SOFTWARE

Open Access

Rin*maker*: a fast, versatile and reliable tool to determine residue interaction networks in proteins

Alvise Spanò[1], Lorenzo Fanton[1], Davide Pizzolato[1], Jacopo Moi[2], Francesco Vinci[1], Alberto Pesce[1], Cedrix J. Dongmo Foumthuim[2], Achille Giacometti[2,3] and Marta Simeoni[1,3*]

*Correspondence:
simeoni@unive.it

[1] Department of Environmental Science, Computer Science and Statistics, University Ca' Foscari of Venice, Via Torino 155, 30172 Venice, Italy
[2] Department of Molecular Science and Nanosystems, University Ca' Foscari of Venice, Via Torino 155, 30172 Venice, Italy
[3] European Centre for Living Technology (ECLT), Dorsoduro 3246, 30123 Venice, Italy

Abstract

Background: Residue Interaction Networks (RINs) map the crystallographic description of a protein into a graph, where amino acids are represented as nodes and non-covalent bonds as edges. Determination and visualization of a protein as a RIN provides insights on the topological properties (and hence their related biological functions) of large proteins without dealing with the full complexity of the three-dimensional description, and hence it represents an invaluable tool of modern bioinformatics.

Results: We present RIN*maker*, a fast, flexible, and powerful tool for determining and visualizing RINs that include all standard non-covalent interactions. RIN*maker* is offered as a cross-platform and open source software that can be used either as a command-line tool or through a web application or a web API service. We benchmark its efficiency against the main alternatives and provide explicit tests to show its performance and its correctness.

Conclusions: RIN*maker* is designed to be fully customizable, from a simple and handy support for experimental research to a sophisticated computational tool that can be embedded into a large computational pipeline. Hence, it paves the way to bridge the gap between data-driven/machine learning approaches and numerical simulations of simple, physically motivated, models.

Keywords: Protein 3D structure, Residue interaction network (RIN), Non-covalent bonds

Background

Recent successes of Alphafold [1] and following up analyses [2, 3] have clearly put in the limelight the power of data driven approaches in structural biology, with the number of predicted folds that is now exceeding the number of experimentally accessible protein crystals by few orders of magnitude. A detailed analysis of the three dimensional structure of a protein is clearly of paramount importance for mapping out its specific properties and hence its functionality. However, a direct use of the three-dimensional structure, when possible, has proven highly computational demanding. Residue Interaction Network (RIN) is a graph where nodes are amino acids and edges represent non-covalent interactions and can be regarded as a possible alternative whenever only information regarding the protein topology and the distribution of the specific non-covalent interactions are required.

While there are various tools in the literature for implementing and analysing RINs—see [4] for a recent survey, most of them are either embedded in a Molecular Dynamics (MD) package or are specifically designed to analyze MD trajectories and to produce as output an analysis of the corresponding networks. Usually such tools determine the C_α or C_β contact maps, or general non-covalent contacts based on atoms proximity. Some tools in the literature calculate specific non-covalent bonds [5–8], but they all come with their own shortcomings and drawbacks. RING 3.0 [5] and RIP-MD [9] are offered as a black box tool as their code is not available. Arpeggio [8] and its recent refactoring PDBe-Arpeggio [10] is an open source tool that considers a wide set of specific bonds. However, the user cannot tune the bonds thresholds, energies are not computed, and the execution is not computationally efficient. In general, a serious drawback of the tools in the literature is that the correctness of the obtained RINs is not addressed. Another common issue concerns the computational efficiency of the RIN calculation: usually the performance evaluation of the proposed algorithm is missing or insufficiently addressed.

A recent study by some of us developed a computational pipeline combining homology modeling, machine learning and graph theory techniques for the analysis of point mutations in the sequence of a specific membrane protein [11]. In that framework, one of the available tools [7] was used to generate the RIN, and that experience prompted the need for a new tool able to overcome all the above drawbacks.

To fill this gap we present RIN*maker*, a tool for determining and visualizing the RIN of a protein given in input. RIN*maker* is offered as a cross-platform and open source software that can be used either as a command-line tool or through a web application featuring a modern 3D user interface. Additionally, we offer a web API service for embedding RIN*maker* into third-party clients. RIN*maker* allows for determining the C_α contact map and a comprehensive set of non-covalent interactions. For each considered bond either default or user specified thresholds can be used for the calculation, resulting in a versatile tool. Moreover, we developed a custom integration test system to ensure the program correctness. The whole test suite is distributed along with the application, allowing anyone to reproduce the test results. Finally, computational efficiency was the main focus while designing and developing RIN*maker*. For this reason, we assessed the tool's efficiency through a series of performance evaluation tests, which highlighted that RIN*maker* can be used in any context where high-throughput production of RINs is required. RIN*maker* is supported by a detailed supplementary document (see Additional file 1) for all employed rules to identify different non-covalent interactions, as well as for the custom test cases used to assess the program correctness.

Implementation

The main goal in designing RIN*maker* was computational efficiency, which guided all the implementation choices. The style of Generic Programming (GP) [12] has been adopted as it enforces a number of optimizations. By properly defining copy-constructors, assignment operators and move-constructors for custom datatypes, C++ and GP allows avoiding unwanted replications of data structures, boosting the overall performance. By using the template system rather than the classic object-oriented programming style based on subtype polymorphism and dynamic dispatching [13], the C++ language grants maximum efficiency thanks to the static dispatching of methods and to a number of other compile-time optimizations.

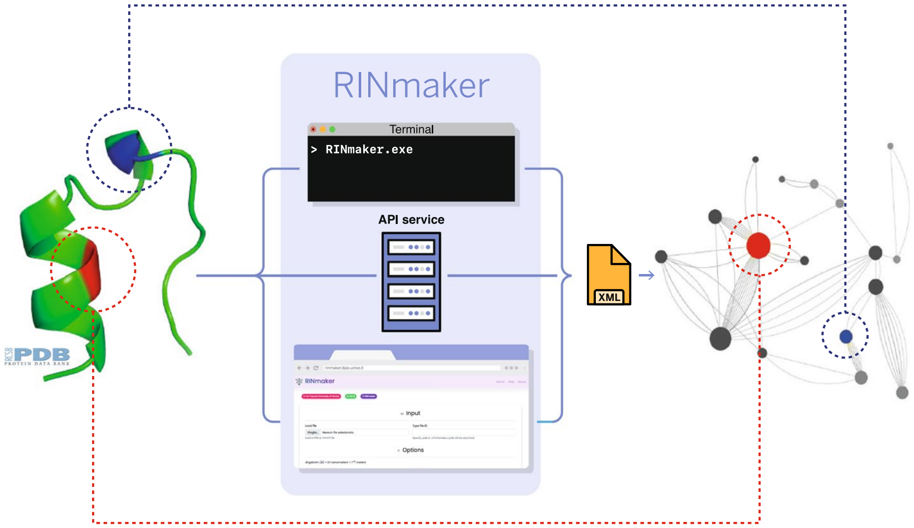
A second goal was to ensure the RIN*maker* correctness. To this aim, the use of advanced preprocessor macros and C++11 lambdas played an important role in designing a programmable and versatile test suite.

In this section we present the RIN*maker* workflow and main features and describe how we addressed the efficiency and correctness issues.

RIN*maker* workflow and main features

The RIN*maker* workflow is illustrated in Fig. 1. The core of RIN*maker* consists of one executable that receives in input the 3D structure of a protein and produces in output a file representing the corresponding RIN. RIN*maker* is offered as an application supporting multiple user interfaces:

- A console application from the command line with a wide range of possible arguments allowing the user to fine-tune bond parameters, such as distance threshold, orientational angles, etc. All such parameters have default values that are reasonable in most cases.
- An HTTP-based API available for third-party applications requiring RIN*maker* as a service. Clients can upload an input file through an HTTP post to the API where the content of the input file appears in the body in JSON format and download the resulting RIN via marshalling and unmarshalling in JSON format.



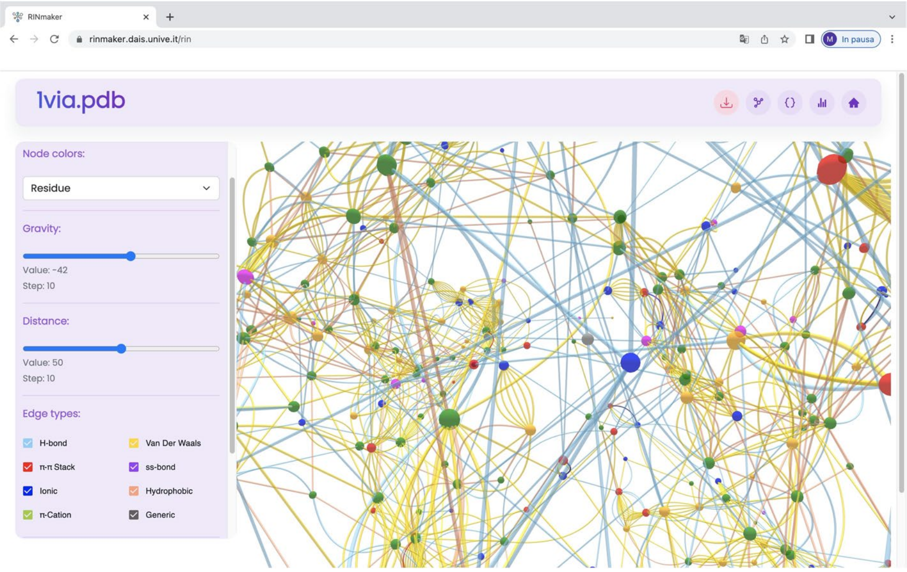
```
> RINmaker.exe
```

Fig. 1 The RIN*maker* flow: The 3D structure of a protein (left) is mapped into a RIN network (right) via different possibilities (command-line, web API and web GUI). As a specific example, the red and blue amino acids highlighted in the protein structure on the left are mapped into the corresponding nodes on the right

- A GUI web application running RIN*maker* in the background on the server side. The user can search an input file in the PDB databank or upload a local file, select the desired parameters for the non-covalent bonds and visualize the resulting RIN through a real-time rendered 2D or 3D representation, see an example in Fig. 2. Smooth motion, rotation and zoom are available, delivering a 60+ fps experience on an average machine (an Intel i5 10th generation processor with an integrated GPU). This helps the user exploring the RIN, observing bonds and retrieving relevant information on the network structure, the details of nodes and edges and some overall statistics.

The main features of RIN*maker* are the following:

- It supports both mmCIF and PDB as input file formats.
- The output RIN is by default generated in GraphML format [14]. A special CLI option allows for the generation of two CSV output files, one for the nodes and one of the edges of the RIN, in place of a single GraphML file.
- It allows for easily processing snapshots in MD trajectories. The entire MD trajectory, stored into a single PDB file, can be handled by RINmaker to output a single RIN for each individual snapshot. Indeed, RIN*maker* supports multiple models within a single input file. A special command line flag makes RIN*maker* perform a separate run for each model, outputting all the RINs as distinct files into a single directory.
- It determines the C_α contact-map as well as the following non-covalent interactions: H-bonds, Van der Waals interactions, ionic bridges, π-π stacks, π-cation interactions. We included also "hydrophobic interactions" in view of their frequent reference in the literature [15], even if, strictly speaking, such bond does not exist as it is the effective combination of several different contributions, partially of entropic origin.



Node colors: Residue

Gravity:
Value: -42
Step: 10

Distance:
Value: 50
Step: 10

Edge types:
☑ H-bond ☑ Van Der Waals
☑ π-π Stack ☑ ss-bond
☑ Ionic ☑ Hydrophobic
☑ π-Cation ☑ Generic

Fig. 2 The RIN*maker* user interface: RIN 3D visualization

- It provides a set of attributes for the RIN nodes and edges, such as the type of bond, the involved atoms, their distance, orientation and energy. For each bond, energy is calculated as a function of the distance between atoms and their relative orientation in space, rather than using approximated constant values available in the literature. This allows for a more accurate evaluation of the interactions. Specific formulas and tables for energy calculation are available in the Additional file 1.

Computational efficiency

Computational efficiency of RIN*maker* deserves a detailed discussion. Searching for non-covalent bonds among all the atoms of a protein would require $O(n^2)$ operations, n being the number of atoms. As a consequence, the choice of an efficient data structure was mandatory. We resorted to the use of kD-Trees, as they allow for an efficient representation of k-dimensional spatial information and a low cost proximity search algorithm [16]. We implemented a kD-Tree library in C++ for achieving maximum performance while retaining a strongly typed approach: dimension k is represented by an integer template argument, in the style of template meta-programming [17], which allows for inlining of constant expressions and other compile-time optimizations for $k = 3$, since the protein structure is represented in a 3D space [18]. The program builds a balanced kD-Tree representing the protein 3D structure in $O(n \log^2 n)$ time, with n being the number of atoms in the input file. With such data structure, the search for non-covalent bonds becomes the search for neighbouring nodes in the balanced kD-tree, which runs in $O(\log n)$ time on average and in $O(n)$ time in the worst case. The advantage of adopting the kD-tree data structure and the related proximity search algorithm clearly emerges when assessing the RIN*maker* performance, as reported in the Results and Discussion section.

Test suite

RIN*maker* is provided with a custom integration test system based on the Google Test Framework [19]. The program can be compiled in *test* mode to run a series of tests. Tests are on two levels: tests of single code portions and functions; and tests of the whole program over a given input file. Code-level tests are modular and reusable thanks to advanced macro usage within C++, allowing for easy creation of new tests. A series of input sample PDB files specifically designed for testing the program behaviour on the various bonds are included with the RIN*maker* distribution. Moreover, a detailed description of all molecules included in tests along with the expected results is included in the Additional file 1.

RIN*maker* can be built either in *application mode* for producing the normal executable, or in *test mode* for producing a special executable aimed at testing the program. The *test mode* executable can be obtained by compiling the program with a specific compile-time flag, or, more simply, by running *cmake* with the *Test* target.

The *test mode* executable simulates multiple runs of the normal *application mode* execution. This allows for performing a series of tests in a single run of the program. Each test is performed with a specific combination of command line arguments. Moreover, to speed-up the testing process, each output RIN is kept in memory rather than being dumped to an output file, thus avoiding the re-parsing of the output files when comparing the output bonds against the expected ones.

The main advantages of such test system are scalability and maintainability. Writing new test cases with new molecules is straightforward as well as adapting existing ones to new scenarios.

Results and discussion

In this section we present a case study where RIN*maker* is employed for the analysis of a molecular dynamic trajectory. We present moreover some performance evaluation tests that confirm the efficiency of RIN*maker*.

Using RIN*maker* for data analysis of a molecular dynamics trajectory

One of the added values of mapping a 3D structure into a network stems from the various probes that can be used to identify emerging patterns that are not immediately visible from the original structure. We provide here an interesting example of this idea using RIN*maker*, thus showing the efficiency of the tool within this framework.

Consider Trp-cage (file 1L2Y.pdb in the Protein Databank [20]), a small protein made of 20 aminoacids. Since it is a fast folding protein (folding time in the nsec time domain), it has been thoroughly studied and its structure and properties are rather well understood [21, 22].

The Trp-cage native 3D structure is depicted in Fig. 3 and contains a short α-helix in residues 2–9, a 310-helix in residues 11–14, and a C-terminal polyproline II helix to pack against the central tryptophan (Trp-6) [21]. The latter residue appears to be involved in the formation/breaking of side-chain bonds during the various folding-unfolding processes, because of its preferential position in the core region [22]. Using molecular dynamics in explicit water TIP3P [23] with the CHARMM36 force-field [24], we first equilibrated the Trp-cage protein for 45 nsec in NPT ensemble with pressure equal to 1 atm kept constant with a Monte Carlo barostat [25]. Then a NVT production run of additional 45 nsec was performed at temperature 480K and at temperature 330 K. In all cases it was used a Langevin Integrator with timestep set to 2 psec. All simulations were performed using the OpenMM [26] python package.

From [21], it is known that with the use of the CHARMM36 force field, the ground-state (folded state) of the protein is well maintained at 330K, and that the melting point corresponding to denaturation of the protein, appears around 440K. Other studies show that the folded state is preserved exclusively by Trp at position 6 in the protein in the

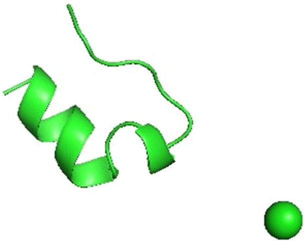


Fig. 3 Native state of the Trp-Cage miniprotein with a Cl^{-} as counter ion (Model 1 in the 1L2Y.pdb file)

folded-state, while a denaturation involves a disruption of Trp's noncovalent interactions with other nearby residues. Thus, the choice of temperatures at 330K and 480K were selected to emphasize these two different states of the protein in the free energy landscape.

During the production run, a number of reaction coordinates can be computed to highlight the behaviour of the protein at that specific temperature. In this case, we selected the radius of gyration that signals the onset of the folding of the protein, and the betweenness centrality computed on the RIN obtained via the use of RIN*maker*. The betweenness centrality is the ratio between the number of shortest paths passing through a specific node (i.e. a residue) and the number of total shortest paths in the network. The betweenness centrality is known to be an important indicator of the residue function of a protein upon a mutation of a given residue [27], and we show that this information is indeed captured by the RIN topology via our RIN*maker* tool.

For each snapshot of the trajectory (every 30ps), the radius of gyration and the RIN were computed. The latter was used to further compute the betweenness centrality using the Networkx [28] python package. Both these quantities were then accumulated in histograms that were then used for a free energy landscape analysis following standard prescriptions [21].

Figure 4 displays the betweenness centrality of each of the 20 residues of the Trp-Cage as a function of the simulation time $t/\Delta t$ (units of $\Delta t = 30$ psec) for two temperatures: (top panel) T = 330 K below the folding temperature ($T \approx 400\,K$) and (bottom panel) T = 480 K above it. The primary role played by Trp6 at T = 330 K can be clearly appreciated at a glance, and it agrees with past results [29] as further elaborated below. Above the folding temperature, precisely at T = 480 K, this prominence is lost at an early stage of the simulation and other residues, notably Arg16 and Gly11, begin to display their importance. This can be rationalized with the breaking and reforming of bonds as well as the stabilization of another local minimun. This process is captured by the free energy landscape calculations as illustrated in Fig. 5.

Figure 5 reports a contour plot of the free energy landscape as a function of the radius of gyration and the betweenness centrality at two different temperatures: $T = 330\,K$ (left panel) and $T = 480\,K$ (right panel). As the folding temperature is expected to be around 400K [30], the first temperature is below the folding temperature whereas the second is above.

In agreement with previous studies [31], we notice that at T = 330 K there is a well defined minimum (A) corresponding to the native state. This minimum becomes unstable at T = 480 K, above the folding temperature, due to a partial denaturation of one particular alpha-helix, thus driving the system toward another minimum (B). It should be emphasized that while this scenario is fully consistent with previous studies [31], specific details are not in view of the different force fields and different reaction coordinates employed. In this respect, the use of the betweenness centrality, and in turn of RIN*maker*, has proven very useful as it provides a much cleaner free energy landscape compared with previous one, thus highlighting the importance of RIN*maker* as a computational tool that can be embedded into a computational pipeline.

The results reported in Fig. 4 have underpinned the fundamental role of Trp6 residue that is unveiled by the analysis of the betweenness centrality as obtained via

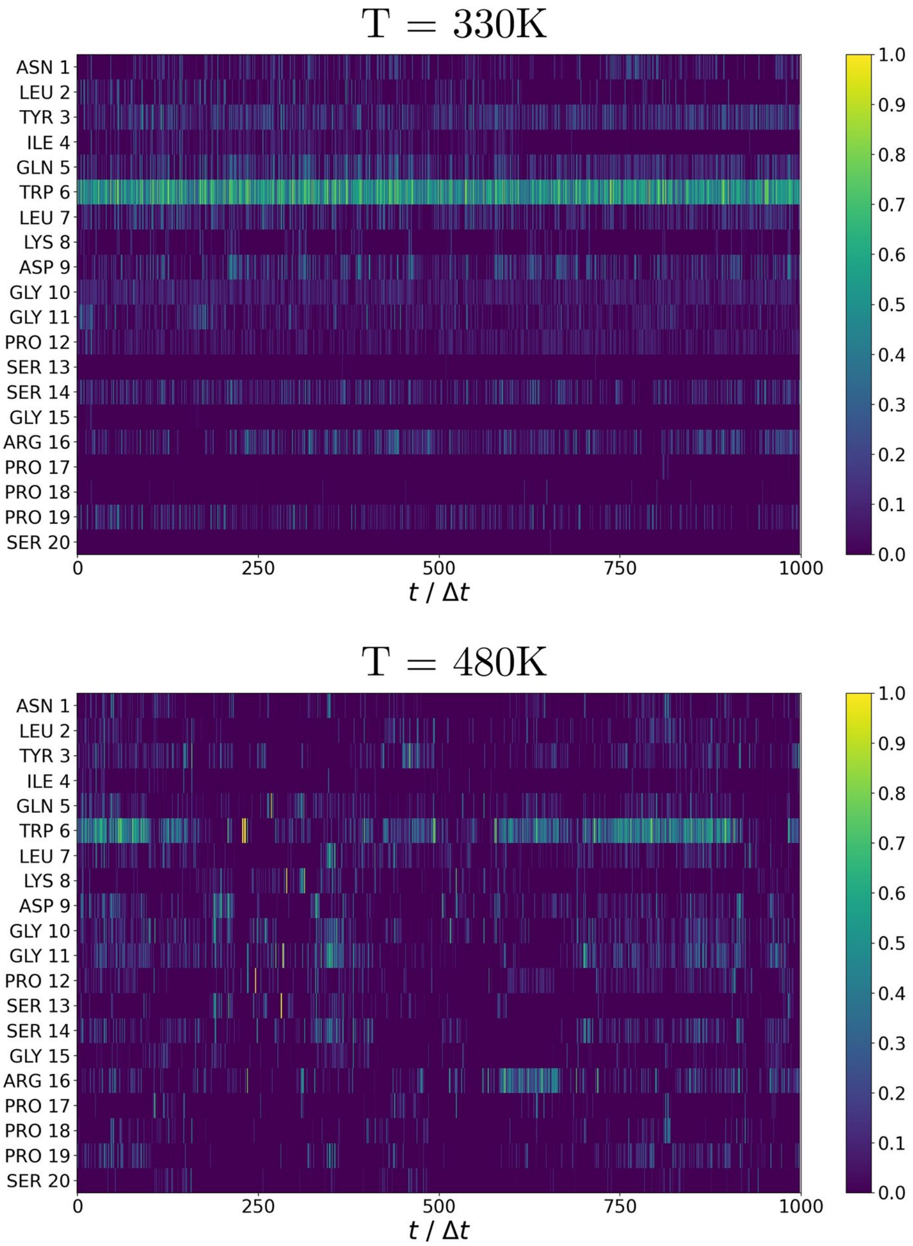


Fig. 4 Betweenness centrality of the Trp-Cage residues as a function of the simulation time $t/\Delta t$ ($\Delta t = 30$ psec) during the trajectory at temperatures above and below the folding temperature ($\approx 400K$). Top panel $T = 330\,K$; Bottom panel $T = 480\,K$

RIN*maker*. Further insights can be obtained by plotting the actual value of betweenness centrality during the time evolution, as illustrated in Fig. 6. At 330 K the betweenness centrality remains stable, indicating an equlibrium localized in minimum A of Fig. 5. At 480K, the betweenness centrality displays a significant oscillatory behavior indicating a coexistence with another minimum (indicated as B in Fig. 5) throughout the entire trajectory. This can be ascribed to the salt-bridges and side-chain H-Bonds

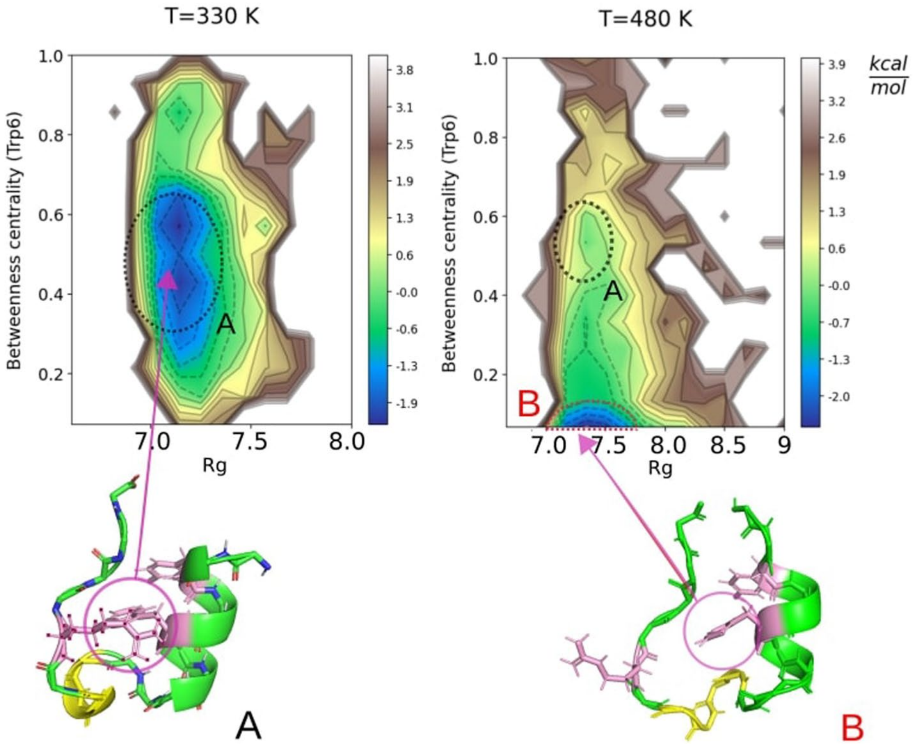


Fig. 5 Free energy landscape of Trp-Cage as a function of the radius of gyration and of the betweenness centrality at temperatures (left) T = 330 K and (right) T = 480 K. Snapshots of the conformation corresponding to stable local minima are given in **A** for T = 330 K and in **B** for T = 480 K with highlighted the Gly11, Arg16 and Tpr6 residues involved in the transition between one minimum to the other. In **B** it is also possible to observe the denaturated α-helix highlighted in yellow

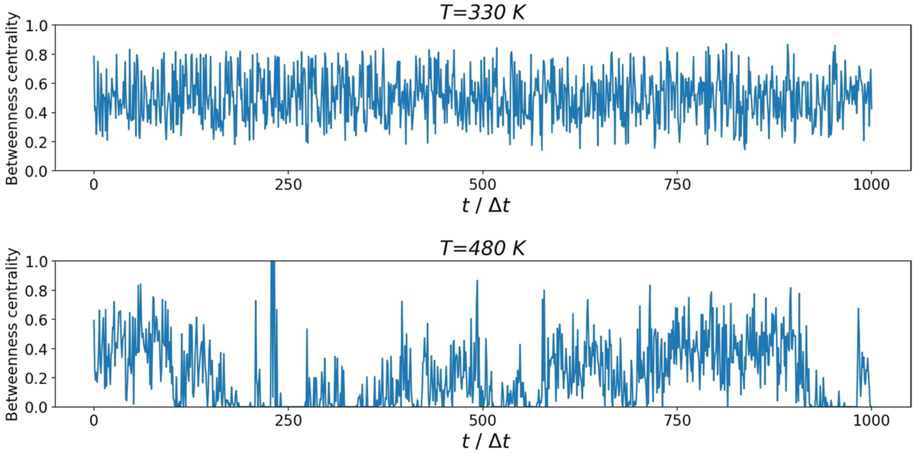


Fig. 6 Betweenness centrality of residue Trp6 in the Trp-Cage trajectory obtained from RIN*maker*, at the two different temperatures (T = 330 K and T = 480 K)

that this residue forms/breaks with the neighbour side-chains which shift the protein conformation between A and B [21]. The various spikes that can be seen in the process are probably due to the solvent noise and the transient interactions found by RIN*maker* when building the RIN.

Performance evaluation

In this section we illustrate the performance tests that have been designed and performed for RIN*maker*. All tests were executed on a mid-end machine, namely a ThinkPad E590 laptop equipped with Intel i7-8565U CPU, 16GB of RAM and a 500GB SSD.

The first test aimed at evaluating the running time of the software for a wide set of proteins. We considered the TOP2018 [32] database and created a script to run RIN*maker* for all proteins stored in the database, which contains 13,678 entries (pdb files) corresponding to all single chains of the considered proteins. All the created RINs were obtained by using the default values for all input parameters. We measured that the average time to build the RIN for each entry was approximately 66 ms, for a total of 15 min for the entire database. We also computed the frequency per bond for the obtained RINs, which is reported in Table 1. The reported frequency values are in agreement with the general knowledge in the literature.

The second test was a performance comparison between RIN*maker*, and the most similar tools in the literature, namely RING 3.0 [5] and PDBe-Arpeggio [10]. Note that, since RING 3.0 source code is unavailable, the test has been performed using the executables of the command-line version of the tool. For PDBe-Arpeggio we used the available Python scripts. While RIN*maker* and RING 3.0 are very similar in their general structure, PDBe-Arpeggio significantly departs from both on them on the count that it does not allow users to customize bond parameters (angles, distances, etc), and it heavily relies on external tools such as Open Babel [33].

The test aimed at evaluating the performance of the three tools for a small set of proteins available in the PDB databank and having an increasing number of amino acids. Table 2 shows the considered set of proteins and reports, for each of them, the *user CPU time* needed by all the three tools for creating their corresponding RIN. Each reported value has been averaged over ten runs values. The addition of hydrogen atoms is an optional operation in RIN*maker* and RING 3.0, while it is always included in PDBe-Arpeggio. Hence, for RIN*maker* and RING 3.0, the user CPU time is reported for both cases (i.e. with and without hydrogen addition). We remark that for each protein and run, RIN*maker* and RING 3.0 received the same input parameters. However, it is worth mentioning that, although the two tools operated at the same conditions, they do not perform exactly the same activities. In particular:

- RING 3.0 run the DSSP algorithm for secondary structure prediction [34], which is not included in RIN*maker*;

Table 1 Bonds frequency (%) calculated for the RINs of the TOP2018 database

Type of bond	Frequency (%)
VdW	57.5
H-Bond	25.5
Hydrophobic	16.2
Ionic	0.5
π–π stacking	0.2
π-cation	0.1

Table 2 Performance tests on an Intel i7-8565U CPU: RIN*maker* versus RING 3.0 versus PDBe-Arpeggio

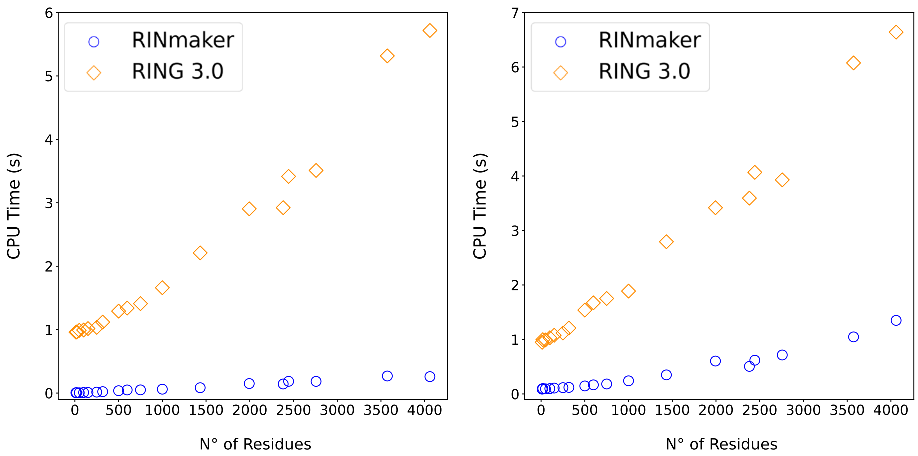
PDB name	Num. of residues	CPU time (s) with hydrogen addition			CPU time (s) without hydrogen addition	
		RIN*maker*	RING 3.0	PDBe-Arpeggio	RIN*maker*	RING 3.0
1al1	12	0.0885	0.948	1.22	0.005	0.961
1l2y	20	0.0987	0.996	1.51	0.005	0.963
1bk8	50	0.0934	0.991	8.17	0.004	0.989
1eod	100	0.0975	1.031	6.64	0.009	0.994
1hhq	150	0.108	1.077	15.26	0.009	1.017
1ab5	250	0.115	1.118	63.25	0.015	1.036
1bge	318	0.120	1.209	131.50	0.022	1.119
1b1y	500	0.147	1.539	185.03	0.037	1.292
2vji	598	0.167	1.672	436.09	0.049	1.340
1miq	750	0.183	1.752	447.50	0.050	1.410
1a3w	1000	0.241	1.888	789.72	0.061	1.661
6j8j	1433	0.350	2.793	2245.17	0.084	2.210
1h1l	1994	0.604	3.417	7771.14	0.150	2.905
1i84	2382	0.506	3.594	> 8000.00	0.143	2.923
3dwk	2444	0.619	4.067	> 8000.00	0.185	3.416
3aae	2758	0.715	3.929	> 8000.00	0.184	3.510
2qmi	3573	1.047	6.075	> 8000.00	0.269	5.316
7fd2	4060	1.35	6.640	> 8000.00	0.259	5.717

- Both tools report the H-bonds, VdW bonds, Ionic bonds, $\pi - \pi$ stacks and π—Cations. However, RIN*maker* reports also the hydrophobic bonds, which are not included in RING 3.0.

By contrast, in PDBe-Arpeggio everything is pre-defined thus lacking on any possibility of a customized use, as already remarked.

Looking at Table 2 it is clear that RIN*maker* outperforms PDBe-Arpeggio and RING 3.0 in all tests by at least one order of magnitude in most cases. It is also interesting to point out that the low user CPU times employed by RIN*maker* when dealing with small molecules imply that the program startup time, consisting of parsing the input and setting up data structures, is negligible compared to the processing time. This feature makes RIN*maker* ideal to be embedded into other tools, scripts or pipelines.

A different view of the comparison between RIN*maker* and RING 3.0 is shown in Fig. 7. It considers the user CPU time distribution w.r.t. the number of residues in the input files and shows the results of both tools in the two cases, namely with and without hydrogen addition. The results highlights even more the different performances of the two tools, RIN*maker* being the most efficient one. Note that the RIN*maker* CPU times grow very slowly w.r.t. the number of residues (it is a linear growth in case no hydrogen atoms are added), making it suitable for proteins of any size.



(a) No hydrogen atoms added (b) hydrogen atoms added

Fig. 7 Performance tests: CPU time as a function of the number of residues. RIN*maker* (blue) vs. RING 3.0 (orange)

Conclusion

RIN*maker* has been designed to be a flexible, reliable and fast tool to build and visualize RINs. Future developments of RIN*maker* include the extension of its web interface with additional visualization features as well as detailed analyses on the obtained RINs. In the current version, the web interface allows the 2D and 3D RIN visualization and offers some basic statistics. We would like to include the topological analysis of the obtained RIN and specific analyses associated to RINs determined on a MD trajectory, which could reveal interesting information on the simulated protein. Moreover, RIN*maker* is going to be linked to XRmol [35, 36], a web-based protein structure viewer that provides interactive 3D display of proteins and nucleic acids in different environments, including desktop computers and devices supporting Augmented Reality. The combination of the two tools, possible thanks to the web API available for RIN*maker*, will allow for depicting all non-covalent bonds calculated by RIN*maker* directly in the protein's 3D structure visualized by XRmol.

The importance of having an efficient tool to construct RINs should not be underestimated. RINs provide a graph theoretical representation of a protein (or a protein domain) that captures the crucial complexity of the biomolecule at a glance and translates it into a set of quantitative indicators that are crucial to understand its function. A major challenge is to be able to achieve this dimensional reduction without loosing crucial information of its high-dimensional conformational space.

Efforts along these lines have recently been proposed [37] whose findings are complementary to ours thus providing further support on the impact that the tool proposed in this study may have in the near future.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s12859-023-05466-y.

Additional file 1. This document contains detailed information about non-covalent bonds and test cases.

Acknowledgements
We thank Giulio Marcolin and Luca Lagunas for their preliminary contribution to the software. We thank Giulia Brolese for improving the RIN*maker* flow picture. The use of the SCSCF cluster at Ca' Foscari University of Venice. is gratefully acknowledged. We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support.

Author contributions
A.S., A.G., M.S.: software conceptualization. A.G., C.D.F, J.M.: non-covalent bonds rules specification. A.S., A.P., D.P., L.F.: command line program implementation. A.S., F.V.: web application and web API implementation. A.S., D.P., J.M.: tests cases development and test suite design and implementation. A.S., A.G., C.D.F., J.M., M.S.: paper and Supplementary Material writing. All authors revised the paper and the Supplementary Material. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
Data analyzed in this paper are available at the PDB Data Bank [20] and at the TOP2018 database [32]. Both of them are public databases. **Availability and requirements**: Program name: RIN*maker*; Web UI: https://rinmaker.dais.unive.it/; Programming language: C++; Other requirements: none. License: The code is open source and available at `github.com/RINmaker` under the GNU Affero General Public License. Downloads: Binaries are available for multiple operating systems at `github.com/RINmaker/RINmaker/releases`. Contact: `rinmaker@unive.it`. Any restrictions to use by non-academics: none.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 2 June 2023 Accepted: 4 September 2023
Published online: 11 September 2023

References

1. Jumper J, Evans R, Pritzel A, et al. Highly accurate protein structure prediction with alphafold. Nature. 2021;596(7873):583–9. https://doi.org/10.1038/s41586-021-03819-2.
2. Tunyasuvunakool K, Adler J, Wu Z. Highly accurate protein structure prediction for the human proteome. Nature. 2021;596:590–6.
3. Baek MFD, Anishchenko I, et al. Accurate prediction of protein structures and interactions using a three-track neural network. Science. 2021;373(6557):871–6. https://doi.org/10.1126/science.abj8754.
4. Liang Z, Verkhivker GM, Hu G. Integration of network models and evolutionary analysis into high-throughput modeling of protein dynamics and allosteric regulation: theory, tools and applications. Brief Bioinform. 2019;21(3):815–35. https://doi.org/10.1093/bib/bbz029.
5. Clementel D, Del Conte A, Monzon AM, Camagni G, Minervini G, Piovesan D, Tosatto SCE. RING 3.0: fast generation of probabilistic residue interaction networks from structural ensembles. Nucleic Acids Res. 2022;50(W1):651–6.
6. Martin AJ, Vidotto M, Boscariol F, Di Domenico T, Walsh I, Tosatto SC. RING: networking interacting residues, evolutionary information and energetics in protein structure. Bioinformatics. 2011;27(14):66.
7. Piovesan D, Minervini G, Tosatto S. The RING 2.0 web server for high quality residue interaction networks. Nucleic Acids Res. 2016;44(W1):66.
8. Jubb HC, Higueruelo AP, Ochoa-Montaño B, et al. Arpeggio: a web server for calculating and visualising interatomic interactions in protein structures. J Mol Biol. 2017;429(3):365–71. https://doi.org/10.1016/j.jmb.2016.12.004.
9. Contreras-Riquelme S, Garate JA, Perez-Acle T, Martin AJM. RIP-MD: a tool to study residue interaction networks in protein molecular dynamics. PeerJ. 2018;6:5998. https://doi.org/10.7717/peerj.5998.
10. PDBe-Arpeggio. 2023. https://github.com/PDBeurope/arpeggio.
11. Toffano AA, Chiarot G, Zamuner S, Marchi M, Salvi E, Waxman SG, Faber CG, Lauria G, Giacometti A, Simeoni M. Computational pipeline to probe nav1. 7 gain-of-function variants in neuropathic painful syndromes. Sci Rep. 2020;10(1):17930.
12. Alexandrescu A. Modern C++ design: generic programming and design patterns applied. Boston: Addison-Wesley; 2001.
13. Milton, S., Schmidt, H. Dynamic dispatch in object-oriented languages. Technical report, The Australian National University; 1994
14. Brandes U, Eiglsperger M, Herman I, et al. Graphml progress report: structural layer proposal. In: Springer-Verlag (ed.) Proceedings of the 9th international symposium graph drawing (GD'01) (LNCS 2265); 2002. p. 501–512.

15. Onofrio A, Parisi G, Punzi G, Todisco S, Di Noia MA, Bossis F, Turi A, De Grassi A, Pierri CL. Distance-dependent hydrophobic–hydrophobic contacts in protein folding simulations. Phys Chem Chem Phys. 2014;16:18907–17. https://doi.org/10.1039/C4CP01131G.
16. Chen Y, Zhou L, Tang Y, et al. Fast neighbor search by using revised k–d tree. Inf Sci. 2019;472:145–62. https://doi.org/10.1016/j.ins.2018.09.012.
17. Abrahams D, Gurtovoy A. C++ template metaprogramming: concepts, tools, and techniques from boost and beyond. 1st ed. London: Pearson Education; 2004.
18. Andrist B, Sehr V. C++ high performance: master the art of optimizing the functioning of your C++ code. 2nd ed. Birmingham: Packt Publishing Ltd; 2020.
19. Inc. G. Google Test Framework. 2022. http://google.github.io/googletest/. Accessed 7 Dec 2022.
20. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE. The protein data bank. Nucleic Acids Res. 2000;28:66. https://doi.org/10.1093/nar/28.1.235.
21. Zhou R. Trp-cage: folding free energy landscape in explicit water. Proc Natl Acad Sci. 2003;100(23):13280–5. https://doi.org/10.1073/pnas.2233312100.
22. Lee IH, Kim SY. Dynamic folding pathway models of the trp-cage protein. BioMed Res Int. 2013;2013:66. https://doi.org/10.1155/2013/973867.
23. Jorgensen WL, Chandrasekhar J, Madura JD, Impey RW, Klein ML. Comparison of simple potential functions for simulating liquid water. J Chem Phys. 1983;79(2):926–35. https://doi.org/10.1063/1.445869.
24. Huang J, MacKerell AD Jr. CHARMM36 all-atom additive protein force field: validation based on comparison to NMR data. J Comput Chem. 2013;34(25):2135–45.
25. Andersen HC. Molecular dynamics simulations at constant pressure and/or temperature. J Chem Phys. 2008;72(4):2384–93. https://doi.org/10.1063/1.439486.
26. Eastman P, Swails J, Chodera JD, McGibbon RT, Zhao Y, Beauchamp KA, Wang L-P, Simmonett AC, Harrigan MP, Stern CD, Wiewiora RP, Brooks BR, Pande VS. Openmm 7: rapid development of high performance algorithms for molecular dynamics. PLOS Comput Biol. 2017;13(7):1–17. https://doi.org/10.1371/journal.pcbi.1005659.
27. Foutch D, Pham B, Shen T. Protein conformational switch discerned via network centrality properties. Comput Struct Biotechnol J. 2021;19:3599–608. https://doi.org/10.1016/j.csbj.2021.06.004.
28. Hagberg A, Swart P, Chult SD. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos; 2008.
29. Kannan S, Zacharias M. Role of tryptophan side chain dynamics on the trp-cage mini-protein folding studied by molecular dynamics simulations. PLoS ONE. 2014;9(2):1–12. https://doi.org/10.1371/journal.pone.0088383.
30. Wu X, Yang G, Zu Y, Fu Y, Zhou L, Yuan X. Molecular dynamics characterisations of the trp-cage folding mechanisms: in the absence and presence of water solvents. Mol Simul. 2012;38(2):161–71. https://doi.org/10.1080/08927022.2011.610795.
31. Paschek D, Hempel S, García AE. Computing the stability diagram of the trp-cage miniprotein. Proc Natl Acad Sci. 2008;105(46):17754–9. https://doi.org/10.1073/pnas.0804775105.
32. Williams C, Richardson D, Richardson J. High quality protein residues: top2018 mainchain- filtered residues. Zenodo. 2021. https://doi.org/10.5281/zenodo.5777651.
33. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR. Open babel: an open chemical toolbox. J Cheminform. 2011;3(1):33. https://doi.org/10.1186/1758-2946-3-33.
34. Kabsch W, Sander C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers. 1983;22(12):2577–637. https://doi.org/10.1002/bip.360221211.
35. XRmol. 2023. https://xrmol.dais.unive.it.
36. Corazza S, Pittarello F, Simeoni M. 3d molecules visualization with XRmol: an AR web tool for mobile devices. In: Springer (ed) 20th EuroXR international conference—EuroXR 2023, lecture notes in computer science. Accepted for publication; 2023.
37. Franke L, Peter C. Visualizing the residue interaction landscape of proteins by temporal network embedding. J Chem Theor Comput. 2023;19(10):2985–95. https://doi.org/10.1021/acs.jctc.2c01228.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.