Jmol 12.0 new features – 7/27/2010

176. prompt() function

Jmol 12.0.RC27 adds the PROMPT function, which allows user-typed input or user response to option buttons.

- if (prompt("Do you want spacefill","Yes|No",true)=="Yes"){spacefill only}else{wireframe 0.15;spacefill 23%}
- x = prompt("Enter a script command",x);if (x != "null"){script inline @x}
- x = prompt("What sort of rendering do you want?",["Spacefill","Wireframe","Ball&Stick","cancel"], true);script inline @{(x == 1 ? "spacefill only" : x == 2 ? "wireframe only;wireframe reset;spacefill reset" : "")}

175. associative arrays

Jmol 12.0.RC27 completes the full-fledged scripting capability of Jmol by adding associative arrays -- arrays that allow you to retrieve information using a string-based key rather than a number. See the documentation for details.

- load 1crn.pdb;a={"area1":{helix},"area2":{sheet}}
- display @{a["area1"]}
- display @{a[prompt("What do you want to see?","area1|area2",true)]}

174. TRY/CATCH

Jmol 12.0.RC27 adds the try/catch option similar to that of JavaScript and Java, allowing you to catch script errors, including file load issues, BEFORE they trash your model.

- try{load "garbage"}catch(e){prompt @e}spin on # notice that we get the full error message and that the script continues on to start the model spinning

173. SWITCH/CASE

Jmol 12.0.RC27 adds one final piece of JavaScript-like scripting -- SWITCH/CASE. See the documentation for details.

172. PROMPT command

Jmol 12.0.RC26 adds the PROMPT command to allow pausing a script until the user presses OK. If no parameter is given, PROMPT returns a script stack trace showing where it is. This may be useful for debugging script files.

- load caffeine.xyz;refresh;prompt "caffeine is loaded"

171. FEATURE CHANGE: halos ON

Jmol 12.0.RC26 fixes a long-standing problem with halos and selection halos. With "halos ON" (the default size) they are supposed to track the spacefill size, appearing slightly larger. In previous versions their size was simply set to 20% of the default van der Waals radius. You can still set them to that if you want, with ***halos 20%****.*

170. wireframe/spacefill RESET

Jmol 12.0.RC26 adds the RESET option, which returns wireframe or spacefill to the Jmol default settings. Thus, a simple "Ball&Stick" rendering is simply ***wireframe only;wireframe reset;spacefill reset****. (The "wireframe only" removes any other rendering.)*

- load caffeine.xyz;wireframe only
- wireframe;delay 1;wireframe reset
- spacefill;delay 1;spacefill reset
- dots only;delay 1;wireframe only;wireframe reset;spacefill reset

169. 3D Jmol in PDF files

... well, ALMOST! Starting with Jmol 12.0.RC26, you can easily write the files necessary to add a Jmol-like 3D model into PDF files. To do this you are going to need a few pieces of software besides Jmol. The idea is Jmol --write--> IDTF + TEX --idtfConverter--> U3D + TEX --pdflatex--> PDF. (I think that equation is balanced!) It is a bit of a process, but it is not too bad. Once you get it working, the process takes a few seconds only. Those who are TeX savvy are at an advantage here, but if you have Adobe Acrobat PRO you should not need to know any TeX at all to do this. Sample files are in the pdf subdirectory. test1.idtf.pdf is an example.

168. full 3D-SEARCH SMILES/SMARTS and bioSMILES/bioSMARTS implementation

Jmol 12.0 implements a full version of SMILES and SMARTS matching, augmenting that with extensive capabilities for 3D conformation matching (distance, angle, and torsion ranges, as well as conformational RMSD comparison measurements of selected atoms) and additional extensions to search biomolecular substructure such as sequence and base-pairing. All aspects of Daylight SMARTS matching are implemented -- see the 3D-SEARCH specifications for details.

167. load DATA

Jmol 12.0.RC25 merges the DATA "model..." and DATA "append..." methods with the LOAD command, thus extending all of the options to the LOAD command to the DATA command.

- load data "model molfile"|line1|line2|line3| 5 4 0 0 0|0.0 0.0 0.00000000 C|0.0 1.0225207 0.00000000 H|0.8855288 -0.511260 0.00000000 H|-0.885528 -0.511260 0.00000000 H|1.0 0.0 0.00000000 Cl| 5 1 1| 4 1 1| 2 1 1| 1 3 1|end "model molfile" FILTER "2D" # Here we are adding 2D-to-3D minimization -- the vertical bars are only necessary here because of the JavaScript being used. Generally one would use new-line characters
- load "" # same file loaded again, this time with no 2D conversion

166. set dotScale

Jmol 12.0.RC24 adds the capability to adjust the size of the dots.

- load caffeine.xyz;dots on
- set dotScale 1
- set dotScale 2
- set dotScale 3
- set dotScale 4
- load 1crn.pdb;dots on
- set dotScale 1
- set dotScale 2
- set dotScale 3
- set dotScale 4

165. show SMILES

*Jmol 12.0.RC25 adds show SMILES; same as **print {selected}.find("SMILES")***

- load cholesterol.mol;show SMILES

164. MAPPROPERTY command

Jmol 12.0.RC23 adds a powerful new way to transfer properties from one atom set to another. The operation involves identifying two sets of atoms and properties and also a common "key" property such as atomno or resno. If no key is given, atomno is assumed. A shortcut allows quick transfer of atom selection.

- load 1crn.pdb;plot ramachandran
- mapProperty {1.1}.temperature {2.1}.property_t RESNO; color {2.1} property_t "bwr"
- selectionHalos on; select 1.1 and within(1, group, pro);mapProperty SELECTED {2.1}
- select {1.1};calculate straightness;mapProperty {1.1}.straightness {2.1}.property_s; color {2.1} property_S

163. {*}.find("SEQUENCE")

Jmol 12.0.RC23 allows quick display of sequence information in Jmol bioSMILES notation, either with or without cross-links.

- load 1crn.pdb
- show sequence # standard way
- print {*}.find("SEQUENCE") # new way
- print {*}.find("SEQUENCE", true) # with cross-links

162. new QuaternionFrame settings

Jmol 12.0.RC23 adds several new quaternion frame settings. See the Jmol 12.0 documentation for details.

161. PLOT command

Jmol 12.0.RC23 combines the quaternion and ramachandran commands into a new command, ***plot****, and adds to that the capability to graph any two or three atom properties. Optional parameters for the PROPERTIES option include MIN and MAX. All plots are scaled to fit a 200 x 200 x 200 angstrom box using a unit cell to convert actual (now "fractional") units to Jmol xyz units. This allows access still to the actual data coordinates via fractional coordinate notation (such as {2.0 3.0 5.0/1}).*

- load 1crn.pdb
- plot quaternion difference
- plot ramachandran
- zap !1.1;select *.CA;plot PROPERTIES phi psi resno
- zap !1.1;select *.CA;plot PROPERTIES phi psi resno MIN {-180 -180 0} MAX {180 180 30}
- zap !1.1;select *.CA;plot PROPERTIES phi psi structure; select visible; color structure;spin on;delay 2;spin off
- zap !1.1;select *;plot PROPERTIES atomno temperature
- select 1.1;plot PROPERTIES structure temperature atomno; select visible; color structure
- load 1d66.pdb
- select *.P;plot PROPERTIES eta theta resno;select 2.1;wireframe only # a "DNA worm"

160. eta/theta for nucleic acids

Jmol 12.0.RC23 adds nucleic acid conformation properties eta and theta as per Carlos M. Duarte, Leven M. Wadley, and Anna Marie Pyle, RNA structure comparison, motif search and discovery using a reduced representation of RNA conformational space, Nucleic Acids Research, 2003, Vol. 31, No. 16 4755-4761. eta is the C4'[i-1]-P[i]-C4'[i]-P[i+1] dihedral angle; theta is the P[i]-C4'[i]-P[i+1]-C4'[i+1] dihedral angle.

159. axes center {x y z}

Jmol 12.0.RC22 adds the capability to move the axes origin to a new center.

- load caffeine.xyz;axes on; axes molecular
- axes center {atomno=1}

158. set picking measure SEQUENCE

Jmol 12.0.RC22 adds the ability to pick two atoms of a biomolecule and have the sequence containing them displayed.

- load 1crn.pdb;set picking measure sequence # now start picking atoms

157. set cartoonBaseEdges

Jmol 12.0.RC21 introduces a new way to render nucleic acid cartoons. ***set cartoonBaseEdges TRUE*** *tells Jmol to displays nucleic acid bases as triangles that highlight the sugar edge (red), Watson-Crick edge (green), and Hoogsteen edge (blue). See Nasalean L, Strombaugh J, Zirbel CL, and Leontis NB in* ***Non-Protein Coding RNAs****, Nils G. Walter, Sarah A. Woodson, Robert T. Batey, Eds., Chapter 1, p 6.*

- load 1d66.pdb;display nucleic;reset;center {35.5123 34.30653 27.29455}; rotate z 140.14; rotate y 11.62; rotate z -138.18; zoom 380.63; set rotationRadius 51.55;cartoons only
- set cartoonBaseEdges true
- set cartoonBaseEdges false

156. isosurface ... MAP MEP functionType

Jmol 12.0.RC19 adds preliminary molecular lipophilic potential mapping by allowing MEP (molecular electrostatic potential) to be mapped using functions other than the standard Coulomb potential (1/d). The data for the mapping comes by default from the partialCharge atom property, but can easily be drawn from any property or a variable. See the Jmol 12.0 documentation for details.

155. measure search("...")

Jmol 12.0.RC19 adds a simple way to create measurements based on SMARTS searching.
- load caffeine.xyz
- measure delete; select *; measure search("Ocn") # measure the angles associated with the CO bonds
- measure delete; select *; measure search("cn") # measure all aromatic CN bonds

154. substructure() function deprecated

Jmol 12.0.RC19 deprecates the substructure() function, which used a SMILES string to search for substructure. Really, SMILES strings were never intended to be for searching; that is what SMARTS is for. The find() and search() functions are recommended.
- load caffeine.xyz;selectionhalos on; select none
- select search("ccc")
- select search("{c}cO")
- print {*}.find("[r5]")
- print {*}.find("SMILES", "Cn1[c]([O])[c]2n(C)[cH][n][c]2n(C)[c]([O])1") # a SMILES string for caffeine -- using "SMILES" checks for a whole-structure match

153. atropisomer SMARTS matching

Jmol 12.0.RC16 adds a new bond symbol to SMILES/SMARTS C^C and C^^C or C!^C (SMARTS) -- atropisomer (dihedral angle) check

152. x.find("SMILES"/"SMARTS","MF")

Jmol 12.0.RC16 lets you quickly find the molecular formula of the SMILES or SMARTS string associated with a set of atoms or SMILES string. The two options allow for a full all-hydrogen formula or one without hydrogen atoms.

151. load $CCC(C)C -- smiles string loading

Jmol 12.0.RC15 will read SMILES strings into 3D using the smi23d server at Indiana University. Only catch is that since this is a third-party server, you will have to use the signed applet to do it. Note that this service uses PCMODEL v9.1 for its conversion.
- load SMILES "C1C(C)C[C@H](F)C1C(=O)C"
- load $CC1CCCC1(C)C
- load "$[H][C@@]12CCCC1[C@@H](C)[C@H](CC)[C@]3(O)CCCC[C@@]23[H]" # from JME
- load SMILES "F/C=C\C=C\C" # correct
- load SMILES "F/C=C\C=C/C" # should be different
- load SMILES "CCCC" as "butane.mol" # save file on your local drive

150. set HIGHLIGHT

*Jmol 12.0.RC15 adds a new sort of selectionHalo. The HIGHLIGHT is a just a small ring around an atom. The default color is red, but you can use **color highlight** to set it to a different color. It is not an "atom property", so individual atoms cannot have their own colored highlight.*

- load caffeine.xyz
- set highlight {_O}
- color highlight yellow
- set highlight off

149. set ModelKitMode and set allowModelKit

*Jmol 12.0.RC15 represents a first attempt at creating a model kit option for Jmol. I cannot seem to get the menu to pop up using "set modelKitMode" via Javascript, but you can call it up yourself from the menu. It is under "computation". Using **set allowModelKit FALSE** you can disallow modelKitMode.*

- set modelKitMode;zap # the magenta bar in the top left corner is the menu

148. FIX command

The FIX command takes an atom expression argument like the display or select commands. It fixes the positions of atoms and ensures that no atoms of this set will be moved or dragged anywhere accidentally.

147. set picking invertStereo

*Starting with Jmol 12.0.RC15, selecting an atom that is part of a ring after **set picking invertStereo** will reverse the two non-ring atoms -- actually rotating them 180 degrees, not doing a planar inversion, thus preserving whatever chirality might be attached to them.*

146. set picking dragMinimizeMolecule -- responsive docking

With Jmol 12.0.RC15 you can move a small molecule around and watch it react to its environment. Grab the caffeine molecule and move it toward the protein. Holding SHIFT down allows rotation. CTRL-Z undoes an action; CTRL-Y does a redo.

- load files "caffeine.xyz" "1crn.pdb";frame *;zoomto {1.1} 0;set picking dragMinimizeMolecule
- load files "1crn.pdb" "water-AM1.sparchive";frame *;isosurface select {1.1} sasurface 0 frontonly translucent 0.3;fix protein;select {2.1}; spacefill;set picking dragMinimizeMolecule # now drag the water around the surface

145. set picking dragMinimize

Wouldn't it be fun to have a real "sculpture" mode, where you can play with a molecule like an artist does with clay, molding it to the shape you want -- within the limitations of the medium? (Don't try this with a protein.) Well, it's very possible this is a bad idea, but Jmol 12.0.RC13 adds this capability.

- load mecy.jme;set picking dragMinimize;set minimizationSteps 500;set echo top left;echo drag an atom...;set echo bottom left;echo @{_minimizationEnergy} *just grab atoms and move them. Or click on an atom to further minimize. I know...quesionable pedagogy...Actually, there is a sort of lesson here in how minimization works...*

144. set picking dragAtom

*Jmol 12.0.RC13 adds **set picking dragAtom**, allowing dragging of atoms to new locations. Attached hydrogens atoms are dragged along with the selected atom.*

- load caffeine.xyz;set picking dragAtom; # go ahead...

143. drag-and-drop to signed applet and from browsers

*Jmol 12.0.RC13 adds drag-and-drop capability for the signed applet and adds MULTIPLE-FILE drag-and-drop for both signed applet and application. Just fire up a page such as drop.htm that uses the signed applet and then drag from a directory listing into the applet as many files as you like. They will be loaded into different frames, and **frame *;reset** will be issued so that they are all displayed. (PNG and JPG images*

created by Jmol can be loaded this way, but only one at a time, since they reset the state when loaded.) For standard model files, if ***set defaultLoadScript*** *is defined, then that script will be executed. (For example, a simple one might be* ***set defaultLoadScript 'select protein or nucleic;cartoons only'****.) Electron density MAP files may also be dropped in. Or you can simply clip file name(s) in a file directory, then CTRL-V into Jmol loads the file(s).*

142. COMPARE with SMILES or SMARTS allows conformational testing and alignment

Jmol 12.0.RC12 adds the capability to align two structures based on SMILES or SMARTS atom matching. The basic idea is to use a SMILES (whole molecule) or SMARTS (substructure) description to find the atoms in one structure that correlate one-for-one with atoms in the second structure, then find the rotation and translation that best aligns them. If no actual atom moving is desired, you can get the standard deviation alone using the compare() function with the "STDDEV" option. A return of "NaN" indicates that the desired SMILES/SMARTS match could not be made in one or the other structure.

- "2-ClBu.spt" # this script loads two versions of (R)-2-chlorobutane an aligns them
- load inline "5 4 C 5.88 -4.59 C 7.09 -3.89 C 8.30 -4.59 C 4.73 -3.79 Cl 7.09 -2.49 1 2 1 2 3 1 1 4 1 2 5 -2";load append "2-ClBu.mol";frame *;select *;wireframe only;label %[atomno];moveto /* time, axisAngle */ 1.0 { 334 -913 -234 46.04} /* zoom, translation */ 57.58 0.13 0.13 /* center, rotationRadius */ {3.205987 -2.11722 -0.18840191} 4.361869 /* navigation center, translation, depth */ {0.0 0.0 0.0} -14.783268 -18.059193 0.0;
- print compare({2.1},{1.1},"SMILES","CC(Cl)CC","stddev"); # just getting the standard deviation
- print compare({2.1},{1.1},"SMILES","C[CH@](Cl)CC","stddev"); # not this enantiomer
- print compare({2.1},{1.1},"SMILES","C[CH@@](Cl)CC","stddev"); # there you go
- compare {2.1} {1.1} SMILES "CC(Cl)CC" rotate translate
- select 2.1;color labels yellow;set labeloffset -5 10;color yellow;zoomTo {visible} 0
- script "2-ClBu.spt" lines 1-2;reset;frame *;compare {2.1} {1.1} SMARTS "[CH3][CH@@](Cl)[CH2]" rotate translate # alignment based just on the stereocenter

141. JmolSmilesApplet.jar

Jmol 12.0.RC11 adds a new light-weight applet (only 43K) that checks SMILES strings. This is particularly useful for comparison of drawn stereochemistry, which is not possible using JME alone. See JmolSmiles.htm, JmolSmilesTest.htm, and JmolSmilesApplet.jar."

140. SMILES stereochemistry matching

Jmol 12.0.RC11 allows unprecedented matching of non-canonical SMILES strings independent of any 3D structure. This includes both atom and bond stereochemistry, including cis/trans, allene, tetrahedral, square planar, trigonal bipyramidal, and octahedral stereochemistry. Starting with a SMILES string from some source, you can test for an equivalent structure WITHOUT any need for "canonicalization" of the SMILES (that is, turning it into some standard form). While canonicalization is important for database searching just in terms of speed, with this addition to Jmol, ***canonicalization is no longer necessary for pattern matching between two SMILES strings or pattern searching of a SMILES string using a SMARTS pattern.*** *This feature is completely independent of any actual Jmol model (although that, too, can be tested against a SMILES or SMARTS string). Thus, for example, you can have a user create a 2D structure in JME and use Jmol to test its equivalence to some reference SMILES or SMARTS string you are expecting.*

- print "CCCO[C@](F)(Cl)I".find("smiles","O(CCC)[C@](F)(Cl)I")
- print "O[C@](F)(Cl)I".find("smiles","[C@](O)(F)(Cl)I")
- print "O[C@](F)(Cl)I".find("smiles","[C@@](O)(Cl)(F)I")
- print "OC(Cl)=[C@]=C(C)F".find("smiles","OC(Cl)=[C@AL1]=C(C)F")
- print "OC(Cl)=[C@]=C(C)F".find("smiles","OC(Cl)=[C@AL2]=C(F)C")
- print "F[Po@SP1](Cl)(Br)I".find("smiles","F[Po@SP1](Cl)(Br)I")
- print "F[Po@SP1](Cl)(Br)I".find("smiles","F[Po@SP2](Br)(Cl)I")
- print "F[Po@SP1](Cl)(Br)I".find("smiles","F[Po@SP3](Cl)(I)Br")
- print "S[As@@](F)(Cl)(Br)C=O".find("smiles","S[As@@](F)(Cl)(Br)C=O")
- print "S[As@@](F)(Cl)(Br)C=O".find("smiles","O=C[As@](F)(Cl)(Br)S")
- print "S[Co@@](F)(Cl)(Br)(I)C=O".find("smiles","S[Co@@](F)(Cl)(Br)(I)C=O")
- print "S[Co@@](F)(Cl)(Br)(I)C=O".find("smiles","O=C[Co@](F)(Cl)(Br)(I)S")

- print "F/C=C/F".find("smiles","F/C=C/F")
- print "F/C=C/F".find("smiles","F\\C=C\\F")
- print "C(/F)(Cl)=C/F".find("smiles","C(/Cl)(F)=C\\F")
- print "C(/F)(Cl)=C/F".find("smiles","C(\\F)=C(/Cl)F")

139. {*}.find("smartsString",asArray)

Jmol 12.0.RC11 adds the SMARTS searching to the find() function. If the optional second parameter is TRUE, the return is an array of atom sets.

- load caffeine.xyz
- print {*}.find("cn")
- print {*}.find("cn", true)

138. load @x where x is an array of file names

Jmol 12.0.RC11 adds the capability to load a set of files that are defined in an array variable.

137. select search() -- 3D-SMARTS

*Jmol 12.0.RC11 adds a completely new way to select atoms based on the SMARTS chemical informatics search language. **3D-SEARCH** is an almost exact implementation of SMARTS, differing primarily in how 3D-SEARCH defines "aromatic" and also adding a "search and select" capability as opposed to just "search." The search language allows very precise selection of atoms based on connectivity. Examples include "all alpha carbons" **{C}C=O**, "all meta-related groups" **{[A&!H]}aaa{[A&!H]}**, "all trans double bonds" ***/{C=C}/***, and "all biphenyl connecting atoms" **a:a**. All stereochemical capability of SMARTS described on that page is implemented, including double-bond, allenic, tetrahedral, square planar, trigonal pyramidal, and octahedral stereochemistry. "Aromatic" is defined simply as any atom in a flat ring that has all sp2 hybridization. This is a compromise definition meant to be inclusive of some compounds that by Hueckel analysis would not be considered aromatic. In addition, setting { } around an atom or group of atoms specifies to select only those atoms, and an optional second parameter allows selecting atoms within a specific subset of atoms.Variables may be assigned within the SMARTS string (which unlike the specification at the Daylight site does allow white space and comments). Just a few examples are shown below.*

- load ketone.jme
- selectionhalos on;select search("{C}C=O") # just the alpha carbons
- selectionhalos on;select search("{[C&!H0]}C=O") # just the alpha carbons with H atoms
- selectionhalos on;select search("{[A&!H]}aaa{[A&!H]}") # just the meta groups
- load caffeine.xyz;selectionhalos on;select none
- select search("a") # all "aromatic"
- select search("[nD2]") # aromatic N with only two bonds
- select search("[Od1]") # oxygen with only one non-hydrogen connection
- select search("[cv3]") # trivalent aromatic carbon
- select search("{[cv3]}O") # trivalent aromatic carbon attached to oxygen
- select search("[cr5]") # aromatic carbon in a five-membered ring
- select search("[r6]") # six--membered ring
- select search("[r9]") # nine-membered ring
- select search("[R2]") # in two rings (ring-fusion atoms)
- select search("[!r&!H]") # non-ring non-hydrogen atoms
- select search("{[!r&!H]}*[R2]") # non-ring non-hydrogen atoms two atoms from a ring-fusion
- select search("[$($([!r]a) ** [R2]) & ! $(**[R2])]") # non-ring atoms attached to an aromatic ring and three atoms from a ring-fusion but not two atoms from a ring fusion atom
- select search("c",search("[r6]")) # only aromatic carbons in 6-membered rings
- select search('$Aro1="[a&!$(aC)]" (i.e. aromatic not attached to aliphatic carbon); $CarbonylO="O";{[$CarbonylO]}[$Aro1]')

136. Spartan/Cygress FILTER "noOrient"

Jmol 12.0.RC10 adds the noOrient filter for loading Spartan and Cygress (CAChe) files. Generally these files are loaded with a default orientation that was the orientation when the file was saved.

135. isosurface "=nnnn"

Jmol 12.0.RC10 adds the capability to automatically load electron density maps from the Uppsala Electron Density Server into the application and signed applet. Supporting this are the global variables edsUrlCutoff and edsUrlFormat, which set the method of getting cutoff and file from electron density server.

134. isosurface lattice {a b c}

Jmol 12.0.RC9 adds the capability to duplicate isosurface areas based on the unit cell lattice. This is a rendering option, so it can be applied any time after an isosurface is created. It is best done with packed unit cells.

- load quartz.cif {1 1 1};zoom 75;center {3/2 3/2 3/2}
- isosurface slab unitcell vdw
- isosurface lattice {3 3 3}

133. wireframe ONLY and wireframe -x.y

Jmol 12.0.RC7 adds two new option for wireframe, stars, halos, spacefill, cartoons, etc. "ONLY" removes all other atom rendering; "-x.y" does the same, but allows setting of the radius.

- load 1crn.pdb
- spacefill only
- cartoons only
- wireframe -0.2

132. PARALLEL MULTIPROCESSING

Jmol 12.0.RC6 introduces parallel processing for Jmol. Jmol 12 will be able to use multiple processors on a multiple-CPU machine. Basically what you can do is to tell Jmol which statements in a script you want to run in parallel, and it will do that. The way this is done is to create a function using the keyword ***PARALLEL*** *in place of the keyword* ***function****. Within that block of code, any group of commands surrounded by* ***PROCESS{ }*** *will be collected and run in parallel just before Jmol returns from the function. Any commands NOT within these sets will be run BEFORE any* ***PROCESS*** *commands. For example:*

```
parallel twoIsosurfaces(model1, model2) {
  var x = 1
  process {
    isosurface s1 model @model1 molecular; color isosurface red
  }
  process {
    isosurface s2 model @model2 molecular; color isosurface green
  }
  x = 2
}

load files "1crn.pdb" "1blu.pdb"
twoIsosurfaces("1.1", "2.1")
frame *
```

In this case, the variable x will be 2 BEFORE the isosurfaces are created. See also multi-mo.txt, multi-surface.txt, and multiProcessTest.txtmultiProcessTest.txt. You can selectively turn on and off the use of multiprocessors using ***set multiProcessor****. If this setting cannot be set true, then it means you do not have a multiprocessor machine. Not all processes will work; currently the only implemented parallel processes are for isosurfaces and molecular orbitals. The parallel capability of Jmol should be considered experimental at this time.*

- set multiProcessor false;script multi-mo.txt
- set multiProcessor true;script multi-mo.txt
- set multiProcessor false;script multi-surface.txt # be patient!
- set multiProcessor true;script multi-surface.txt # somewhat faster on my machine

131. set minimizationSilent and minimize SILENT

*Jmol 12.0.RC5 adds the option to make minimization totally silent. The default value is FALSE. In association with **set useMinimizationThread FALSE**, this option is used automatically to effect the loading of 2D JME strings into Jmol as 3D objects directly. Note that in some cases, more than the default **set minimizationSteps 100** may be necessary for a satisfying result.*

- load test.jme
- load test.jme FILTER "NOMINIMIZATION";wireframe only # the unminimized structure - note that Jmol has interpreted some bond angles as representing sterochemistry.
- load stereo.jme # delay is due to processing the minimization

130. load INLINE "JME string" and load "@x"

Jmol 12.0.RC5 allows inline load of data that have no new-line characters (JME strings) using load INLINE, and any model using load "@x", where x is a Jmol variable containing the model. Note that JME models are automatically turned into 3D.

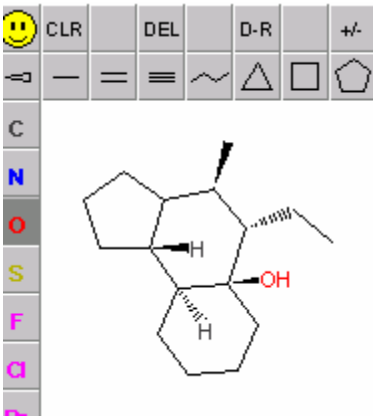
- load inline "5 4 C 5.88 -4.59 C 7.09 -3.89 C 8.30 -4.59 C 4.73 -3.79 Cl 7.09 -2.49 1 2 1 2 3 1 1 4 1 2 5 -2";
- x = load("ch3cl.mol");load "@x"

129. select CYSTINE

Jmol 12.0.RC5 adds select CYSTINE synonymous with within(group, cys.sg and connected(cys.sg))

- load 1crn.pdb;select none;selectionhalos on
- select CYSTINE

128. 2D MOL reader to 3D



Jmol 12.0.RC5 adds the FILTER "2D" option to read MOL files that have all zero z coordinates and automatically minimize the structure into 3D, as with JME, below.

- load flat.mol;rotate X 80 # still 2D
- load flat.mol FILTER "2D";rotate X 80

127. JME reader reads stereochemistry and automatically turns 2D to 3D with addition of H atoms

*The JME format is a two-dimensional file format that is very easy to create using, for example, **JmeToJmol.htm** but not easily adapted to 3D (until now!). Jmol 12.0.RC5 adds the automatic addition of H atoms and transformation of simple JME structures (including stereochemistry) to 3D. The 2D-to-3D conversion can be prevented using the FILTER "noMinimization" option. The transfer is not perfect, and it may take some testing to get this right.*

- load stereo.jme
- load stereo.jme FILTER "NOMINIMIZATION";

126. select within(SEQUENCE,"1-letter-code sequence")

Jmol 12.0.RC5 allows selecting groups that are in groups that are within a specific sequence such as "GCAUGGC".

- load 1d66.pdb
- display within(SEQUENCE,"GAC")

125. within(nResidues, GROUP, {atoms})

Jmol 12.0.RC5 will allow selection of all groups within a given number of residues of a specified set of atoms.

- load 1d66.pdb
- display within(3,GROUP,within(SEQUENCE,"GAC"))

124. show BASEPAIRS

Jmol 12.0.RC5 will display a table of base pairs.

- load 1d66.pdb
- show basepairs

123. select within(nResidues, GROUP, atoms)

Jmol 12.0.RC5 adds the capability to select atoms within a certain number of residues along a chain of a specified group of atoms. This feature works for nucleic acids, proteins, and carbohydrates.

- load 1crn.pdb;cartoons only
- display within(HELIX) # just the helices
- display within(3,GROUP,within(HELIX)) # within 3 groups of a helix
- load 2qnh.pdb.gz;zoom 300;wireframe -0.4;display rna;select *.P and within(BASEPAIR,"AC");label %n%r;set picking center
- display within(BASEPAIR,"AC") # all AC pairs
- color {within(BASEPAIR,"AC")} white; display within(5, GROUP, within(BASEPAIR,"AC")) # within 5 residues of the AC pairs

122. select within(BASEPAIR)

Jmol 12.0.RC5 adds the capability to select atoms based on RNA or DNA base pairing. Thank you, Neena Grover!

- load 2qnh.pdb.gz;zoom 300;wireframe -0.4;display rna;select *.P;label %n%r;select rna;set rangeSelected;
- color property polymer;set picking center;slab on;depth 40;slab 60
- display within(BASEPAIR,"GC") # just the GC pairs
- display within(BASEPAIR,"AU") # just the AU pairs
- color cpk;calculate hbonds {displayed} {displayed};hbonds 0.3;color hbonds yellow;set hbonds sidechain
- display rna and !within(BASEPAIR) # bases that are not paired
- display within(BASEPAIR) and !within(BASEPAIR,"GCAU") # noncanonical base pairs

121. negative sizes implies ONLY

Jmol 12.0.RC4 reads negative decimals for cartoons, wireframe, etc. as "ONLY" at the corresponding positive size in angstroms

- load 1crn.pdb
- cartoons -0.5
- trace -0.3
- wireframe -0.2

120. better RNA hydrogen bond calculation

Jmol 12.0.RC4 extends and improves the canonical (purine-pyrimidine) hydrogen bond calculation

- load 2qnh.pdb.gz;zoom 300;backbone -0.5;display rna;select rna;set rangeSelected;color property polymer
- calculate hbonds {rna} {rna};hbonds 0.2;color hbonds yellow;set hbonds backbone

119. select RANGESELECTED extended

Jmol 12.0.RC4 extends the RANGESELECTED parameter to all properties, not just temperature.

- load 2qnh.pdb.gz;cartoons only
- select rna;set rangeSelected;color property polymer

118. generalized hydrogen bond calculation

Jmol 12.0.RC3 will calculate hydrogen bonds of the standard sort -- connecting H atoms with other atoms. See the documentation for details. In short, ***calculate HBONDS*** *will behave as previously with standard no-hydrogen PDB files -- creating "pseudo-hydrogen" bonds between O and N atoms. However, with PDB*

*files having hydrogen atoms and other files with hydrogen atoms, the **calculate HBONDS** command will generate standard hydrogen bonds from hydrogen atoms on O or N to any other atoms. If it is desired to have RasMol-like behavior even in the case of PDB files with no H atoms, one can simply **select not hydrogens**. The two settable parameters **hbondsAngleMinimum** (default value 90) and **hbondsDistanceMaximum** (default 3.25 for pseudo-hydrogen bonds; 2.5 for standard hydrogen bonds) can be adjusted to suit.*

- load maleic.cif 5;zoom 200
- calculate hbonds
- load ice.pdb;rotate z -72.64; rotate y 72.56; rotate z 88.29; translate x 0.13; translate y 0.13;
- calculate hbonds
- load Water!Liquid.xodydata
- calculate hbonds # note the missing H-bonds
- set hbondsAngleMinimum 75;select *;calculate hbonds;set hbondsAngleMinimum 90 # better in this case
- load 1cdr.pdb 1; rotate z -0.78; rotate y 54.15; rotate z 1.69; zoom 400.0;display sheet;wireframe only
- hbonds delete; select displayed;calculate hbonds;hbonds 0.2 # new
- hbonds delete; set hbondsRasmol TRUE; select displayed and not hydrogen; calculate hbonds;color hbonds yellow;hbonds 0.2 # RasMol version
- hbonds delete; set hbondsRasmol FALSE;select displayed and not hydrogen; calculate hbonds;hbonds 0.2 # Jmol version -- not just backbone
- hbonds delete; set hbondsDistancemaximum 3.8;set hbondsRasmol FALSE;select displayed and not hydrogen; calculate hbonds;hbonds 0.2 # Jmol version -- not just backbone

117. invertSelected STEREO option

*Jmol 12.0.RC2 adds the STEREO option to the **invertSelected** command. This option rotates branches connected to a center by 180 degrees around a line connecting that center and the geometric center of specified connected atoms not to invert. Simply put, thus carries out the standard organic chemist's "stereochemical inversion" at that point -- which does not invert stereochemistry along the branches, but instead switches two connected branches by simple rotation only.*

- load ala_5_180_0.pdb; select {atomno=12};halo 1.0;color halo yellow; reset;center {1.3850001 1.8339999 -0.65849996}; rotate z -6.42; rotate y 47.52; rotate z 80.26; translate x 0.13; translate y 0.13; connect {_O} {_C} double modify;set dragselected;set minimizationSteps 25;minimize addhydrogens;select H40;color yellow;
- invertSelected STEREO {atomno=12} {atomno=15 or atomno=38}

116. new ROTATE TRANSLATE {x y z} option

Jmol 11.9.37 extends the ROTATE command to include rotation/translation -- that is, helical or screw motion.

115. new ROTATE HELIX option

Jmol 11.9.37 adds the HELIX option to the rotate command. Given a 4x4 matrix or a combination of translation and rotation, Jmol will determine the helical (screw-motion) path instead of the direct-line path of motion. Notice that the SELECTED parameter is unnecessary when in this context because the group being moved (model 2.1) is listed explicitly after the definition of the motion.

- load 3cc2_900-1100.pdb;color orange;cartoons only;select 936-941 or 1025-1034;color red;load append 3cc2_Kt7.pdb; select 2.1;color yellow;cartoons only;frame *;reset;center {42.475437 88.69623 89.13157}; rotate z 76.29; rotate y 129.77; rotate z 39.64; zoom 268.05; set rotationRadius 114.69;;save coord sc;kt7coord = {2.1}.xyz.all; m = compare({2.1 and spine and 77-82}, {1.1 and spine and 936-941});
- restore coord sc; rotate {2.1 and spine and 77-82} @m {2.1} -3 # ...straight in
- restore coord sc; rotate HELIX {2.1 and spine and 77-82} @m {2.1} -3 # ...along a helical path
- rotate COMPARE {2.1} @kt7coord -3 # ...and back to the original position

114. new ROTATE COMPARE option

- load 3cc2_900-1100.pdb;color orange;cartoons only;select 936-941 or 1025-1034;color red;load append 3cc2_Kt7.pdb;select 2.1;color yellow;cartoons only;frame *;reset;center {42.475437 88.69623 89.13157};

rotate z 76.29; rotate y 129.77; rotate z 39.64; zoom 268.05; set rotationRadius 114.69;;save coord sc;kt7atoms = {2.1 and spine and 77-82};kt38atoms = {1.1 and spine and 936-941};kt7coords = {2.1}.xyz.all; kt7ac = kt7atoms.xyz.all;m = compare(kt7atoms, kt38atoms);

- restore coord sc; compare {2.1} {1.1} @kt7atoms @kt38atoms rotate translate -2
- restore coord sc; compare {2.1} {1.1} @kt7atoms @kt38atoms rotate translate
- rotate COMPARE {2.1} @kt7coords -2
- restore coord sc; rotate compare @kt7atoms @kt38atoms {2.1}
- select {2.1}; restore coord sc; rotate selected compare @kt7atoms @kt38atoms -2

113. new ROTATE (matrix variable)

Jmol 11.9.37 adds the capability to define rotations in the ROTATE command as 3x3 or 4x4 matices. In the case of a 4x4 matrix, the indicated translation will also be conducted along with the rotation to provide a screw motion; in the case of a 3x3 matrix, one can use the new TRANSLATE option for the ROTATE command to also apply a translation separately.

112. new COMPARE function

*Jmol 11.9.37 adds the **compare** function. The function takes either two sets of atoms or two arrays of atom positions (or any other sort of positions) and returns a 4x4 matrix **m** indicating the rotation (**m%1**) and translation (**m%2**) required to BEST transform set 1 into set 2. Unlike the COMPARE command, no filtering is done -- every atom or position in set1 is compared with every position in set2. An optional third parameter "stddev" allows return of just the RMSD for this comparison.*

- load files "3CC2_HmKt7.pdb" "3CC2_HmKt38.pdb";frame *;cartoons only;color {1.1} red;color {2.1} yellow
- m = compare({2.1 and spine and 936-941}, {1.1 and spine and 77-82}); show m
- print compare({2.1 and spine and 936-941}, {1.1 and spine and 77-82}, "stddev")
- print m%1
- print m%2

111. new 4x4 matrix functions

Jmol 11.9.37 adds %1 and %2 modulus for 4x4 matrices: m%1 returns the 3x3 rotation matrix, and m%2 returns the translation vector. Note that quaternion(m%1) then returns a quaternion equivalent of the rotation matrix, which has its own % operations.

110. COMPARE command animation

With Jmol 11.9.37, the COMPARE command adds the option to indicate the number of seconds to use to animate the comparison. This is primariy just for WOW effect. But associated with new ROTATE SELECTED options, can be reversed.

- load 3cc2_900-1100.pdb;color orange;cartoons only;select 936-941 or 1025-1034;color red;load append 3cc2_Kt7.pdb;select 2.1;color yellow;cartoons only;frame *;zoomto 2 {*} 0 # the two together in their native orientations.
- compare {2.1} {1.1} SUBSET {spine} ATOMS {77-82} {936-941}, {92-93 or 97-100} {1025-1026 or 1031-1034} ROTATE TRANSLATE 3

109. ROTATE options

*Jmol 11.9.37 introduces a number of new options for the ROTATE command. First, **rotateSelected** is deprecated. The SELECTED option for the **rotate** command takes care of this need. More significantly,*

108. rotateSelected deprecated

*Jmol 11.9.37 deprecates the **rotateSelected** command. Just add the **SELECTED** option to the **rotate** command.*

107. translateSelected deprecated

*Jmol 11.9.37 deprecates **translateSelected**. Simply add the **SELECTED** option to the **translate** command.*

106. Natural Bond Orbitals

NBO PLOT output may be visualized by Jmol. Starting with version 11.9.36, Jmol can LOAD any of the files xx.31 - xx.41 as well as the ASCII GenNBO output file xx.48 (sometimes referred to as the "xx.nbo" file). These files are generated by adding PLOT to any NBO parameter list ($NBO $END), which may appear in Gaussian, GAMESS, Jaguar, Psi, QChem, or GenNBO input decks. The xx.48 file may be used as input for visualization specifically if AONBO=P is added to the input parameters. (This visualization is equivalent to that of xx.37 files -- NBO orbitals in AO basis.) Any of the files xx.31 - xx.41 may be used in the LOAD command (or drag-dropped into the Jmol application) provided the necessary xx.31 and xx.46 files are also present in the same directory. Note that JPG and PNG files create by Jmol using the WRITE command and stored in the directory of the .nn files contain a full state description along with the image and can be loaded or dragged into Jmol the same as the original .nn file, with the added benefit of recreating the exact 3D model state as shown in their image.

- load t.37 # also reads t.31 (atomic orbital basis set) and t.46 (orbital labels)
- mo homo
- mo homo - 1
- mo homo - 2

105. translucent color schemes

Jmol 11.9.35 adds a translucent option for color schemes. The idea is that a color scheme can go from "tranparent" to "opaque" while it goes from one color to another.

- load 1crn.pdb;cartoons only;color straightness
- color "BW" TRANSLUCENT;color property straightness
- load 3hyd.pdb;reset;center {16.626621 2.7783334 1.7281427}; rotate z 100.36; rotate y 112.6; rotate z -72.02; zoom 200; translate x 0.13; translate y 0.13;boundbox scale 1.2 {tyr};isosurface boundbox SIGMA 1 color DENSITY colorscheme "BW" TRANSLUCENT "3hyd_map.ccp4.gz"

104. property SELECTED

*Jmol 11.9.35 adds SELECTED to the list of properties of atoms that can be checked, averaged, listed, and set. The value of the property is 1.0 if an atom is selected and 0.0 if not. As for all properties, the value for a set of atoms is the average, which in this case is the fraction of atoms selected. The example used here uses **{*}.selected** to get the fraction of all atoms selected and displays that in an echo at the top left.*

- load caffeine.xyz;selectionhalos on;select none; select carbon;set picking select;set echo top left; echo @{"" + (100*{*}.selected)%1 + "% of the atoms are selected"} # click on atoms to change the selection.

103. new COMPARE command

*Jmol 11.9.35 introduces the capability to compare two models and to reorient one model relative to another based on a given atom-atom coordinate pairing or quaternion-based group-group orientation pairing. References to the atom-atom correlation algorithm can be found in the literature [1] and [2]. Quaternion-based orientation pairing is an unpublished technique specific to Jmol at this point. It minimizes the standard deviation of the correlated quaternion frames for groups in the two models using spherical averaging. (Results of this option depend upon the setting of **set quaternionFrame**.)*

By default the command does not move any atoms and just reports RMSD. The independent options ROTATE and TRANSLATE allow the option to do just rotation, do just center-of-mass translation, or do both. The basic command involves specifying at the very least two models. If no other parameters are included, the models are matched atom-for-atom based on "SPINE" atoms (or, with orienational comparisons, group by group). A subset of atoms for pairing other than SPINE, such as {.CA} can be specified using the SUBSET option. Finally, any number of atom sets to be correlated atoms can be given. The result of atom-atom pairing comparison is essentially the same as the PyMol pair_fit command, though easier to implement and using an exact form of the structure-structure correlation rather than an iterative process.*

Given here is a comparison that can be found at the Lilley group Kink-Turn site comparing the Kt-38 site to the Kt-7 site.

- load 3cc2_Kt7.pdb;color yellow;cartoons only # the kink-turn we will compare
- load 3cc2_900-1100.pdb;color orange;cartoons only;select 936-941 or 1025-1034;color red # A section of the 23S ribosome subunit containing a similar kink-turn, highlighted in red
- load 3cc2_900-1100.pdb;color orange;cartoons only;select 936-941 or 1025-1034;color red;load append 3cc2_Kt7.pdb;select 2.1;color yellow;cartoons only;frame *;zoomto 2 {*} 0 # the two together in their native orientations.
- moveto /* time, axisAngle */ 1.0 { 386 700 -600 176.39} /* zoom, translation */ 500.0 0.26 0.26 /* center, rotationRadius */ {25.776789 71.75962 74.88684} 90.03123 /* navigation center, translation, depth */ {0.0 0.0 0.0} 82.41045 -549.58514 0.0;depth 40;slab 60; slab on;
- compare {2.1} {1.1} SUBSET {spine} ATOMS {77-82} {936-941}, {92-93 or 97-100} {1025-1026 or 1031-1034} ROTATE TRANSLATE # best correlation of atoms
- set quaternionFrame "C"; compare {2.1} {1.1} SUBSET {spine} ORIENTATIONS {77-82} {936-941}, {92-93 or 97-100} {1025-1026 or 1031-1034} ROTATE TRANSLATE # best correlation of base orientations

102. SPINE predefined set

*Jmol 11.9.34 adds SPINE to the list of predefined sets that can be used in Jmol. Spine includes just the atoms that form the continuous backbone set: *.CA, *.N, *.C for proteins; *.P, *.O3*, *.O5*, *.C3*, *.C4*, *.C5 for nucleic acids. Phosphate oxygens OP1 and OP2 are not included but can be added using* ***spine or *.OP1 or *.OP2****.*

101. set quaternionFrame "A", "C", and "P" for nucleic acids

quaterion frames are predefined orientations used by Jmol to compare relative or absolute orientations in space. Frame "A" was introduced in Jmol 11.8.1 and relates one alpha carbon to the next along a protein chain. In Jmol 11.9.34, this is extended to the phosphorus atoms along a nucleic acid chain. Specifically, this frame allows calculation and display of straightness and identification of secondary structure in models that only contain alpha carbons and phosphorus backbone atoms. Frame "C" is a frame that indicates the orientation of the sidechain of a protein in relation to the backbone. In Jmol 11.9.34 this is extended to purines and pyrimidine bases. Frame "P" for proteins indicates the absolute orientations of the peptide planes. Jmol 11.9.34 extends this to nucleic acids, indicating the absolute orientations of the phosphorus backbone tetrahedra.

100. quaternion arrays, differences, means, and standard deviations

Quaterions can be used in Jmol to characterize the orientation of groups. In particular, Jmol can assign quaternion frames to amino acid and nucleic acid residues automatically. This suggests the possibility of manipulating arrays of quaternions and investigating their properties. Jmol 11.9.35 adds the capability to create arrays of quaternions, find the mean, and determine their standard deviation. In addition, Jmol 11.9.35 adds the capability to calculate the quaternion differences of an array of quaternions. Quaternion differences are intersting in the area of protein-protein alignment and in quantifying the regularity of structure in helixes, strands, and other secondary structure motifs.

- load 1crn.pdb;cartoons only;select within(helix);x = quaternion({selected})
- print x # an array of quaternions
- print quaternion(x) # mean quaternion
- print x.average # mean quaternion
- print x.stddev # standard deviation
- dq = quaternion({8-18}, {9-19});print "helicalAxis/angle = " + (dq.average)%-6 + "\nhelicalError = " + dq.stddev
- draw scale 10.0 VECTOR {8-18.CA} @{(dq.average)%-1}

99. webExport enhancements

Jmol 11.9.33 adds widgets in WebExport pages (background colorpicker, spin on/off, stereo mode)

98. xy**

*Jmol 11.9.33 adds exponentiation to its math operators. 2**3 = 8; -2**2 = (-2)**2*

97. .x .y .z

Jmol 11.9.33 adds .x, .y, and .z as equivalents to .atomX, .atomY, and .atomZ.

96. translateSelected x/y/z

Jmol 11.9.32 adds the capability to translate a model programmatically a given number of pixels, nanometers, or Angstroms in the x, y, or z direction.

- load caffeine.xyz
- translateselected x 10
- translateselected y 10 # note that on the screen a positive y direction is downward.
- translateselected x 2 Angstroms
- translateselected Z 10

95. connect XX% YY% ...

Jmol 11.9.32 adds the option to connect atoms based on ranges of percentage of bonding/ionic radii instead of fixed values.

94. lcaoCartoon CPK

Jmol 11.9.32 adds a new option to lcaoCartoon, CPK, which creates a sphere at the current spacefill radius. The reason this could be useful is that such spheres, though associated with an atom, can be slabbed and capped like an isosurface. This allows for a useful "unit cell only" rendering of spacefill models, for example.

- load nacl.cif {1 1 1}; {_Na}.formalCharge = 1; {_Cl}.formalCharge = -1
- spacefill ionic
- lcaocartoon scale 1.0 CAP unitcell "cpk";spacefill off # we turn the spacefill off -- it was just to provide the reference sizes

93. isosurface SLAB/CAP

Jmol 11.9.32 allows the clean slabbing of an isosurface based on a plane definition such as x=3 or {1 0 0 -3} or the currently defined unitcell or boundbox. A negative sign prior to a plane definition indicates "the opposite-facing plane". The distinction between SLAB and CAP is that SLAB leaves the isosurface open at the boundaries, whereas CAP closes it off.

- load 1crn.pdb
- isosurface sasurface translucent
- isosurface SLAB x=3 sasurface fullylit
- isosurface SLAB - x=3 sasurface fullylit
- isosurface CAP xy sasurface
- boundbox {0 0 0} {10 10 10};isosurface SLAB boundbox sasurface

92. specialized file data atom properties

Jmol 11.9.33 allows readers to create atom-based property data that is unique to their specific reader. For example, the Crystal09 reader can read atomic spin and magnetic moment properties.

- load full_PBE40.out;reset;center {1.2758056 2.209848 -1.3375282E-4}; rotate z -114.4; rotate y 82.07; rotate z 95.48; translate x 0.13; translate y 0.13;
- select *; font labels 35
- select property_spin = 1;label "↑" # up arrow
- select property_spin = -1;label "↓" # down arrow
- color {*} property_spin
- display property_spin <= 0
- display property_spin >= 0

91. 1D and 2D unit cells

Jmol 11.9.32 generalizes unit cells to 1D (polymer) and 2D (slab) periodicity. To date, the only reader that supports this sort of symmetry is the Crystal09 solid state computation reader. When reading these files, unit cell designations {1 2 3} are adjusted to only involve the appropriate dimensionality.

- load mgo_slab.out {3 3 1} # note that the Z direction is Angstroms
- load polymer.out {3 1 1} # note that the Y and Z directions are Angstroms

90. CRYSTAL reader

Jmol 11.9.32 adds a reader for Crystal09 solid state computation output files.

- load mgo_slab.out
- load polymer.out

89. isosurface DSN6/O reader

Jmol 11.9.31 adds a reader for DSN6/O map files as served by the Uppsala Electron Density Server. These maps are alternatives to CCP4/MRC format.

88. frame TITLE enhancements

Jmol 11.9.31 allows setting the frame title similar to the way echos are set, using variables, and to set all frame titles at once. The default title is the model name.

- load cyclohexane_movie.xyz
- frame *;frame title;frame 1
- animation on

87. load "somefilename" 0

Jmol 11.9.30 adds the capability to load specifically only the LAST model in a file containing multiple models.

86. isosurface SIGMA

*Jmol 11.9.30 adds the SIGMA option for automatically setting the isosurface cutoff to multiples of root mean square deviation (RMSD) as found specifically in CCP4/MRC electron density data files. Specifically, "sigma 2.0" delivers an electron density plot that is tighter around the nuclei. If, for example, a cutoff of 0.35 is associated with "sigma=1", then **sigma 2.0** sets the isosurface cutoff at 0.75.*

85. set picking DELETEATOM

Jmol 11.9.28 adds the capability to simply click an atom to delete it.

84. set isKiosk

Jmol 11.9.28. This flag, when set TRUE, forces focus at all times and presumes no underlying applets. This allows multi-touch to activate immediately and not wait for a second click on the applet, using the first to establish focus.

83. _multiTouchServer, _multiTouchClient flags

Jmol 11.9.28 sets _multiTouchServer and _multiTouchClient to indicate whether it has been successful in connecting to a multi-touch device (and is thus working as a server) or to a server (and is thus working as a client).

82. FRANK OFF

Jmol 11.9.28 allows the local signed applet to turn off the frank. Especially useful for KIOSK applications in science exhibitions such as the recent installation of "Touch a Molecule" at the Epcot Theme Park.

81. set mouseDragFactor and mouseWheelFactor

Introduced in Jmol 11.9.24, Jmol 12.0 will change the default sensitivity of the mouse wheel to work less agressively. To use the older Jmol 11.8 sensitivity, use ***set mouseWheelFactor 1.15****; the default for Jmol 12.0 is 1.02. Similarly, Jmol 12.0 will have a less agressive drag sensitivity for the mouse, allowing the mouse to work more appropriately, especially in a touch-screen environment, (where the "mouse" is a finger). The default value of 1.0 for* ***set mousedragFactor*** *allows a 180-degrees rotation when the pointer drags across the full window width.*

- set mouseDragFactor 2.0 # more like 11.8 -- drag the model and observe
- set mouseDragFactor 1.0 # Jmol 12.0 - note how the model tracks the mouse better
- set mouseWheelFactor 1.15 # Jmol 11.8 - zooming super sensitive
- set mouseWheelFactor 1.02 # Jmol 12.0 - zooming not so sensitive

80. implicit SCRIPT command

Jmol 11.9.24 makes any command given of the type ***xxxx.xxx*** *into an implicit* ***SCRIPT xxxx.xxx*** *command. This is just for convenience when working with the console, primarily.*

79. ionicRadius

Jmol 11.9.24 adds the ionicRadius property, and allows setting it.

- load quartz.cif {1 1 1}
- {_O}.ionicRadius=1.2; {_Si}.ionicRadius=0.5; spacefill ionic

78. new LOG command

Jmol 11.9.24 adds a new command specifically for the signed applet and the application. The LOG command works the same as ***print*** *but records the information in a log file. If the printed data starts with the characters NOW, then those are replaced by the date and time. For example:* ***log "NOW " + getProperty("modelInfo")****. The file to log to must first be designated using* ***set logFile "someName"****. This name will be prepended with "JmolLog_" and must not contain any directory path. The file will always be created in the Jar file directory. Note that logging is not ever possible with the web-based version, even with the signed applet, but signed applet or application running locally can log to a file. In addition to explicit use of the LOG command, two settings,* ***set logCommands*** *and* ***set logGestures*** *allow automatic tracking of commands and gestures (swipe, pinch, zoom, spin) to the designated log file.*

77. set waitForMoveTo FALSE

In Jmol versions prior to 11.9.24, all moveTo operations executed to completion prior to continuing a script. With ***set waitForMoveTo FALSE****, one can allow moveTo operations to work asynchronously, meaning they have their own thread, and the script will continue immediately. To stop an asynchronous moveTo operation, use* ***moveTo STOP****. In addition, the mouse action DOUBLE-CLICK-LEFT is bound to "_stopMotion".*

76. set allowMultiTouch

Jmol 11.9.24 allows turning off multi-touch gestures (two-finger spread for zoom, two-finger drag for translation).

75. set pickingStyle DRAG

Jmol 11.9.24 adds the DRAG picking style option. ***set pickingstyle DRAG*** *makes the LEFT button a click-and-drag button when associated also with* ***set PICKING select*** *(molecule, group, chain, etc.) and* ***set dragSelected****.*

74. print {*}.polymer

Jmol 11.9.24. The sequential number of the polymer, starting with 1 for each model in the set.

73. select POLYMER=n and select within(POLYMER, {someAtoms})

Jmol 11.9.24. Same as within GROUP, CHAIN, etc., but for polymers. The number n starts with 1 for each new model.

72. set picking select STRUCTURE/POLYMER

Jmol 11.9.24 adds STRUCTURE (i.e. helix, strand or turn), and POLYMER to picking select option. POLYMER is distinct from CHAIN in that some chains contain more than one connected sets of groups, while a "polymer" is defined as a connected set of groups.

- load 1crn.pdb;selectionHalos on;select none
- set picking select STRUCTURE # now click on an atom atom
- set picking select POLYMER # now click on an atom atom

71. new TIMEOUT command and show TIMEOUTS

*Jmol 11.9.24 adds the option to set a script to execute at a future time, either as an isolated event or repetitively. The command takes the form **TIMEOUT name mSecDelay "script"**, where a name is given for future reference along with a delay in milliseconds and a script to run. If the millisecond delay is given as a negative number, then the event repeats every -(mSecDelay) milliseconds. The minimum practical value for the delay is about 100 ms. If the delay is 0 or the script is the empty string "", then any currently waiting timeout with that name is canceled. In addition, **timeout name OFF** cancels a timeout, and **timeout OFF** cancels all timeouts. All timeouts are also canceled when a new structure is loaded. The **show timeouts** command displays a list of all currently active timeouts.*

- timeout mytimeout 2000 "background white"
- timeout mytimeout -1000 "set echo top left;echo @{now()}"
- timeout mytimeout off
- timeout mytimeout -1000 "rotate x 30"
- show timeouts
- timeout OFF

70. set preserveState FALSE

This option, introduced in Jmol 11.9.23, turns off many memory-consuming features of Jmol that are necessary for preserving the state. It can be used in situations where memory is at a premium and there is no desire to write or save the current Jmol state.

69. write MESH

*Jmol 11.9.23 will generate a relatively compact JVXL XML "vertex-only" mesh surface file from an isosurface. A typical command, after creation of an isosurface, would be **write MESH t.jvxl**. Note that standard JVXL files are considerably smaller, however.*

- load caffeine.xyz
- isosurface molecular
- write MESH

68. simple PDB trajectories

Jmol 11.9.23 allows reading of PDB files that contain no MODEL line and are instead simply concatentated versions of the same atoms as trajectories using the TRAJECTORY keyword. Each model must start with atom number 1 for this to work.

- load TRAJECTORY hb.pdb.gz;cartoons only;select not protein;wireframe 0.3
- animation on

67. set command autocompletion

Jmol 11.9.22 adds a new feature for the console. If in the middle of typing a set command one starts pressing the TAB key, Jmol cycles through all possible SET options starting with whatever characters have been typed so far.

66. new STRUTS shape

In rapid prototyping of protein models, it is sometimes necessary to add short connectors between strands and helixes to provide strength to the plastic model. Jmol 11.9.22 adds a new shape, STRUT, that creates these supports. Created especially for the folks at the Center for Biomolecular Modeling at the Milwaukee School of Engineering, these stick-like objects can be added using the ***CONNECT ... STRUTS*** *or* ***SET PICKING STRUTS*** *commands, and they can be calculated automatically using* ***calculate STRUTS****, which utilizes an algorithm contributed by George Phillips at the University of Wisconsin. STRUTS are not measures and they are not covalent bonds. When calculating struts, three parameters are used (defaults given):* ***set strutSpacing 6*** *sets the minimum spacing between struts,* ***set strutLengthMaximum 7.0*** *sets the maximum length that is allowed for a strut, and* ***set strutsMultiple FALSE*** *when set TRUE allows multiple struts on a given atom. In addition,* ***set strutDefaultRadius 0.3*** *sets the default radius for struts. (Their color by default is translucent white.)*

- load 1crn.pdb;wireframe only; wireframe 0.1
- calculate struts
- connect {atomindex=100} {atomindex=73} STRUT
- set picking STRUTS #now select two atoms

65. isosurface INLINE

Jmol 11.9.21 adds an INLINE option to the isosurface command, complementing the inline option for the PMESH command. Data would generally already be present in a variable. It is advisable to reset the variable after use to improve performance, however note that the state will only be preserved if the value of the variable is left unchanged.

- load ch3cl.mol
- x = load("ch3cl.jvxl")
- isosurface INLINE @x
- reset x

64. set picking CONNECT/DELETEBOND

Jmol 11.9.21 adds two new picking options. CONNECT performs like measuring distances, except bonds are created; DELETEBOND does the opposite -- deleting the bonds between selected atoms (and with ***set BONDPICKING TRUE****, deletes bonds as they are clicked.*

- load caffeine.xyz
- set picking CONNECT # now click on atoms
- set picking DELETEBOND # now click on atoms
- set bondPicking true # now click on bonds

63. isosurface COLOR DENSITY

Jmol 11.9.21 adds a new option for isosurface allowing rendering of the actual grid of numbers (volume rendering) of the data rather than an actual isosurface. With CUTOFF 0.0, this setting delivers the entire set of data points.

- load 3hyd.pdb;reset;center {16.626621 2.7783334 1.7281427}; rotate z 100.36; rotate y 112.6; rotate z -72.02; zoom 231.78; translate x 0.13; translate y 0.13;
- boundbox scale 1.2 {tyr};isosurface boundbox cutoff 1.6 "3hyd_map.ccp4.gz" mesh nofill
- boundbox scale 1.2 {tyr};isosurface boundbox cutoff 0.0 color DENSITY "3hyd_map.ccp4.gz"
- boundbox scale 1.2 {tyr};isosurface boundbox cutoff 1.6 color DENSITY "3hyd_map.ccp4.gz"
- set drawhover # now hover over a data point

62. color schemes BW and WB

Jmol 11.9.21 adds Black/White and White/Black (grey-scale) color schemes

- load caffeine.xyz

- color property atomno
- color "BW"
- color property atomno
- color "WB"
- color property atomno

61. load "filename" AS "localFileName"

Jmol 11.9.20 (signed applet and application) allows reading and immediately saving a file to a local drive.

60. extended XYZ file format

Jmol 11.9.20 adds a ninth column to XYZ files that allows for the atom number (as in PDB files) to be specified. Primarily this is for internal use.

59. set slabByMolecule and slabByAtom

*Jmol 11.9.19 adds the **slabByMolecule** and **slabByAtom** options. The first removes entire molecules when they are partially clipped by a slabbing plane; the second removes whole atoms and bonds only.*
- load Water!Liquid.xodydata;slab 60;slab on;zoomto 1 400
- set slabByMolecule !slabByMolecule
- set slabByMolecule false;set slabByAtom !slabByAtom

58. draw POLYGON

*Jmol 11.9.19 allows the ability to draw polygons based on a set of vertices and a set of faces. This capability allows drawing any number of flat triangular (not quadrilateral) faces with or without edges around each face. The description is somewhat like that for PMESH files and involves (a) giving the number of vertices, (b) listing those vertices, (c) giving the number of faces, and (d) listing the faces with a special syntax. Each face is described as an array indicating the three (0-based) vertex indices followed by a number from 0 to 7 indicating which edges to show a border on when the **mesh** option is given: 0 (no edge), 1(v0-v1 edge), 2(v1-v2 edge), 4(v2-v3 edge), and then combinations of these to give combinations of edges.*
- load caffeine.xyz
- draw POLYGON 3 {0 0 0} {1 1 1} {1 2 1} 1 [0 1 2 6] # 6 here means "edges only between v1-v2 and v2-v3"
- draw POLYGON 3 {0 0 0} {1 1 1} {1 2 1} 1 [0 1 2 6] mesh nofill
- draw POLYGON 4 {0 0 0} {1 1 1} {1 2 1} {0 5 0} 2 [0 1 2 6] [0 3 2 6] mesh nofill

57. calculate/delete HYDROGENS and minimize ADDHYDROGENS

Jmol 11.9.19 adds the ability to add and delete hydrogens, and to add hydrogens and minimize the structure in one operation.
- load thyroxinenoh.mol;rotate x 30;rotate y 30
- calculate HYDROGENS
- delete HYDROGENS
- minimize ADDHYDROGENS

56. isosurface WITHIN x.x {points}

*Jmol 11.9.19 adds a WITHIN option to produce the surface within a given distance of ANY atom in the set, not just the center. (**isosurface WITHIN** was introduced in Jmol 11.1.21 but not documented).*
- load caffeine.xyz;rotate x 30;rotate y 30
- isosurface WITHIN 2.0 {O11} sasurface 0

55. boundbox SCALE x.x option

Jmol 11.9.19 adds a SCALE option for oundbox, allowing scaling with a variety of definition options.

- load caffeine.xyz;boundbox on;rotate x 30;rotate y 30
- boundbox scale 0.5
- boundbox scale 1.1 {_O}
- boundbox scale 1.1 CORNERS {0 0 0} {3 3 3}
- boundbox scale 1.1 {-3 3 3} {3 3 3}
- isosurface sa1 select(O11) sasurface 0;boundbox scale 1.1 $sa1

54. boundbox $isosurface1

Jmol 11.9.19 allows the boundbox to be display around an isosurface.

53. XPLOR electron density reader

Jmol 11.9.19 adds an XPLOR electron density reader.

52. set zshadePower

Jmol 11.9.18 allows for an exponential sort of fog shading. The parameter is log_2(p) where the opacity function f = [(zDepth - z) / (zDepth - zSlab)]^p. This provides a more dramatic effect of depth.
- set background white;load Water!Liquid.xodydata;reset;center {6.5661736 6.6120844 6.639286}; rotate z -125.08; rotate y 55.12; rotate z 120.68; zoom 400.0; translate x 0.13; translate y 0.13;spin on
- set zshade on
- set zshadePower 2
- set zshadePower 1 # default

51. VASP vasprun.xml

Jmol 11.9.18 adds a Vienna Ab Initio Simulation Package reader for vasprun.xml files.
- load vasprun.xml

50. MEASURE function

Jmol 11.9.18 adds the MEASURE() function, allowing programmatic return of measurement information. The syntax is: measure({exp1},{exp}[,{exp}[,{exp}]], min, max, format, units, "CONNECTED"). That is, two to four expressions, minimum and maximum range in Angstroms, a format expression of the sort "%a1 %a2 %5.3VALUE %UNITS", a specification of desired units, including "pm", "nm", "au", or "angstroms", and the keyword "CONNECTED" if the atoms must be connected.
- load caffeine.xyz
- print measure({carbon}, {oxygen}, 1.2, 1.5, "%a1 %a2 %3.0VALUE %UNITS", "pm", "CONNECTED")

49. select configuration=1

Jmol 11.9.18 adds CONFIGURATION to select options. When a model has two alternative locations for some atoms, one can now simply select (or display, hide, color, etc.) the set of atoms that have no alternative location and the set of the nth alternative location using ***select configuration=n****.*
- load 1VIF.cif;reset;center {44.914 36.531498 39.387}; rotate z -65.12; rotate y 83.6; rotate z 97.82; translate x 0.13; translate y 0.13;restrict protein;cartoons only;select not protein; wireframe 0.3 # two inhibitor alignments
- display configuration=1
- display configuration=2

48. hkl(a,b,c)

Jmol 11.9.17 adds the function hkl, which generates the plane associated with a given set of Miller plane indices.
- load quartz.cif {1 1 1}
- print hkl(1,1,1)

47. draw/show SYMOP

Jmol 11.9.17 introduces the ability to describe and illustrate symmetry operations. In addition, you can pick any two atoms or groups or any two positions in space and see the symmetry relations between those groups.

- load maleic.cif 3;moveto /* time, axisAngle */ 1.0 { 539 -803 -254 57.63} /* zoom, translation */ 100.0 0.13 0.13 /* center, rotationRadius */ {0.62577057 3.7338495 4.0119743} 9.027727 /* navigation center, translation, depth */ {0.0 0.0 0.0} 13.809929 7.7656565 0.0;
- show symop
- show symop 2
- draw delete;draw symop 2
- select *;wireframe only;color translucent;draw symop {molecule=1} {molecule=2}; select molecule=1 or molecule=2;Halos 0.1;show symop {molecule=1} {molecule=2}
- select *;wireframe only;color translucent;draw symop {molecule=1} {molecule=3}; select molecule=1 or molecule=3;Halos 0.1;show symop {molecule=1} {molecule=3}
- select *;wireframe only;color translucent;draw symop {molecule=1} {molecule=4}; select molecule=1 or molecule=4;Halos 0.1;show symop {molecule=1} {molecule=4}

46. draw planes intersecting planes with unit cells and bounding boxes

Jmol 11.9.17 expands the "intersection" idea to allow the drawing of planes within boundaries.

- load quartz.cif packed;moveto /* time, axisAngle */ 1.0 { -996 -73 -60 78.27} /* zoom, translation */ 100.0 0.0 0.0 /* center, rotationRadius */ {1.2289999 2.1286902 2.7027001} 5.8060317 /* navigation center, translation, depth */ {0.0 0.0 0.0} -12.054248 29.988888 0.0;
- draw intersection unitcell hkl {0 2 0}
- draw intersection unitcell plane x=3
- draw intersection unitcell plane x=@{{1/2 0 0}.x}

45. draw BOUNDBOX; draw UNITCELL

Jmol 11.9.17 adds the capability to draw a bound box (which can be defined) and the unit cell and to do so with scaling.

- load 1crn.pdb;
- draw boundbox mesh nofill
- select structureID=H2;color group;display selected;boundbox {selected};draw box1 boundbox color white mesh nofill;zoomTo {selected} 0;
- select structureID=H1;color group;display selected;boundbox {selected};draw box2 boundbox color yellow mesh nofill;zoomTo {selected} 0;
- load caffeine.xyz;draw boundbox backlit;draw b2 boundbox translucent 0.7;rotate x 30;spin on # possibly interesting... or perhaps just too weird.
- spin off

44. new Van der Waals default 23%AUTO

Jmol 11.9.17 changes the default Van der Waals radius to "AUTO" to allow non-PDB files and PDB files with H atoms to load with a slightly different look than PDB files with no H atoms. This brings Jmol's default parameter set in line with OpenBabel 2.2.] Along with this change there is a simple syntax for specifying SPACEFILL, HALO, and STAR size.

- load caffeine.xyz
- spacefill JMOL # the old look -- bloated carbons for caffeine. Jmol 11.8 default -- proposed 12.0 default for 1CRN
- spacefill AUTO # the new look -- proposed 12.0 default for all models
- spacefill BABEL # same as AUTO for caffeine -- new for Jmol; proposed 12.0 default for caffeine and 1CD$
- spacefill 20%JMOL # the OLD default ball and stick look
- spacefill 23%AUTO # the NEW default ball and stick look
- load 1crn.pdb # now try the above links -- this time JMOL and AUTO will be the same
- load 1cdr.pdb # this PDB file with H atoms uses the more appropriate BABEL set of parameters automatically

43. "mountain" plots from plane mappings

Jmol 11.9.17 adds the SCALE3D option to isosurface. This generates a 3D plot of the desired scale from a mapped plane. It can be introduced either with the original definition of the isosurface or later.

- load C6H6.smol;mo homo;reset;center {0.0 0.0 0.0}; rotate z -129.9; rotate y 63.91; rotate z 95.81; translate x 0.13; translate y 0.13;
- isosurface plane z=0.5 MO homo
- isosurface scale3d 5 offset {0 0 2}
- isosurface scale3d 2
- isosurface mesh nofill

42. write PMESH

Jmol 11.9.14 adds the PMESH option to the write command, creating XJVXL files.

- zap;pmesh "pmesh.bin"
- write PMESH

41. getproperty SHAPEINFO

Jmol 11.9.14 expands the getproperty command to include information about an isosurface extent

- load caffeine.xyz;isosurface molecular translucent
- getProperty "shapeInfo.isosurface[1].xyzMin"
- getProperty "shapeInfo.isosurface[1].xyzMax"
- isosurface delete;pmesh "pmesh.bin"
- getProperty "shapeInfo.pmesh[1].xyzMin"
- getProperty "shapeInfo.pmesh[1].xyzMax"

40. DGRID reader

Jmol 11.9.14 adds a DGRID reader. These files are generalized representations of output from a variety of quantum mechanical calculation packages, including especially ADF.

- load H2O.adf.dgrid;frame last;MO lumo

39. isosurface and pmesh bounding box information

Jmol 11.9.14 allows access to the bounding box that contains a graph.

- zap;isosurface ID "s1" FILE "func.jvxl"
- print getProperty("shapeInfo.isosurface[1].xyzMin")
- print getProperty("shapeInfo.isosurface[1].xyzMax")
- zap;pmesh "pmesh.bin"
- print getProperty("shapeInfo.pmesh[1].xyzMin")
- print getProperty("shapeInfo.pmesh[1].xyzMax")

38. surface scaling and repositioning

Jmol 11.9.13/14 adds the ANISOTROPY option for all PMESH and ISOSURFACE objects. Just add a point or array after the ANISOTROPY keyword to indicate the scaling in the X, Y, and Z directions. An optional CENTER can also be defined rather than the standard {0 0 0}. The OFFSET option allows an after-the-fact repositioning capability.

- zap;Pmesh "pmesh.bin";axes on;axes molecular
- Pmesh anisotropy {0.1 1 1} "pmesh.bin"
- Pmesh anisotropy {1 2 1} "pmesh.bin"
- Pmesh anisotropy {1 2 1} center {-1 -1 -1} "pmesh.bin"
- Pmesh anisotropy {1 2 1} offset {2 2 2} "pmesh.bin"
- pmesh offset {1 1 1}
- pmesh offset {1 -1 1}

- load C6H6.smol;moveto /* time, axisAngle */ 1.0 { -999 34 -8 65.28} /* zoom, translation */ 21.85 0.13 0.13 /* center, rotationRadius */ {0.0 0.0 0.0} 3.657167 /* navigation center, translation, depth */ {0.0 0.0 0.0} 0.0 0.0 0.0;
- isosurface "func.jvxl" color translucent
- isosurface anisotropy {0.5 0.5 0.5} offset {-5 -5 -5} "func.jvxl" color translucent fullylit
- for (var i = 0; i < 20; i++) {isosurface offset @{point(-5, -5, -i/4.0)};delay 0.05 }

37. set phongExponent

*Jmol 11.9.13 adds the **phongExponent** parameter. This is a standard parameter relating to specular reflection (the bright dot on a ball) and is related to Jmol's **specularExponent** as 2^(specularExponent)*

36. set saveProteinStructureState

Jmol 11.9.13 adds the option to not save helix/sheet/turn data to the state. Generally this information is unnecessary, and in certain situations it can be a large amount of information. The flag is generally set TRUE by default.

35. axes labels

Jmol 11.9.12 allows specification of axis labels. Either three or six labels can be indicated. If only three axes labels are given, the axes will start at 0.

- axes molecular; axes on;
- axes labels "a" "b" "c"
- axes labels "a" "b" "c" "-a" "-b" "-c"

34. scales for axes, boundbox, measures, and unitcells

Jmol 11.9.12 adds optional scales to axes, boundbox, unit cells, and measures. There are three levels of ticks - major, minor, and "subminor." Only the major ticks have labels. Which of these tick levels are displayed and the distance between ticks depends upon the parameter that looks like a point given after the keyword TICKS in the AXES, BOUNDBOX, UNITCELL, or MEASURE command. For a specific axis, include "x" "y" or "z" just after TICKS. The additional optional keyword FORMAT defines the format of the labels for the major ticks. These are based on an array of strings given after the FORMAT keyword. If the array is shorter than the number of ticks, the formats in the array are repeated. The label numbers and tick separations do not have to be angstroms. For the UNITCELL command, simply use fractional coordinates for the ticks to provide a scale in unit cell coordinates. For the other commands, scaling is accomplished using the SCALE keyword followed by a general scaling number or a point indicating how much scaling to apply in each of the directions x, y, and z. For MEASURE you can also add a FIRST keyword followed by a starting value.

- load caffeine.xyz
- boundbox ticks {2.0, 1.0, 0.2} # major, minor, subminor
- measure ticks {1.0 0.2 0} format ["%0.0f"] {-3 0 0} {3 0 0}
- measure ticks {1.0 0.2 0} format ["%0.0f"] first -1.3 {-3 0 0} {3 0 0}
- boundbox off;measures delete;
- axes ticks {1 0 0} format ["%0.0f"]
- axes ticks none
- axes ticks x {2 0 0} format ["%0.0f"] scale 2
- axes ticks y {1 0 0} format ["%0.0f"] scale 1.5
- axes ticks z {3 0 0} format ["%0.0f"] scale 3
- axes ticks none; axes ticks {1 0 0} format ["%0.0f"] scale {1 2 3}
- axes ticks none; axes ticks x @{point(3.14159/2,0,0)} format ["π/2" "π"] # Note that the origin for axes and unit cells is not given a tick or a label, The π character is \u03C0.
- load quartz.cif packed;rotate x -60;axes off;unitcell ticks {1/2 1/10 0}

33. new multi-touch interface

Jmol 11.9.11 introduces a modified Sparsh-UI gesture server interface that allows Jmol to talk to a special multi-touch driver (our first C++ component of Jmol). The interface so far allows multi-touch action only

an HP TouchSmart computer (or any computer with NextWindow technology. The stand-alone Jmol application, Jmol embedded in other Java programs, and the signed applet are supported. One must first start JmolMultiTouchDriver.exe. After this small C++ program is running, all messages from the touch-screen are passed to Jmol and interpreted as "gestures." The signed applet must be started with the "multitouchSparshUI" parameter set to "true". The application must be started with the -Msparshui flag, and embedded Jmol must be passed the command option -multitouch-sparshui. Gestures include a standard two-finger spread/pinch for zoom in/out, a two-finger slide for translation, and a one-finger "flick" to start spinning.

32. zoom in/out; zoomTo in/out

*Jmol 11.9.11 adds **in** and **out** as parameters to the **zoom** and **zoomTo** commands. These parameters are synonymous with "*2" and "/2", respectively.*

- zoom in
- zoom out
- zoom *2 #same as zoom IN
- zoomTo in
- zoomTo out
- zoomTo 3 {1} in
- zoomTo 1 {30} out

31. matrix math

Jmol 11.9.10 allows a broader range of matrix math, including two new functions, m.col(n), and m.row(n).

- q = axisangle({1 0 0},30);m = q%-9;print q;show @m
- print m * [1 0 0]
- print [1 0 0] * m
- show @m
- show @{-m} # transpose
- show @{!m} # inverse
- show @{m.col(1)}
- show @{m.row(2)}
- show @{m.col(3)}

30. draw lineData

*Since Jmol 11.7.1 users have been able to draw arbitrary lines using **draw LINE** followed by a set of points. Starting with Jmol 11.9.10, you can draw line segments based on a set of start/stop pairs. The command was designed to allow for saving the state of the **draw INTERSECTION** command, but may have other practical uses.*

- load caffeine.xyz;
- draw lineData [{0 0 0} {1 1 1}, {0 0 0} {1 1 -1}, {0 0 0} {1 -1 1}]

29. draw INTERSECTION $myIsosurfaceID PLANE|HKL [plane definition]

Jmol 11.9.10 allows drawing of the intersetion of an isosurface with a plane.

- set antialiasdisplay;load caffeine.xyz;isosurface s1 molecular;
- draw intersection $s1 plane x=0 color red
- draw intersection $s1 plane x=1 color green
- draw intersection $s1 plane x=2 color blue
- moveto /* time, axisAngle */ 1.0 { -962 -199 188 105.82} /* zoom, translation */ 100.0 0.0 0.0 /* center, rotationRadius */ {-0.23140144 0.61279976 0.071704745} 5.580881 /* navigation center, translation, depth */ {0.0 0.0 0.0} 1.814305 -1.5505937 0.0;isosurface off;for (var i = -3; i < 3; i+=0.1){draw intersection $s1 plane x = @i;delay 0.1}
- draw diameter 0.02 intersection $s1 plane z = 0 color blue
- for (var i = -3; i < 3; i+=0.1){draw ID @{"x"+i} intersection $s1 plane x = @i color red;delay 0.1}

28. isosurface color MESH [color]

Jmol 11.9.10 adds the option to color the mesh aspect of an isosurface differently from the isosurface itself. Just indicate the color of the mesh prior to defining the object itself.

- load caffeine.xyz;isosurface color mesh red molecular mesh fill

27. new mouse action: LEFT-DOUBLE-CLICK-DRAG

Jmol 11.9.10 adds a new mouse action. Pressing the mouse twice and dragging translates the model.

26. new commands BIND and UNBIND

Jmol 11.9.9 introduces two new commands, BIND and UNBIND. These commands allow customized connections between user mouse actions and Jmol actions. In addition, you can now tie a mouse action to a script of your choice, and Jmol will insert into that script the variable names _X, _Y, _DELTAX, _DELTAY, _TIME, and _MODE. Current Jmol actions that can be bound to mouse actions include: _clickFrank, _depth, _dragDrawObject, _dragDrawPoint, _dragLabel, _dragSelected, _navTranslate, _pickAtom, _pickIsosurface, _pickLabel, _pickMeasure, _pickNavigate, _pickPoint, _popupMenu, _reset, _rotate, _rotateSelected, _rotateZ, _rotateZorZoom, _select, _selectAndNot, _selectNone, _selectOr, _selectToggle, _selectToggleOr, _setMeasure, _slab, _slabAndDepth, _slideZoom, _spinDrawObjectCCW, _spinDrawObjectCW, _swipe, _translate, and _wheelZoom

- unbind "CTRL-ALT-LEFT";bind "CTRL-ALT-LEFT" "_rotate" # _rotate is a special keyword now press CTRL-ALT-LEFT and drag the model -- same as just LEFT alone
- unbind "CTRL-ALT-LEFT" #remove that binding from all Jmol actions
- unbind "_popupMenu"; unbind "_clickFrank" # now try to pull up the popup menu
- rotate x @{random()*90};rotate x @{random()*90};bind "ALT-SHIFT-LEFT" "boundbox on;moveto LEFT;boundbox off" # press ALT-SHIFT-LEFT
- set echo bottom left;bind "ALT-LEFT" "echo _X _Y _MODE" # now press ALT-LEFT and drag the cursor around the screen. Note, this could be a function call, such as myfunc(_X,_Y)"
- unbind # resets the binding to Jmol defaults

25. getProperty mouseInfo

Jmol 11.9.9 adds the MOUSEINFO property to getProperty. This information includes detailed information about how ALL mouse actions correlate with Jmol actions.

- getProperty mouseInfo
- getProperty mouseInfo.bindingName
- getProperty mouseInfo.bindings
- print getProperty("mouseInfo.actionNames")
- print getProperty("mouseInfo.actionInfo")[10]

24. show mouse [option]

*Jmol 11.9.9 introduces a command that lists the current Jmol action/mouse button bindings. The **option** provides a quick way to search for any word in the command description.*

- show mouse
- show mouse pick
- show mouse move
- show mouse select

23. right-edge zoom

In Jmol 11.9.9, a vertical motion in the right 2% of the window interpreted as a zoom motion (as in a touchpad).

22. set allowGestures

Jmol 11.9.9 represents a major upgrade in the way mouse events are handled internally. Events are "bound" to Jmol "actions" in a dynamic way. This allows for future expansion to include customized mouse action. A Java programmer using Jmol can now completely rewrite Jmol's mouse button mapping in just a

few minutes. This upgrade also allows for "gestures" -- dragging motions that are bound to specific actions based on their temporal or spacial pattern. One gesture is included in Jmol 11.9.9: a swipe of the screen that starts the model spinning on an axis perpendicular to the swipe and parallel to the screen. Additional multi-touch gestures can now also be bound to specific actions. For example an iPod-like "pinch" could indicate "zoom out", or a two-finger swipe could mean "advance animation". The gesture capability is turned on using ***set allowGestures TRUE****. It is turned off by default to be compatible with previous versions of Jmol.*

- set allowGestures # now swipe the screen with the mouse (that is, press the LEFT button and move the mouse, but let go of the LEFT button during the motion). Clicking anywhere on the screen stops this motion.

21. label HIDE and DISPLAY

Jmol 11.9.8 adds the capability to temporarily hide or display selected labels. Select the desired atoms, then issue the command ***label HIDE*** *or* ***label DISPLAY****. Unlike label ON and label OFF, these options do not reset the label to the default label. Jmol 11.9.8 also changes the action of* ***set toggleLabel*** *to toggle labels hidden or displayed without resetting them.*

- load caffeine.xyz; label Atom %e-%i
- select carbon;labels HIDE
- select carbon;labels DISPLAY
- select oxygen;set toggleLabel #click me multiple times
- for (var i = 0; i < 10; i++) {set toggleLabel;delay 0.2}
- label "ON" # note that you can use quotes now to force the label even if it is a keyword

20. load single vibration

Jmol 11.9.8 allows loading of a single vibration by number, starting with 1. Simply include the negative of the vibration number after the file name. (A positive number here is a model number.)

- load "C6H6.smol" -1;rotate x -70;vibration on
- load "C6H6.smol" -3;rotate x -70;vibration on

19. JVXL XML format

The JVXL format allows compression factors in the range 10-300 for file delivery of isosurface data. Jmol 11.9.7 is an initial attempt to define an XML standard for JVXL files. To date, the JVXL file format in its original form has been only marginally extensible -- but has been extended by use of XML-like fields. This version creates a simple XML format that allows for a wide variety of future implementation, including additional methods of ASCII-encoded compression, discretely contour-mapped plane and other surface data, pmesh triangle/vertex data storage capability, and generalized contour containers. The default writing format is now "XJVXL" for these extended types, and in this version use of ".xjvxl" in the file name of the file enables writing of all isosurface types as XML files.

18. now()

Jmol 11.9.7 adds the now(i) function to show milliseconds of time since an event. Just assign now() to a variable such as x, and then later check now(x).

- print now()
- x=now(); delay 3;print "the number of milliseconds was " + now(x)

17. isosurface ... map CONTOUR DISCRETE and INCREMENT

Jmol 11.9.7 allows creation of specific contour levels for an isosurface, and also changes the visualization of the solid sections between the contours to be solid colors (FILL attribute of the isosurface command). The DISCRETE keyword allows any values; the INCREMENT keyword allows creation of a set of contours based on the set {from, to, step}.

- load C6H6.smol; isosurface plane {0 0 1 0} map contour 20 mep
- isosurface plane {0 0 1 0} map contour DISCRETE [-0.05, -0.04, -0.03, -0.02, -0.01, -0.0, 0.01, 0.02, 0.03, 0.04 0.05] mep
- isosurface FILL # contour lines black

- isosurface plane {0 0 1 0} map contour INCREMENT {-0.05, 0.05, 0.004} mep
- isosurface FILL NOMESH # contour lines removed

16. zshade ("fog")

*Jmol 11.9.7 perfects the effect of fog -- a blending of distant areas of the model into the background. To turn on this effect, issue **set zshade;**. The standard SLAB and DEPTH values (default 100 and 0, respectively) determine both the clipping plane and the points where the structure appears clearest (slab value) and where it blends into the background (depth value). Note that issuing **slab on** is no longer necessary to enable zShading.*

- load 1crn.pdb;cartoons on;color group;set zshade;spin on
- background white;zoomto 10 *4;delay 10;spin off
- slab 100;set zshade on;for(var i = 0; i < 100; i++){delay 0.01;depth @i}
- slab 100;set zshade on;for(var i = 100; i >= -300; i--){delay 0.01;depth @i};

15. getproperty shapeInfo.isosurface

Jmol 11.9.7 allows access to the colors and contour values used in contour mapping.

- load nacl.cif {1 1 1};isosurface hkl {1 1 1} map contour 6 molecular color "red green blue"
- getproperty shapeInfo.isosurface
- print getproperty("shapeInfo.isosurface[1].contours.colors")
- print getproperty("shapeInfo.isosurface[1].contours.values")

14. isosurface ... COLOR "color1 color2 color3..."

Jmol 11.9.7 adds the capability to specify contour colors directly, without a color scheme. Simply place a set of color names in quotes after the COLOR keyword for the isosurface. If fewer colors are given than contours, the color sequence is repeated as many times as necessary. The FILL keyword fills the spaces between the contour levels with solid colors, as for discrete contours.

- load nacl.cif {1 1 1};isosurface hkl {1 1 1} map contour 6 molecular color "red green blue"

13. isosurface contour DISCRETE and contour INCREMENT

Jmol 11.9.7 adds two new options for isosurface plane mappings that allow for a more GNUplot-like contouring. The first option, CONTOUR DISCRETE, accepts a set of contour level values in the form of an array. The second, CONTOUR INCREMENT, takes three numbers -- first, last, and step -- in the form of a point {from, to, step}. If the isosurface is given the FILL attribute, the lines are drawn in black, and regions between the contours are given solid colors (that is, with no color blending).

- load caffeine.xyz;isosurface iz plane z=0 contour DISCRETE [-2.0, -1.8, -1.6, -1.5, -1.0, -0.5, -0.1] molecular
- load C6H6.smol;wireframe 0.05;spacefill off;isosurface plane z=0 color absolute -0.2 0.2 contour increment {-0.2 0.2 0.01} MO 15
- load C6H6.smol;isosurface molecular contour increment {-0.2 0.2 0.01} MO 15
- load 1crn.pdb;isosurface sasurface 0 map contour increment {4 30 2} property temperature

12. Shape size testing and setting

Jmol 11.9.5 adds the capability to test for sizes of several shapes, including backbone, cartoon, dots, ellipsoid, geosurface, halo, meshRibbon, ribbon, rockets, spacefill, star, strands, and trace. Settable shape sizes include all of the above except dots, ellipsoid, and geosurface.

- load 1crn.pdb; color {34} yellow;cartoons on
- print {34}.cartoon
- if ({34}.cartoon > 0.5)print "yes" else print "no" endif
- {34}.cartoon *=0.75
- load caffeine.xyz;dots on
- print {_O}.dots
- print {_C}.dots

11. Jmol Archive Format (.jmol) -- in review

*Starting with Jmol 11.9.4, you can use **write "myfile.jmol"** to save a single ZIP-format file that contains all files necessary to load the model state. Drag/dropping this file into Jmol or using **load "myfile.jmol"** recreates the state. The file contains (1) a file JmolManifest.txt, which indicates the date of creation, Jmol version used, and the file to open, (2) a JPG image with embedded script, (3) an SPT state file, and (4) all additional files required (PDB files, JVXL files, etc.). If a different file extension is desired, the keyword ZIPALL can be used prior to the file name: **write ZIPALL "myfile.zip"**. In addition, rather than using ZIPALL, if the keyword is "ZIP", then only files local to the file system are saved -- any remote file references starting with http:, https:, or ftp: are left as such, and those files will be accessed remotely when Jmol opens the archive. An example of a Jmol Archive Format file is data/test1.jmol. It can also be tested using the Jmol Crystal Structure Explorer, where it is invoked by a JmolButton (lower right panel). The JmolManifest method was introduced in Jmol 11.4; it can be used with this format to load specific files in specific ways. This format is under review, and comments are appreciated.*

- load test1.jmol
- load test1.jmol MANIFEST "nacl.cif"
- load test1.jmol MANIFEST "nacl.cif" {3 3 3}

10. $SCRIPT_PATH$

Jmol 11.9.4 can insert into a script file the path to that file. The special string $SCRIPT_PATH$ will be replaced. The path will either end with "/" or "|", depending upon whether it is into a subdirectory or a ZIP file. When the replacement is done, "$SCRIPT_PATH$/" will first be replaced by the path, then $SCRIPT_PATH$ alone will be replaced. See, for example, data/test1.txt.

- script test1.txt;print path
- print load(path + "1crn.pdb").lines.find("COMPND")

9. symop(n) * symop(m)

Perhaps one of the most difficult aspects of symmetry is seeing that the product of two symmetry operations is still a symmetry operation. And which one? Jmol 11.9.3 allows multiplication of symmetry operations, because they are simply matrices. Combined with the capabilities of the symop() function to calculate symmetry operations from matrices, you have a very powerful symmetry operation "calculator".

- load quartz.cif packed
- draw symop 3
- draw symop @{!symop(3)}
- draw symop @{symop(3) * symop(3)}
- draw symop 4
- draw symop @{symop(4) * symop(3)}
- print "atom 1 is at \n" + {atomindex=1}.xyz
- print "its related atom is at:\n" + symop(3,{atomindex=1})
- y = symop(3,{atomindex=1})
- print "and hopefully we will get atom 1 back from this:\n" + symop(-3, y)

8. symop(n) and matrix math

Jmol 11.9.3 adds matrix mathematics and, in particular, represenation of symmetry operations as matrices. The syntax is the same as for JavaScript: [[a, b, c],[d, e, f],[g, h, i]] (3x3 matrix) or [[a, b, c, x],[d, e, f, y],[g, h, i, z], [j, k, l, w]] for a 4x4 matrix.

- load nacl.cif packed;x = symop(33)
- print x
- print x.lines
- print (!x).lines
- print x[1] # row 1
- print x[-1] # column 1
- print x[13] # row 1, col 3
- print x[1][3] # same thing
- x[13]= 999; print x.lines
- x[1]= [1,2,3,4]; print x.lines

- x[-1]= [1,2,3,4]; print x.lines
- print symop(192,"description")
- print symop(192).lines
- print symop(-192).lines
- print (!symop(192)).lines # same thing
- print symop(symop(192)*symop(192)).lines
- print symop(symop(192)*symop(192)*symop(192)).lines
- print symop(symop(192),"description")
- print symop(symop(192)*symop(192),"description")
- print symop(symop(192)*symop(192)*symop(192),"description")

7. print symop(n,"...")

Jmol 11.9.3 allows extracting of symmetry operation information directly from the symop() function. The information is in the form of an array, as described below in context.

- load nacl.cif packed;set echo top right
- n = 3;draw symop @n;echo @{"" + n + ": " + symop(n,"description")}
- n = 6;draw symop @n;echo @{"" + n + ": " + symop(n,"description")}
- n = 34;draw symop @n;echo @{"" + n + ": " + symop(n,"description")}
- n = 106;draw symop @n;echo @{"" + n + ": " + symop(n,"description")}
- n = 192;draw symop @n;echo @{"" + n + ": " + symop(n,"description")}
- print symop(n) # by itself, its matrix representation
- print symop(n, "array") # an array
- print symop(n,"xyz") # xyz description (from matrix)
- print symop(n,"array")[1] # xyz description (original)
- print symop(n,"description") # text description
- print symop(n,"array")[2] # translation vector (fractional)
- print symop(n,"translation") # translation vector (Cartesian)
- print symop(n,"center") # inversion point (Cartesian)
- print symop(n,"axispoint") # axis point (Cartesian)
- print symop(n,"axis") # axis vector (Cartesian, normalized)
- print symop(n,"angle") # angle of rotation

6. isosurface FULLPLANE

Jmol 11.9.3 adds an option to ISOSURFACE to allow data mapped onto a plane to extend throughout the whole plane, not just in localized spots.

- load C6H6.smol;moveto /* time, axisAngle */ 1.0 { 84 886 -456 176.91} /* zoom, translation */ 100.0 0.0 0.0 /* center, rotationRadius */ {0.0 0.0 0.0} 3.657167 /* navigation center, translation, depth */ {0.0 0.0 0.0} 0.0 0.0 0.0;
- isosurface plane {0 0 1 -1} map mo homo
- isosurface plane {0 0 1 -1} map FULLPLANE mo homo

5. script/write state LOCALPATH/REMOTEPATH

*Jmol 11.9.2 adds two options to the SCRIPT and WRITE STATE commands. When reading or writing a script, the LOCALPATH keyword instructs Jmol to strip all paths beginning with "file:/" down to the indicated path. So, for example, script LOCALPATH "" "myfile.spt" indicates that all local files referenced in the state script should be read from the current default directory. Similarly, REMOTEPATH strips paths from file refrences starting with "http:", "https:", and "ftp:". LOCALPATH and REMOTEPATH are desigend specifically to be used with scripts created by Jmol in the process of defining its state, but the option can be used with scripts you write yourself. The mechanism is simply looking for instances of /*file*/"some_file_name". If this construction is found in any script read, the replacement will be made.*

- script test1.spt #this won't work, because the script saved was for a state that had a file read from my laptop.
- script LOCALPATH "" "test1.spt" # this works, because the "file:" prefix and all subdirectory path information have been removed from the LOAD command. Note that the default directory for this page is "data", so the file read is 1crn.pdb in the data subdirectory.

4. {xxx}.volume("type")

Jmol 11.9.2 introduces a calculation for molecular volume. This calculation depends upon what set of Van Der Waals radii one uses. "type" can be one of "Jmol", "Babel", "Rasmol" or "User". If not provided, "type" defaults to the default type specified by the command ***set defaultVanDerWaal****. The calculation is a simple one that just adds up all the Van Der Waals volumes and subtracts off the overlap of spheres associated with bonds as described by A. Bondi (****J. Phys. Chem. 68, 1964, 441-451.****). The built-in Van der Waals set "Babel" most closely represents this setting.*

- load C6H6.smol;
- print "" + {1.1}.volume() # default is too large
- print "" + {1.1}.volume("Jmol") # same as default
- print "" + {1.1}.volume("Babel") # this is better
- print "" + {1.1}.volume("Rasmol"); # use unknown
- {_C}.vdw = 1.65;print {1.1}.volume(); # custom setting
- set defaultVDW Babel;show VDW;{*}.vdw = 0;label %3.2[vdw] # after setting the default you can do more with volume
- label %3.2[volume]
- set defaultVDW RasMol;show VDW;{*}.vdw = 0;label %3.2[vdw]
- label %3.2[volume]
- set defaultVDW Jmol;show VDW;{*}.vdw = 0;label %3.2[vdw]
- label %3.2[volume]

3. script demo

- script c6h12.txt

The script c6h12.txt illustrates several advanced features of Jmol. In this script, a set of 35 cyclohexane conformations are fitted with quaternion frames on C1 and C4 carbons. A calculation is made of the similarity of each pair of quaternion frames within each model, allowing the rings to be colored based on how similar they are to a cyclohexane chair conformation. A bar graph near the bottom of the window is used to allow the user to quickly scan through conformations. The JavaScript callback for this action is created by the script itself, defining a JavaScript function on the page using the ***javascript*** *command within Jmol.*

The measure is calculated as

grp.property_st = 1.0 - acos(abs(q1.dot(q2)))/90

This measure is directly related to the "distance" between the quaternions on the four-dimensional hypersphere and is thus a direct measure of how well the two frames are aligned. If the frames are perfectly aligned, then this measure is 1.0; if they are completely oppositely aligned -- differing by a 180 degree rotation along any axis -- then this measure will be 0. The reason this works is that the dot product of two unit quaternions is the cosine of half their angular distance on the four-dimensional hypersphere: **q1.dot(q2) = cos(theta/2)***, similar to how for two 3-dimensional unit vectors v1 and v2,* v1.dot(v2) = cos(theta)*, where theta is the their angular distance on the unit sphere. We don't care about the sign of this value, because angles 0 and 180 degrees (cosines 1 and -1) for a quaternion relate to rotation angles 0 and 360 degrees, and those are the same for our purposes here. So taking the arccosine of the absolute value of the quaternion dot product returns a quaternion angle between 0 and 90, with 0 being "perfectly aligned" and 90 being "oppositely aligned." Dividing by 90 and subtracting from 1 gives a measure that is 1.0 for perfect alignment and 0.0 for opposite alignment.*

2. clickCallback

Starting with Jmol 11.9.1, you can detect clicking and dragging within the applet window, and you can cancel that operation if desired. The callback function takes the form:

function myClickCallback(app,x,y,modifiers,clickCount,Ret){}

app *is the applet identifier, usually "jmolApplet0".* ***x*** *and* ***y*** *are screen coordinates with [0,0] in the bottom left corner.* ***modifiers*** *is a set of bits indicating what keys and mouse buttons were pressed: 1(shift), 2(ctrl), 4(right), 8(alt), 16(left). If none of these bits are set, this is a "mouse-up" event.* ***clickCount*** *is the number*

of clicks. 0 here indicates the mouse was pressed (as in the initiation of a drag operation); negative numbers indicate dragging; positive numbers are standard clicks of the mouse. ***Ret*** *is an array with just one element that allows returning a new modifier. Setting Ret[0]=0 cancels Jmol processing of the event.*

- load caffeine.xyz;draw [0 0%] [0 10%] [100 10%] [100 0%] translucent white
- set clickcallback "myClickCallback" # now watch the browser title line as you click on the screen
- set clickcallback "myClickCallback2" # now click or drag within the translucent strip

1. acos(x) and q1.dot(q2)

Jmol 11.9.1 adds the acos() function, returning a value in degrees, and quaternion dot product.

- print acos(0)
- print acos(0.5)
- print acos(1)
- print acos(-1)
- print " " + quaternion(1,2,3,4)
- print quaternion(1,2,3,4).dot(quaternion(1,2,3,4))%2 # q.dot(q) = 1
- print quaternion(1,2,3,4).dot(-quaternion(1,2,3,4))%2 #note that q and -q represent the SAME rotation
- print quaternion(1,0,1,0) # 90 degrees around Y
- print quaternion(1,0,0,1) # 90 degrees around Z
- print quaternion(1,0,1,0).dot(quaternion(1,0,0,1))%2
- print (quaternion(1,0,1,0) / quaternion(1,0,0,1))%0 %2 # q0 term of q2 / q1 is q1.dot(q2)