TOPXD

VERSION 2.0

TOPOLOGICAL ANALYSIS PROGRAM FOR EXPERIMENTAL STATIC ELECTRON DENSITY BASED ON HANSEN-COPPENS MULTIPOLE FORMALISM

by

ANATOLIY VOLKOV
Department of Chemistry, State University of New York at Buffalo
Buffalo, New York 14260, USA
e-mail: `volkov@chem.buffalo.edu`

and

CARLO GATTI
Centro CNR per lo Studio delle Relazioni tra Struttura e Reattivita' Chimica
via Golgi 19, 20133 Milano, ITALY
e-mail: `c.gatti@csrsrc.mi.cnr.it`

December 13, 2000

Disclaimer

The request of the TOPXD program should be sent to Prof. Philip Coppens to the address below:

Prof. Philip Coppens
732 NSM Complex
Department of Chemistry
Sate University of New York at Buffalo
Buffalo NY 14260-3000

Tel: 716 645 6800 x2217, 2218
Fax: 716 645 6948
E-mail: coppens@acsu.buffalo.edu

USE OF TOPXD WILL IMPLY THAT THE USER WILL ADHERE TO THE FOLLOWING TERMS AND CONDITIONS:

The user of TOPXD program must be a registered user of the XD package.

The use of TOPXD should be acknowledged in any publications, which use results obtained with TOPXD. The acknowledgment should be of the form:

Volkov, A.; Gatti, C.; Abramov, Yu.; Coppens, P. *Acta Cryst. A.* **2000**, *56*, 252.

The user will not supply a copy of the code to any other laboratory or institution for any reason whatsoever. This in no way limits making copies of the code for backup purposes.

The user will not incorporate any part of TOPXD program into any other program system, without prior written permission of the authors.

The TOPXD program will NOT be used for profit-making purposes.

The user understands that no large program such as TOPXD can be considered to be bug free, and accordingly the authors supply the software on an “as is” basis, with no additional responsibility or liability. This software also comes without any kind of warranty.

Problem and bug reports, requests for enhancement, or other comments should be sent to:

`volkov@chem.buffalo.edu`

When reporting problems, please provide as much information as possible, including:

- ♦detailed description of the problem
- ♦platform you are running on, including compiler
- ♦input files
- ♦output file

Notations

Throughout this manual the following conventions in notation are used:

ARIAL bold font denotes keywords, which should be typed as shown (uppercase and lowercase should not matter, they are only used to distinguish between section and command keywords)

Arial italic font is used for parameters, which should be replaced with actual numerical or character values.

`Courier` font is used for definition of filenames.

`Courier bold` font denotes the system commands.

Symbols "**{ }**" enclose optional parts in command descriptors

Symbol "⇒" denotes the start of a new line (record)

The lists which show a number of choices are enclosed in square brackets "[ ]" with a vertical bar "|" as the delimiter.

Description of Acronyms Used in This Manual

Acronym	Description
QTAM	Quantum Theory of Atoms in Molecules (R.F.W. Bader)
$\rho(\mathbf{r})$	Electron density
$\nabla^2\rho(\mathbf{r})$	Laplacian ($\nabla \bullet \nabla$) of $\rho(\mathbf{r})$
$\boldsymbol{H}(\rho(\mathbf{r}))$	The Hessian (second derivatives) matrix of $\rho(\mathbf{r})$
λ_k	Eigenvalues ($\lambda_1 \le \lambda_2 \le \lambda_3$) of $\boldsymbol{H}(\rho(\mathbf{r}))$
CP(s)	Critical point(s): a point $\mathbf{r}$ where a given scalar f has $\nabla f(\mathbf{r})=0$
(m,n) CP	A critical point with rank m and signature n. The rank is the number of non-zero eigenvalues, the signature is the difference between the number of positive and negative eigenvalues of $\boldsymbol{H}(\rho(\mathbf{r}))$ at the CP.
BCP	Bond Critical Point [a (3,-1) CP in the $\rho(\mathbf{r})$ scalar field]
ZFS	Zero-Flux-Surface $[\nabla\rho(\mathbf{r}) \cdot \mathbf{n}(\mathbf{r}) = 0 \quad \forall \mathbf{r} \in surface]$
Atomic Basin	The space traversed by all the uphill $\nabla\rho(\mathbf{r})$ paths which terminate at a nucleus, which acts as a 3D attractor for its basin. The atomic basin is also the portion of space enclosed by a ZFS and including a nucleus.
NNA	Non-nuclear attractor: a (3,-3) CP of r at a position other than nuclei
NEA	Non-Equivalent (unique) Atom
AIL	Atomic interaction line (a curved path joining two nuclei along which ρ is a maximum with respect to any lateral displacement)
"Crystal Graph"	The network of AILs for a given crystal geometry (it is the crystal correspondent of the QTAM molecular graph for an isolated molecule)
AGL	Atomic graph line (a curved path joining two $-\nabla^2\rho(\mathbf{r})$ (3,-3) CPs along which $-\nabla^2\rho(\mathbf{r})$ is a maximum with respect to any lateral displacement)
VSCC	Valence shell charge concentration (the *atomic* valence shell region where $\nabla^2\rho(\mathbf{r})$ is negative)
IR	Integration Ray (in atomic properties evaluation)
EF	Eigenvector Following method (CP search)
NR	Newton-Raphson method (CP search)

CONTENTS

1. INTRODUCTION[1]

1.1 General

In order to fully incorporate the Quantum Theory of Atoms in Molecules[2] (QTAM) into routine X-ray charge density studies, we have interfaced the existing program TOPOND98,[3] originally written for the CRYSTAL98 package,[4] to the experimental-charge-density package XD.[5] While the evaluation of charge density properties at the critical points is included in XD (XDPROP), the main feature of this new program, TOPXD, is its capability to define atomic basin boundaries and integrate density functions within the basins, thus producing an extensive set of atomic properties, including net charges, dipole and higher electrostatic moments. The entire code is written in “almost standard” FORTRAN77 and can be ported to virtually any computer system.

The main features of the program are those of TOPOND98, i.e. a fully-automated chain-like search for critical points in ρ and $\nabla^2\rho$ fields, using either the conventional Newton-Raphson technique or the eigenvector following method,[6,7] a grid search of critical points in the asymmetric unit, the evaluation of atomic properties, finely tuned algorithms for the evaluation of atomic interaction lines or of atomic graphs and extensive 2D and 3D graphics capabilities.

The experimental electron density and its analytical derivatives up to order 2 are calculated using subroutines from the XD package, slightly modified for optimal performance. However, derivatives of a higher order (up to 4) are required when searching for Laplacian critical points in the field of the Laplacian of the electron density. Derivatives of 3rd and 4th order are evaluated in TOPXD as a numerical finite-difference approximation of the 1st and the 2nd order analytical derivatives. The numerical derivative approach has been described before[8] and was shown to be extremely useful when no analytical derivatives are available. For that purpose well known central-difference expressions with fourth-order error ($O(h^4)$) have been used:[9]

$$f'_x = \frac{-f_{x+2h} + 8f_{x+h} - 8f_{x-h} + f_{x-2h}}{12h} \quad ,$$

$$f''_x = \frac{-f_{x+2h} + 16f_{x+h} - 30f_x + 16f_{x-h} + f_{x-2h}}{12h^2} \quad ,$$

where x is the point at which the numerical derivative is evaluated and h is the step size. Higher order numerical derivatives (or partial derivatives) are not needed, because every derivative of order 3 to 4 can be represented as 1st or 2nd order finite-difference numerical approximation of the 1st or 2nd order analytical derivative using a simple chain rule, for example:

$$\frac{d^3\rho}{dx^2dy}=\frac{d}{dx}\left[\frac{d^2\rho}{dxdy}\right]=\frac{d}{dy}\left[\frac{d^2\rho}{dx^2}\right]=\frac{d^2}{dx^2}\left[\frac{d\rho}{dy}\right],$$

in which expressions in square brackets are analytical derivatives while the outer part is evaluated numerically.

The accuracy of the numerical differentiation of the electron density has been extensively tested by comparison of the numerical 1st and pure 2nd derivatives with those obtained analytically for a number of (3,-1) critical points and for some arbitrarily selected points. With a step size of $h=5\times10^{-3}$, the expected error in the numerical derivatives is only $O(h^4)= 6.25\times10^{-10}$. Numerical examples show the actual error to be less than 1×10^{-9} and practically nonexistent when DOUBLE PRECISION variables are used. A comparison of analytical mixed 2nd derivatives with those obtained by numerical finite-difference differentiation of the 1st analytical derivative shows the difference to be less than 1×10^{-9}. A drawback of numerical differentiation is that in order to approximate one derivative, several evaluations of the function are required. Indeed, in order to obtain a numerical approximation of a pure 2nd derivative, for example $d^2\rho/dx^2$, evaluation of density is required at 5 different points. Fortunately, due to the exceptional computational power of modern computers such evaluations are only slightly more costly than using pure analytical expressions.

1.2 Coordinate systems and units

TOPXD works in the XDPROP-like ‘global’ cartesian system (see XD manual). All input and output atomic Cartesian atomic coordinates are assumed to be in Ångstroms. Also, in some cases, fractional atomic coordinates can be used. Please, refer to TOPXD INSTRUCTIONS section for specifications of atomic coordinates for a particular option. The units of ALL OUTPUT parameters related to length are also Ångstroms. The charge density and its derivatives can be in either ATOMIC UNITS (AU) or ELECTRONS & ANGSTROMS. Internally, TOPXD uses ONLY atomic units.

2. INSTALLATION

The general TOPXD distribution includes the following programs:

TOPXD – main code

P2DCRY2000 – program for conversion of 2D plot data generated by TOPXD to HPGL format

4TOPXD – program for calculation of molecular dipole moments (and more) from TOPXD atomic basin integration output (provided on special request)

2.1 TOPXD

The latest version of TOPXD can always be downloaded from TOPXD WWW site:

http://harker.chem.buffalo.edu/public/topxd/index.html

TOPXD is supplied as a single **`tar`** archive file compressed with GNU **`gzip`** utility. Assuming that the archive name is `topxd.tar.gz` uncompress the file:

```
gunzip topxd.tar.gz
```

Expand the archive:

```
tar -xvf topxd.tar
```

At this stage **`./topxd`** subdirectory is created which contains the source code of TOPXD

Go to that subdirectory

```
cd topxd
```

Compile TOPXD using command:

```
make <comp>
```

Where *`<comp>`* is one of:

`sgi`	SGI MIPS Fortran77 compiler
`lf95`	LAHEY/FUJITSU Fortran95 compiler
`g77`	GNU Fortran77 compiler
`sun`	SUN Fortran77 compiler
`pgi`	Portland Group Fortran77 compiler
`alpha`	COMPAQ Alpha compiler

If you want to compile for another architecture or if your platform is not listed above, edit the file `mk.generic` and use command

```
make generic
```

to compile the code.

Compilation will be done in an architecture-dependant subdirectory. For example, when compiling with LAHEY/FUJITSU Fortran95 compiler (**`make lf95`**), object files will be created in **`topxd/obj.lf95`** directory while the executable file created will be **`topxd/topxd`**.

Please note, that TOPXD could also be compiled and executed on virtually any non-UNIX system, including *Macintosh* machines, *ALPHA* and *VAX* stations running *OpenVMS* and IBM-compatible PC's running any version of *Windows* or *DOS*. TOPXD should(?) compile and run(??) as long as a Fortran77/90/95 compiler is available and there is enough RAM.

TOPXD was tested to compile and run properly using:

1. SGI MIPS Fortran77 compiler on SGI Irix 6.x computers
2. SUN Fortran77 compiler on SUN Solaris and LINUX workstations
3. Lahey/Fujitsu Fortran95 and Portland Group Fortran77 compilers on Intel LINUX machines
4. IBM Fortran77 compiler under IBM AIX 4.x
5. COMPAQ Fortran77 compiler on Alpha LINUX machines.
6. GNU Fortran77 compiler on Alpha and Intel LINUX machines.

TOPXD should also know the location of the XD databank file, `xd.bnk`. This is done via setting of one of the following environmental variables (usually, in the **$HOME**`/.cshrc`) file:

```
setenv XD_DATADIR <directory where xd.bnk file can be found>
```

or

```
setenv XD_BANKDIR  <directory where xd.bnk file can be found>
```

Usually one doesn't have to worry about this if XD package has already been properly installed and configured on the system.

2.1 Supplementary programs

The provided Makefile will only build the executable of TOPXD, so the supplementary program P2DCRY2000 will have to be compiled manually using the following command:

```
<fc>  <options> p2dcry00.f utilplot.f -o p2dcry00
```

, where <*fc*> is the name of the Fortran77 complier and <*options*> are compiler-specific options (see `mk.*` files for a specific platform). This command produces the executable **`p2dcry00`** which can be put anywhere in the path (usually, **`/usr/local/bin`** is a good place).

3. RUNNING TOPXD AND TOPXD FILES

TOPXD requires only two input files:

`xd.mas`	– XD master file
`xd.res(xd.inp)`	– XD parameter file

XD parameter file with atomic positional and multipole parameters is usually created by XDLSM program and should NOT be modified for TOPXD.

XD master file (`xd.mas`) should contain TOPXD specific instructions (in addition to XD general instructions) described in the next section of this manual.

Once the desired TOPXD instructions are included and activated in XD master file, TOPXD can be run via command line:

```
topxd > "TOPXD-output-file" &
```

where **topxd** is the name of the TOPXD executable file. Both XD master and parameter files must be present in the current directory, otherwise the program stops and the error message is printed out.

In addition to "`TOPXD-output-file`", the TOPXD always creates `xdprep.log` file with information on multipole parameters, symmetry information etc. The file `xdprep.log` should be checked carefully when running TOPXD for the first time for a specific compound to check the correctness of selected TOPXD general parameters (see section 4.1). Other TOPXD output files (`gen_eq.log` and `debug_rho.log`) are optional and NOT created by default.

4. TOPXD INSTRUCTIONS

From XD master file (`xd.mas`) the TOPXD reads both general XD instructions (**CELL**, **LATT**, **SYMM** and **SCAT**) and TOPXD-specific instructions, the latter should begin with ***MODULE TOPXD** and be terminated by the **END TOPXD** line, just like for any other XD module. The input is in free format and NOT case sensitive (internally converted to UPPER case), but each instruction line should NOT be longer than 256 symbols. No concatenation of lines is allowed. Blank lines and lines beginning with the exclamation mark (**!**) are treated as comments and ignored. In general, most of the XD conventions regarding the format and style of `xd.mas` file should also be valid for TOPXD instructions.

The "logical" keywords are marked in this manual with "**(*)**". This means that the keyword can be "activated" by inserting asterisk "*****" right before the corresponding keyword, i.e. ***key** would activate the keyword **key** thus enabling the option described for keyword **key**.

The template of TOPXD instruction section (file `topxd.mas`) is provided in the distribution and can be appended to any conventional `xd.mas` file.

There are five "sections" in TOPXD program, each identified by a special keyword:

keyword	meaning
TRHO	topological analysis of $\rho(\mathbf{r})$
TLAP	topological analysis of $\nabla^2\rho(\mathbf{r})$
ATBP	atomic basin properties
PL2D	2D plots
PL3D	3D plot

4.1 General Instructions

The first section in ***MODULE TOPXD** part of `xd.mas` file is the general TOPXD instructions, which will apply to all other TOPXD instructions. All these instructions are optional, i.e. the defaults are provided internally for all of these options, yet it is recommended to always include these instructions in `xd.mas` file.

The description of selected general instructions is provided in file `xdprep.log`, which is created in the same directory from which TOPXD was started. It is important to examine `xdprep.log` file carefully for the first run of TOPXD for a particular compound…

The following general instructions can be specified:

⇒**TITL** *title-string*

title-string – the title for TOPXD run. It is read in free format as CHARACTER*80 variable starting from symbol number 5 in the **TITLE….** line.
Default: untitled

⇒**DEBG (*)symequiv (*)deriv**

(*)symequiv – [create/DO NOT create] file `gen_eq.log` with extensive information about symmetry-equivalent atoms generated using **SYMM** (see Chapter 2.2 of XD manual), **ALIM**, **BLIM** and **CLIM** (see below) instructions. This file can be large, so one would want to use this option only for the first time one runs TOPXD for a particular compound in order to check if symmetry-equivalent atoms have been generated properly.
Default: Do not create `gen_eq.log` file

(*)deriv – when activated it enables the debugging printout of charge density and its derivatives to file `debug_rho.log` for each ***xyz*** point. When using this option, please make sure that plenty of the disk space is available. In general, one should not use this option at all.
Default: No derivative debugging printout

⇒CGEN alim *ixmin ixmax* blim *iymin iymax* clim *izmin izmax*

ixmin ixmax – min. and max. cell limits for generating of symmetry equivalent molecules (atoms) along unit cell axis **A**.
Default: `-2 2`, i.e. generate `-2..+2` cells along unit cell axis **A**

iymin iymax – min. and max. cell limits for generating of symmetry equivalent molecules (atoms) along unit cell axis **B**.
Default: `-2 2`, i.e. generate `-2...+2` cells along unit cell axis **B**

izmin izmax – min. and max. cell limits for generating of symmetry equivalent molecules (atoms) along unit cell axis **C**.
Default: `-2 2`, i.e. generate `-2..+2` cells along unit cell axis **C**

⇒MISC rcut *rcut* dstep *dstep* (*)au

rcut – Maximum distance from ***xyz*** point to contributing pseudoatom in Ångstroms (as in XDPROP).
Default: 4.0 Å as in XD

dstep– Numerical differentiation step in Ångstroms (as in XDPROP). This parameter will only be applied to numerical evaluation of 3rd and 4th derivatives of $\rho(\mathbf{r})$ (see INTRODUCTION) since 1st and 2nd derivatives are evaluated analytically.
Default: 0.005 (Å)

(*)au – If activated all output parameters related to charge density and its derivatives are in **Atomic Units**.
Note: **This keyword has no effect in ATBP section of TOPXD, in which the output is ALWAYS in Atomic Units !!!!!**
Default (except ATBP section): Electrons & Ångstroms

4.2 Topological Analysis of Electron Density

4.2.1 Auto critical point search within molecular clusters built around "seed" point(s).

This is a fully-automated and chain-like search strategy for all kinds of critical points[*], using at each stage the eigenvector following (EF) step suitable for the kind of CP searched for. The search is performed within a finite region of space, which encloses a finite molecular cluster built-up around a specified "seed point" A. Size and origin of the cluster are given in following input.

Syntax:

⇒**TRHO (*)seed (*)all (*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **th** *th*
⇒ **[(*)fra | (*)car]** *x y z*
⇒ **. . . . .**

Description:

(*)seed – [perform/DO NOT perform] the search

(*)all – If this keyword is activated, all kinds of CPs are searched for, otherwise chain-like search stopped after (3,-1) CPs stages. This option saves the largest part of CPU time (if keyword **ail** is not activated) required by the automatic search (useful when very large clusters are defined around the seed point).

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated

(*)debug – Activation of this keyword enables the debug printing during the CP search

nstep *nstep* – *nstep* determines the maximum number of EF steps along each search

nnb *nnb* – Value of *nnb* determines the maximum number of symmetry-related stars of atoms to be included in the cluster generated around the "seed point". *nnb* also defines the number of neighbors in the nearest neighbor analysis around each unique CP (of any kind) found

rmax *rmax* – *rmax* is maximum radius of the clusters (Å). Each cluster includes all atoms within a sphere of radius *rmax*, centered on the "seed-point" A. *rmax* may locally reduce the actual value of *nnb*

---

[*] Contrary to TOPOND, the search of (3,-3) CPs associated to nuclear maxima is skipped in TOPXD, since the Slater type basis functions used in the multipolar expansion correctly yield a cusp at the nuclear position (and hence no CP). The non-nuclear attractor (3,-3) CPs can be recovered either as termini of a bond path associated to a (3,-1) CP or in the grid search for CPs (cfr. 4.2.6)

th *th* – If *th*≠0., (3,-1) CP search is performed among all the unique atom pairs whose internuclear distance falls below *th* (Å), otherwise (*th*=0.) the default value is used (5Å)

⇒ [**(*)fra** | **(*)car**] *x* *y* *z*

Coordinates (*x y z*) of the "seed point" A (center of the cluster) in fractional (**fra**) or Cartesian (Å) (**car**) units. Repeat this line ***n*** times for ***n*** "seed points" (one "seed point" per line). Search for a particular "seed point" will only be performed if the keyword **fra**|**car** is activated (i.e. ***car** or ***fra**)

4.2.2 Auto critical point search within molecular clusters built around each of the unique atoms.

This is a fully automated and chain-like search strategy for all kinds of critical points,[*] using at each stage the eigenvector following (EF) step suitable for the kind of CP searched for. The search is performed within a finite region of space, which is defined by building-up finite molecular clusters centered on each of the unique atoms of the unit cell. Size of clusters is given in input

Syntax:

⇒**TRHO (*)cluster (*)all (*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **th** *th*

Description:

(*)cluster – [perform/DO NOT perform] the search

(*)all – If this keyword is activated, all kinds of CPs are searched for, otherwise chain-like search stopped after (3,-1) CPs stages. This option saves the largest part of CPU time (if keyword **ail** is not activated) required by the automatic search (useful when very large clusters are defined around each unique atom).

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP. [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF steps along each search

nnb *nnb* – Value of *nnb* determines the maximum number of symmetry-related stars of atoms to be included in the cluster generated around each unique atom. *nnb* also defines the number of neighbors in the nearest neighbor analysis around each unique CP (of any kind) found

rmax *rmax* – *rmax* is maximum radius of the clusters (Å). Each cluster includes all atoms within a sphere of radius *rmax*, centred on the unique atom. *rmax* may locally reduce the actual value of *nnb*

th *th* – If *th*≠0., (3,-1) CP search is performed among all the unique atom pairs whose internuclear distance falls below *th* (Å), otherwise (*th*=0.0) the default value is used (5Å)

---

[*] Contrary to TOPOND, the search of (3,-3) CPs associated to nuclear maxima is skipped in TOPXD, since the Slater type basis functions used in the multipolar expansion correctly yield a cusp at the nuclear position (and hence no CP). The non-nuclear attractor (3,-3) CPs can be recovered either as termini of a bond path associated to a (3,-1) CP or in the grid search for CPs (cfr. 4.2.6)

4.2.3 Auto critical point search between unique atom pairs.

Syntax:

⇒**TRHO (*)pairs** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **th** *th* **{ pc** *pc* **}**

Description:

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**ncp**[*]	(3,+1)	**rcp**
(3,-1)	**bcp**	(3,+3)	**ccp**

= **an** – Cioslowski's analytical determination of atomic interaction lines [10]

(*)pairs – [perform/DO NOT perform] the search

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP. [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated. This keyword can not be activated if *meth*=**an**

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps along each search

nnb *nnb* – (3,-1) CPs are searched among all the unique pairs generated from a set of nuclei. The set is generated by constructing clusters of atoms around each unique atom of the unit cell. The value of *nnb* determines the number of stars of neighbors used in the cluster construction around each unique atom.

rmax *rmax* – *rmax* is maximum radius of the clusters (Å) (see **nnb** *nnb* above). *rmax* value may (locally) reduce the actual value of *nnb*

th *th* – If *th*≠0, (3,-1) CP search is performed among all the unique atom pairs whose internuclear distance falls below *th* (Å), otherwise (*th*=0) the default value is used (4Å)

---

[*] *type*=**ncp** is suggested only when a non-nuclear attractor is searched for, since (3,-3) CPs are not associated to nuclear positions in TOPXD.

{ pc *pc* **}** – should only be specified if *meth*=**nr**

pc≠0 : if a CP is not found between A-B atom pair, the starting point of the NR search is displaced along the internuclear axis from the mid-point of the axis to the following two positions :

$\mathbf{r}'_{\text{start}} = \mathbf{r}_A + pc * (\mathbf{r}_B - \mathbf{r}_A);\quad \mathbf{r}''_{\text{start}} = \mathbf{r}_A + (1. - pc) * (\mathbf{r}_B - \mathbf{r}_A),$

pc=0 : specifies the default value of *pc* (0.4).

4.2.4 Critical point search from a starting set of points.

Syntax:

⇒**TRHO (*)points** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **[fra | car]**
⇒ *x y z*
⇒ . . . .

Description:

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**ncp**[*]	(3,+1)	**rcp**
(3,-1)	**bcp**	(3,+3)	**ccp**

(*)points – [perform/DO NOT perform] the search

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP. [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps along each search

nnb *nnb* – The value of *nnb* defines the number of star of neighbors in the nearest neighbor analysis around each unique CP (of any kind) found.

rmax *rmax* – *rmax* is the maximum distance (Å) from the CP, considered in the nearest neighbor analysis around each unique CP found (see **nnb** *nnb* above). *rmax* value may locally reduce the actual value of *nnb*

[car | fra] – The coordinates of starting points are assumed to be in fractional (**fra**) or Cartesian (Å) (**car**) coordinates

⇒ *x y z* – coordinates of the starting point (units depend on **car** | **fra** keyword above). Repeat this line ***n*** times for ***n*** starting points (one set of coordinates per line)

---

[*] *type*=**ncp** is suggested only when a non-nuclear attractor is searched for, since (3,-3) CPs are not associated to nuclear positions in TOPXD.

4.2.5 Critical point search along the line joining two nuclei (*A* and *B*) and/or two general points (*a* and *b*)

Syntax:

⇒**TRHO (*)line** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax*
⇒ < *line specification* >
⇒ . . . .

Description:

meth is the method for a CP search and can be specified in one of the following formats:

= **nr**	–	Newton-Raphson (NR) algorithm used in the CP search (strongly recommended in this case, unless one is looking for a specific type of CP along the line and does not care of the other CPs which may be present along the line)
= **ef** *type*	–	Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**ncp**[*]	(3,+1)	**rcp**
(3,-1)	**bcp**	(3,+3)	**ccp**

(*)line – [perform/DO NOT perform] the search

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP. [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – determines the maximum number of EF or NR steps along each search (use a very small number of steps in this case, say no more than 5-8, since the search is repeated 40 times, starting from 40 evenly distributed points along the line)

nnb *nnb* – The value of *nnb* defines the number of star of neighbors in the nearest neighbor analysis around each unique CP (of any kind) found.

rmax *rmax* – *rmax* is the maximum distance (Å) from the CP, considered in the nearest neighbor analysis around each unique CP found (see **nnb** *nnb* above). *rmax* value may locally reduce the actual value of *nnb*

---

[*] *type*=**ncp** is suggested only when a non-nuclear attractor is searched for, since (3,-3) CPs are not associated to nuclear positions in TOPXD.

<*line specification*> can be given in one of the formats:

⇒ **(*)atom** *label* **toneighbor** *i1… i(n)*

CP search along the line(s) joining the unique atom ***A*** with label *label* and its *i1..i(n)* neighbor(s) (atom ***B***), where *i* is the "NEW"" number in the "Clusters around each of the unique atoms" printing at the beginning of the TOPXD output. The search will only be performed if keyword **atom** is activated (i.e., ***atom**). Repeat this line ***n*** times for ***n*** unique atoms.

⇒ **(*)point [car | fra]** *x1 y1 z1 x2 y2 z2*

CP search along the line joining two points ***a*** and ***b*** with coordinates (*x1 y1 z1*) and (*x2 y2 z2*), respectively. The coordinates are in Cartesian (Å) (**car**) or fractional (**fra**) units. The search will only be performed if keyword **point** is activated (i.e., ***point**). Repeat this line ***n*** times for ***n*** point pairs

4.2.6 Grid search for critical points.

Syntax:

⇒**TRHO (*)grid** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax*
⇒ **xmin** *xmin* **xmax** *xmax* **xstep** *xstep*
⇒ **ymin** *ymin* **ymax** *ymax* **ystep** *ystep*
⇒ **zmin** *zmin* **zmax** *zmax* **zstep** *zstep*

Description:

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search (strongly recommended in this case, unless one is looking for a specific type of CP in the cell volume explored and does not care of the other CP types which may be there present)

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**ncp**[*]	(3,+1)	**rcp**
(3,-1)	**bcp**	(3,+3)	**ccp**

(*)grid – [perform/DO NOT perform] the search (**WARNING !!!! The grid search is very costly if the whole asymmetric unit is explored!!!!**)

(*)ail – If this keyword is activated (***ail**) atomic interaction line (AIL) lengths and termini are evaluated numerically for each unique (3,-1) CP. [Warning: costly!], otherwise atomic interaction line lengths and termini are not evaluated

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps along each search

nnb *nnb* – The value of *nnb* defines the number of star of neighbors in the nearest neighbor analysis around each unique CP (of any kind) found.

rmax *rmax* – *rmax* is the maximum distance (Å) from the CP, considered in the nearest neighbor analysis around each unique CP found (see **nnb** *nnb* above). *rmax* value may locally reduce the actual value of *nnb.*

---

[*] *type*=**ncp** is suggested only when a non-nuclear attractor is searched for, since (3,-3) CPs are not associated to nuclear positions in TOPXD

Grid specifications:

⇒**xmin** *xmin* **xmax** *xmax* **xstep** *xstep* (fractional units)
xmin xmax xstep determine the minimum, maximum and grid interval along crystal X-axis.

⇒**ymin** *ymin* **ymax** *ymax* **ystep** *ystep* (fractional units)
ymin ymax ystep determine the minimum, maximum and grid interval along crystal Y-axis.

⇒**zmin** *zmin* **zmax** *zmax* **zstep** *zstep* (fractional units)
zmin zmax zstep determine the minimum, maximum and grid interval along crystal Z-axis.

4.2.7 Profiles of ρ(r), ∇^2ρ(r) and λ_3 along the line joining two nuclei (*A* and *B*) and/or two general points (*a* and *b*)

Profiles of ρ(**r**), ∇^2ρ(**r**) and λ_3 are written to Fortran units 95, 96, 97, respectively. Units of ρ(**r**), ∇^2ρ(**r**) and λ_3 are determined by keyword **(*)au** in TOPXD GENERAL INSTRUCTIONS section of this manual.

Syntax:

⇒**TRHO (*)profile perstep** *n*
⇒ < *profile specification* >
⇒ . . . .

Description:

(*)profile – [evaluate/DO NOT evaluate] the profile

perstep *n* – determines the percentage step ***s*** along **A-B** or ***a-b***
if *n*=1 , s= 0.01×$R_{\mathbf{A-B}}$ (or R_{a-b})
if *n*=100, s= 1×$R_{\mathbf{A-B}}$ (or R_{a-b})

< *profile specification* > can be given in one of the following formats:

⇒**(*)atom** *label* **toneighbor** *i1… i(n)*
Profile along the line(s) joining the unique atom ***A*** with label *label* and its *i1..i(n)* neighbor(s) (atom ***B***), where *i* is the "NEW'" number in the "Clusters around each of the unique atom'" printing at the beginning of the TOPXD output. The profiling will only be performed if keyword **atom** is activated (i.e., ***atom**). Repeat this line ***m*** times for the ***m*** unique atoms to be considered.

⇒**(*)point** [**car** | **fra**] *x1 y1 z1 x2 y2 z2*
Profile along the line joining two points ***a*** and ***b*** with coordinates (*x1 y1 z1*) and (*x2 y2 z2*), respectively. The coordinates are in Cartesian (Å) (**car**) or fractional (**fra**) units. The profiling will only be performed if keyword **point** is activated (i.e., ***point**). Repeat this line ***n*** times for ***n*** point pairs.

4.3 Topological Analysis of the Laplacian of Electron Density.

4.3.1 Auto critical point search within the concentration (or depletion) shells of unique atoms and/or non-nuclear attractors.

Usually the search is performed in the valence shell charge concentration (VSCC) of each selected unique atom. Yet, a suitable choice for the sphere radius (*rstar* parameter, see below) allows for a CP search in (any of) the depletion shells of the selected unique atom.

Syntax:

⇒**TLAP (*)auto** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **ntheta** *ntheta* **nphi** *nphi*
⇒< *atom(s) specifications* >
⇒< *NNA specifications* >
⇒. . . . .

Description:

(*)auto – [perform/DO NOT perform] the search

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**cccp**	(3,+1)	**s2cp**
(3,-1)	**s1cp**	(3,+3)	**cdcp**

(*)ail – If this keyword is activated (***ail**) atomic graph line (AGL) lengths and termini are evaluated numerically for each unique (3,-1) CP [Warning: costly!]. The AGL is the union of the unique pair of $\nabla(\nabla^2\rho)$ trajectories that originate at the (3,-1) $-\nabla^2\rho$ CP and terminate at neighboring (3,-3) $-\nabla^2\rho$ CPs.

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps for each search

nnb *nnb* – *nnb* is the number of stars of neighboring atoms which are considered in the nearest neighbor analysis around each unique CP found.

rmax *rmax* – *rmax* is maximum sphere radius (Å) used to determine the neighboring atoms around each unique CP in the nearest neighbor analysis (see **nnb** *nnb* explanation). *rmax* may, locally, reduce the actual value of *nnb*

ntheta *ntheta* nphi ***nphi***

CPs search is started from points located on the surface of a sphere, centered on the nucleus of a given unique atom or at the NNA location. The number of starting points is determined by the intervals *ntheta, nphi* chosen for the polar coordinates θ and φ, respectively

< *atom(s) specifications* > are given in the following format:

⇒ **(*)atoms** *label1…label(n)*   **nmax** *nmax*   **rstar** *rstar*

Description:

The CP search will only be performed if the keyword **atoms** is activated (***atoms**).

label1..label(n) – labels of unique atoms for which the CP search will be performed.

nmax = 0 –normal search for
≠ 0 – if EF method is used the search for each atom is stopped when *nmax* different CPs of the required type are found

rstar = 0 - the default sphere radius is adopted in the CP search. The radius is taken to be equal to the distance from the nucleus to the spherical surface where $-\nabla^2\rho$ attains its maximum value in the valence shell of the isolated atom.
≠ 0 - the sphere radius is taken to be equal to *rstar* value (Å)

Note: Insert this record as many times as needed for unique atoms for which the CP search is desired. One may group in a single record those unique atoms that are characterized by equal *rstar* and *nmax* values.

< *NNA specifications* > are given in the following format (insert 1 record per each NNA):

⇒ **(*)nna**   **x** *x*   **y** *y*   **z** *z*   **nmax** *nmax*   **rstar** *rstar*

Description:

The CP search for this NNA will only be performed if the keyword **nna** is activated (***nna**).
x y z - Cartesian coordinates of the current NNA (Å)
nmax = 0 –normal search for current NNA
≠ 0 – if EF method is used the search for the current NNA is stopped when *nmax* different CPs of the required type are found
rstar - the sphere radius is taken to be equal to *rstar* value (Å)

4.3.2 Critical point search started from a given set of points.

Syntax:

⇒**TLAP (*)points** *meth* **(*)ail (*)debug nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **nmax** *nmax*
⇒ [**(*)car** | **(*)fra**] x y z
⇒ . . . .

Description:

(*)points – [perform/DO NOT perform] the search

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**cccp**	(3,+1)	**s2cp**
(3,-1)	**s1cp**	(3,+3)	**cdcp**

(*)ail – If this keyword is activated (***ail**) atomic graph line (AGL) lengths and termini are evaluated numerically for each unique (3,-1) CP [Warning: costly!]. The AGL is the union of the unique pair of $\nabla(\nabla^2\rho)$ trajectories that originate at the (3,-1) $-\nabla^2\rho$ CP and terminate at neighboring (3,-3) $-\nabla^2\rho$ CPs.

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps along each search (use a very small number of steps in this case, say no more than 5-8, since the search is repeated 40 times, starting from 40 evenly distributed points along the line)

nnb *nnb* – *nnb* is the number of stars of neighboring atoms which are considered in the nearest neighbor analysis around each unique CP found.

rmax *rmax* – *rmax* is maximum sphere radius (Å) used to determine the neighboring atoms around each unique CP in the nearest neighbor analysis (see **nnb** *nnb* explanation). *rmax* may, locally, reduce the actual value of *nnb*

nmax *nmax*

nmax = 0 –normal search for current NNA

nmax ≠ 0 – if EF method is used the search for the current NNA is stopped when *nmax* different CPs of the required type are found

⇒ **[(*)car | (*)fra]** x y z

Starting point coordinates (*x y z*) in Cartesian (Å) (**car**) or fractional (**fra**) units. The search will only be performed if the **car/fra** keyword is activated (i.e., ***car** or ***fra**). Insert this record ***n*** times to start CP search from ***n*** starting points.

4.3.3 Critical point search along the line joining two nuclei (*A* and *B*) and/or two general points (*a* and *b*)

Syntax:

⇒**TLAP (*)line** *meth* **(*)ail** **(*)debug** **nstep** *nstep* **nnb** *nnb* **rmax** *rmax* **nmax** *nmax*
⇒ *< line specifications >*
⇒ . . . .

Description:

(*)line – [perform/DO NOT perform] the search

meth is the method for a CP search and can be specified in one of the following formats:

= **nr** – Newton-Raphson (NR) algorithm used in the CP search

= **ef** *type* – Eigenvector Following (EF) algorithm used in the CP search: The value of variable *type* depends on the kind of CP to be searched for (only one *type* can be specified per instruction):

CP type	Value of *type*	CP type	Value of *type*
(3,-3)	**Cccp**	(3,+1)	**s2cp**
(3,-1)	**s1cp**	(3,+3)	**cdcp**

Note: The *meth* = **nr** is generally the best choice in the CP search along a line, unless one is only interested in a particular kind of CP and does not care of the CPs of different type which might be found along or in the neighbors of such a line.

(*)ail – If this keyword is activated (***ail**) atomic graph line (AGL) lengths and termini are evaluated numerically for each unique (3,-1) CP [Warning: costly!]. The AGL is the union of the unique pair of $\nabla(\nabla^2\rho)$ trajectories that originate at the (3,-1) $-\nabla^2\rho$ CP and terminate at neighboring (3,-3) $-\nabla^2\rho$ CPs.

(*)debug – Activation of this keyword enables the debug printing during CP search

nstep *nstep* – *nstep* determines the maximum number of EF or NR steps for each search

nnb *nnb* – *nnb* is the number of stars of neighboring atoms which are explored from any unique CP found. A nearest neighbor analysis of atoms around each unique CP is printed.

rmax *rmax* – *rmax* is maximum sphere radius (Å) used to determine the neighboring atoms around each unique CP in the nearest neighbor analysis (see **nnb** *nnb* explanation). *rmax* may, locally, reduce the actual value of *nnb*

nmax *nmax*
nmax = 0 –normal CP search
nmax ≠ 0 – if EF method is used, the CP search along the line is stopped when *nmax* different CPs of the required type are found

<*line specifications*> can be given in one of the following formats:

⇒ **(*)atom** *label* **toneighbor** *i1… i(n)*

CP search along the line(s) joining the unique atom ***A*** with label *label* and its *i1..i(n)* neighbor(s) (atom ***B***), where *i* is the "NEW"" number in the "Clusters around each of the unique atom"" printing at the beginning of the TOPXD output. The search will only be performed if keyword **atom** is activated (i.e., ***atom**).

⇒ **(*)point [car | fra]** *x1 y1 z1 x2 y2 z2*

CP search along the line joining two points ***a*** and ***b*** with coordinates (*x1 y1 z1*) and (*x2 y2 z2*), respectively. The coordinates are in Cartesian (Å) (**car**) or fractional (**fra**) units. The search will only be performed if keyword **point** is activated (i.e., ***point**). Repeat this line ***n*** times for ***n*** point pairs

4.4 Evaluation of atomic and/or NNA basin properties

Atomic (and NNA) basin integration part of TOPXD consists of the following 3 types of instructions:

1. Capture sphere specifications for all unique atoms (optional)
   ATBP spheres …..

2. Instructions for integration of unique atoms (required)
   ATBP (*)atoms …..

3. NNA(s) specifications, if present (optional).
   ATBP NNA …..
   ………………….

4.4.1 Capture sphere specifications for unique atoms.

Although this(these) instruction(s) is(are) optional, it is still recommended to specify the capture spheres for all atoms as it should considerably reduce the program runtime.

Syntax:

⇒**ATBP spheres** *label1 rad1 …. label(n) rad(n)*

Description:

rad is the radius of a capture sphere (Å) for unique atom with label *label*. *Rad* should be generally taken equal to the distance from the nucleus to the nearest of the $\rho(\mathbf{r})$ BCPs which lie on the $\nabla\rho(\mathbf{r})$ zero-flux surface (ZFS), enclosing the atomic basin of the unique atom with label *label*.

It is recommended to specify the capture sphere radius for ALL unique atoms !!!

4.4.2 Instructions for integration of unique atoms.

This required instruction requests the integration of atom basin(s) of unique atom(s).

Syntax:

⇒**ATPB (*)atoms** *label1…label(n)* **[izfs | zfs]** **nvi** *nvi* **irsur** *irsur* **(*)irsav** **(*)rest** **(*)debug** **phi** *nphi* **th** *nth* **rad** *rad* **accur** accur **{ nbcp** *nbcp* **[car | fra] }**
⇒ **{. . . .** *if* nbcp > 0 *insert* nbcp *records with BCP* x y z *coordinates* **. . . . . }**

Notes:

There can be as many lines with **ATBP (*)atoms .....** instructions as the number of unique atoms

Description:

atoms – The activation of this keyword (***atoms**) requests the integration of unique atoms with labels *label1…label(n)*. If this keyword is not activated the integration will not be performed

label1…label(n) – Labels of unique atoms to be integrated if **atoms** keyword is activated

izfs – indirect determination of the ZFS.[11] This approach seldom fails to deliver accurate atomic ZFSs but its computational cost is often exceedingly high

zfs – the determination of the ZFSs is achieved in two steps:
a) direct determination[12]
b) indirect determination[11] for those integration rays (IRs) whose length was not correctly recovered in step a)

nvi *nvi*

nvi is the number of star of neighbors of the current unique atom(s) which have to be considered as possible attractors of the $\nabla\rho(\mathbf{r})$ paths launched from points along the integration rays

irsur *irsur*

irsur = 0 – normal run
irsur = 1 – the lengths of the integration rays obtained in a previous run are read from Fortran unit 97 and used as an initial guess for the IRs' lengths
irsur = -1 – the lengths of the integration rays are kept fixed to those obtained in a previous run and read from Fortran unit 97

(*)irsav – If this keyword is activated the lengths of the integration rays are saved in Fortran unit 98

Notes:

The use of *irsur*≠0 requires to have set ***irsav*** in a previous ATBP run. The ZFS thus saved on Fortran unit 98 may be used in a following run (*irsur*≠0), by copying it on Fortran unit 97. Use *irsur* = -1 to run the integration step separate from the ZFS determination step; put *irsur* = 1 to use the ZFS obtained in a previous run for a given unique atom (obtained, for example, using a different multipole model) as a starting guess for the new ZFS determination

(*)rest – If this keyword activated the run is restarted from (partial) surface data stored in Fortran unit 96 from a previous aborted run

(*)debug – Activates the debug printing during the ZFS determination and integration

phi *nphi* **th** *nth*

Angular integration parameters outside β-sphere: number of φ (*nphi*) and θ (*nth*) grid points (see also FAQ section)

rad *rad*

rad is the number of radial integration points inside β-sphere

accur accur

Parameter *accur* determines the numerical accuracy of each IR length (in **AU**) and thus of the overall ZFS determination. The default value is 0.001 au. A larger *accur* value reduces the computational time (see also FAQ section).

{ nbcp *nbcp* [**car** | **atom**] **}**

These optional parameters should only be used for a two-step procedure in ZFS determination (see keyword **zfs** above). *nbcp* is number of (3,-1) CPs associated to atomic interaction lines (AIL) [see **TRHO** section] linking the current unique atom to other atoms and/or NNAs. Keyword **car**|**atom** determines the format for specification of (3,-1) critical points to be read in the following *nbcp* records (if *nbcp* > 0).

1. Keyword **car** specifies that the Cartesian coordinates of a (3,-1) critical point should be given:

⇒ **{** *x y z* **}**

2. Keyword **atom** specifies the atom linked to the current unique atom

⇒ **{** *inum itx ity itz* **}**

inum is the sequence number of a linked atom in the TOPXD printing of all atoms in the unit cell, while *itx* *ity* and *itz* specify the indices (direct cell) of the cell where the linked atom *inum* is located.

4.4.3 . NNA(s) specifications.

Use this optional instruction if NNA(s) is(are) present in the structure.

Syntax:

⇒**ATBP NNA** *nna*

nna

nna is the number of NNAs in the structure
Default: 0 (no NNAs in the structure)

If *nna > 0* insert *nna* records with NNA specifications in the following format:

Syntax:

⇒ **x** *x* **y** *y* **z** *z* **(*)integ sphere** *rad* **{ [izfs** | **zfs] nvi** *nvi* **irsur** *irsur* **(*)irsav (*)rest (*)debug phi** *nphi* **th** *nth* **rad** *rad* **accur** accur **} { nbcp** *ncp* [**car** | **fra**] **}**
⇒ **{ . . . .** *if* **nbcp** *> 0 insert are* **nbcp** *records with BCP x y z coordinates* **. . . . . }**

Notes:
There should be as many lines with NNA specifications as the number of NNAs in the structure.

Description:

x *x* **y** *y* **z** *z*
Cartesian coordinates of this NNA (Å)

(*)integ – If this keyword is activated (***integ**) the integration of the basin of this NNA will be performed

sphere *rad*
rad is the radius of a capture sphere for this NNA (Å). *Rad* should be generally taken equal to the distance from the NNA to the nearest of the ρ(**r**) BCPs which lie on the ∇ρ(**r**) zero-flux surface (ZFS), enclosing the NNA basin

Note: The following keywords should only be used if keyword **integ** is activated (***integ**)

izfs – indirect determination[11] of the ZFS. This approach seldom fails to deliver accurate atomic ZFSs but its computational cost is often exceedingly high

zfs – the determination of the ZFSs is achieved in two steps:
a) direct determination[12]
b) indirect determination[11] for those integration rays (IRs) whose length was not correctly recovered in step a)

nvi *nvi*

nvi is the number of star of neighbors of the current NNA which have to be considered as possible attractors of the ∇ρ(**r**) paths launched from points along the integration rays.

irsur *irsur*

irsur = 0 – normal run

irsur = 1 – the lengths of the integration rays obtained in a previous run are read from Fortran unit 97 and used as an initial guess for the IRs' lengths

irsur = -1 – the lengths of the integration rays are kept fixed to those obtained in a previous run and read from Fortran unit 97

(*)irsav – If this keyword is activated the lengths of the integration rays are saved in Fortran unit 98

Notes: The use of *irsur*≠0 requires to have set ***irsav** in a previous ATBP run. The ZFS thus saved on Fortran unit 98 may be used in a following run (*irsur*≠0), by copying it on Fortran unit 97. Use *irsur* = -1 to run the integration step separate from the ZFS determination step; put *irsur* = 1 to use the ZFS obtained in a previous run for a given unique NNA (for example obtained using a different multipole model) as a starting guess for the new ZFS determination

(*)rest – If this keyword activated the run is restarted from (partial) surface data stored in Fortran unit 96 from a previous aborted run

(*)debug – activates the debug printing during the ZFS determination and integration

phi *nphi* **th** *nth*

Angular integration parameters outside β-sphere: number of φ (*nphi*) and θ (*nth*) grid points

rad *rad*

rad is the number of radial integration points inside β-sphere

accur accur

Parameter *accur* determines the numerical accuracy of each IR length (in **AU**) and thus of the overall ZFS determination. The default value is 0.001 au. A larger *accur* value reduces the computational time

{ nbcp *nbcp* **[car | atom] }**

These optional parameters should only be used for a two-step procedure in ZFS determination (see keyword **zfs** above). *nbcp* is number of (3,-1) CPs associated to atomic interaction lines (AIL) [see **TRHO** section] linking the current NNA to other atoms and/or NNAs. Keyword **car|atom** determines the format for specification of (3,-1) critical points to be read in the following *nbcp* records (if *nbcp* > 0).

1. Keyword **car** specifies that the Cartesian coordinates of a (3,-1) critical point should be given:

⇒ **{** *x y z* **}**

2. Keyword **atom** specifies the atom linked to the current NNA

⇒ **{** *inum itx ity itz* **}**

inum is the sequence number of a linked atom in the TOPXD printing of all atoms in the unit cell, while *itx ity* and *itz* specify the indices (direct cell) of the cell where the linked atom *inum* is located.

4.5 2-Dimensional (2D) Plots

The 2D (and 3D) plot instructions have a slightly different format than others. The general format consists of the following 2 sections:

1. General instructions

   PL2D general

   ....

2. Specific plot instruction(s) (single instruction per each specific plot type)

   PL2D plot ....

The **PL2D general** instructions apply to all the specific plot instructions **PL2D plot** until the next **PL2D general** is given and so on. There is no limit on neither the number of **PL2D general** nor **PL2D plot** instructions.

The visualization of 2D plots using P2DCRY2000 and **`hp2xx`** programs is described in section 4.5.3 of this chapter.

Some of the TOPXD 2D plot files can also be visualized with program SURFER (a PC-DOS program for 3D plots). See next section for more information.

4.5.1 2D plot general instruction section.

This section MUST precede specific plot instructions. All parameters specified in this section will apply to the following specific plot instructions until the next general instruction section is given.

Syntax:

⇒**PL2D general**
⇒ *<point/atom A specification>*
⇒ *<point/atom B specification>*
⇒ *<point/atom C specification>*
⇒ **plotdim xmin** *xmin* **xmax** *xmax* **xstep** *xstep* **ymin** *ymin* **ymax** *ymax* **ystep** *ystep*
⇒ **origin shift** *ishft* **origin** *xo yo zo* **vmod** *vmod*
⇒ **misc size** *ax* **scale** *scale* **name** '*name*' **title** '*titile*'

Description:

⇒*<point/atom A/B/C specification>*

Three atoms or points (A,B,C) must be given to define the plot plane. The specification format is different for atoms and points:

1. Atoms can be specified using the following format

⇒**atom** *inum itx ity itz*

inum is the serial number of atom in the TOPXD printout of all atoms in the unit cell
itx ity itz – translations applied to fractional coordinates of atom with number *inum* along X, Y and Z-crystal axis, respectively

2. General points can be specified using the following format:

⇒**point** *x y z*

where *x y z* are Cartesian (Å) coordinates

Notes: If evaluation of molecular/crystal graph or, generally, of $\nabla\rho(\mathbf{r})$ trajectories is required, it is important to define the three atom/point in such a way that their associated clusters of neighboring atoms (see below) span adequately the plot plane.

⇒**plotdim xmin** *xmin* **xmax** *xmax* **xstep** *xstep* **ymin** *ymin* **ymax** *ymax* **ystep** *ystep*

The plot plane is **XY.** *xmin* and *xmax* define the minimum and maximum values along the plot X-axis, respectively, while *xstep* defines the grid interval along X. *ymin ymax* and *ystep* have the same meaning but for Y-axis of the plot. All these parameters must be given in Ångstroms.

⇒**origin shift** *ishft* **origin** *xo yo zo* **vmod** *vmod*

ishift = 0 - origin as in the original Cartesian frame. A warning message is issued if, as a consequence of a given choice of the origin, the atoms/point A,B,C do not longer lie in the XY plot plane.
= 1 – the origin of the plot is translated to a point lying on ABC plane (must specify the *xo yo* and *yo* coordinates, see below)
= 2 – the origin of plot is put at barycentre of the atoms/points, which define the ABC plane. A mass equal to 1 is assigned to all atoms (and points).
= 3 – the origin of the plot lies along the A-B axis and its actual position is defined by the value of *vmod* variable (see below)
= 4 – the origin of the plot is at atom/point A

xo yo zo – Cartesian (Å) coordinates of the origin of the plot

vmod – the *vmod* value shifts the origin along A–B, so if *vmod*=0 the origin is at A and if *vmod*=1 the origin is at B; Negative *vmod* values as well as values greater then 1 are also allowed.

⇒**misc size** *ax* **scale** *scale* **name** '*name*' **title** '*titile*'

ax = **a4** – plot size is A4
= **a3** – plot size is A3

scale – Plot scale (Å/cm)

name – Suffix name (maximum 24 characters, no blanks and no special symbols like `,″,^ etc) for the files which contain the values of each computed function *func* (see below) and the common information for the XYZ plot (system geometry, Euler rotation angles from crystal to plot frame, plot size, etc.). These files form the input for the P2DCRY2000 program.
Full names of the files are listed in Table 1 ('//' means character concatenation).
Common information for the XY plot is saved in file: `P2DCRYIN`//*name*
The files prefixed with '`SURF`' may also be read from the SURFER program (a PC-DOS program for representing the scalar function in the plot plane as a 2D surface in 3D space, something that it is often referred to as a 3D graph).

title – Plot title (maximum 80 symbols, blanks are allowed, no special symbols like `,″,^)

Table 1.

Scalar function (or $\nabla\rho(\mathbf{r})$ plot) type		File name
Electron density	$\rho(\mathbf{r})$	`SURFRHOO`//*name*
Laplacian of $\rho(\mathbf{r})$	$\nabla^2\rho(\mathbf{r})$	`SURFLAPP`//*name*
Negative of the Laplacian $\rho(\mathbf{r})$	$-\nabla^2\rho(\mathbf{r})$	`SURFLAPM`//*name*
Magnitude of the gradient of $\rho(\mathbf{r})$	$\lvert \nabla\rho(\mathbf{r}) \rvert$	`SURFGRHO`//*name*
$\nabla\rho(\mathbf{r})$ trajectories only		`TRAJGRAD`//*name*
Molecular/crystal graph (and atomic basin boundaries)		`MOLGRAPH`//*name*
$\nabla\rho(\mathbf{r})$ trajectories and molecular/crystal graph		`TRAJMOLG`//*name*

4.5.2 2D plot specific instructions

These instructions must follow the general 2D plot instructions.

Syntax:

⇒**PL2D plot** *func* **nstar** *nstar* **rmax** *rmax* **(*)test (*)cut** *cutr cutl* **{** *func* – dependent instructions **}**

Note: There is no limit for the number of **PL2D plot....** instructions (one instruction per line).

Description:

(*)plot – Activate this keyword (***plot**) to enable plotting of the following function.

func - function type; the following function types are recognized (the names of output files created are given in description of 2D plot general instructions and in Table 1) :

= **rhoo** – electron density ($\rho(\mathbf{r})$)
= **lapp** – laplacian of the electron density ($\nabla^2\rho(\mathbf{r})$)
= **lapm** – negative laplacian of the electron density ($-\nabla^2\rho(\mathbf{r})$)
= **grho** – magnitude of the gradient of electron density ($|\nabla\rho(\mathbf{r})|$)
= **trajgrad** – $\nabla\rho(\mathbf{r})$ trajectories only
= **molgraph** – molecular/crystal graph (and atomic basin boundaries)
= **trajmolg** – $\nabla\rho(\mathbf{r})$ trajectories and molecular/crystal graph

nstar *nstar*

nstar is number of star of neighbors used in cluster construction around A, B and C atoms/points. From these clusters the atoms lying (not lying) in the ABC plane are selected and their position denoted with *(+) in the plots. The *nstar* variable is also used in and determines:

a) Atom pairs considered in the evaluation of the molecular/crystal graph on the ABC plane
b) Number of origins (nuclei) for the downhill $\nabla\rho(\mathbf{r})$ trajectories (gradient paths) on the ABC plane.

rmax *rmax*

rmax determines the maximum radius (Å) of each cluster (see *nstar* variable) and may (locally) reduce the actual value of *nsta*

(*)test – If this keyword is activated (***test**) the program stops after printing the coordinates of A, B, C and corresponding clusters of atoms in the plot frame. Use this option to check if the choice of the plane ABC was correct at the first run.

(*)cut *cutr cutl*

If the keyword **cut** is activated (***cut**) the scalar functions of $\rho(\mathbf{r})$ and $|\nabla\rho(\mathbf{r})|$ are cut at the value of *cutr* and/or values of $\nabla^2\rho(\mathbf{r})$ and $-\nabla^2\rho(\mathbf{r})$ are cut at ±*cutl*, according to their sign (*cutr cutl* must be given Electrons and Angstroms). Cutting of the scalar function is generally required, especially in the case of the Laplacian, for representing the function in the plot plane as a 3D graph.

{ *func – dependant parameters in* **Plot 2D** *specific instructions* **}**

1. If *func* = **trajgrad** - add the following parameters to **PL2D plot trajgrad ...** instruction:

toler *tol1 tol2* **(*)plane npath** *npath* **nextr** *nextr*
{ if *nextr* > 0 add *nextr* lines with Cartesian (Å) *x y z* of attractors ∇ρ trajectories (one set per line) **}**

An atom in the XY plot plane is considered as an origin of downhill gradient paths or as a terminus of an atomic interaction line if its distance from plot edge is less than *tol1* (Å) and its | z-plot coordinate | is less than *tol2* (Å).

<u>**Note:**</u>

A “correct” tracing of ∇ρ(**r**) trajectories in the plot plane would require this plane to be a mirror plane. However, this is not always the case. For example, in the study of molecular crystals at experimental geometries, it is common practice to deal with quasi-mirror planes. To cover such cases, ∇ρ(**r**) trajectories are projected on the plot plane whenever their | z-plot coordinate | is less than *tol2* (Å).

If keyword **plane** is activated (***plane**) the starting point of each ∇ρ(**r**) trajectory segment (which are typically 10^{-4} to 10^{-2} Å long) is forced on the XY plane, while default is the normal tracing of ∇ρ(**r**) trajectories (with projection regulated by *tol2*).

The number of downhill ∇ρ(**r**) trajectories is defined by variable *npath* (the recommended value for *npath* is 36).

Variable *nextr* defines the number of other attractors, like non-nuclear (NNAs) attractors or the 2D attractors associated with a BCP, to be considered as origins of downhill ∇ρ(**r**) trajectories. A BCP is seen as a (2,-2) attractor whenever its associated IAS lies in the plot plane.

If *nextr* > 0 add *nextr* lines with Cartesian (Å) *x y z* coordinates of attractors of ∇ρ(**r**) trajectories (one set of *x y z* coordinates per line)

2. If *func* = **molgraph** - add the following parameters to **PL2D plot molgraph ...** instruction:

tol *tol1* *tol2* **(*)plane thr** *thr* **(*)tr1 (*)tr2 (*)tr3**

An atom in the XY plot plane is considered as an origin of downhill gradient paths or as a terminus of an atomic interaction line if its distance from plot edge is less than *tol1* (Å) and its | z-plot coordinate | is less than *tol2* (Å).

Note:

A "correct" tracing of molecular/crystal graphs in the plot plane would require this plane to be a mirror plane. However, this is not always the case. For example, in the study of molecular crystals at experimental geometries, it is common practice to deal with quasi-mirror planes. To cover such cases, $\nabla\rho(\mathbf{r})$ trajectories associated to the molecular/crystal graph are projected on the plot plane whenever their | z-plot coordinate | is less than *tol2* (Å).

If keyword **plane** is activated (***plane**) the starting point of each $\nabla\rho(\mathbf{r})$ trajectory segment is forced on the XY plane, while default is the normal tracing of $\nabla\rho(\mathbf{r})$ trajectories (with projection regulated by *tol2*).

The maximum distance between atomic pairs, which is taken into account during the evaluation of bonded pairs and the automated tracing of the molecular/crystal graph, is defined by the value of *thr* (Å).

The $\nabla\rho(r)$ trajectories that originate at BCPs and have as initial direction the *K*-th eigenvector of Hessian of $\rho(r)$ at BCPs, are traced out (***tr***K*) or skipped (**tr***K*). The *K*-th eigenvector is associated with the *K*-th eigenvalue of the Hessian λ_K ($\lambda_1 \leq \lambda_3 \leq \lambda_3$).

3. If *func* = **trajmolg** - add the following parameters to **PL2D plot trajmolg ...** instruction:

tol *tol1 tol2* **(*)plane thr** *thr* **(*)tr1 (*)tr2 (*)tr3 npath** *npath* **nextr** *nextr*
{ if *nextr* > 0 add *nextr* lines with Cartesian (Å) *x y z* of attractors ∇ρ trajectories (one set per line) **}**

An atom in the XY plot plane is considered as an origin of downhill gradient paths or as a terminus of an atomic interaction line if its distance from plot edge is less than *tol1* (Å) and its | z-plot coordinate | is less than *tol2* (Å).

Note:

A "correct" tracing of ∇ρ(**r**) trajectories in the plot plane (including those associated to the molecular/crystal graph) would require this plane to be a mirror plane. However, this is not always the case. For example, in the study of molecular crystals at experimental geometries, it is common practice to deal with quasi-mirror planes. To cover such cases, ∇ρ(**r**) trajectories are projected on the plot plane whenever their | z-plot coordinate | is less than *tol2* (Å).

If keyword **plane** is activated (***plane**) the starting point of each ∇ρ(**r**) trajectory segment is forced on the XY plane, while default is the normal tracing of ∇ρ(**r**) trajectories (with projection regulated by *tol2*).

The maximum distance between atomic pairs, which is taken into account during the evaluation of bonded pairs and the automated tracing of the molecular/crystal graph, is defined by the value of *thr* (Å).

The ∇ρ(r) trajectories that originate at BCPs and have as initial direction the *K*-th eigenvector of Hessian of ρ(r) at BCPs, are traced out (***tr***K*) or skipped (**tr***K*). The *K*-th eigenvector is associated with a *K*-th eigenvalue of the Hessian λ_K ($\lambda_1 \le \lambda_3 \le \lambda_3$).

The number of downhill ∇ρ(**r**) trajectories is defined by variable *npath* (the recommended value for *npath* is 36).

Variable *nextr* defines the number of other attractors, like non-nuclear (NNAs) attractors or the 2D attractors associated with a BCP, to be considered as origins of downhill ∇ρ(**r**) trajectories. A BCP is seen as a (2,-2) attractor whenever its associated IAS lies in the plot plane.

If *nextr* > 0 add *nextr* lines with Cartesian (Å) *x y z* coordinates of attractors of ∇ρ(**r**) trajectories (one set of *x y z* coordinates per line)

4.5.3 Visualization of 2D plots.

The visualization of 2D plots created by TOPXD consists of the following two steps:

1. Conversion of TOPXD output 2D plot data to HPGL format using program P2DCRY2000
2. Visualization or conversion of HPGL graphics file to other graphics format using program **hp2xx**.

The 2D plots created by TOPXD must first be converted to HPGL graphics format using program P2DCRY2000 provided in the distribution. The compilation of P2DCRY97 is described in section 2.2 of this manual. The P2DCRY2000 works in interactive mode and is very easy to use.

The HPGL graphics files created by P2DCRY2000 program can be visualized or converted to some other graphics format by program **hp2xx** (part of GNU software), which can be downloaded from:

http://www.gnu.org/software/hp2xx/hp2xx.html

At this time the latest version of **hp2xx** is 3.3.2 (00/02/12).

Once the **hp2xx** is installed, the HPGL file created by P2DCRY2000 can be visualized in graphical display using command:

hp2xx <*name of HPGL graphics file from P2DCRY2000*>

For better resolution one can use **–d** *dpi_value* option, i.e. command:

hp2xx <*name of HPGL graphics file from P2DCRY2000*> **–d** 300

will display an image with 300 DPI rasterization.

In order to change the thickness of the lines in the image one can use the **–p** option, which controls size (in pixels) of the virtual plotting pen. There are total 8 pens simulated. Each pen can be assigned a different size. Thus the command:

hp2xx <*name of HPGL graphics file from P2DCRY2000*> **–p** 43568111

requests size 4 for pen 1, size 3 for pen 2, size 5 for pen 3, size 6 for pen 4, size 8 for pen 5, and size 1 for pens 6,7 and 8.

It should be noted that P2DCRY2000 program uses different pens to draw different objects in the TOPXD 2D drawing. The assignment of pens is as follows:

Pen number	Corresponding object in the drawing
1	contours (and ∇ρ(**r**) trajectories)
2	dash lines
3	BCP position
4	bond path
5	nuclear positions
6	plot info and border
7	atomic basin boundaries
8	not used

Each pen can also be assigned a different color using the option **–c**. The use of this option is similar to that of **–d**, i.e. for each pen one has to specify a color number instead of a size. The color-coding scheme is as follows: 0=off, 1=black, 2=red, 3=green, 4=blue, 5=cyan, 6=magenta, 7=yellow. Thus, the command:

hp2xx *<name of HPGL graphics file>* **–c** 276431

will produce a plot with pen 1 (contours, if any) being drawn in red color, pen 2 (dashed lines, if any) in yellow color, pen 3 (BCP position, if any) in magenta, pen 4 (bond path, if any) in blue, pen 5 (nuclei positions) in green, pen 6 (plot info and border) in black color.

hp2xx also provides an option (**–r** *rotation_angle_in_degrees*) to rotate the object (image) prior to all scaling operations. Thus, the command **-r** 90 rotates the entire picture on 90°, converting from portrait to landscape format and vice versa. In general, any reasonable rotation angle is valid.

In addition to visualization, **hp2xx** also provides an option to convert HPGL graphics file to more common and more supported vector and/or raster formats. There two options which control the output format type (**–m** *format*) and output filename (**–f** *filename*).

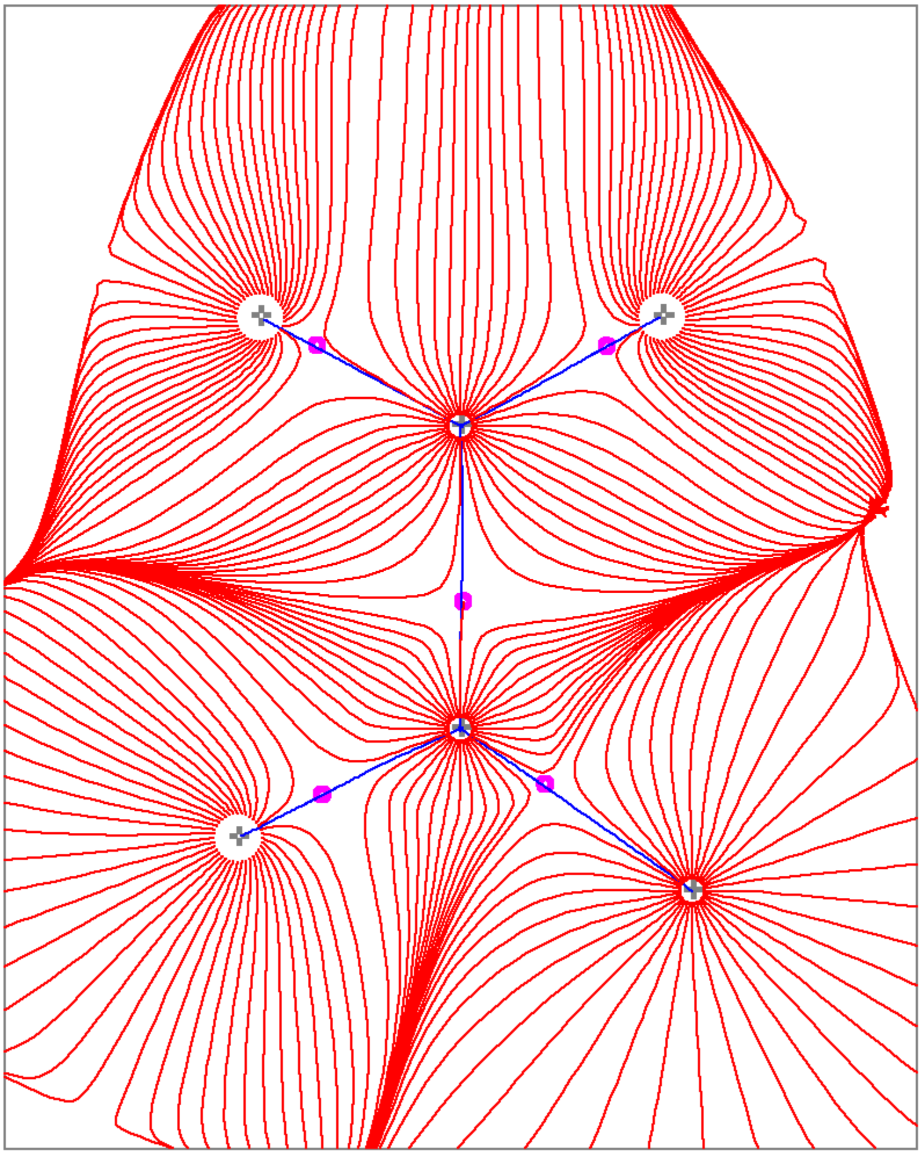
From vector formats the most useful is the PostScript (**–m eps**)

From raster formats the most useful is probably PCX (**–m pcx**) since the image in this format can be easily inserted into Microsoft Word documents without any modifications. The example of a PCX image is shown in Figure 1. The image was created with the following **hp2xx** instruction:

hp2xx `PL2Dform` **-d** 150 **-p** 218282 **-c** 276411 **-f** `form.pcx` **-m** pcx **-r** 270

where `PL2Dform` is the name of the P2DCRY2000 output file and `form.pcx` is the hp2xx output PCX image.

There are many other useful options provided in **hp2xx** program. Please refer to **hp2xx** manual pages for more information.



```
TOPXD rules !!!!!!                                              GradRho Traj
   Xlen (cm)      Ylen(cm)      Xinc(cm)      Yinc(cm)      Scale(Ang/cm)
   12.50          10.00         0.25          0.25          0.40
CNR Center for the Study of Structure/Reactivity Relations - Milano - Italy
```

Figure 1. $\nabla\rho(\mathbf{r})$ trajectories and molecular graph of formamide molecule in the crystal created from TOPXD/P2DCRY2000 data with **hp2xx**.

The $\nabla\rho(\mathbf{r})$ trajectories are shown with read lines, nuclear positions are marked with ✚ (black), BCPs are marked with ○ (**magenta**), bond paths are shown with **blue** lines.

4.6 3-Dimensional (3D) plots

The 3D plot section concerns the evaluation of a number of scalar functions on a 3D grid. The data so obtained may be used for 3D representations of specific envelopes of the scalar function.

The 3D plot instructions have the format, which is quite similar to 2D plots. The general format consists of the following two sections:

1. General instructions
   PL3D general
   ....

2. Specific plot instruction(s)
   PL3D plot ....

The **PL3D general** instructions apply to all the specific plot instructions **PL3D plot** until the next **PL3D general** is given and so on. There is no limit on neither the number of **PL3D general** nor **PL3D plot** instructions.

The TOPXD output 3D files can be processed with **SciAn** visualization package,[13] which is freely available. Please visit **SciAn** WWW home page for more information:

http://www.scri.fsu.edu/~lyons/scian/index.html

The interface between TOPXD 3D plotting section and XDGRAPH is planned for the nearest future…

4.6.1 3D plot general instruction section

This section MUST precede specific plot instructions. All parameters specified in this section will apply to the following specific plots until the next general instruction section is given.

Syntax:

⇒**PL3D general [car | fra]**
⇒ **xmin** *xmin* **xmax** *xmax* **xstep** *xstep*
⇒ **ymin** *ymin* **ymax** *ymax* **ystep** *ystep*
⇒ **zmin** *zmin* **zmax** *zmax* **zstep** *zstep*
⇒ **name** ‘*name*’

Description:

xmin xmax xstep

Minimum and maximum values and grid interval along the crystal ***X***-axis (**fra**) (in fractional units) or cartesian ***X***-axis (**car**) (in Ångstroms).

ymin ymax ystep

Minimum and maximum values and grid interval along the crystal ***Y***-axis (**fra**) (in fractional units) or cartesian ***Y***-axis (**car**) (in Ångstroms).

zmin zmax zstep

Minimum and maximum values and grid interval along the crystal ***Z***-axis (**fra**) (in fractional units) or cartesian ***Z***-axis (**car**) (in Ångstroms).

name ‘*name*’

Character variable *name* (maximum 24 characters, no blanks and no special symbols like ‵,″,^ etc) defines the part of the name of files containing the values of each computed function *func* (see 3D plot Specific Instructions). Full names of the files are listed in Table 2 (‘//’ means character concatenation).

Table 2.

Scalar function type		File name
Electron density	$\rho(\mathbf{r})$	**3DRHOO**//*name*
Laplacian of $\rho(\mathbf{r})$	$\nabla^2\rho(\mathbf{r})$	**3DLAPP**//*name*
Negative of the Laplacian $\rho(\mathbf{r})$	$-\nabla^2\rho(\mathbf{r})$	**3DLAPM**//*name*
Magnitude of the gradient of $\rho(\mathbf{r})$	$\lvert \nabla\rho(\mathbf{r}) \rvert$	**3DGRHO**//*name*

4.6.2 3D plot specific instructions

These instructions must follow the general 3D plot instructions.

Syntax:

⇒**PL3D plot** *func(1)… func(n)*

Note:

There is no limit for **PL3D plot....** instructions and number of *func* instructions in each line, except the 256 character string limitation in the latter case.

Description:

func(1)…func(n) - function type(s). The following function types are recognized:

rhoo	– electron density ($\rho(\mathbf{r})$)
lapp	– laplacian of the electron density ($\nabla^2\rho(\mathbf{r})$)
lapm	– negative laplacian of the electron density ($-\nabla^2\rho(\mathbf{r})$)
grho	– magnitude of the gradient of electron density ($\|\nabla\rho(\mathbf{r})\|$)

5. Description of atomic properties evaluated by TOPXD

Populations		
Atomic population	N	$N(\Omega)=\int_\Omega \rho(\mathbf{r})\, d\tau$
Net charge	Q	$q(\Omega)=Z(\Omega)-N(\Omega)$
Energies		
Atomic Lagrangian (the error in L is a measure of the accuracy of the numerical integration)	L	$L(\Omega)=-1/4\int_\Omega \nabla^2\rho d\tau$
Atomic value of nuclear-electron potential energy with its own nucleus; $\mathbf{R}_\Omega$ is the position vector of Ω in the system frame	VNEO	$VNEO(\Omega)=-\int_\Omega (Z_\Omega/r_\Omega)\rho(\mathbf{r})d\tau$ $\mathbf{r}_\Omega=\mathbf{r}-\mathbf{R}_\Omega$; $r_\Omega=\lvert \mathbf{r}_\Omega \rvert$
Atomic Forces		
Atomic force components Force on nucleus of atom Ω by the electron density of atom Ω	FAXA FAYA FAZA	$FAXA(\Omega)=\int_\Omega (Z_\Omega /r^3_\Omega)x_\Omega\rho(\mathbf{r})d\tau$ $FAYA(\Omega)=\int_\Omega (Z_\Omega /r^3_\Omega)y_\Omega\rho(\mathbf{r})d\tau$ $FAZA(\Omega)=\int_\Omega (Z_\Omega /r^3_\Omega)z_\Omega\rho(\mathbf{r})d\tau$ $x_\Omega,y_\Omega,z_\Omega$, xyz comp. of $\mathbf{r}_\Omega$
"Radial"Atomic expectation values		
Atomic expectation value of the operator r^n_Ω	R(-1) R(+1) R(+2) R(+3) R(+4)	$R^N(\Omega)=\int_\Omega r^n_\Omega\, \rho(\mathbf{r})d\tau$
Atomic expectation values of r^n_Ω averaged over $r_\Omega\bullet\nabla\rho(\mathbf{r})$. It reflects the distortion of the$\nabla\rho(\mathbf{r})$ field of the charge density that is caused by the formation of chemical bond. For n=0 and for a free atom (or a perfectly spherical atom in a molecule or in a crystal) is equal to -3N(Ω) (p.241-242 of Ref.2)	GR(-1) GR(0) GR(1) GR(2)	$GR^N(\Omega)=\int_\Omega r^n\mathbf{r}_\Omega \bullet\nabla\rho(\mathbf{r})d\tau$.
Atomic volumes and related populations		
Volume of the region of the atomic basin where $\rho(\mathbf{r})$ is greater or equal 0.001 au	V001	$V001(\Omega)=\int_\Omega d\tau_{0.001}$ where $d\tau_{0.001}$ are the infinitesimal volume elements where $\rho(\mathbf{r})$ exceeds or is equal to 0.001 au
Electron population in the V001 region	N001	$N001(\Omega)=\int_\Omega \rho(\mathbf{r})d\tau_{0.001}$
Ratio of electron populations in V001 and in the atomic basin	R001	R001=N001/N
Volume of the region of the atomic basin where $\rho(\mathbf{r})$ is greater or equal 0.002 au	V002	$V002(\Omega)=\int_\Omega d\tau_{0.00}$

Electron population in the V002 region	N002	$N002(\Omega)=\int_\Omega \rho(\mathbf{r})d\tau_{0.002}$
Ratio of electron populations in V002 and in the atomic basin	R002	$R002=N002/N$
Total atomic volume	VTOT	$VTOT(\Omega)=\int_\Omega d\tau$
Atomic dipole and quadrupole moments		
Atomic dipole components	DX DY DZ	$DX(\Omega)= -\int_\Omega \rho(\mathbf{r}) x_\Omega d\tau$ $DY(\Omega)= -\int_\Omega \rho(\mathbf{r}) y_\Omega d\tau$ $DZ(\Omega)=-\int_\Omega \rho(\mathbf{r}) z_\Omega d\tau$ $x_\Omega, y_\Omega, z_\Omega$, xyz comp. of $\mathbf{r}_\Omega$
Atomic dipole magnitude	DM	$DM(\Omega)=\lvert \mathbf{DM}(\Omega)\rvert$ where $\mathbf{DM}(\Omega)$ is the atomic dipole vector
Components of atomic displacement vector	DCX DCY DCZ	$DCX= - DX/N(\Omega)$ $DCY= - DY/N(\Omega)$ $DCZ= - DZ/N(\Omega)$
Coordinates of the centroid of negative charge	CX CY CZ	$CX= DCX+X_\Omega$ $CY= DCY+Y_\Omega$ $CZ= DCZ+Z_\Omega$ $X_\Omega, Y_\Omega, Z_\Omega$, xyz components of $\mathbf{R}_\Omega$
Atomic quadrupole moment traceless tensor components	QXX QXY QXZ QYY QYZ QZZ	$QXX(\Omega)= -\int_\Omega [3 x_\Omega^2 - r_\Omega^2]\rho(\mathbf{r})\, d\tau$ $QXY(\Omega)= -\int_\Omega [3 x_\Omega y_\Omega]\rho(\mathbf{r})\, d\tau$ $QXZ(\Omega)= -\int_\Omega [3 x_\Omega z_\Omega]\rho(\mathbf{r})\, d\tau$ $QYY(\Omega)= -\int_\Omega [3 y_\Omega^2 - r_\Omega^2]\rho(\mathbf{r})\, d\tau$ $QYZ(\Omega)= -\int_\Omega [3 y_\Omega z_\Omega]\rho(\mathbf{r})\, d\tau$ $QZZ(\Omega)= -\int_\Omega [3 z_\Omega^2 - r_\Omega^2]\rho(\mathbf{r})\, d\tau$
Eigenvalues and eigenvectors of the atomic quadrupole moment traceless tensor		
Atomic Shannon information entropy		
tomic information (missing information function): is the integral of ρ′lnρ′ where ρ′ is the un-normalized electron density (IUN) or the normalized electron density to unity over the atomic basin (INO) (M. Hô et al. JCP, 108, 5469, 1998).	IUN INO	$I(\Omega)= - \int_\Omega \rho' \ln\rho' d\tau$

6. FREQUENTLY ASKED QUESTIONS

Q: When should/shouldn't I use the Neuton-Raphson (NR)/Eigenvector Following(EF) method?

A: In general NR is only suitable for the location of a critical point if one is already in a region where the Hessian of ρ(**r**) has the correct structure.[7] Thus, NR will fail to find a ring critical point if the starting point has the same eigenvalue signs as a bond critical point. The EF method has proved to be much less sensitive to the choice of good starting search points. The EF method, in practice a NR method with a suitable and locally defined shift for the NR step, seeks for the CPs of a given type, independently on the structure of the Hessian at the starting point. This is particulary important in the case of the $\nabla^2\rho(\mathbf{r})$ field, since this scalar function varies quite rapidly.

Q: When integrating an atomic basin TOPXD gives the following error message:

```
PATHE2: OSCILLATION OF PATHS
PATHE2: THE ATTRACTOR OF THIS PATH WAS PROBABLY NOT INCLUDED IN THE CLUSTER
```

A: Did you check the list of atoms reached into the feeler rays determination step? Do you think that some neighboring atoms were missed? If so, you have to increase parameter *nvi* in order to include the missing atoms into the list of possible attractors of the $\nabla\rho(\mathbf{r})$ trajectories.

The `OSCILLATION OF PATHS` message may also appear in some cases where the integration will be anyhow successful. In many instances it represents just a warning. Especially, if you noticed that the list of neighboring atoms (after the feeler ray step) corresponds to your expectations.

Check the list of atoms reached in the feeler ray step. If needed, increase *nvi* and then start again. Once you have used a very large *nvi* value, leave your calculation to try to end its task (even if the message appear many times).

Q: What grid should be used for integration of atomic basins and how does it affect the computing time ?

A: In order to obtain satisfactory results you should use something like:

64×48×120 (φ×θ×radial) for non-H atoms
32×24×96 (φ×θ×radial) for hydrogen atoms (if not involved in H-bond)
48×32×96 (φ×θ×radial) for hydrogen atoms (if involved in H-bond)

Note that the number of angular points and of radial points refers to the integration within beta sphere and outside beta sphere, respectively.

Computing time is roughly proportional to $n\varphi \times n\theta$. The number of radial points is very important for the precision, but hardly affects the total integration time, as it is operative only in the integration step and NOT in the ZFS determination (which takes about 95% of the total time).

Q: Integration of an atomic basin takes a very long time. What options do we have to speed up the calculation ?

A: Unfortunately the integration step is very very long (especially the ZFS determination which takes about 95% of this time). You can try with the other proposed method, which is much faster but often fails.

Using the indirect method you can save some time by decreasing the accuracy of the surface determination. It is set as a default to 0.001 (see parameter *accur*)

You could try to increase it up to 0.003 (no more than 0.005, I would say). You loose somewhat in precision, but you certainly increase in speed.

You could compare the results of these two computations on one of the atoms you have already integrated, N(Ω), L (Ω), etc. using:

a) first test : 64×48×120 *accur*=0.001
b) second test : 64×48×120 *accu* =0.003

Then you can decide if it is worth varying such a parameter and how much you can vary it.

Q: How do I check the accuracy of the integration ?

A: Check the value of the integrated Lagrangian. For an "exact" integration it should vanish (for the divergence theorem). In practice:

a) it should be less than 5×10^{-5} for H atoms, possibly around 1×10^{-5}. A value of 1×10^{-4} could be perhaps acceptable, but not too precise.

b) for second row atoms (C,N,O, etc,) it should not exceed 1×10^{-3}. Possibly 1×10^{-4}

Q: You've mentioned that the computing time increases by a factor of φ×θ planes, but how does the *nvi* parameter affect the elapsed time?

A: It will affect it, but in a very limited way, especially after the feeler ray step. Indeed the atoms reached during the feeler ray step are put at the top of the list of the *nvi* reachable atoms. So that the DO loop in PATHEN and PATHEN2 (these DO's run on the 3 × *nvi* × STARMULTIPLICITY coordinates of the possible $\nabla\rho(\mathbf{r})$ attractors) are in most cases (>99%) terminated much before the end of the loop.

In practice you shouldn't notice a CPU time increase with *nvi* increase. Rather you could notice a decrease, if you have added an attractor that had to be enclosed. In this case the path oscillation is avoided and CPU time considerably saved.

Q: When trying to increase the number of points to something like 128×96 it TOPXD gives the following error message:
ERROR ** YIELD **** GAUSS QUADRATURE NOT AVAILABLE - NUMBER OF POINTS= 128**

A: Actually the current programs works with these number of points for the quadrature:
1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 36, 40, 48, 64, 96, 120.
So 128 it is not an available option. 120×96 should work.

Q. Sometimes I have problems with integrated Lagrangian, which stays above 1×10^{-3} despite the fact that I use *accur*=0.001 and $n\varphi\times n\theta$ as large as 96×64. I remember that you've mentioned that decreasing the number of points might help, but when I reduce these numbers to 48×32 or 64×48 it still doesn't help. These problems usually occur with carbon and nitrogen atoms, never with oxygens or hydrogens. What do I do ?

A: What about electroneutrality? Are you very far from it ?

The fact that one may get problems with carbon or nitrogen atoms and never with oxygen or hydrogen atoms seems to indicate that the former have more complicate ZFS's than the latter (at least in the systems you are presently investigating). You could try to solve such a problem, by increasing the radius of the beta sphere for such atoms, thus reducing the size of the remaining part of the atomic basin. You could use for the beta sphere something like the distance of the closest BCP multiplied by 1.15 (the program then reduces this number by 20%). Furthermore, the increase (inside the code) of the number of theta and phi points in the inner beta sphere might help. Please contact us and we will send you instructions on how to do it…

7. REFERENCES

[1] Volkov, A.; Gatti, C.; Abramov, Yu.; Coppens, P. *Acta Cryst. A.* **2000**, *56*, 252.

[2] Bader, R. F. W. *Atoms in Molecules: A Quantum Theory;* Claredon Press: Oxford, 1990.

[3] Gatti, C. *TOPOND98 User's Manual*; CNR-CSRSRC, Milano, Italy, 1999.

[4] Saunders, V. R.; Dovesi, R.; Roetti, C.; Causà, M.; Harrison, N.M.; Orlando, R.; Zicovich-Wilson, C. M. *CRYSTAL98 User's Manual;* University of Torino, Italy, 1998.

[5] Koritsanszky, T.; Howard, S.; Su Z.; Mallinson, P. R.; Richter, T.; Hansen, N. K. *XD - A Computer Program Package for Multipole Refinement and Analysis of Electron Densities from Diffraction Data*; Free University of Berlin, Berlin, Germany, 1997.

[6] Baker, J. *J. Comput. Chem.* **1986**, *7*, 385.

[7] Popelier, P. L. A. *Chem. Phys. Lett.* **1994**, *228*, 160.

[8] Aray, Y.; Rodríguez, J.; López-Boada, R. *J. Phys. Chem. A* **1997**, *101*, 2178.

[9] Gerald, C. F.; Wheatley, P. O. *Applied numerical analysis*. Addison-Wesley Pub. Co., 1989.

[10] Cioslowski, J.; Nanayakkara, A. *Chem. Phys. Lett.* **1994**, *219*,151.

[11] Keith, T.A. *Ph.D. thesis*, McMaster University, Ontario, Canada,1993

[12] Biegler-König, F. W.; Bader, R. F. W.; Tang, T.H. *J. Comput. Chem.* **1982**, *13*, 317.

[13] Pepke, E.; Lyons, J. *SciAn.* Supercomputer Computations Research Institute, Florida State University, Talahasseee, Florida.