PACUPP Methods

(**P**ockets **A**nd **C**avities **U**sing **P**seudoatoms in **P**roteins)

by Eric Martz

The term **cavities** will be used to include pockets, tunnels, and channels. **Pockets** have a single mouth to the surface of the macromolecule, and **channels** have two or more mouths. A **tunnel** connects two or more locations, and may or may not have mouths. A cavity or tunnel that has no mouths will be described as **buried**.

Please report concerns, bugs, or suggestions for improvement to m0lviz (at) yahoo.com.

During a PACUPP job, progress is described in the Jmol Script Console. A more detailed progress report can be obtained by setting Verbose to true in support-scripts/defaults.spt. The verbose report lists every script file, and major functions, as they are executed. Other actions that occur between execution of script files and functions will be found in support-scripts/0-master.spt. That script also has a summary list of its actions near the top.

What follows is a step-by-step description of the methods in the order of execution.

[Continued on Next Page]

CONTENTS

Startup Checks and Preparation

PACUPP requires that a macromolecular model be loaded before PACUPP is invoked. Before PACUPP starts processing the model, the following checks are made:

1. If Jmol.jar was not **started in the PACUPP folder**, PACUPP asks that you quit Jmol and start it from that location (which gives it access to all the PACUPP Jmol scripts).
2. **Atoms must be present** in Jmol.

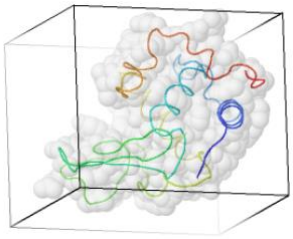
3. Sidechains must be present. If there are >= 10 alpha carbons and zero beta carbons, PACUPP exits explaining that the model appears to contain **only alpha carbons**. (Example: 1a1d.)
4. The loaded model must **not be in mmCIF** format. (About 1% of entries in the Protein Data Bank, the largest models, are available intact only in mmCIF format.)
5. The loaded model must contain at least one ATOM record in **PDB format**.
6. If the **Jmol Script Console** is not already open, it is opened.
7. PACUPP attempts to determine a **PDB accession code** for the loaded model. See *Design Specifications* in 1-How-To-Use-PACUPP for details.
8. **If pseudoatoms** (_ho.ho1 in Jmol language) **are already present**, this signals that a PACUPP job has already been run on the loaded model. In this case, the PACUPP-specific components of the loaded model are removed, restoring the model to its original state (newjob.spt).
    a. If the original model contains **alternate locations**, they will have been deleted by PACUPP. The loaded model is deleted (Jmol zap command) and the original model loaded (Jmol load var headerpdb).
    b. The block of REMARK lines inserted by PACUPP into the header are removed from headerpdb.
    c. The pseudoatoms added to the end of the PDB file are removed from headerpdb.
    d. The model in Jmol's memory is deleted (Jmol zap command), and the original model is loaded (Jmol load var headerpdb).
    e. If the originally loaded PDB file had **multiple models** (typical for an NMR experiment), PACUPP has deleted all but the first. The user is required to quit Jmol, start a new Jmol session, and load the desired PDB entry. (Possibly this is unnecessary.)
9. If the user loaded a single model from a multiple-model entry with a Jmol command such as load =2bbn 7, this loads only model 7 into Jmol memory, but keeps all models in Jmol's record of the file contents. This is not a workable condition for PACUPP, so the user is required to quit Jmol, start a new Jmol session, and load the desired model.
10. If **multiple models** are loaded, all models after model 1 are deleted from the PDB file contents, Jmol is cleared with zap, and the single model is loaded from the variable from which all other models are deleted. Importantly, the PDB header still contains information about all models.
11. If the loaded model contains **alternate locations**, they are deleted. Pseudoatoms will be created based on conformation 1 alone. However, the output PDB file will preserve the alternate locations.
12. If the loaded model specifies a biological assembly differing from the asymmetric unit, the user is encouraged to stop and load the biological assembly. Instructions are provided for easily obtaining the biological assembly in the most practical/useful format (which is not the assembly offered by the Protein Data Bank).

Specification of Job Settings

Dialogs pop up that enable the user to select either the default settings (cavity detail: fine, cavity size: small) or to customize the settings to the desired degree. In addition to cavity detail and size, the user may specify that the pseudoatom 3D grid be "offset", atoms to be inside cavities (such as a peptide ligand), or atoms to be outside cavities (possibly ligands such as heme). Demonstrations of these settings dialogs can be seen in the latter portion of the demonstration video.

Boundbox Corners

After you have specified settings for the job, PACUPP begins by determining the positions of the boundbox corners. The boundbox is the smallest "box" (rectangular parallelopiped) with sides parallel to the X, Y, and Z axes that contains the centers of all atoms in the model. At right is the boundbox of lysozyme (1LZR).



Surface Spheres: Empty & Tangent Spheres

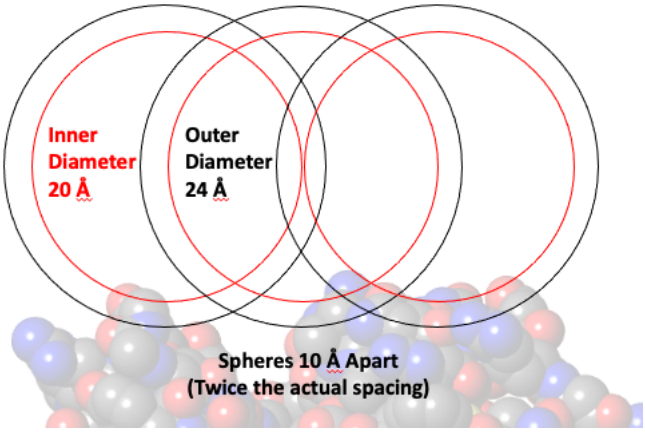
Next, PACUPP creates "surface spheres" that are used to define a smoothed macromolecular surface. Pseudoatoms will later be created to be inside the smoothed surface, and not clashing with atoms of the macromolecule, thus representing pockets and cavities.

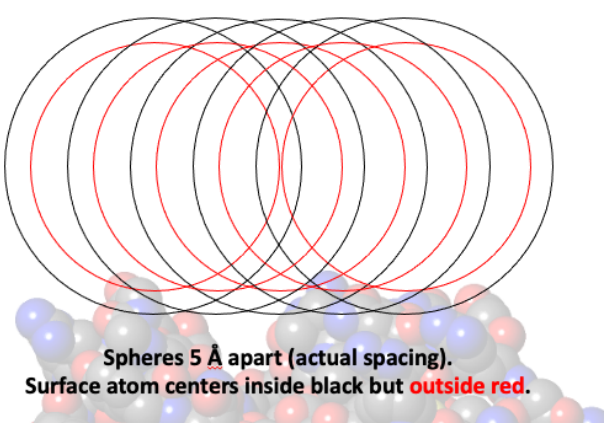
Surface spheres have inner and outer diameters, which are 20 Å and 24Å by default. Two sets are distinguished: Empty Spheres and Tangent Spheres.

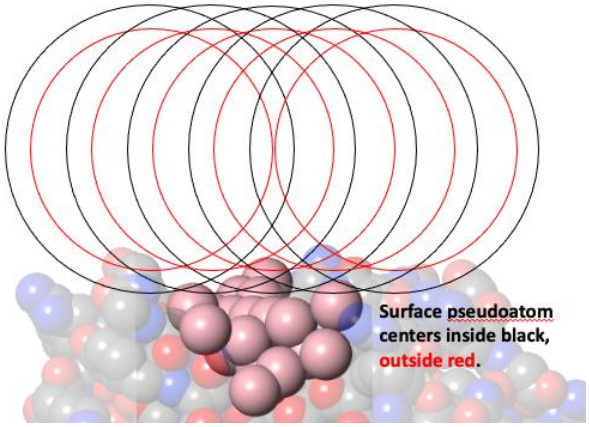
- **Empty spheres** are those containing no macromolecule atoms inside their outer diameters. There are **1,368 empty spheres** for lysozyme (default settings).
- **Tangent spheres** are those containing macromolecule atoms in their outer, but not in their inner diameters. There are **171 tangent spheres** for lysozyme (default settings).
- Surface spheres containing macromolecule atoms in their inner diameter are not used.

[Continued on Next Page]

Potential sphere positions are placed in a 3D grid filling the boundbox every ¼ of the inner sphere diameter. So, for the default inner diameter of 20 Å, starting at the boundbox corner with the lowest X, Y, and Z values, every 5 Å in the X, Y, and Z directions is evaluated as a potential empty or tangent sphere position.



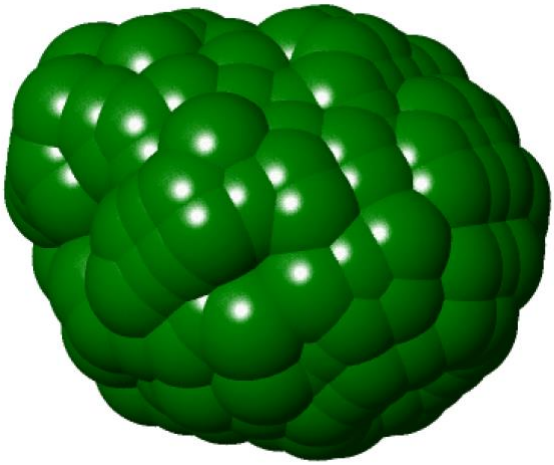




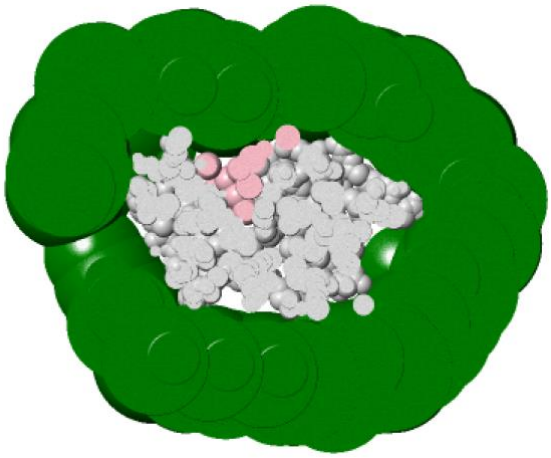
The images above show tangent spheres along only the X axis. But in actuality, the tangent spheres are placed along the Y and Z axes as well, in a 3D grid.

After the tangent sphere positions have been established, pseudoatoms are created wherever they don't clash with macromolecule atoms, with the outer surface pseudoatoms limited to those with centers outside the inner diameters of the tangent spheres. In the section "Cavity Size" in "How To Use PACUPP" is explained how tangent spheres need to be larger than the default diameters to prevent them from fitting inside large cavities, to prevent the pseudoatom fillings of large cavities from being hollow.

[Continued on Next Page]



Tangent spheres surrounding Lysozyme (1LZR).



Slice through the middle (thickness 4%). Protein **light gray**. Pseudoatoms **pink**.

(The command 'cavhelp3' lists the commands 'showTangents' and 'hideTangents', used above to display the tangent spheres in green.)

Pseudoatom Positions To Fill the Boundbox

After the Empty and Tangent Spheres have been established, the **boundbox is filled with a 3D grid of pseudoatom *positions*** ("points"). Pseudoatoms are not created yet – only the *positions* are established so they can be evaluated. With the default setting, cavity detail "fine", pseudoatoms positions are spaced 3.0 Å apart in the X, Y, and Z directions. However, pseudoatom positions are **NOT established where a pseudoatom would clash with macromolecule atoms**. The clash distance is 1.5 Å (see constants.spt) plus half of the pseudoatom effective diameter, where the pseudoatom effective diameter is equal to the center-to-center spacing of pseudoatoms. So for the default pseudoatom spacing/diameter of 3.0 Å, the clash distance is (3/2) + 1.5 = 3.0 Å.

For lysozyme 1LZR with default settings, the capacity of the boundbox (if there were no macromolecule) is ~3,000 pseudoatoms. However only **2,218 pseudoatom positions** do not clash with the macromolecule.

For a fairly large macromolecule (7AHL, in the largest 6% of entries in the Protein Data Bank) this process takes **52 seconds**. It is the slowest part of the entire PACUPP process. It could likely be made more efficient, but since the overall performance seems satisfactory for the vast majority of entries in the Protein Data Bank (see timings in the table in the section *Processing Time* in 1-How-To-Use-PACUPP), effort was not put into this for version 1.0 of PACUPP.

Eliminating Pseudoatom Positions Inside Empty Spheres

Pseudoatom positions inside spheres that contain no macromolecule atoms within their outside diameters, the "**Empty Spheres**", are of no further interest. They are outside the smoothed surface defined by the "Tangent Spheres". (If any Empty Spheres are inside the macromolecule, it means that the pseudoatom filling of some cavities will be **hollow,** and the diameters of the spheres must be increased, and a new job executed.)

So the first task is to delete pseudoatom positions that are inside Empty Spheres. During initial development, this was the slowest part of the entire PACUPP process. Considerable effort was invested to make it more efficient.

Looping for each pseudoatom position, to determine whether it was within an Empty Sphere, took far too long. Next I tried looping through each Empty Sphere (tanOutsidePoints[]) to find and delete the pseudoatom positions (PAPoints[]) outside the macromolecule. This still took too long. I surmised that the problem was the large number of pseudoatom positions that needed to be interrogated for each Empty Sphere. To speed this up, it seemed desirable to **reduce the number of remaining pseudoatom positions as quickly as possible**.

Looping through *consecutive* Empty Spheres is inefficient because adjacent **Empty Spheres have a large overlap**, so one is gathering the same unwanted pseudoatom positions many times. So I re-designed the loop to *jump ahead* to non-overlapping Empty Spheres, and to delete the exterior pseudoatom positions before entering the next cycle. Empirical tests showed that for large macromolecules, the **shortest execution times were obtained by starting with a jump of 64 Empty Sphere indices.** Thus, in the first cycle, the pseudoatoms in every 64th Empty Sphere were deleted. In the 2nd cycle, the PA in every 32nd Empty Sphere were deleted (skipping over the already processed every-64th-Empty-Spheres). In the 3rd cycle, every 16th, and so forth for 7 cycles. For a fairly large macromolecule, 7AHL (in the largest 6% of entries in the Protein Data bank), below is what the report from this scheme looks like.

- A "block" is a cycle.
- The 7 blocks use jumps of 64, 32, 16, 8, 4, 2, and 1. A "jump" is a loop index step size.

For 7AHL (cavity detail: fine, cavity size: large) the process **starts with 2,302 Empty Spheres, and 38,215 pseudoatom positions** that don't clash with macromolecule atoms. You can see that the number of PA positions remaining to be interrogated in each cycle ("block") decreases rapidly, making the next block more efficient.

```
Eliminating pseudoatom (PA) positions that are outside the smoothed surface.
  Block 1 of 7 completed in 1.4 sec. Eliminated: 9% this block, 9% cumulative.
  Block 2 of 7 completed in 1.3 sec. Eliminated: 16% this block, 25% cumulative.
  Block 3 of 7 completed in 1.6 sec. Eliminated: 17% this block, 42% cumulative.
  Block 4 of 7 completed in 2.0 sec. Eliminated: 15% this block, 57% cumulative.
  Block 5 of 7 completed in 2.4 sec. Eliminated: 13% this block, 70% cumulative.
  Block 6 of 7 completed in 2.8 sec. Eliminated: 7% this block, 77% cumulative.
  Block 7 of 7 completed in 4.0 sec. Eliminated: 5% this block, 82% cumulative.
```

When completed, only 7,000 (18%) of the 38,215 pseudoatom positions remain. The rest are outside the macromolecule and have now been deleted. This process took **15 seconds** for 7AHL (cavity detail: fine, cavity size: large).

Eliminating Pseudoatom Positions Inside the Inner Diameters of Tangent Spheres

We also need to delete pseudoatom (PA) positions inside the inner diameters of Tangent Spheres. In our example of 7AHL, there are only 149 Tangent Spheres and 7,000 remaining PA positions. A simple loop identifies and deletes the PA positions that are inside the inner diameters of Tangent Spheres in **1 second**. 5,413 PA positions remain that are inside the smoothed surface of the macromolecule defined by the Tangent Spheres. These define the pockets and cavities.

Locating Pseudoatom Positions at the Smoothed Macromolecular Surface

Next, the subset of the 5,413 PA positions that are at the smoothed macromolecule surface are identified. These are the PA positions **inside the outer diameters of the Tangent Spheres**. In our example of 7AHL, 987 (18%) of the 5,413 PA positions are on the surface. This takes < 1 second for 7AHL.

Creating Pseudoatoms

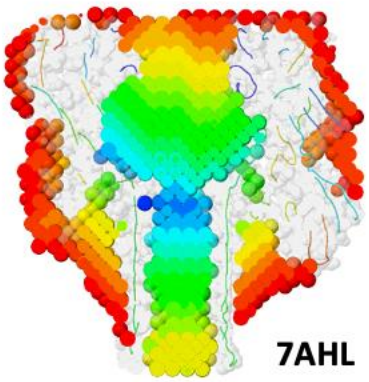
Next, a holmium pseudoatom is created (Jmol 'data' command) at each of the 5, 413 PA *positions* (Jmol points) that define pockets and cavities. Creating the PA takes 1 second for 7AHL.

The subset of PA that match the **surface PA** positions is identified, taking <1 second.

[Continued on next page.]

Defining Layers

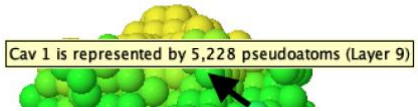
Pseudoatoms (PA) are now assigned to layers. Layer 1 contains the outermost PA, those on the smoothed surface of the macromolecule. Layer 2 contains the deeper PA that contact PA in Layer 1, and so forth. In the case of 7AHL, there are 17 layers. PA that do not touch any PA leading to the surface are "buried".



Cavity detail: Fine
Cavity size: **Large**

Note that PA in Layer 17 are not necessarily the PA farthest from the surface. They are the PA that require traversing the largest number of intervening contiguous PA layers. PACUPP's "Depth" color scheme for PA colors **Layer 1 red**, and the **deepest layer blue**, with a spectral sequence of intermediate colors. Notice that the **blue PA** are **no farther from the surface** than some of the **green PA**, but starting with the **red PA**, one must traverse the largest number of contiguous layers to reach the **blue PA**.

When you touch a PA, its layer is reported.



For those interested in using Jmol command language (as distinct from PACUPP commands), PA in a layer can be selected thus: "select L3" for Layer 3 (no space after the "L"). Buried PA: "select unlayered" (not "buried", which is a Jmol term that selects amino acids usually buried in soluble proteins).

Grouping Pseudoatoms into Discrete "Cavities"

Next, pseudoatoms (PA) are grouped into discrete clusters or clumps. A cluster consists of PA in contact with each other. Each cluster is deemed a "cavity" and each cavity is assigned an ID number.

A list is displayed of the number of PA in each cavity, along with an estimate of the volume of the cavity. A separate list sorts cavities by PA/cavity. For the example 7AHL (cavity detail: fine, cavity size: large), there are 44 discrete cavities. 35 of these have only one or two PA each. Seven cavities have 3-8 PA each. One cavity has 104 PA, and the largest one has 5,228 PA.

Volume Estimation

The volume of a cavity is estimated as the number of PA times the volume of a single PA, taken as the volume of a cube that has a dimension equal to the PA separation/diameter value. For the default settings (cavity detail: fine), PA separation/diameter is 3.0 Å. So the volume of a cube with that dimension on a side is 3 x 3 x 3 = 27 $Å^3$. The estimated volume of a cavity with

100 PA is therefore 2,700 Å^3. See also the section "**Uncertainty in Cavity Volumes**" in 1-How-To-Use-PACUPP.

Coding Cavity ID's and Layers into PDB Atom Properties

The **cavity ID** number for the cavity to which each PA belongs is divided by 100 and assigned as its PDB **temperature**. This is a temporary convenience for use later in PACUPP processing. The **layer number** of each PA is divided by 100 and assigned to its **occupancy**. This is permanent.

Generation of Pseudoatom PDB ATOM Records

Data lines for all atoms in Jmol's memory (macromolecule atoms plus pseudoatoms) are written into a Jmol variable in PDB format. The macromolecule atom lines will later be discarded. The point of writing out all atoms is so that the atom serial numbers for pseudoatoms begin above the highest number for a macromolecule atom. When Jmol writes PDB atom lines, it writes CONECT records for all bonds at the end. These are deleted for convenience. Next, **all pseudoatom lines are extracted:** the lines from the first HO1 line to the end are extracted.

Next, a **chain name** is assigned to the PA. This is not very important, but possibly might be useful. Unless it is already in use, the PA are assigned to chain "P" (for Pseudoatom). If "P" is in use, other names are tried. If the chain names have cases (Jmol chainCaseSensitive true), names considered are P, p, Z, z, and 0 (numeral zero). If chain names do not have cases, names considered are P, Z, 0, 9. If none of those chain names are available, the chain name for PA is left blank.

A number of features of the format used by Jmol to write PA PDB lines are changed:

- ATOM is changed to HETATM.
- Ho## is changed to HO1. (## is the first 2 digits of the Ho atom's atom serial number.)
- Residue name UNK given by Jmol is changed to HO.
- The blank chain name given by Jmol is changed to one of those listed above, if available.
- Sequence number zero given by Jmol is changed to the ID number of the cavity to which the PA belongs (taken from its temporary storage as temperature).
- The temperature values that were used temporarily to store cavity ID numbers are changed to 0.00.

PA are named HO1 by PACUPP for the PDB format rather than simply HO. This distinguishes them from the rare PDB entry that contains real holmium. In November, 2020, there are 11 such entries. All use the atom name HO except 1w6z which uses HO3 for trivalent Ho.

Here are two PDB lines for PA, the first as Jmol writes it, and the 2nd after changes by PACUPP:

```
                                                                  Layer  CavID
Jmol:    ATOM   1296 Ho12 UNK     0      28.480  14.970  25.890  0.02  0.07          HO
PACUPP:  HETATM 1296 HO1   HO P   7      28.480  14.970  25.890  0.02  0.00          HO
                              CavID                               Layer
```

Report

PACUPP constructs a report in a variable that will later be written into a text file in the folder *output-files/reports*.

Job Description Summarized in Jmol

PACUPP writes a brief job description into the upper left corner of Jmol's molecule display window ("echo" in Jmol language). This is retained when PACPP's output PDB or PNGJ output file is later dropped into Jmol.

The 4 numbers are: pseudoatom separation/diameter, tangent sphere separation, tangent sphere inner diameter, tangent sphere outer diameter, all in Å.

Jmol Commands Inserted into the PDB Header

Early in the PACUPP job, the original contents of the PDB file being processed were saved in a variable (headerpdb).

The job settings and other information that will be needed when an output PDB or PNGJ file are dropped into Jmol are written into the variable containing the original PDB file as Jmol commands **inserted as REMARK lines into the PDB header**. You can see these by viewing the contents of any PACUPP output PDB file in a text editor or word processor.

Dropping PACUPP Output PDB/PNGJ Files Into Jmol

When you drop a PACUPP output PDB file into Jmol, the Jmol commands in the header are executed automatically by Jmol. In the case of PNGJ files, execution is not automatic: the commands must be extracted from the header into a text variable, and then executed.

Dropping a PACUPP output PDB file into Jmol automatically displays the molecule as it is shown at the completion of a PACUPP job, and automatically runs Jmol scripts that define all the PACUPP commands. Also, a summary report of the job is automatically displayed in the Jmol Script Console.

When a PACUPP PNGJ file is dropped into Jmol, a message appears above the molecule instructing the user to drag the file menu/menu.spt and drop it into Jmol. This manual action has the same effects as the automatic execution of the Jmol commands in the PDB file header.

The above behaviors require that the Jmol into which the PACUPP output PDB or PNGJ file is dropped have been started from the **1-Jmol.jar** in the *Fill_Cavities_PACUPP* folder. If such a file is dropped into Jmol started from another location, a message appears asking the user to quit Jmol and restart it from the *Fill_Cavities_PACUPP* folder.

Pseudoatoms Added to PDB File

The END record causes problems if it precedes the pseudoatom records. The MASTER record does not appear to cause any problems, but both the END and MASTER records are deleted from the variable containing the original PDB entry.

Then, the pseudoatom ATOM records in PDB format are added to the end of the string variable containing the PDB file to be written, along with REMARK lines identifying PACUPP as the source of these lines.

PACUPP PDB File is Loaded, Replacing the Original

1. A Jmol zap command is executed. This deletes all atoms and atom properties from Jmol's memory. However, variables and user-defined functions are retained.
2. The contents of the variable *headerpdb* are loaded (Jmol: load var headerpdb). Jmol now contains the original model, with its header intact, plus the added pseudoatoms.
3. Some operations that were previously executed are now repeated to restore their results:
    a. Grouping pseudoatoms into discrete clusters or "cavities".
    b. The molecular display for PACUPP output is restored (render.spt).
    c. The job summary is re-displayed at the upper left of Jmol's molecular display window.

Output Files Written

Three output files are written into the *output-files* folder: a PDB file, a PNGJ file, and a report.txt file.

Help for PACUPP Commands

A compact overview is displayed of PACUPP commands for displaying the cavities in various ways, along with commands for displaying more detailed help or a list of alphabetized commands, or some more advanced help.

Elapsed Time

The total time taken for PACUPP execution is displayed.

This completes the PACUPP job. Exploration of the cavities is now up to the user, using PACUPP commands.

Please report successes, concerns, bugs, or suggestions for improvement to m0lviz (at) yahoo dot com.