DelPhi User Manual

version 8.5.0

Produced by: Delphi Development Team

2022

delphi@g.clemson.edu

Table of Contents

ABOUT

This manual describes the main features of the old program, as well as the new features. Whenever possible, we have preserved compatibility with previous versions of DelPhi. People who are used to older versions of DelPhi should not encounter any difficulties in this one. DelPhi is a software package that calculates electrostatic potentials in and around macromolecules or geometrical objects. It can solve the non-linear and linear forms of the Poisson Boltzmann equation using finite difference methods on a $GSZ \times GSZ \times GSZ$ cubical lattice. The user can specify the size of the ion exclusion (or Stern) layer around the molecule and a variable probe radius to define the solvent accessible surface. Different objects and molecules (or a combination of them) can be specified using their own dielectric constant. Various boundary conditions such as periodic and focusing can be used to model different systems like long periodic molecules or cell membranes. The output from the program can be used to calculate molecular interactions, changes in pKa, solvation energies and many other properties of interest.

Authors: Delphi is maintained and developed by Delphi team: email: delphi@clemson.edu

References: The following references should be quoted if the use of the DelPhi v.7.0 and later results to a publication. In particular, reference 1-5 describes some of the new features introduced since DelPhi v.7.0 and carried into v8.0; references 6 and 7 describe the implementations of parallel computing since DelPhi V.7.0.

1. 1. Z. Jia, L. Li, A. Chakravorty, and E. Alexov, 'Treating ion distribution with Gaussian-based smooth dielectric function in DelPhi', J Comput Chem. 2017 August 15; 38(22): 19741979

2. A. Chakravorty, Z. Jia, L. Li, and E. Alexov, 'A New DelPhi Feature for Modeling Electrostatic Potential around Proteins: Role of Bound Ions and Implications for Zeta-Potential', Langmuir, (2017) 3:9, 2283-2295

3. L. Li, C. Li, S. Sarkar, J. Zhang, S. Witham, Z. Zhang, L. Wang, N. Smith, M. Petukh, E. Alexov, 'DelPhi: a comprehensive suite for DelPhi software and associated resources', BMC, Biophys, (2012) May 14; 4(1): 9.

4. Smith N, Witham S, Sarkar S, Zhang J, Li L, Li C, Alexov E. 'DelPhi Web Server v2: Incorporating atomic- style geometrical figures into the computational protocol', Bioinformatics. 2012 Apr 23.

5. L. Li, C. Li, Z. Zhang, E. Alexov, 'On the Dielectric Constant of Proteins: Smooth Dielectric Function for Macromolecular Modeling and its Implementation in DelPhi', J. Chem, Theory Comput. 2013 Apr 9; 9(4): 2126-2136.

6. C. Li, L. Li, J. Zhang, E. Alexov, 'Highly efficient and exact method for parallelization of grid-based algorithms and its implementation in DelPhi', J. Comput Chem, 2012 Sep 15: 33(24): 1960-1966.

7. C. Li, M. Petukh, L. Li, E. Alexov, 'Continuous Development of Schemes for Parallel Computing of the Electrostatics in Biological Systems: Implementation in DelPhi', J comput chem (2013), Article first published online: 3 JUN 2013 DOI: 10.1002/jcc.23340.

8. Rocchia, W.; Alexov, E.; Honig, B. 'Extending the applicability of the nonlinear Poisson-Boltzmann equation: Multiple dielectric constants and multivalent ions'. J Phys. Chem. B 105, 6507-6514 (2001) (pdf)

9. W. Rocchia, S. Sridharan, A. Nicholls, E. Alexov, A. Chiabrera and B. Honig 'Rapid Grid-based Construction of the Molecular Surface for both Molecules and Geometric Objects: Applications to the Finite Difference Poisson-Boltzmann Method'. J. Comp. Chem. 23, 128-137,2002

Additional references:

10. Klapper, I., Hagstrom, R., Fine, R., Sharp, K., Honig, B. (1986). 'Focusing of electric fields in the active site of Cu-Zn Superoxide Dismutase: Effects of ionic strength and amino-acid modification'. Proteins 1, p 47.

11. K.A. Sharp, M.K.Gilson, R.M.Fine and B.H. Honig. (1987). 'Electrostatic interactions in proteins'. UCLA Symposium on Molecular and Cell Biology, Vol 69: Protein Structure, Folding and Design, Ed. D.L. Oxender, p235.

12. Gilson, M., Sharp, K., Honig, B. 'Calculating electrostatic interactions in bio-molecules: Method and error assessment'. J. Computational Chem. 9, pp327-335.

13. Gilson, M., Honig, B. Total 'Electrostatic Energy of a Protein'. Proteins, 4, p7 (1988).

14. B. Jayaram, K.A.Sharp and B.H.Honig. 'The electrostatic potential of B-DNA'. Biopolymers, 28, p975 (1989).

15. K. Sharp, and B. Honig. 'Lattice Models of Electrostatic Interactions: The Finite Difference Poisson- Boltzmann Method'. Chemica Scripta, 29A: 71 (1989)

16. K. Sharp, and B. Honig. 'Electrostatic Interactions in Macromolecules: Theory and Applications'. Ann. Rev. Biophys. Chem. 19:301-32 (1990).

The original reference to the use of the finite difference method for macromolecular electrostatics is:

17. J. Warwicker and H.C. Watson, J. Mol. Biol., 157, p671 (1982).

INTRODUCTION

DelPhi takes as input a Brookhaven database coordinate file format of a molecule or equivalent data for geometrical objects and/or charge distributions and calculates the electrostatic potential in and around the system, using a finite difference solution to the Poisson-Boltzmann equation. This can be done for any given concentration of any two different salts. The system can be composed of different parts having different dielectrics.

2.1 Installation

DelPhi v.8.5.0 is distributed in four versions:

- Linux version, compiled with GCC compiler with version 5.4.0 and higher.
- Mac version, compiled with GCC (5.4.0+) or CLANG (7.3.0+)

Their way of working is very similar; however, unexpected differences may appear due to different numerical precision or to the porting of the software to different architectures. Each distribution contains one executable, the source codes with corresponding makefile when needed, and some worked examples. With parallelized versions now available for Delphi that can run on Linux, separate executable for OMP and MPI can also be generated using the source code.

2.2 Basic Tutorial

This section provides an overview of using DelPhi in an energy calculation. A quick introduction is given in the first section, and then details are given in the sections that follow. Briefly, running DelPhi consists of the following steps:

1. Prepare run parameters file (named fort.10 (fortran only) or [namefile].prm)
2. If at least one molecule is going to be introduced, then prepare at least three more files containing:

File Type	Default Convention	Alternative Convention
Atom coordinates	fort.13(fortran only)	`[namefile].pdb`
Atomic Radii	fort.11(fortran only)	`[namefile].siz`
Atomic Charges	fort.12(fortran only)	`[namefile].crg`

Several sets of sample parameter files are provided with the distribution, so it is not necessary to generate them from scratch. These include PARSE, CHARMM and Amber charge and radii files. They all are developed for Brookhaven protein databank (pdb) files with hydrogen. Thus, for successful modeling, the input pdb file should be protonated

prior to running DelPhi. If the accuracy of the calculations is not crucial, then using the unprotonated pdb file is possible, using the proper charge (crg) and radii (siz) files. Importantly, the names of the atoms and residues should be consistent between the pdb, crg and siz files. In the simplest case, DelPhi is applied to a single molecule in a pdb file. To do this, the pdb file can be renamed to fort.13 (fortran only) in the directory where DelPhi will be run, or the following line can be added to the run parameters (prm) file:

```
in(pdb,file=''[namefile].pdb'')
```

Likewise, the crg, siz, and other parameter files should be set up in a similar manner. This will be discussed in more detail in Statement and Functions. After the input and the parameter files have been properly set up, DelPhi can be run from the command line. For instance if one wants to use the parameter file `test.prm` as the parameter file one types:

```
delphicpp_release test.prm
```

Typing only:

```
delphicpp_release
```

Flashes out the Delphi header indicating that the software is correctly installed. Run the program DELPHI in batch or interactively directing the output to unit 6 (standard output) or into a log file, if necessary. Example:

```
delphicpp_release test.prm > out.log
```

Analyze the results. The primary output file from the program is a three dimensional array of potentials calculated at the lattice points. This is a large file $[(gsize)^3]$ and is written in binary to save space and time. Much more information from the run can be extracted and saved in suitable files. Delphi prints out the grid energy, reaction field energy and coulombic interaction energy. These energies can be used for variety of biophysical applications.

As an option, the site coordinates (FRC) file can be provided in order to collect the potential and electrostatic field components at specific positions. It has the same format as a pdb file. The calculated potential and the components of electrostatic field will be reported at the positions of atoms given in the frc file. With the ability to supply FRC input with charges at the sites mentioned therein, it can also be used to compute the interaction energy of the source and the target (FRC object) or the electrostatic potential energy of the target in the field of the source.

The most important file is the parameter file. It contains the parameters that control the run and output files. Lines within the parameter file can be either Statement or Functions.

This manual will describe meaning and structure of the Statements and Functions, together with a description of input/output file naming and format, energy calculation, and a description of the new features available in DelPhi v.8.0. we also offer various advice on choosing parameters and using DelPhi.

Note: Older versions of the program provided utilities for file format conversion together with specific flags for output format. Unfortunately, not all these options have been tested

and updated in the new version. However, most of them are expected to work properly, at least if the input is a single molecule with only one dielectric constant.

2.3 Overall Program Flow

1. Header with time and date is written.
2. Parameters are read from `fort.10` (fortran only) or prm ftle and echoed to output.
3. Radius data read from `fort.11` (fortran only) or siz ftle and stored in hash table for efficient look up.
4. Charge data read from `fort.12` (fortran only) or crg ftle and stored in hash table for efficient look up.
5. Atomic coordinates are read from `fort.13` (fortran only) or pdb ftle and scaling is computed. In accordance with charge and size files, radius and charge are assigned to each atom. Distribution of dielectric values, ionic strength parameters and charge values over the lattice are determined from the coordinate/charge/radius data files. Using a PQR ftle for the structure obviates the need of specifying the `crg` and `siz` files separately.
6. Arrays that describe 3D distribution of dielectric and ion accessibility in space are initialized.
7. Atom file with charge and radii records are outputted to `fort.19` (fortran only) if requested or to a PQR formatted file.
8. Centers of + and − charge distributions, and net charge calculated for check on charge distribution.
9. Arrays are set up for the difference eqn. iteration.
10. Boundary values are set, either through analytical expressions or interpolated from the potential map read from `fort.18` (fortran only) or CUBE format file.
11. Linear then non-linear iterative relaxations are done and convergence histories are printed out as simple log/lin line plots, if requested.
12. Potentials are converted to concentrations if requested.
13. Potentials and fields are calculated at the coordinates of the atoms read from `fort.15` (fortran only) or the FRC input ftle, and outputted to `fort.16` (fortran only) or FRC output ftle if requested.
14. Grid of potentials outputted to `fort.14` (fortran only) or a CUBE format ftle, if requested.
15. Energy contributions and overall surface induced polarization charge are printed out.
16. The dielectric map is outputted to `fort.17` (fortran only) or a CUBE format ftle, if requested.

2.4 Statements and Functions

DelPhi uses a command interpreter that allows commands to be used in the parameter file. The concept of the command in DelPhi comes in two forms, statements and functions. Statements have the form:

```
variable-name=value
```

e.g.

```
scale=2.0
gridsize=65
pbx=t
```

Functions have the form:

```
operation(specifier, file="xxx.yyy", format="abc")
```

e.g.

```
in(pdb,file="lys.pdb")
out(phi, unit=20, format=2)
center(file="test.pdb")
```

Statements simply set values or flags. Functions tell DelPhi to perform an immediate operation using the specifiers in the function as parameters for the operation.

2.5 Syntax

In general, statements and functions can each be placed on a new line. Since this is the clearest way to organize statements, functions and comments, this is what we would recommend. However, several statements and functions can be placed on the same line, separated by commas ",", vertical bars "—" or colons ":". Comments can also be included on the same line as functions or statements. These are set apart by surrounding them with a pair of exclamation points "!". If a comment extends to the end of the line, then only a single exclamation point "!" is necessary. Spaces and capitalization are ignored only if they appear outside of quotation marks. A very long line can be split into two lines using the backslash "\". This is illustrated in the following examples:

```
scale=2.0, gridsize=65, center (file="mid.pdb")
in(pdb,file="lys.pdb") !this is a comment at the end of a line
in(pdb,file="Lys.pdb") !this line reads the file Lys.pdb, not lys.pdb
scale=1.5 !this is a comment surrounded by two statements! proberadius
    =1.4
```

Note that in the last example, both scale and probe radius will be set by DelPhi. Please be careful with the slightly unconventional use of the comment. We have tried to anticipate some input errors and to inform the user of them, but the hardest part of every complex program is error handling. At the moment DelPhi will only pipe back to you what it doesn't

understand and continue on with the program. If DelPhi does not understand a command, it will attempt to use a default value and continue running anyway. Therefore, it would be worthwhile to pay attention to syntax to avoid running an unintended calculation.

2.6 Shorthand and Longhand Statements

Many statements have abbreviated names. These may come in various forms from three to six letters long. Although longer descriptions are easier to read, the shorter forms are easier to type and as such they may be less prone to typing error. They are a matter of taste. A complete listing of abbreviations and full names appears in the Index of Statements. Yes, No, Maybe: When setting logical values the following are case insensitive and equivalent:

```
yes, on
true, t
no, off
false,f
```

Return to TOC

FUNCTIONS IN DETAIL

The present set of allowed functions is:

```
CENTER
ACENTER
READ/IN (Equivalent)
WRITE/OUT (Equivalent)
ENERGY
BUFFZ
SITE
QINCLUDE
```

We shall cover these one by one since they vary somewhat more in format than statements. But first, some of the common features:

```
function (file="test.file")
```

Will open the file `test.file`, whether for centering, output or input.

```
Function(unit=14)
```

Will do the same but with `fort.14` or whatever is linked to it.

```
Function(format=abc)
```

Will perform operations on files with a particular format, or in a specified way. The default format is always zero (i.e. "0"). The format can be a number or a string. Users are advised not to change the format and to use the default settings.

3.1 Center

```
Center(0.2,3,2)
```

Will offset the molecule by 0.2 grid units in the $x-$, 3 in the $y-$ and 2 in the $z-$directions, respectively. Center was created as a function to allow the following possibility:

```
Center(unit=15)
```

This opens `fort.15` (usually called frc file), reads its atoms and centers the current calculation using the geometrical center of the atoms in the file. An alias for opening `fort.15` and take is center as the system center is:

```
Center(999,0,0).
```

To read just the first atom of a file and use its coordinates use the following,

```
Center(file="whatever",an =1)
Center(999,999,0)
```

is an equivalent of

```
Center(unit=15,an=1)
```

Note that

```
an=1
```

is a string and that

```
an=n
```

is not going to take the nth atom position as the center. Other aliases are: `Center(777,0,0)` for `Center(unit=27)` and `Center(777,777,0)` for `Center(unit=27, an=1)`.

This function is used to specify the offset (expressed in grid units) with respect to the lattice center at which the center of the molecule [pmid(3)] is placed. This will influence what point in the real space (expressed in Angstroms) is placed at the center of the grid [oldmid(3)]. The relationship between real space r(i) and grid g(i) coordinates for a grid size of igrid, with a scale of gpa grids/angstrom is as follows: The centre of the grid is:

$$midg = (igrid + 1)/2 \tag{1}$$

$$oldmid(i) = pmid(i) - OFFSET(i)/gpa \tag{2}$$

$$g(i) = (r(i) - pmid(i)) * gpa + midg + OFFSET(i) \tag{3}$$

$$r(i) = (g(i) - midg)/gpa + oldmid(3) \tag{4}$$

The scale, the system center and the shift are printed in the logfile.

Note that a certain error inevitably results from the mapping of the molecule onto the grid. By moving the molecule slightly (changing CENTER offset between 0,0,0 and 1,1,1) and repeating the calculations, it is possible to see whether the results are sensitive to the particular position on the grid, and if so, to improve the accuracy by averaging (this is related to rotational averaging, discussed in the J. Comp Chem paper of Gilson et al.). However using a larger scale is a more effective way of improving accuracy than averaging.

3.2 Acenter

Acenter takes three absolute coordinates, i.e. in Å and uses those as the center, so:

```
Acenter(1.0,5.6,7.0)
```

centers the grid box at x=1.0 Å, y=5.6 Å, z=7.0 Å

3.3 Read/In

This function allows files to be read as input. It comes with several specifiers, namely:

```
SIZ: for the radius file
CRG: for the charge file
```

```
SGG: for multiple sigma values similar to charge file format
TOPOL: topology file if DelPhi is to be run for every frame of
       a trajectory
TRAJ: trajectory file for trajectory processing
PDB: for the pdb structure file
    (possible alternative formats: frm=UN and frm=MOD)
MODPDB4: for the modified pdb structure file
    (possible alternative formats: frm=PQR and frm=MOD) which contain
    charges and radii values with 4 digits precision after decimal
    points.
FRC: for positions of site potentials.
    (possible alternative format for C++ version only: frm=PQR)
PHI: for the phimap used in focusing (CUBE format)
```

The main use, at present will be to give the user flexibility to specify the file name or unit number of any of these files. Note that the default files for all read (and write) operations are the standard DelPhi files.

Example:

```
in(modpdb4, file="test.mod", format="mod")
```

Read a mod file called test.mod, which contains charge and radius value in 4 digits after decimal

```
in(modpdb4, file="test.pqr",format="pqr")
```

Read a `pqr` file called `test.pqr`, which contains charge and radius value in 4 digits after decimal. Using this option, DelPhi can directly read PQR files which are generated by other programs (such like pdb2pqr program).

```
in(frc,file="namefile")
```

Opens the file namefile and logically assigns to it the unit 15 (see Files for details). For the C++ version of Delphi, a new feature allows writing FRC input file in the PQR format. Besides the coordinates of the sites where certain quantities are requested, one can include the charge and radius to indicate that an atom could be present at that site. The radius column is ignored and the charge, along with the potential at a coordinate, is used to compute the potential energy of the FRC object in the field of the source molecule.

```
in(frc,file="namefile", format=PQR)
```

3.4 Write/Out

Equally obviously this deals with output. The specifiers are:

```
PHI : for phimaps
    (possible other formats: frm=BIOSYM, frm=GRASP, frm=CUBE;
    see Unit14 in Files)
FRC : for site potentials
    (possible other formats: frm=RC, frm=R, frm=UN;)
```

```
EPS : for epsmaps
MODPDB: for modified pdb files
MODPDB4: modified pdb files that contain charges and radii with
    4 digits precession after decimal points
UNPDB: for unformatted pdb file UNFRC : for unformatted frc files
ENERGY: writes the file "energy.dat" containing energy data.
(Example: out(energy))
-Note that this is different from the Energy function!

ZPHI: writes the file ‘‘surfacePot.zphi’’ containing the values of
    the potentials on an exterior 2D-
```

As an example of use,

```
out(eps,file=‘‘epsmap.txt’’)
```

Writes an epsmap with cube format, which can be visualized by softwares such as Chimera and VMD.

```
out(modpdb, file=‘‘test.out’’)
```

Writes a modified pdb file called test.out

```
out(modpdb4, file=‘‘test.mod’’,format=mod)
```

Writes a modified pdb file called test.mod, which contains charge and radius value in 4 digits after decimal

```
out(modpdb4, file=‘‘test.pqr’’,format=pqr)
```

Writes a pqr file called test.pqr, which contains charge and radius value in 4 digits after decimal.

3.5 Energy

At present it takes as its argument any of the following:

```
G or GRID for the grid energy,
S or SOL or SOLVATION for the corrected reaction field energy
C or COULOMBIC or COU for the columbic energy
ION or IONIC or IONIC\_C for the direct ionic contribution
    (see Ionic direct Contribution)
```

Separated by commas. (As always there is no case sensitivity here.) So, for example,

```
energy(s,g,Cou,ion,n,lj)
```

Gives the solvation, coulombic, grid energies and ionic contribution.

Note that the calculations of the non-linear contributions are automatically turned on whenever non-linear PBE solver is invoked.

For the energy definition we recommend the Rocchia et al. J. Phys. Chem, however a brief explanation is given below:

The **grid energy** is obtained from the product of the potential at each point on the grid and the charge at that point, summed over all points on the grid. However, the potential computed for each charge on the grid includes not only the potentials induced by all other charges, but also the "self" potential. The partitioning of the real charges into the grid points causes the effect. Thus, two neighboring grid points might have partial charges that originate from the same real charge. Since the product of a charge with its own potential is not a true physical quantity, the grid energy should not be taken as a physically meaningful number by itself. Instead, the grid energy is only meaningful when comparing two DelPhi runs with exactly the same grid conditions (e.g. constant structure and constant scale). The difference can then be used to extract solvation energies, salt effects, and others.

The **coulombic energy** is calculated using Coulomb's law. It is defined, as the energy required bringing charges from infinite distance to their resting positions within the dielectric specified for the molecule. This term has been revised in the new DelPhi to be consistent with the new multiple dielectric model. For the most recent definition, we again refer the reader to the previously mentioned paper.

The **reaction field energy** (also called the solvation energy) is obtained from the product of the potential due to induced surface charges with all fixed charges of the solute molecule. This includes any fixed charge in the molecule that happens to be outside of the grid box. The induced surface charges are calculated at each point on the boundary between two dielectrics, e.g. the surface of the molecule. If the entire molecule lies within the box and salt is absent, this energy is the energy of transferring the molecule from a medium equal to the interior dielectric of the molecule into a medium of external dielectric of the solution. Depending on the physical process being described, this may be the actual solvation energy, but in general the solvation energy is obtained by taking the difference in reaction field energies between suitable reference states - hence we make the distinction between this physical process and our calculated energy term. For other Energy contributions, see here.

The **non-polar** component of the solvation energy is computed proportional to the surface area of the system.

The **Lennard-Jones or van der Waals** energy is also computed for all the atom-atom pairs and sum is reported. This options requires the vdw file to be provided with the input.

3.6 Site

```
SITE(argument)
```

Reports the potentials and electrostatic field components at the positions of the subset of atoms specified in the frc file. The atoms specified in frc file should not be charged in the Delphi run. The argument is a list of identifiers that can be:

```
Atom or A
```

```
Charge or Q
Potential or P
Field or F
Reaction or R
Coulomb or C
Coordinates or X
Salt or I
```

Total or T Examples:

```
SITE(atom,potentials)
Site(a,p) ! specifies what printed to frc file (see above).
```

3.7 Buffz

Defines a box with sides parallel to grid unit vectors that the reaction field energy will then be calculated using ONLY the polarization charges contained in that box. The fixed format is BUFFZ(6i3).

Example:

```
BUFFZ(001002003004005006) will fill a matrix:
Bufz(1,1)=1 distance in grid units from the negative x side
Bufz(2,1)=2 distance in grid units from the negative y side
Bufz(3,1)=3 distance in grid units from the negative z side
Bufz(1,2)=4 distance in grid units from the positive x side
Bufz(2,2)=5 distance in grid units from the positive y side
Bufz(3,2)=6 distance in grid units from the positive z side
```

3.8 Qinclude

The qinclude function is a feature that has not been tested in the latest versions of DelPhi, so it may behave a bit differently than expected. It works in the same way as an include statement works in FORTRAN or C, i.e., it inserts lines from another file into the current one. For instance, suppose we have the following files:

test.prm:

```
scale=3.0, write(frc),
write(modpdb,file="test.out")
acenter(0.123,4.55,2.34)
```

test2.prm:

```
Boundary type=2,
read(pdb,file="test.pdb")
```

Then the file:

```
scale=3.0,
write(frc),
write(modpdb,file="test.out")
qinclude(test2.prm)
acenter(0.123,4.55,2.34)
```

Is equivalent to:

```
scale=3.0,
write(frc),
write(modpdb,file="test.out")
boundary type=2,
read(pdb,file="test.pdb")
acenter(0.123,4.55,2.34)
```

Or one could even write:

```
qinclude(test1.prm)
qinclude(test2.prm)
```

Clearly the motivation behind this form is to allow the user to create his/her own default file and qinclude this file at the beginning of subsequent parameter file. One then needs only a qinclude statement plus and lines indicating those parameters that need to be changed from the default file.

Note that `qinclude` is immediate, i.e. it includes the lines from the indicated file at the position of the `qinclude` command. This is important to remember that if you define a quantity multiple times, then only the last instance is used. In other words, a file containing

```
scale=2.0
scale=3.0
```

tells DelPhi to set the scale to 3 grids/Å. This is the reason we include a `write(specifter, off)` command. If you have a default file, which enables a write, you can still turn it off without modifying the default file.

Can a `qinclude` file contain a `qinclude` file? But of course, at present one can nest `qinclude` files up to ten deep. If a `qinclude` file does not exist DelPhi will tell you so and move on to the next command. If there is no file passed to `qinclude`, i.e. `qinclude()` then, if it exists, the default include file `~/qpref.prm` is passed. `Qinclude` is a special command and as such always requires its own line, i.e. do NOT add more commands to a line, which start with a `qinclude` command (not even comments).

INSOBJ(Removed and OBJECTS are no longer supported! Instead users are suggested to use Protein Nano Object Integrator ProNOI to create, visualize and manipulate atomistic-style objects and use them in conjunction with standard Protein Data Bank files.)

This function is somehow different from the others in the sense that it doesn't have any argument, if it is written in a line of a prm file, it launches the routine that allows the user to insert objects, charge distributions etc. (see description)

Return to TOC

STATEMENTS IN DETAIL

The following provides you with the preamble/formatting for lessons and exams as well as how to organize your code and document.

4.1 Index of Statements and their shorthand

Here we provide the table of parameter statements in DelPhi with their long form, short form, two character abbreviation, and default value.

Table 1: Table of Parameter Statements in DelPhi.

Statement Long form	**Short form**	**2 Char Abbr**	**Default**
AUTOCON	AUTOC	AC	TRUE
AUTOCONVERGENCE			
AUTOMATICCONVERGENCE			
ATOMPOTDIST	ATPODS		0.5
BOUNDARYCONDITION	BNDCON	BC	2 (=Dipolar)
BOUNDARYCONDITIONS			
PERCENTFILL	PERFIL	PF	80
BOXFILL			
PERCENTBOXFILL			
CHEBIT	CHEBIT	CI	FALSE
CLCSRF	CLCSRF	CS	FALSE
CONVERGENCEFRACTION	CONFRA	CF	1
CONVERGENCEINTERVAL	CONINT	CI	10
CUTOFF			0
DENCUT	DENCUT	DC	-1.0
EXDI2	EXDI2	E2	1.00
EXITUNIFORMDIELECTRIC	EXITUN	XU	FALSE
EXTERNALDIELECTRIC	EXDI	ED	80
FANCYCHARGE	FCRG	FC	FALSE
SPHERICALCHARGEDISTRIBUTION			
FRAMEFIRST			1
FRAMELAST			1
FRAMESTRIDE			1
GRIDCONVERGENCE	GRDCON	GC	0.0
GRIDSIZE	GSIZE	GS	AUTOMATIC
GAUSSIAN	GAUSSIAN	GN	0
GEXPMULTIPLIER		GM	1
GAPDI			80
INTERIORDIELECTRIC	INDI	ID	2.0

Continued on next page

Table 1 – continued from previous page

Statement Long form	Short form	2 Char Abbr	Default
SALTCONC	SALT	IS	0.0
IONICSTRENGTH			
SALTCONCENTRATION			
IONRADIUS	IONRAD	IR	0.0/2.0
LINEARITERATION	LINIT	LI	AUTOMATIC
ITERATION			
ITERATIONS			
LOGFILECONVERGENCE	LOGGRP	LG	FALSE
LOGFILEPOTENTIALS	LOGPOT	LP	FALSE
MAXC	MAXC	XC	0
MEMBRANEDATA	NOT	MD	FALSE
	USED		
MAXWARN (new!)	MAXWARN	MW	INT MAX
MULTISIGMAGAUSSIAN	MSIGMAG	MS	FALSE
NONLINEARITERATION	NONIT	NI	0
NONLINEARITERATIONS			
PERIODICBOUNDARYX	PBX	PX	FALSE
PERIODICBOUNDARYY	PBY	PY	FALSE
PERIODICBOUNDARYZ	PBZ	PZ	FALSE
PHICON	PHICON		FALSE
PRESSURECOEFF			1.0
PROBERADIUS	PRBRAD	PR	1.4
RADIPZ	RADIPZ	RZ	-1.0
RADPOLEXT	RADPOLEXT	RL	1.0
RADPR2	RADPR2	R2	PRBRAD
RELAXATIONFACTOR	RELFAC	RF	0.9975
RELPAR	RELPAR	RR	1.0
RMSC	RMSC	MC	0.0
SALT2	SALT2	S2	0.0
SCALE	SCALE	SC	1.2
SOLVPB	SOLVPB	SP	TRUE
SIGMA	SIGMA	SG	1.00
SRFCUT	SRFCUT	SF	20.0
SURFACEPOTENTIAL (new!)	SURFPOT	SU	0
SURFACEDISTANCE (new!)	SURFDIST	SD	0.0
TEMPERATURE	TEMPER		297.3342119 K
VAL+1 and similar	VAL+1	+1	1

Return to TOC

4.2 Full list and Statement description

4.2.1 MULTISIGMAGAUSSIAN

A flag, which controls the use of atom/residue dependent multi sigma for gaussian. Normally DelPhi will use single-sigma-gaussian model (Default: False), unless user wishes to use multi-sigma-gausian. If this is set to True, user must proved a the sgg file also. The sgg file is a crg format file where charge column is replaced by the sigma-gaussian value.

4.2.2 GAPDI

The gap-dielectric (protein) is a maximum dielectric constant inside protein. This value is used only in case of Gaussian. The default value is same as the external dielectric value.

4.2.3 GSIZE

An odd integer number of points per side of the cubic lattice, min=5, max=571 (=NGRID, platform dependent). A larger grid size will in general mean a better resolution representation of the molecule on the lattice. This will result in more accurate potentials, but will require more time. The number of iterations required to reach a certain convergence will increase approximately linearly with parameter GS. Since the time per iteration will go up as the cube of this parameter the amount of calculation will thus increase at about the fourth power of GS. Example.

```
gsize=65 or gs=65
```

4.2.4 SCALE

The reciprocal of one grid spacing (grids/Å). Example:

```
scale=1.2 or sc=1.2.
```

List of Parameters

4.2.5 PERFIL

A percentage of the object longest linear dimension to the lattice linear dimension. This will affect the scale of the lattice (grids/Å). The percentage fill of the lattice will depend on the application. A large percentage fill will provide a more detailed mapping of the molecular shape onto the lattice. A PERFIL less than 20% is not usually necessary or advisable. A very large filling will bring the dielectric boundary of the molecule closer to the lattice edge. This will cause larger errors arising from the boundary potential estimates,

which are set to zero or approximated by columbic/Debye-Huckel-type functions using a uniform solvent dielectric. The error will be minimal for higher salt concentrations or weakly charged molecules. Smaller percentages will increase the accuracy of the boundary conditions, but result in a coarser representation of the molecule. Higher resolution can be achieved more efficiently using focusing. Example

```
perfil=40 or pf=40.
```

NOTES: If the molecule is not centered in the origin of the coordinate system, the PERFIL reflects the percentage of the system that is actually contained in the lattice. For example, if the maximum dimension of a molecule is 100Å, there is no offset and PERFIL is 50%, then the box side will be 200 Å; but if there is an offset of 20Å in the maximum dimension direction, then the box side will be 280Å.

List of Parameters

4.2.6 SCALE

, **GSIZE** and **PERFIL** are not independent variables so they cannot all be assigned simultaneously in a single run. In any quantitative calculation, the largest possible scale should be used, preferably greater than 2 grids/Å. Without focusing a **PERFIL** of around 50% or 60% is reasonable. For example if scale is set to 2 and PERFIL is set to 50%, the grid size is calculated automatically given the size of the structure. For larger molecules this could mean a prohibitively large memory requirement. In this case a compromise must be found or focusing could be used. Regardless of grid scale, calculations should be repeated at different scales to assess the size of lattice resolution errors. A good approach to the calculation could start with a small percentage, say 20%, using Debye-Huckel boundary conditions, and then focus in to say 90% or more, in one (or two) stages, using focusing boundary conditions for the second (and third) runs. It is not necessary for the molecule to lie completely within the grid although then focusing must generate the potential boundary conditions. However, when calculating solvation energies with box fills of ¿ 100% remember that unexpected results may be obtained since parts of the surface, (and perhaps some charges) are not included in the grid.

List of Parameters

4.2.7 INDI

The internal (molecules) dielectric constant. It is used only in single molecule systems for compatibility with the old version. A value of `INDI=1` corresponds to a molecule with no polarizability the state assumed in most molecular mechanics applications. `INDI=2` represents a molecule with only electronic polarizability (i.e. assuming no reorientation of fixed dipoles, peptide bonds, etc). A value of 2 is based on the experimentally observed high frequency dielectric behavior of essentially all organic materials. INDI=4-6 represents a process where some small reorganization of molecular dipoles occurs which is not represented

explicitly (for example in modeling the effects of site directed mutagenesis experiments, when the structure of the wild type, but not mutant protein is known). According to M.K. Gilson and B. Honig, Biopolymers, 25:2097 (1986) for instance, materials having similar dipole density, dipole moment and flexibility as globular proteins have a dielectric between 4 and 6. In modeling any process where large reorientations of dipoles, or large conformational change occurs, i.e. upon folding or denaturation, using a simple dielectric constant for the molecule would be inappropriate, and the change in conformation should be modeled explicitly. Example:

```
indi=2 or id=2.
```

List of Parameters

4.2.8 EXDI

The external (solution) dielectric constant. A value of EXDI=1 corresponds to the molecule in vacuum, EXDI=80 to the molecule in water. Depending on the application runs with EXDI equal to either of these values may be used to represent different states in a thermodynamic cycle. Example:

```
exdi=80 or ed=80.
```

List of Parameters

4.2.9 PRBRAD

A radius (Å) of probe molecule that will define solvent accessible surface in the Lee and Richard's sense. In combination with the atomic van der Waals radii in the siz file, PRBRAD determines the regions of space, and hence the lattice points, that are inaccessible to solvent molecules (water). Suggested value is PRBRAD=1.4 for water. To understand how these parameters work, you should be familiar with the concepts of contact and solvent accessible surface, as discussed by Lee and Richards, and by Mike Connolly. For the purpose of DelPhi, any region of space that is accessible to any part of a solvent (water) molecule is considered as having a dielectric of EXDI. A value of zero for PRBRAD used with a siz file containing the standard van der Waals radii values will assign any region of space not inside any atom's van der Waals sphere to the solvent. For more details, please refer to Rocchia et al. J. Comp. Chem. paper. Example:

```
prbrad = 1.4 ! for water
```

List of Parameters

4.2.10 IONRAD

The thickness of the ion exclusion layer around molecule (Å). IONRAD, in combination with the atomic van der Waals radii in the siz file, determines the regions of space, and hence the lattice points, which are inaccessible to solvent ions. Suggested values is IONRAD = 2.0 for sodium chloride. For the purpose of DelPhi, a solvent ion is considered as a point charge, which can approach no closer than its ionic radius, IONRAD, to any atoms van der Waals surface. The ion excluded volume is thus bounded by the contact surface, which is the locus of the ion center when in van der Waals contact with any accessible atom of the molecule. A zero value for IONRAD will just yield the van der Waals surface. A non zero value of IONRAD will thus introduce a Stern, or ion exclusion layer, around the molecule where the solvent ion concentration will be zero and whose dielectric constant is that of the solvent, EXDI. Example:

```
ionrad=2 or ir=2.
```

List of Parameters

4.2.11 SALT

The concentration of first kind of salt, (moles/liter). In the case of a single 1:1 salt, it coincides with ionic strength. Example:

```
salt=0.14 or is=0.14.
```

List of Parameters

4.2.12 BNDCON

An integer flag specifying the type of boundary condition imposed on the edge of the lattice. Example:

```
bndcond=4 or bc=4.
```

Allowed options for BNDCON:

(1) - Potential is zero.

(2) - Dipolar. The boundary potentials are approximated by the Debye-Huckel potential of the equivalent dipole to the molecular charge distribution. Φ is the potential estimated at a given lattice boundary point, q^+ (q^-) is the sum of all positive (negative) charges, and r^+ (r^-) is distance from the point to the center of positive (negative) charge, lambda is the Debye length.

(3) - Focusing. The potential map from a previous calculation is read in unit 18, and values for the potential at the lattice edge are interpolated from this map- clearly the

first map should have been generated with a coarser grid (greater distance between lattice points) and positioned such that current lattice lies completely within old lattice or the program will protest. For focusing boundary conditions, the program reads in a potential map from a previous run, and compares the scale of the focusing map with that for the current run. If they are the same, it assumes that this is a continuation of a previous run, and iteration of the potentials contained in the previous potential map is continued. If the scale is not the same, it checks to ensure that the new lattice lies completely within the old lattice before interpolating the boundary conditions.

CAUTION: With BNDCON=3, one must provide the ACENTER to prevent Delphi from exiting with a FATAL ERROR.

(4) - Coulombic. They are approximated by the sum of Debye-Huckel potentials of all the charges. qi is the i'th charge, and ri is the distance from the lattice boundary point to the charge.

List of Parameters

4.2.13 LINIT

An integer number of iterations with linear equation. The convergence behavior of the finite difference procedure is reported in the log file as both the mean and maximum absolute change in potential at the grid points between successive iterations. The latter is probably more important since it puts an upper bound on how much the potential is changing at the grid points. It is suggested that sufficient iterations be performed to give a final maximum change of less than 0.001 kT/e. The number of iterations per se is not important, as long as its sufficient to give the required convergence. The convergence behavior can also be judged from the slope of the semi-log plot of the mean and max changes given in the log file. LINIT is best determined by experience, since the convergence rate depends on several factors. Start with say 100 iterations, and then increase the number of iterations until sufficient. Note that a run can be restarted by using focusing boundary conditions with exactly the same SCALE, PERFIL and ACENTER values (see note 5). Some guidelines are: The number of iterations needed will increase with grid size (GSIZE). It will decrease with decreasing PERFIL, since the potentials converge more rapidly in the solvent. It will decrease with increasing ionic strength. The number is fairly insensitive to the size and number of charges on the molecule. Example:

```
linit=400 or li=400.
```

List of Parameters

4.2.14 NONIT

An integer number (leq0) of non-linear iterations. If linear PB equation only is required, NONIT is set to be 0. Example:

```
nonit=400 or ni=400.
```

List of Parameters

4.2.15 FCRG

A flag, normally set to false indicating a linear cubic interpolation of charges to grid points; set to true this turns on a spherical charge interpolation. If an atomic charge does not lie exactly on a grid point, then it must somehow be distributed onto the grid points. If this flag is set false, the standard algorithm is used which distributes a charge to the nearest 8 grid points (quick and simple, see the Proteins paper of Klapper et al.). If this flag is set true, then an algorithm is used which gives a more spherically symmetric charge distribution, although the charge is now spread over a wider region of space. For certain cases this gives higher accuracy for potentials less than 3 grid units from a charge (see Gilson et al. J.Comp. Chem paper), although this point has not been exhaustively explored.

List of Parameters

4.2.16 LOGPOT

A flag that activates the potential listing during the run. Example:

```
logpot=t or lp=t or logfilepotentials=t
```

List of Parameters

4.2.17 LOGGRP

A flag that activates the convergence plot during the run. Example:

```
loggrp=t or logfileconvergence=t or lg=t
```

List of Parameters

4.2.18 CONINT

A flag that determines at what iteration interval convergence is checked, by default it equals 10.(usually not modified from default) The idea behind this parameter is to allow convergence to be checked less frequently to reduce the amount of time spent. Example:

```
conint=10 or ci=10 or convergenceinterval=10
```

List of Parameters

4.2.19 CONFRA

A flag that determines the convergence fraction. Iit decides what fraction of grid points are used in assessing convergence (1=all, 2=half, 5=fifth etc). By default, it equals 1 (usually not modified from default). Example:

```
confra=10 or cf=1 or convergencefraction=1
```

List of Parameters

4.2.20 PBX, PBY, PBZ

They are the three logical flags (t/f) for periodic boundary conditions for the x,y,z edges of the lattice respectively. Note that periodic boundary conditions will override other boundary conditions on edges to which they are applied. Periodic boundary conditions can be applied in one or more of the x, y or z directions. When applied, the potential at each periodic lattice boundary point is iterated by supplying its missing neighbor(s) from the corresponding point on the opposite edge of the lattice. This can be used for example to model an infinite length of DNA. Assume that the helical axis of the DNA in the pdb file is aligned along the Z axis. The periodic boundary flags are set to false, false, true, and the percent fill of the box, PERFIL, is adjusted so that an integral number of turns just fill the box in the Z direction. Normal boundary conditions are applied to the X,Y boundaries. By setting two, or three of the boundary flags to true, one can simulate 2 dimensional or 3 dimensional cubic lattices of molecules. Example:

```
pbx=t or px=t or periodicboundaryx=t
```

List of Parameters

4.2.21 AUTOC

A flag for automatic convergence. The program by default will automatically calculate the number of iterations needed to attain convergence. It is automatically set if no number of iteration is specified otherwise. See also LINIT and GC options Example:

```
autoc=t or automaticconvergence=t or
autoconvergence=t or autocon=t or ac=t
```

List of Parameters

4.2.22 EXITUN

A flag to terminate the program if uniform dielectric is present (INDI=EXDI). By default it is false. (Usually not modified). Example:

```
exitun=f or exituniformdielectric=f or xu=f
```

List of Parameters

4.2.23 GRDCON

The value for grid convergence. When set, the criterion used to stop the iterative process is the difference on values of grid energy; this option might slow down the calculation a bit, but provides a very strong criterion. Example:

```
grdcon=0.001 or gc=0.001 or gridconvergence=0.001
```

List of Parameters

4.2.24 RELFAC

The externally assigned value for spectral radius (define spectral radius). (Usually not modified from default). Example:

```
relfac=0.9975 or relaxationfactor=0.9975 or rf=0.9975
```

List of Parameters

4.2.25 CHEBIT

A flag, that if it is true the relaxation parameter for linear convergence process is set equal to 1. (Usually not modified from default). Example:

```
chebit=t or ci=t
```

List of Parameters

4.2.26 SOLVPB

A flag, which controls the Poisson-Boltzmann solver. Normally DelPhi will invoke the Poisson-Boltzmann solver but if you are interested in using DelPhi for other things such as calculating surface area or producing a GRASP viewable surface file, you can turn off the solver using this option. Example:

```
solvpb=t or sp=t
```

List of Parameters

4.2.27 CLCSRF

A flag, that when set to true, outputs a GRASP viewable surface file in the name grasp.srf. Example:

```
clcsrf=t or cs=t
```

List of Parameters

4.2.28 PHICON

A flag, that maps charge density in a .phi file, with a procedure that is equivalent to the one that saves the potential map. phicon=f produces standard potential output in kT/e (approximately equal to 25.6 mV at 298 K, or to 0.593 kcal/mole of charge). phicon=t will give net solvent ion concentration output in M/l, where for every lattice point inside the molecule the concentration is 0, and the outside concentration is obtained from: (-2 * I * sinh(Φ)) or its linearized version if linear PBE is used. Example:

```
phicon=t
```

List of Parameters

4.2.29 RADPOLEXT

A default radius for point charges in a continuum (only in objects) (see self-reaction field energy). Example:

```
radpolext=1 or radpol=1 or rl=1
```

List of Parameters

4.2.30 RELPAR

A manually assigned value for relaxation parameter in non-linear iteration convergence process. (see non-linear equation convergence) Example:

```
relpar=1.0 or rr=1.0
```

RELPAR is strongly recommended to be used in non-linear calculation. RELPAR=1.0 is good for most of the cases.

List of Parameters

4.2.31 SALT2

The concentration of second salt (if present) expressed in Moles/liters. (See multi-salt option). Example:

```
salt2=0.2 or s2=0.2
```

List of Parameters

4.2.32 RADPR2

The value for effective probe radius relative to the part of the molecule, which is internal to an object. (See geometric objects) Example:

```
radpr2=2 or r2=2
```

List of Parameters

4.2.33 VAL+1

(VAL-1 VAL+2 VAL-2) A number ¿ 0, valence of positive (negative) ion constituting salt one (two). (See multi-salt option) Example:

```
val+1=1 or +1=1
```

List of Parameters

4.2.34 RMSC

The convergence threshold value based on root mean square change of potential. (See convergence hints). Example:

```
rmsc=0.0001 or mc=0.0001
```

List of Parameters

4.2.35 MAXC

The convergence threshold value based on maximum change of potential (suggested). (See convergence hints) Example:

```
maxc=0.0001 or xc=0.0001
```

List of Parameters

4.2.36 GAUSSIAN

gaussian=1 indicates that the Gaussian smooth dielectric method is selected. gaussian=0 is for the traditional homogeneous method. Default value of gaussian is 0.

List of Parameters

4.2.37 SIGMA

Sigma is the value of the variance of Gaussian distribution, in equation

$$\rho_i(r) = exp\left(\frac{-|r - r_i|^2}{\sigma^2 R_i^2}\right) \tag{5}$$

Example:

```
sigma=2.0
```

List of Parameters

4.2.38 SRFCUT

When calculating the solvation energy using Gaussian smooth method, a cutoff of dielectric value is needed to determine the border between protein and solvent phases. SRFCUT is used to specify this cutoff. This option is unnecessary if there is no solvation energy calculated in the run, and say it is being used to output the potential map only. Note that solvation energy is calculated with respect to media with dielectric constant equal to the internal dielectric constant INDI(sin) Example:

```
srfcut=20.0
```

List of Parameters

4.2.39 EXDI2

The second external (solution) dielectric constant in Gaussian runs. A combination of values of EXDI=80 and EXDI2=1 indicates calculating the energy of transferring the molecule from vacuum to water. The default value of EXDI2 is 1. Example:

```
exdi=1.0
```

List of Parameters

4.2.40 DENCUT

When calculating the solvation energy using Gaussian smooth method, a cutoff of atomic density can be used to determine the border between protein and solvent phases. DENCUT is used to specify this cutoff. Notice that users can use either DENCUT or SRFCUT. When DENCUT value is set properly (between 0 to 1), the DENCUT will be used and SRFCUT will not be recognized Example:

```
dencut=0.8
```

List of Parameters

4.2.41 RADIPZ

Radipz is the cutoff of 'neighbor charges'. One charge is considered as a neighbor charge if the distance between this charge and any grid is less than the Radipz. Neighbor charges are neglected in MEMPOT algorithm, because the charges too close to grid may result artificial large potential. Default value of Radipz is -1.0, which means this option is off. When the value is set to be greater than 0, the MEMPOT option is turned on. Example:

```
radipz=0.5
```

List of Parameters After the run, a pz.txt file will be generated; here is one example of pz.txt: The 2nd column is the grid index on z axis (If gsize = N , then there should be N

z:	1	-7.958	n:	11025	Pz:	-0.0000	kt/e	or	-0.0000	mv
z:	2	-7.458	n:	11024	Pz:	0.0571	kt/e	or	1.4761	mv
z:	3	-6.958	n:	11020	Pz:	0.1051	kt/e	or	2.7171	mv
z:	4	-6.458	n:	11013	Pz:	0.1371	kt/e	or	3.5440	mv
z:	5	-5.958	n:	11004	Pz:	0.1225	kt/e	or	3.1656	mv
z:	6	-5.458	n:	11006	Pz:	0.1495	kt/e	or	3.8646	mv

rows in this file); The 3rd column is the z-coordinate of each x-y plane; the 5th column is the total grid number which are involved in the average potential calculation (note some grids are neglected during the calculation if they are too closed to charged atoms, so the value of this column should be $\leq$ N * N); the 7th column is the averaged potential on each plane in kT/e unit; the 9th column is the averaged potential on each plane in mV unit.

List of Parameters

4.2.42 SURFPOT

SURFPOT=1 or SU=1 executes surface potential calculations (=0 is the default). These calculations are only run if the traditional 2-dielectric model is used. DelPhi automatically

suppresses the surface potential calculations if GAUSSIAN=1. The following line in the parameter file can invoke surface potential calculations. Example:

```
surfpot = 1 or SU = 1
```

List of Parameters

4.2.43 SURFDIST

This is the distance (in Å) from the vdW surface of the solute at which the potential values are probed. The grid-points which are located at this distance from the vdW surface are first identified and the potential values on them (in kT/e) are reported in the .zphi output file at the end of the program. This output reports the average of all the potentials (also printed in the output console) besides the 3D-coordiantes of each grid-point in the Euclidean space in the first three columns and the corresponding potential on the fourth column. With this data, the user can choose to analyze the collection of potentials on the surface in any manner he/she desires. To prevent undesirable results, DelPhi only allows values between 0 and 10 A. This is because the number of grid points created for the run is constrained by PERFIL and hence, at larger distances, a closed surface will not be obtained (no grid-points present to close the surface). The output. zphi file can be used to visualize the distribution of potentials on the grid-points on VMD using this script. Example:

```
surfdist = 8 or SD = 8
```

List of Parameters

4.2.44 GAPDI

this option gives the user the ability to set the maximum limit for dielectric value for any point inside the van der Walls surface for the protein, with Gaussian model. The default value for this parameter is same as external dielectric. Example:

```
gapdi = 70 or
```

List of Parameters

4.2.45 MAXWARN

this option gives the user the ability to control the number of harmless warnings that are printed after a Delphi run finishes without crashing. Sometimes, the nnumber of warnings can be very large and it maskes the actual results printed on the command line. This option can help avoid that. By default, all the warnings will be printed. Example:

```
maxwarn = 8 or
MW = 8 ! to print at most 8 of the warnings after Delphi
       ! run is finished without problems
```

```
        maxwarn = -1 or
        MW = -1 ! to print all the warnings after Delphi run is
                ! finished without problems
```

List of Parameters Return to TOC

FILES

Many files are used to input and output data. Each type of data or output has its own file. The default name for the unit [number] file is fort.[number] (fortran and C++ version). Here follows a description and format for each of them.

5.1 Input Files

5.1.1 UNIT 10

Default extension prm. Contains input parameters. See the section on Statements and Functions for full details.

Return to TOC

5.1.2 UNIT 11

Default extension siz. List describing the van der Waals radii to be assigned to each atom/residue pdb record type. A sample file is provided together with the code. Note the atom and residue fields ignore case and leading blanks. The residue field may be left blank (wild card), causing a match with the given atom type of any residue. ONLY if the residue field is left blank, the LAST 5 characters of the atom record may be left blank. In this case all atom types beginning with the letter in column 1 will be matched. Records of greater specificity override those of less specificity. Beware of ambiguities like calcium (ca) and alpha carbon! All atoms of an input pdb file must be assigned a radius through the siz file, even if it is 0, or the output will be flagged with a warning.

Return to TOC

5.1.3 UNIT 12

Default extension crg. List of the atomic charges to be assigned to each atom/residue/number/chain pdb record type. A sample file is provided together with the code. The ascii fields for atom, residue, number and chain ignore case and leading blanks. Any field except the atom name may be left blank and will be treated as a wild card. Records of greater specificity override those of lesser specificity as for the siz file above. Search order:

```
atom_res_num_chain
atom_res_num______
atom_res_____chain
atom_res__________
atom_____num_chain
atom_____num______
atom_________chain
```

```
atom______________
```

Atoms that do not find a match in the crg file will be neutral (q=0.0) file must have a line:

```
atom resnumbc_charge_
```

Examples: A line as shown below will charge only the N atom of ALA residues.

```
N        ALA      -0.400
```

A line as show below will charge all N atoms.

```
N        -0.400
```

Note that position of text phases and numbers is strictly determined and cant be changed!

Return to TOC

5.1.4 UNIT 13

A Brookhaven protein data bank standard format file containing atom labels and coordinates, or a modified OB- JECTFILE. Only records starting with ATOM or HETATM are read; if objects or multi-dielectric option are used, also the keywords MEDIA, OBJECT, CRGDST, DATA are also read. The default extension is pdb. The precise format is essential; using Fortran syntax:

```
(6A1,I5, 1X,A4,A1, A3,1X,A1,I4,A1,3X,3F8.3,2F6.2,1X,I3)
```

is used for the atom record. From left to right, the fields contain 'ATOM ' or 'HETATM' atom serial number, atom name, alternate location indicator, residue name, chain identifier, residue sequence number, residue insertion code, x, y, and z coordinates, occupancy, temperature factor, footnote number. Note that the program treats the residue number as an ascii string, not as an integer. As a warning to the user, there are many variations, and even outright errors found in the format of pdb files obtained from the web. It would be wise to double-check the contents of a file to save any heartache.

Return to TOC

5.1.5 UNIT 15

Default extension: pdb or frc. List of coordinates where site potentials are output in Unit 16. Format as for Unit 13. With Version 8.1, FRC input can also be supplied in a PQR format file. In the parameter file, a statement of the form:

```
in(FRC,file=<file_name>,format=PQR)
```

ensures that the FRC input is read in the PQR format. It is very important that the input structure file (or unit 13) is also supplied in the PQR format.

Return to TOC

5.1.6 UNIT 18

Default extension phi, potential map for focusing boundary conditions. Potentials are in kT/e (25.6mV, 0.593 kcal/mole/charge at 298 K). The format of the file is given below in case that the user wants to adopt the file to its own software. If the users want to visualize the file with Grasp or Insight, no action should be taken. unformatted (binary file) character*20 uplbl character*10 nxtlbl,character*60 toplbl real*4 phi(65,65,65) character*16 botlbl real*4 scale,oldmid(3) uplbl, nxtlbl, toplbl, botlbl are ascii information. Phi is the 3D array containing values of potential for all the lattice points. Index order is x,y,z. Scale is lattice scale in grid/. Oldmid is the x,y,z coordinates in real space (Å) of the centre of the lattice: thus the real space coordinates x,y,z of the lattice point for phi(IX,IY,IZ), for the case where IGRID = 65, are:

$$x = (IX - 33)/scale + oldmid(1) \tag{6}$$

$$y = (IY - 33)/scale + oldmid(2) \tag{7}$$

$$z = (IZ - 33)/scale + oldmid(3) \tag{8}$$

where 33 = (65 + 1)/2 is the middle point of the grid.

Return to TOC

5.2 Output Files

5.2.1 UNIT 6

Output from the program, including error messages and convergence history. When run interactively, appears on standard output. Default extension log when run in batch.

5.2.2 UNIT 14

If the flag IBIOS (BIOSYM) is false, then output is in DELPHI format, default extension phi. The output can be either a potential map or a concentration map, with format same as for unit 18 above. The output phi map has the same scale as used in the calculation (i.e, variable) unless format=grasp is specified. The grasp-style phi map format will always interpolate to a 65 x 65 x 65 grid for use in Grasp (or other hardwired display/analysis programs). If the flag IBIOS (BIOSYM) is true, then output is in INSIGHT format, default extension ins. This is an unformatted (binary) file. As it was explained above, the format

is provided only for completeness in case that one wants to visualize the file with different than Insight software.

```
character*132   toplbl  !ascii header
integer*4       ivary   !0=> xindex     varys most rapidly
integer*4  nbyte        !=4, # of bytes in data
integer*4 inddat        !=0, floating point data
real*4 xang,yang,zang   !=90,90,90 unit cell angles
integer*4 intx,inty,intz !=igrid-1, # of intervals/grid side
real*4 extent   !maximum extent of grid
real*4 xstart,xend      !beginning, end of grid sides
real*4 ystart,yend      !in fractional
real*4 zstart,zend      !units of extent write(14)toplbl
write(14) ivary, nbyte, intdat, extent, extent, extent, xang, yang,
    zang, xstart, xend, ystart, yend,   zstart, zend, intx, inty,
    intz
do k = 1,igrid
    do j = 1,igrid
        write(14)(phimap(i,j,k),i=1,igrid)
    end do
end do
```

Note that for grid sizes less than 65, INSIGHT format files will occupy less disk space than the corresponding DELPHI files. ins files are designed as input to a Biosym Corp. stand alone utility called CONTOUR, supplied with INSIGHT Version 2.4. This program will produce contour files for display with INSIGHT.

If the flag CUBE is true, then output is in CUBE format (Gaussian Cube). Example:

```
Out(phi,file=phimap.txt,form=cube)
```

creates file ‘phimap.txt in the cube-format. There is a source code for saving in the cube-format.

```
write(6,*) ‘Potential map in cube format ’
write(6,*) ‘written to file’, filnam
write(14,*) ‘qdiffxs4 with an improved surfacing routine’
write(14,*) ‘Gaussian cube format phimap’
coeff=0.5291772108
stepsize=1.0/scale
do i=1,3
    origin(i)=oldmid(i)-stepsize*(igrid-1)/2/coeff
enddo
write(14,‘(i5,3f12.6)’) 1, (origin(i),i=1,3)
write(14,‘(i5,3f12.6)’) igrid, stepsize/coeff,0.0,0.0
write(14,‘(i5,3f12.6)’) igrid, 0.0,stepsize/coeff,0.0
write(14,‘(i5,3f12.6)’) igrid, 0.0,0.0,stepsize/coeff
write(14,‘(i5,4f12.6)’) 1,0.0,0.0,0.0,0.0
do i = 1,igrid
    do j = 1,igrid
        write(14,‘(6E13.5)’)(phimap(i,j,k),k=1,igrid)
    end do
end do
```

Return to TOC

5.2.3 UNIT 16

Default extension frc. A list of potentials and fields at coordinates in pdb file read on unit 15. Format: 12 lines of ascii header information, followed by a variable number of records written as: 230 format(8G10.3) write(16,230)xo,chrgv,phiv,fx,fy,fz where xo(3) are x ,y ,z coordinates of charge, chrgv is the charge value, phiv is the potential (in kT/e) at that point, and fx, fy, fz are the field components (in kT/e/Å). The last line of the file is the sum of chrgv * phiv/2 over all the charges in the file. This quantity can be used for calculating solvation and interaction energies.

Return to TOC

5.2.4 UNIT 19

If the "modified pdb file" option is activated in a WRITE/OUT function, a logical flag (t/f), iatout, will be set to true and will produce a modified PDB file written on unit 19, containing the radius and charge assigned to each atom written after the coordinates in the fields used for occupancy and B factor. It is recommended that this option be set initially so that the user can check that all the radius and charge assignments are correct. Looking at the total charge written to the log file can make an additional check on the charge assignment.

Return to TOC

IMPORTANT FEATURES IN DELPHI

Here below are described some of the new features that have been introduced to DelPhi v.6.0+. Features 1 and 2 are new to Delphi v8.0. Features 3 and 4 were introduced with Delphi v7.0.

1. Parallelized Delphi That can be Run on OpenMP and MPI Platforms.
2. FRC Input Readable in the pqr Format to Enable Computation of Interaction and/or Potential Energy
3. Gaussian with an Appropriate Treatment of Electrolytes/Salt in the Solvated System
4. SURFPOT: Computing Potential on a Surface Located at a User Defined Distance From the vdW Surface.
5. Gaussian Smooth Dielectric Function to Distribute Dielectric Values In Space
6. MEMPOT
7. Multi-Dielectric Code
8. Non-Linear PBE Solving Routine
9. Multi-Salt Code
10. Energy Partitioning
11. Selecting Output Format Using "ideveloper" Option (DelPhi v.6.2 and onward)

6.1 Delphi That can be Run on OpenMP and MPI Platforms

Delphi v8.0+ comes with the capacity, which enables it to be run on parallel computers. The OpenMP compatible version harnesses multiple threads made available to the program. The MPI version, with its novel memory distribution technique, can be run on multiple computing nodes.

Return to TOC

6.2 FRC Input Readable in the pqr Format to Enable Computation of Interaction and/or Potential Energy

Delphi v8.1 allows the user to provide FRC input in PQR format. This way the users can provide the coordinates of the sites to request electrostatic potential and electric field values there and the charges can be used to denote a presence of an atom at those sites. The radius field is ignored. Using this feature, once can compute the electrostatic potential energy (and interaction energy) of the collection of atoms provided by the FRC input file in the field of the source molecule. To provide input FRC in PQR format, the main structure file should also be in the PQR format

6.3 Gaussian with an Appropriate Treatment of Electrolytes/Salt in the Solvated System

The Gaussian-based smooth dielectric distribution, introduced in the earlier versions of Delphi (6.0+), can now handle the presence of implicit electrolytes/salt in the solvent. Using the Born formula of the polar solvation energy, a penalty term is added to the PBE that penalizes salt ions as they 'move' close to the solute (lower dielectric regions). This penalty term originates from the concept of desolvation energy and can be expressed as:

$$\Delta G_{penalty} = \frac{N_A Z^2 e^2}{8\pi\epsilon_o r_0}\left(\frac{1}{\epsilon_r} - \frac{1}{\epsilon_w}\right) \tag{9}$$

where N_A is the Avogadro constant, z is the valence of the ion, e is the elemental charge, ϵ_o is the permittivity of vacuum, r_0 is the effective radius of the ion, ϵ_r is the dielectric constant at a given location and ϵ_w is the dielectric constant of bulk water. The above energy term is incorporated into the standard PBE

$$\nabla.\left[\epsilon(\vec{r})\nabla\phi(\vec{r})\right] = -4\pi\left(\rho_{solute}(\vec{r}) + \sum_{i=1}^{N} q_i c^{bulk} exp\left(\frac{q_i\phi(\vec{r}) - \Delta G_{penalty}}{RT}\right)\right) \tag{10}$$

where $\epsilon(\vec{r})$, $\phi(\vec{r})$ and $\rho_{solute}(\vec{r})$ are the space-dependent dielectric constant, electrostatic potential, and charge density of solute at given locations, respectively. q_i is the ionic charge, c^{bulk} is the ion concentration in bulk solvent, R is the ideal gas constant, and T is the temperature.

For more details, please refer to:

Z. Jia, L. Li, A. Chakravorty, and E. Alexov, 'Treating ion distribution with Gaussian-based smooth dielectric function in DelPhi', J Comput Chem. 2017 August 15; 38(22): 19741979

Return to TOC

6.4 SURFPOT: Computing Potential on a Surface Located at a User Defined Distance From the vdW Surface

This feature enables a user to obtain the electrostatic potential on the grid points present on an approximate surface that lies exterior of the solute at some user-defined distance from its vdW surface. For more details, please refer to:

A. Chakravorty, Z. Jia, L. Li, and E. Alexov, 'A New DelPhi Feature for Modeling Electrostatic Potential around Proteins: Role of Bound Ions and Implications for Zeta-Potential', Langmuir, (2017) 3:9, 2283-2295

Return to TOC

6.5 Gaussian Smooth Dielectric Function

DelPhi v.7.0+ provides Gaussian smooth dielectric function to users. Previous versions of DelPhi treat the molecule as a homogeneous media with low dielectric constant; and treat water as another homogeneous media with high di- electric constant; thus at the boundary between water and molecule, there is a sharp jump of dielectric constants. In DelPhi v.7.0+, users can still use the same scenario as in previous versions without any changes to the parameter files. Besides, users are also able to use the new Gaussian smooth dielectric function instead of the homogeneous dielectric functions. For more details, please refer to:

L. Li, C. Li, Z. Zhang, E. Alexov, 'On the Dielectric Constant of Proteins: Smooth Dielectric Function for Macromolecular Modeling and its Implementation in DelPhi', J. Chem, Theory Comput. 2013 Apr 9; 9(4): 2126- 2136.

Return to TOC

6.6 Multi-Sigma Gaussian Smooth Dielectric Function

DelPhi v.8.5.0+ provides multi-sigma Gaussian smooth dielectric function to users. In multi sigma Gaussian model user can provide different values for Gaussian variance parameter for different atoms, residues, or groups of atoms in the system.

Return to TOC

6.7 Gaussian Smooth Dielectric Function with Gap Dielectric

DelPhi v.8.5.0+ provides Gaussian Smooth Dielectric Function with Gap Dielectric to users. With this parameter users can set a maximum value of dielectric inside the protein, this value must be strictly between internal and external dielectric values.

Return to TOC

6.8 MEMPOT

MEMPOT (MEMbrane POTential) uses the potential map to generate the electrostatic potential profile of membrane while avoiding artificial contributions from grid points being too close to charged atoms or ions. The structure of the membrane is placed in a grid box defined by N * N * N periodic grids (users need to make sure that the surface of the membrane is set along the plane defined by the x and y axes whereas the z axis is parallel to the membrane normal). MEMPOT is used to calculate the profile of average potential value along z-axis, which is a macroscopic quantity independent of the position along the x-y plane. For more details, please refer to:

Roberta P. Dias, Lin Li, Thereza A. Soares and Emil Alexov, Modeling the Electrostatic Potential of Asymmet- ric Lipopolysaccharide Membranes: The MEMPOT Algorithm Implemented in DelPhi, Journal of Computational Chemistry, DOI: 10.1002/jcc.23632

Return to TOC

6.9 Multi-Dielectric Code

Earlier versions of DelPhi handle a scenario where a single molecule is immersed in a solution. In other words, only a "two-media" world was considered. In the present version, a system with many different objects having different dielectric constant can be modeled. These objects can be either sets of atoms obtained from a pdb file or geometric objects. Note: In order to gain better accuracy for reaction field energy, the location of polarization charges is normally projected onto the molecular surface. If a molecule is immersed in an object, the molecular surface would be built inside the object. If two molecules with different dielectric constants come in contact or overlap, the molecular surface at their interface is not built. (In fact, it doesn't make sense to do so in that case.) Instead, the polarization charges are not projected anywhere but are left at their own grid point locations.

Return to TOC

6.10 Non-Linear PBE Solving Routine

An algorithm has been implemented that can solve the non-linear PBE, as described in Rocchia et. al. Journal of Physical Chemistry B paper. A good relaxation parameter must be used for fast and reliable convergence of the non-linear equation. A heuristic algorithm is used to estimate it; nonetheless, for some particularly pathological conditions, it is possible that the choice is not optimal and that the convergence is very slow or even absent.

In order to deal also with those systems, the new statement `relpar=[value]` can be put in the parameter file to manually assign the relaxation parameter.

A general set of guidelines to the choice of the relaxation parameter follows: The optimal relaxation parameter for a non-linear system generally turns out to be less than 1. Pathological conditions like a very high charge in proximity to the surface, ionic strength close to zero but not null and some particular scale values, can require a parameter much lower, down to 0.001 or even lower. On the other hand, a too low value has the drawback of slowing down the convergence and, in the worst case, to give only an illusion of convergence. Thus, the advice is to start with an intermediate value and to decrease it until the convergence is reached.

Convergence hints again: The hardest part for the convergence process is at the beginning, when the non-linearity of the equation is slowly added. In this phase the relaxation parameter could be quite low, and thus can hinder the continuation of the convergence

process. This is the rationale behind the heuristic algorithm. A good convergence is characterized by a low max change in the potential together with a relaxation parameter not too close to zero, say greater than 0.1 or similar. If the convergence is absent or not satisfying, the user can decide to use a fixed, but suitably tuned relpar, which should be as high as possible, provided it allows for convergence.

Two more statements have been added, that are valid for both linear and nonlinear cases: rmsc = potential root mean square change threshold, [kT/e] and maxc = potential maximum change threshold, [kT/e] that allows two new convergence criteria. The first concerns the rms change and the second the maximum change of grid potential between successive iterations. The second one is of particular interest because it sets an upper bound on the error in potential throughout the grid. Both criteria can be assigned together with the number of iterations, linit or nonit; the program will stop iterating as soon as one of the assigned criteria is satisfied.

Return to TOC

6.11 MULTI-SALT CODE

Including the effects of salts of different valences makes a large difference when calculating the electric field of a solute in solution. Handling multiple valences properly requires using the more general expression for the charge in the original PBE. The ionic strength is then calculated accordingly to its general definition:

$$I = \left(\frac{\sum_j z_j^2 C_j^{bulk}}{2} \right) \tag{11}$$

the concentration of each ion is related to the one of the salt assuming electro-neutrality for each salt, that is:

$$\sum_j z_j^2 C_j^{bulk} \tag{12}$$

for the ions of every single salt in solution. Some new statements have been added to handle this new situation:

salt=[conc first kind of salt, moles/liter] val+1=[valence of positive ion in salt type 1] val-1=[valence of negative ion in salt type 1] salt2=[conc second kind of salt, moles/liter] val+2=[valence of positive ion in salt type 2] val-2=[valence of negative ion in salt type 2]

Return to TOC

6.12 ENERGY PARTITIONING

A new partitioning of the energy is considered and described in details in the article of the Journal of Physical Chemistry B paper. Accordingly, the electrostatic energy is subdivided into: coulombic, reaction field, self-reaction field, external ion contribution, osmotic pressure

term, “rho*phi” (electrostatic stress). The last three terms only appear when the ionic strength is greater than zero. The last two terms cancel out in the linear PB equation.

Coulombic and reaction field energies are calculated using Coulomb's law and Gauss's theorem as described in the J. Phys. Chem. paper.

The osmotic and electrostatic stress terms, defined in the same paper, are calculated whenever the program is asked to run nonlinear iterations; their contribution arises from the solvent inside the box The calculation of the external ion contribution is toggled by the flag ion in the energy statement; it calculates the direct interaction of ions to real charges.

This direct calculation needs some comments: it calculates the columbic interaction between the ions in solution (screened by the surrounding polarized water). First of all, it can be computationally expensive, especially if the grid size is very high. Secondly, this energy neglects all of the ionic contribution due to the ions located outside the box. This can result in a underestimation of the real contribution, especially if the percentage filling is high. In the linear case this problem has been reduced almost completely by a mixed numerical/analytical technique. The analytical contribution of the first term in a spherical expansion of the molecular system outside the box is calculated. So the direct ionic contribution is better estimated by the sum of two terms: the one internal to the box and the one external to the box.

For the non-linear case this is not so easy, because an analytical solution of the non-linear PBE for the spherical case is not available. The use of the linear approximation routine would result in an overestimation of the ionic contribution. Another way to get an estimation of the same energy term is to subtract the grid and reaction field energy in two cases: with and without salt. This means running the program twice on the system.

In order to help the user decide if these two runs are necessary, the program estimates and reports the number of Debye lengths bounded within the grid. This number, together with the ionic strength, aids the user in estimating the error due to neglecting the ionic contribution outside the box.

Note: If periodic boundary conditions are used, the estimated external ion contribution will be taken only in the directions where periodicity is NOT invoked. This is done in order to give an estimate of energy per periodic cell.

The self-reaction field energy is calculated whenever the usual reaction field energy is calculated. It is an attempt to estimate the energy that the system gains when a point charge is moved from vacuum to a dielectric medium. In fact, when a charge moves from vacuum to a dielectric a polarization charge of opposite sign builds up around it. This results in an energetically favorable process. The higher the dielectric constant of the medium, the higher is the amount of polarization charge and the more energetically favorable is the process. In our model, it is assumed that polarization charge builds up spherically around any real point charge, at a distance the user can decide via the statement `radpolext=[radius]`, which by default is set to be 1Å. This information is no attempt to give a quantitative estimation of this energy but just to take into account in a more detailed way the fact that

the most favorable place for a charge is within a high dielectric medium. This necessity arises when a system with many different dielectric regions is studied.

Return to TOC

6.13 SELECTING OUTPUT FORMAT USING ideveloper OPTION (DelPhi v.6.2 onward)

Users now are able to select the output format either in single or in double precision by flipping over the logical switch ideveloper in subroutine defprm. When ideveloper is set to be. true., the output values have more decimals printed out, which would be useful to have a closer insight of the changes of certain values.

Return to TOC

APPENDIX

A sample parameter file that is input to DelPhi looks like this:

```
!example prm file for DelPhi v.7.0+ scale=2.0
perfil=70
!print all the warnings maxwarn = -1 in(pdb,file="test.pdb") bndcon=2
indi=4 exdi=80 ionrad=2.0 gaussian=1 sigma=0.93 srfcut=20.0
!salt=0.0
!out(frc)
!out(modpdb)
!out(phi,file=phimap.txt,format=cube) energy(s,g,c)
```

Return to TOC

7.1 Examples

Various parameter files are present in the examples that are packaged with DelPhi distribution. Two sets of examples are provided. The first one is for beginners and provides basic examples of computing various energy terms. The corresponding file is called Examples_Basic.tar.gz. The second set of examples targets more advanced users and provides examples of more advanced DelPhi features. The corresponding file is called Examples_Adavanced.tar.gz. Below is the list of examples with short descriptions.

7.1.1 Basic Examples

- Example 1: Calculating the electrostatic component of solvation energy of a single atom. This is the energy that can be obtained with Born formula and thus, the results can be compared with the analytical solution of the Born formula.
- Example 2: Calculating the electrostatic component of solvation energy of a protein (PDB ID: 1BRS) using AMBER charge/radii parameters. PDB ID: 1BRS corresponds to Barnase + Barstar complex. For this example only the Barnase chain is used.
- Example 3: FRC module Calculating the electrostatic potential and field values on an atom or a group of atoms (a residue or multiple residues). Here the potential and field are calculated on a residue of Barstar when its binding partner Barnase is taken to be the source. It also demonstrates the use of FRC module when PQR formatted ftles are used.
- Example 4: Using Focusing method to calculate electrostatic potential and field in a particular region.
- Example 5: Calculating the electrostatic component of solvation energy of two charges inside a spherical dielectric cavity.
- Example 6: Calculating electrostatic component of solvation energy using the GAUSSIAN-BASED smoothing of dielectric

- Example 7: Obtaining electrostatic potentials on grid points that are placed at a specified distance from the van der waals surface of the molecule using the new SURFPOT module in DelPhi.
- Example 8: Solving Non-linear Poisson-Boltzmann equation to obtain the total electrostatic energy of a protein.
- Example 9: Calculating energy of Saltation using the Gaussian module that accounts for the presence of electrolytes in the solvated system.

7.1.2 Advanced Examples

- Example 1: Calculating electrostatic component of binding energy of two proteins forming a complex.
- Example 2: Using site-specific potential routine (FRC) to calculate electrostatic energy of interaction between peptide and membrane.
- Example 3: Computing electrostatic component of the binding energy of peptide and membrane using Focusing technique.
- Example 4: Manipulating a 3D grid-map output of a particular quantity (potential/dielectric value) in CUBE format to obtain a value of that quantity at a user-specified coordinate.
- Example 5: Calculating surface electrostatic potential on a pre-computed molecular (Connolly) surface.