

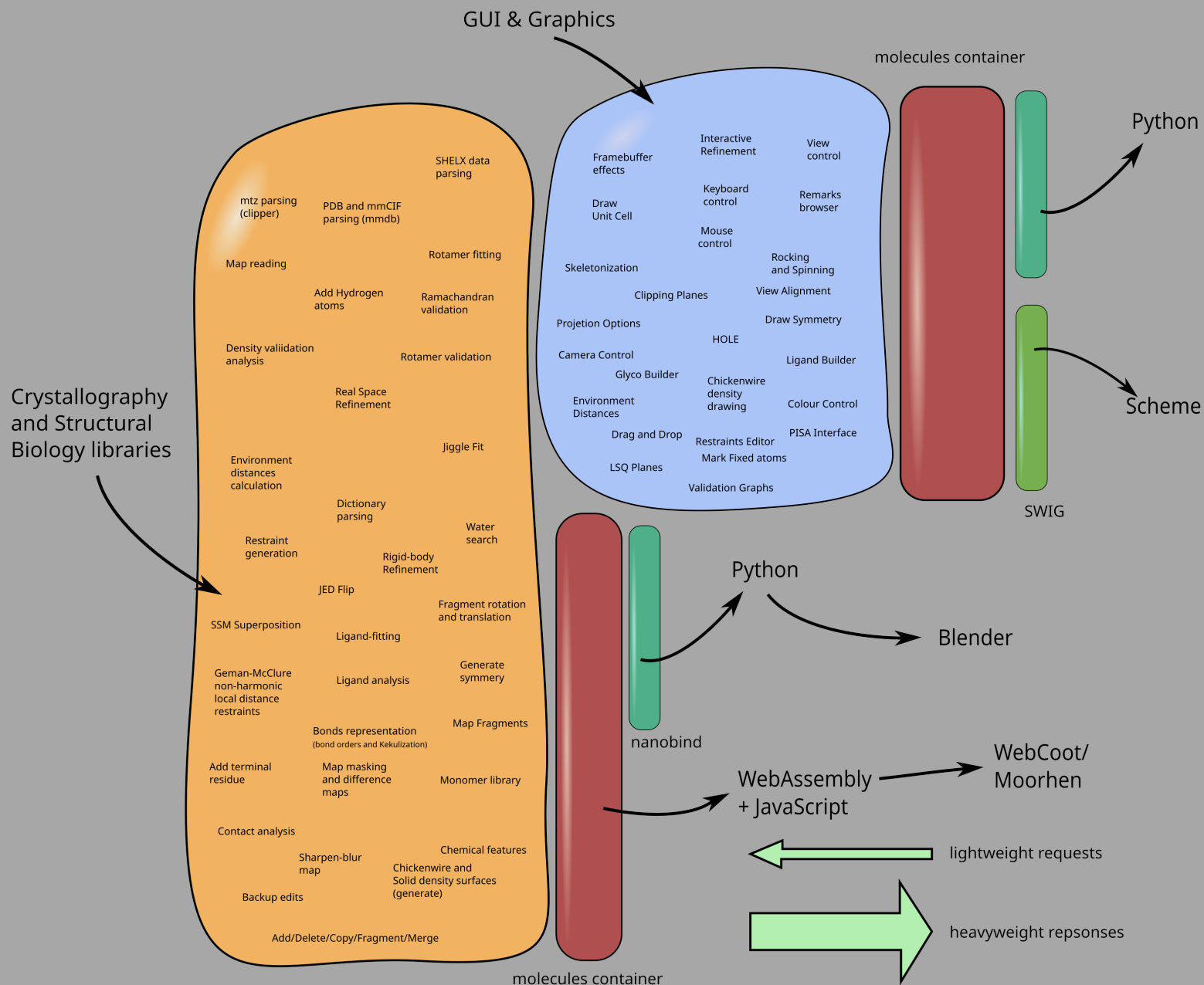
# What's New in The Land Of Coot?



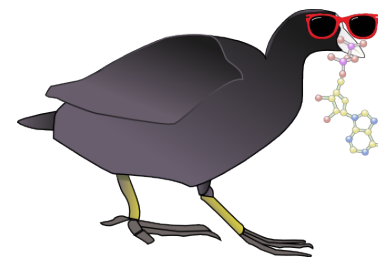
Lucrezia Catapano

CCP4 Study Weekend 2025

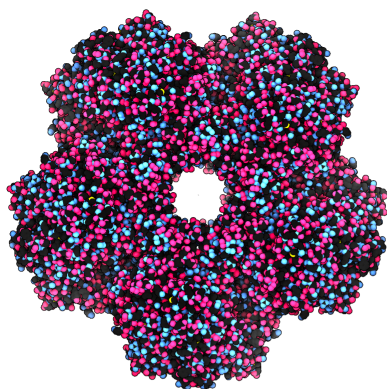
# Coot infrastructure



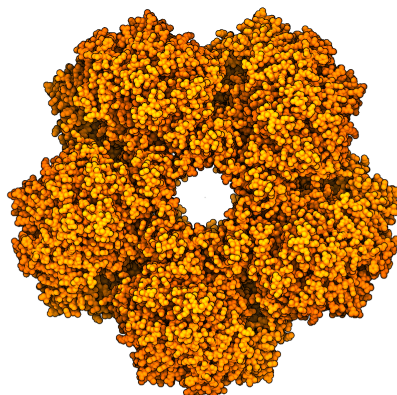
# What's new in Coot 1.1?



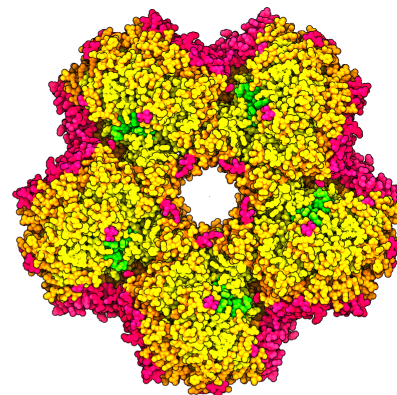
Attractive graphics



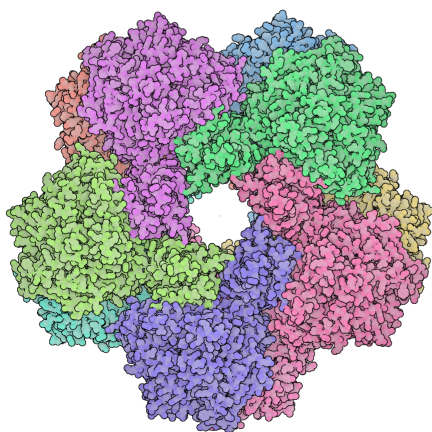
colour by atoms



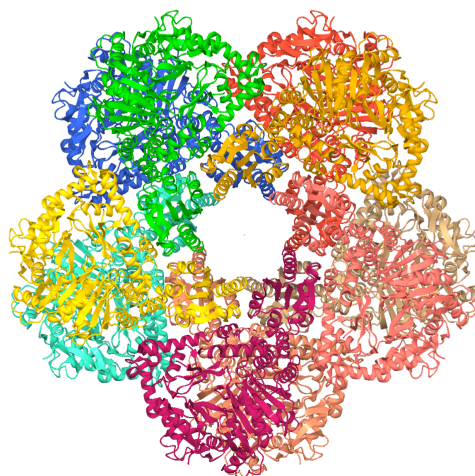
colour by molecule



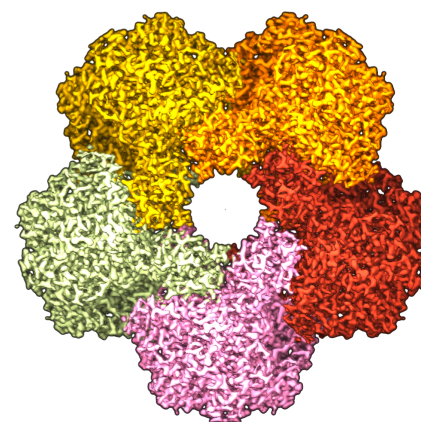
colour by B factors



Goodsell colours

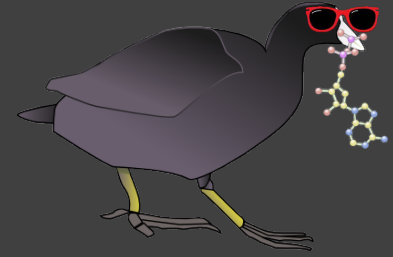


ribbons

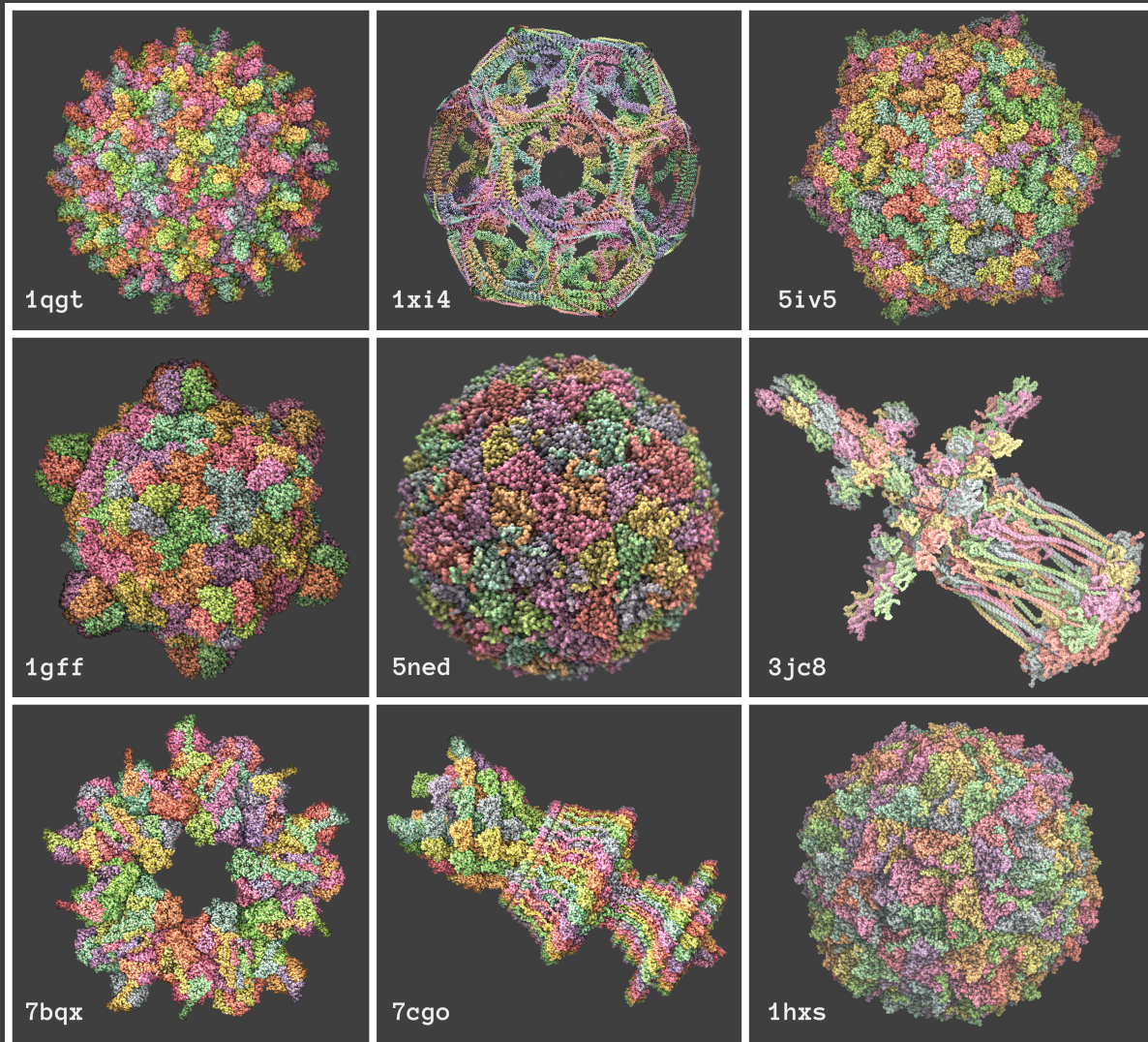


masked map

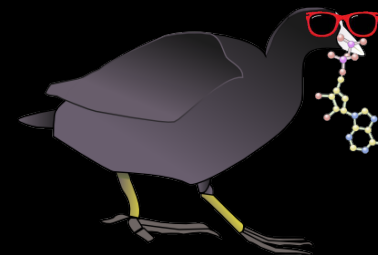
# What's new in Coot 1.1?



Attractive graphics

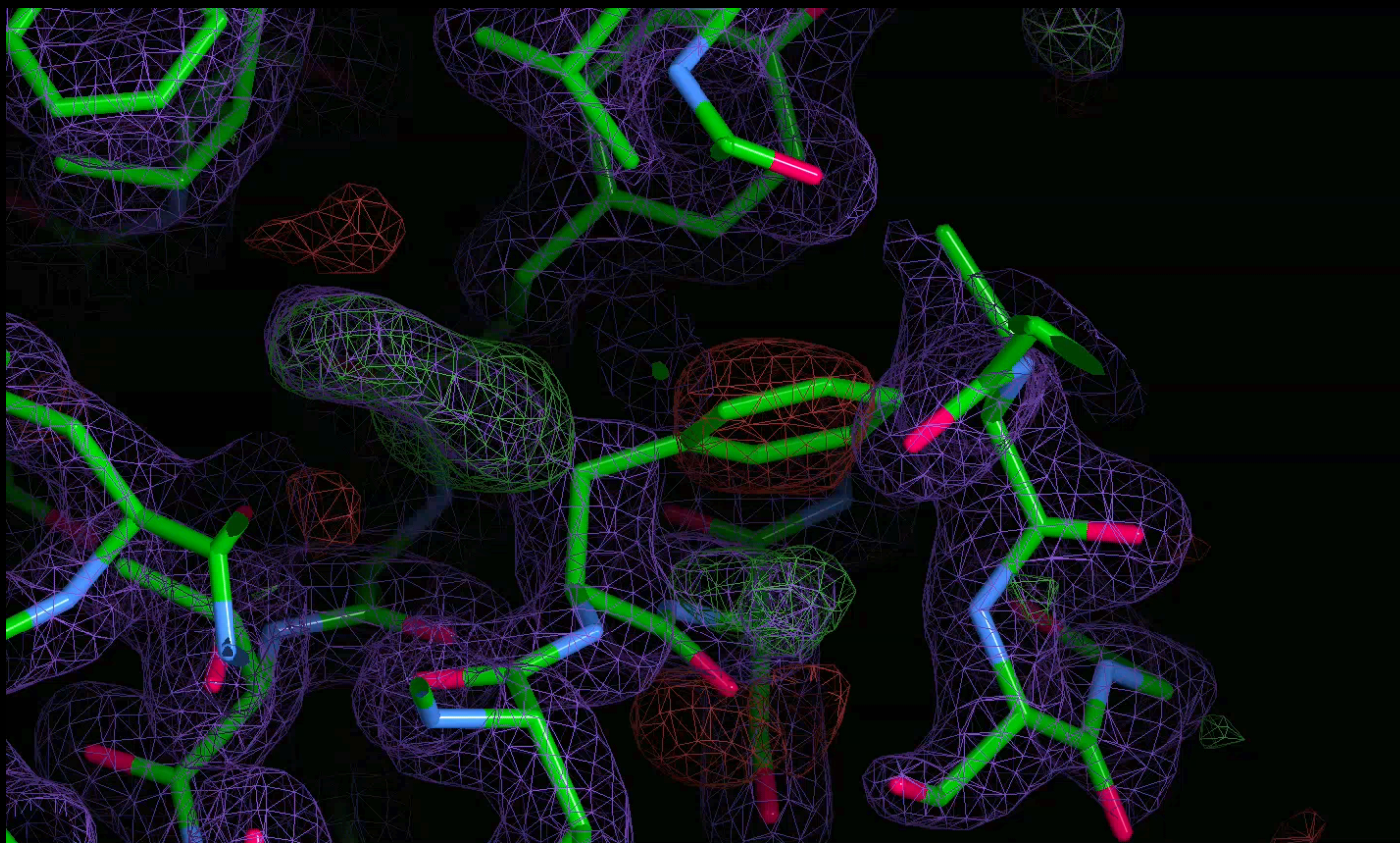


# What's new in Coot 1.1?

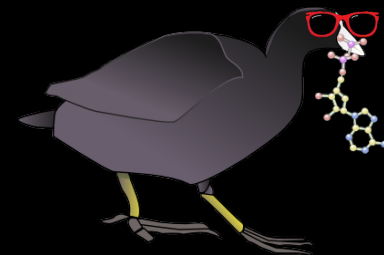


Noughties Physics has been re-introduced

Standard physics of atom movement

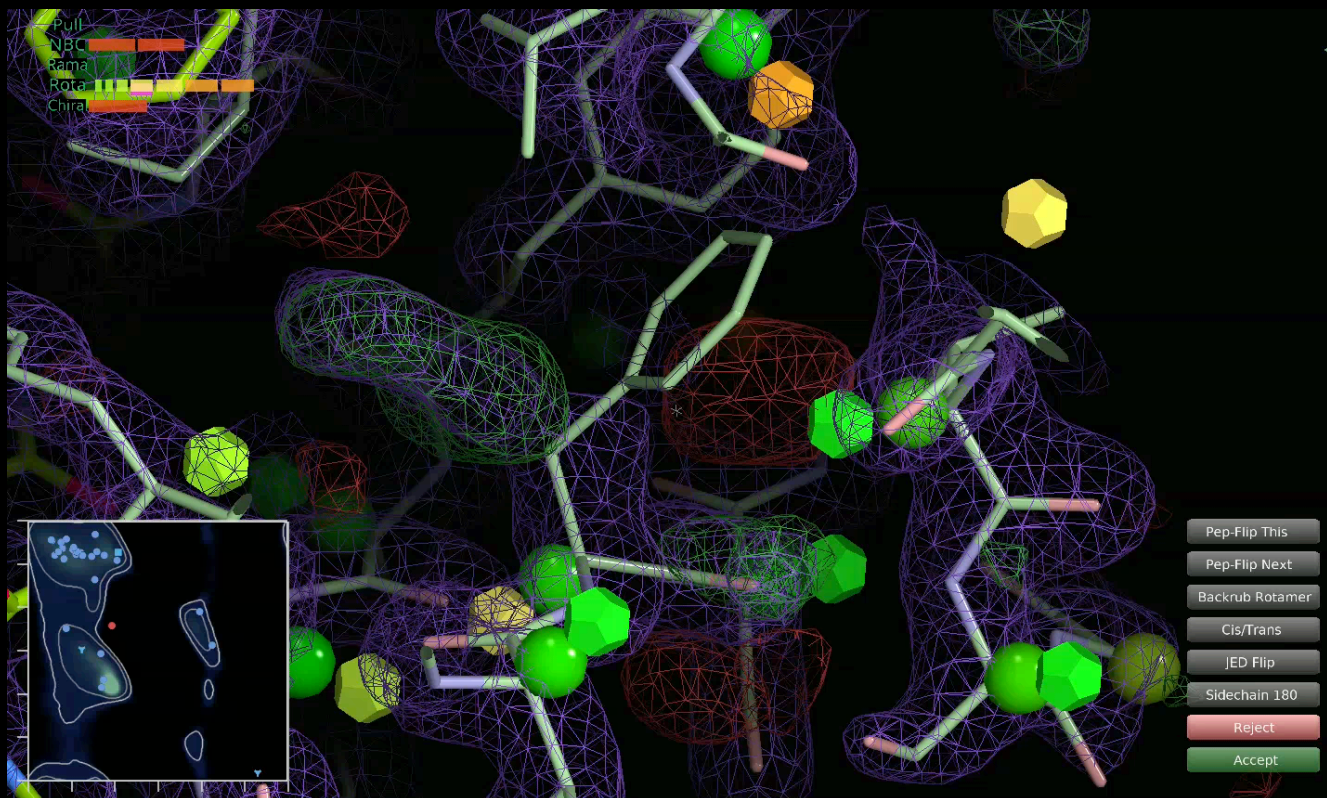


# What's new in Coot 1.1?

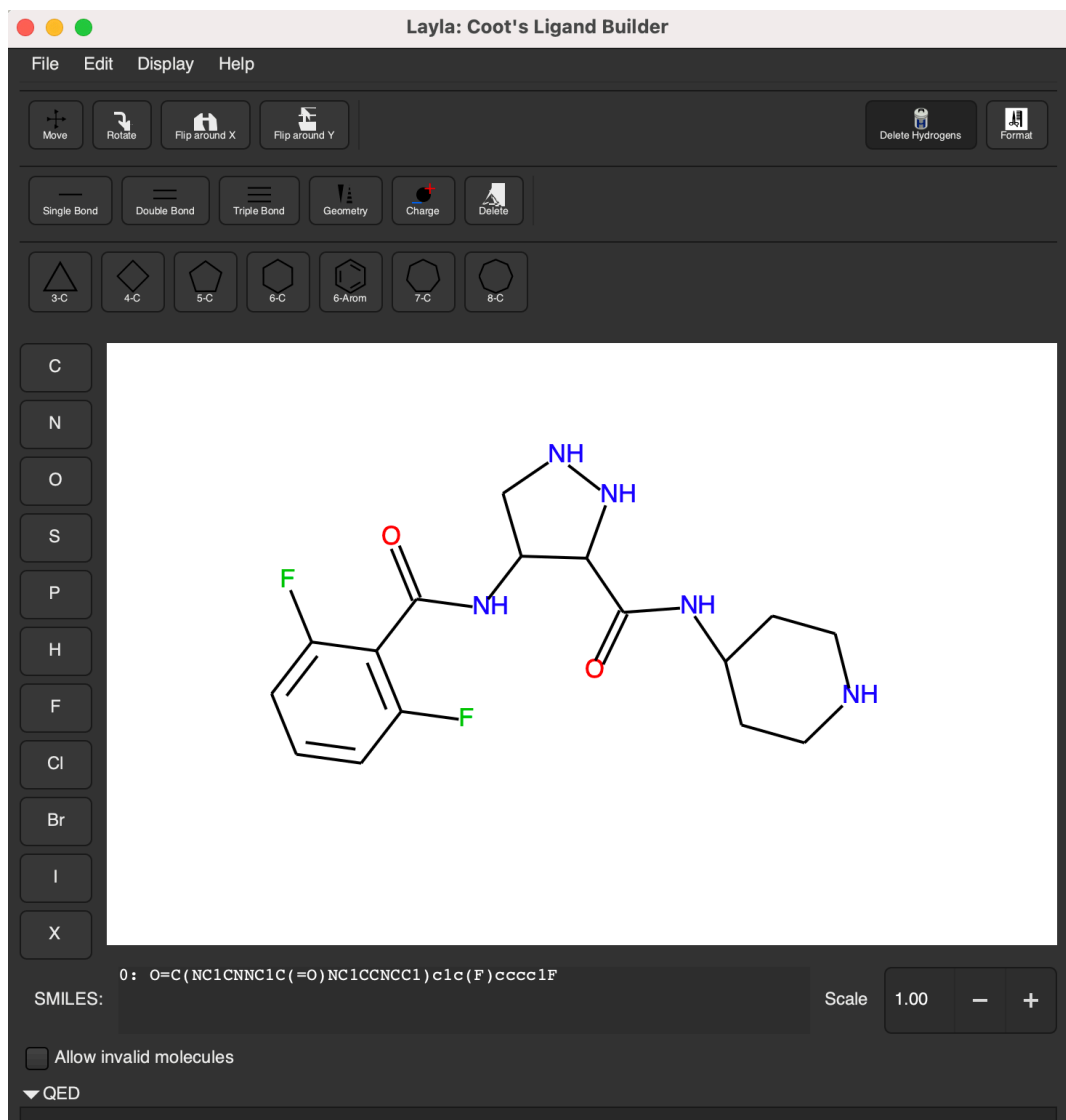
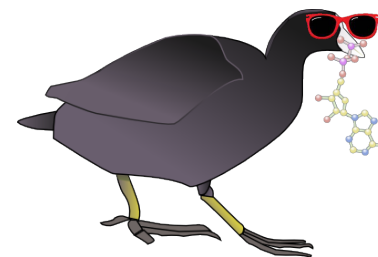


## Noughties Physics has been re-introduced

- We restore noughties physics ( from coot 0.8)
- elastic deformation of the atom positions as the picked atom is dragged without refinement (it doesn't care about bond distortions until you release the mouse)

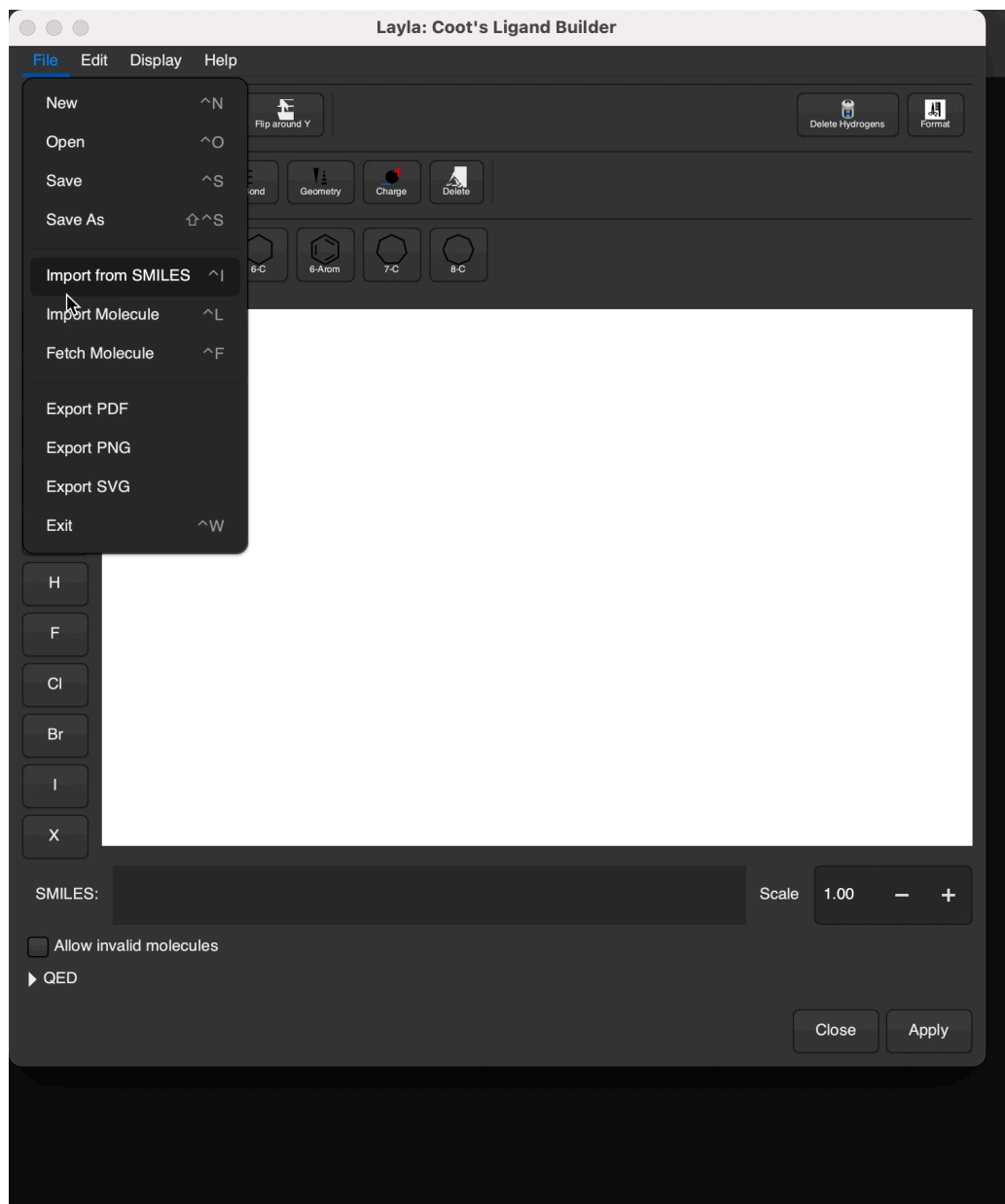
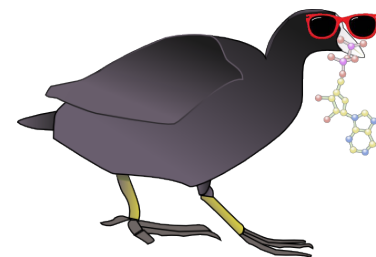


# What's new in Coot 1.1?



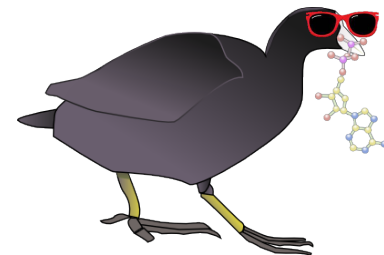
Ligand Builder and QED

# What's new in Coot 1.1?



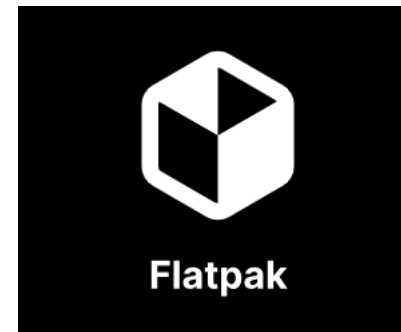
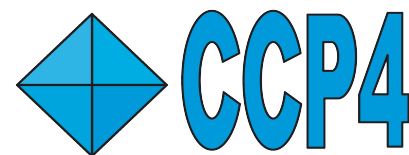
Ligand Builder and QED

# What's new in Coot 1.1?



## Availability

1. **CCP4 9.0** (not installed by default)
2. **Homebrew**
3. **Flatpak**
4. **Debian**
5. **Arch Linux**



# Moorhen

## Web-Based Interactive Model Building

This is a coot



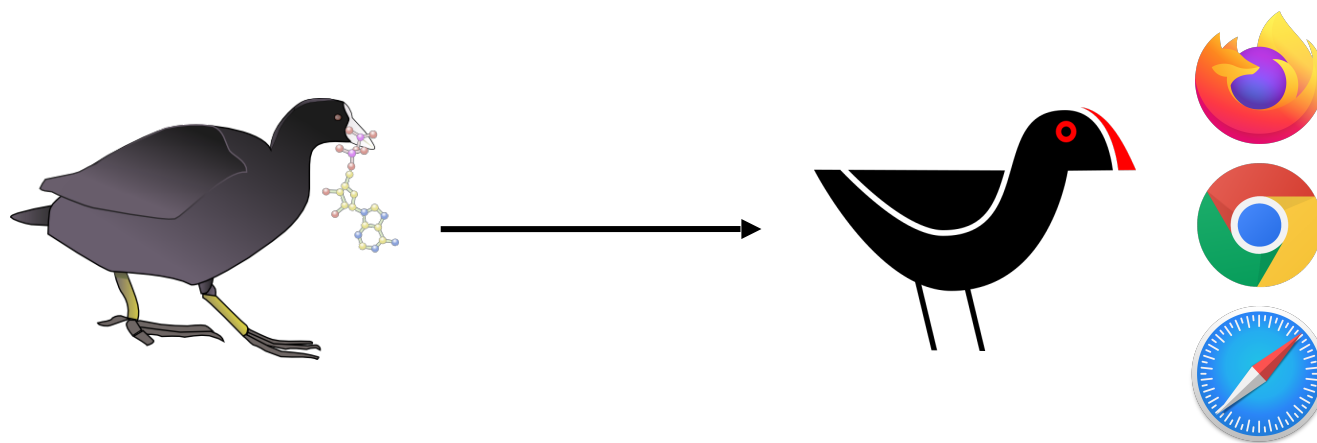
This is a moorhen



# What is Moorhen?

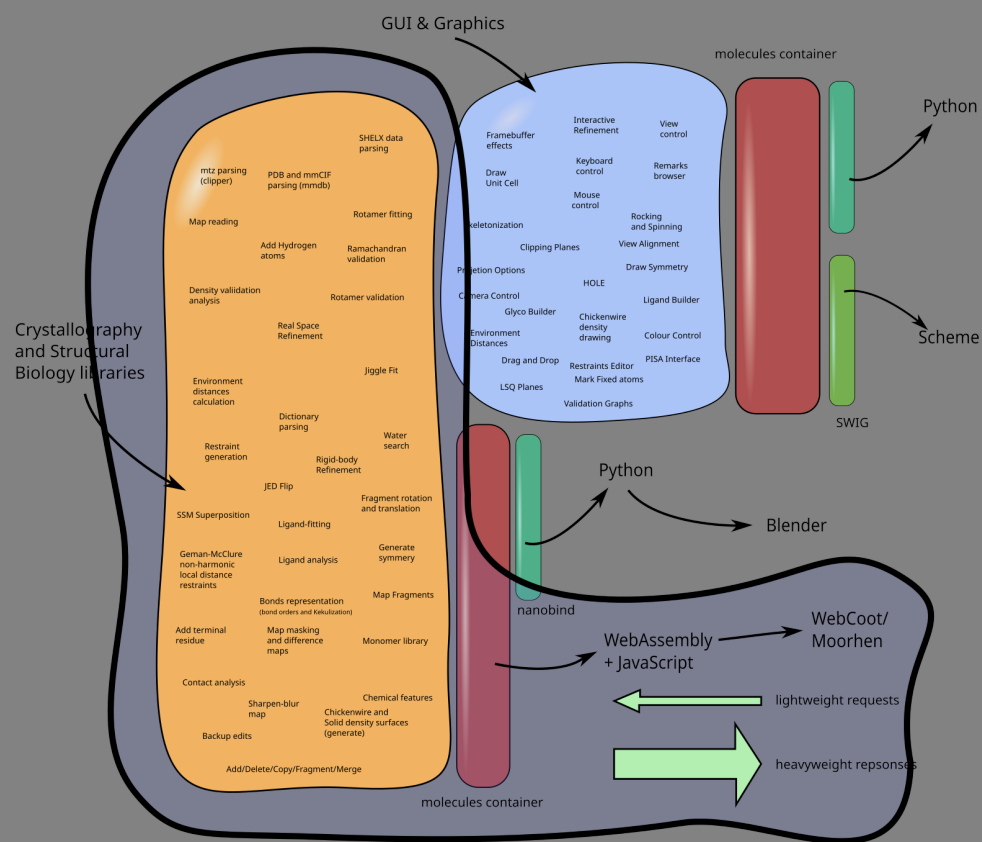
- Moorhen is a next-generation web-based application for the visualisation and manipulation of molecules in structure determination and analysis

→ In short, Coot on the web browser



# What is Moorhen?

→ Moorhen extends libccoot API with a web-based React GUI.



## Moorhen

React-based UI



## Coot API

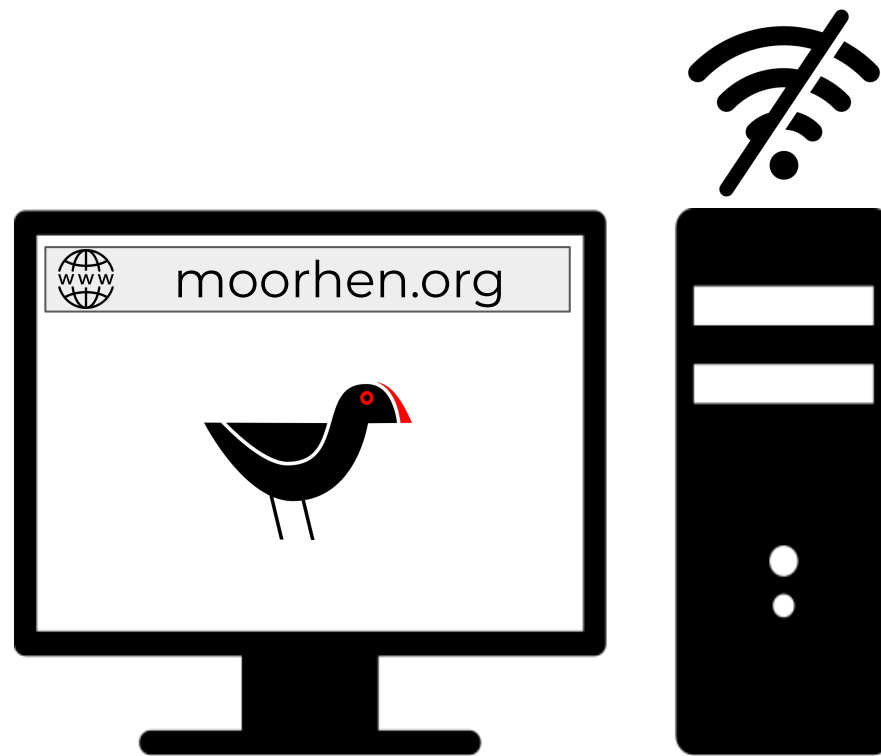
*\* shared with Coot \**

Model-Building  
Refinement  
Model Validation



# What is Moorhen?

- Moorhen is a client-side-only app.  
This means **there is no server-side computation**



# Current state of Moorhen

## Model Editing Features

Mutate Residue	Peptide Flip
Real Space Refinement	Auto-fit Rotamer
JED Flip	Add Residue
Check/Delete Waters	Rotate/Translate Residue
Delete Item	Drag Atoms
Edit Chi Angles	Fill Sidechain

## Presentation Features

SSM Superpose

Map Contouring

Map and Model colour change

Env. Distances

## Validation Features

Rama. Plot	Unmodeled Blobs
Density Fit	Diff. Map Peaks
Geom. Analysis	Combined Validation Plot
Rotamers	

# Current state of Moorhen

→ Moorhen is also intended to be a web-based replacement of CCP4MG

## Figure-Making Features

Multiple Model  
Representation  
Styles

Arbitrary Colour  
schemes

Basic Movie  
Making

Shadows

Depth Blur

Perspective  
Projection

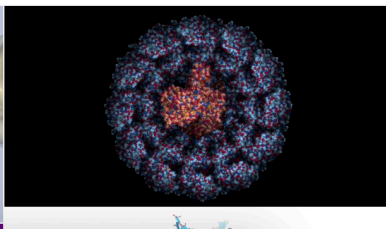
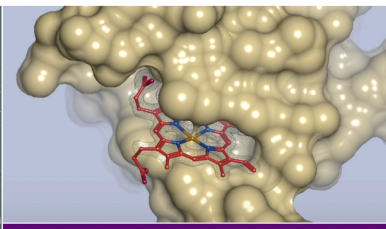
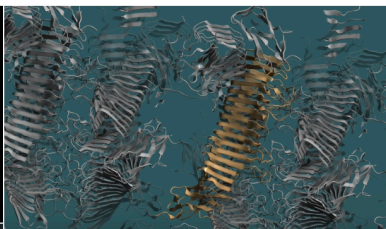
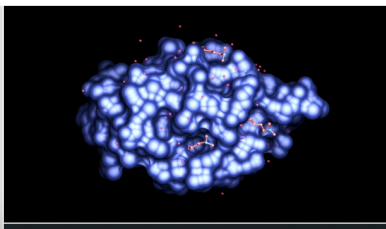
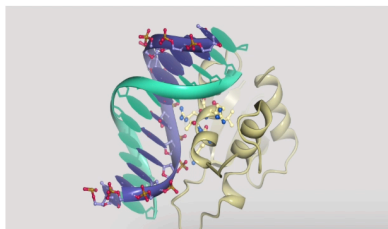
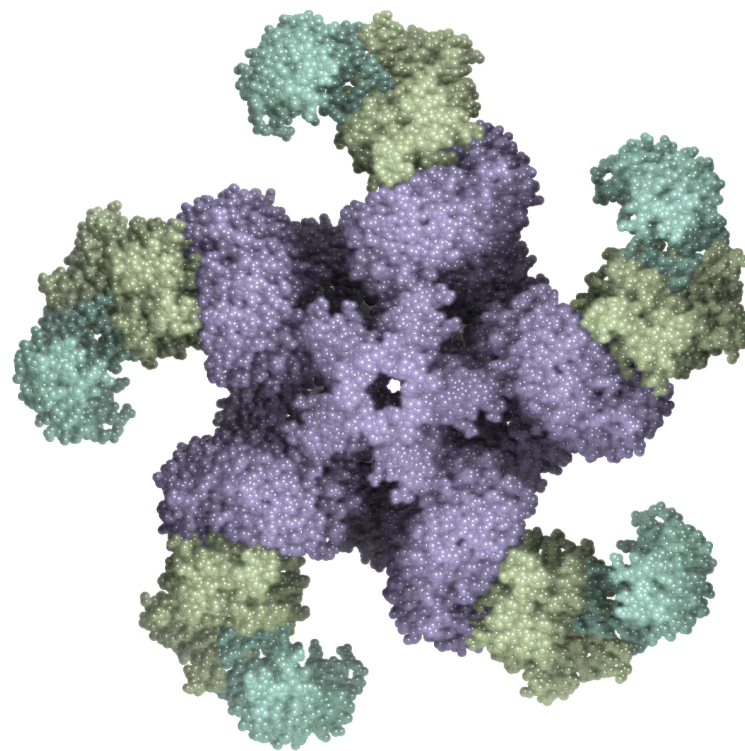
Clipping/Fogging

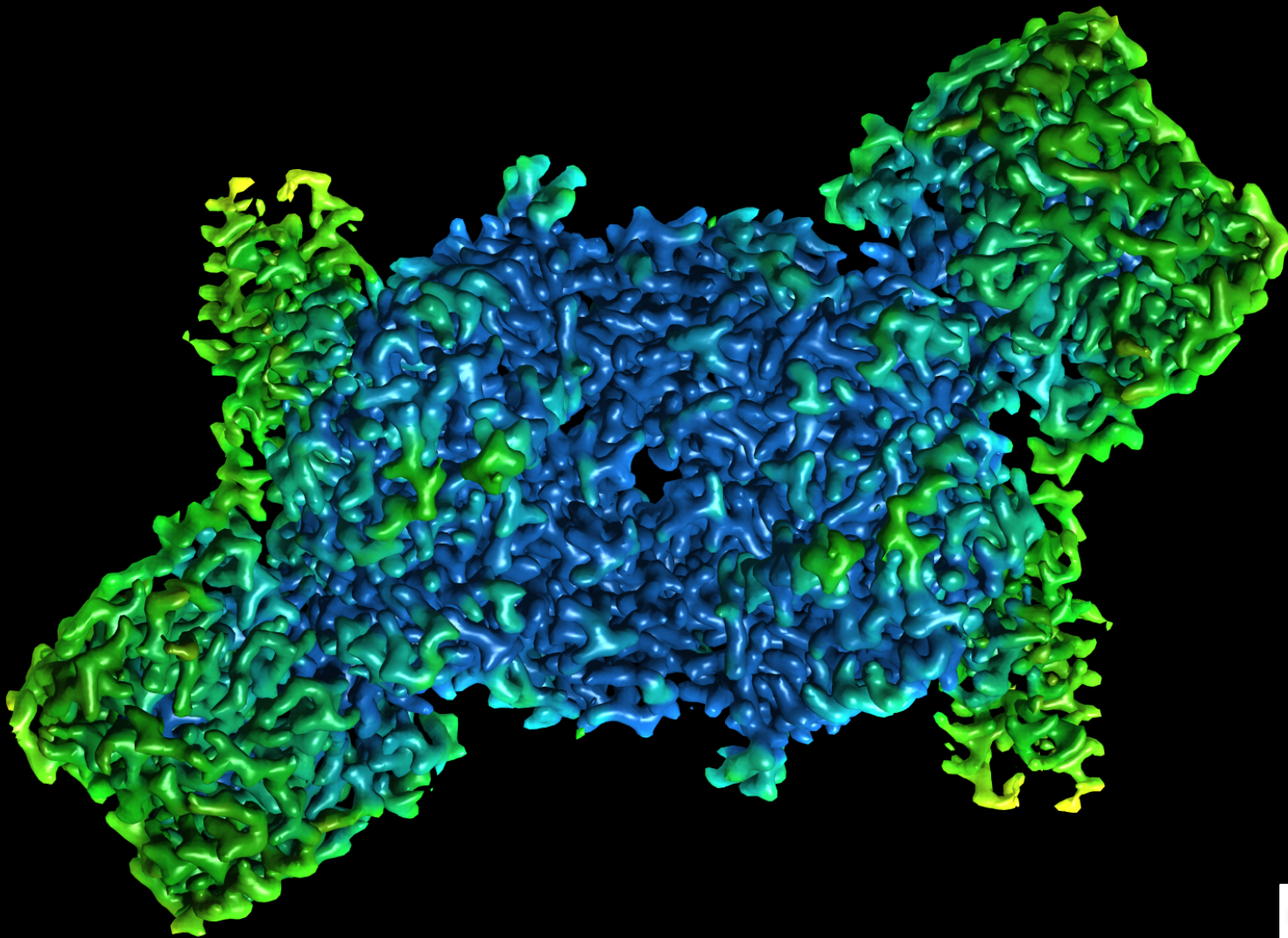
Ambient  
Occlusion

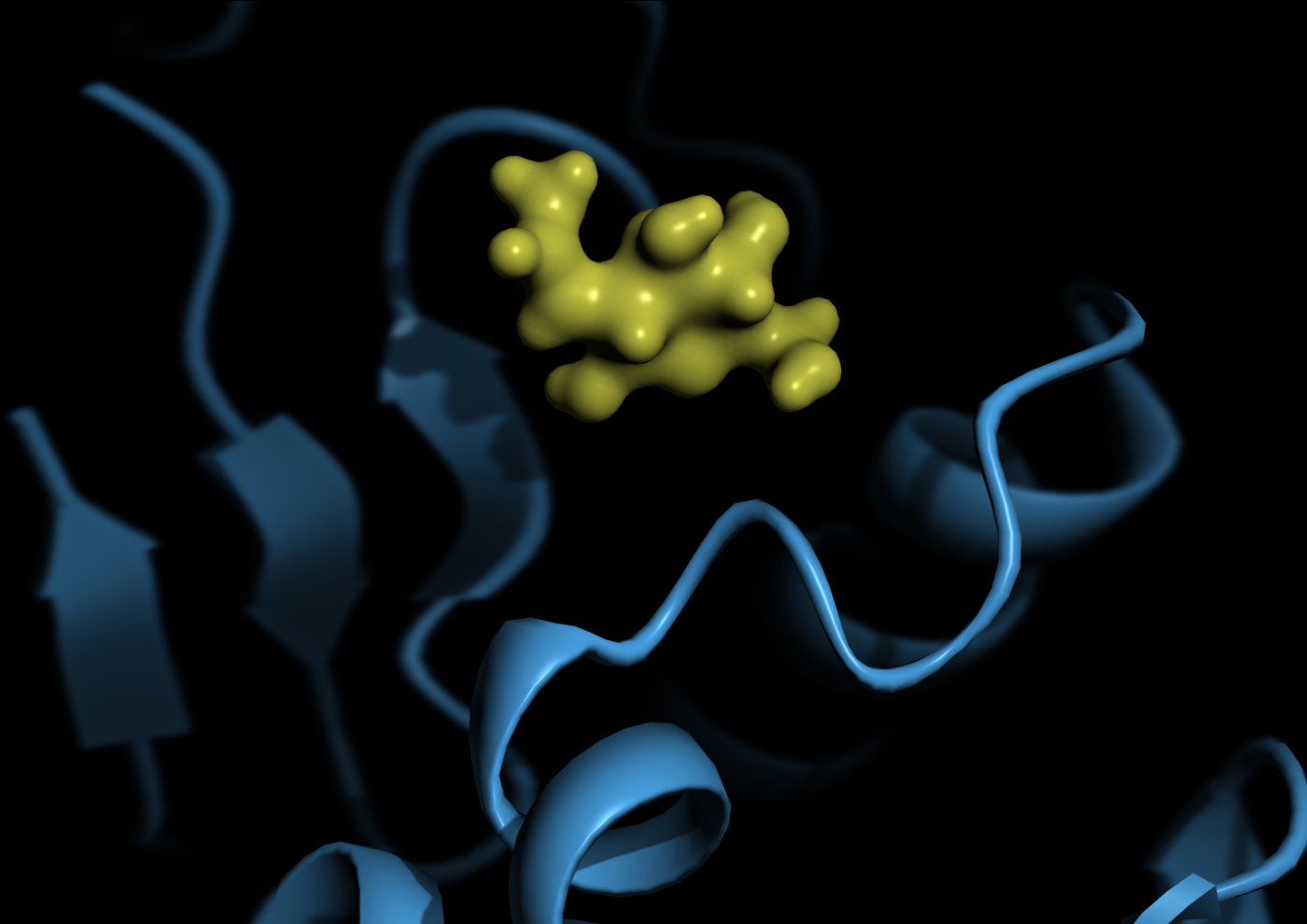
Screenshots

# Moorhen Gallery

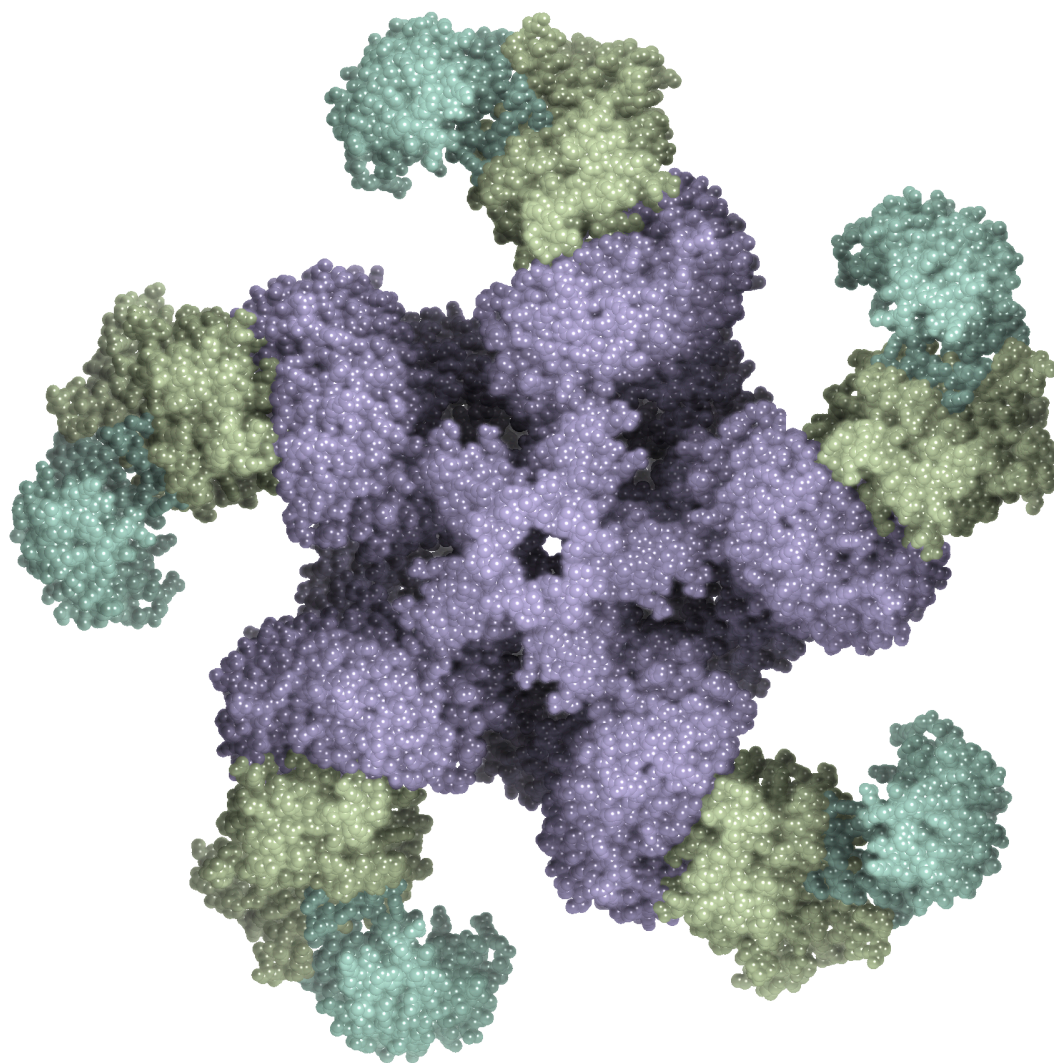
Screenshots and Animations provided by [moorhen.org](https://moorhen.org)



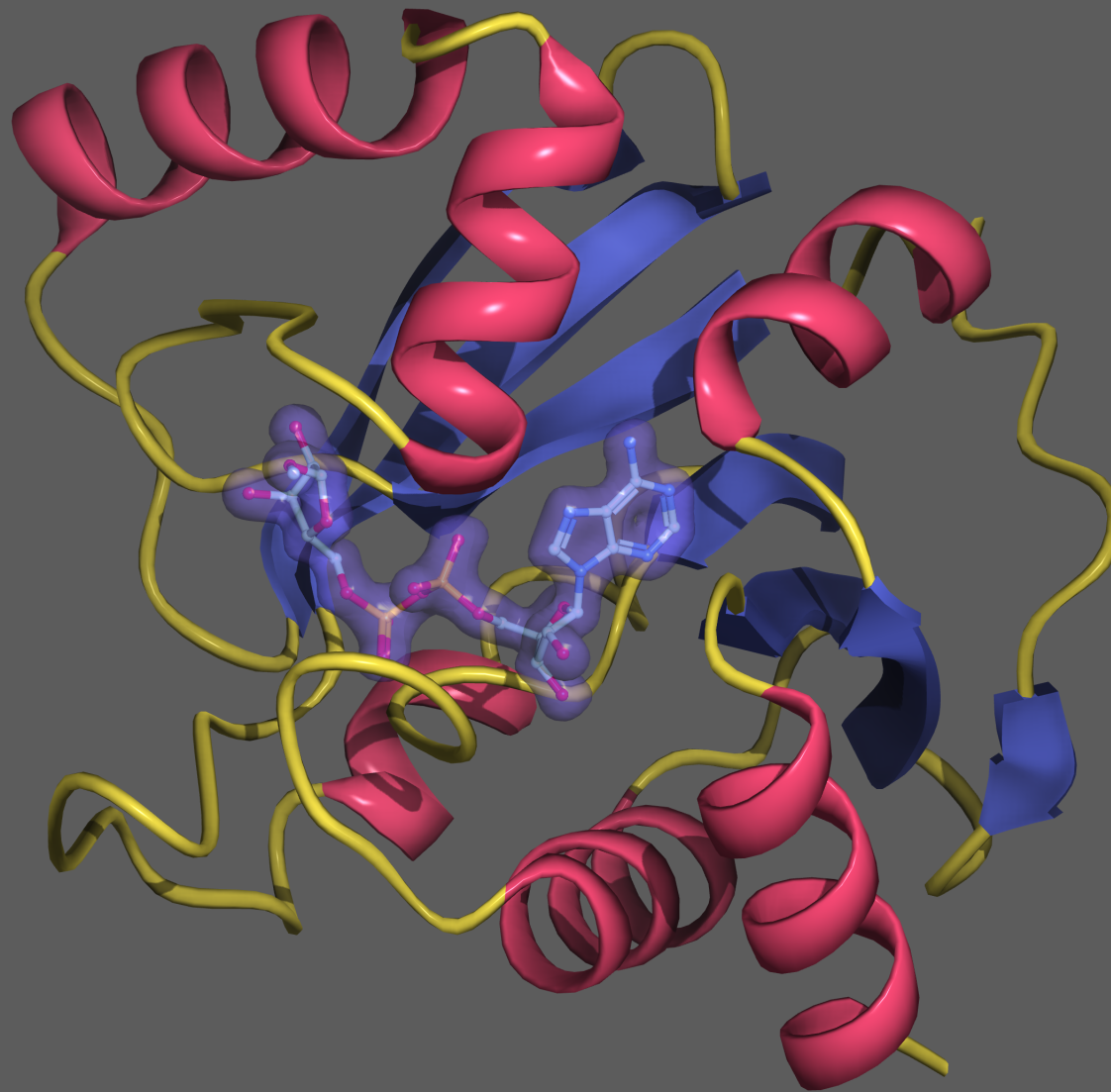




Shot on Moorhen

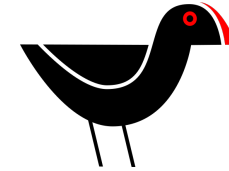


Shot on Moorhen

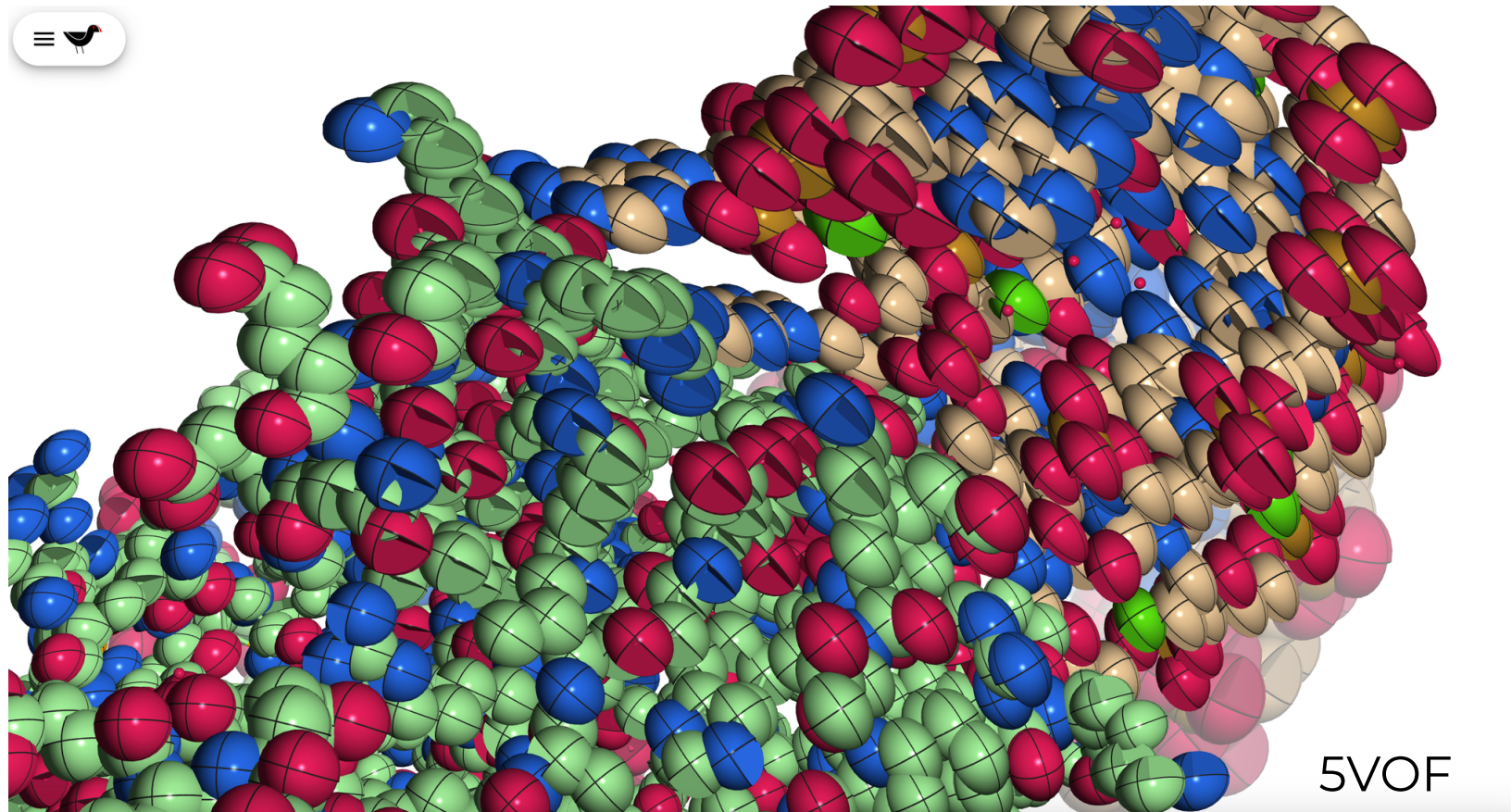


Shot on Moorhen

# What's new in Moorhen?

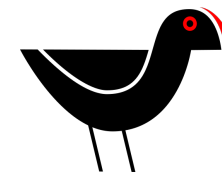


Anisotropic atoms representation

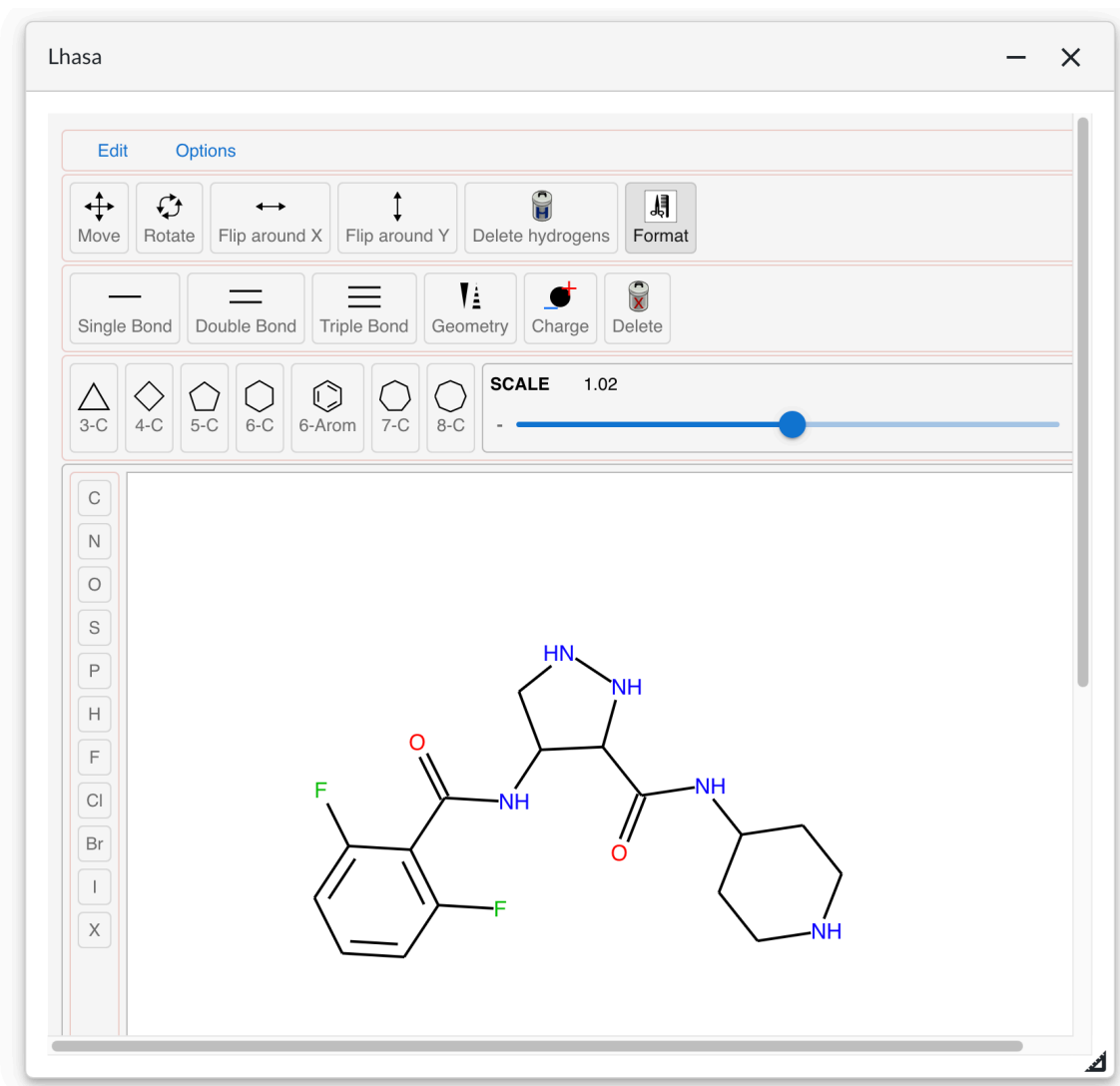


Thanks to Global Phasing

# What's new in Moorhen?



## Lhasa - Ligand Builder



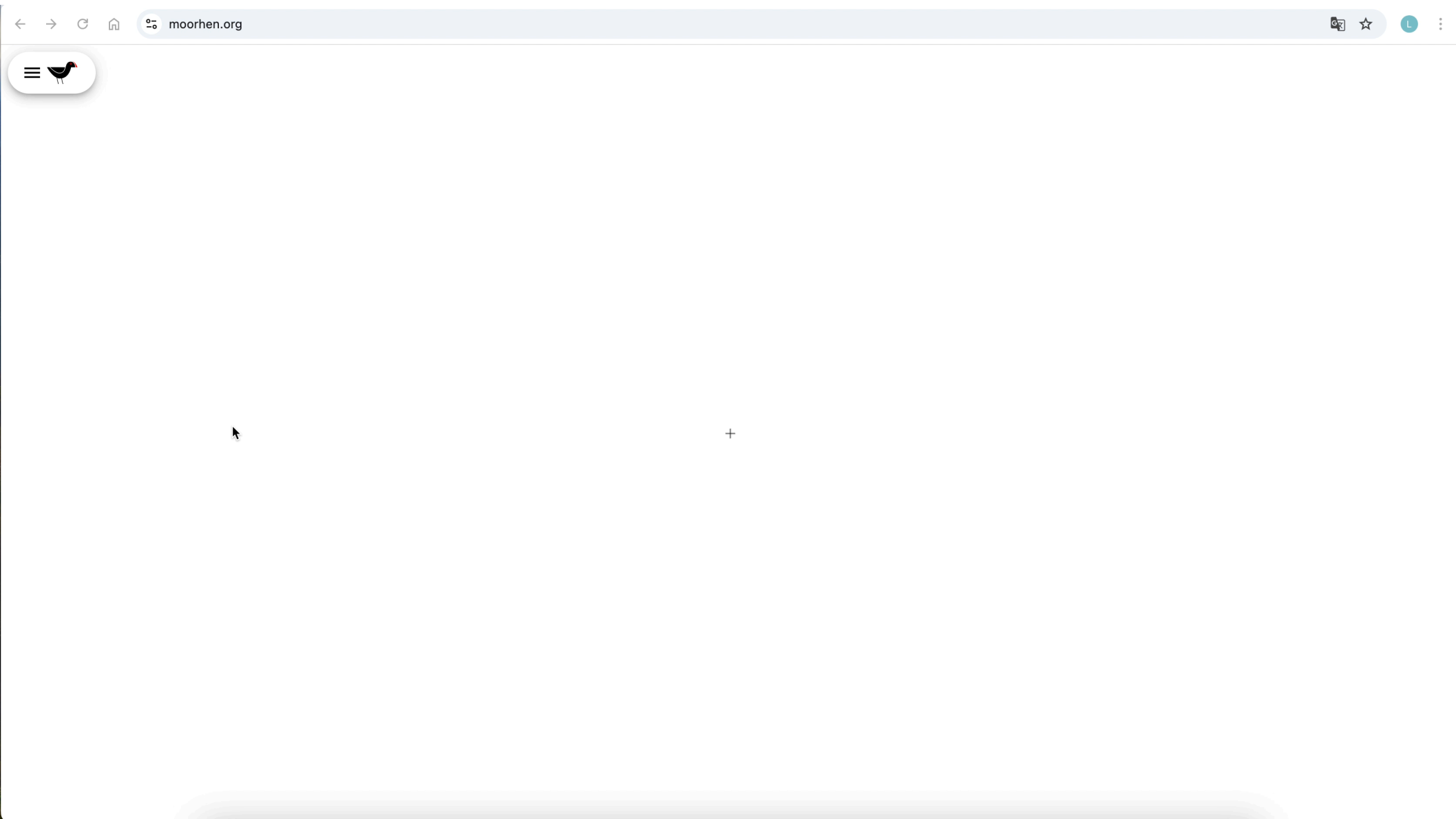
Will provide input for dictionary generators:

- AceDRG
- Grade 2
- eLBOW

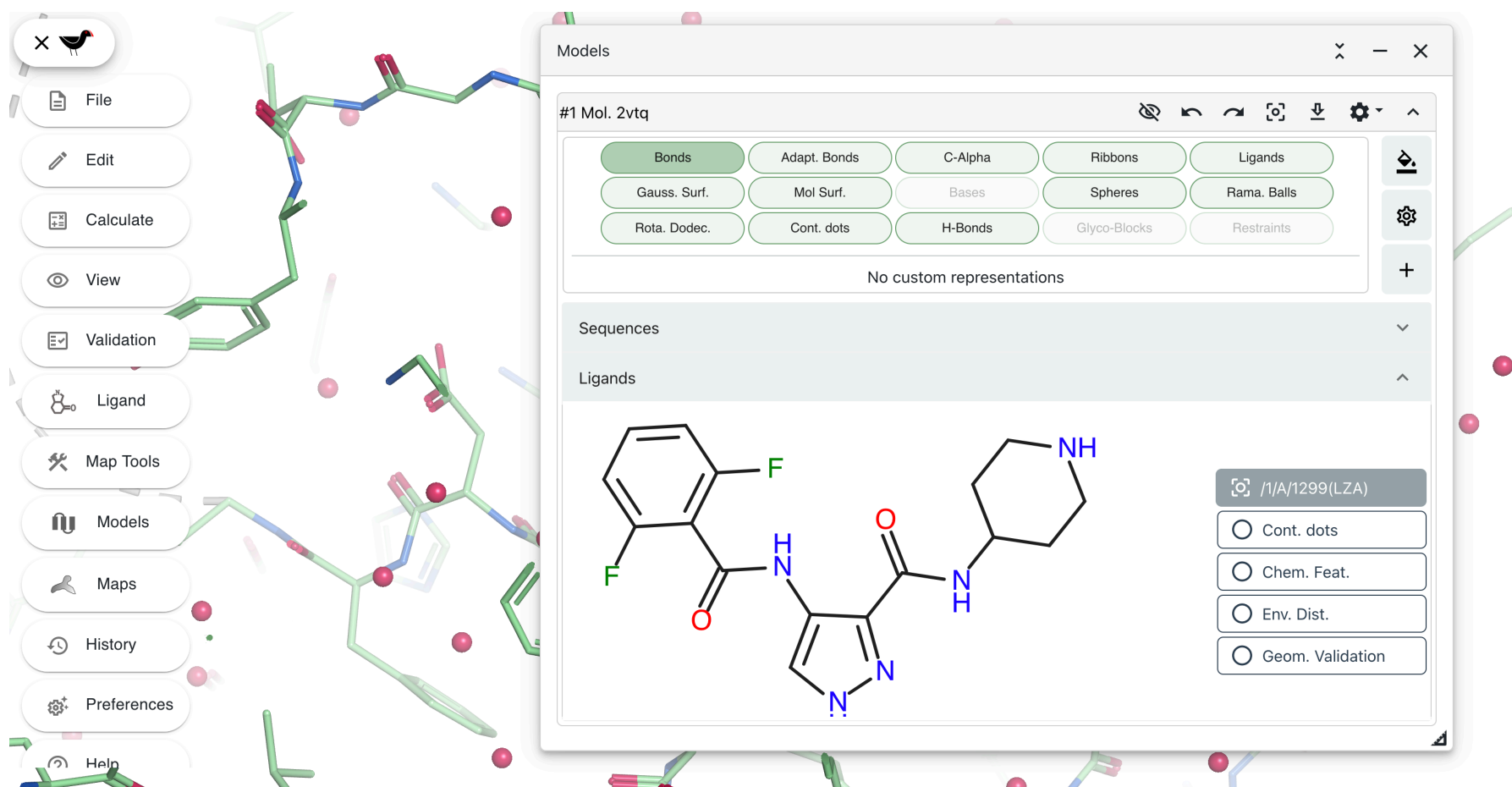


Jakub Smulski

# Moorhen – Ligand fitting

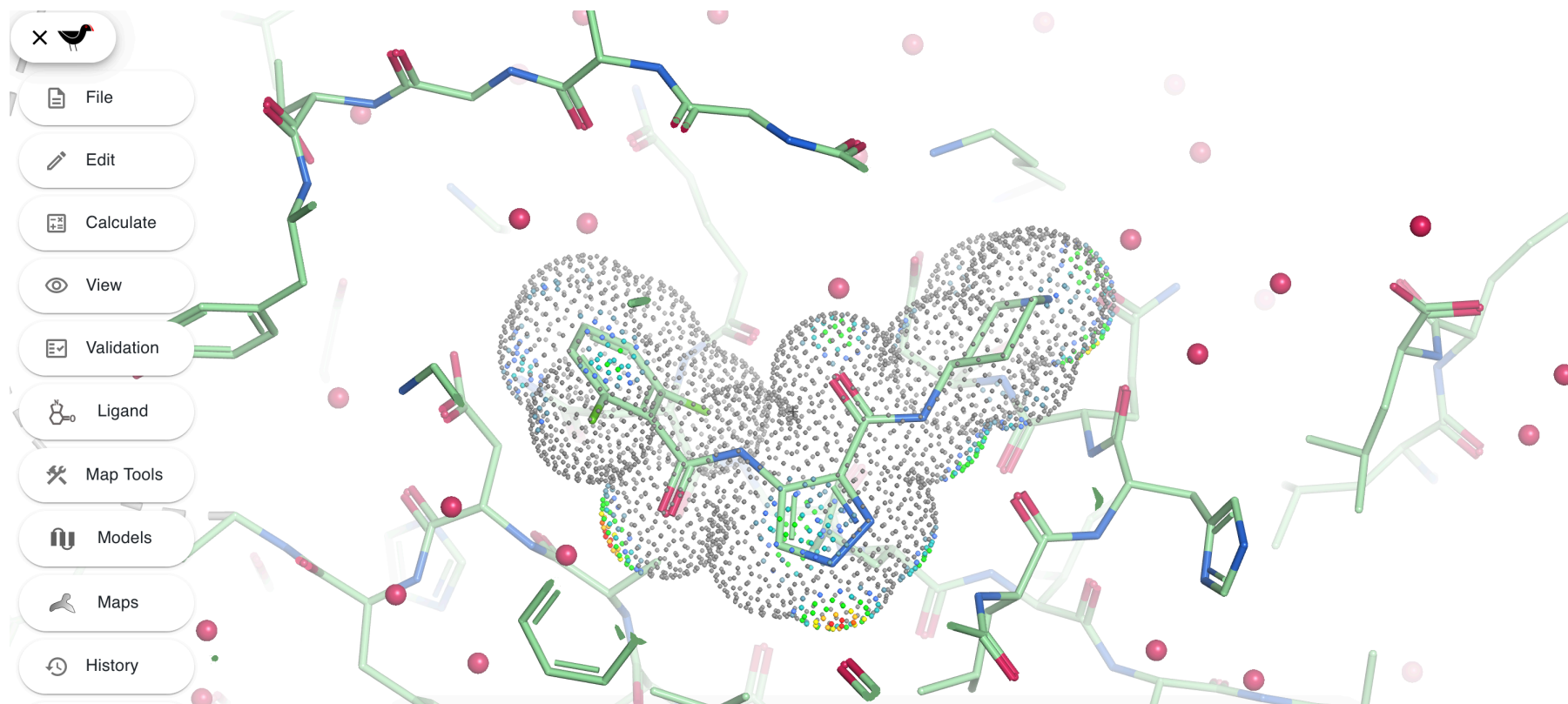


# Moorhen – Ligand validation



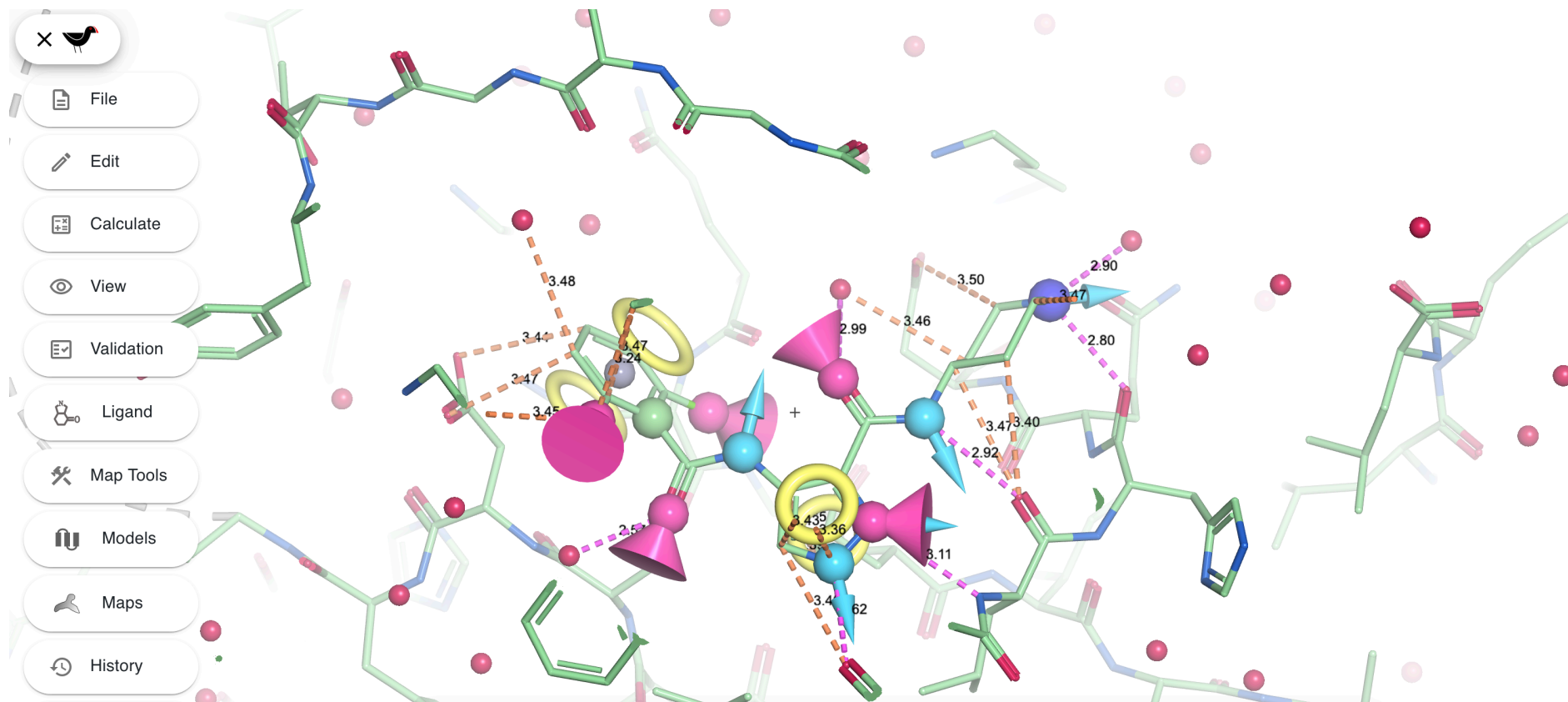
# Moorhen – Ligand validation

Contact dots



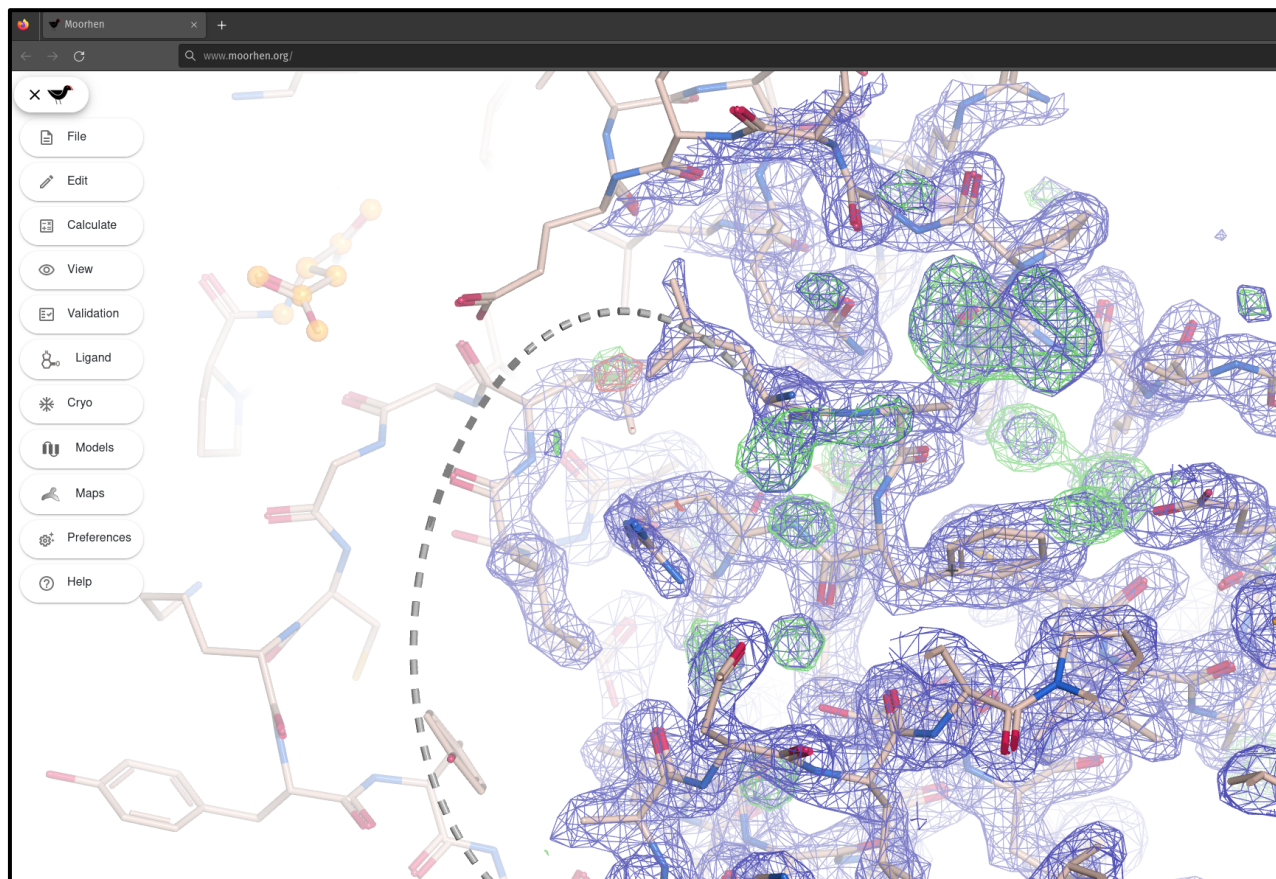
# Moorhen – Ligand validation

## Chemical features

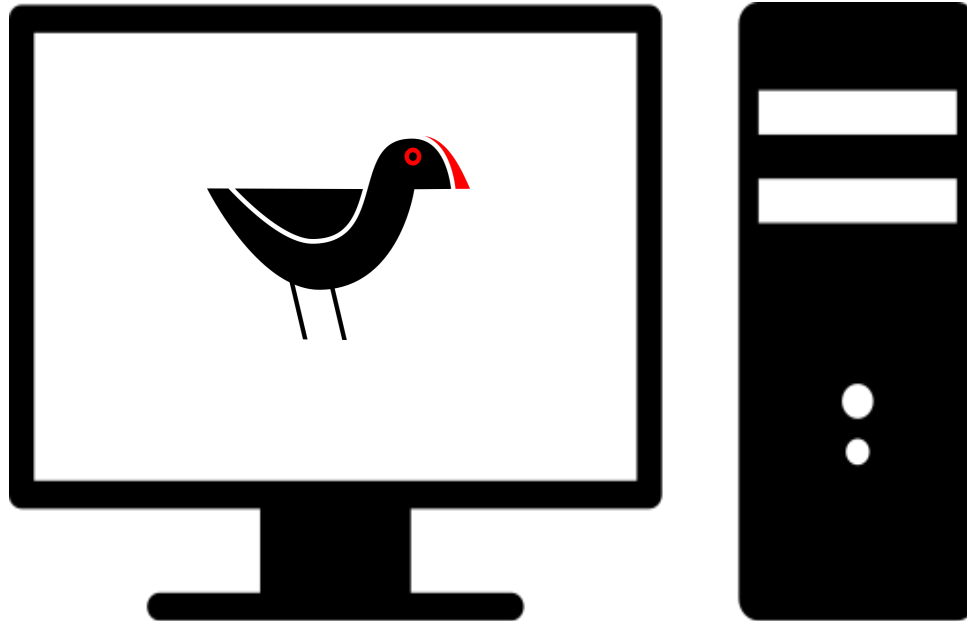
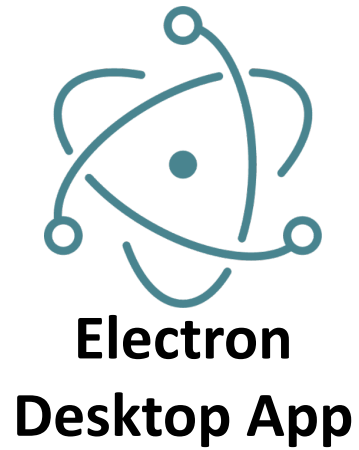


# Where is Moorhen available?

[www.moorhen.org](http://www.moorhen.org)



# Where is Moorhen available?



# Where is Moorhen available?



CCP4 Cloud

**Gamma**

[0052] modelcraft -- Compl=91.0%, R=0.239 R<sub>free</sub>=0.272

[0075] fit waters -- N<sub>waters</sub>=113

[0123] ccp4build -- running

Input Output End Stop




Report Main Log Service Log Errors

[0123] CCP4Build current structure

molecule: 1/A/117(SER)/CA

Cycle 3: N<sub>res</sub>=123 (100.0%), R/R<sub>free</sub>=0.2685/0.3142

▼ Coot (1)

-  **Model Building with Coot**  
\*\* task is available only if started via CCP4 Cloud Client
-  **Edit Coordinates with Coot**  
\*\* task is available only if started via CCP4 Cloud Client
-  **Model Building with WebCoot/Moorhen**  
-- !!EXPERIMENTAL!! fast-developing version of Coot for browsers

0.26  
0.25  
0.5 1.0 1.5 2.0 2.5 3.0 3.5  
Cycle No.

Print

Build in progress

[0107] buccaneer -- Compl=100.0% R=0.2729 R<sub>free</sub>=0.3032

Powered by CCP4 v8.0.010

CCP4  
v8.0.010

Iris

CCP4 Cloud v 1.7.11 [10.03.2023]

# Where is Moorhen available?



Project: ~/ccpem-project

PROJECT JOBS NODES NEW JOB

Filter jobs by type or alias

- 7 - Fetch - coot-em-tutorial
- 6 - Fetch - occupy-16890

RESULTS LOGS I/O PARAMS

Open with: PDF VIEWER TEXT EDITOR UGLYMOL MOL\* MOORHEN TERMINAL

Inputs to this job: no inputs

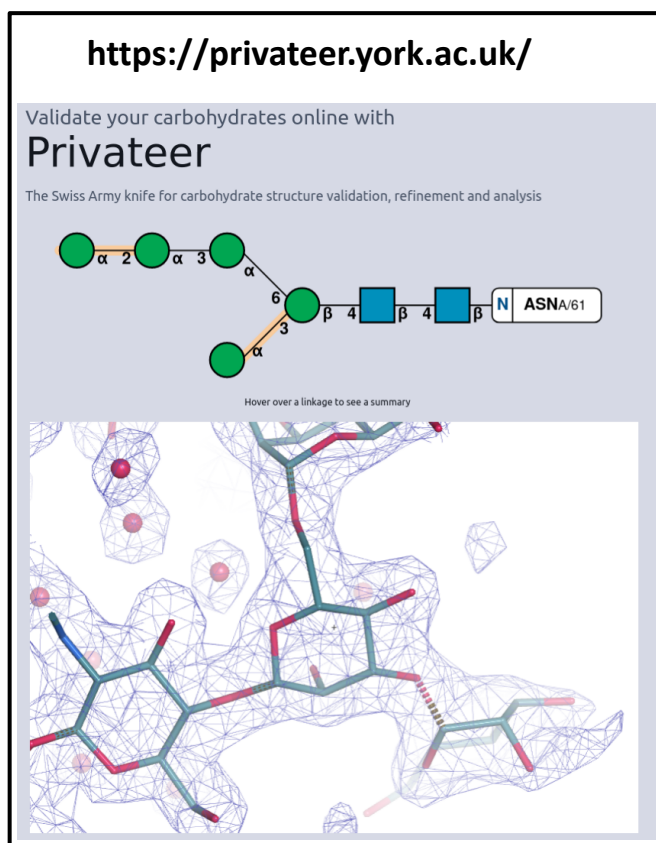
Outputs from this job:

- Fetch/job007/emd\_32143.mrc DensityMap from\_emdb
- Fetch/job007/pdb7vvl.pdb AtomCoords from\_pdb

An arrow points to the MOORHEN button in the "Open with:" section.

# Moorhen as a React component

→ Moorhen can be easily integrated to any other website to extend its capabilities.



Source: Dialpuri J. et al., (2024). Online carbohydrate 3D structure validation with the Privateer web app. (Manuscript submitted for publication)

# GitHub

https://github.com/moorhen-coot/Moorhen

## Moorhen

npm package 0.8.5 Nightly tests failing Deploy moorhen.org passing Dev docs passing Wiki passing

Moorhen is a web browser molecular graphics program based on the Coot desktop program. It is developed by porting some [CCP4](#) libraries and programs, [Coot](#), [FFTW2](#), [Privateer](#) and the [Gnu Scientific Library](#) to Web Assembly.

The emscripten suite of tools is required to do the compilation.

The sources of CCP4, Coot, Privateer, FFTW, and GSL are not included. They are downloaded and (possibly) patched by the running the `get_sources` script, which is part of the build process of this project.

The following libraries/programs are compiled to Web Assembly:

- libccp4 (8.0.0)
- clipper (20240123)
- ssm (1.4.0)
- mmdb2 (2.0.22)
- gemmi 0.6.4
- Coot 1.0 ('gtk3' git branch)
- fftw 2.1.5
- gsl 2.7.1
- Boost 1.83.0
- glm 0.9.9.8
- RDKit 2023\_09\_1

Moorhen is available to use at <https://moorhen.org>.

https://moorhen-coot.github.io/wiki/

## Moorhen Wiki

### Posts

Nov 3, 2023  
[Creating Figures with Moorhen](#)

Nov 2, 2023  
[Fetch data from Moorhen in your React app](#)

Jul 6, 2023  
[Using Moorhen in a react app](#)

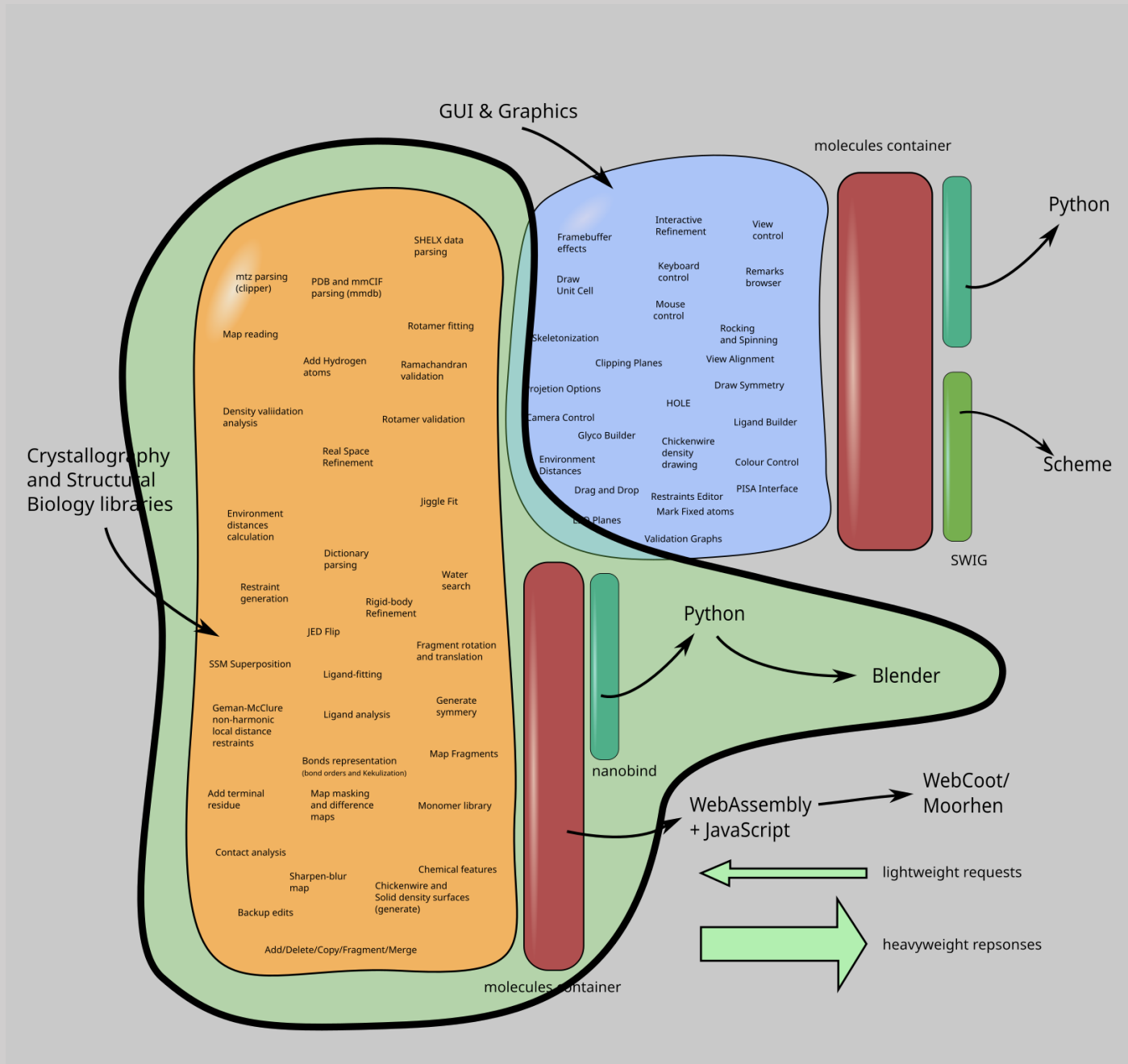
Apr 16, 2023  
[Moorhen Tutorial 1: Fix up the Cyclin-Dependent Kinase](#)

Moorhen mailing list:



<https://groups.google.com/a/york.ac.uk/g/moorhen-group>

# Programming with Coot



# Chapi documentation

<https://www.mrc-lmb.cam.ac.uk/lucrezia/libcootapi-documentation/>



[🏠 Coot API Documentation](#)

[Introduction](#)  
[Installation](#)  
[Reading Files](#)  
[Writing Files](#)  
[Molecular Models](#)  
[Python API](#)  
[C++ API](#)

[🏠 / Coot API Documentation](#)[View page source](#)

## Coot API Documentation

### Chapi

Cchapi is the alternative name for `coot_headless_api` and is the Pythonic interface to `libcootapi`. It is a clear and consistent and easy to use high level interface to the functions of **Coot**. On creating a new molecule, a *molecule index* will be returned. Molecules are referred to by this index and using the functions of `molecules_container_t`. This is unlike many other functions of Python modules, which return a Python representation of the data.

### Contents

- [Introduction](#)
- [Installation](#)
- [Reading Files](#)
  - [Coordinate Files](#)
  - [MTZ and Map Files](#)
- [Writing Files](#)
- [Molecular Models](#)
  - [Molecular Information](#)
  - [Molecular Editing](#)
- [Python API](#)

# Chapi documentation

## Detailed Python Reference API



[Coot API Documentation](#)

Search docs

[Introduction](#)

[Installation](#)

[Reading Files](#)

[Writing Files](#)

[Molecular Models](#)

**Python API**

[Basics Utilities](#)

[Reading and Writing](#)

[Molecular Information](#)

[Geometry and Dictionaries](#)

[Model Manipulation](#)

[Map Tools](#)

[Structure Factor](#)

[Real Space Refinement](#)

[Fitting](#)

[Validation](#)

[Molecular Graphics Representation](#)

[Testing functions](#)

[Blender functions](#)

[C++ API](#)

[Python API](#)

[View page source](#)

## Python API

### Basics Utilities

`class chapi.molecules_container_t` [\[source\]](#)

`set_make_backups(state: bool)→ None` [\[source\]](#)

Allow the user to disable/enable backups

**Parameters:** `state` – is True to mean that it is enabled. The default is True.

`get_make_backups()→ bool` [\[source\]](#)

Get the state of the backups

**Returns:** the backup-enabled state

`contains_unsaved_models()→ bool` [\[source\]](#)

Check if there are unsaved changes for this model

e.g. as yet not written to disk

**Returns:** a flag of unsaved models state - e.g. if any of them are unsaved, then this returns True.

`save_unsaved_model_changes()→ None` [\[source\]](#)

Save the unsaved model - this function has not yet been written!

`set_show_timings(s: bool)→ None` [\[source\]](#)

Set the show\_timings flag

Various (not all) functions in this class can calculate how long they took to run. Setting this will write the time to taken (in milliseconds) to stdout.

# Chapi documentation



## Python script examples

### Example #1: adding water molecules

```
import chapi

mc = chapi.molecules_container_t(True)

# read coordinates and map
imol = mc.read_pdb('tutorial-modern.pdb')
imol_mtz = mc.read_mtz("rnasa-1.8-all_refmac1.mtz", "FWT", "PHWT", "W", False, False)

# set the parameters for waters addition (the default values are given as arguments)
mc.set_add_waters_water_to_protein_distance_lim_min(2.4)
mc.set_add_waters_water_to_protein_distance_lim_max(3.4)
mc.set_add_waters_variance_limit(0.1)
mc.set_add_waters_sigma_cutoff(1.75)

# add waters
mc.add_waters(imol, imol_mtz)
```

# Chapi documentation



## Python script examples

### Example #2 : deleting water molecules outliers

```
# read coordinates and map
imol = mc.read_pdb('tutorial-modern.pdb')
imol_mtz = mc.read_mtz("rnasa-1.8-all_refmac1.mtz", "FWT", "PHWT", "W", False, False)

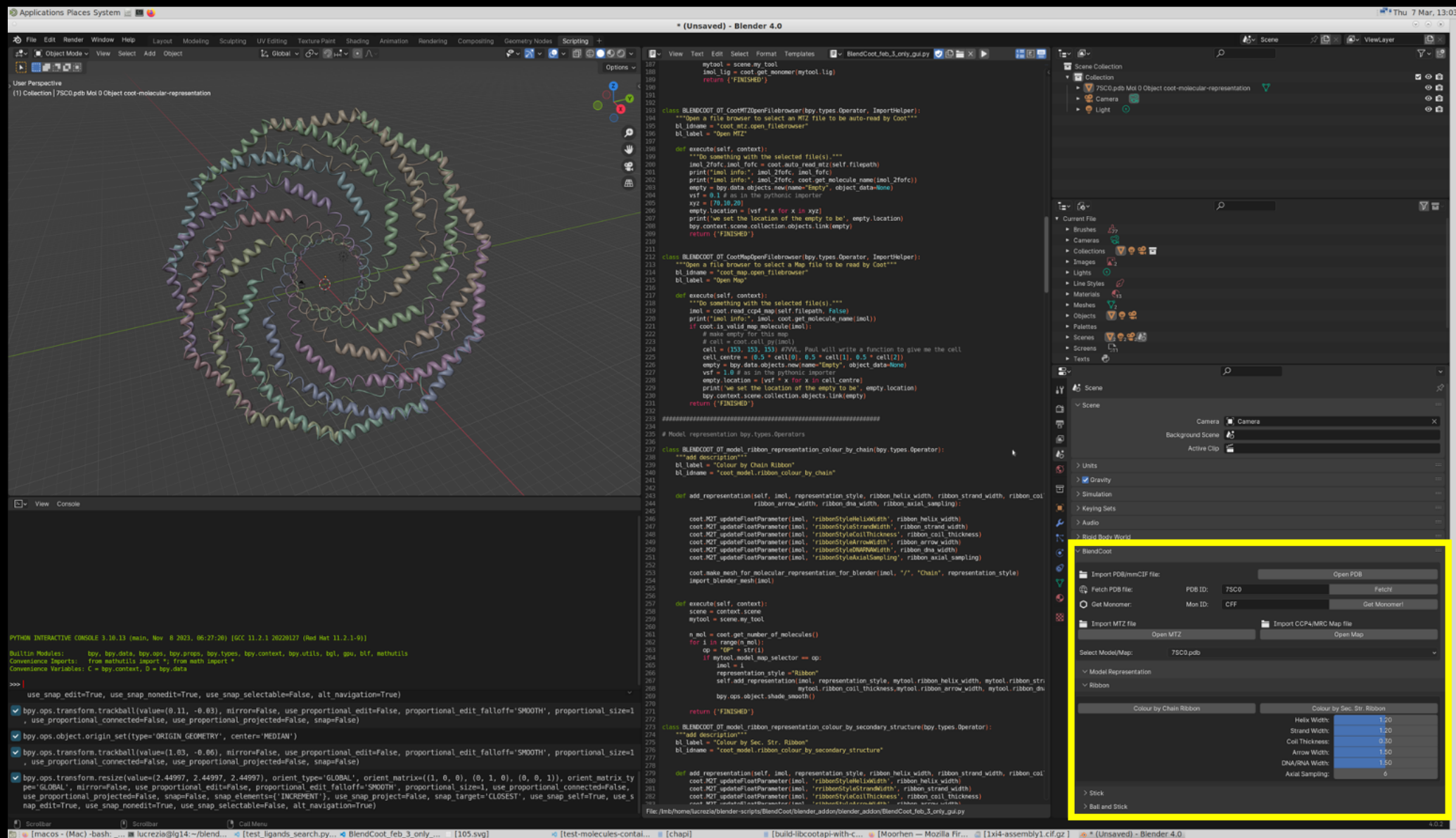
# delete water "outliers" - e.g., those with a distance to the protein less than 2.5
# or more than 3.5
min_dist = 2.5
max_dist = 3.5
median_temperature_factor = mc.get_median_temperature_factor(imol)
b_factor_limit = 2.0 * median_temperature_factor
outlier_map_rmsd_level = 1.0
ignore_part_occ_contact_flag = False
ignore_zero_occ_flag = False
water_outliers = mc.find_water_baddies(imol,
                                       imol_mtz,
                                       b_factor_limit,
                                       outlier_map_rmsd_level,
                                       min_dist,
                                       max_dist,
                                       ignore_part_occ_contact_flag,
                                       ignore_zero_occ_flag)

for res in water_outliers:
    cid = '/' + res.chain_id + '/' + str(res.res_no)
    print("Deleting water", cid)
    mc.delete_atom_using_cid(imol, cid)
```

# BlendCoot

- **Blender** is 3D modelling software for graphics & ray-tracing
- **BlendCoot** is an interface between Coot and Blender
- BlendCoot builds on the Pythonic non-graphical interface **chapi**
- We have added a blender-based GUI to chapi to provide an easy means to import molecules, ligands and maps into Blender

# BlendCoot



# BlendCoot

BlendCoot

Import PDB/mmCIF file:

Open PDB

Fetch PDB file:

PDB ID:

7SC0

Fetch!

Get Monomer:

Mon ID:

CFF

Get Monomer!

Import MTZ file

Open MTZ

Import CCP4/MRC Map file

Open Map

Select Model/Map:

Model Representation

Ribbon

Colour by Chain

Colour by Sec. Str.

Helix Width:

1.20

Strand Width:

1.20

Coil Thickness:

0.30

Arrow Width:

1.50

DNA/RNA Width:

1.50

Axial Sampling:

6

> Bases

> Stick

> Ball and Stick

> Sphere

> Gaussian Surface

> Goodsell Style

Map Representation

Wireframe

Surface

Contour Level:

0.02

Map Radius:

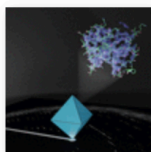
99.00

Shannon Sampling ...

1.00

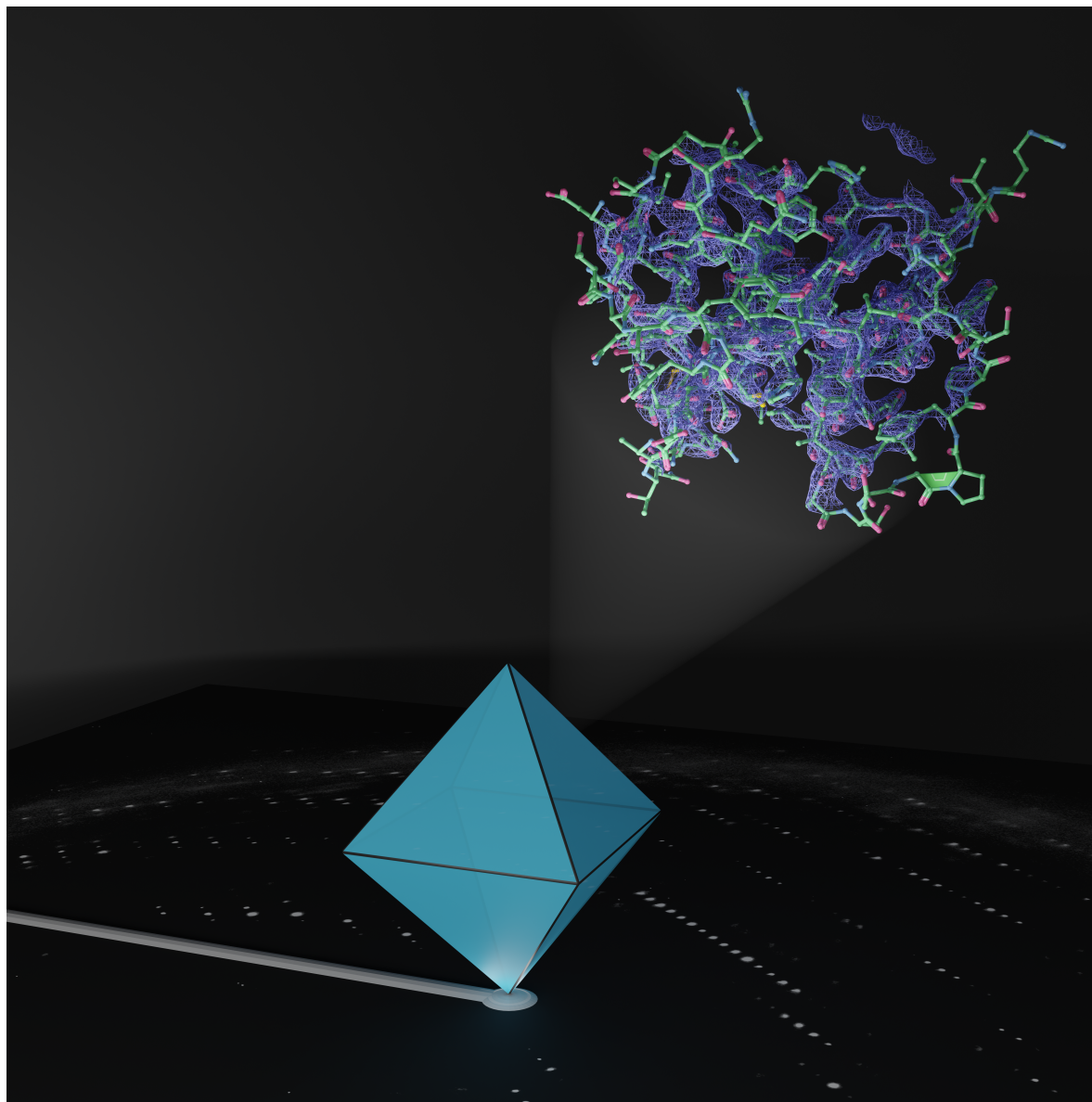
Sharpen/Blur Factor:

0.00

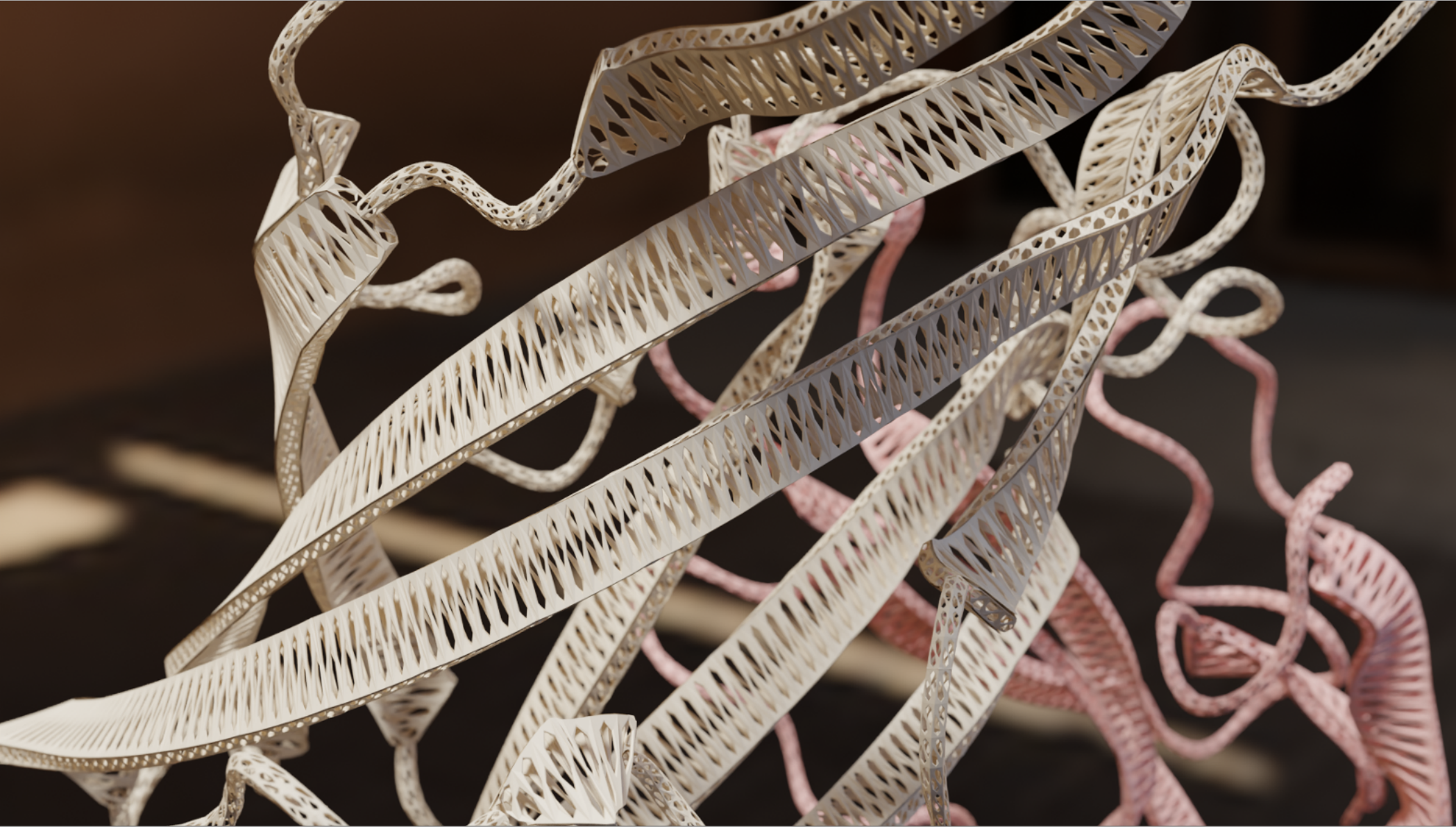


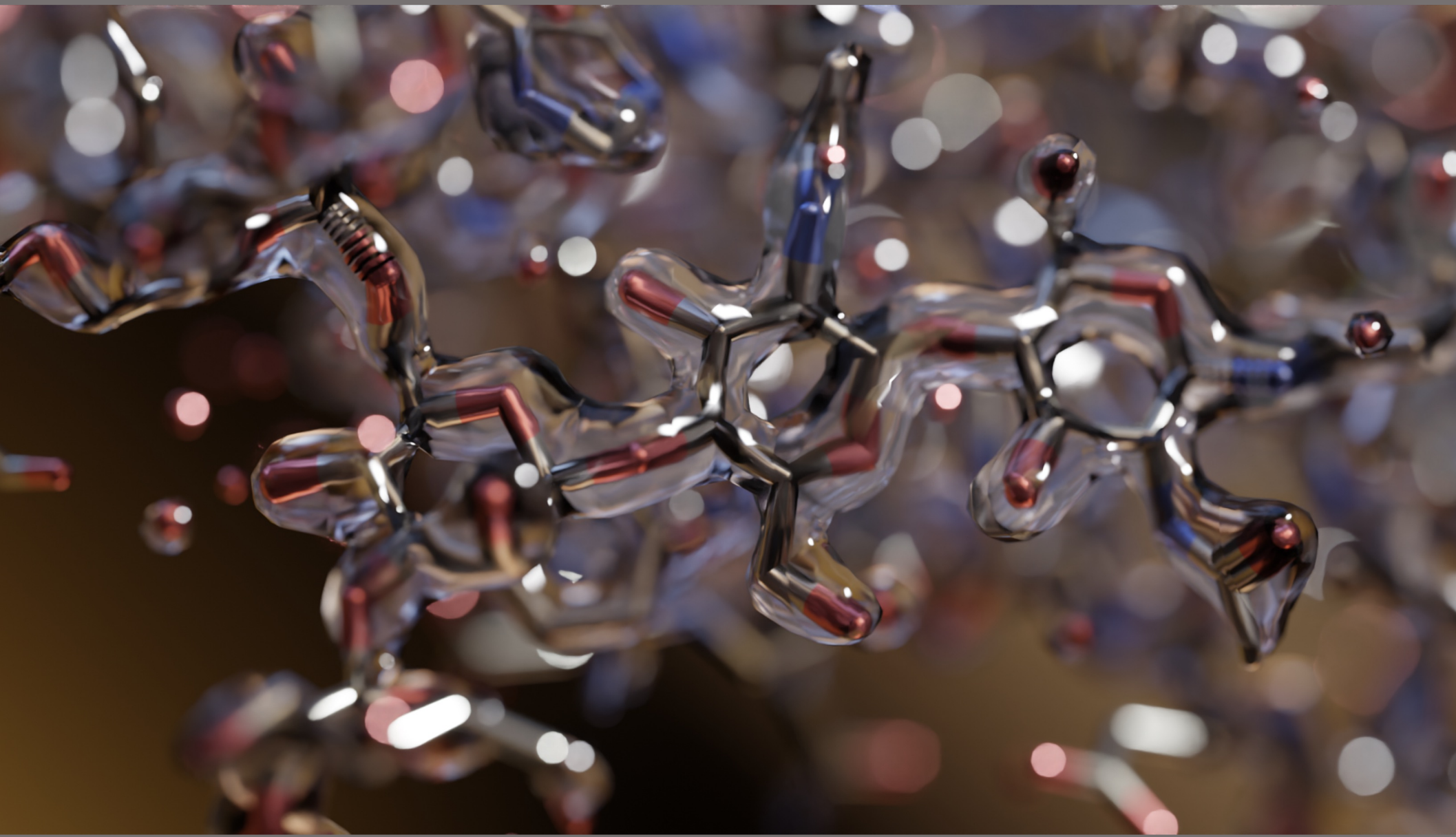
### The *CCP4* suite: integrative software for macromolecular crystallography

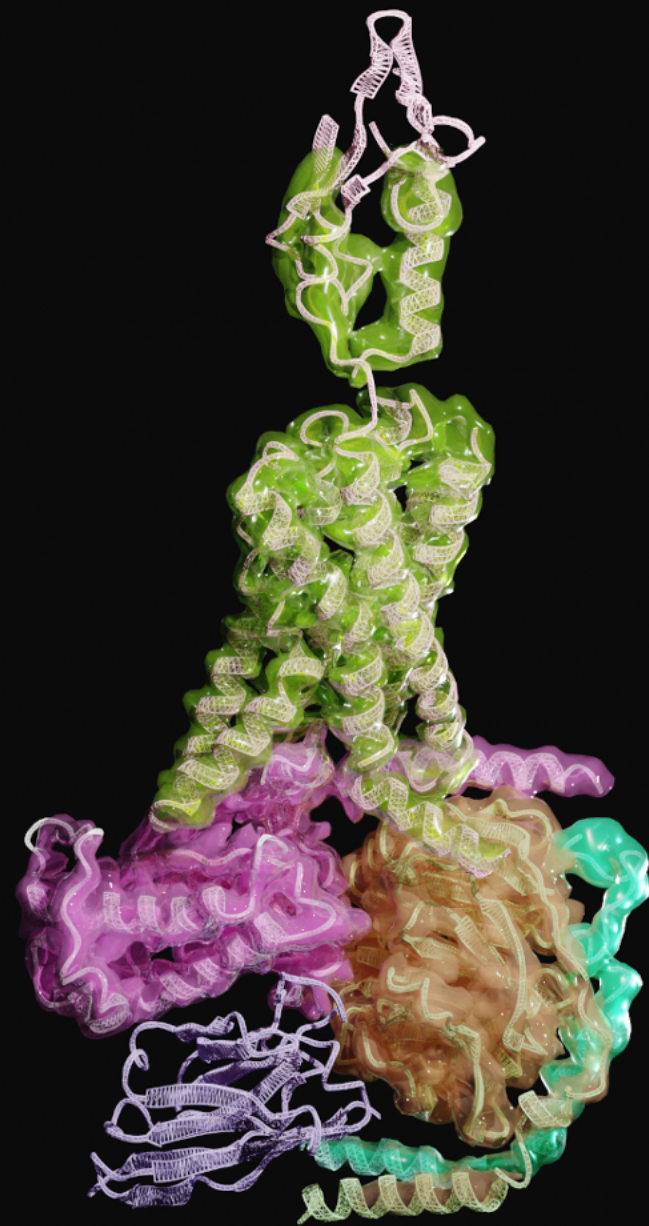
J. Agirre, M. Atanasova, H. Bagdonas, C. B. Ballard, A. Baslé, J. Beilsten-Edmands, R. J. Borges, D. G. Brown, J. J. Burgos-Mármol, J. M. Berrisford, P. S. Bond, I. Caballero, L. Catapano, G. Chojnowski, A. G. Cook, K. D. Cowtan, T. I. Croll, J. É. Debreczeni, N. E. Devenish, E. J. Dodson, T. R. Drevon, P. Emsley, G. Evans, P. R. Evans, M. Fando, J. Foadi, L. Fuentes-Montero, E. F. Garman, M. Gerstel, R. J. Gildea, K. Hatti, M. L. Hekkelman, P. Heuser, S. W. Hoh, M. A. Hough, H. T. Jenkins, E. Jiménez, R. P. Joosten, R. M. Keegan, N. Keep, E. B. Krissinel, P. Kolenko, O. Kovalevskiy, V. S. Lamzin, D. M. Lawson, A. A. Lebedev, A. G. W. Leslie, B. Lohkamp, F. Long, M. Malý, A. J. McCoy, S. J. McNicholas, A. Medina, C. Millán, J. W. Murray, G. N. Murshudov, R. A. Nicholls, M. E. M. Noble, R. Oeffner, N. S. Pannu, J. M. Parkhurst, N. Pearce, J. Pereira, A. Perrakis, H. R. Powell, R. J. Read, D. J. Rigden, W. Rochira, M. Sammito, F. Sánchez Rodríguez, G. M. Sheldrick, K. L. Shelley, F. Simkovic, A. J. Simpkin, P. Skubak, E. Sobolev, R. A. Steiner, K. Stevenson, I. Tews, J. M. H. Thomas, A. Thorn, J. T. Valls, V. Uski, I. Usón, A. Vagin, S. Velankar, M. Vollmar, H. Walden, D. Waterman, K. S. Wilson, M. D. Winn, G. Winter, M. Wojdyr and K. Yamashita



**Cover:**  
Acta Cryst D  
June 2023

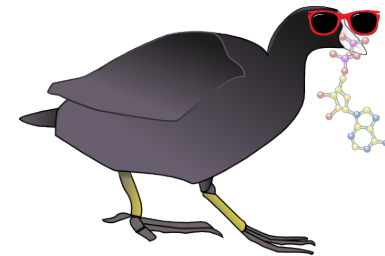
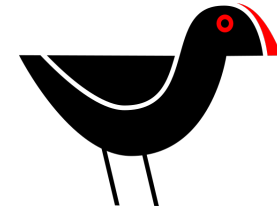






# Lunchtime Byte

- **Moorhen:** Today at 1pm
- **Coot 1.1:** Tomorrow at 1pm



# Acknowledgements

Filo  
Sanchez



Stuart  
McNicholas



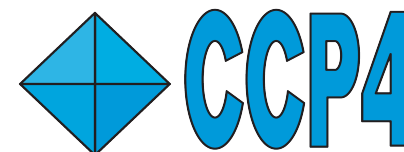
Paul  
Emsley



Jakub  
Smulski



Martin  
Noble



... And everyone who has contributed to CCP4

lucrezia@mrc-lmb.cam.ac.uk

 @lulu\_catapano