

RESEARCH ARTICLE

PDB-tools web: A user-friendly interface for the manipulation of PDB files

Brian Jiménez-García¹  | João M. C. Teixeira²  | Mikael Trellet¹  |
João P. G. L. M. Rodrigues³  | Alexandre M. J. J. Bonvin¹ 

¹Bijvoet Centre for Biomolecular Research, Utrecht University, Utrecht, The Netherlands

²Program in Molecular Medicine, The Hospital for Sick Children, Toronto, Ontario, Canada

³Department of Structural Biology, Stanford University School of Medicine, Stanford, California

Correspondence

Alexandre M. J. J. Bonvin, Bijvoet Centre for Biomolecular Research, Utrecht University, Utrecht, 3584 CH, The Netherlands.
Email: a.m.j.j.bonvin@uu.nl

Brian Jiménez-García, Bijvoet Centre for Biomolecular Research, Utrecht University, Utrecht, 3584 CH, The Netherlands.
Email: bjimenezgarcia@uu.nl

Funding information

Horizon 2020 Framework Programme, Grant/Award Numbers: BioExcel/823830, EOSC-hub/777536; National Institutes of Health, Grant/Award Number: R35GM122543

Abstract

The Protein Data Bank (PDB) file format remains a popular format used and supported by many software to represent coordinates of macromolecular structures. It however suffers from drawbacks such as error-prone manual editing. Because of that, various software toolkits have been developed to facilitate its editing and manipulation, but, to date, there is no online tool available for this purpose. Here we present PDB-Tools Web, a flexible online service for manipulating PDB files. It offers a rich and user-friendly graphical user interface that allows users to mix-and-match more than 40 individual tools from the *pdb-tools* suite. Those can be combined in a few clicks to perform complex pipelines, which can be saved and uploaded. The resulting processed PDB files can be visualized online and downloaded. The web server is freely available at <https://wenmr.science.uu.nl/pdbtools>.

KEYWORDS

bioinformatics, PDB, structural biology, web server

1 | INTRODUCTION

The Protein Data Bank (PDB) format, which was created in 1976 to allow researchers to store and share 3D structures, remains a popular file format used by many software to represent coordinates of macromolecular structures such as proteins or nucleic acids,¹ even though the macromolecular Crystallographic Information Framework (mmCIF) dictionary² is now the standard for the worldwide PDB (wwPDB).³ Understanding how the PDB file format remains in use after four decades and several technological leaps requires traveling back to the time of its inception. Despite multiple changes and revisions over the years, the core of the PDB format remains a series of lines limited to 80 characters in length, a leftover requirement from the computer punch cards used to exchange the atomic coordinates in the early days of structural biology. Each line in a PDB file refers to a specific

type of record, such as coordinates or details on the experimental setup. Each record then stores data in multiple fixed-width columns as plain text, lending very easily to visual inspection and editing.

Other file formats were developed in the past decades to overcome the limitations of the PDB format. The Crystallographic Information File (CIF) was proposed in 1991⁴ by a working group from the International Union of Crystallography (IUCr) to support electronic publication of small molecule crystal structures. Later, this format was adapted for larger macromolecules, creating the mmCIF format.² The concept behind mmCIF files is a dictionary of data items describing macromolecular structure and information of macromolecular crystallographic experiments. Unlike the PDB format, mmCIF files store data in variable-width fields and as such, there are no limits to the number of atoms, residues, or chains.

In July 2019, the mmCIF format was adopted as the official format for structural data in the worldwide PDB database (wwPDB).⁵

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2020 The Authors. *Proteins: Structure, Function, and Bioinformatics* published by Wiley Periodicals LLC.

Despite this change, the PDB format remains very much in use as the de facto file format for a large variety of structural calculation, modeling, and analysis software.

Naturally, such a long-lived and important file format gave rise to a wide and colorful variety of parsing and editing software toolkits written in nearly all popular general-purpose programming languages. Examples of such toolkits include *Bioperl*⁶, *BioJava*⁷, *BioPython*^{8,9}, *BioRuby*¹⁰, *BioJulia*¹¹, and *ESBTL*¹², for the Perl, Java, Python, Ruby, Julia, and C++ programming languages respectively. Other recent contributions include *atomium*¹³ and *Biotite*¹⁴, both written in Python. In addition to these computational frameworks, molecular visualization software such as UCSF Chimera,¹⁵ ChimeraX,¹⁶ or PyMOL,¹⁷ offer powerful parsing and editing capabilities through user-friendly graphical interfaces that require little to no programming knowledge. Operating on large collections of PDB files, however, particularly in high-performance computing environment and pipelines, requires solutions in between fully-fledged software libraries and graphical interfaces. One such solution is the *pdb-tools* project,¹⁸ a set of dependency-free command line tools written in Python and similar in philosophy to the GNU core utilities: Each tool was designed to perform a simple operation on a given PDB input, but multiple tools can be chained together in complex pipelines. The *pdb-tools* can download both PDB and mmCIF files from the wwPDB database, interconvert between the two formats, and perform a variety of selection, editing, and validation routines. This approach has been shown to be very useful for the structural biology community, judging by the popularity of the package on PyPI (<https://pypi.org/packages/pdb-tools>) and the number of clones and forks of the public repository (<https://github.com/haddock/pdb-tools>).

Our team develops several widely-used web servers, among which HADDOCK for integrative modeling of biomolecular complexes,^{19,20} DisVis²¹ for explorative modeling of protein complexes, and PowerFit²² for rigid-body fitting in cryo-EM density maps.²³ In our experience as developers, users and educators, researchers tend to favor using a web interface over downloading, compiling, and installing software. Indeed, despite increasing levels of literacy in programming and computing,²⁴ many users remain unfamiliar or uncomfortable using command-line interfaces and/or a GNU/Linux operating system. As such, we developed a web version of *pdb-tools* (PDB-Tools Web). Our web server offers a rich and user-friendly graphical user interface that allows users to mix-and-match the different tools and perform complex pipelines in a few clicks. The web server is available at <https://wenmr.science.uu.nl/pdbtools>.

2 | IMPLEMENTATION

2.1 | Code development

The PDB-Tools Web interface is available to researchers for free and does not require registration. It uses the Python 3 Flask framework (<https://flask.palletsprojects.com>, version 1.1.1) and the Jinja ([\[jinja.palletsprojects.com\]\(https://jinja.palletsprojects.com\), version 2.10.3\) templating language for server-side logic, and JavaScript for client-side logic. The web server is running alongside the other web portals operated under \[wenmr.science.uu.nl\]\(https://wenmr.science.uu.nl\).](https://</p></div><div data-bbox=)

In addition to developing the web server, we set up a support forum (<https://ask.bioexcel.eu/c/pdb-tools>) where users can easily ask questions, report problems, provide suggestions, and give feedback. Having PDB-Tools Web endorsed by the BioExcel consortium, which also operates forums for HADDOCK, DisVis and PowerFit, ensures a large visibility and a proof of its usefulness for the molecular modeling community.

2.2 | Molecular Visualization

We used the NGL Viewer v2.0.0^{25,26} JavaScript package to implement an on-the-fly molecule visualizer that users can use to inspect the PDB before and after executing the pipeline. This visualizer includes several features, such as showing molecular surfaces or water and ion molecules, that users can be enable or disable at will.

2.3 | Limitations

Because PDB-Tools Web server depends on the original *pdb-tools*, it is constrained to the input/output data formats provided by the latter. In addition, since all *pdb-tools* (except the converters) read and write PDB files, our web service is unable to handle very large structures that are available only in mmCIF format, for example, complete ribosomes. Generally, mmCIF files provided by the user will be first converted to PDB format by the *pdb_fromcif* script. If this conversion is not possible, the server will alert the user to the issue and halt execution of the pipeline. Finally, the user has the option to download the results of his pipeline in either PDB or mmCIF format.

2.4 | Documentation

The server includes a *manual* section that includes detailed documentation on how to use the server and includes a table describing all the available tools (Table 1).

3 | RESULTS AND DISCUSSION

By integrating the *pdb-tools* package with modern web technologies, as described in the section above, our PDB-Tools Web server provides a straightforward and powerful interface to manipulate both individual PDB files and (compressed) archives. The landing page offers a quick summary of the functionalities of the service and two main entries, one pointing to a pre-calculated example, and another allowing users to submit a new pipeline. In addition, the navigation bar provides links to the manual and help pages.

TABLE I List of all *pdb-tools* included in the web server

Tool name	Description
<i>pdb_splitchain*</i>	Splits a PDB file into several, each containing one chain.
<i>pdb_splitmodel*</i>	Splits a PDB file into several, each containing one MODEL.
<i>pdb_splitseg*</i>	Splits a PDB file into several, each containing one segment.
<i>pdb_b</i>	Modifies the temperature factor column of a PDB file (default 10.0).
<i>pdb_chain</i>	Modifies the chain identifier column of a PDB file (default is an empty chain).
<i>pdb_chainxseg</i>	Replaces the segment identifier column by the value in the chain identifier column of the PDB file.
<i>pdb_chkensemble</i>	Performs a basic check on a multi-model PDB file (ensemble) to ensure all models have exactly the same atoms.
<i>pdb_delchain</i>	Deletes all atoms matching specific chains in the PDB file.
<i>pdb_delelem</i>	Deletes all atoms matching the given element in the PDB file. Elements are read from the element column.
<i>pdb_delhetatm</i>	Removes all HETATM records in the PDB file.
<i>pdb_delinsertion</i>	Deletes insertion codes in a PDB file, shifting the residue numbering of downstream residues. Allows for picking specific residues to delete insertion codes for.
<i>pdb_delres</i>	Deletes a range of residues from a PDB file. The range option has three components: start, end, and step. Start and end are optional and if omitted the range will start at the first residue or end at the last, respectively. The step option can only be used if both start and end are provided. Note that the start and end values of the range are purely numerical, while the range actually refers to every N-th residue, regardless of their sequence number.
<i>pdb_delresname</i>	Removes all residues matching the given name in the PDB file. Residues names are matched without taking into consideration spaces.
<i>pdb_element</i>	Assigns the element column to the PDB file, guessing the element from the atom.
<i>pdb_gap</i>	Detects gaps between consecutive residues in the sequence, both by a distance criterion or discontinuous residue numbering. Only applies for protein residues.
<i>pdb_head</i>	Returns the first N coordinate (ATOM/HETATM) lines of the file.
<i>pdb_keeppcoord</i>	Removes all non-coordinate records from the file. Keeps only MODEL, ENDMDL, END, ATOM, HETATM, and CONECT.
<i>pdb_occ</i>	Modifies the occupancy column of a PDB file (default 1.0).
<i>pdb_reatom</i>	Renumbers atom serial numbers of the PDB file starting from a given value (default 1).
<i>pdb_reres</i>	Renumbers the residues of the PDB file starting from a given number (default 1).
<i>pdb_rplchain</i>	Performs in-place replacement of a chain identifier by another.
<i>pdb_rplresname</i>	Performs in-place replacement of a residue name by another. Affects all residues with that name.
<i>pdb_seg</i>	Modifies the segment identifier column of a PDB file (default is an empty segment).
<i>pdb_segxchain</i>	Replaces the chain identifier column by the value in the segment identifier column of the PDB file. Truncates the segment identifier if longer than one character.
<i>pdb_selaltloc</i>	Picks one location for each atom with fractional occupancy values.
<i>pdb_selatom</i>	Selects all atoms matching the given name in the PDB file. Atom names are matched without taking into consideration spaces, so 'CA' (alpha carbon) and 'CA' (calcium) will both be kept if -CA is passed.
<i>pdb_selchain</i>	Extracts one or more chains from a PDB file.
<i>pdb_selelem</i>	Selects all atoms that match the given element(s) in the PDB file. Elements are read from the element column.
<i>pdb_selhetatm</i>	Selects all HETATM records in the PDB file.
<i>pdb_selres</i>	Extracts residues from a PDB file, either arbitrarily or in a range. The range option has three components: start, end, and step. Start and end are optional and if omitted the range will start at the first residue or end at the last, respectively.
<i>pdb_selresname</i>	Selects all residues matching the given name in the PDB file. Residues names are matched without taking into consideration spaces.
<i>pdb_selseg</i>	Extracts one or more segments from a PDB file based on their segment identifiers.
<i>pdb_shiftres</i>	Renumbers the residues of the PDB file by adding/subtracting a given number from the original numbering.
<i>pdb_sort</i>	Sorts the ATOM/HETATM/ANISOU/CONNECT records in a PDB file.
<i>pdb_tidy</i>	Modifies the file to adhere (as much as possible) to the format specifications.
<i>pdb_tocif</i>	Converts a PDB file to the mmCIF format.
<i>pdb_tofasta</i>	Extracts the residue sequence in a PDB file to FASTA format.
<i>pdb_validate</i>	Validates the PDB file ATOM/HETATM lines according to the format specifications.
<i>pdb_wc</i>	Summarizes the contents of a PDB file, like the wc command in UNIX.

TABLE I (Continued)

Tool name	Description
<i>pdb_merge</i> [#]	Merges several PDB files into one. The contents are not sorted, and no lines are deleted (eg, END, TER statements) so we recommend combining it with <i>pdb_tidy</i> .
<i>pdb_mkensemble</i> [#]	Merges several PDB files into one multi-model (ensemble) file. Strips all HEADER information and adds REMARK statements with the provenance of each conformer.

Note: For each of the tools a short description is provided. Tools flagged with a "*" are pre-processing tools, usable only at the beginning of the pipeline. Those with a "#" are post-processing tools; only one of those can be used at a time and always with a previous pre-processing tool selected, for example, for merging previously split structures.

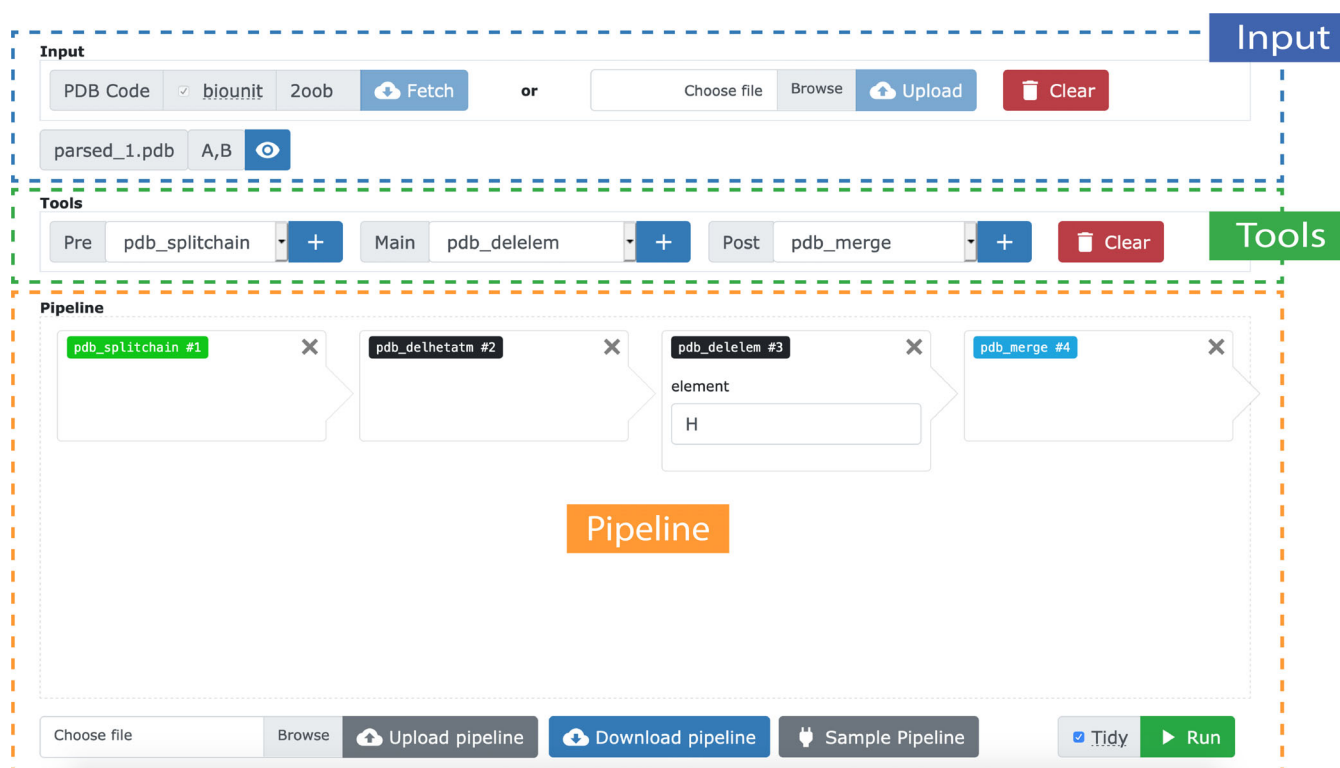


FIGURE 1 View of the submission interface. The PDB-Tools Web submission interface is composed of three main sections: (1) Input of the service, (2) available tools, and (3) pipeline canvas and controls

The submit page is a fully-featured interface providing access to all possible actions when setting up a pipeline and is organized as follows. There are three main sections: *Input*, *tools* and *pipeline* (Figure 1).

The *input* section prompts users to either provide a PDB ID code, which triggers a download from the database, or upload a file from their computer. After the structure is loaded, a small panel pops up, providing general information about the input molecule to which users can refer during the pipeline setup. Clicking the eye-shaped button opens the molecular visualization panel. Finally, a *clear* button resets the page to its initial state, discarding all previously uploaded input data and associated results.

The *tools* section is composed of three dropdown menus corresponding to pre-processing, manipulation and post-processing actions respectively. In short, the user can browse the available actions through the different dropdown menus and add them to the pipeline using the "+" button. Some actions accept options, for

example, a chainID for *pdb_selchain*, which the user can input directly in the pipeline canvas. Users can clear the current pipeline via the "clear" button on the right of this section, without losing the information provided as *input*.

The pipeline section is further sub-divided in a canvas that displays the current state of the pipeline, and a bottom section with buttons to load and save pipelines. Users can also save their pipeline (in JSON format) using the button "Download pipeline" at the bottom of the pipeline canvas. Together with the possibility to load a pipeline, through the "Upload pipeline" button, this feature allows users to reproduce previous pipelines with minimal effort. Finally, users may load an example input and pipeline using the button "Sample pipeline".

A few implementation details are worthy of note. First, while the first selected action will always act on the input structure, consecutive actions will act on the result of the previous action. In addition, by

definition, some tools can only be used at the beginning or end of a pipeline. For example, *pdb_splitchain* splits the input structure into several files, each containing an individual chain. After this action, all subsequent steps in the pipeline apply individually to all chains. On the other hand, users can use *pdb_merge* to combine all the separate files again into one single structure. Alternatively, users can simply download each individual chain.

The bottom of the submit page shows a “Tidy” option that, if checked, will run the *pdb_tidy* tool to process the output to adhere (as much as possible) to the PDB format specifications. A green “Run” button submits the pipeline to the server and executes it on the input file. The user is then redirected to a results page, where they can view and download the resulting structure and accessory output files. As expected, the output generated by *pdb-tools* and the web server can be used as input itself on another run. If a similar or identical pipeline using the same input of a previous job is required, there is a button in the results page “Resubmit this pipeline” which will facilitate this action. This might be useful for example to apply a similar pipeline to another chain of the same input PDB file.

The example page presents the result of a pre-calculated pipeline submission, allowing users to experience the look and feel of a real submission.

Finally, since the first version of our web server was made available online, in March 2020, it has processed more than 1200 individual jobs and received very good feedback from our users with an average score of 4.95 on a scale from 1 (worst) to 5 (best) based on 58 respondents.

4 | CONCLUSIONS

In this manuscript, we present PDB-Tools Web, an online tool to manipulate PDB files with a modern interface and a rich user experience. Our server is freely available and does not require registration from users. Our target audience are users that cannot, or prefer not to, install software locally. In addition, we have documented all of the server's main features exhaustively in the user manual and we offer user support through the ask.bioexcel.eu platform. We also linked the web server to our education and tutorials portal (<http://www.bonvinlab.org/education/>), as the tool of choice for manipulating PDB files. Finally, we constantly collect feedback to gauge user satisfaction and help improve our service.

In summary, we believe that PDB-Tools Web should be a valuable resource for many students and researchers in computational structural biology, further contributing to closing the existing gap between powerful command-line toolkits and user-friendly interfaces.

ACKNOWLEDGEMENTS

This work has been done with the financial support of the European Union Horizon 2020 projects BioExcel (675728, 823830) and EOSC-hub (777536). JPGLMR acknowledges funding from the National Institutes of Health USA (R35GM122543) to Michael Levitt.

PEER REVIEW

The peer review history for this article is available at <https://publons.com/publon/10.1002/prot.26018>.

ORCID

Brian Jiménez-García  <https://orcid.org/0000-0001-7786-2109>

João M. C. Teixeira  <https://orcid.org/0000-0002-9113-0622>

Mikael Trellet  <https://orcid.org/0000-0001-6337-581X>

João P. G. L. M. Rodrigues  <https://orcid.org/0000-0001-9796-3193>

Alexandre M. J. J. Bonvin  <https://orcid.org/0000-0001-7369-1322>

REFERENCES

- Berman HM. The protein data Bank: a historical perspective. *Acta Crystallogr A*. 2008;64:88-95.
- Bourne PE, Berman HM, McMahon B, Watenpaugh KD, Westbrook JD, Fitzgerald PMD. [30] Macromolecular crystallographic information file. *Methods Enzymol*. 1997;277:571-590. [https://doi.org/10.1016/S0076-6879\(97\)77032-0](https://doi.org/10.1016/S0076-6879(97)77032-0).
- Berman H, Henrick K, Nakamura H. Announcing the worldwide protein data Bank. *Nat Struct Biol*. 2003;10:980.
- Hall SR, Allen FH, Brown ID. The crystallographic information file (CIF): a new standard archive file for crystallography. *Acta Crystallogr A*. 1991;47:655-685.
- Adams PD, Afonine PV, Baskaran K, et al. Announcing mandatory submission of PDBx/mmCIF format files for crystallographic depositions to the protein data Bank (PDB). *Acta Crystallogr D*. 2019;75:451-454.
- Stajich JE, Block D, Boulez K, et al. The Bioperl toolkit: Perl modules for the life sciences. *Genome Res*. 2002;12:1611-1618.
- Lafita A, Bliven S, Prlić A, et al. BioJava 5: a community driven open-source bioinformatics library. *PLoS Comput Biol*. 2019;15:e1006791.
- Hamelryck T, Manderick B. PDB file parser and structure class implemented in python. *Bioinformatics*. 2003;19:2308-2310.
- Cock PJA, Antao T, Chang JT, et al. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*. 2009;25:1422-1423.
- Goto N, Prins P, Nakao M, Bonnal R, Aerts J, Katayama T. BioRuby: bioinformatics software for the ruby programming language. *Bioinformatics*. 2010;26:2617-2619.
- Greener JG, Selvaraj J, Ward BJ. BioStructures.jl: read, write and manipulate macromolecular structures in Julia. *Bioinformatics*. 2020;36(14):4206-4207.
- Loriot S, Cazals F, Bernauer J. ESBTL: efficient PDB parser and data structure for the structural and geometric analysis of biological macromolecules. *Bioinformatics*. 2010;26:1127-1128.
- Ireland SM, Martin ACR. Atomium—a python structure parser. *Bioinformatics*. 2020;36:2750-2754.
- Kunzmann P, Hamacher K. Biotite: a unifying open source computational biology framework in python. *BMC Bioinf*. 2018;19:346.
- Pettersen EF, Goddard TD, Huang CC, et al. UCSF chimera—a visualization system for exploratory research and analysis. *J Comput Chem*. 2004;25:1605-1612.
- Goddard TD, Huang CC, Meng EC, et al. UCSF ChimeraX: meeting modern challenges in visualization and analysis. *Protein Sci*. 2018;27:14-25.
- Schrodinger, L.L.C., 2010. The PyMOL molecular graphics system. Version 1.5
- Rodrigues JPGLM, Teixeira JMC, Trellet M, Bonvin AMJJ. Pdb-tools: a swiss army knife for molecular structures. *F1000Res*. 2018;7:1961.
- de Vries SJ, van Dijk M, Bonvin AMJJ. The HADDOCK web server for data-driven biomolecular docking. *Nat Protoc*. 2010;5:883-897.

20. van Zundert GCP, Rodrigues JPGLM, Trellet M, et al. The HADDOCK2.2 web server: user-friendly integrative modeling of biomolecular complexes. *J Mol Biol.* 2016;428:720-725.
21. van Zundert GCP, Bonvin AMJJ. DisVis: quantifying and visualizing accessible interaction space of distance-restrained biomolecular complexes. *Bioinformatics.* 2015;31:3222-3224.
22. van Zundert GCP, Bonvin AMJJ. Fast and sensitive rigid-body fitting into cryo-EM density maps with PowerFit. *AIMS Biophys.* 2015;2:73-87.
23. van Zundert GCP, Trellet M, Schaarschmidt J, et al. The DisVis and PowerFit web servers: explorative and integrative modeling of biomolecular complexes. *J Mol Biol.* 2017;429:399-407.
24. Stevens SLR, Kuzak M, Martinez C, Moser A, Bleeker P, Galland M. Building a local community of practice in scientific programming for life scientists. *PLoS Biol.* 2018;16:e2005561.
25. Rose AS, Hildebrand PW. NGL viewer: a web application for molecular visualization. *Nucleic Acids Res.* 2015;43:W576-W579.
26. Rose AS, Bradley AR, Valasatava Y, Duarte JM, Prlic A, Rose PW. NGL viewer: web-based molecular graphics for large complexes. *Bioinformatics.* 2018;34:3755-3758.

How to cite this article: Jiménez-García B, Teixeira JMC, Trellet M, Rodrigues João P. G. L. M., Bonvin AMJJ. PDB-tools web: A user-friendly interface for the manipulation of PDB files. *Proteins.* 2021;89:330–335. <https://doi.org/10.1002/prot.26018>