Selection commands

Since many BFD commands operate on the currently selected atoms/bonds, there is a need for flexible ways to control the selection. There are two main selection commands, **select** and **selection**. Although similarly named, each has sub-commands, and each can have its own sort of expressions. The conceptual model is that a select predicate is applied to every loaded atom. The result, one bit per atom, is recorded with each atom itself. Thereafter, the selection state can be retrieved and manipulated.

The select command

The **select** command sets the selection, i.e., the set of selected atoms/bonds. Each atom, whether protein or fragment, is either selected or it is not. Each bond is selected or not, depending on the selection state of the atoms at its ends, according to the current bond mode (see **display bond-mode**).

select ***expr***

Sets the selection to correspond to the specified *expr*. *Expr* can be a selection primitive, or a selection expression. Conceptually, the expression is applied to every loaded atom. Those atoms for which the expression is true, become selected; those for which the expression is false, become not selected.

Select primitives

There are many selection primitives. They can be combined into selection expressions, or used as is.

select all
select none

Specifies all atoms, or no atoms.

select selected

Selects only those atoms already selected. Useful in and/or/not expressions.

select visible

Selects only those atoms which are not hidden (see **display show**, **display hide**).

select file –ordinal ***n***
select file –name ***name***
select file –list ***file-list***

Selects atoms loaded from the file with the specified ordinal, which starts with zero; or those atoms loaded from the file with the specified name, or list of names.The file name comparison is case-sensitive (which might be a surprise to Windows users) and not canonicalized (directory slash variety, relative pathnames). It is best to use filenames coming directly from **loadfile –list**.

select protein
select dna
select rna
select hetero
select protein –chain ***id***
select protein –chains ***id-list***
select protein –model-number ***num***
select protein –model-numbers ***num-list***

Select all atoms in a protein, dna, rna, hetero (only non-protein, non-nucleotide atoms such as zeolites or carbon nanotubes) or those within the specified chains or model numbers. The chain specification can also be a Tcl list of chains. A chain id can include a numerical suffix (e.g. A1) designating an NMR model number or a loadfile chain suffix (used to distinguish chains when loading multiple files which contain the same chain id's). When the model number is specified, select just the NMR model number *num*, which can also be a Tcl list of model numbers. * can be used to indicate 'all' for a chain id or model number.

select nucleic
select nucleic ***subqty***

Select all atoms in a nucleic acid (RNA or DNA), or those satisfying the subquantity relation (see “Selection primitive quantities”).

select solute
select solute –name *name*
select solute –chain *id*
select solute –chains *id-list*
select solute –model-number *num*
select solute –model-numbers *num-list*

Select all atoms of all solutes of all files. In the second form, only select those solutes for which the name matches. The solute name is typically the PDB accession code—for instance, 1A9U. If chain id or model number is specified, select just those residues with the chain id or model number. Chain id's and model numbers can be Tcl lists to specify multiple chains/models. Chain id's can have model numbers appended to them (e.g. A1). * can be used to indicate 'all' for a chain id or model number.

For more specific solute selections, use something like **select solute-obj [get solute of snapshot *snap*]**.

select chain
select chain *chain-id*
select chains *chain-id-list*

Select all atoms of residues with a chain-id of *chain-id* which can also be a Tcl list of chain id's. Chain id's can have NMR model numbers appended. In addition to PDB MODEL statements, model numbers are also provided by loadfile chain suffix specification when loading multiple files, designating a file ordinal as a model number. * can be used to indicate 'all' for a chain id or model number.

select model-number
select model-number *model-number*
select model-numbers *model-number-list*

Select all atoms of residues in NMR model number *model-number* which can also be a Tcl list of model numbers. In addition to coming from PDB MODEL statements, model numbers are also provided by loadfile chain suffix specification when loading multiple files, designating a file ordinal as a model number. * can be used to indicate 'all'.

select fragments
select fragments *fragspec*
select fragments –name *name*
select fragments *subqty*

Selects all atoms in fragments coming from BMOC snapshot files. Subsets can be specified with *name* or subquantities (see “Selection primitive quantities”). A *fragspec* is a fragment name optionally followed by a period and a fragment ordinal, optionally followed by a colon and a snapshot ordinal, as in benzene.2:4 or just benzene. A list of fragspec's is also allowed.

select sketch
select sketch –name *group-name*
select sketch-group *group-name*

Selects all atoms in sketches (i.e. atoms created with the “sketch add atom ...” command). With –*name* select only atoms in the sketch with the name *group-name* (i.e. atoms created with a –*group* argument in the call to “sketch add atom ...n” command). The sketch-group variant takes a name directly, like 'select residue', without the –*name*.

select waters

Selects all the fragments named "water".

select crystal-waters

Selects all the ligands named "HOH".

select hydrogens

Selects all hydrogen atoms.

select element *name*

Selects all atoms of the specified element. Name can be a number (e.g. 6 for carbon) or a letter (e.g. C for carbon).

select residue
name
.digits
residue spec
digits-digits
***residue spec1* | *residue spec2* | ...**
first
last
select ligand
name
.digits
residue spec

Selects atoms in the specified residue(s). This works for nucleos as well as aminos. The general form for a residue specifcation is:
<name>.<seq number>:<chain>#<model spec>. Note, the name can sometimes (rarely) consist of pure digits, in which case the name can be disambiguated by adding a period after it. A sequence number alone must be preceeded by a period, the reason being that residue names can be purely numeric, so there is ambiguity as to whether 100 means a ligand named 100 or a sequence number 100. The residue name is generally optional, that is, a sequence number is sufficient, although the name is sometimes provided as an extra check for correctness of the specification. The name can consist of letters/underscore, all digits, or letters followed by digits, e.g. VAL, VAL_CT, VAL_NT, VAL101, VAL.101, etc. You can also select a range of residues with from-to (inclusive). For example, " select residue VAL" or "select residue MET_CT" would select all residues with the specified name in all chains or models. "select residue 123." selects the residue coded 123 in the cif/pdb file of all chains or models. "select residue .123" selects the residue with sequence number 123 in all chains or models. "select residue 102-105" selects four residues. Multiple residues can also be specified in a single command by providing a Tcl list of residue specifications enclosed in braces, separated by spaces. Residue specifications can be also combined with vertical bar "|" with no spaces around them, as in "select residue 101|105+|490" (think 'or'). Note, we wanted to use '+' as in Pymol but consider NA+. The keywords "first" and "last" can be used in place of residue names to indicate the first residues of a chain (or all chains if no chain specifier is provided), or the last residue. A chain/model specifier is a colon followed by one or more alphanumeric characters (cif files now allow multiple character chains which are also case sensitive). Use underscore ('_') for space or blank chain. To specify a model number (as in NMR files, but can also be X-ray files), after the chain id, add # followed by the number number which can be any number of digits. Asterisk '*' can be used in place of the chain id or the model number. Examples of the use of :* are :*#1 meaning all chains of model 1 or A#* meaning chain A of all models. (not clear why just :* or :*#* are useful -- just leave off the chain spec altogether). When the "select ligand" command is used, only ligands, designated as HETATM's in pdb files (generally non-polymer molecules) are selected. Conversely, "select residue" only selects ATOM or polymer residues. There is some ambiguity in the case of modified peptide ligands which consist of both standard amino acid residues and non-amino acid residues, in which case using the "select chain" command might be better. Name suffixes _CT and _NT must be specified, but should be optional.

select atom *name*

Selects atoms having the specified *name* within their respective group. For instance, select atom CG2 selects all atoms in respective residues (THR, ILE, VAL have a CG2). Also takes a list of names or the plural form 'atoms'.

select atom-set *name*

Selects all atoms in the named atom-set. Each loaded file is inspected for Sybyl mol2 atom-set definitions. For any given loaded file, use **get atom-sets of file *fn*** to get the list of atom-set names.

select atom-type *type*

Selects atoms having the Amber atom type specified *type*. Note, *type* is case sensitive. Also takes a list of types or the plural form 'atom-types'.

select selection *selobj*

Selects those atoms in *selobj*, which must have been returned from **[selection]**. Although obviously useless by itself, **select selection** can be valuable in *select expressions*.

Select objects

A script can select specific objects, such as atoms, residues or fragments, when it has a representing opaque object. Such opaque objects can come from a **get** command, or from an item-clicked notification (see **set-script –item-clicked**).

select atom-obj *atomobj*

Select the atom represented by *atomobj*. Actually, *atomobj* can be a Tcl list of atom objects, in which case all the atoms are selected.

select fragment-obj *fragmentobj*

Select the atoms in the fragment represented by the opaque *fragmentobj*. Actually, *fragmentobj* can be a Tcl list of fragment objects.

select ligand-obj *ligandobj*

Select the atoms in the ligand represented by the opaque *ligandobj*. Actually, *ligandobj* can be a Tcl list of ligand objects.

select residue-obj *residueobj*

Select the atoms in the residue represented by the opaque *residueobj*. Actually, *residueobj* can be a Tcl list of residue objects.

select sketch-obj *sketchobj*
select sketch-group-obj *sketchobj*

Select the atoms in the sketch group represented by the opaque *sketchobj*. Actually, *sketchobj* can be a Tcl list of sketch group objects.

select snapshot-obj *snapshotobj*

Select the atoms in the snapshot represented by the opaque *snapshotobj*. Actually, *snapshotobj* can be a Tcl list of snapshot objects.

select solute-obj *soluteobj*

Select the atoms in the solute represented by the opaque *soluteobj*. Actually, *soluteobj* can be a Tcl list of solute objects.

These commands have an equivalent construction using a **get** command. For instance, **select atom-obj *atomlist*** could have been written **selection [get selection of atoms *atomlist*]**. Writing it with a direct **select**, however, doesn't require constructing an intermediate selection object.

Select primitive quantities

Selection primitive quantities make a numeric comparison between a property of an atom or fragment, and a number specified with the primitive.

Properties are:

–B	Fragment's B value
–T	Fragment's temperature
–snapshot-ordinal	Ordinal of a loaded snapshot
–-ordinal	Alias for –snapshot-ordinal (deprecated)
–file-ordinal	Ordinal of snapshot within file
–snapshot-in-file	alias for –file-ordinal
–fragment-ordinal	Ordinal of fragment within snapshot
–energy	Fragment's total energy (frag-solute plus frag-frag)
–energyfp	Fragment's energy w.r.t. solute
–energyff	Fragment's energy w.r.t. other fragments in the system

Comparisons are:

eq, =, ==	Equal
ne, !=	Not equal
gt, >	Greater than
ge, >=	Greater than or equal
lt, <	Less than
le, <=	Less than or equal

Quantity equality is computed with some "fuzz". Quantities are considered equal if either absolute difference < 1.5e-5 or relative difference < 1.5e-5.

For example, to select all atoms in fragments with frag-solute energies more favorable than -5.0, use **select fragments –energyfp < -5**.

Select expressions

A selection expression can be a selection primitive; or a combination of primitives or expressions. Selection expressions can be of the form:

select subset *expr*

Subsets the current selection. Equivalent to **select selected and *expr***.

select within (*dist expr*)

Select all atoms within *dist* Angstroms of any atom for which the selection expression *expr* applies.

select within(*dist* –position *x y z*)

Select all atoms within *dist* Angstroms of the position *x,y,z* (in Angstroms).

select within(–box *corner edge1 edge2 edge3*)

Select all atoms within the specified 3D box. One corner is at *corner*. The other corners are found by adding the specified edge vectors to the corner.

If you wanted to specify the box with its center, and include a scale factor, you could use a script something like this:

```
proc ScaledBoxSpecs {center vec1 vec2 vec3 scale} {
   set corner $center

   set vec1 [Mul3_1 $vec1 $scale]
   set vec2 [Mul3_1 $vec2 $scale]
   set vec3 [Mul3_1 $vec3 $scale]

   set corner [Sub3_3 $corner [Mul3_1 $vec1 0.5]]
   set corner [Sub3_3 $corner [Mul3_1 $vec2 0.5]]
   set corner [Sub3_3 $corner [Mul3_1 $vec3 0.5]]

   return [list $corner $vec1 $vec2 $vec3]
}
```

where the Mul3_1 and Sub3_3 are appropriate vector (3-list) operations.

select not *expr*

Select all atoms for which *expr* does not apply.

select *expr1* and *expr2*

Select all atoms for which both *expr1* and *expr2* apply. The operator precedence between **and** and **or** is not defined; use parentheses where it is important.

select *expr1* or *expr2*

Select all atoms for which either or both of *expr1* and *expr2* apply. The operator precedence between **and** and **or** is not defined; use parentheses if it is important.

select (*expr*)

Parenthesized expressions are useful in combination with and/or/not or for otherwise clarifying one's intent in complex expressions.

The selection command

The **selection** command bridges between the selection state of displayed atoms (the realm of the **select** command) and data available to Tcl scripting.

selection

Encodes the current set of selected atoms as a bit vector in Tcl representation, and returns it as the value of the command. This value can be saved and used to set the selection later on. For instance: **set myfrags [selection]**.

selection –round-up

"Rounds up" the selection to include entire fragments or residues. Since atom selection may well include only some of the atoms of a fragment or residue, this is provided to include the entire unit.

selection *bitvec*

Restores the selection to whatever it was when the bitvec was returned from an earlier **[selection]**.

selection -count
selection -count *bitvec*

Counts the currently selected atoms, or counts those in the selection bitvector *bitvec*. Also counts the number of protein or DNA/RNA residues, and the number of ligands, fragments, and chains. The returned value is a Tcl list of alternating keywords and numbers, of the form {atoms *nAtoms* chains *nChains* aminos *nAminos* nucleos *nNucleos* ligands *nLigands* fragments *nFragments*}.

selection –expr *selexpr*

Computes and returns a bitvector representing the logical combination of the bitvector values appearing in the expression *selexpr*, as follows.
There should be a round-up operator.

lhs* and *rhs

Takes those atoms selected in both *lhs* and *rhs*.

lhs* or *rhs

Takes those atoms selected in either *lhs* or *rhs*.

lhs* and-not *rhs

Takes those atoms selected in *lhs* and not *rhs*. Set subtraction.

lhs* xor *rhs

Takes those atoms in lhs or rhs but not both.

(*selexpr*)

The parentheses group expressions, as in ordinary algebra, so that you can use expressions as lhs or rhs in other expressions. Use parentheses in complex expressions to clarify your intent, because the precedence between the various operators is undefined.

Note that each parenthesis must be surrounded by spaces—they are Tcl "words", after all.

Further, note that when Tcl does $variable substitution, it considers the entire replacement value to be a single Tcl "word" (see the section *Tcl words*. Tcl values (strings, opaque objects, lists) work fine. However, scripting phrases won't work if substituted in the obvious way. This means, for instance, that **set sel "protein -chain A"; select {fragment or $sel}** won't work – Tcl will look up the entire string "protein -chain A" as a command name; of course, it won't find it. Perhaps you could use "arithmetic" on selection objects, construct an entire **select** command and use Tcl **eval**, or use the **{*}$sel** form which splices the string in **sel** in-line.