Jean-Yves Sgro, Ph.D.

# *PyMOL - Exploring the 3D PDB structures of SARS-CoV-2/COVID-19*



Dr. Jean-Yves Sgro

Department of Biochemistry

University of Wisconsin-Madison

425 Henry Mall

Madison WI 53706

USA

©2020 Jean-Yves Sgro

CC BY-NC 4.0

This work is licensed under a Creative Commons "Attribution-NonCommercial-ShareAlike 4.0 International"<sup>a</sup> license. CC () (S) (D) BY NC SA

<sup>*a*</sup>https://creativecommons.org/ licenses/by-nc-sa/4.0/deed.en <sup>a</sup>https://
creativecommons.
org/licenses/by-ncsa/4.0/deed.en

Cover: created by JYS with MeshLab from 3D design file by Alejandro León (@alelepd). See text for more details

# Contents

Lis	st of I	ables	vii		
Lis	st of F	igures	ix		
Pr	eface		xi		
Ab	out tl	he Author	xiii		
1	Intr	oduction	1		
2	SAR	S-Cov-2	5		
	2.1	PDB Structures	5		
	2.2	SARS-CoV-2 - Artistic renderings	6		
		2.2.1 Painting	6		
	2.3	3D models	7		
3	PyMOL Scripts: Short Primer				
	3.1	Scripts are "plain text"	11		
	3.2	Scripts formats	12		
	3.3	Activating a command script	12		
		3.3.1 Command script	13		
		3.3.2 Python script	13		
	3.4	PyMOL command vs API	13		
		3.4.1 PyMOL commands are API aliases	14		
	3.5	Shell-like commands and PATH	15		
4	SAR	S-CoV-2 Protease Mpro	17		

iii

iv	(	Contents	
	4.1 Change record type	19	
	4.2 Surface representation: still an issue	20	
	4.2.1 One solution: extract	22	
	4.2.2 Final solution	22	
	4.3 Biological assembly	25	
	4.4 Comparing structures	25	
5	Spike Protein	29	
6	NSP3	37	
	6.1 ADP ribose phosphatase of NSP3 from SARS CoV-2 .	37	
7	NSP15		
8	NSP12 polymerase		
9	Epilogue		
Ар	pendix	49	
A	SARS-CoV-2 PDB codes		
	A.1 Main protease	51	
	A.2 Papain-like protease	55	
	A.3 Spike protein	55	
	A.4 Other SARS-CoV-2 structures	56	
B	6LU7: PRD_002214 coordinates	59	
С	Scripts for figures in Chapter 4	61	
	C.1 Stick and surface	61	
	C.2 Biological assembly	63	
D	Scripts for figures in Chapter 5	65	
	D.1 Display content of 6LZG	65	
	D.2 Display content of trimeric structures 6VSB and 6VXX	66	

|\_\_\_\_

|

\_\_\_\_

Соп	itents		v
	D.3	Superimpose 6LZG and trimeric 6VSB	67
	D.4	6ACK Spike glycoprotein trimer with bound ACE2	68
Ε	E Scripts for figures in chapter 6		
	E.1	Nsp3 - ribose ADP	71
F	Scripts for figures in chapter 7		73
	F.1	Nsp15 - asymetric unit and biological assembly	73
G	Scripts for figures in chapter 8		77
	G.1	SARS-CoV-2 nsp12 polymerase bound to nsp7 and	
		nsp8 co-factors	77
Bil	Bibliography		
Inc	lex		83

|\_\_\_\_ | \_\_\_\_

# List of Tables

5.1	Spike protein PDB entries	29
6.1	PDB entries for ADP ribose phosphatase portion of	
	NSP3	37
A.1	PDB IDs of SARS-CoV-2 main protease	52
A.2	PDB IDs of SARS-CoV-2 Papain-like protease	55
A.3	PDB IDs of SARS-CoV-2 spike glycoprotein	56
A.4	PDB IDs of other SARS-CoV-2	56

vii

|\_\_\_\_ | \_\_\_\_

# List of Figures

2.1	SARS-CoV-2 painting and key, Pr. David Goodsell.	6
2.2	Details of coronavirus3d file shown on spline.design	
	showing artisitic rendering of surface trimeric spike	
	protein	8
2.3	Stereo rendering of coronavirus3d file in Photoshop .	8
2.4	MeshLab rendering of coronavirus3d file	9
4.1	PRD_002214, cyan, appears broken into three parts	18
4.2	PRD_002214, sticks, CPK colors	20
4.3	PRD_002214 middle portion ALA VAL LEU is included	
	in surface	21
4.4	PRD_002214, extracted, in protein groove	22
4.5	PRD_002214, partial surface	23
4.6	No surface created for PRD_002214 when reloaded	24
4.7	6LU7 biological assembly. PRD_002214 shown as	
	spheres or sticks in each monomer	26
5.1	6LZG: ACE2 and binding domain. NAG in red	30
5.2	6VXX (closed) and 6VSB (1 up) trimeric structure. Out-	
	side view.	31
5.3	6VXX (closed) and 6VSB (1 up) trimeric structure. Side	
	view	32
5.4	6LZG (ACE2 and binding domain) superimposed onto	
	6VSB receptor-binding up domain	33
5.5	6ACK: ACE2 bound to spike glycoprotein trimer	34

List	0	Figures

5.6	6ACK: contact amino acids from spike glycoprotein trimer (chain C) with AC2 (surface.)	35
6.1	6W02 aligned with 5HOL.	38
6.2	Close view of ADP within 6W02	39
7.1	6W01 asymetric unit.	42
7.2	6W01 biological assembly: 2 trimers	42
7.3	6W01 biological assembly: all chains colored	43
8.1	6M71 RNA-dependent RNA polymerase nsp12 in com-	
	plex with cofactors nsp7 and nsp8	46
8.2	Comparison between Cov (6NUR) and Cov2 (6M71) RNA-dependent RNA polymerase nsp12 in complex	
	with cofactors nsp7 and nsp8	47
9.1	SARS-CoV-2 3D model for crayon coloring	50

х

# Preface

This booklet is meant for those that are already familiar with PyMOL [Schrödinger, LLC, 2020] as a molecular graphics software for macro-molecular biological structures such as proteins and nucleic acids.

If you are novice you may benefit from finding the book "*Exploring Protein Structure: Principles and Practice*" [Skern, 2018]. If you are at the UW-Madison the information on my blog post<sup>1</sup> will guide you to download this book from the library.

You can also learn how to use PyMOL from my course material now availabe online [Sgro, Jean-Yves, 2017].

# Software information and conventions

I used the **knitr** package [Xie, 2015], the **bookdown** package [Xie, 2016] enhanced by the **bookdownplus** package to compile this book within the **RStudio** software [RStudio Team, 2015] in various formats (PDF, HTML etc.)

PyMOL figures were created *on the fly* by scripts run from within the **RStudio** framework every time the book was compiled. This was possible on a Macintosh by calling the internal PyMOL engine with/Applications/PyMOL.app/Contents/MacOS/PyMOL and providing a script.

<sup>&</sup>lt;sup>1</sup>http://tiny.cc/pybook

Preface

On a Linux system the command would simply be pymol if the software is installed and within the PATH environment. On Windows I am not sure how this would or could work.

# Acknowledgments

"Summary" icon for special text boxes is from https://www.flaticon.com/free-icon/recorder\_2450952

xii

# About the Author



Jean-Yves Sgro, a senior scientist with years of experience in using and teaching computer programs, creates, organizes and teaches hands on workshops.

Jean-Yves has been at UW since 1986 after a Master in Physiology and a Ph.D. in Cellular and Molecular Biology from Joseph Fourier University<sup>2</sup>, Grenoble, France, and researched at the European Molecular Biology Laboratory (EMBL<sup>3</sup>) where he already used large computers for sequence analysis.

In Madison, at the Institute for Molecular Virology (IMV<sup>4</sup>) he continued developing computer expertise in addition to his wet-lab research – 3D molecular visualization (virusworld<sup>5</sup>), RNA-folding predictions, sequence and data analysis...

In 1996 he joined the UW Biotechnology Center<sup>6</sup> to better help Campus biologists analyze and visualize their data while continuing research at IMV until 2014 when this part-time position was transferred to the

<sup>&</sup>lt;sup>2</sup>https://www.univ-grenoble-alpes.fr/english/

<sup>&</sup>lt;sup>3</sup>https://www.embl.fr/

<sup>&</sup>lt;sup>4</sup>http://virology.wisc.edu/

<sup>&</sup>lt;sup>5</sup>http://www.virology.wisc.edu/virusworld

<sup>&</sup>lt;sup>6</sup>https://www.biotech.wisc.edu/

Biochemistry Department<sup>7</sup> where he organizes and teaches hands-on tutorials on molecular graphics, data analysis as a support to the department personnel.

Tutorials are available on line from the The Biochemistry Computational Research Facility (BCRF<sup>8</sup>.)



xiv

<sup>&</sup>lt;sup>7</sup>https://biochem.wisc.edu/ <sup>8</sup>https://bcrf.biochem.wisc.edu/

# Introduction

We are in the midst of the COVID-19 pandemic, caused by the virus named SARS-CoV-2<sup>1</sup>

This small booklet is an attempt to illustrate some of the SARS-CoV-2 know structures available as 3D atomic coordinates at the Protein Data Bank (PDB) with the PyMOL software.



# How to use this booklet:

The original purpose was to provide a starting point to visualize known 3D strutures from the SARS-CoV-2 virus using the PyMOL molecular visualisation software.

All figures were created by script, and all complete scripts are availabe in the Appendix sections C, D, E,F,G.

Script lines can be copied one-by-one and pasted within the PyMOL command line.

Alternatively, the entire script can be copied within a *plain text* file and saved on the computer, then activated with a command such as: @myscript.txt from within the PyMOL command line area. (See section 3.3.1.)

<sup>&</sup>lt;sup>1</sup>SARS: Severe Acute Respiratory Syndrome-related coronavirus

note

If your interest is to learn more PyMOL commands, I recommend to follow the scripts one line at a time:

Watch the result of each command (some results might not be visible immediately.)

Inquire about the command structure:

-. Use PyMOL help followed by command name for immediate info.

-. Find same info in https://pymol.org/pymol-command-ref. html

- -. Search within https://pymolwiki.org/
- -. Be curious with a search engine (e.g. Google)

I am creating this booklet with the principles of "Reproducible Research" in the sense that this very document can be recreated in its entirety, including the recreation of all PyMOL illustrations as they can be redrawn "on the fly" thanks to the use of the scripts that create them.

I am using *PyMOL 2.3.5 Incentive Product* but this version number would also be automatically adjusted should I update my PyMOL program and recompile the book.

I had started a different booklet and changed course when the pandemic started. I levied Chapter 4 from that prior (unfinished) project which explains its different style with some suggested exercises and more of the script written within the text. There were some "strange" difficulty with the ligand of the structure studied in this chapter which started the idea of these booklets.

The original format is **PDF** but other formats might be made available,

1.0

albeit there will be some incompleteness regarding cross-references, index, or even numbering of chapters etc.

|\_\_\_\_ | \_\_\_\_

# SARS-Cov-2

## 2.1 PDB Structures

Some SARS-Cov-2 viral proteins have been solved, mostly by X-ray crystallography, as of early April 2020:

- Main protease: 83 structures with various ligands (76 of which from the *PanDDA analysis group*.)
- Spike protein: 5 structures
- Papain-like SARS-CoV-2 structures: 1 structure
- Other SARS-CoV-2 structures (binding domains etc.): 17 structures

Tables with the PDB accession code and struture titles are available in Appendix A with information for the data source.

PDB structure 6LU7 is the first high-resolution crystal structure of SARS-CoV-2 main protease (released February 5, 2020) and is the object of study of chapter 4.

This booklet intention is to help with the rendering of the PDB files in PyMOL and provide all scripts to create PyMOL illustrations.

However, the complete virus particle has not been solved but various artistic "renderings" are available as discussed briefly in the next section.

## 2.2 SARS-CoV-2 - Artistic renderings

### 2.2.1 Painting

Scripps Prof. David Goodsell<sup>1</sup><sup>2</sup>, famous for his non-photorealistic illustrations of molecules, has created a beautiful rendering of the coronavirus entering the lung as a painting displayed on the Protein Data Bank web site<sup>3</sup>.



FIGURE 2.1: SARS-CoV-2 painting and key, Pr. David Goodsell.

This painting depicts a coronavirus just entering the lungs, surrounded by mucus secreted by respiratory cells, secreted antibodies, and several small immune systems proteins. The virus is enclosed by a membrane that includes the S (spike) protein, which will mediate attachment and entry into cells, M (membrane) protein, which is involved in organization of the nucleoprotein inside, and E (envelope) protein, which is a membrane channel involved in budding of the virus and may be incorporated into the virion during that process. The nu-

<sup>3</sup>https://pdb101.rcsb.org/sci-art/goodsell-gallery/coronavirus

<sup>&</sup>lt;sup>1</sup>https://ccsb.scripps.edu/goodsell/

<sup>&</sup>lt;sup>2</sup>Twitter: @dsgoodsell

#### 2.3 3D models

cleoprotein inside includes many copies of the N (nucleocapsid) protein bound to the genomic RNA.

## 2.3 3D models

Other artisitc renderings have been done by computer artists. The 3D file formats used are different than the format used for PDB files and cannot readily be opened with PyMOL. Many of them are for purchase on web sites.

Author Alejandro León<sup>4 5</sup> created a free version that can be accessed already rendered within a web site<sup>6</sup> and that can be manipulated in 3D (See figure 2.2.)

A 12 seconds video of what this looks like is available on YouTube<sup>7</sup>.

According to the author this 3d model of Coronavirus (SARS-CoV-2) based on the images from CDC/PHIL. The model is available to down-load for free (MIT License)<sup>8</sup>.

In addition the 3D model file can be downloaded<sup>9</sup> as a compressed binary in the .fbx format that is typically used to allow transfer to and from various 3D illustration software.

A web site<sup>10</sup> can convert this file in other 3D formats<sup>11</sup>. Noteworthy is the fact that the most recent Photoshop version can open the .obj for-

```
<sup>4</sup>Twitter:@alelepd
<sup>5</sup>https://alejandro.fun/
<sup>6</sup>https://spline.design/coronavirus3d/
<sup>7</sup>https://youtu.be/KOORIaAuXb4
<sup>8</sup>https://opensource.org/licenses/MIT
<sup>9</sup>https://spline.design/coronavirus3d/_assets/coronavirus3d.
fbx
<sup>10</sup>https://products.aspose.app/3d/conversion/
```

<sup>&</sup>lt;sup>11</sup>Available formats: FBX, OBJ, 3DS, DRC, AMF, RVM, DAE, GLTF, GLB, PDF, HTML, PLY, STL, U3D

2 SARS-Cov-2



**FIGURE 2.2:** Details of coronavirus3d file shown on spline.design showing artisitic rendering of surface trimeric spike protein.

mat into a 3D layer where the various "layers" can be assigned a color or a texture (figure 2.3.)

Here is a stereo version created with this file with Photoshop:



FIGURE 2.3: Stereo rendering of coronavirus3d file in Photoshop

The coronavirus3d (*e.g.* in . obj format) can also be rendered with the free 3D software MeshLab<sup>12</sup> available for Windows/Mac/Linux. Within this software all "layers" are given the same color, but vertices can be

<sup>&</sup>lt;sup>12</sup>http://www.meshlab.net/

2.3 3D models

colored which can create "artistically nice" renderings as well (figure 2.4.)



FIGURE 2.4: MeshLab rendering of coronavirus3d file.

|\_\_\_\_ | \_\_\_\_

# PyMOL Scripts: Short Primer



3

# SUMMARY:

- Scripts are plain text
- PyMOL command ccripts are activated with @
- PyMOL python form scripts are activated with run

## 3.1 Scripts are "plain text"

A *plain text* file will:

- only contain printable characters that can be typed from the simplest of keyboards: letters, numbers and symbols such as !, @ and \$.
- is not specific to any particular word processor format and is displayed with the default font within any word processor.
- contain a "line-break" or "end-of-line" code (specific to the operating system, which can be an inconvenience sometimes.)

Software that can easily create a *plain text* file:

Windows: Notepad or one of many suggested replacements<sup>1</sup>

<sup>&</sup>lt;sup>1</sup>https://alternativeto.net/software/notepad/?platform=windows

3 PyMOL Scripts: Short Primer

- MacOS: TextEdit or one of the many suggested replacements<sup>2</sup>,<sup>3</sup>
- Linux: nano, vim, emacs or some other replacement<sup>4</sup>
- Cross-platform: Atom<sup>5</sup>

# 3.2 Scripts formats

There are different PyMOL scripts formats:

• .pml: Scripts containing lists of PyMOL commands

• .py:

- Scripts comtaining lists of PyMOL command in Python form (See section 3.4.)
- Scripts containing new functions, written in Python language.

*Note*: When activated through the command line, a file ending in .txt works as well.

#### 3.3 Activating a command script

The activation of scripts may different depending on their nature if not activated by mouse menu.

With a mouse the following menu cascade can be used to activate any script form: **File > Run Script...** which then offers the choices:

- All Runnable (\*.pml \*.py \*.pym)
- PyMOL Command Script (\*.pml)

<sup>&</sup>lt;sup>2</sup>https://alternativeto.net/software/textedit/?platform=mac <sup>3</sup>"use menu: Format > Make plain text"

<sup>&</sup>lt;sup>4</sup>https://alternativeto.net/software/nano/?platform=linux <sup>5</sup>https://atom.io/

- 3.4 PyMOL command vs API
  - PyMOL Command Script (\*.txt)
  - Python Script (\*.py \*.pym)
  - Python Script (\*.txt)
  - All Files (\*)

Note: in older versions of PyMOL a command file would need to end with .pml in order to be used with the mouse menu.

#### 3.3.1 Command script

A PyMOL Command Script can be activated with the command line:

PyMOL>@myscript.pml

This assumes that the script file is located within the current directory.

## 3.3.2 Python script

A Python language script should use the PyMOL command run to be activated, even if @ may work in most cases when the script is only a series of commands. PyMOL will provide a warning.

```
PyMOL>@myscript.py
Warning: use 'run' instead of '@' with Python files?
```

Therefore the command sould be given as:

PyMOL>run myscript.py

This is particilar true for files that provide new functions to PyMOL as we'll see later.

### 3.4 PyMOL command vs API

PyMOL is a molecular graphics software based on the Python language (hence its name.) This document is using *PyMOL 2.3.5 Incentive Product* 

to create graphics automatically with scripts. (Scripts are located in the Appendix.)

#### 3.4.1 PyMOL commands are API aliases

A PyMOL command has an equiallent in API<sup>6</sup> form (written in python) and there are also less documented API commands that do not exist in PyMOL command form.

This is best understood with a simple example, let's take the PyMOL command color. I can issue commands such as:

```
PyMOL>color blue
```

I can also specify the chain I'dlike to color if there are more than one:

```
PyMOL>color blue, chain A
```

The command color<sup>7</sup> is defined as:

```
color color [, selection ]
```

- The first word color is the command name
- The second color is the mandatory argument and should be a color name or number
- the selection is everything unless specified

But behind the scenes, PyMOL is running python language functions, and the command color is in fact defined with all its parameters<sup>8</sup> as:

cmd.color(string color, string selection, int quiet)

This can also be used directly within PyMOL, for example

<sup>&</sup>lt;sup>6</sup>Application Programming Interface

<sup>&</sup>lt;sup>7</sup>https://pymol.org/dokuwiki/doku.php?id=command:color

<sup>&</sup>lt;sup>8</sup>https://pymol.org/dokuwiki/doku.php?id=api:cmd:color

3.5 Shell-like commands and PATH

```
PyMOL>cmd.color("blue")
```

Of course a chain name can also be specified:

```
PyMOL>cmd.color("blue", "chain A")
```

It is important that the arguments be placed within quotes or an error will occur. This is because within the definition the arguments are expeted to be *string* which means of *character* type (as opposed to a number for example.)

My opinionated conclusion is that the command form appears more concise to the eyes (less need for quotes or parenthesis) and therefore more "English-like" which in turn allows a better comprehension, or understanding of the tasks that PyMOL is asked to perform, such as changing a color.

# 3.5 Shell-like commands and PATH

One of the trickiest things about computer is all the assumptions behind **where things are** and what does the computer knows anbout where things are.

By default PyMOL will "look" within the home user directory. On modern desktop computers it is usually within the /Users directory. For example for me it would be /Users/jsgro.

If a script is not within the current directory, a *path* to the script can be given, either as absolute or relative path with the appropriate preceding @ symbol or run command. For example @../pyscripts/myscript.pml. Most systems would also understand the ~ shortcut representing the home directory.

PyMOL also provides three commands identical to the shell commands<sup>9</sup>.

- cd: Change the current working directory
- 1s: List contents of the current working directory.
- pwd: Print current working directory.

In addition the command system executes a command in a subshell under Unix or Windows (not Mac.)

Therefore it is possible to change directory (cd,) check in which directory PyMOL is looking (pwd) and list current files in that directory (ls.)

The cd command works in the same way as in a Unix/Linux shell. BY default PyMOL will look within the home directory as detailed above. The standard shell commands apply, including

For example:

- cd MyDirName
- cd Desktop
- cd ~/Desktop
- cd ~/MyDirName2
- cd ../

The ls command includes usage of wild card \* and regular expressions in the search (*e.g.* ls \*[t].py, ls cla??29\*.py) and *tab-extension* is operational.

<sup>&</sup>lt;sup>9</sup>https://pymol.org/pymol-command-ref.html

# SARS-CoV-2 Protease Mpro



4

## SUMMARY:

Unexpected problems with a seemingly simple rendering of PDB coordinates of SARS-CoV-2 main proteinase Mpro.

This is an example of PyMOL Command Scripting to create figures that can automatically be generated from a script. The final script(s) can be found in Appendix C.

At this writing the Covid-19 pandemic caused by the 2019 novel coronavirus (SARS-CoV-2) is going on. The RNA genome is 82% identical to the SARS coronavirus (SARS-CoV) and has been named SARS-CoV-2 (Gorbalenya et al. [2020].) The disease caused by SARS-CoV-2 is called COVID-19.

The structure of the virus protease has been solved by X-ray diffraction (Jin et al. [2020]). The structure was deposited 2020-01-26 and released 2020-02-05 with PDB code 6LU7<sup>1</sup> with title *The crystal structure* of COVID-19 main protease in complex with an inhibitor N3. The biological assembly is a dimer.

The peptide-inhibitor is logdged within a groove on the protease surface. Its name is *PRD\_002214<sup>2</sup>* in the *Biologically Interesting Molecule Reference Dictionary* (BIRD).

<sup>&</sup>lt;sup>1</sup>http://dx.doi.org/10.2210/pdb6LU7/pdb

<sup>&</sup>lt;sup>2</sup>https://www.rcsb.org/bird/PRD\_002214

4 SARS-CoV-2 Protease Mpro

Within the PDB file it has sequence:

SEQRES 1 C 6 02J ALA VAL LEU PJE 010

When PyMOL (2.x) opens the file, showing the structure with the new default of cartoon, the protein is fine but zooming on the PRD\_002214 it appears broken in 3 parts (see figure 4.1.)



FIGURE 4.1: PRD\_002214, cyan, appears broken into three parts.

The cause is the interpretation of the 3D coordinates for PRD\_002214 in chain C. Closer inspection of the coordinates shows that its peptidelike nature causes its coordinates to be a mixture of ATOM and HETATM records (see Appendix B.)

Thanks to PyMOL command options we'll be able to modify the coordinates so that compound PRD\_002214 may be represented as full in *stick* visual format.

# 4.1 Change record type

Even though chain C does not contain any termination TER records the result of the mixed atom records as ATOM and HETATM causes PyMOL to split the representation in 3 parts. One solution would be to change all records to HETATM which can be accomplished with PyMOL command alter<sup>3</sup>.

The script below will accomplish this task:

```
fetch 6lu7, async=0
as cartoon
# Change ATOM records
alter (chain C), type="HETATM"
show stick, chain C
hide cartoon, chain C
util.cbaw chain C
zoom 6lu7 and chain C
turn y, -50
turn z, -90
turn x, -30
```

This should now produce a complete stick representation of PRD\_002214 compound.

## Exercise:

In the script we used alter (chain C), type="HETATM". You can also try alter (seg B), type="HETATM".

<sup>&</sup>lt;sup>3</sup>https://pymol.org/pymol-command-ref.html#alter

4 SARS-CoV-2 Protease Mpro



FIGURE 4.2: PRD\_002214, sticks, CPK colors.

Does it work for both asymetric unit and biological assembly? Why not?

# Exercise:

Try to reproduce a similar image for the biological assembly (type=pdb1.)

Does the rotation translation provided above for the asymetric unit work as well?

Why or why not?

# 4.2 Surface representation: still an issue

If we now would like to use the standard surface representation that constructs a surface for the protein but not on the organic ligands, there will sill be a problem of representation, this time due to the fact

#### 4.2 Surface representation: still an issue

that the name of the residues that were originally presented as ATOM within the compound have natural amino acid names, and therefore, in spite of now being labeled HETATM will be included in the surface.

Residue names for the compound are indeed 02J ALA VAL LEU PJE 010 (see above) and the portion ALA VAL LEU will be selected for surface construction.



**FIGURE 4.3:** PRD\_002214 middle portion ALA VAL LEU is included in surface.

One may think that a possible solution would simply to change the name of those amino acid to a generic UNK for unknown compound.

# Exercise:

Test the hypothesis that changing the amino acid names within PRD\_002214 to an "unknown" does (or does not) change the surface options.

Hint: alter resn ala+val+leu and chain C, resn =
"UNK"

4 SARS-CoV-2 Protease Mpro

# 4.2.1 One solution: extract

One solution that works consists in extracting the ligand to a different PyMOL object (not a selection) so that the surface construction of the ramining structure will not include any of the ligand itself.

Assuming we already have the structure within PyMOL, this can be accomplished by the few commands below that:

- 1. Create a selection name for chain C as "C"
- 2. Extract the selection into a new object "PRD"
- 3. Show again surface for the rest of 61u7

select C, chain C extract PRD, C show surface, 61u7



**FIGURE 4.4:** PRD\_002214, extracted, in protein groove.

## 4.2.2 Final solution

At this point we could think that we are done. But we are not!
4.2 Surface representation: still an issue

If we now try to create a surface for the new, independent "PRD" object we still obtain a partial surface of the molecule in the area of the (now) UNK residues.

For example with the command show surface, PRD.



FIGURE 4.5: PRD\_002214, partial surface.

The likely reason is that while we changed actual amino acid names from ALA, VAL, LEU to unknow UNK, PyMOL probably remembers the *properties* of these residues from *before they were changed*. Hence the surface creation.

To erase this residual memory of properties, one solution is the write the coordinates of PRD into a file, delete the object and re-read the coordinates within PyMOL.

Let's try if this will work with the few commands that follow.

```
save PRD.pdb, PRD
delete PRD
load PRD.pdb
hide cartoon, PRD
```

#### 4 SARS-CoV-2 Protease Mpro

util.cbaw chain C show surface, PRD



FIGURE 4.6: No surface created for PRD\_002214 when reloaded.

## Exercice:

Why is it necessary to have command hide cartoon, PRD? What is the mouse menu equivalent to the save command? Where is the mouse equivalent of the delete command?

Finally, the following settings may help when saving the coordinates into a file that will:

- Create CONECT records specifying which atoms are bonded
- Prevent the addition of TER records which could create gaps within graphics representations.

These were not necessary for the creation of the embedded illustration here but could be useful in other situations.

### 4.4 Biological assembly

See https://pymolwiki.org/ for more information on these commands

### Exercice:

Where is the file PRD.pdb saved? Could the file be saved in a different format? Does it contain TER records? Are there UNK residues present?

### 4.3 Biological assembly

The biologically active assembly is a dimer as shown in figure 4.7. See script in Appendix 4.

#### 4.4 Comparing structures

Beyond scripting: if you are interested in the science of the protease the following papers would be of interest: Adem et al. [2020], Zhang et al. [2020] reporting *in silico* search for inhibitorts of the protease we have used in this chapter (Jin et al. [2020], PDB 6LU7.) This structure is very similar to that of the SARS coronavirus protease reported by Tan

4 SARS-CoV-2 Protease Mpro



**FIGURE 4.7:** 6LU7 biological assembly. PRD\_002214 shown as spheres or sticks in each monomer.

et al. [2005], (PDB 2BX4, no ligand,) and the new protease of SARS-CoV-2 has also been solved in *apo* form, without ligand (PDB: 6M03.)

Zhang et al. [2020] provide a good summary description of the protease function and cleavage specificity:

"[...] this enzyme is essential for processing the polyproteins that are translated from the viral RNA (Hilgenfeld [2014]). The Mpro operates at no less than 11 cleavage sites on the large polyprotein 1ab (replicase 1ab, ~790 kDa); the recognition sequence at most sites is Leu-Gln4 (Ser, Ala, Gly) (+ marks the cleavage site). Inhibiting the activity of this enzyme would block viral replication. Since no human proteases with a similar cleavage specificity are known, inhibitors are unlikely to be toxic."

These authors have also released the SARS-CoV-2 Mpro protease struc-

#### 4.4 Comparing structures

tures with and without bound ligands with the following PDB codes not cited within the paper:

- 6Y2E, no ligand
- 6Y2F and 6Y2G with complex ligand 06K

In the paper they state that the r.m.s. deviation between the two freeenzyme structures is 0.53 Å for all C $\alpha$  positions (comparison between SARS-CoV-2 Mpro structure and SARS-CoV Mpro, PDB entry 2BX4.

How did they calculate this number?

My own search for calculating root-mean-square deviation (RMSD) *with PyMOL* pointed to many commands. However I believe that the following method was used as it provide the same answer. It is from the pymolwiki of PyMOL command super that aligns two selection with a sequence-independent structure-based dynamic programming alignment followed by a series of refinement cycles intended to improve the fit.<sup>4</sup>

The following script is taken from that page with only the PDB codes modified to those of the protease without ligand.

```
fetch 6Y2E, async=0
fetch 2BX4, async=0
# super 6Y2E, 2BX4
test=cmd.super("6Y2E","2BX4")
```

reinitialize

<sup>&</sup>lt;sup>4</sup>https://pymolwiki.org/index.php/Super

4 SARS-CoV-2 Protease Mpro

```
python
writefile=open("rmsd_file_super.txt","a")
writefile.write(' '.join('%s' % x for x in test))
writefile.write('\n')
writefile.close()
python end
```

File rmsd\_file\_super.txt contains only one line:

 $0.5308237075805664\ 1928\ 5\ 1.130696415901184\ 2280\ 1361.6253662109375\ 303$ 

The first value is indeed 0.53.

One "trick" was to use the cmd.super() python form which allows the recoding of the data into a variable called test. The rest of the code is python for writing the content of the variable test within the rmsd\_file\_super.txt file.

### Optional exercise:

Create a figure representing both structures superimposed and shown as ribbon or as cartoon.

Then create a new figure showing them side-by-side in the exact same orientation.

Hint:grid\_mode

# Spike Protein

(Find all image scripts for this chapter in Appendix D.)

The spike (S) protein is a surface trimer glycoprotein located on the envelope of the virions. It is cleaved into S1 and S2 subunits. The S1 subunit is responsible for host-receptor binding while the S2 subunit contains the membrane-fusion machinery<sup>1</sup>.

There are 5 published structures as shown in the following table:

## **TABLE 5.1:** Spike protein PDB entries

ID	Title
6LZG	Structure of novel coronavirus spike receptor-binding
	domain complexed with its receptor ACE2
6MoJ	Crystal structure of 2019-nCoV spike receptor-binding
	domain bound with ACE2
6VXX	Structure of the SARS-CoV-2 spike glycoprotein $closed state$
6VYB	SARS-CoV-2 spike ectodomain structure $openstate$
6VSB	Prefusion 2019-nCoV spike glycoprotein with a single
	receptor-binding domain up

The first two 6M0J and 6LZG are both showing a spike receptor region (S1) bound to the Angiotensin-converting enzyme 2 (ACE2 ) receptor.

<sup>&</sup>lt;sup>1</sup>https://www.rekombiotech.com/en/blog/detection-covid-19

#### 5 Spike Protein

Structures are almost indistinguishabel with an RMDS of 0 . 156  $^{\rm 2}$  are almost identical.

So choosing just one of them *e.g.* 6LZG is enough. We can illustrate the content of this PDB file showing how ACE2 and S1 fragment are bound. Both proteins have *N*-Acetylglucosamine<sup>3</sup> labeled NAG within the PDB file and bound to ASN residues (shown in red in figure 5.1.)



FIGURE 5.1: 6LZG: ACE2 and binding domain. NAG in red.

The other three structures are different configuration of the trimeric structure. Figures 5.2 and 5.3 show a side by side comparison between the closed structure (6VXX) and the trimeric structure of the spike gly-coprotein with a single receptor-binding domain up 6VSB.

The figures are automatically colored by chain. In 5.1 structures are shown in the original orientation when the PDB is opened, looking into the trimeric structure as from the outside.

<sup>&</sup>lt;sup>2</sup>see previous chapter 4.4 for RMSD calculation method

<sup>&</sup>lt;sup>3</sup>https://en.wikipedia.org/wiki/N-Acetylglucosamine



31

**FIGURE 5.2:** 6VXX (closed) and 6VSB (1 up) trimeric structure. Outside view.

To obtain figure 5.3 a simple rotation of 90 degrees around the x axis is sufficient. It then becomes clearer to spot the "1 up" fragment (green) from structure 6VSB. This is the equivallent portion labeled "*receptor-binding domain*" of 6LZG displayed in figure 5.1.

*Note*: removed in these figures are the numerous glycosylation sites that would be shown by default as "floating" sticks.)

We'll use 6VSB trimeric structure of the spike glycoprotein with a single receptor-binding domain up. This domain corresponds to the spike receptor region (S1) of the 6M0J and 6LZG structures.

As an exercise we'll superimpose 6LZG over 6VSB to see how the receptor would look like and be positioned in relation to the trimeric structure. Color scheme is as in figure 5.1 for 6LZG and figures 5.2 and 5.3 for 6VSB.

The scripts can be found in Appendix D.

Within the script the structures are superimposed with the PyMOL

5.0





**FIGURE 5.3:** 6VXX (closed) and 6VSB (1 up) trimeric structure. Side view.

command super that is not relying on sequence. However, in this case the command align would work equally well.

To orient the molecules the script uses the PyMOL function set\_view for which the data can be obtained when using the graphical interface by pressing the button "Get View" at the top right or on the commandline with the PyMOL command get\_view. This method is an alternative to PyMOL commands turn and move used in other scripts, for simplicity.

After all this work we can now compare this "modeled" structure combination with a crystal structure of PDB file 6ACK with title: "Trypsincleaved and low pH-treated SARS-CoV spike glycoprotein and ACE2 complex, ACE2-bound conformation 3".

This structures contains the trimeric spike glycoprotein trimer, one of which is attached to a molecule of *angiotensin-converting enzyme 2* (ACE2) in a similar way as our model above.



**FIGURE 5.4:** 6LZG (ACE2 and binding domain) superimposed onto 6VSB receptor-binding up domain.

We can select residues that are near ACE2 and show them as stick models and zoom for a more closer look (figure 5.6.)

Let's review some of the critical or new PyMOL commands from the scripts for this chapter (Appendix D.)

- align and super are two methods to superimpose structures. Unlike align, super does not rely on sequence information.
- extract is a useful command to separate a set of atoms that were previously selected. One of the effect is to allow a surface to be complete and independent of other molecules.
- select byres chain C within 5 of ACE2 creates a selection based on distance with within
- util.cbc is a simple utility meaning color by chain
- util.cbaw is also a simple utility meaning color by atom white

5.0

5 Spike Protein



**FIGURE 5.5:** 6ACK: ACE2 bound to spike glycoprotein trimer.



**FIGURE 5.6:** 6ACK: contact amino acids from spike glycoprotein trimer (chain C) with AC2 (surface.)

5.0

|\_\_\_\_ | \_\_\_\_

## NSP3

6

(Find all image scripts for this chapter in Appendix E.)

The multi-domain non-structural protein 3 (Nsp3) is the largest protein encoded by the coronavirus (CoV) genome, with an average molecular mass of about 200 kD. Nsp3 is a multifunctional protein comprising up to 16 different domains and regions and is an essential component of the replication/transcription complex. It comprises various domains, the organization of which differs between CoV genera, due to duplication or absence of some domains. However, eight domains of Nsp3 exist in all known CoVs. (Review: Lei et al. [2018].)

### 6.1 ADP ribose phosphatase of NSP3 from SARS CoV-2

There are 3 published structures for the SARS CoV-2 ribose phosphatase portion of NSP3 as shown in the following table:

### **TABLE 6.1:** PDB entries for ADP ribose phosphatase portion of NSP3

ID	Title
6W02	Crystal Structure of ADP ribose phosphatase of NSP3 from
	SARS CoV-2 in the complex with ADP ribose
6VXS	Crystal Structure of ADP ribose phosphatase of NSP3 from
	SARS CoV-2

ID Title

**6W6Y**Crystal Structure of ADP ribose phosphatase of NSP3 from SARS CoV-2 in complex with AMP

Crystal Structure 6W02 with ADP ribose phosphatase of NSP3 from SARS CoV-2 in the complex with ADP ribose is very similar to that of MERS-Cov<sup>1</sup> (5H0L) with an RMSD of  $0.63^2$  which is depicted in Figure 4 by Lei et al. [2018]. The two structures are easily superimposed (figure 6.1.)



FIGURE 6.1: 6W02 aligned with 5HOL.

Figure 6.2 illustrates a closer view of ADP within structure 6W02. Also visible is ethylene glycol (alias 1,2-ethanediol) and labeled EDO within the PDB file. Amino acid ILE 131 has been shown only as sticks to facilitate view.

<sup>&</sup>lt;sup>1</sup>Middle-East Respiratory Syndrome Coronavirus

<sup>&</sup>lt;sup>2</sup>see 4.4 for RMSD calculation method

#### 6.1 ADP ribose phosphatase of NSP3 from SARS CoV-2



FIGURE 6.2: Close view of ADP within 6Wo2.

Here are a few highlights from the scripts for this chapter (Appendix E.)

- util.cbaw, util.cbab and util.cbay are utilities to color by atom white, blue, and yellow respectively.
- hide nonbonded is a way to hide the waters and other "floating" solvent molecules.
- organic usually serves to select ligands, but can also select specific organic solvents.

|\_\_\_\_ | \_\_\_\_

## NSP15

7

(Find all image scripts for this chapter in Appendix F.)

The title for structure 6W01 is The 1.9 A Crystal Structure of NSP15 Endoribonuclease from SARS CoV-2 in the Complex with a Citrate.

This structure is a good illustration for the understanding of "asymetric unit" vs "biological assembly."

In simple terms, crystallographers solve the mathematical problem to obtain the smallest unit that can be reproduced almost *ad infinitum* to recreate the crystal. This "asymetric unit" is what gets deposited at the Protein Data Bank.

Fortunately these days, a set of symmetry information is built-in the PDB file so that it is possible to download the "biological assembly" which is the biologically active compound.

For this entry, the "asymetric unit" is composed of 2 molecules while the "biological assembly" is composed of a total of 6, consisting of 2 trimers.

Note that in other cases, the "biological assembly" may consist of a smaller number than the "asymetric unit."

Figure 7.1 illustrates the "asymetric unit" composed of 2 chains. Residues in the points of contact are shown as spheres. Within the PDB, information for the crystallographers would inform them that the "space group" if P 63 and would also provide relevant information about symmetry operators.

7 NSP15



**FIGURE 7.1:** 6W01 asymetric unit.



**FIGURE 7.2:** 6W01 biological assembly: 2 trimers.

Figure 7.2 is the first of two illustrations for the "biological assembly" that will appear as two trimers. All chains within a trimer have the same chain ID. One trimer is A and the other B. Therefore within one trimer it is not possible to uniquely address residues or atoms. This makes labeling more difficult and therefore there are no internal title within this illustration. (We'll fix that for the next illustration.) In addition, by default PyMOL will show only one of the chain for each trimer. Each trimer contains 3 "models" or "state", see command explanation below.

In this illusration alpha helices are shown as a helix reprentation. Compare with figure 7.3 using a different representation.



FIGURE 7.3: 6W01 biological assembly: all chains colored.

The script (Appendix F) contains a few critical PyMOL commands:

• set all\_states, on will show all chains within each trimer

#### 7.0

#### 7 NSP15

- set cartoon\_cylindrical\_helices, on to change the helix representaion mode
- run flatten\_obj.py to acquire a new function to allow the renumbering of all chains. *Please note* that file flatten\_obj.py has to be downloaded within the current directory (See pymolwiki<sup>1</sup> or download directly from Github<sup>2</sup> short URL<sup>3</sup>.

In addition the raytrace rendering was changed to provide a different look, with more "flat" coloring and a black line outline. This is thanks to the command set ray\_trace\_mode, 1. In addition cast shadows were removed with set ray\_shadow, off as they would add confusion to this large structure.

<sup>&</sup>lt;sup>1</sup>https://pymolwiki.org/index.php/Flatten\_obj

<sup>&</sup>lt;sup>2</sup>https://raw.githubusercontent.com/Pymol-Scripts/Pymol-

<sup>&</sup>lt;sup>3</sup>http://tiny.cc/flatten-obj

## NSP12 polymerase

(Find all image scripts for this chapter in Appendix G.)

The title for structure 6M71 is 2019-*nCoV* RNA-dependent RNA polymerase in complex with cofactors.(Deposited: 2020-03-16 Released: 2020-04-01)

It is very similar to structure 6NUR with title *SARS-Coronavirus NSP12 bound to NSP7 and NSP8 co-factors* (Deposited: 2019-02-01 Released: 2019-05-29) Kirchdoerfer and Ward [2019].

Figure 8.1 shows the structure as colored and oriented as in Kirchdoerfer and Ward [2019] figure 1b, middle figure with legend: Structure of SARS-CoV nsp12 bound to nsp7 and nsp8 co-factors. b SARS-CoV nsp12 contains a large N-terminal extension composed of the NiRAN domain (dark red) and an interface domain (purple) adjacent to the polymerase domain (orange). nsp12 binds to a heterodimer of nsp7 (blue) and nsp8 (green) as well as to a second subunit of nsp8.

With some extra PyMOLcommands the labels could be made to colormatch the proteins.

Figure 8.2 compares structures 6M71 and 6NUR for which more of the nsp8 protein is visible (top right of the structure.)

Figure 1 legend continues as: *c*–*e* Comparison of SARS-CoV nsp12 to the polymerase proteins of poliovirus24 (3OL6.pdb) [10.2210/pdb3OL6/pdb] and dengue virus55 (4VoR.pdb) [10.2210/pdb4VoR/pdb]. Viral RNA polymerases share an overall structural architecture with fingers (blue), palm (yellow), and

8 NSP12 polymerase



**FIGURE 8.1:** 6M71 RNA-dependent RNA polymerase nsp12 in complex with cofactors nsp7 and nsp8.

thumb domains (red). Both SARS-CoV and dengue virus polymerase proteins contain N-terminal extensions colored green. The SARS-CoV nsp7 and nsp8 co-factors are colored in white and gray, respectively

I leave the pleasure to the reader(s) to create the corresponding images for the SARS and the other viral polymerases.

The script (Appendix G) contains a few critical PyMOL commands:

- hide everything, 6NUR.Fetching the 6NUR1 structure early allows to color both structures at the same time. For the first image structure 6NUR is hidden from view.
- translate [40, 0, 0], 6M71 this command permits to separate the 2 structures for better comparison.



**FIGURE 8.2:** Comparison between Cov (6NUR) and Cov2 (6M71) RNAdependent RNA polymerase nsp12 in complex with cofactors nsp7 and nsp8.

8.0

|\_\_\_\_ | \_\_\_\_

## Epilogue

9

I hope that this booklet will help you with your PyMOL skills. I probably left a few PyMOL commands unexplained that you can easily learn from a simple web search, as I often do.

I certainly learned or relearned many aspects of using PyMOL while working on this. Perhaps even more so as I wanted the image to be able to be reproduced *automatically* by script with the booklet creation method itself. As detailed in the Introduction (1) I have created this booklet using the "*markdown*" language augmented for *RStudio, bookdown* and *bookdownplus*. I have tried to use this method to add crossreferences, appendices and a book index that I hope are useful and consistent. It sure was a learning curve for me, but it was an interesting endeavor even with some late night frustrations.

To know more about **bookdown**, see https://bookdown.org.

The next step will be to create a (smaller) booklet for printing and coloring structures with crayons. May this help those at home with little ones and ride the difficulties we face during the pandemic.



FIGURE 9.1: SARS-CoV-2 3D model for crayon coloring.

# SARS-CoV-2 PDB codes

Α

The information in these tables is derived from the Protein Data Bank web site SARS-CoV-2 article:

- Original link: "https://www.rcsb.org/news?year= 2020&article=5e74d55d2d410731e9944f52&feature=true"
- Short URL: http://tiny.cc/PDBCovid19
- Short URL: https://bit.ly/2UXfwPQ
- Archived: 03April2020
  - Short URL: http://tiny.cc/PDBCovid19ARC
  - Short URL: https://bit.ly/2X8c0ob

## A.1 Main protease

*Note*: For clarity *PanDDA analysis group deposition – Crystal Structure of* has been removed for all PanDDA structures titles.

A SARS-CoV-2 PDB codes

#### TABLE A.1: PDB IDs of SARS-CoV-2 main protease

#### ID Title

5R8T PanDDA analysis group deposition of ground-state model of SARS-CoV-2 main protease screened against DSI poised (Enamine), Fraglites and Peplites (Newcastle university), Mini Frags (Astex), York 3D (York university), electrophile cysteine covalent (Weizman institute) fragment libraries 5RE4 SARS-CoV-2 main protease in complex with Z1129283193 5RE5 SARS-CoV-2 main protease in complex with Z33545544 5RE6 SARS-CoV-2 main protease in complex with Z54571979 5RE7 SARS-CoV-2 main protease in complex with Z30932204 5RE8 SARS-CoV-2 main protease in complex with Z2737076969 5RE9 SARS-CoV-2 main protease in complex with Z2856434836 5REA SARS-CoV-2 main protease in complex with Z31432226 5REB SARS-CoV-2 main protease in complex with Z2856434899 5REC SARS-CoV-2 main protease in complex with Z1587220559 5RED SARS-CoV-2 main protease in complex with Z2856434865 5REE SARS-CoV-2 main protease in complex with Z2217052426 5REF SARS-CoV-2 main protease in complex with Z24758179 5REG SARS-CoV-2 main protease in complex with Z1545313172 5REH SARS-CoV-2 main protease in complex with Z111507846 5REI SARS-CoV-2 main protease in complex with Z2856434856 5REJ SARS-CoV-2 main protease in complex with PCM-0102241 5REK SARS-CoV-2 main protease in complex with PCM-0102327 5REL SARS-CoV-2 main protease in complex with PCM-0102340 5REM SARS-CoV-2 main protease in complex with PCM-0103016 5REN SARS-CoV-2 main protease in complex with PCM-0102425 5REO SARS-CoV-2 main protease in complex with PCM-0102578 5REP SARS-CoV-2 main protease in complex with PCM-0102201 5RER SARS-CoV-2 main protease in complex with PCM-0102615 5RES SARS-CoV-2 main protease in complex with PCM-0102281

## ID Title

- DET	SAPS Coll a main protocol in complex with PCM execution
SKEI	SARS-Cov-2 main protease in complex with PCM-0102269
5REU	SARS-CoV-2 main protease in complex with PCM-0102395
5REV	SARS-CoV-2 main protease in complex with PCM-0103072
5REW	SARS-CoV-2 main protease in complex with PCM-0102275
5REX	SARS-CoV-2 main protease in complex with PCM-0102287
5REY	SARS-CoV-2 main protease in complex with PCM-0102911
5REZ	SARS-CoV-2 main protease in complex with POB0129
5RF0	SARS-CoV-2 main protease in complex with POB0073
5RF1	SARS-CoV-2 main protease in complex with NCL-00023830
5RF2	SARS-CoV-2 main protease in complex with Z1741969146
5RF3	SARS-CoV-2 main protease in complex with Z1741970824
5RF4	SARS-CoV-2 main protease in complex with Z1741982125
5RF5	SARS-CoV-2 main protease in complex with Z3241250482
5RF6	SARS-CoV-2 main protease in complex with Z1348371854
5RF7	SARS-CoV-2 main protease in complex with
	Z316425948_minor
5RF8	SARS-CoV-2 main protease in complex with Z271004858
5RF9	SARS-CoV-2 main protease in complex with Z217038356
5RFA	SARS-CoV-2 main protease in complex with Z2643472210
5RFB	SARS-CoV-2 main protease in complex with Z1271660837
5RFC	SARS-CoV-2 main protease in complex with Z979145504
5RFD	SARS-CoV-2 main protease in complex with Z126932614
5RFE	SARS-CoV-2 main protease in complex with Z509756472
5RFF	SARS-CoV-2 main protease in complex with PCM-0102704
5RFG	SARS-CoV-2 main protease in complex with PCM-0102372
5RFH	SARS-CoV-2 main protease in complex with PCM-0102277
5RFI	SARS-CoV-2 main protease in complex with PCM-0102353
5RFJ	SARS-CoV-2 main protease in complex with PCM-0103067
5RFK	SARS-CoV-2 main protease in complex with PCM-0102575
5RFL	SARS-CoV-2 main protease in complex with PCM-0102389
5RFM	SARS-CoV-2 main protease in complex with PCM-0102539
	· ·

A SARS-CoV-2 PDB codes

## ID Title

5RFN	SARS-CoV-2 main protease in complex with PCM-0102868
5RFO	SARS-CoV-2 main protease in complex with PCM-0102972
5RFP	SARS-CoV-2 main protease in complex with PCM-0102190
5RFQ	SARS-CoV-2 main protease in complex with PCM-0102179
5RFR	SARS-CoV-2 main protease in complex with PCM-0102169
5RFS	SARS-CoV-2 main protease in complex with PCM-0102739
5RFT	SARS-CoV-2 main protease in complex with PCM-0102432
5RFU	SARS-CoV-2 main protease in complex with PCM-0102121
5RFV	SARS-CoV-2 main protease in complex with PCM-0102306
5RFW	SARS-CoV-2 main protease in complex with PCM-0102243
5RFX	SARS-CoV-2 main protease in complex with PCM-0102254
5RFY	SARS-CoV-2 main protease in complex with PCM-0102974
5RFZ	SARS-CoV-2 main protease in complex with PCM-0102274
5RG0	SARS-CoV-2 main protease in complex with PCM-0102535
6W63	Structure of COVID-19 main protease bound to potent
	broad-spectrum non-covalent inhibitor X77
6YB7	SARS-CoV-2 main protease with unliganded active site
	(2019-nCoV, coronavirus disease 2019, COVID-19)
5R7Y	COVID-19 main protease in complex with Z45617795
5R7Z	COVID-19 main protease in complex with Z1220452176
5R80	COVID-19 main protease in complex with Z18197050
5R81	COVID-19 main protease in complex with Z1367324110
5R82	COVID-19 main protease in complex with Z219104216
5R83	COVID-19 main protease in complex with Z44592329
5R84	COVID-19 main protease in complex with Z31792168
6M03	The crystal structure of COVID-19 main protease in apo
	form
6Y84	SARS-CoV-2 main protease with unliganded active site
	(2019-nCoV, coronavirus disease 2019, COVID-19)
6Y2E	Crystal structure of the free enzyme of the SARS-CoV-2
	(2019-nCoV) main protease

ID	Title
6Y2F	Crystal structure (monoclinic form) of the complex resulting
	from the reaction between SARS-CoV-2 (2019-nCoV) main
	protease and tert-butyl
	(1-((S)-1-(((S)-4-(benzylamino)-3,4-dioxo-1-((S)-2-
	oxopyrrolidin-3-yl)butan-2-yl)amino)-3-cyclopropyl-1-
	oxopropan-2-yl)-2-oxo-1,2-dihydropyridin-3-yl)carbamate
	(alpha-ketoamide 13b)
6Y2G	Crystal structure (orthorhombic form) of the complex
	resulting from the reaction between SARS-CoV-2
	(2019-nCoV) main protease and tert-butyl
	(1-((S)-1-(((S)-4-(benzylamino)-3,4-dioxo-1-((S)-2-
	oxopyrrolidin-3-yl)butan-2-yl)amino)-3-cyclopropyl-1-
	oxopropan-2-yl)-2-oxo-1,2-dihydropyridin-3-yl)carbamate
	(alpha-ketoamide 13b)
6LU7	The crystal structure of COVID-19 main protease in complex
	with an inhibitor N3

## A.2 Papain-like protease

**TABLE A.2:** PDB IDs of SARS-CoV-2 Papain-like protease

6W9C The crystal structure of papain-like protease of SARS CoV-2

## A.3 Spike protein

A SARS-CoV-2 PDB codes

ID	Title
6LZG	Structure of novel coronavirus spike receptor-binding domain complexed with its receptor ACE2
6MoJ	Crystal structure of 2019-nCoV spike receptor-binding domain bound with ACE2
6VXX	Structure of the SARS-CoV-2 spike glycoprotein (closed state)
6VYB	SARS-CoV-2 spike ectodomain structure (open state)
6VSB	Prefusion 2019-nCoV spike glycoprotein with a single receptor-binding domain up

**TABLE A.3:** PDB IDs of SARS-CoV-2 spike glycoprotein

## A.4 Other SARS-CoV-2 structures

**TABLE A.4:** PDB IDs of other SARS-CoV-2

ID	Title
6M71	2019-nCoV RNA-dependent RNA polymerase in complex
	with cofactors
6W41	Crystal structure of SARS-CoV-2 receptor binding domain in
	complex with human antibody CR3022
6W61	Crystal Structure of the methyltransferase-stimulatory
	factor complex of NSP16 and NSP10 from SARS CoV-2.
6W6Y	Crystal Structure of ADP ribose phosphatase of NSP3 from
	SARS CoV-2 in complex with AMP

ID	Title
6W75	1.95 Angstrom Resolution Crystal Structure of NSP10 -
	NSP16 Complex from SARS-CoV-2
6M3M	Crystal structure of SARS-CoV-2 nucleocapsid protein
	N-terminal RNA binding domain
6W4B	The crystal structure of Nsp9 RNA binding protein of SARS
	CoV-2
6W4H	1.80 Angstrom Resolution Crystal Structure of NSP16 -
	NSP10 Complex from SARS-CoV-2
6M17	The 2019-nCoV RBD/ACE2-BoAT1 complex
6VYO	Crystal structure of RNA binding domain of nucleocapsid
	phosphoprotein from SARS coronavirus 2
6W01	The 1.9 A Crystal Structure of NSP15 Endoribonuclease from
	SARS CoV-2 in the Complex with a Citrate
6W02	Crystal Structure of ADP ribose phosphatase of NSP3 from
	SARS CoV-2 in the complex with ADP ribose
6VW1	Structure of 2019-nCoV chimeric receptor-binding domain
	complexed with its receptor human ACE2
6VWWCrystal Structure of NSP15 Endoribonuclease from SARS	
	CoV-2.
6VXS	Crystal Structure of ADP ribose phosphatase of NSP3 from
	SARS CoV-2
6LVN	Structure of the 2019-nCoV HR2 Domain
6LXT	Structure of post fusion core of 2019-nCoV S2 subunit

|\_\_\_\_ | \_\_\_\_
# 6LU7: PRD\_002214 coordinates

The 3D coordinates of compound PRD\_002214 in PDB format, from structure 6LU7. The peptide-like n nature of compund PRD\_002214 causes its coordinates to be a mixture of ATOM and HETATM records.

HETATM	2369	C4	02J	С	1	-10.425	3.420	72.447	1.00	54.38
HETATM	2370	C5	02J	С	1	-9.924	2.857	73.642	1.00	63.03
HETATM	2371	C6	02J	С	1	-9.345	1.458	73.806	1.00	64.36
HETATM	2372	01	02J	С	1	-10.048	3.755	74.575	1.00	63.01
HETATM	2373	N2	02J	С	1	-10.585	4.861	74.114	1.00	56.79
HETATM	2374	СЗ	02J	С	1	-10.848	4.738	72.810	1.00	50.26
HETATM	2375	C41	02J	С	1	-11.475	5.798	71.891	1.00	47.56
HETATM	2376	042	02J	С	1	-11.062	5.943	70.793	1.00	47.13
ATOM	2377	Ν	ALA	С	2	-12.583	6.625	72.374	1.00	41.65
ATOM	2378	CA	ALA	С	2	-13.166	7.625	71.495	1.00	42.30
ATOM	2379	С	ALA	С	2	-12.115	8.750	71.241	1.00	41.96
ATOM	2380	0	ALA	С	2	-11.466	9.159	72.125	1.00	41.95
ATOM	2381	CB	ALA	С	2	-14.397	8.218	72.132	1.00	36.98
ATOM	2382	Ν	VAL	С	3	-11.943	9.281	69.911	1.00	34.91
ATOM	2383	CA	VAL	С	3	-10.955	10.336	69.666	1.00	36.29
ATOM	2384	С	VAL	С	3	-11.663	11.606	69.273	1.00	40.78
ATOM	2385	0	VAL	С	3	-12.381	11.617	68.328	1.00	36.69
ATOM	2386	CB	VAL	С	3	-10.004	9.895	68.515	1.00	37.53
ATOM	2387	CG1	VAL	С	3	-9.130	11.086	68.012	1.00	40.26
ATOM	2388	CG2	VAL	С	3	-9.116	8.687	68.955	1.00	43.10
ATOM	2389	N	LEU	С	4	-11.452	12.859	70.085	1.00	37.32

B 6LU7: PRD\_002214 coordinates

ATOM	2390	CA	LEU	С	4	-12.124	14.096	69.716	1.00	39.95
ATOM	2391	С	LEU	С	4	-11.582	14.561	68.334	1.00	38.38
ATOM	2392	0	LEU	С	4	-10.411	14.570	68.131	1.00	44.55
ATOM	2393	CB	LEU	С	4	-11.825	15.155	70.765	1.00	41.97
ATOM	2394	CG	LEU	С	4	-13.001	15.397	71.764	1.00	45.86
ATOM	2395	CD1	LEU	С	4	-13.834	14.100	72.019	1.00	43.70
ATOM	2396	CD2	LEU	С	4	-12.487	15.982	73.082	1.00	43.98
HETATM	2397	C19	PJE	С	5	-11.993	15.425	65.951	1.00	40.54
HETATM	2398	C20	PJE	С	5	-12.499	16.817	65.647	1.00	36.98
HETATM	2399	C21	PJE	С	5	-11.365	17.802	65.289	1.00	45.43
HETATM	2400	C22	PJE	С	5	-10.167	17.764	66.278	1.00	52.38
HETATM	2401	C25	PJE	С	5	-12.456	14.464	64.858	1.00	30.76
HETATM	2402	C26	PJE	С	5	-11.827	13.003	65.048	1.00	34.19
HETATM	2403	C27	PJE	С	5	-10.215	13.041	64.725	1.00	40.55
HETATM	2404	C28	PJE	С	5	-9.925	12.061	63.836	1.00	39.60
HETATM	2405	N6	PJE	С	5	-11.238	11.571	63.275	1.00	35.08
HETATM	2406	C29	PJE	С	5	-12.383	12.148	64.256	1.00	34.70
HETATM	2407	08	PJE	С	5	-13.440	12.730	63.491	1.00	31.54
HETATM	2408	N5	PJE	С	5	-12.524	14.987	67.260	1.00	34.60
HETATM	2409	07	PJE	С	5	-10.354	17.731	67.451	1.00	57.27
HETATM	2410	С	010	С	6	-7.830	17.582	66.759	1.00	63.14
HETATM	2411	0	010	С	6	-8.840	17.782	65.766	1.00	52.13
HETATM	2412	C1	010	С	6	-5.739	18.930	67.322	1.00	68.45
HETATM	2413	C2	010	С	6	-5.067	20.138	67.502	1.00	70.02
HETATM	2414	CЗ	010	С	6	-5.744	21.347	67.320	1.00	58.91
HETATM	2415	C4	010	С	6	-7.085	21.339	66.961	1.00	57.63
HETATM	2416	C5	010	С	6	-7.760	20.134	66.778	1.00	62.65
HETATM	2417	C6	010	С	6	-7.082	18.927	66.959	1.00	66.95
TER	2418		010	С	6					

# C

# Scripts for figures in Chapter 4

### C.1 Stick and surface

# Script for "Protease Mpro" # Load coodinates and change visual fetch 6lu7, async=0 as cartoon color darksalmon, chain A # Alter atom type alter (chain C), type="HETATM" show stick, chain C hide cartoon, chain C util.cbaw chain C # Zoom and rotate zoom 61u7 and chain C turn y, -50 turn z, -90 turn x, -60 turn y, 25 move z, 10

show surface

```
C Scripts for figures in Chapter 4
```

```
# label settings and labelling
set label_size, 20
set label_color, black
set label_position =[0.0, 1.5, 8]
label n. CG1 and resi 3 and chain C , "%s, %s" % (resn, resi)
label n. C and (i. 2 or i. 4) and chain C , "%s, %s" % (resn, resi)
```

```
bg_color white
png images/cov2-2-surf-01.png, 1000, dpi=300, ray=1
```

### ##############

```
# Next figure
hide surface, 6lu7
alter (chain C), type="HETATM"
alter resn ALA+VAL+LEU and chain C, resn = "UNK"
# remove labels
label
select C, chain C
extract PRD, C
show surface, 6lu7
png images/cov2-2-surf-02.png, 1000, dpi=300, ray=1
```

### ##############

# Next figure hide surface, 6lu7 show surface, PRD png images/cov2-2-surf-03.png, 1000, dpi=300, ray=1

### ##############

# Next figure
save PRD.pdb, PRD

```
C.2 Biological assembly
delete PRD
load PRD.pdb
hide cartoon, PRD
util.cbaw chain C
show surface, PRD
png images/cov2-2-surf-04.png, 1000, dpi=300, ray=1
```

### C.2 Biological assembly

```
reinitialize
fetch 6lu7, type=pdb1, async=0
set all_states, 1
split_states 61u7
delete 6lu7
util.cbab 6lu7_0001
util.cbak 61u7_0002
as cartoon
bg_color white
alter (chain C), type="HETATM"
show sphere, chain C and 6lu7_0001
show sticks, chain C and 6lu7_0002
hide cartoon, chain C
util.cbaw chain C
orient
move z, 30
png images/cov2-dimer-1.png, 1400, dpi=300, ray=1
```

|\_\_\_\_ | \_\_\_\_

# D

# Scripts for figures in Chapter 5

```
D.1 Display content of 6LZG
# spike ACE2 + S1 fragment
reinitialize
fetch 6LZG, async=0
color wheat, chain A
color slate, chain B
remove solvent
orient
move z, 50
move x, -7
# NAG = N-Acetylglucosamine
# Show bonded AA: ASN 53 and ASN 322
show stick, (resi 53 or resi 322) and chain A
color red, organic
bg_color white
# lablels:
set label_size, 20
set label_color, black
```

```
66 D Scripts for figures in Chapter 5
set label_position =[0, 3, 0]
label /6LZG/A/A/ASP`111/CA, "ACE2"
label /6LZG/B/B/ASN`481/CA, "receptor-binding \n domain"
png images/cov2-spike-ace2-1.png, 1200, dpi=300, ray=1
```

### D.2 Display content of trimeric structures 6VSB and 6VXX

```
# spike trimerics
reinitialize
# one open
fetch 6VSB, async=0
# closed
fetch 6VXX, async=0
util.cbc
as cartoon
align 6VSB, 6VXX
# Grid mode does not work in batch
# set grid_mode, 1
# Alternate option: translate
translate [50, 0, 0], 6VSB
translate [-70, 0, 0], 6VXX
bg_color white
# Labels
```

D.3 Superimpose 6LZG and trimeric 6VSB

```
set label_size, 20
set label_color, black
set label_position =[2, 2, 8]
label /6VSB/A/A/ARG`246/CA, "6VSB (1 up)"
label /6VXX/A/A/HIS`245/CA, "6VXX (closed)"
```

```
png images/cov2-trimerics-1.png, 1400, dpi=300, ray=1
# cmd.png('images/cov2-trimerics-1.png', '30cm', '10cm', dpi=300, ray=1)
```

#### ###########

```
# Next figure
turn x, -90
label
label /6VSB/B/B/THR`500/CA, "6VSB (1 up)"
label /6VXX/A/A/LYS`444/CA, "6VXX (closed)"
```

```
png images/cov2-trimerics-2.png, 1400, dpi=300, ray=1
```

### D.3 Superimpose 6LZG and trimeric 6VSB

```
# spike trimer and ACE2 + S1 fragment
reinitialize
fetch 6VSB, async=0
fetch 6LZG, async=0
util.cbc
select S1, 6LZG and chain B
super S1 (6VSB and chain A)
```

super S1, (6VSB and chain A)
as cartoon
# as ribbon

```
68
                                   D Scripts for figures in Chapter 5
color wheat, 6LZG and chain A
color slate, S1
# show cartoon, 6LZG and chain A
# show cartoon, S1
# extract S1, S1
bg_color white
set_view (\
     0.569421589,
                    -0.808852911,
                                      0.146665663, \
     0.195740119,
                    -0.039873410,
                                     -0.979845762, \
     0.798400283,
                    0.586652577,
                                     0.135621011,\
                     0.000237614, -561.803344727,\
    -0.000902131,
   216.354858398,
                   211.111328125,
                                    255.279617310,\
                   607.512878418,
                                    -20.00000000)
   516.146301270,
select none
# lablels:
set label_size, 20
set label_color, black
set label_position =[0, 0, 8]
label /6LZG/A/A/ALA`614/CA, "ACE2"
label /6LZG/B/B/THR`478/CA, "receptor-binding \n domain"
png images/cov2-spike-trimer-1.png, 1400, dpi=300, ray=1
```

### D.4 6ACK spike glycoprotein trimer with bound ACE2

```
# spike glycoprotein trimer with ACE2 bound
reinitialize
fetch 6ACK, async=0
util.cbc
```

```
D.4 6ACK spike glycoprotein trimer with bound ACE2
                                                        69
orient
turn z, 45
move z, 40
bg color white
# lablels:
set label_size, 20
set label color, black
set label_position =[0, 1, 8]
label /6ACK//D/GLU`87/CA, "ACE2"
label /6ACK//B/THR`433/CA, "spike glycoprotein trimer"
png images/cov2-spike-trimer-2-ace2-1.png, 1400, dpi=300, ray=1
##############
# Next figure
extract ACE2, chain D
select byres chain C within 5 of ACE2
show stick, sele
util.cbaw sele
show surface, ACE2
set_view (\
    -0.100384109,
                     -0.168694198, 0.980534792, \setminus
    -0.242882282,
                     0.959854066,
                                       0.140271023,\
    -0.964840829, -0.224080622,
                                      -0.137331456, \setminus
                    -0.002141094, -107.068008423,
     0.004348084,
   218.490600586, 243.002914429,
                                     125.151054382,\
```

160.088317871,

-20.00000000)

52.256690979,

D Scripts for figures in Chapter 5

zoom sele
bg\_color black
label

png images/cov2-spike-trimer-2-ace2-2.png, 1400, dpi=300, ray=1

# E

# Scripts for figures in chapter <mark>6</mark>

### E.1 Nsp3 - ribose ADP

```
# spike glycoprotein trimer with ACE2 bound
reinitialize
fetch 6ACK, async=0
util.cbc
orient
turn z, 45
move z, 40
bg_color white
# lablels:
set label_size, 20
set label_color, black
set label_position =[0, 1, 8]
label /6ACK//D/GLU`87/CA, "ACE2"
label /6ACK//B/THR`433/CA, "spike glycoprotein trimer"
png images/cov2-spike-trimer-2-ace2-1.png, 1400, dpi=300, ray=1
##############
# Next figure
```

```
72
                                    E Scripts for figures in chapter 6
extract ACE2, chain D
select byres chain C within 5 of ACE2
show stick, sele
util.cbaw sele
show surface, ACE2
set_view (\
    -0.100384109,
                    -0.168694198,
                                    0.980534792,\
    -0.242882282,
                    0.959854066,
                                      0.140271023,\
                    -0.224080622,
    -0.964840829,
                                    -0.137331456, \setminus
                    -0.002141094, -107.068008423,\
     0.004348084,
   218.490600586,
                    243.002914429, 125.151054382,\
    52.256690979,
                   160.088317871, -20.000000000)
```

```
zoom sele
bg_color black
label
```

png images/cov2-spike-trimer-2-ace2-2.png, 1400, dpi=300, ray=1

# Scripts for figures in chapter 7

F

### F.1 Nsp15 - asymetric unit and biological assembly

```
# NSP15 asymetric unit
fetch 6W01,async=0
as cartoon
util.cbc
orient
move z, 70
turn z, -5
# Select points of contact with 5 A
select byres chain A within 5 of chain B
show sphere, sele
select byres chain B within 5 of chain A
show sphere, sele
bg_color white
set label_size, 20
set label_color, black
set label_position =[0, 6, 8]
label /6W01/B/B/GLN<sup>2</sup>0/CA, "Asymetric unit: 2 chains"
set ray_trace_mode, 1
png images/cov2-nsp15-1.png, 1200, dpi=300, ray=1
```

### 

# Next image

```
74
                                     F Scripts for figures in chapter 7
# Biological assembly version 1
reinitialize
fetch 6W01, type=pdb1, async=0
set all_states, on
bg color white
orient
move z, 30
turn y, 180
as cartoon
util.cbc
# set cartoon_cylindrical_helices, on
set label_size, 20
set label_color, black
set label_position =[2, 1, 8]
label /flat//F/SER`316/CA, "Biological assembly: 2 trimers"
set ray_trace_mode, 1
set ray_shadow, off
png images/cov2-nsp15-2.png, 1400, dpi=300, ray=1
```

```
# Next image
# Biological assembly version 2
reinitialize
fetch 6W01, type=pdb1, async=0
# From: https://pymolwiki.org/index.php/Flatten_obj
run flatten_obj.py
flatten_obj flat, 6W01
bg_color white
orient
move z, 30
as cartoon
util.cbc
```

```
F.1 Nsp15-asymetric unit and biological assembly 75
set cartoon_cylindrical_helices, on
set label_size, 20
set label_color, black
set label_position =[2, 1, 8]
label /flat//F/SER`316/CA, "Biological assembly: 2 trimers, 6 chains"
set ray_trace_mode, 1
set ray_shadow, off
png images/cov2-nsp15-3.png, 1400, dpi=300, ray=1
```

|\_\_\_\_ | \_\_\_\_

# G

## Scripts for figures in chapter 8

```
G.1 SARS-CoV-2 nsp12 polymerase bound to nsp7 and nsp8 co-
    factors
# SARS-CoV-2 NSP12
fetch 6M71, async=0
fetch 6NUR, async=0
as cartoon
color sand
color splitpea, chain B or chain D
color teal, chain C
align 6M71,6NUR
hide everything, 6NUR
set_view (\
    -0.086567082,
                     -0.070623048,
                                     -0.993739665, \setminus
    -0.930820167,
                    0.361255229,
                                      0.055411216, \setminus
     0.355083704,
                      0.929789126,
                                      -0.097008832,\
     0.000127658,
                     -0.000919968, -318.655242920,\
   150.189804077,
                    138.051406860,
                                     156.410263062,\
   264.730682373,
                    372.563507080,
                                     -20.00000000)
bg_color white
set label size, 20
```

G Scripts for figures in chapter 8

```
set label_color, black
set label outline color, white
set label position =[0.5, 6, 16]
label /6M71/C/D/ARG`111/CA, "nsp8"
label /6M71/B/C/ARG`21/CA, "nsp7"
label /6M71/D/B/ARG`96/CA, "nsp8"
label /6M71/A/A/ASP`824/CA, "polymerase"
set ray_trace_mode, 1
png images/cov2-nsp12-1.png, 1400, dpi=300, ray=1
# Next image - continue
# Side by side with SARS-CoV NSP12 6NUR
# reinitialize
show cartoon, 6NUR
translate [40, 0, 0], 6M71
translate [-40, 0, 0], 6NUR
move z, -75
label /6NUR/B/B/ASP`78/CA, "6NUR"
bg_color white
set ray_trace_mode, 1
set ray_shadow, off
png images/cov2-nsp12-2.png, 1400, dpi=300, ray=1
```

### Bibliography

- Sevki Adem, Volkan Eyupoglu, Iqra Sarfraz, Azhar Rasul, and Muhammad Ali. Identification of potent covid-19 main protease (mpro) inhibitors from natural polyphenols: An in silico strategy unveils a hope against corona. *Preprints*, 2020. URL https://www. preprints.org/manuscript/202003.0333/v1.
- A. E. Gorbalenya, S. C. Baker, R. S. Baric, R. J. de Groot, C. Drosten,
  A. A. Gulyaeva, B. L. Haagmans, C. Lauber, A. M. Leontovich,
  B. W. Neuman, D. Penzar, S. Perlman, L. L. M. Poon, D. V. Samborskiy, I. A. Sidorov, I. Sola, and J. Ziebuhr. The species Severe acute respiratory syndrome-related coronavirus: classifying 2019-nCoV and naming it SARS-CoV-2. Nat Microbiol, Mar 2020. [DOI:10.1038/s41564-020-0695-z<sup>1</sup>] [PubMed:32123347<sup>2</sup>].
- R. Hilgenfeld. From SARS to MERS: crystallographic studies on coronaviral proteases enable antiviral drug design. *FEBS J.*, 281(18):4085– 4096, Sep 2014. [DOI:10.1111/febs.12936<sup>3</sup>] [PubMed:25039866<sup>4</sup>].
- Zhenming Jin, Xiaoyu Du, Yechun Xu, Yongqiang Deng, Meiqin Liu, Yao Zhao, Bing Zhang, Xiaofeng Li, Leike Zhang, Chao Peng, and et al. Structure of mpro from covid-19 virus and discovery of its inhibitors. *bioRxiv*, (2020.02.26.964882), Jan 2020. URL https:// www.biorxiv.org/content/10.1101/2020.02.26.964882v2.

<sup>&</sup>lt;sup>1</sup>https://dx.doi.org/10.1038/s41564-020-0695-z

<sup>&</sup>lt;sup>2</sup>https://www.ncbi.nlm.nih.gov/pubmed/32123347

<sup>&</sup>lt;sup>3</sup>https://dx.doi.org/10.1111/febs.12936

<sup>&</sup>lt;sup>4</sup>https://www.ncbi.nlm.nih.gov/pubmed/25039866

- R. N. Kirchdoerfer and A. B. Ward. Structure of the SARS-CoV nsp12 polymerase bound to nsp7 and nsp8 co-factors. *Nat Commun*, 10(1): 2342, 05 2019. [PubMed Central:PMC6538669<sup>5</sup>] [DOI:10.1038/s41467-019-10280-3<sup>6</sup>] [PubMed:25451065<sup>7</sup>].
- J. Lei, Y. Kusov, and R. Hilgenfeld. Nsp3 of coronaviruses: Structures and functions of a large multi-domain protein. *Antiviral Res.*, 149:58-74, 01 2018. [DOI:10.1016/j.antiviral.2017.11.001<sup>8</sup>] [PubMed:29128390<sup>9</sup>].
- RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA, 2015. URL http://www.rstudio.com/.
- Schrödinger, LLC. The PyMOL molecular graphics system, version 2.0. 2020. URL https://pymol.org/.
- Sgro, Jean-Yves. Molecular Graphics Essentials PyMOL. 2017. URL https://bcrf.biochem.wisc.edu/2019/06/20/pymoltutorial-books-released/.
- Tim Skern. *Exploring protein structure: principles and practice*. Springer, 2018. ISBN 978-3-319-76857-1. URL doi:10.1007/978-3-319-76858-8.
- J. Tan, K. H. Verschueren, K. Anand, J. Shen, M. Yang, Y. Xu, Z. Rao,
  J. Bigalke, B. Heisen, J. R. Mesters, K. Chen, X. Shen, H. Jiang, and R. Hilgenfeld. pH-dependent conformational flexibility of the SARS-CoV main proteinase (M(pro)) dimer: molecular dynamics simulations and multiple X-ray structure analyses. J. Mol. Biol., 354(1):25-40, Nov 2005. URL doi.org/10.1016/j.jmb.2005.09.012.

<sup>&</sup>lt;sup>5</sup>https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6538669 <sup>6</sup>https://dx.doi.org/10.1038/s41467-019-10280-3 <sup>7</sup>https://www.ncbi.nlm.nih.gov/pubmed/25451065 <sup>8</sup>https://dx.doi.org/10.1016/j.antiviral.2017.11.001

<sup>%</sup> https://www.ncbi.nlm.nih.gov/pubmed/29128390

### G.1 Bibliography

- Yihui Xie. Dynamic Documents with R and knitr. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition, 2015. URL http:// yihui.name/knitr/. ISBN 978-1498716963.
- Yihui Xie. bookdown: Authoring Books and Technical Documents with R Markdown. Chapman and Hall/CRC, Boca Raton, Florida, 2016. URL https://github.com/rstudio/bookdown. ISBN 978-1138700109.
- Linlin Zhang, Daizong Lin, Xinyuanyuan Sun, Ute Curth, Christian Drosten, Lucie Sauerhering, Stephan Becker, Katharina Rox, and Rolf Hilgenfeld. Crystal structure of sars-cov-2 main protease provides a basis for design of improved α-ketoamide inhibitors. Science, 2020. ISSN 0036-8075. doi: 10.1126/science.abb3405. URL https://science.sciencemag.org/content/early/2020/03/20/science.abb3405.

|\_\_\_\_ | \_\_\_\_

### Index

ACE2, 30, 32 Angiotensin-converting enzyme 2, 30 Asymertic unit, 41 Biological assembly, 41 coordinates, record type, 18 COVID-19, 17 MERS-Cov, 38 N-Acetylglucosamine, 30 Nsp3, 37 PDB

> 2BX4, SARS-CoV Mpro, 25 5HOL, Nsp3 rib. phos., 38 6ACK,trimer spike+ACE2, 32 6LU7, SARS-CoV-2 Mpro, 17 6LZG, Spike+ACE2, 30 6M03, SARS-CoV-2 Mpro, 25 6M0J, Spike+ACE2, 30 6M71, Nsp12 polymerase, 45 6NUR, Nsp12 polymerase, 45 6W01, Nsp15 Endoribonuclease, 41 6W02, Nsp3 rib. phos., 38 6Y2E, SARS-CoV-2 Mpro, 27

6Y2F, SARS-CoV-2 Mpro, 27 6Y2G, SARS-CoV-2 Mpro, 27 ATOM, 18 CONECT, 24 HETATM, 18 NAG, **30** TER, 24 plain text, 11 polymerase, RNA dependent, 45 Protein Data Bank, 1 PyMOL API cmd.color, 14 cmd.super, 28 PyMOL Argument byres, 33 everything, 46 nonbonded, 39 organic, 39 within, 33 PyMOL Command align, 32, 33 alter, 19 cat, 16 color, 14 delete, 23 extract, 21, 33

get\_view, 32 hide, 39, 46 load, 23 ls, 16 pwd, 16 run, 44 save, 23 select, 22 set\_view, 32 show surface, 23 super, 27, 32, 33 util.cbab, 39 util.cbaw, 33, 39 util.cbay, 39 util.cbc, 33 PyMOL python flatten\_obj.py, 44 PyMOL Setting all\_states, 43 cartoon\_cylindrical\_helices, 44 pdb\_conect\_all, 24 pdb\_conect\_nodup, 24 pdb\_use\_ter\_records, 24 ray\_shadow, 44 ray\_trace\_mode, 44 R package bookdown, ix bookdownplus, ix knitr, ix RMSD, 27

SARS-CoV-2, 17

```
SARS-CoV-2 protease, 17
Software
nano, 12
Notepad, 11
RStudio, ix
TextEdit, 12
Spike
S1, receptor region, 31
bound to ACE2 receptor, 32
trimeric structure, 30
```

### Trypsin, 32

84

### Index