sec. 2.5, yields a file (with the extension '.cp.str') with the coordinates of the critical points.

Using this file, the interconnection line (ICL) graph can be created using the command:

```
dgrid  filename.cp.str   <icl>
```

where <icl> is one of 'bcp', 'rcp', or 'full'. With 'bcp' the interconnection lines from saddle points to the attractors will be created. Using 'rcp' creates the ICLs from ring critical points to the minima (cages), whereas 'full' creates both graphs in one run. The resulting data (the coordinates of the critical points as well as the steps along the ICLs) are written in the STR format to a file with '.graph.str' appended.

3.7 Operations on single grid

The utility operates on data of a single *property* grid. It is invoked by the DGrid command:

```
dgrid  gridfilename  op1
```

DGrid reads in the *grid* file and asks for the desired operation which can be one of:

```
Available operations:
--------------------------------------------------------------------------------
  +         -             x          ->  number
  /         ^             over       ->  number

  mirror    translation   inversion  ->  {i, j, k, in, jn, kn}
                          convert    ->  {cube, dgrid, grace, lmto}
                          spin       ->  {alpha, beta, both, singlet, triplet}
                          pair       ->  {alpha-alpha, beta-beta, triplet-pair}

  sqr       crop          reduce
  elf2eli   eli2elf
  trp2aa    aa2trp

  save      quit
--------------------------------------------------------------------------------
```

The chosen operation is performed at each grid position. Some of the operations need additional parameters. After the operation is done, DGrid awaits the next operation. The transformed grid data are held in the memory. The procedure is finished either by exiting the program without saving the data (command '**quit**') or by saving the data (command '**save**') in which case the resulting grid will be written to the file with the same name as the original file with an appendix reflecting the chosen operation. Following is the description of the operations.

The operations '$+$, $-$, $\times$, $/$, ^, **over**' need a number as a parameter in the input line. For instance the command '$\times$ 5' will multiply the property value at

each grid point by 5. With '**over** 2' the value 2 will be divided by the property value at each grid point. The operation **sqr** takes the square root of each grid value and need no additional parameters.

The operations '**mirror**, **translation**, **inversion**' perform the given symmetry operation of the grid. An additional parameter, which is one of '`i j k`', respectively '`in jn kn`'is needed, determining at which position the operation is acting. Thus, the command '**mirror** `j`' will mirror the grid in the `j`-direction at the origin of the `j`-axis, whereas '**mirror** `jn`' will perform the mirroring at the end of the `j`-axis. The resulting grid will be twice as large as the original one. The command '**inversion** `k`' will perform an inversion of the grid in the `k`-direction around the grid midpoint of the `ij`-plane. This utility can be used for molecules or solid with the mentioned symmetry to save computational time, for instance, to compute just one octant for a homonuclear dimer and enlarge the grid by mirroring.

The operations **spin** and **pair** are used to set the appropriate value for the spin and pair-spin variable in a grid file.

The last operation which needs an additional parameter is **convert**. It is followed by the string determining to which format the grid file should be converted. The possible formats are given on page 24. It is also possible to use this utility to convert files written with older DGrid versions into the current 4.5 format (in some cases not all data will be available, like e.g. the energy of the system, which will be set to a default value).

The **crop** operation will ask for the name of the file containing the molecular structure (in STR or PIC format). Then all grid values within the atomic radii (given in the structure file) will be set to zero (cropped).

The **reduce** operation can be used only for grid files with odd number of points in each direction (i.e., even number of intervals). The number of grid points will be reduced in such way, that every second point in each direction will be removed without changing the size of the region-box (doubling the mesh size).

With the operations **elf2eli** and **eli2elf** the ELF grid files can under specific conditions formally be converted into ELI-D grid files (and vice versa). The conditions are that the property must stem from a single-determinantal wavefunction, the occupations must not be fractional and in case of triplet-coupling only restricted basis is allowed.

The operations **trp2aa** and **aa2trp** convert ELI-D files for closed-shell calculation, in which case there is a simple conversion factor between the single spin-channel and the triplet-coupled form of ELI-D [23].

3.8 Operations on two grids

Sometimes it is useful to add, subtract, multiply, or divide the values in two grids, for instance, to create a grid with promolecular electron density or compute a density difference maps. The operation can be accomplished by the DGrid command:

```
dgrid  gridfilename1  <oper>  gridfilename2
```

DGrid reads in the two *grid* files (which must have identical mesh size and grid dimensions) and performs the required operation at each grid position. The possible operations <oper> are summarized in Table 3.1.

Table 3.1 Operations on two grids

<fmt>		Format	
$+$	$\mapsto$	add the grid values	$f_1(x,y,z)+f_2(x,y,z)$
$-$	$\mapsto$	subtract the grid values	$f_1(x,y,z)-f_2(x,y,z)$
x	$\mapsto$	multiply the grid values	$f_1(x,y,z)*f_2(x,y,z)$
$/$	$\mapsto$	divide the grid values	$f_1(x,y,z)/f_2(x,y,z)$

The resulting grid values, with the same grid dimensions as the original ones, will be written to the file named *grid.add*, *grid.sub*, *grid.mult*, or *grid.div*, respectively. The header of the resulting *grid* file is identical with the header of the first *grid* file on the input line (only the title is changed). This means that, for instance the spin labels will be the ones from the first *grid* file. Such data, if desired, must be changed manually.

3.9 Cropped basins

This utility works on basin files created by the DGrid program, i.e., grids of integer values yielding the basins.

For a molecule the outer basins extend to infinity, i.e., such basins are bounded by the region-box chosen for the grid calculation. However, it might be useful to crop the basins by an isosurface of another property field, for instance the electron density. Using the 0.001 bohr^3 density isosurface would correspond to basins cropped by the 'molecular envelope'. The procedure is accomplished by the DGrid command:

```
dgrid  basinfilename  crop  <val>  gridfilename
```

DGrid reads in both the *basin* file and the *property* file (the files must have the identical grid dimensions and mesh size). Each basin grid point outside the <val>-isosurfaces (i.e., outside the <val>-localization domains) will be set to zero (i.e., marked as no basin). The resulting cropped *basin* grid, together with the information

about the cropping isovalue and the *property* file name, will be written to the file with the same name as the *basin* file with the string '*.crop*' appended.

The procedure yields the same result as a basin search with the command **crop** included in the *control* file, cf. Sec. 2.8.5. The utility can be used if one do not want to perform a new basin search for different cropping isovalues.

3.10 Basin intersections

This utility works on basin files created by the DGrid program, i.e., grids of integer values yielding the basins written in the DGrid format.

Basins computed for two different scalar fields yield two different sets of space partitioning. It can be of interest to examine, how a particular basin from the first set is intersected by the basins of the second set [25]. Of course, not all basins of the second set will intersect the chosen basins. The intersection utility has the following syntax:

```
dgrid  basinfilename1  intersect  <bas>  basinfilename2
```

DGrid reads in the two *basin* files (which must have identical mesh dimensions). In the first *basin* file the basin number <bas> is chosen. Then, the intersections of this basin with the basins of the second set are determined. The resulting basin grid containing only the intersections of basin number <bas> is written to the file with the same name as the first *basin* file with the string '.*<bas>_isect*' appended. The extent (in percent) to which the basins of the second set intersect the chosen basin is given in the output (written to the console). Of course all the intersections together have the same volume as the intersected basin.

For example, for the C_3H_3NO molecule two grids for the electron density and ELI-D, respective, will be computed using the following *control* file (with the Gaussian03 *basis* file *C3H3NO_HF_6-31G.g03* from the *example* directory):

```
:TITLE
:-------------------------------------------|
::C3H6   HF    6-31G
:-------------------------------------------|

:KEYWORDS
:-------------------------------------------
 basis=C3H3NO_HF_6-31G.g03

 compute=rho
 compute=ELI-D     triplet

 mesh=0.05  4.0

END
```

Two grid files will be written, namely *C3H3NO_HF_6-31G.g03.rho_r* for the electron density and *C3H3NO_HF_6-31G.g03.elid_r_t_tr* for the ELI-D. The corresponding *basin* grids will be created with the following *control* file (for the ELI-D grid by changing the property file name accordingly):

```
:TITLE
:-------------------------------------------------|
::C3H3NO basins
:-------------------------------------------------|

:KEYWORDS
:-------------------------------------------------
 property=C3H3NO_HF_6-31G.g03.rho_r
 crop    =C3H3NO_HF_6-31G.g03.rho_r      0.001

END
```

Note that the basins will be cropped by the 0.001 isosurface of the electron density. The basin search yields the file *C3H3NO_HF_6-31G.g03.rho_r.bsn* with 8 electron density basins, respectively the file *C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn* describing 15 ELI-D basins.

According to the QTAIM approach the electron density basins, cf. Fig. 2.1 on page 35, describe the atoms in molecule [5]. Because non-nuclear maxima are not present in case of the C_3H_3NO molecule, the number of the density basins corresponds to the number of atoms. In case of ELI-D the number of basins is larger than for the electron density, because there are separate basins corresponding to the ELI-D attractors between the atoms (which can be assumed as bond descriptors) and around the oxygen and nitrogen core basins (lone-pair descriptors), cf. Fig. 2.2 on page 43.

There are always two possibilities for the intersection analysis depending on the choice of the intersecting set (*A* intersect *B*, or *B* intersect *A*). In the above case either an ELI-D basins is intersected by the electron density basins or vice versa.

With the command:

```
dgrid C3H6_HF_6-31G.g03.elid_r_t_tr.bsn  intersect  6
          ↪  C3H6_HF_6-31G.g03.rho_r.bsn
```

the ELI-D basin number 6, which corresponds to the ‘bond’ basin between the oxygen and carbon C5, will be intersected by the electron density basins. The output will be written to the console. It prints the following information concerning the intersection of the ELI-D basin:

```
Intersections of ELI-D[r] basin No. 6 (O1-C5):

 rho[r]
 basin  descriptor      volume          %
-----------------------------------------
   6           C5       0.939       17.19
```

```
   7            O1       4.523        82.82
-----------------------------------------
                          5.462       100.00

Basin intersections written to file
-> C3H3NO_HF_6-31G.g03.rho_r.bsn.isect_C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn_6
```

The result shows that the ELI-D basin number 6 (O1-C5 bond) is intersected by two density basins, namely the ones corresponding to the atoms O1 and C5, respectively. The density basin of the oxygen (the oxygen 'atom') participate to almost 83% on the ELI-D basin volume. The resulting basin intersections are saved in a *basin* file (with the long name documenting that the density basins intersected the ELI-D basin number 6), which actually contains the density basins within the region of ELI-D basin nr. 6. The file can be used for the integration of the electron density to yield the charges within each intersection. The intersection charges can be connected with the bond polarity [25]. In the case of the above intersection the oxygen contribute with 84.8% to the charge in the O1-C5 ELI-D bond basin, whereas the C5 contribution amounts to 15.2% only ($1.01 + 0.18 = 1.19$ electrons total). Thus, the bond can be regarded as polar, with the oxygen as the more negative bonding partner.

Intersecting the ELI-D basin number 9, which can be assigned to the N3-C2 bond, with the electron density basins of the oxazole yields somewhat different picture:

```
Intersections of ELI-D[r] basin No. 9 (N3-C2):

 rho[r]
 basin  descriptor      volume           %
-----------------------------------------------
   7            O1       1.139         2.80
   8            N3      15.339        37.69
   4            C2      24.218        59.51
-----------------------------------------------
                         40.696       100.00

Basin intersections written to file
  -> C3H3NO_HF_6-31G.g03.rho_r.bsn.isect_C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn_9
```

Now the larger volume contribution of 59.5% is given for the carbon density basin. Nevertheless, the integration of the electron density within the intersections yields more charge (58%) for the nitrogen density basin then for the carbon basin (1.64 electrons for N and 1.15 electrons for C). The N-C bond can be regarded as slightly polar with the nitrogen as the more negative participant. The intersection of the two ELI-D basins by the electron density basins are shown in Fig. 3.1.

Other possibility is to intersect chosen electron density basin with the ELI-D basins. Using the command:

```
dgrid C3H3NO_HF_6-31G.g03.rho_r.bsn intersect 7
      ↪ C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn
```

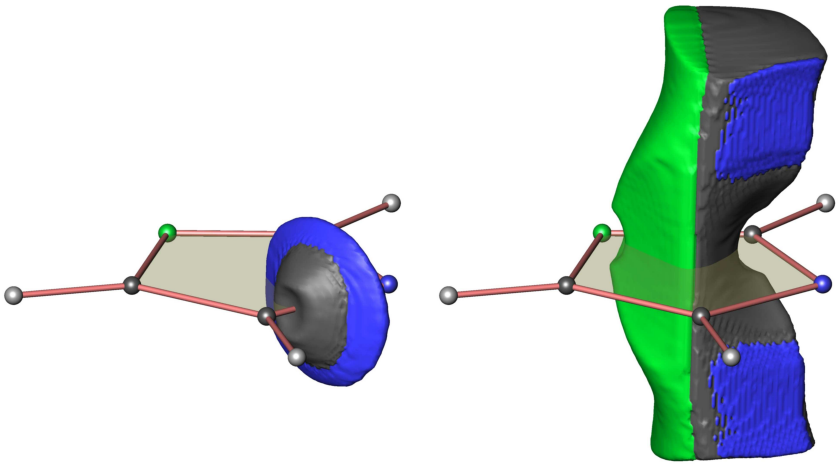

Fig. 3.1 ELI-D basins (cropped by the 0.001 bohr^{-3} isosurface of electron density) of C_3H_3NO intersected by electron density basins. Left: intersection of ELI-D bond basin O1-C5. Right: intersection of ELI-D bond basin N3-C5. The color coding is blue (oxygen), green (nitrogen), black (carbon), gray (hydrogen)

the electron density basin number 7, which corresponds to the basin of the oxygen atom, is intersected by the ELI-D basins. On the console the following output appears:

```
Intersections of rho[r] basin No. 7 (O1):

 ELI-D[r]
 basin  descriptor      volume           %
-------------------------------------------
   1            O1       0.268         0.27
  10            H8       0.706         0.71
  12            H6       0.908         0.91
   9         N3-C2       1.139         1.14
  15         C5-C4       1.716         1.72
   6         O1-C5       4.523         4.54
   7         O1-C2       4.688         4.70
  13            LP      85.723        86.00
-------------------------------------------
                        99.673       100.00

Basin intersections written to file
  -> C3H3NO_HF_6-31G.g03.elid_r_t_tr.bsn.isect_C3H3NO_HF_6-31G.g03.rho_r.bsn.7
```

The oxygen density basin (QTAIM basin) is intersected mainly by 3 ELI-D basins, with the largest contribution of the ELI-D oxygen lone-pair (86%, marked LP). The two other contributions of roughly 5% are the intersections of the density basin with the respective O-C ELI-D bond basins. Observe that the ELI-D core basin of the oxygen take up just a small fraction (ca. 0.3%) of the QTAIM basin volume (however, containing 2.1 electrons). The 3 intersections are shown in Fig. 3.2.

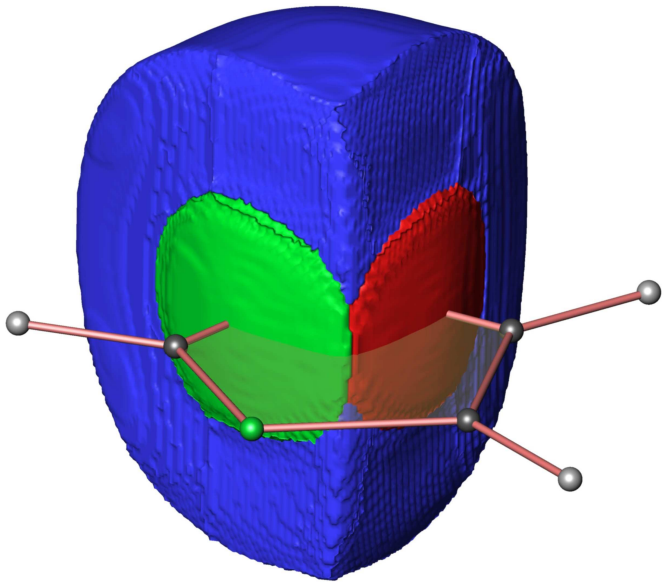

Fig. 3.2 Electron density basin (cropped by the 0.001 bohr^{-3} isosurface of electron density) for the oxygen in the C_3H_3NO molecule intersected by ELI-D basins. The blue intersection correspond to oxygen lone-pair, the red and green intersections are due the respective C-O bond basins

Appendix A
Visualization with Avizo

DGrid does *not* perform visualization of any kind of the computed value fields. For this a separate visualization tool (not included in the package) is required. The value fields from a DGrid calculation are written in a special format, which needs to be recognized by the visualization tool of your choice. There must be an interface provided by you (for instance a program reading the data in DGrid format and writing a file in the format suitable for the visualization tool). Some visualization tools understand the Cube format [2], which is supported by DGrid. For the visualization program Avizo there is a module for the DGrid data available. The module is described in the following section.

A.1 Avizo module installation

Information about the visualization package Avizo can be found at the web page of the distributor, http://www.vsg3d.com. The program is able to read many graphical standards, but DGrid format is not included. To visualize DGrid results an interface (module) is available as separate part of the DGrid package. There are 2 ways how to implement the module:

- For 64-bit Linux version of Avizo precompiled module library is available. The installation is described in Sec. A.1.1.

- Otherwise, so called developer version of Avizo is needed to compile the source code of the module. This route is described in Sec. A.1.2.

In both cases, the module set up is described only for the Linux version of Avizo. However, the set up for the Windows version of Avizo is similar to the installations described below.

A.1.1 Precompiled Avizo module

The precompiled library for the interface is called *libdgbas.so* and works with the 64-bit Linux version (created and tested on SuSE 11.1 operating system). The library must be copied into the following place in the Avizo root directory:

```
lib/arch-LinuxAMD64-Optimize/
```

Additionally, so called resource file is needed. For the DGrid module it is the *dgbas.rc* file. Put it into:

```
share/resources/
```

Now the interface should be available to Avizo. The usage of the module is described in Sec. A.2.

The *libdgbas.so* library is not included in the DGrid package. Send me an e-mail if you want to install the precompiled module.

A.1.2 Compilation of Avizo module

To compile the module source the developer version of Avizo is needed (which is more expensive the evaluation version). Few step must be completed in exact order.

1. Start Avizo and click on Help → Development Wizard in the main menu. A box will appear. Select Set local Avizo directory and click on Next.
2. Choose your local Avizo directory (for instance, create the directory *AvizoLocal* in the Avizo installation directory). Click on OK (Avizo will copy some files into the local directory).
3. Select Add package to local Avizo directory and choose the package name *dgbas*. Click on OK. Avizo informs you that new package was created.
4. Close the box and leave Avizo (exit).
5. You will find the local directory at the chosen location (for instance, *AvizoLocal* in the Avizo installation directory). In *AvizoLocal* is the directory *src* including the package directory *dgbas*.

6. Copy the module source files from the DGrid package directory *avizo/src* into your local package directory (*AvizoLocal/src/dgbas*).

7. Start Avizo and click on Help → Development Wizard in the main menu. In the box select Create build system and click on Next.

8. GNUmakefiles should be selected. Click on OK.

9. Close the box and leave Avizo (exit).

At this stage all the makefiles should be prepared by Avizo for compilation. For some reasons, unknown to me, one step is not performed by Avizo. To patch the new module into the Avizo system the routine *taglib* is needed in the *AvizoLocal/bin* directory but not copied to this location. Thus, copy (respectively make a link):

```
bin/arch-LinuxAMD64-Optimize/taglib
```

from the Avizo installation directory into:

```
AvizoLocal/bin
```

Avizo will not recognize the DGrid format unless the resource file *dgbas.rc* from the DGrid package directory *avizo* is copied into your local package directory *AvizoLocal/src/dgbas/share/resources*.

The last before the compilation is set the make variable to ‘optimize’ on the command line:

```
MAKE_CFG=Optimize
export MAKE_CFG
```

Now change to the local module directory:

```
cd AvizoLocal/src/dgbas
```

and compile the module with the command:

```
gmake
```

The last message from the compilation should be ‘processing libdgbas.so’.

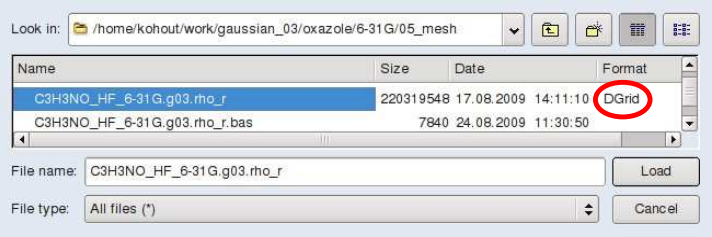

Fig. A.1 Avizo selector box for grid files. The format of the grid files must be recognized

A.2 Usage of Avizo module

For the usage of the program Avizo read the corresponding user's guide. After successful compilation Avizo is ready for the DGrid format. To test it prepare some property grid (for instance the electron density grid for oxazole given in the manuscript, cf. Sec. 1.6. page 7).

1. Start Avizo and click on the green button Open Data... which opens a selector box.

2. In the selector box the format of the grid file is marked as 'DGrid', cf. Fig. A.1.

3. Select the grid file. It will be read in and in the 'Pool' window of Avizo a green box appears with the name of the grid file.

Remark A.1. If the DGrid format was not listed in the selector box and cannot be chosen, then there is possibly a problem with the module installation, respective the location of the *dgbas.rc* file.

The visualization of slices or isosurfaces for the grids and the usage of all the tool is described in Avizo user's guide. Although given in Avizo manual, short guide how to create basins from *basin* files will be given in Sec. A.3. In the following the focus is on the *structure* module, which is not part of Avizo and the description thus cannot be found in Avizo manual.

Grid files in DGrid format, respectively in Cube format, include also the structure data of the molecule or atoms within the unit cell (which is not the case for the

Fig. A.2 Initialization of Avizo STR module. Left: Green box with the electron density grid data. Right: Creating yellow box with the structure data for the oxazole molecule

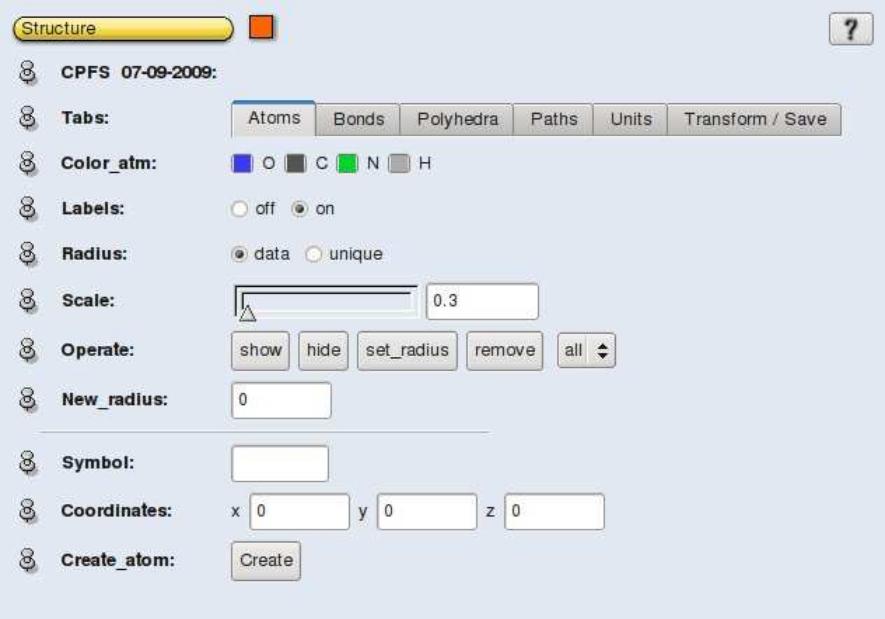

Fig. A.3 Avizo STR module – Atoms tab

LMTO format). To visualize the structure click in the 'Pool' window on the green box representing the electron density grid. Click on the Structure button below the Open Data... button, cf. the left window in Fig. A.2 (if the Structure button is not present, rightclick on the green grid box and select from the menu Local → Structure). Yellow box representing the structure appears in the 'Pool' window, cf. the right window in Fig. A.2. The click on the yellow structure box shows the available tools within structure module in the 'Properties' window of Avizo. The structure module tools are accessible via tabs.

A.2.1 Atoms tab

The buttons for the Atoms tab are shown in Fig. A.3.

- Color code is assigned for each atom type ('Color_atm'). With a click on the colored square next to the atomic symbol the color code can be changed.
- With the buttons in the 'Labels' section the atomic symbols can be switched on and off.

- Each atom type has a radius given in the STR file, cf. Sec. C.6.1. This determines the radius of the corresponding spheres. If it is desired to represent all atoms by spheres of identical radius (0.5), switch on the button unique in the 'Radius' section.

- With the selector 'Scale' the multiplier of the sphere radius is set.

- The 'Operate' buttons affect the selected atoms. With show and hide the correspond atom can be made visible (or hide). set_radius changes the radius given in the STR file to new value, typed below in the 'New_radius' box.

- The bottom part of the toolbox is used to create a new atom. The chosen symbol and coordinates are given in the corresponding boxes. After pressing the Create button the new atom appears.

A.2.2 Bonds tab

The buttons for the Bonds tab are shown in Fig. A.4.

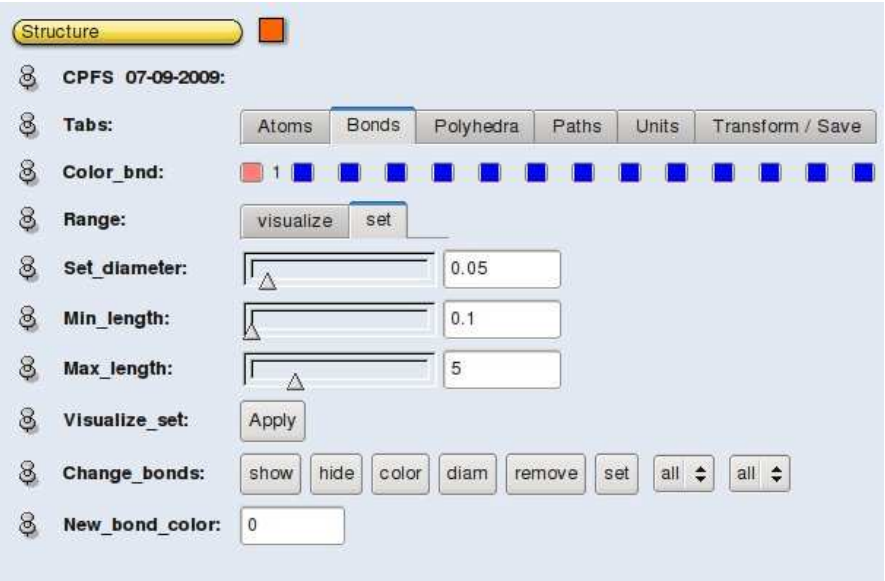

Fig. A.4 Avizo STR module – Bonds tab

- Different colors can be assigned to the bond cylinders ('Color_bnd'). With a click on the colored square next to the color number the color code can be changed.

- With the tabs visualize and set the bonds can be manipulated.

- The set tab shows the selectors for the bond radius as well as the minimal and maximal bond length region that will be visualized. Bond with lengths outside this region will be hidden (but created in the memory).

- The manipulation of the bonds are performed with the buttons in the 'Change_bonds' section. The manipulations affect only the chosen pair of atom types. The effect of show and hide buttons is the same as within the 'Atoms' tab. With the color button new color number, typed in the 'New_bon_color' box below.

- The 'Operate' buttons affect the selected atoms. With show and hide the correspond atom can be made visible (or hide). set_radius changes the radius given in the STR file to new value, typed below in the 'New_radius' box. The button diam assigns the diameter given in the 'Set_diameter' box to the selected atom types pairs (and restricted to the chosen bond length region). The buttons remove and set either removes or creates bonds between the selected atom types restricted to chosen bond length region.

- With the visualize tab the overall scaling for bonds is set.

A.2.3 Polyhedra tab

The buttons for the Polyhedra tab are shown in Fig. A.5.

- The Apply button in the 'Materials' section sets the parameters according to the 3 selectors below. The parameters ('Specular', 'Shininess', and 'Transparency') applies to the polyhedron and face selected in the 'Set_polyhedra' section (respectively to all faces of a polyhedron).

- To create new polyhedron select 'new' in the first selector of the 'Set_polyhedra' section and type the sequence of symbols (accessible via the atomic labels, cf. Sec. C.6.1) creating the first face into the 'Vertices' box at the lowest part of the toolbox. In Fig. A.5 the molecular ring of the oxazole is chosen. Clicking on the Apply button of the 'Create_face' section creates the ring pentagon (blue colored).

- In the 'Set_polyhedra' section the polygon 'P_1' can be selected. The polygon 'P_1' is formed by single face 'F_1' (second selector, cf. Fig. A.5). Creating an-

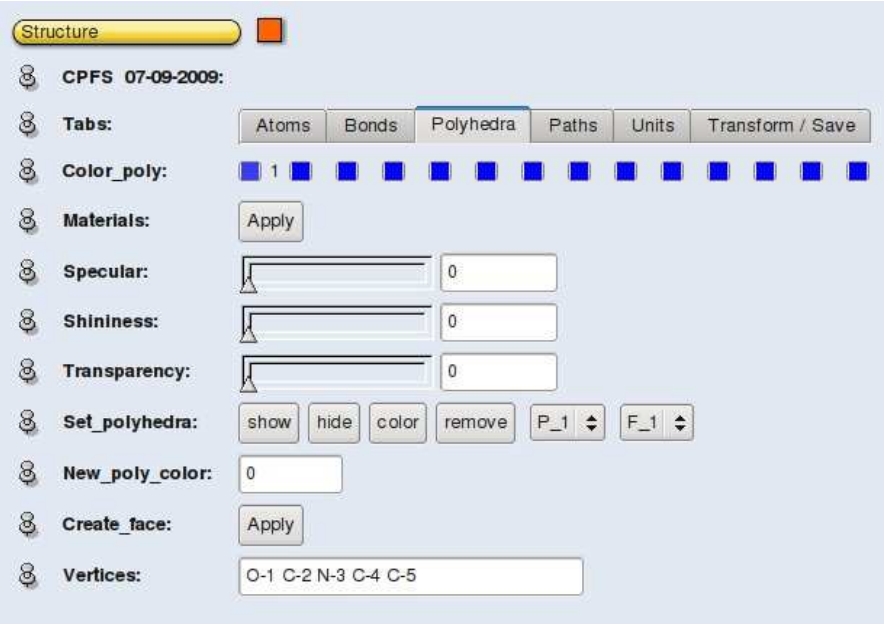

Fig. A.5 Avizo STR module – Polyhedra tab

other face with 'P_1' selected would add the new face (marked 'F_2) to the polyhedron 'P_1. The faces on a polyhedron need not to really form a polyhedron. It is just a convenient way to attribute the same parameters (color, transparency, etc.) to a set of faces. To create a face belonging to new polygon 'new' must be selected.

- The effect of the buttons show and hide in the 'Set_polyhedra' section is self explaining. With the button color a color number can be assigned to the polyhedron, see the color buttons in the first section 'Color_poly'.

A.2.4 Paths tab

To visualize the interconnection paths first the file with the data for the critical points and the path points, written in the STR format, must be read in. Fig. A.6 shows that in this case the STR format must be recognized by the interface. After the file is read the structure module can be started the same way as in case of the molecular structure data, cf. page 94. In contrast to the atomic data, now the Atoms tab does not contain atomic symbols but the symbols for the critical points (and cores), i.e.,

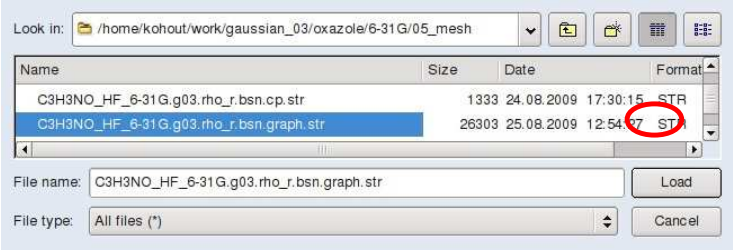

Fig. A.6 Avizo selector box for STR files. The format of the STR files must be recognized

Attr, *Bcp*, *Rcp*, *Cage*, respectively *Core*. Of course, if desired the critical points can be connected by 'bonds' and the critical points can be used as vertices of polyhedra. The buttons for the Paths tab are shown in Fig. A.7.

- The 'Color_path' section 3 colors are defined (to change by click on the colored square). First color is for ICLs from saddle points (*Bcp*) to the attractors (*Attr*) – termed in the toolbox 'Bond' paths, the second one for the ICLs from ring critical points (*Rcp*) to the minima (*Cage*) – termed here 'Ring' paths. The third color is reserved for field trajectories starting from arbitrary points.

- Each path has unique label, cf. Sec. C.6.5), which can be identified by switching on the button in the 'Path_labels' section.

- With the 'Scale_p_diameter' selector the diameter of the selected paths can be scaled.

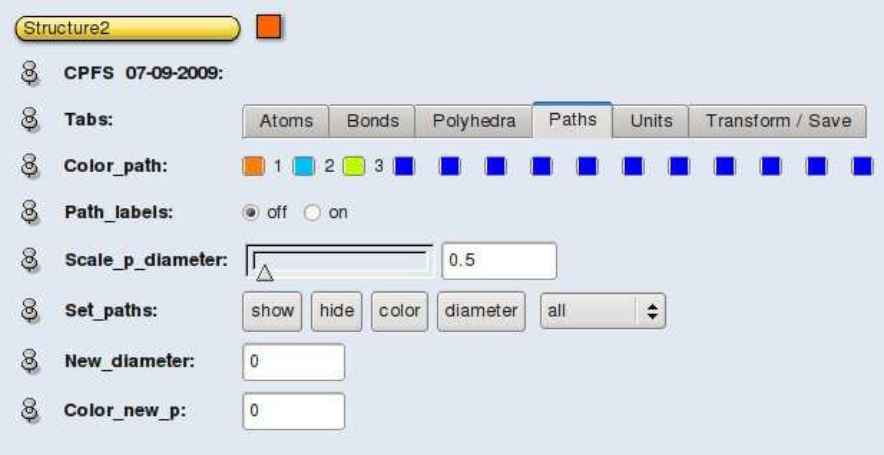

Fig. A.7 Avizo STR module – Paths tab

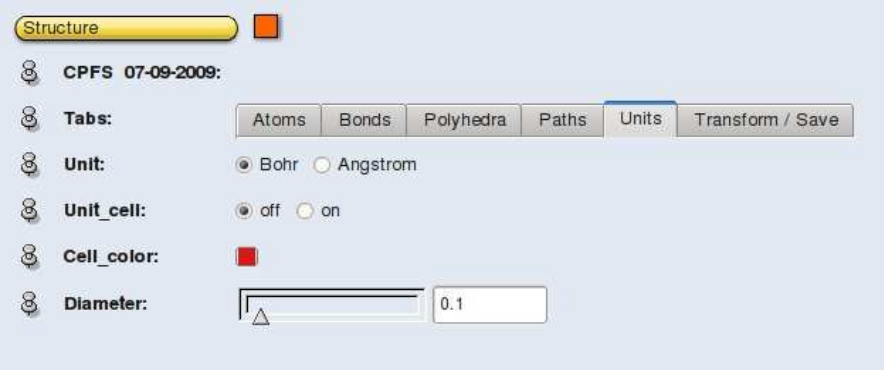

Fig. A.8 Avizo STR module – Units tab

- With the color and diameter buttons the color and diameter of the individually selected ICL segment can be set (first type the desired values in the 2 boxes below.

A.2.5 Units tab

The buttons for the Units tab are shown in Fig. A.8.

- If the structure data file was created by an external program or set up manually then the units used there could differ from the one used for the grid generation (for instance, from a DGrid calculation the grid dimensions are in bohr, whereas the separate STR file with atomic coordinates in Ångstrom). In the 'Unit' section the atomic coordinates can be adjusted.

- With the next 3 sections the unit cell can be visualized. The unit cell data are taken from the 'Lattice vectors' part of the *property* file, cf, Sec. 1.7. The color and diameter of the cell cylinders can be set.

A.2.6 Transform/Save tab

The buttons for the Transform/Save tab are shown in Fig. A.9.

- With the transform part the atomic coordinates can be described by new Cartesian coordinates. This is useful if the atomic coordinates comes from external source with the axes origin at some distant place. For the definition of the new coordinates 3 positions must be given:

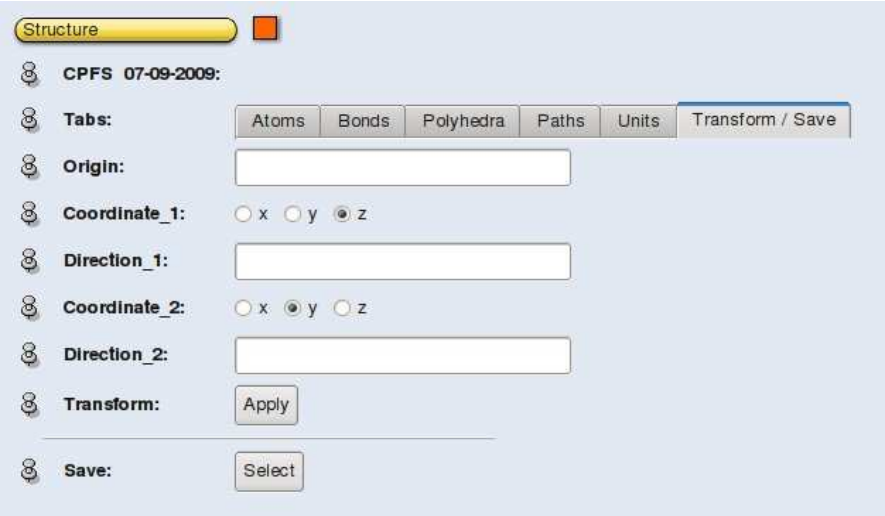

Fig. A.9 Avizo STR module – Transform/Save tab

- In the 'Origin' section input 3 coordinates for the new origin.
- In the 'Coordinate_1' section choose the new axis you would like to define.
- In the 'Direction_1' section input 3 coordinates for the direction to which the axis points (from the new origin).
- In the 'Coordinate_2' section choose the second axis you would like to define.
- In the 'Direction_2' section input 3 coordinates for point within the plane spanned by the two axes. The routine computes the new axis to be within this plane perpendicular to the first axis.
- The third axis will be determined by the routine (starting from the origin normal to the plane spanned by the first 2 coordinates.

- All atomic coordinates will be transformed to the new axes system.

- To save the structure expressed in the new coordinate system click on the select button in the 'Save' section. File selector box opens where the file name can be chosen.

A.2.7 Saving networks in Avizo

The visualization with Avizo can be very complex with several grid files interconnected and few structures generated. When Avizo is closed the whole arrangement, called *network*, is lost unless saved. The saving of a *network* is performed with File → Save Network As...

In case of the structure module this will save only the information about the module itself, but not the actual (changed) structure data setup which are part of the grid information (i.e., within the green grid boxes). To save the changed structure setup (colors, bonds, paths, etc.) the grid files must be saved as well (which will differ from the grid data originally read into Avizo). For this choose 'Avizo Script and data file (pack & go)' for the file format. The changed grid files will be written to a separate directory. Thus, the original grid files will be not changed.

A.3 Basin visualization with Avizo

The *basin* file contains a field of integer values which can be read into Avizo with the interface as described for the property grids on page 94. Special route must be followed for the visualization of the basins as separate solid objects.

- After the *basin* file is read rightclick on the green box representing the *basin* grid and select SurfaceGen from the menu. Red box named 'SurfaceGen' appears in the 'Pool' window.

- Click on Apply in the 'Properties' window of the 'SurfaceGen' box below the 'Pool' window. New green box appears in the 'Pool' window.

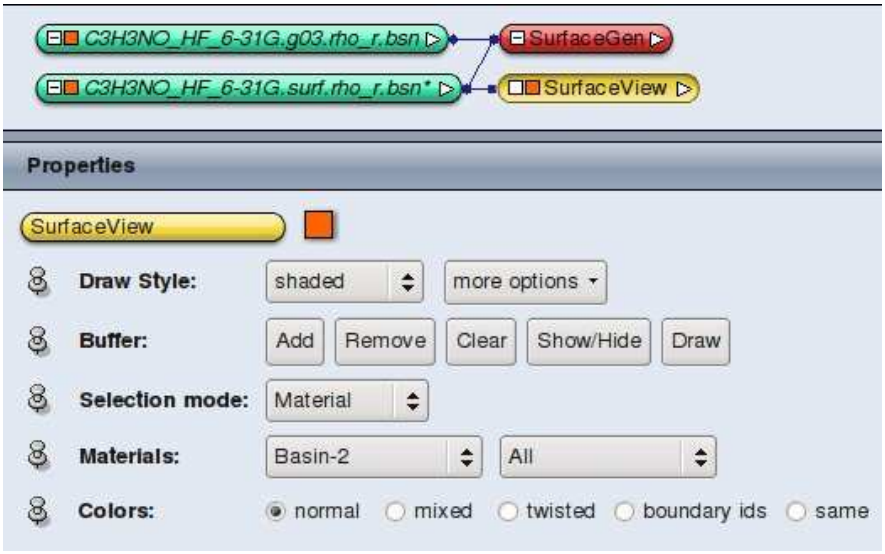

Fig. A.10 Avizo basin module

- Rightclick on the new green box and select SurfaceView from the menu. Yellow 'SurfaceView' box appears in the 'Pool' window.

- In the 'Properties' window corresponding to the 'SurfaceView' box the basins can be chosen in the section 'Materials', cf. Fig. A.10.

The basin visualization in Avizo is based on so called label field, which restricts the number of basins to 256. For molecules this usually entails no difficulties.

A.4 Color map

For convenience the 'standard' ELF colormap (slightly modified version of colormap originally proposed by A. Savin) is included in the *avizo* directory of the *dgrid-4.5* package. The corresponding file *elfmap_mk.icol* must be copied to the location *data/colormaps* of your Avizo installation. The colormap can be activated with the Load colormap... button in the 'Properties' window of Avizo.

Appendix B
Visualization with OpenDX

Alexey I. Baranov

The visualization of DGrid generated data can be performed with the freeware OpenDX program. Two main parts are responsible for this:

- an OpenDX module, providing routines which read DGrid file formats and extend its built-in modules. The module itself is a single binary file.

- an OpenDX visual program providing simple GUI for end user. The visual program itself is a text file which is read by OpenDX to create the necessary GUI and do the proper visualization.

The OpenDX module installation is described in Sec. B.1. Very short introduction into OpenDX controls is done in Sec. B.2. The usage of the visual program is described in Sec. B.3.

Detailed information about the visualization package OpenDX can be found in the Internet at http://www.opendx.org. In many Linux distributions (e.g., openSUSE) it is available via the software repositories (usually called opendx or dx) so one may easily install it using the package manager.

B.1 OpenDX module installation

An OpenDX module is made available as separate part of the DGrid package. There are 2 ways to deploy the module:

- Precompiled modules are available for x86-64 Linux or x86-32 Microsoft Windows versions of OpenDX. They were tested on x86-64 Linux openSUSE 11.1 and on x86-32 Microsoft Windows Vista Premium. These binary modules are not included in the DGrid package. Send a message to baranov@cpfs.mpg.de to request the precompiled modules.

- The module can be built from the source code provided with the DGrid. The so called ‘devel’ version of OpenDX (with header files and libraries included) needs to be installed additionally to create the module from the source. The next Subsec. B.1.1 describes how to built the module on the Linux machine.

B.1.1 Compilation of OpenDX module

Be sure that the ‘devel’ version of OpenDX is installed so that the OpenDX header files and libraries are available. Go to the folder:

```
dgrid-4.5/DXDGrid
```

and modify the *makefile* to specify the path to the OpenDX folder (BASE=). One might need to modify as well the target platform when it is not Linux and the C/C++ compiler used when it is not GNU.
The module is built with the command:

```
make
```

The module called *dxdgrid* should appear in the folder.

B.1.2 Configuration of the startup script

An OpenDX start script must be configured before starting OpenDX. This script called *startdxdgrid* sets up all necessary command line options for OpenDX so that the DGrid visual program can be executed.

The following files are used in the visualization of DGrid data with OpenDX:

- *startdxdgrid* (Linux) or *startdxdgrid.bat* (Windows) — launch script
- *dxdgrid* (Linux) or *dxdgrid.dll* (Windows) — module binary code
- *dxdgrid.mdf* — module description file
- *dxdgrid.net* — visual program file
- *dxdgrid.cfg* — visual program configuration file
- *ELF_mk.cm* — classical ELF colormap

They all can be found in *dgrid-4.5/DXDGrid* folder.

Following locations must be set up in *startdxdgrid* script:

- OpenDX installation path (BASE=)
- DGrid module file *dxdgrid* path (DXMODULES=)
- Module description file *dgrid.mdf* path (MDF_PATH=)
- Visual program file *dgrid.net* path (NET_PATH=)

It is recommended to add the location of the start script to the user PATH environment variable.

Now OpenDX is ready for the DGrid format. Start OpenDX by typing the name of the start script (possibly including the full path):

```
startdxdgrid
```

To test it prepare some property grid (for instance the electron density grid for oxazole given in the manuscript, cf. Sec. 1.6. page 7).

B.2 OpenDX Controls

The DGrid-specific controls of visualization program are implemented as so called control panels. There are four control panels – two for structure-like data (Structure-I and Structure-II) i.e. molecular structures or interconnection paths and two for scalar property data, e.g, charge density or ELI-D basins (Property-I and Property-II). They can be opened from the main menu of Image Window using (cf. Fig. B.1): Windows → Open Control Panel by Name

General controls of OpenDX are accessible through other menu options (cf. Fig. B.1). Most important among them are:

- File → Save Image or File → Print Image — to save the image.
- Execute → Execute on Change — to activate on-the-fly visualization
- Execute → Execute Once — to update the picture manually when on-the-fly visualization is too slow.

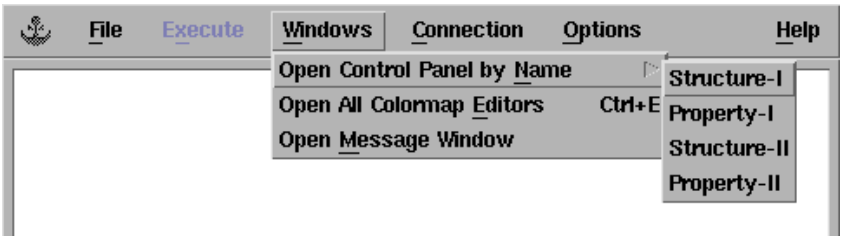

Fig. B.1 Accessing Control Panels in OpenDX Image Window

- Execute → End Execution — to abort the visualization when it takes too much time.
- Windows → Open All Colormap Editors – to open colormap editor used for coloring of the slices.
- Windows → Open Message Window – to open diagnostic message window.
- Options → View Control — to select some visualization options like the use of perspective.
- Options → Mode — to select mouse behavior like rotate or zoom.
- Options → Reset — to reset the view.
- Options → Set Background Color – self explanatory.
- Options → Rendering options – to select rendering options. Whenever possible, choose the hardware rendering.

Detailed information about all these menu items can be found in *Chapter 8. Graphical User Interface: Menus, Options, and the Message Window* of OpenDX User's Guide.

B.3 Visualization

1. Start OpenDX by typing the name of the start script *startdxdgrid*.

2. In the Image Window activate on-the-fly visualization mode by clicking Execute → Execute on Change (cf. Sec. B.2)

   If on-the-fly visualization is too slow due to heavy data set, do not select this menu. Instead, each time the picture is modified and need to be redrawn, select Execute → Execute Once or press Ctrl-O key combination to update the drawing.

B.3.1 Visualizing molecular structures and graphs

B.3.1.1 Load

1. Open the Structure-I control panel. (cf. Sec. B.2)

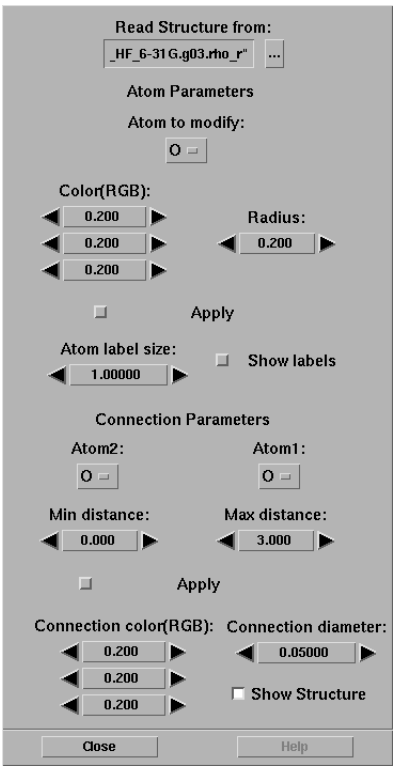

Fig. B.2 Structure Control Panel

2. Press the ... button to open a file selector box (cf. Fig. B.2).

3. Select the grid or STR file to be loaded (e.g., *C3H3NO_HF_6-31G.g03.rho_r*).

   If on-the-fly visualization was enabled, remark that Execute menu item lights green for certain time indicating the OpenDX is processing the data.

4. To show the structure on the screen activate Show Structure checkbox in the Structure-I menu (cf. Fig. B.2). Select Execute → Execute Once or press Ctrl-O key combination to update the drawing, if on-the-fly visualization was disabled. The structure will appear on the screen as a ball-and-stick model (cf. Fig. B.3).

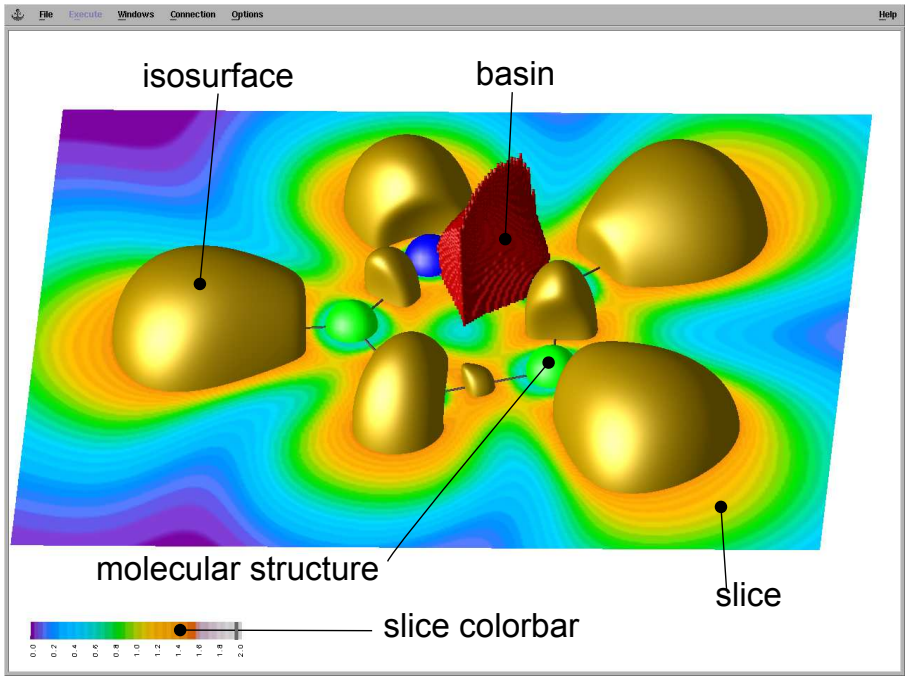

Fig. B.3 OpenDX Image Window showing molecular structure, slice, isosurface and basin.

B.3.1.2 Navigate

- To **rotate** the structure, go to Options → Mode menu of the Image Window (cf. Sec. B.2) and select Rotate item. Alternatively, Ctrl-R key combination can be pressed. Then the structure can be rotated with the left or right mouse button pressed.

- To select the area to be **zoomed** go to Options → Mode and select Pan/Zoom item. Alternatively Ctrl-G (under Windows: Ctrl-Space) combination can be pressed. Then the structure can be zoomed/unzoomed by selecting the proper area with the left or right mouse button pressed.

B.3.1.3 Modify

- To modify the **atom** properties, select it from the dropdown list Atom to modify (cf. Fig. B.2), set up new color and radius and press Apply. Atom labels can

also be shown to identify the atoms. To do this, activate Show Labels checkbox and adjust label size. The label size is the same for all the atoms.

Remark B.1. Please pay attention that both the color (color coordinates must be in the range from -0.999999 to 0.999999) and radius of the atom will be set to the values from that control panel even if they were not touched.

- To modify the **connections** drawn, select the connection partner atoms from the dropdown lists Atom1 and Atom2 (cf. Fig. B.2), set up allowed distance range and press Apply.

  The color and diameter of connections can be modified as well. To modify them, just set up new values. The color and diameter are the same for all the connections.

B.3.1.4 Some remarks to the visualization of critical points and interconnection paths

To visualize the interconnection paths first the file with the data for the critical points and the path points, written in the STR format, must be read in. After the file is read the graph can be visualized the same way as in case of the molecular structure data, cf. page 108. In contrast to the atomic data, now the Atom to modify list does not contain atomic symbols but the symbols for the critical points (and cores), i.e., *Attr*, *Bcp*, *Rcp*, *Cage*, respectively *Core*.

- For the molecular graphs the CPs are treated like atoms. One may modify their color and size and plot the labels. For different CPs of the same type (i.e. attractors Attr_1 and Attr_2) it is not possible to set different colors or radii, these parameters are always the same for all the CPs of certain type.

- One may control the visualization of interconnection lines between critical points via Connection Parameters controls. To hide certain interconnection line, set maximal distance to zero and press Apply. To visualize certain interconnection line, select the line endpoint CPs and set minimal distance to zero and maximal distance to any nonzero number and press Apply. By default all the interconnection lines are visualized. It is not possible to set up a straight connection line between tho CPs, they can have a connection only if the interconnection path was found by DGrid and written into STR file.
  The color and diameter are the same for all the interconnection lines.

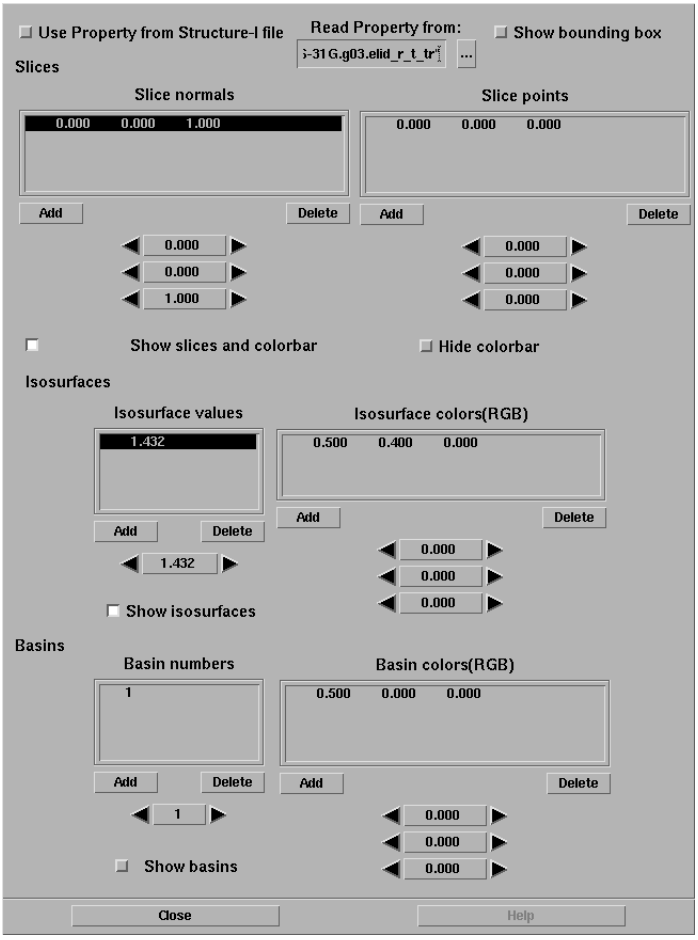

Fig. B.4 Property Control Panel

B.3.1.5 Loading second structure file

The Structure-II control panel (cf. Sec. B.2) allows to load a second structure with either molecular structure or CPs and interconnection paths (e.g., *C3H3NO_HF_6-31G.g03.rho_r.bsn.graph.str*) in addition to the structure loaded with the Structure-I control panel. This can be useful for the comparison between two diagrams or for plotting CP graph and molecular structure in the same time. The Structure-II control panel has the same controls as Structure-I.

B.3.2 Visualizing scalar properties

1. Open the Property-I control panel (cf. Sec. B.2).

2. If grid file (e.g., *C3H3NO_HF_6-31G.g03.rho_r*) is already loaded with Structure-I control page, one can activate the checkbox Use Property from Structure-I to visualize the charge density field from this grid file (cf. Fig. B.4).
   Alternatively, the ... button of the Read Property from field may be pressed and the desired grid file selected.

3. - To visualize the **slice** of the scalar property, specify at least one non-zero normal to the slice plane in the Slice normals (cf. Fig. B.4). To modify the normal, select it from the list and set new normal components using the number controls below the list. In the same way one may modify the coordinates of the point on the slice plane in the right Slice points list. Finally activate the Show slices and colorbar checkbox to show slices on the screen (cf. Fig. B.3). Arbitrary many slices can be plot at the same time using Add button.

     To select another colormap or modify current colormap used by slices open the menu Window → Open All Colormap Editors (cf. Sec. B.2). Detailed information about Colormap Editor can be found in Subsection 8.1 *Colormap Menu Bar* of OpenDX User's Guide.

     Remark B.2. The 'holes' that can appear on the slice marks the regions where the scalar property value is outside of the colormap value range.

   - To plot the **isosurface** of the scalar property, specify at least one isosurface value in the Isosurface values list (cf. Fig. B.4). To modify the isosurface value, select it from the list and set new value using the control below the list. In the right window one specifies the isosurface colors. To modify the color select it with the mouse in the Isosurface colors list and then modify the color coordinates. When no color is specified for certain isosurface, the default color will be used. Finally activate the Show isosurfaces checkbox to show isosurfaces on the screen (cf. Fig. B.3). Arbitrary many isosurfaces can be plot at the same time using Add button.

   - If the *basin* file containing a field of integer values showing basin number of each grid point is read, the **basins** can be visualized as separate solid objects. To visualize the **basins**, specify at least one basin number in the Basin numbers list (cf. Fig. B.4). To modify the basin number, select it from the list and set new number using the control below the list. In the right window one specifies the basin colors. To modify the color select it with

the mouse in the Basin colors list and then modify the color coordinates. When no color is specified for certain basin, the default color will be used. Finally activate the Show basins checkbox to show basins on the screen (cf. Fig. B.3). Arbitrary many basins can be plot at the same time using Add button.

B.3.2.1 Loading second property file

The Property-II control panel (cf. Sec. B.2) allows to load a second scalar property (e.g., *C3H3NO_HF_6-31G.g03.elid_r_t_tr*) in addition to the property loaded with the Property-I control panel. The Property-II control panel has the same controls as Property-I.

B.3.2.2 Some remarks to the properties plots

- All the color coordinates must be in the range from -0.999999 to 0.999999
- The Show bounding box can be activated to show the bounding box of the region inside which the scalar property was calculated (cf. Fig. B.4).

  Remark B.3. This box is correct only for orthogonal grids. For non-orthogonal grids it shows orthogonal region which is larger than real region of the property calculated.
- Any added slice, isosurface or the basin can be deleted with the appropriate Delete button (cf. Fig. B.4).
- Colorbar from the Slice plot can be switched off with the Hide colorbar button (cf. Fig. B.4).

When closing OpenDX, **DO NOT** save the visualization program. Doing this may affect the control panels. The visual program and its configuration file are made write-protected to avoid saving.

Appendix C
Comments to property calculations

C.1 Grid mesh

For the visualization (with an external program) a distance of 0.1 bohr between the grid points (0.1 bohr mesh size) is sufficient. However, only for light elements (Li – F) the precision of numerical charge integration is satisfactory for such mesh. For a steep property with lot of critical points (like ELI-D or density Laplacian) a 0.05 bohr mesh (or better) is recommended. For heavier elements (metals) even much more dense mesh would be necessary, however, creating huge grid files. In this case, use a 0.05 bohr mesh with additional refinement performed with the `refine` utility, cf. section 3.5. This approach is possible only if the *basis* file is present (i.e., not possible yet for solid state calculations). For Gauss-type basis set analytical charge integration can be used, cf. Sec. 2.7.

C.2 Orbitals

The orbital amplitude is computed with the assignment **compute**=phi <spin>. Only 1 orbital (keyword **occupation**) and particular spin channel must be chosen. In momentum space (keyword **space**=momentum) the orbitals have in general a real and an imaginary part. The respective part can be chosen with the assignment **wf_part**=real, respectively **wf_part**=imaginary. The default value (i.e., without the appropriate choice) is `real`. This differs from the calculation of orbital density, where the default value for the orbital part is to use both, real and imaginary part.

C.3 Spin density

For the spin density, computed with the assignment **compute**=rho-spin, cf. Table 2 on page 13, both spin parts are needed. Thus, the spin density will be not computed if a particular spin part is chosen.

C.4 ELI

In case of ELI-D, computed with the assignment **compute**=eli-d <pair>, three results are possible, depending on the choice of the pair-spin component. Choosing the keyword 'alpha-alpha' respectively 'beta-beta' for the pair-spin yields the ELI-D for the corresponding same-spin electron pairs [19]. In each respective cases the electron density of corresponding spin will be sampled.

ELI-D for triplet coupled electron pairs is computed with the keyword 'triplet-pair' for the pair-spin. In this case the charge of triplet-coupled electrons is sampled. Note that the value of ELI-D for triplet pairs is influenced by the total number of electrons of the system, see Ref. [23].

In ELI-D for triplet-coupled electrons both spin parts are included. The formula used in DGrid is derived for restricted basis only. Bear in mind, that for unrestricted basis (like for spin-polarized system computed with ADF), the formula is not the correct one (as in this case the expectation value of S^2 yields an improper value, contaminated by higher multiplets). In case of unrestricted basis ELI-D for triplet-coupled electrons will be evaluated, *however*, the density and the pair-volume function of triplet-coupled electrons is computed according to Eqs. 5 and 37 of Ref. [23], which is just a way to show an ELI-D value.

The other variant of ELI, namely ELI-q (cf. Ref. [23]), is derived from the space partitioning based on fixed charge. If both spins should be included it must be specified that the partitioning is based on the fixed charge of singlet-coupled electrons, i.e., with the command '**compute**=ELI-q singlet', in which case the singlet-coupled electron pairs are sampled. It is still possible to compute the variant ELIA, which is derived from space partitioning based on fixed product of α-spin and β-spin charge, cf. Ref. [20].

C.5 ELF, LOL

The assignments '**compute**=ELF alpha', respectively '**compute**=ELF beta', result in the calculation of ELF according the formula of Becke and Edgecombe [9]. If both spin channels are chosen (i.e., '**compute**=ELF'), then ELF for the total density will be computed according the 'spin-polarized' formula, cf. Table 2 as well as Eq. 9 in Ref. [21]. The assignment '**compute**=ELF-cs' yields for the total density the ELF computed according the 'closed-shell' formula, cf. Table 2 as well as Eq. 7 in Ref.

[21]. The ‘spin-polarized’ formulation follows the idea that ELF is based on kinetic energy densities, whereas the ‘closed-shell’ formula is more related to the electron pair densities and can be, in certain sense, rationalized from ELI for triplet-coupled electron pairs [23].

With the assignment ‘**compute**=ELF-Kirshnitz’ (with or without the spin descriptor) the ELF variant of Tsirelson [31] is computed. It is based on kinetic energy density approximated by the Kirshnitz formula.

In case of LOL, computed with the assignment ‘**compute**=LOL’, single spin channel should be chosen to comply with the definition of Schmider and Becke [28].

C.6 STR format

The STR format is used to support structure information for visualization. The data are written in sections into a file. The data start with the sections, termed ‘UNITS’, ‘GRID_DESCRIPTION’, and ‘SCALE’:

```
UNITS
BOHR             1

GRID_DESCRIPTION
FILENAME         C3H3NO_HF_6-31G.g03.elid_r_t_tr
ELI_CORE         1

SCALE
ATOMIC_RADIUS    1.0
BOND_DIAMETER    0.5
BOND_MIN         0.1
BOND_MAX         5.0
PATH_DIAMETER    0.5
```

The ‘GRID_DESCRIPTION’ section is used for the ICL graphs. The data can be evaluated by chosen visualization tool (and are actually used by the module written for the programs Avizo and OpenDX). This header is followed by one or more sections called:

- ATOMS ↦ spheres
- CONNECTIONS ↦ cylinders
- UNIT_CELL ↦ unit cell
- POLYHEDRA ↦ polyhedra and polygons
- PATHS ↦ cylinders connected to form a path

Each section is followed by a separate part defining the colors used in the corresponding section (always using the same scheme, e.g., the 'ATOMS' section is followed by the 'ATOM_BUTTONS' part, the 'CONNECTIONS' section is followed by the 'CONNECTION_BUTTONS' part, etc.).

The last keyword in the file must be the string 'END'.

C.6.1 ATOMS

In this section the data for spheres representing atoms, maxima, saddle points, etc. are given. Following is an example for the C_3H_3NO molecule, written out by the Avizo visualization program:

```
ATOMS
:atom      radius      x         y         z     vis
:-----------------------------------------------------
 O_1        0.551   -2.0618    0.5748    0.0000   1
 C_2        0.451    0.0000    2.1211   -0.0000   1
 N_3        0.503    2.0869    0.8955   -0.0000   1
 C_4        0.451    1.4135   -1.6748   -0.0000   1
 C_5        0.451   -1.1101   -1.8596    0.0000   1
 H_6        0.095   -0.3077    4.1025   -0.0000   1
 H_7        0.095    2.8049   -3.1219   -0.0000   1
 H_8        0.095   -2.4310   -3.3680    0.0000   1

ATOM_BUTTONS
:atom      r       g       b
:-----------------------------
 O      0.2300  0.2300  0.9700
 C      0.3300  0.3300  0.3300
 N      0.0000  0.8400  0.1700
 H      0.6700  0.6700  0.6700
```

The (atomic) symbol is a 5 character string possibly followed by '_<num>' digit differentiating between atoms with identical symbols. It is followed by the radius and the Cartesian position of the sphere. The last number shows the visibility (0 - not rendered by the visualization tool; 1 - visible). The 'ATOM_BUTTONS' section gives the RGB colors for each atomic symbol. If the 'ATOM_BUTTONS' part is not present, which is the case if the structure file is created with the command '`dgrid` *basisfile* `str`' (cf. Sec. 3.3) then the visualization tool must choose its own default values.

C.6.2 CONNECTIONS

In this section the data for the cylinders (usually representing the bonds) are given:

```
CONNECTIONS
:connection diameter    x1        y1       z1        x2       y2       z2    vis c
:-------------------------------------------------------------------------------------
 O-C_1-2      0.100  -2.0618    0.5748   0.0000    0.0000   2.1211  -0.0000   1  1
 O-C_1-5      0.100  -2.0618    0.5748   0.0000   -1.1101  -1.8596   0.0000   1  1
 C-N_2-3      0.100   0.0000    2.1211  -0.0000    2.0869   0.8955  -0.0000   1  1
 C-H_2-6      0.100   0.0000    2.1211  -0.0000   -0.3077   4.1025  -0.0000   1  1
 N-C_3-4      0.100   2.0869    0.8955  -0.0000    1.4135  -1.6748  -0.0000   1  1
 C-C_4-5      0.100   1.4135   -1.6748  -0.0000   -1.1101  -1.8596   0.0000   1  1
 C-H_4-7      0.100   1.4135   -1.6748  -0.0000    2.8049  -3.1219  -0.0000   1  1
 C-H_5-8      0.100  -1.1101   -1.8596   0.0000   -2.4310  -3.3680   0.0000   1  1

CONNECTION_BUTTONS
:col      r       g       b
:----------------------------
  1:   1.0000  0.4900  0.4900
```

The connection symbol shows which atoms are connected (can be also some chosen label). It is followed by the diameter as well as the positions of the start and end points of the cylinder (bond). The last numbers show the visibility (0 - not visible in the visualization tool; 1 - visible) and color number. The 'CONNECTION_BUTTONS' part gives the corresponding the RGB colors.

C.6.3 UNIT_CELL

In this section the data for the unit cell are given (from which the unit cell can be constructed):

```
UNIT_CELL
:   diameter     x1        y1        z1         x2        y2        z2    col
:-------------------------------------------------------------------------------
A:   0.100     0.0000    0.0000    0.0000     1.0000    0.0000    0.0000   1
B:   0.100     0.0000    0.0000    0.0000     0.0000    1.0000    0.0000   1
C:   0.100     0.0000    0.0000    0.0000     0.0000    0.0000    1.0000   1

UNIT_CELL_BUTTONS
:col      r       g       b
:----------------------------
  1:   0.8600  0.0900  0.0900
```

For the three vectors A, B, C the diameter as well as the positions of the start and end points of the cylinders are given. The last number shows the color of the cylinder, given by the RGB values in the 'UNIT_CELL_BUTTONS' section.

C.6.4 POLYHEDRA

In this section the data for polyhedra build up from polygons (faces) are given:

```
POLYHEDRA
:           faces     vis col     spec      shin     transp
:-------------------------------------------------------------
Polyhedron    1        1   1     0.0000    0.0000    0.6000
```

```
   FACE vertices:   5
      -2.0618    0.5748    0.0000
       0.0000    2.1211   -0.0000
       2.0869    0.8955   -0.0000
       1.4135   -1.6748   -0.0000
      -1.1101   -1.8596    0.0000

POLYHEDRA_BUTTONS
:col      r       g       b
:-----------------------------
  1:   0.9220  0.8980  0.4750
```

For each polyhedron the number of faces is given, followed by the visibility and color number (see the 'POLYHEDRA_BUTTONS' section) together with the data concerning the shininess and transparency of the faces. Each face has its own subsection giving the number of vertices and the corresponding Cartesian coordinates (the face in the example is a pentagon formed by the ring atoms of the oxazole molecule).

C.6.5 PATHS

In this section the data for the ICL trajectories running from the saddle point to the corresponding attractor (respective from ring critical points to the minima) are given (cf. Sec. 2.5):

```
PATHS
:        name     points   vis col  diameter
:-----------------------------------------
Path     B1-A2      36      1   1      0.02

:         x         y         z
:---------------------------------
        0.7730  1.624593  0.000000
        0.7908  1.615388  0.000000
                ...
        2.0833  0.897110  0.000000

:        name     points   vis col  diameter
:-----------------------------------------
Path     B1-A4      21      1   1      0.02

:         x         y         z
:---------------------------------
        0.7730  1.624593  0.000000
        0.7553  1.633799  0.000000
                ...
        0.0045  2.117934  0.000000
```

```
PATH_BUTTONS
:col      r       g       b
:---------------------------
  1:   1.0000  0.5000  0.0000
  2:   0.0000  0.7500  1.0000
  3:   0.7500  1.0000  0.0000
```

For each path a separate subsection is written. It contains the name of the path and the number of points, which connected by cylinders form the path. The label ‘B1-A2’ stands for the path from the saddle point number 1 (which is between the nitrogen and carbon C2 ρ-basins, cf. the table on page 45) to the attractor number 2 (the nitrogen, cf. the table on page 44). If ICL graph for the ring points is present, then the path descriptors start with ‘R’ instead of ‘B’. Additionally, color number 2 is chosen for this paths.

This is followed by the visibility, color, and diameter of the path. Then the Cartesian coordinates of each point of the path are given, 1 per line. The ‘PATH_BUTTONS’ section gives the RGB colors of the paths.

C.6.6 END

The STR format is terminated by the string ‘END’.

And this is also the end of the User’s Guide

References

1. ADF (density functional program). Baerends EJ, Ellis DE, Ros P (1973) Chem Phys 2: 41; Versluis L, Ziegler T (1988) J Chem Phys 88: 322; te Velde G, Baerends EJ (1992) J Comput Phys 99: 84; Fonseca Guerre C, Snijders JG, G. Velde G, Baerends EJ (1998) Theor Chem Acc 99: 391
2. Cube format. http://astronomy.swin.edu.au/∼pbourke/geomformats/cube/
3. Grace. http://plasma-gate.weizmann.ac.il/Grace/
4. Andersson, K., Barysz, M., Bernhardsson, A., Blomberg, M.R.A., Cooper, D.L., Fülscher, M.P., de Graaf, C., Hess, B.A., Karlström, G., Lindh, R., Malmqvist, P.A., Nakajima, T., Neogrády, P., Olsen, J., Roos, B.O., Schimmelpfennig, B., Schütz, M., Seijo, L., Serrano-Andr'es, L., Siegbahn, P.E.M., Stålring, J., Thorsteinsson, T., Veryazov, V., Widmark, P.O.: MOLCAS. Lund University, Sweden
5. Bader, R.F.W.: Atoms in molecules. Oxford University Press, Oxford (1990)
6. Bader, R.F.W., Gatti, C.: A Green's function for the density. Chem. Phys. Lett. **287**, 233 (1998)
7. Bader, R.F.W., Stephens, M.E.: Fluctuation and correlation of electrons in molecular systems. Chem. Phys. Lett. **25**, 445–449 (1974)
8. Bader, R.F.W., Stephens, M.E.: Spatial localization of the electronic pair and number distributions in molecules. J. A. Chem. Soc. **97**, 7391–7399 (1975)
9. Becke, A.D., Edgecombe, K.E.: A simple measure of electron localization in atomic and molecular systems. J. Chem. Phys. **92**, 5397 (1990)
10. Clementi, E., Roetti, C.: At. Data. Nucl. Data Tables **14**, 218 (1974)
11. Cremer, D., Kraka, E.: A description of the chemical bond in terms of local properties of electron density and energy. Croat. Chem. Acta **57**, 1259–1281 (1984)
12. Fradera, X., Poater, J., Simon, S., Duran, M., Solà, M.: Electron-pairing analysis from localization and delocalization indices in the framework of the atom-in-molecules theory. Theor. Chem. Acc. **108**, 214–224 (2002)
13. Frisch, M.J., Trucks, G.W., Schlegel, H.B., Scuseria, G.E., Robb, M.A., Cheeseman, J.R., Zakrzewski, V.G., Montgomery, J.A.J., Stratmann, R.E., Burant, J.C., Dapprich, S., Millam, J.M., Daniels, A.D., Kudin, K.N., Strain, M.C., Farkas, O., Tomasi, J., Barone, V., Cossi, M., Cammi, R., Mennucci, B., Pomelli, C., Adamo, C., Clifford, S., Ochterski, J., Petersson, G.A., Ayala, P.Y., Cui, Q., Morokuma, K., Malick, D.K., Rabuck, A.D., Raghavachari, K., Foresman, J.B., Cioslowski, J., Ortiz, J.V., Baboul, A.G., Stefanov, B.B., Liu, G., Liashenko, A., Piskorz, P., Komaromi, I., Gomperts, R., Martin, R.L., Fox, D.J., Keith, T., Al-Laham, M.A., Peng, C.Y., Nanayakkara, A., Gonzalez, C., Challacombe, M., Gill, P.M.W., Johnson, B., Chen, W., Wong, M.W., Andres, J.L., Gonzalez, C., Head-Gordon, M., Replogle, E.S., Pople, J.A.: Gaussian 03. Gaussian Inc., Wallingford CT (2004)
14. Gatti, C., Cargnoni, F., Bertini, L.: Chemical information from the source function. J. Comp. Chem. **24**, 422 (2002)

15. Hunter, G.: The exact one-electron model of molecular structure. Int. J. Quantum Chem. **29**, 197 (1986)
16. Jepsen, O., Andersen, O.K.: The stuttgart TB-LMTO-ASA program, version 4.7. Max-Planck-Institut für Festkörperforschung, Stuttgart (2000)
17. Kohout, M.: A measure of electron localizability. Int. J. Quantum Chem. **97**, 651–658 (2004)
18. Kohout, M.: Bonding indicators from electron pair density. Faraday Discuss. **135**, 43–54 (2007)
19. Kohout, M., Pernal, K., Wagner, F.R., Grin, Y.: Electron localizability indicator for correlated wavefunctions. I parallel-spin pairs. Theor. Chem. Acc. **112**, 453–459 (2004)
20. Kohout, M., Pernal, K., Wagner, F.R., Grin, Y.: Electron localizability indicator for correlated wavefunctions. II antiparallel-spin pairs. Theor. Chem. Acc. **113**, 287–293 (2005)
21. Kohout, M., Savin, A.: Atomic shell structure and electron numbers. Int. J. Quantum Chem. **60**, 875–882 (1996)
22. Kohout, M., Wagner, F.R., Grin, Y.: Electron localization function for transition-metal compounds. Theor. Chem. Acc. **108**, 150–156 (2002)
23. Kohout, M., Wagner, F.R., Grin, Y.: Electron localizability indicator for correlated wavefunctions. III: singlet and triplet pairs. Theor. Chem. Acc. **119**, 413–420 (2008)
24. Ponec, R.: Electron pairing and chemical bonds. chemical structure, valences and structural similarities from the analysis of the fermi holes. J. Math. Chem. **21**, 323–333 (1997)
25. Raub, S., Jansen, G.: A quantitative measure of bond polarity from the electron localization function and the theory of atoms in molecules. Theor. Chem. Acc. **106**, 223–232 (2001)
26. Savin, A., Jepsen, O., Flad, J., Andersen, O.K., H., P., von Schnering, H.G.: Electron localization in solid-state structures of the elements: the diamond structure. Angew. Chem. Int. Ed. Engl. **31**, 187 (1992)
27. Schaftenaar, G., Noordik, J.H.: Molden: a pre- and post-processing program for molecular and electronic structures. J. Comput.-Aided Mol. Design **14**, 123 (2000)
28. Schmider, H.L., Becke, A.D.: Chemical content of the kinetic energy density. J. Mol. Struct. (Theochem) **527**, 51 (2000)
29. Shaffer, A., Wierschke, S.G.: Comparison of computational methods applied to oxazole, thiazole, and other heterocyclic compounds. J. Comp. Chem. **14**, 75–88 (1993)
30. Silvi, B., Savin, A.: Classification of chemical bonds based on topological analysis of electron localization function. Nature **371**, 683 (1994)
31. Tsirelson, V., Stash, A.: Determination of the electron localization function from electron density. Chem. Phys. Lett. **351**, 142 (2002)
32. Wagner, F.R., Bezugly, V., Kohout, M., Grin, Y.: Charge decomposition analysis of the electron localizability indicator: A bridge between the orbital and direct space representation of the chemical bond. Chem. Eur. J. **13**, 5724–5741 (2007)
33. Wagner, F.R., Kohout, M., Grin, Y.: Direct space decomposition of eli-d: Interplay of charge density and pair-volume function for different bonding situations. J. Phys. Chem. A **112**, 9814–9828 (2008)
34. Werner, H.J., Knowles, P.J., Amos, R.D., Bernhardsson, A., Berning, A., Celani, P., Cooper, D.L., Deegan, M.J.O., Dobbyn, A.J., Eckert, F., Hampel, C., Hetzer, G., Korona, T., Lindh, R., Lloyd, A.W., McNicholas, S.J., Manby, F.R., Meyer, W., Mura, M.E., Nicklass, A., Palmieri, P., Pitzer, R., Rauhut, G., Schütz, M., Schumann, U., Stoll, H., Stone, A.J., Tarroni, R., Thorsteinsson, T.: MOLPRO, a package of ab initio programs

Index