AutoDock Vina 1.2.0: new docking methods, expanded force field, and Python bindings

Jerome Eberhardt[1,a,✉], **Diogo Santos-Martins**[1,a], **Andreas F. Tillack**[a], **and Stefano Forli**[a,✉]

[1]These authors contributed equally to this work. [a] Department of Integrative Structural and Computational Biology, The Scripps Research Institute, La Jolla, California, USA

AutoDock Vina is arguably one of the fastest and most widely used open-source docking engines. However, compared to other docking engines in the AutoDock Suite, it lacks features that support modeling of specific systems such as macrocycles or modeling water explicitly. Here, we describe the implementation of these functionality in AutoDock Vina 1.2.0. Additionally, AutoDock Vina 1.2.0 supports the AutoDock4.2 scoring function, simultaneous docking of multiple ligands, and a batch mode for docking a large number of ligands. Furthermore, we implemented Python bindings to facilitate scripting and the development of docking workflows. This work is an effort toward the unification of the features of the AutoDock4 and AutoDock Vina docking engines. The source code is available at https://github.com/ccsb-scripps/AutoDock-Vina

docking | autodock | vina | drug discovery | virtual screening
Correspondence: ***forli@scripps.edu***

Introduction

AutoDock Vina (Vina)[1] is one of the docking engines in the AutoDock Suite[2], together with AutoDock4 (AD4)[3], AutoDockGPU[4], AutoDockFR[5], and AutoDock-CrankPep[6]. Vina is arguably among the most widely used docking engines, probably because of its ease of use and speed, when compared to the other docking engines in the suite and elsewhere, as well as being open source.

Research groups around the world have modified and built upon the Vina source code, improving the search algorithm (QuickVina2[7]), made the interface more user friendly and allow modification of scoring terms through the user interface (Smina[8]), and improved the scoring function for carbohydrate docking (Vina-Carb[9]), halogen bonds (VinaXB[10]), as well as ranking and scoring (Vinardo[11]).

Beside these valuable developments, there are still several methods within the AutoDock Suite that are not available in Vina because they have been implemented specifically for either the scoring function or the docking engine in AD4. Examples of such methods include docking with macrocyclic flexibility[12], specialized metal coordination models[13], modeling of explicit waters[14], coarse-grained ligand models[15], and ligand irreversible binding[16]. Despite being a less efficient docking engine, AD4 allows the user to modify a large number of docking parameters, providing direct access to some of the engine internals, making it well-suited for the development of new docking methods. Conversely, the Vina interface is highly specialized and optimized, and one of its hallmarks is the very limited amount of user input necessary to perform a docking. In turn, this makes it impossible to implement additional functionality without significant changes in the source code.

The usefulness of such specialized methods is hindered by the poor search efficiency of AD4. In fact, AD4 can be up to 100x slower than Vina[1], depending on the search complexity. The large performance difference is due to the better search algorithm used in Vina, a Monte-Carlo (MC) iterated search combined with the BFGS[17] gradient-based optimizer. In comparison with the Lamarckian Genetic Algorithm (LGA) and Solis-Wets local search of AD4[3], the search efficiency of Vina leads to better docking results with fewer scoring function evaluations.

We implemented some of the specialized AD4 features in the Vina source code, enabling their use of the powerful MC/BFGS search algorithm. Then, we further extended the Vina engine enabling simultaneous docking of multiple ligands, and adding Python bindings to facilitate programmatic access to the docking engine functionalities.

Scoring function extensions and improvements

AutoDock4.2 scoring function. One major improvement is the availability of the AD4 scoring function in Vina. This allows users to access it using the Vina MC-based search algorithm and explore with equal efficiency its energy landscape. This will likely facilitate large-scale consensus docking virtual screening campaigns[18,19].

The AD4 and Vina scoring functions are quite different. AD4 uses a physics-based[3] model with van der Waals, electrostatic, directional hydrogen-bond potentials derived from an early version of the AMBER force field[3,20], a pairwise-additive desolvation term based on partial charges, and a simple conformational entropy penalty. On the other hand, Vina lacks electrostatics and solvation[1], and consists of a van der Waals-like potential (defined by a combination of a repulsion term and two attractive gaussians), a non-directional hydrogen-bond term, a hydrophobic term, and a conformational entropy penalty.

Performance-wise, the average time required to perform energy evaluations with the AD4 scoring function is nearly 3x larger than with the Vina scoring function. This is due to the presence of additional electrostatic and desolvation maps that need to be interpolated for each movable atom.

Grid map files support. Both AD4 and Vina calculate intermolecular interactions by performing trilinear interpolations of grid maps pre-calculated on the target structure. Vina also uses the target structure to perform a post-processing minimization of the docked poses. In AD4, maps are pre-calculated using a separate program (AutoGrid[2]) prior to docking and loaded at runtime, while Vina calculates them on-the-fly prior to running the MC search. The availability to accessible grid map files generated by AutoGrid provided the foundations for a number of specialized methods, such as the zinc-coordination potentials in the AutoDock4$_{\mathrm{Zn}}$ force field[13], biasing docking using information from molecular dynamics simulations in AutoDock-Bias[21], and the integration of Grid Inhomogenous Solvation Theory (GIST)[22–24] in AutoDock-GIST[25].

In AutoDock Vina 1.2.0 we added the support to optionally load external grid map files, enabling all these methods in both the AD4 and Vina scoring functions. These methods can be applied by following the existing protocols to prepare target structures and the corresponding grid maps, then replace the AutoDock4 binary with the new version of Vina. The availability of reading and writing maps facilitates the development of similar methods for the Vina scoring function.

New atom types. We extended both the Vina and AD4 scoring functions to support new atom types for atoms and pseudo-atoms as required by the hydrated docking method and the macrocycle sampling methods. These atom types are implemented in the source code. Additionally, we also added parameters for silicon to address user requests for better support to the chemical space covered in public repositories such as the ZINC database[26].

New docking methods

We increased the number of the docking methods available in Vina leveraging the availability of new atom types, the possibility of specifying grid map files to be used during docking, and by extending the existing code.

Simultaneous multiple ligand docking. Vina is now able to dock simultaneously multiple ligands. This functionality may find application in fragment based drug design, where small molecules that bind the same target can be grown or combined into larger compounds with potentially better affinity.

The protein PDEδ in complex with two inhibitors (PDB 5x72)[27] was used as a proof of concept to test the ability of Vina to successfully dock multiple ligands simultaneously. The two inhibitors in this structure are stereoisomers, and only the R-isomer is able to bind in a specific region of the pocket, while both the R- and S-isomers can bind to the second location. Using the Vina scoring function, the best set of poses (top 1) shows an excellent overlap with the crystallographic coordinates for one of the isomers, and reasonable overlap with the electron density for the other isomer, which shows some degree of ambiguity (Fig. 1 A). Using the AutoDock4 scoring function, similar performance in overlapping the crystallographic poses is found, but only when considering the first two sets of poses (top 2).

Hydrated docking. The hydrated docking protocol[14] has been developed to model waters directly involved in the ligand-receptor interaction. The method is based on docking ligands explicitly hydrated with spherical waters, and can be used to predict the position and the role (i.e., bridging or displaced) of individual water molecules and generally improve ligand pose predictions. Waters are represented by a single atom of type W, and are added to the ligand molecule at the end of each hydrogen bond vector. During docking, W atoms move along with the ligand, do not contribute to intramolecular interactions, and are allowed to overlap with the protein. In fact, when that happens, a water is considered displaced (i.e., removed from the system), and an energy reward is added to the ligand score to reflect the entropy gain resulting from releasing the water to bulk solvent. Following the standard hydrated docking protocol[14], the W map, which represents water-receptor interactions is obtained by combining the oxygen-acceptor (OA) and hydrogen donor (HD) maps of the AD force field.

	Pose rank				
PDB ID	1	2	3	4	5
4ykq	**0.45**	**0.43**	2.37	6.16	4.01
4ykt	9.23	8.24	8.50	3.40	2.86
4yku	6.02	**1.14**	**0.67**	6.04	5.89
4ykx	**0.98**	**0.95**	6.24	**1.79**	**1.75**
4ykw	6.27	**0.64**	6.50	**1.32**	**1.34**
4ykz	5.30	**1.68**	**0.77**	5.30	6.00

Table 1. RMSD of 6 ligands redocked against HSP90 using the hydrated docking protocol. Values under 2 Å in bold.

To validate the implementation of this docking protocol in Vina v.1.2.0, we used six HSP90 protein-ligand complexes from the D3R Grand Challenge 2015[28]. This is an interesting system for the hydrated docking because different ligands bind with a different number of waters bridging hydrogen bonds with the protein. The RMSD of the redocked ligands in reported in Table 1, and a hand-picked system (PDB 4ykq) is depicted in Figure 1B. When looking at the best pose, only two ligands could be redocked with an RMSD below 2 Å but this number increases to 5 if the top 2 poses are considered.

AutoDock4$_{\mathrm{Zn}}$. One of the most used methods developed for AD4 is the AutoDock4$_{\mathrm{Zn}}$, a specialized force field to model zinc-coordinating ligands[13]. It is based on the use of pseudo-atoms to describe the optimal tetrahedral coordination geometry of the zinc ion complexed in proteins, and the definition of improved potentials to describe its interaction with coordinating elements in the ligand (i.e., nitrogen, oxygen, and sulfur). The coordination geometry is encoded in the grid maps for the standard AD4 atom types. The results of the implementation of this method in Vina are shown in Figure 1C. The method is capable of reproducing the improved docking performance reported for the original work with AD4, showing an excellent overlap with the crystallographic pose of the ligand and optimal zinc coordination geometry.

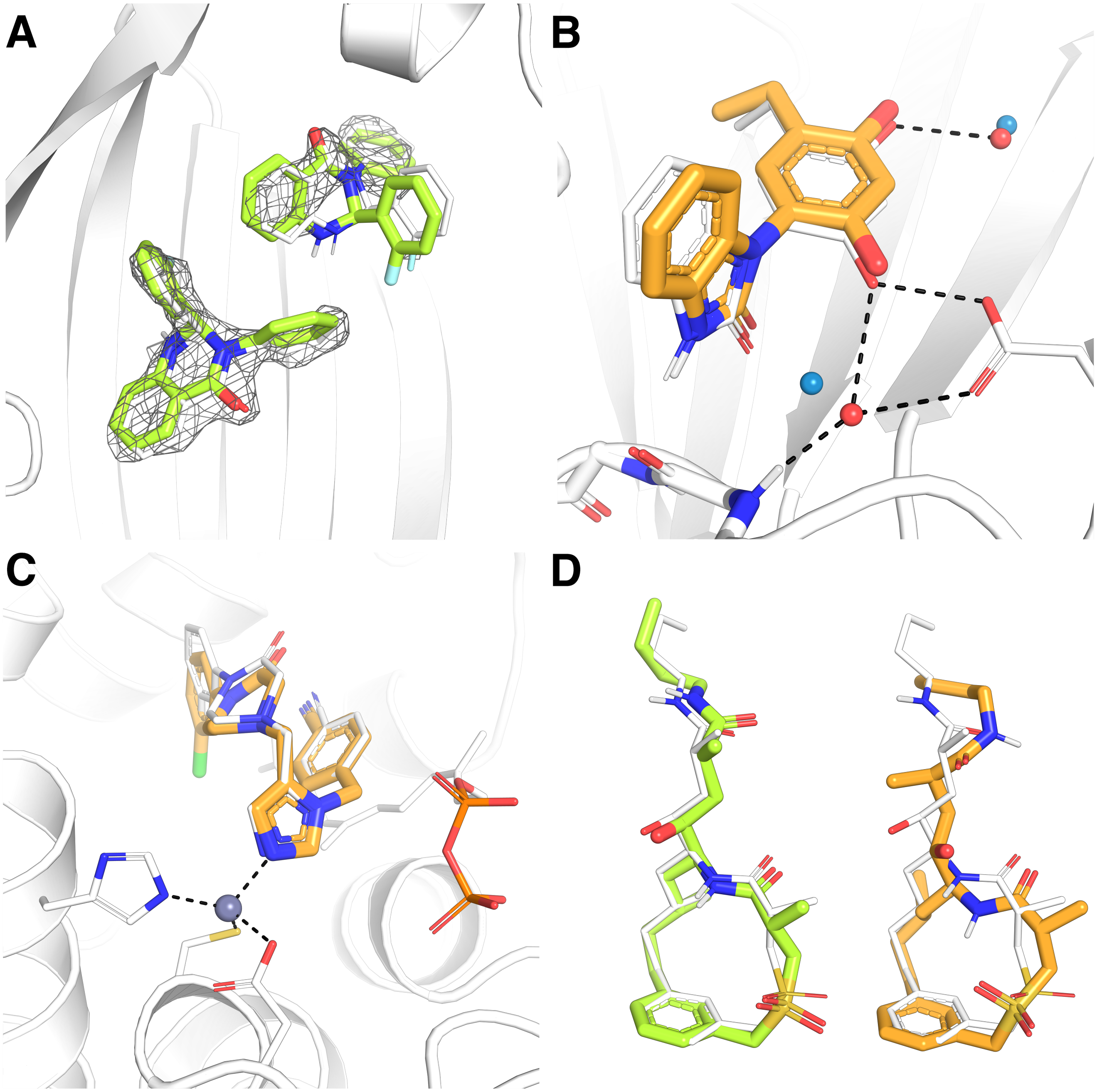

Fig. 1. Example applications of AutoDockVina v.1.2.0 for docking (A) multiple ligands (PDB 5x72), (B) with water molecules using the hydrated docking protocol from AutoDock4 (PDB 4ykq), (C) in presence of zinc using the AutoDock4$_{\mathrm{Zn}}$ forcefield (PDB 1s63), or (D) flexible macrocycles (compound 19 from the BACE dataset of the D3R Grand Challenge 4). Proteins are represented in white cartoon and crystal poses and protein residues in white thin sticks. The 2F$_o$-Fc electron-density map, contoured at 2.0σ, is colored grey. The docking poses are represented in sticks, and colored in green and orange when docked using the Vina or AutoDock4 scoring function, respectively. Docking with zinc was done in presence of the farnsesyl disphosphate molecule, represented in sticks and colored in white.

Macrocycle conformational sampling. Docking of macrocycles is a challenging task because of the difficulty of sampling the ring flexibility by modeling the correlated torsional changes resulting in different conformations. AD4 has a specialized protocol to dock macrocycles while modeling their flexibility on-the-fly[12]. One of the bonds in the ring structure is broken, resulting in an open form of the macrocycle that removes the need for correlated torsional variations, enabling torsional degrees of freedom to be explored independently. During the docking, an attractive potential is applied to restore the bond resulting in the closed ring form. Thus, macrocycle conformations are sampled while adapting to the binding pocket, at the cost of increased search complexity with the added extra rotatable bonds. This method was successfully applied in the D3R Grand Challenge 4[29], both by us[30,31] and others[32].

The current implementation of macrocycle sampling in AutoDock Vina 1.2.0 is the same as in AutoDock-GPU[4], which differs from the original approach[12] by the use of dummy atoms. The dummy atom implementation was previously described[30], and is summarized herein. To each of the atoms previously connected by the broken bond, a dummy atom is added. The distance between each dummy atom and its parent atom corresponds to the length of the broken bond, and the 1-3 angle matches the original bond geometry. During docking, a linear potential attracts each dummy atom to overlap with the opposite parent atom, restoring the broken bond with the proper distance and 1-3 angles.

To validate our implementation in Vina we used 19 macrocycles from the BACE-1 set of the D3R Grand Challenge 4 (Figure 1D). We tested both the AD4 and Vina scoring functions, an attractive potential of 5 or 50 kcal/mol/Å[30], and search exhaustiveness of 8 or 64 (Table 2). The lowest RMSD with respect to the experimental coordinates was obtained using the Vina scoring function with an attractive potential of 50 kcal/mol/Å. Given the search complexity of fully flexible macrocycles, not suprisingly the search exhaustiveness is the most important parameter driving the result quality. The AD4 scoring function seemed to perform better at lower exhaustiveness, while the Vina scoring function required higher exhaustiveness values to achieve good performance. Overall, the best RMSD results were achieved with the Vina scoring function, using exhaustiveness of 64 and an attractive potential of 50 kcal/mol/Å.

scoring	exhaust.	attractive pot.	RMSD	RMSD
function		(kcal/mol/Å)	average	median
AD4	8	5	2.33	1.52
AD4	8	50	3.03	1.74
AD4	64	5	2.11	1.54
AD4	64	50	2.04	1.50
Vina	8	5	5.93	7.71
Vina	8	50	5.10	5.73
Vina	64	5	1.82	1.02
Vina	64	50	1.22	0.77

Table 2. Redocking of 19 macrocycles of the BACE-1 set from the D3R Grand Challenge 4

Python bindings

Leveraging the popularity and utility of the Python language[33], we added bindings for the language in the version 1.2.0. In order to generate a Python interface as compliant (i.e., *pythonic*) as possible with the language guidelines, the Vina code was refactored as a library. A Python extension module was created automatically from the C++ code using SWIG (Simplified Wrapper and Interface Generator)[34]. Most of the features are provided either by binding directly to the existing the C++ code, or via additional convenience functions to simplify the access from the Python environment.

The availability of Python bindings facilitates the use and integration of the Vina docking engine in complex and articulated pipelines, reducing the code burden necessary to integrate the docking process with the numerous Python packages and other software suites that support the language. Through these bindings, users can embed the docking engine directly in any Python pipeline by importing directly the Vina package instead of spawning and managing external processes. We anticipate that this will allow users from the community to more rapidly design, implement, and distribute multi-step docking protocols, as well as facilitating its integration in web services.

The Python interface provides the following features:

- create an instance of the AutoDock Vina engine (scoring function choice, CPU cores, random seed)
- read/write one or more PDBQT files
- compute Vina affinity maps
- read/write Vina affinity maps and read AutoDock affinity maps
- randomize orientation and position of the input ligand(s) (`randomize_only`)
- evaluate the energy of the current pose or poses (`score_only`)
- perform local optimization (`local_only`)
- set Monte-Carlo global search parameters (exhaustiveness, number of output poses, maximum evaluations, etc,...)

Thus, a basic Vina calculation can be configured and performed as follow:

```python
#!/usr/bin/env python
# Simple example with Vina Python bindings
#

from vina import Vina

v = Vina()

v.set_receptor("protein.pdbqt")
v.set_ligand_from_file("ligand.pdbqt")

v.compute_vina_maps([0., 0., 0.], [30, 30, 30])
v.dock(exhaustiveness=32)

v.write_poses("docking_results.pdbqt")
```

The code is documented using Python docstrings, and the documentation is automatically generated using Sphinx[35].

Miscellaneous improvements

Batch ligand docking. AutoDock Vina 1.2.0 can dock an arbitrary number of ligands with a single launch of the program. Multiple ligand file names can be specified with the new option `-batch` and each ligand is docked without recalculating or loading the maps every time for each ligand. This improves computing efficiency when running very large virtual screenings.

Setting the number of evaluations. Vina performs 8 independent MC runs by default. For more complex searches (i.e., more flexible ligands, larger binding sites), this number can be modified with the `exhaustiveness` parameter. Conversely, the number of energy evaluations performed in each run is determined using heuristics that take into account the number of atoms and rotatable bonds. In this new version, we added an option `-max_evals` that allows users to specify the number of evaluations to be performed (analogous to the `ga_num_evals` option in AD4), providing more control over the search algorithm.

Optionally disable pose refinement. By default Vina uses the receptor structure prior to docking to pre-calculate grid maps, and after dockings are completed to minimize poses using direct pairwise interactions with the receptor (instead of using the pre-calculated grid maps as during docking). However, when map files are loaded instead of calculated internally, the refinement with receptor atoms is disabled because there is no way to guarantee consistency between the internal energy potentials used for docking and those used for calculating the grid maps. In fact, one of the purposes of loading maps from external files is to explicitly allow the user to modify them. To avoid any ambiguity, rigid receptor file and maps are not allowed to be specified at the same time. When docking with the AD4 scoring function, the post-processing minimization is never available, and grid maps must be provided. Post-docking refinement for the Vina scoring function can now be to disable with the `-no_refine` option.

Virtual screening performance comparison

With the possibility of using the same search method for both AD4 and Vina scoring functions, it is now possible to homogeneously assess their screening performance (i.e., without the uncertainty of the different search methods). Therefore, we performed virtual screenings using 50 representative systems selected from the DUD-e dataset[36], for a total of 4938 actives and 292778 decoys compounds. For each target, active and decoy sets, and co-crystallized ligands, were docked. Details about the screening library and receptor preparation and analysis are discussed in Supplementary Information.

The results show that overall both Vina and AD4 scoring function perform similarly in early recognition, but the Vina scoring function reproduces crystal poses with higher accuracy. For Vina and AD4, respectively, the average AUC were 0.71 ± 0.15 and 0.60 ± 0.21, BEDROC 0.26 ± 0.19 and 0.27 ± 0.20, and for EF 10.87 ± 11.72 and 9.92 ± 12.58. However, the two scoring functions show different performance depending on the targets. Based on the BEDROC metric, the AD4 scoring function outperforms the Vina scoring function for the following targets: *pur2*, *fpps*, *tryb1*, *xiap* and *nram* but under performs for *thb*, *fak1*, *kif11*, *sahh* and *jak2*. For 17 out of 50 targets, both scoring functions perform poorly, with BEDROC metrics lower than 0.1 (see Tables S2 and S3). In terms of success rate in reproducing experimental coordinates within 2 Å RMSD, 74 and 58 % of them are correctly predicted when considering only the top pose (top 1) for the Vina and AD4 scoring functions, respectively. When considering the first two poses (top 2), the success rates increase to 80 and 68 %, and using the first three poses (top 3) to 84 and 70 % for Vina and AD4, respectively. When using a more stringent cutoff of 0.5 Å RMSD, only 28 and 14 % of the top poses (top 1) are correctly predicted for the Vina and AD4 scoring functions, respectively. Those results are aligned with recent studies showing that on average the Vina scoring function outperforms the AD4 scoring function for pose prediction[37,38]. However, a more accurate ranking using the AD4 scoring function was not observed as results from a previous study shown[38]. These results show that the scoring functions performance is target-dependent, and the availability of the two scoring functions in the same docking engine simplifies the process of testing and selecting the most effective one for a given target.

Discussion and conclusion

This work is an effort toward the unification of the different functionalities developed within the AutoDock Suite. AutoDock Vina 1.2.0 allows users to access the powerful iterated local search of Vina with many of the features implemented in AutoDock4, among which the AutoDock4.2 scoring function itself, and the capability of reading and writing grid maps with pre-calculated target interactions. This latter feature unlocked the possibility of porting a number of existing methods and specialized scoring functions to Vina, such as hydrated docking[14], the AutoDock4$_{\mathrm{Zn}}$ [13] force field, and the AutoDock-Bias docking[21].

For other methods, such as sampling of macrocycle conformations during docking, that require the definition of *ad-hoc* intramolecular terms, the modifications have been implemented in the source code. This was necessary because AutoDock Vina 1.2.0 does not allow the user to create new atom types or modify pairwise interactions without changes to the source code.

AutoDock Vina 1.2.0 facilitates the design and execution of simple and complex docking simulations. The new version provides Python bindings, enabling easier scripting for virtual screening and other advanced applications. We also implemented batch processing to streamline high-throughput virtual screenings, as well as simultaneous multiple-ligand docking against a single target structure. All new features can be accessed both from the command line interface when using a compiled Vina binary, or from Python.

Having both Vina and AD4 scoring functions available with a common search algorithm allowed a direct comparison of their screening power in a number of targets from the DUD-E set. The scoring functions performed similarly overall across all targets that we considered. However, when considering individual targets, either scoring function can outperform the other, highlighting the need for a better scoring function that performs consistently well for every target.

Due to the added functionality and the array of scoring methods that are now available, we believe that AutoDock Vina 1.2.0 is a useful tool for molecular docking for both novice and expert users.

Availability

AutoDock-Vina is released as open source under a Apache license. The source code, the documentation, and up-

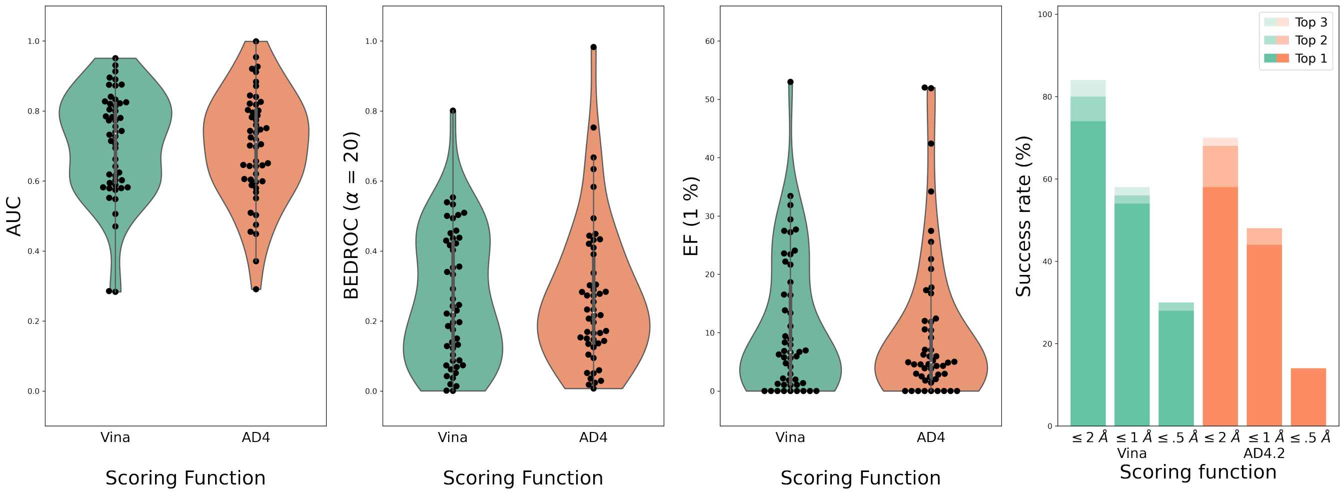

Fig. 2. Early recognition of active compounds from the DUD-e dataset and crystal pose prediction. In total 50 targets from the DUD-e dataset were selected and used to compare Vina and AutoDock4.2 scoring functions in AutoDock Vina. Violin plots of (A) AUC, (B) BEDROC using an α of 160.9 and (E) EF at 1%. (D) Docking success rate for Vina and AutoDock4.2 scoring functions using crystal poses considering the top 1, top 2 and top 3 poses. The pose prediction was considered as successful if the RMSD was inferior than 2, 1 or 0.5 Å from the crystal pose.

dates are available on GitHub: https://github.com/ccsb-scripps/AutoDock-Vina.

ACKNOWLEDGEMENTS
We thank David Goodsell for the insightful discussions, and Paolo Governa for the help with the beta testing. This work was supported by the NIH grant GM069832. This is manuscript #30066 from Scripps Research. We acknowledge the use of NumPy[39], Matplotlib[40], Seaborn[41], Pandas[42] and Jupyter Notebook[43]. This manuscript is dedicated to the memory of Prof. Maurizio Botta, whose encouragement to "*always do more*" was instrumental to the development of some of the methods described here.

Bibliography

1. Trott, O.; Olson, A. J. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry* **2010**, *31*, 455–461.
2. Forli, S.; Huey, R.; Pique, M. E.; Sanner, M. F.; Goodsell, D. S.; Olson, A. J. Computational protein–ligand docking and virtual drug screening with the AutoDock suite. *Nature protocols* **2016**, *11*, 905–919.
3. Huey, R.; Morris, G. M.; Olson, A. J.; Goodsell, D. S. A semiempirical free energy force field with charge-based desolvation. *Journal of computational chemistry* **2007**, *28*, 1145–1152.
4. Santos-Martins, D.; Solis-Vasquez, L.; Koch, A.; Forli, S. Accelerating autodock4 with gpus and gradient-based local search. **2019**,
5. Ravindranath, P. A.; Forli, S.; Goodsell, D. S.; Olson, A. J.; Sanner, M. F. AutoDockFR: advances in protein-ligand docking with explicitly specified binding site flexibility. *PLoS computational biology* **2015**, *11*, e1004586.
6. Zhang, Y.; Sanner, M. F. Docking flexible cyclic peptides with AutoDock CrankPep. *Journal of chemical theory and computation* **2019**, *15*, 5161–5168.
7. Alhossary, A.; Handoko, S. D.; Mu, Y.; Kwoh, C.-K. Fast, accurate, and reliable molecular docking with QuickVina 2. *Bioinformatics* **2015**, *31*, 2214–2216.
8. Koes, D. R.; Baumgartner, M. P.; Camacho, C. J. Lessons learned in empirical scoring with smina from the CSAR 2011 benchmarking exercise. *Journal of chemical information and modeling* **2013**, *53*, 1893–1904.
9. Nivedha, A. K.; Thieker, D. F.; Makeneni, S.; Hu, H.; Woods, R. J. Vina-Carb: improving glycosidic angles during carbohydrate docking. *Journal of chemical theory and computation* **2016**, *12*, 892–901.
10. Koebel, M. R.; Schmadeke, G.; Posner, R. G.; Sirimulla, S. AutoDock VinaXB: implementation of XBSF, new empirical halogen bond scoring function, into AutoDock Vina. *Journal of cheminformatics* **2016**, *8*, 27.
11. Quiroga, R.; Villarreal, M. A. Vinardo: A scoring function based on autodock vina improves scoring, docking, and virtual screening. *PloS one* **2016**, *11*, e0155183.
12. Forli, S.; Botta, M. Lennard-Jones potential and dummy atom settings to overcome the AUTODOCK limitation in treating flexible ring systems. *Journal of chemical information and modeling* **2007**, *47*, 1481–1492.
13. Santos-Martins, D.; Forli, S.; Ramos, M. J. a.; Olson, A. J. AutoDock4Zn: an improved AutoDock force field for small-molecule docking to zinc metalloproteins. *Journal of chemical information and modeling* **2014**, *54*, 2371–2379.
14. Forli, S.; Olson, A. J. A force field with discrete displaceable waters and desolvation entropy for hydrated ligand docking. *Journal of medicinal chemistry* **2012**, *55*, 623–638.
15. Serrano, P.; Aubol, B. E.; Keshwani, M. M.; Forli, S.; Ma, C.-T.; Dutta, S. K.; Geralt, M.; Wüthrich, K.; Adams, J. A. Directional phosphorylation and nuclear transport of the splicing factor SRSF1 is regulated by an RNA recognition motif. *Journal of molecular biology* **2016**, *428*, 2430–2445.
16. Bianco, G.; Forli, S.; Goodsell, D. S.; Olson, A. J. Covalent docking using autodock: Two-point attractor and flexible side chain methods. *Protein Science* **2016**, *25*, 295–301.
17. Nocedal, J.; Wright, S. *Numerical optimization*; Springer Science & Business Media, 2006.
18. Houston, D. R.; Walkinshaw, M. D. Consensus docking: improving the reliability of docking in a virtual screening context. *Journal of chemical information and modeling* **2013**, *53*, 384–390.
19. Cuzzolin, A.; Sturlese, M.; Malvacio, I.; Ciancetta, A.; Moro, S. DockBench: an integrated informatic platform bridging the gap between the robust validation of docking protocols and virtual screening simulations. *Molecules* **2015**, *20*, 9977–9993.
20. Weiner, S. J.; Kollman, P. A.; Case, D. A.; Singh, U. C.; Ghio, C.; Alagona, G.; Profeta, S.; Weiner, P. A new force field for molecular mechanical simulation of nucleic acids and proteins. *Journal of the American Chemical Society* **1984**, *106*, 765–784.
21. Arcon, J. P.; Modenutti, C. P.; Avendaño, D.; Lopez, E. D.; Defelipe, L. A.; Ambrosio, F. A.; Turjanski, A. G.; Forli, S.; Marti, M. A. AutoDock Bias: improving binding mode prediction and virtual screening using known protein–ligand interactions. *Bioinformatics* **2019**, *35*, 3836–3838.
22. Lazaridis, T. Inhomogeneous fluid approach to solvation thermodynamics. 1. Theory. *The Journal of Physical Chemistry B* **1998**, *102*, 3531–3541.
23. Lazaridis, T. Inhomogeneous fluid approach to solvation thermodynamics. 2. Applications to simple fluids. *The Journal of Physical Chemistry B* **1998**, *102*, 3542–3550.
24. Nguyen, C. N.; Kurtzman Young, T.; Gilson, M. K. Grid inhomogeneous solvation theory: Hydration structure and thermodynamics of the miniature receptor cucurbit[7]uril. *137*.
25. Uehara, S.; Tanaka, S. AutoDock-GIST: Incorporating thermodynamics of active-site water into scoring function for accurate protein-ligand docking. *Molecules* **2016**, *21*, 1604.
26. Irwin, J. J.; Tang, K. G.; Young, J.; Dandarchuluun, C.; Wong, B. R.; Khurelbaatar, M.; Moroz, Y. S.; Mayfield, J.; Sayle, R. A. ZINC20—A Free Ultralarge-Scale Chemical Database for Ligand Discovery. *Journal of Chemical Information and Modeling* **2020**,
27. Jiang, Y.; Zhuang, C.; Chen, L.; Lu, J.; Dong, G.; Miao, Z.; Zhang, W.; Li, J.; Sheng, C. Structural biology-inspired discovery of novel KRAS–PDEδ inhibitors. *Journal of Medicinal Chemistry* **2017**, *60*, 9400–9406.
28. Gathiaka, S.; Liu, S.; Chiu, M.; Yang, H.; Stuckey, J. A.; Kang, Y. N.; Delproposto, J.; Kubish, G.; Dunbar, J. B.; Carlson, H. A.; Burley, S. K.; Walters, W. P.; Amaro, R. E.; Feher, V. A.; Gilson, M. K. D3R grand challenge 2015: evaluation of protein–ligand pose and affinity predictions. *Journal of computer-aided molecular design* **2016**, *30*, 651–668.
29. Parks, C. D.; Gaieb, Z.; Chiu, M.; Yang, H.; Shao, C.; Walters, W. P.; Jansen, J. M.; McGaughey, G.; Lewis, R. A.; Bembenek, S. D.; Ameriks, M. K.; Mirzadegan, T.; Burley, S. K.; Amaro, R. E.; Gilson, M. K. D3R grand challenge 4: blind prediction of protein–ligand poses, affinity rankings, and relative binding free energies. *Journal of Computer-Aided Molecular Design* **2020**, *34*, 99–119.
30. Santos-Martins, D.; Eberhardt, J.; Bianco, G.; Solis-Vasquez, L.; Ambrosio, F. A.; Koch, A.; Forli, S. D3R Grand Challenge 4: prospective pose prediction of BACE1 ligands with AutoDock-GPU. *Journal of Computer-Aided Molecular Design* **2019**, *33*, 1071–1081.
31. El Khoury, L.; Santos-Martins, D.; Sasmal, S.; Eberhardt, J.; Bianco, G.; Ambrosio, F. A.; Solis-Vasquez, L.; Koch, A.; Forli, S.; Mobley, D. L. Comparison of affinity ranking using AutoDock-GPU and MM-GBSA scores for BACE-1 inhibitors in the D3R Grand Challenge 4. *Journal of computer-aided molecular design* **2019**, *33*, 1011–1020.
32. Lam, P. C.-H.; Abagyan, R.; Totrov, M. Macrocycle modeling in ICM: benchmarking and evaluation in D3R Grand Challenge 4. *Journal of Computer-Aided Molecular Design* **2019**, *33*, 1057–1069.
33. van Rossum, G. Python programming language. USENIX annual technical conference. 2007; p 36.
34. Beazley, D. M. SWIG: An Easy to Use Tool for Integrating Scripting Languages with C and C++. Tcl/Tk Workshop. 1996; p 74.
35. Brandl, G. Sphinx: Python documentation generator. URL https://www.sphinx-doc.org/ (accessed Feb 19, 2021).

36. Mysinger, M. M.; Carchia, M.; Irwin, J. J.; Shoichet, B. K. Directory of useful decoys, enhanced (DUD-E): better ligands and decoys for better benchmarking. *Journal of medicinal chemistry* **2012**, *55*, 6582–6594.
37. Gaillard, T. Evaluation of AutoDock and AutoDock Vina on the CASF-2013 benchmark. *Journal of chemical information and modeling* **2018**, *58*, 1697–1706.
38. Nguyen, N. T.; Nguyen, T. H.; Pham, T. N. H.; Huy, N. T.; Bay, M. V.; Pham, M. Q.; Nam, P. C.; Vu, V. V.; Ngo, S. T. Autodock vina adopts more accurate binding poses but autodock4 forms better binding affinity. *Journal of Chemical Information and Modeling* **2019**, *60*, 204–211.
39. Harris, C. R. et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362.
40. Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in science & engineering* **2007**, *9*, 90–95.
41. Waskom, M. et al. mwaskom/seaborn: v0.8.1 (September 2017). 2017; https://doi.org/10.5281/zenodo.883859.
42. Wes McKinney, Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference. 2010; pp 56 – 61.
43. Kluyver, T.; Ragan-Kelley, B.; Pérez, F.; Granger, B.; Bussonnier, M.; Frederic, J.; Kelley, K.; Hamrick, J.; Grout, J.; Corlay, S.; Ivanov, P.; Avila, D.; Abdalla, S.; Willing, C. Jupyter Notebooks – a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas. 2016; pp 87 – 90.