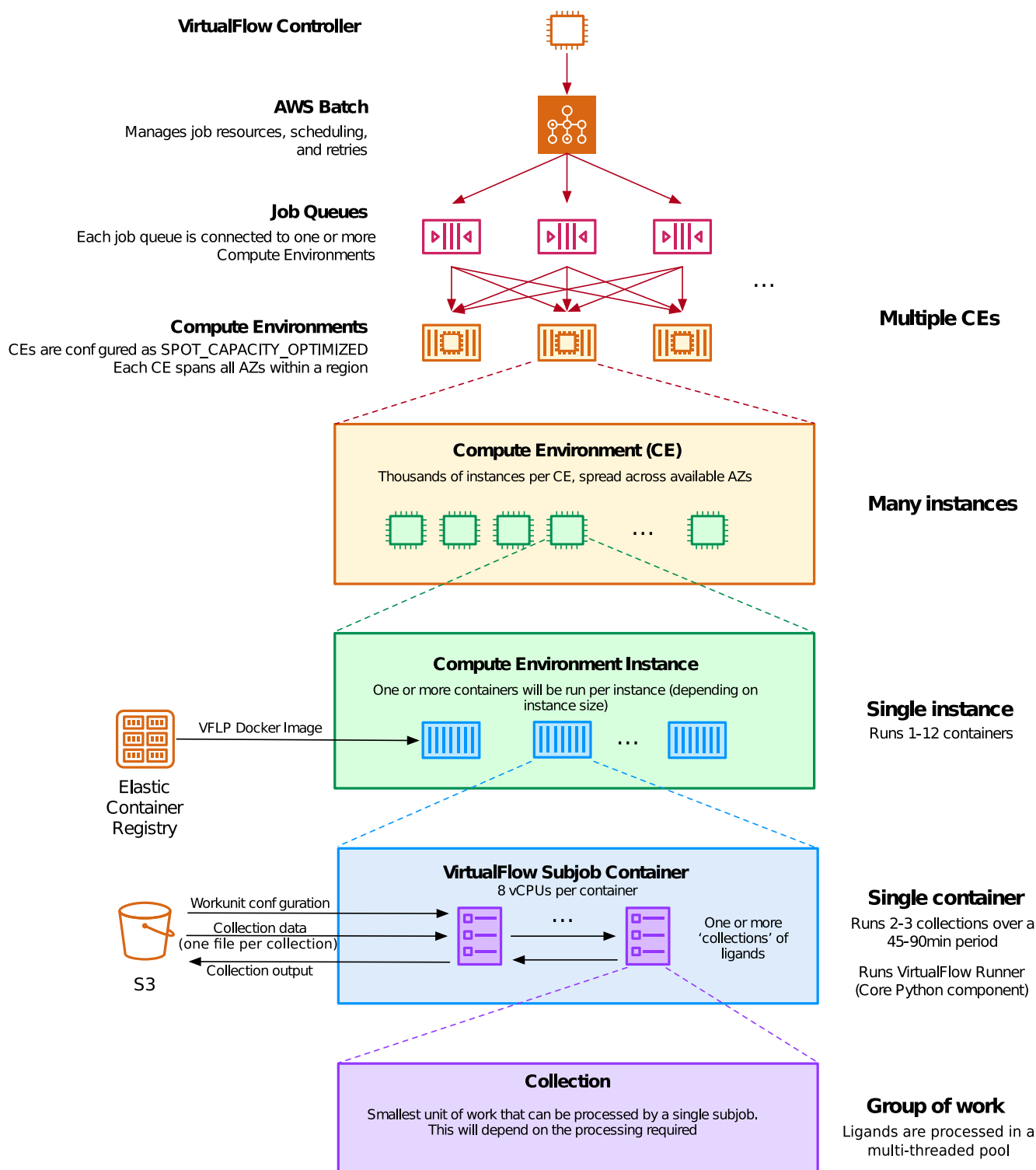

AI-enhanced adaptive virtual screening of large libraries for ligand discovery

In the format provided by the
authors and unedited

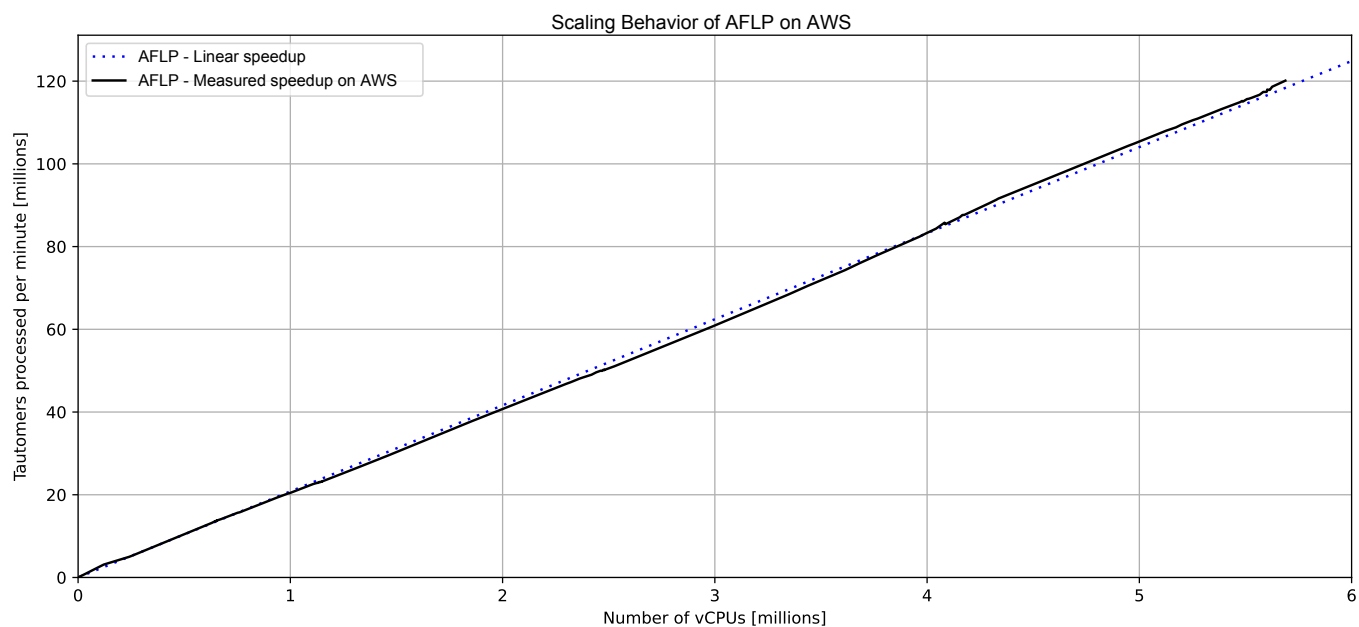
Supplementary Information

Contents

A	Available docking protocols	11
B	Supported Output File Formats by AFLP	40
C	AdaptiveFlow File and Folder Structures	43
D	AFLP Workflow	50
E	AFVS	53
F	AdaptiveFlow Unity Workflow	56
G	FSP1 Ligand Discovery with AdaptiveFlow	56
H	ADME analysis of afi-FSP1-1, afi-FSP1-2, afi-FSP1-3, afi-FSP1-4	81
I	Experimental Validation of AdaptiveFlow using PARP1	83
I.1	Targeting PARP1	83
I.2	Applying AdaptiveFlow to PARP1	83
I.3	Selection and Characterization of Initial PARP1 Inhibitor Compounds	83
I.4	Characterization of PARP1-Inhibitor Interactions via Protein NMR Spectroscopy	85
I.5	Co-Crystallization Experiments	88
I.6	Cellular Activity of Identified Hits	88
I.7	Comparison to Approved Drugs	89
I.8	Reagents	89
I.9	Receptor Preparation	90
I.10	Prioritizing Hits for Experimental Validation	90
I.11	Protein Expression and Purification	90
I.12	NADH/NAD ⁺ Activity-Based Assay	91
I.13	NMR Spectroscopy	91
I.14	Crystallography	91
I.15	Colony Formation Assays	92
J	Computational Overhead	93
J.1	Hierarchical Strategy	93
J.2	File Usage Analysis for One Million Molecules	94
K	Configuration File of AFLP	97
L	Configuration File of AFVS	105
M	Configuration File of AF Unity	109



Supplementary Fig. 1. Compute Stack deployed by AdaptiveFlow in AWS. The login node is used by the user to control the workflow in AWS via the workload manager AWS Batch. AWS Batch has multiple Job Queues to which jobs are submitted by AdaptiveFlow, and each job queue deploys multiple Compute Environments that can spin up and use a large number of virtual machines (instances) of specific types. On each instance, one or more subjobs are run, and each subjobs deploys a docker container that carries out the computations of AdaptiveFlow by executing AdaptiveFlow Runner (a core python component of AdaptiveFlow). Each subjob typically processes a few ligand collections, where each ligand collection contains multiple ligands (e.g. 1000).



Supplementary Fig. 2. Scaling behavior of AFLP of on AWS. The scaling behavior is perfectly linear even when using extremely large numbers of CPUs. Up to 5.6 million vCPUs were used in parallel across multiple AWS regions. As can be seen in the plot, the scaling behavior is perfectly linear, implying there is no slowdown even when using extremely large numbers of vCPUs.

Feature	AdaptiveFlow
General Features	
Codebase	Python
Native cloud support	available (AWS)
Linear scaling behavior	demonstrated up to 5.6 million CPUs
AdaptiveFlow Ligand Libraries	
Total size of ligand library	68.7 billion
Additional predicted ligand properties	28
PDB format	available
PDBQT format	available
SDF format	available
MOL2 format	available
SMILES format	available
SELFIES format	available
Parquet format	available
Sparse version (for ATG-VSs)	available
Interactive web interface	yes (18 dimensions)
Accessibility	AdaptiveFlow website and AWS Open Data
AdaptiveFlow for Ligand Preparation (AFLP) Module	
Stereoisomer enumeration	supported
External open-source tools for ligand preparation	extended support
SMILES as input format	supported
SELFIES as input format	supported
Amino acid sequences as input format	supported
SELFIES as output format	supported
Prediction of physical molecular properties	supported (27 properties)
Automatic multidimensional matrix generation	extended support
Quality control	extended support
Enhanced speed	extended support
AdaptiveFlow for Virtual Screening (AFVS) Module	
Adaptive target-guided virtual screens	supported
- Machine learning enhancement	supported
docking protocols	1500
Deep learning-based scoring functions	supported
Deep learning-based pose prediction methods	supported
Molecular dynamics-based methods	supported
Protein as target	supported
RNA as target	supported
DNA as target	supported
Covalent docking	extended support
Protein ligands (protein-receptor docking)	supported
Peptide ligands	supported
Parquet output format	supported
Docking quality control	supported
ARM supported	supported
Enhanced x86 performance	available (for several docking protocols)
GPU support	supported
AdaptiveFlow Unity (AFU) Module	
Standalone mode	available
API mode	available

Supplementary Table 1. Features Supported by AdaptiveFlow. Features grouped by AdaptiveFlow module: General, AFLP, AFVS, and AF Unity. AdaptiveFlow has significantly more features compared to the original version.

Index	Property	AdaptiveFlow Parameter	Preparation Method	Used for 18-Dimensional Matrix
1	molecular weight	mw_file	Input library	yes
2	logP	logp_jchem	JChem	yes
3	hydrogen bond acceptor count	hba_jchem	JChem	yes
4	hydrogen bond donor count	hbd_jchem	JChem	yes
5	rotatable bond count	roth_jchem	JChem	yes
6	topological polar surface area	tpsa_jchem	JChem	yes
7	molecular refractivity	mr_jchem	JChem	yes
8	solubility [molar]	logs	JChem	yes
9	number of bonds	bondcount_jchem	JChem	no
10	aromatic proportion	ap	JChem	yes
11	ring count	ringcount	JChem	no
12	aromatic ring count	aromaticringcount	JChem	no
13	logD	logd	JChem	no
14	formal charge	formalcharge	AdaptiveFlow	yes
15	positive charge count	positivechargecount	AdaptiveFlow	yes
16	negative charge count	negativechargecount	AdaptiveFlow	yes
17	fraction of sp3 carbon atoms	fsp3_file	Input library	yes
18	total atom count	atomcount_jchem	JChem	no
19	heavy atom count	heavyatomcount_file	Input library	no
20	chiral center count	chiralcentercount	JChem	yes
21	doublebond stereoisomer count	doublebondstereoisomercount	JChem	yes
22	halogen atom count	halogencount	JChem	yes
23	sulfur atom count	sulfurcount	JChem	yes
24	nitrogen and oxygen atom count	NOcount	JChem	no
25	electronegative atom count	electronegativeatomcount	JChem	no
26	Enamine compound class	enamine_compound_class	Input library	yes
27	quantitative estimate of druglikeness (QED)	qed_rdkit	RDKit	no
28	Murcko scaffold	scaffold_rdkit	RDKit	no

Supplementary Table 2. Calculated Properties of Ligands in the Enamine REAL Space. All properties of the ligands that were determined/calculated during the preparation of the REAL Space (version 2022q1-2) with AFVS. A few properties (molecular weight, fsp3, heavy atom count, Enamine compound class) were available in the input library and thus could be directly used without any calculations.

Property	Available Calculation Methods
Preparation of Ligands into Ready-To-Dock Format	
Desalting	AFLP (internal)
Neutralization	JChem, Open Babel
Stereoisomer Generation	JChem, RDKit
Tautomer Generation	JChem, Open Babel, RDKit
Protonation State Calculation	JChem, Open Babel, Epik, QupKake
3D Coordinate Conversion	JChem, Open Babel, RDKit
Target Format Conversion	Open Babel (for all formats but SELFIES) and SELFIES Python Package (only for SELFIES)
Calculation of Additional Molecular Properties	
Molecular weight	JChem, Open Babel, Input Library
logP	JChem, Open Babel, Input Library
HBA count	JChem, Open Babel, Input Library
HBD count	JChem, Open Babel, Input Library
Rotatable bond count	JChem, Open Babel, Input Library
TPSA	JChem, Open Babel, Input Library
Molecular refractivity	JChem, Open Babel, Input Library
Solubility [molar]	JChem, Open Babel, Input Library
Bond count	JChem, Input Library
Aromatic proportion	JChem, Input Library
Ring count	JChem, Input Library
Aromatic ring count	JChem, Input Library
logD	JChem, Input Library
Formal charge	AFLP (internal)
Positive charge count	AFLP (internal)
Negative charge count	AFLP (internal)
Fraction of sp ³ carbon atoms (fsp ³)	JChem, Input Library
Total atom count	JChem, Open Babel
Heavy atom count	Input Library
Chiral center count	JChem, Input Library
Doublebond stereoisomer count	JChem
Halogen atom count	JChem, Input Library
Sulfur atom count	JChem, Input Library
NO atom count	JChem, Input Library
Electronegative atom count	JChem, Input Library
Enamine compound class	Input Library
Quantitative Estimate of Druglikeness (QED)	RDKit
Murcko scaffold	RDKit

Supplementary Table 3. Supported Processing Steps of AFLP. AFLP can prepare molecules into a ready-to-dock format, and calculate additional properties of each molecule. For many of these calculations, external tools (JChem from ChemAxon, RDKit, Epik from Schrödinger (79), QupKake (80) and Open Babel) are used, while for a few AFLP calculates them internally. Many properties can also be used directly from the input library if available, avoiding the need to calculate them during the runtime.

Docking program	License model	Special features	DL-based	Hardware support	Reference
AutoDock-GPU	Open Source	GPU-accelerated version of AutoDock4	✗	GPU	(57)
AutoDock4	Open Source	Improved modelling of waters and macrocycles	✗	CPU	(81)
AutoDockZN	Open Source	Special support for zinc-coordinating ligands	✗	CPU	(82)
AutoDock Vina 1.2.0	Open Source	Python support; macromolecule support	✗	CPU	(83)
AutoDock Vina 1.1.2	Open Source	Improved optimization, scoring and multithreading	✗	CPU	(84)
AutoDockFR (ADFR)	Open Source	Improved receptor side-chain flexibility modelling	✗	CPU	(85)
AutoDock Koto	Open Source	Improved sampling	✗	CPU	(86)
ATPdock	Open Source	ATP-specific protein-ligand docking	✗	CPU	(87)
CovDock	Licence Required	Covalent docking	✗	CPU	(88)
DOCK 6.0	Free for academics	RNA-ligand, DNA-ligand, protein-protein docking	✗	CPU	(89)
FlexX	Licence Required	Covalent docking. Ensemble docking	✗	CPU	(90)
FitDock	Academic Licence	Incorporation of template fitting for docking	✗	CPU	(91)
GalaxyDock3	Licence Required	Incorporation of side chain flexibility	✗	CPU	(92)
Glide (HTVS)	Paid Academic Licencing	Less exhaustive sampling and scoring	✗	CPU	(93)
Glide (SP)	Paid Academic Licencing	More exhaustive sampling (relative to Glide HTVS) & multiple levels of accuracy	✗	CPU	(93)
Glide (XP)	Paid Academic Licencing	Most accurate sampling and scoring for Glide & different levels of accuracy	✗	CPU	(93)
Gnina	Open Source	Deep learning scoring of complex	✓	GPU/CPU scoring	(68)
GOLD	Licence Required	Genetic algorithm based docking; Covalent docking	✗	CPU	(94)
GWOVina	Open Source	Grey wolf optimization	✗	CPU	(95)
HDOCK	Academic Licence	Protein-RNA/DNA docking	✗	CPU	(96)
HSYMDOCK	Academic Licence	Protein-protein/rna docking with Cn and Dn symmetry	✗	CPU	(97)
idock	Open Source	AutoDock Vina derivative with improved speed	✗	CPU	(98)
iGEMDOCK	Open Source	-	✗	CPU	(99)
LeDock	Freeware for academic use	Excellent Sampling power (comparable to GOLD)	✗	CPU	(100)
LigandFit	Open Source	Identification of active site regions for docking	✗	CPU	(101)
LightDock	Open Source	Protein-protein docking	✗	CPU	(102)
MDock	Licence Required	Ensemble docking for multiple protein conformations	✗	CPU	(103)
MCDock	Open Source	Use of Monte Carlo simulation technique	✗	CPU	(104)
Molegro	Licence Required	-	✗	CPU	(105)
MpsDockzn	Licence Required	Zinc metalloprotein-specific scoring function	✗	CPU	(106)
PSOVina	Open Source	Improved pose sampling and speed	✗	CPU	(107)
PLANTS	Open Source	Use of ant colony optimization	✗	CPU	(108)
PI-LZerD	Open Source	Protein-protein docking	✗	CPU	(109)
PIPER	Open Source	Use of pairwise interaction potentials for protein-protein docking	✗	CPU	(110)
QuickVina 2	Open Source	Extremely fast	✗	CPU	(111)
QuickVina-W	Open Source	Improved speed for blind docking	✗	CPU	(112)
QVina2-GPU	Open Source	GPU-accelerated version of QVina2	✗	GPU	(58)
QVina2-W-GPU	Open Source	GPU-accelerated version of QVina2-W	✗	GPU	(58)
rDock	Open Source	Support for RNA targets	✗	CPU	(113)
Rosetta Ligand	Licence Required	-	✗	CPU	(114)
RLDOCK	Open Source	RNA-ligand docking	✗	CPU	(115)
SEED	Open Source	Fragment-based docking	✗	CPU	(116)
SMINA	Open Source	User specific scoring functions	✗	CPU	(117)
TANKBind	Open Source	Fast neural network based complex prediction	✓	CPU/GPU	(12)
VinaCarb	Open Source	Improved glycosidic modelling	✗	CPU	(118)
VinaXB16	Open Source	Improved halogen bonding scoring function	✗	CPU	(119)
Vina-GPU	Open Source	GPU-accelerated version of Vina	✗	GPU	(58)
Vina-GPU-2.0	Open Source	GPU-accelerated version of Vina-2.0	✗	GPU	(58)

Supplementary Table 4. Supported docking programs. The collection of unique underlying docking programs supported within AFVS and AF Unity. These programs serve as the foundational components for the 1,500 supported docking protocols (specific combinations of pose prediction and scoring algorithms) enabled by the platform.

Docking program	License model	Special features	DL-based	Hardware support	Reference
AD4 Scoring	Open Source	Improved over AutoDock 3 forcefield	X	CPU	(120)
ASP Scoring	Licence required	Atom-atom distance potential	X	CPU	(94)
Bump Filter Scoring	Free for academics	Consideration of molecule overlap with receptor atoms	X	CPU	(89)
ChemScore	Licence required	Improved speed compared to GoldScore	X	CPU	(94)
Contact Scoring	Free for academics	Incorporates the scoring performed with the DISTMAP program	X	CPU	(89)
Continuous Scoring	Free for academics	No grid calculations	X	CPU	(89)
ChemPLP	Licence required	Slightly faster than ChemScore and up to four times faster than GoldScore	X	CPU	(94)
DOCK3.5 Scoring	Free for academics	Incorporation of solvation effects	X	CPU	(89)
DeepAffinity	Open Source	Use of convolutional neural networks	✓	CPU/GPU	(70)
DeepBindRG	Open Source	Use of recurrent and convolutional neural networks	✓	CPU/GPU	(71)
Footprint Scoring	Free for academics	Calculates intermolecular hydrogen bonds and footprint comparisons, in addition to standard interactions	X	CPU	(89)
Gnina Scoring	Open Source	Convolutional neural network scoring	✓	CPU/GPU	(68)
GOLD Scoring	Licence required	Use of the chemscore function	X	CPU	(94)
GoldScore	Licence required	Original scoring function provided with GOLD	X	CPU	(94)
Grid-Based Scoring	Free for academics	Based on the non-bonded terms of the molecular mechanic force field	X	CPU	(89)
Hawkins GB/SA Scoring	Free for academics	Use of pairwise GB solvation model	X	CPU	(121)
NNScore 2.0	Open Source	Use of a fully-connected neural networks	✓	CPU/GPU	(69)
MM/PBSA	Open Source	Calculation of end-state free energies of molecules in solution	X	CPU	(74)
MM/GBSA	Open source	Improved scoring accuracy	X	CPU	(74)
MultiGrid FPS Scoring	Free for academics	Pair-wise interaction energies are computed over multiple grids rather than in Cartesian space	X	CPU	(89)
OpenBPM	Open Source	Improved scoring accuracy	X	CPU	(72)
PLP Scoring	Open source	Pairwise additive scoring function	X	CPU	(108)
PLP95 Scoring	Open source	Origin of PLP	X	CPU	(108)
PB/SA Scoring	Free for academics	Application of the ZAP electrostatics toolkit from OpenEye	X	CPU	(89)
Pharmacophore Matching Similarity Scoring	Free for academics	Calculates the level of pharmacophore overlap between a reference molecule and a candidate molecule	X	CPU	(89)
Rosetta Scoring	Open Source	Improved scoring for side-chain flexibility	X	CPU	(114)
RF-Score-VS	Open Source	Random Forest-based scoring function	X	CPU	(122)
SASA Scoring	Free for academics	Calculates the percent exposure of a ligand	X	CPU	(123)
SEED Scoring	Open Source	Continuum dielectric approximation of electrostatic contribution	X	CPU	(116)
SMINA Scoring	Open Source	Improved AutoDock Vina scoring	X	CPU	(117)
Vinardo Scoring	Open Source	Improved Vina scoring	X	CPU	(124)
Vina Scoring	Open Source	Improved scoring compared to AD4	X	CPU	(84)
Zou GB/SA Scoring	Free for academics	Fast algorithm, pairwise free energy model, for ligand binding affinity calculations	X	CPU	(123)

Supplementary Table 5. Supported Protein-Ligand Scoring Functions. The collection of docking programs supported within AFVS and AF Unity that perform both pose prediction and scoring.

Docking program	License model	Special features	DL-based	Hardware support	Reference
DiffDock	Open Source	Blind docking pose prediction	✓	GPU/CPU	(66)
EquiBind	Open Source	Blind docking pose prediction	✓	GPU/CPU	(67)
EnzyDock	Open Source	Identifies probable binding modes of multiple ligand states in an enzyme active site	✗	CPU	(125)
FRED	Academic Licence	Improved pose sampling	✗	CPU	(?)

Supplementary Table 6. Supported pose prediction methods. The collection of docking programs supported within AFVS and AF Unity that perform both pose prediction and scoring.

Target Protein	Structure ID	Structure Source	Target Site	Target Site Type
KRAS ^{G12V}	4TQ9	Protein Data Bank (RCSB)	Active site	Enzymatic site
CB1	6KPG	Protein Data Bank (RCSB)	Primary ligand pocket	Orthosteric site
PTPRB	2H04	Protein Data Bank (RCSB)	Active site	Enzymatic site
JNK3	7S1N	Protein Data Bank (RCSB)	Active site	Enzymatic site
TLR8	6ZJZ	Protein Data Bank (RCSB)	Dimerization site	Protein-protein interaction
PIK3CG	2CHZ	Protein Data Bank (RCSB)	Active site	Enzymatic site
NTR1	4GRV	Protein Data Bank (RCSB)	Primary ligand pocket	Orthosteric site
MED15-KIX	2GUT	Protein Data Bank (RCSB)	SREBP interaction site	Protein-protein interaction
KEAP1	5FNQ	Protein Data Bank (RCSB)	NRF2 interaction site	Protein-protein interaction
Raptor	AFQ8N122_F1	AlphaFold Structure Database	TOS motif	Protein-protein interaction

Supplementary Table 7. ATG-VS Benchmark Systems. The target proteins and binding sites used in the benchmark studies comparing ATG-VSs with standard ULVSs.

1071 **A Available docking protocols**

1072 AFVS and AFU support the same docking protocols. docking protocols consist of two key components, a pose prediction
1073 component (or sampling algorithm) and a scoring function. Supported docking protocols in AdaptiveFlow are either standalone
1074 docking programs that contain both a scoring function and a sampling algorithm or composite docking programs that are
1075 assembled by AdaptiveFlow from individual scoring functions and pose prediction methods. A list of all docking protocols
1076 currently available in AdaptiveFlow is contained in Supplementary Table [8](#).

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
0	AutoDock-GPU	ASP Scoring	X	X
1	AutoDock-GPU	AutoDock-GPU	X	X
2	AutoDock-GPU	Bump Filter Scoring	X	X
3	AutoDock-GPU	ChemPLP	X	X
4	AutoDock-GPU	ChemScore	X	X
5	AutoDock-GPU	Contact Scoring	X	X
6	AutoDock-GPU	Continuous Scoring	X	X
7	AutoDock-GPU	DOCK 3.5 Scoring	X	X
8	AutoDock-GPU	DeepAffinity	X	✓
9	AutoDock-GPU	DeepBindRG	X	✓
10	AutoDock-GPU	Footprint Scoring	X	X
11	AutoDock-GPU	GOLD Scoring	X	X
12	AutoDock-GPU	Gnina Scoring	X	✓
13	AutoDock-GPU	GoldScore	X	X
14	AutoDock-GPU	Grid-Based Scoring	X	X
15	AutoDock-GPU	Hawkins GB/SA Scoring	X	X
16	AutoDock-GPU	MM/GBSA Scoring	X	X
17	AutoDock-GPU	MM/PBSA Scoring	X	X
18	AutoDock-GPU	MultiGrid FPS Scoring	X	X
19	AutoDock-GPU	NNScore 2.0	X	✓
20	AutoDock-GPU	OpenBPMD	X	X
21	AutoDock-GPU	PB/SA Scoring	X	X
22	AutoDock-GPU	PLANTS Scoring	X	X
23	AutoDock-GPU	PLP	X	X
24	AutoDock-GPU	PLP95	X	X
25	AutoDock-GPU	Pharmacophore Matching Similarity Scoring	X	X
26	AutoDock-GPU	RF-Score-VS	X	X
27	AutoDock-GPU	Rosetta Scoring	X	X
28	AutoDock-GPU	SASA Scoring	X	X
29	AutoDock-GPU	SEED Scoring	X	X
30	AutoDock-GPU	SMINA Scoring	X	X
31	AutoDock-GPU	Vina scoring	X	X
32	AutoDock-GPU	Vinardo Scoring	X	X
33	AutoDock-GPU	Zou GB/SA Scoring	X	X
34	AutoDock4	ASP Scoring	X	X
35	AutoDock4	AutoDock4	X	X
36	AutoDock4	Bump Filter Scoring	X	X
37	AutoDock4	ChemPLP	X	X
38	AutoDock4	ChemScore	X	X
39	AutoDock4	Contact Scoring	X	X
40	AutoDock4	Continuous Scoring	X	X
41	AutoDock4	DOCK 3.5 Scoring	X	X
42	AutoDock4	DeepAffinity	X	✓
43	AutoDock4	DeepBindRG	X	✓
44	AutoDock4	Footprint Scoring	X	X
45	AutoDock4	GOLD Scoring	X	X
46	AutoDock4	Gnina Scoring	X	✓
47	AutoDock4	GoldScore	X	X
48	AutoDock4	Grid-Based Scoring	X	X
49	AutoDock4	Hawkins GB/SA Scoring	X	X
50	AutoDock4	MM/GBSA Scoring	X	X
51	AutoDock4	MM/PBSA Scoring	X	X
52	AutoDock4	MultiGrid FPS Scoring	X	X
53	AutoDock4	NNScore 2.0	X	✓
54	AutoDock4	OpenBPMD	X	X

Supplementary Table 8. docking protocols supported by the AFVS and AFU Modules. List of Supported docking protocols. Both the pose prediction/Sampling Method, as well as the scoring function, are listed. For both the sampling method and the scoring function, it is indicated whether they are based on deep learning (DL) or not (continued).

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
55	AutoDock4	PB/SA Scoring	X	X
56	AutoDock4	PLANTS Scoring	X	X
57	AutoDock4	PLP	X	X
58	AutoDock4	PLP95	X	X
59	AutoDock4	Pharmacophore Matching Similarity Scoring	X	X
60	AutoDock4	RF-Score-VS	X	X
61	AutoDock4	Rosetta Scoring	X	X
62	AutoDock4	SASA Scoring	X	X
63	AutoDock4	SEED Scoring	X	X
64	AutoDock4	SMINA Scoring	X	X
65	AutoDock4	Vina scoring	X	X
66	AutoDock4	Vinardo Scoring	X	X
67	AutoDock4	Zou GB/SA Scoring	X	X
68	AutoDockZN	AD4 Scoring	X	X
69	AutoDockZN	ASP Scoring	X	X
70	AutoDockZN	AutoDockZN	X	X
71	AutoDockZN	Bump Filter Scoring	X	X
72	AutoDockZN	ChemPLP	X	X
73	AutoDockZN	ChemScore	X	X
74	AutoDockZN	Contact Scoring	X	X
75	AutoDockZN	Continuous Scoring	X	X
76	AutoDockZN	DOCK 3.5 Scoring	X	X
77	AutoDockZN	DeepAffinity	X	✓
78	AutoDockZN	DeepBindRG	X	✓
79	AutoDockZN	Footprint Scoring	X	X
80	AutoDockZN	GOLD Scoring	X	X
81	AutoDockZN	Gnina Scoring	X	✓
82	AutoDockZN	GoldScore	X	X
83	AutoDockZN	Grid-Based Scoring	X	X
84	AutoDockZN	Hawkins GB/SA Scoring	X	X
85	AutoDockZN	MM/GBSA Scoring	X	X
86	AutoDockZN	MM/PBSA Scoring	X	X
87	AutoDockZN	MultiGrid FPS Scoring	X	X
88	AutoDockZN	NNScore 2.0	X	✓
89	AutoDockZN	OpenBPMO	X	X
90	AutoDockZN	PB/SA Scoring	X	X
91	AutoDockZN	PLANTS Scoring	X	X
92	AutoDockZN	PLP	X	X
93	AutoDockZN	PLP95	X	X
94	AutoDockZN	Pharmacophore Matching Similarity Scoring	X	X
95	AutoDockZN	RF-Score-VS	X	X
96	AutoDockZN	Rosetta Scoring	X	X
97	AutoDockZN	SASA Scoring	X	X
98	AutoDockZN	SEED Scoring	X	X
99	AutoDockZN	SMINA Scoring	X	X
100	AutoDockZN	Vina scoring	X	X
101	AutoDockZN	Vinardo Scoring	X	X
102	AutoDockZN	Zou GB/SA Scoring	X	X
103	AutoDock Vina 1.2.0	AD4 Scoring	X	X
104	AutoDock Vina 1.2.0	ASP Scoring	X	X
105	AutoDock Vina 1.2.0	AutoDock Vina 1.2.0	X	X
106	AutoDock Vina 1.2.0	Bump Filter Scoring	X	X
107	AutoDock Vina 1.2.0	ChemPLP	X	X
108	AutoDock Vina 1.2.0	ChemScore	X	X
109	AutoDock Vina 1.2.0	Contact Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
110	AutoDock Vina 1.2.0	Continuous Scoring	X	X
111	AutoDock Vina 1.2.0	DOCK 3.5 Scoring	X	X
112	AutoDock Vina 1.2.0	DeepAffinity	X	✓
113	AutoDock Vina 1.2.0	DeepBindRG	X	✓
114	AutoDock Vina 1.2.0	Footprint Scoring	X	X
115	AutoDock Vina 1.2.0	GOLD Scoring	X	X
116	AutoDock Vina 1.2.0	Gnina Scoring	X	✓
117	AutoDock Vina 1.2.0	GoldScore	X	X
118	AutoDock Vina 1.2.0	Grid-Based Scoring	X	X
119	AutoDock Vina 1.2.0	Hawkins GB/SA Scoring	X	X
120	AutoDock Vina 1.2.0	MM/GBSA Scoring	X	X
121	AutoDock Vina 1.2.0	MM/PBSA Scoring	X	X
122	AutoDock Vina 1.2.0	MultiGrid FPS Scoring	X	X
123	AutoDock Vina 1.2.0	NNScore 2.0	X	✓
124	AutoDock Vina 1.2.0	OpenBPMD	X	X
125	AutoDock Vina 1.2.0	PB/SA Scoring	X	X
126	AutoDock Vina 1.2.0	PLANTS Scoring	X	X
127	AutoDock Vina 1.2.0	PLP	X	X
128	AutoDock Vina 1.2.0	PLP95	X	X
129	AutoDock Vina 1.2.0	Pharmacophore Matching Similarity Scoring	X	X
130	AutoDock Vina 1.2.0	RF-Score-VS	X	X
131	AutoDock Vina 1.2.0	Rosetta Scoring	X	X
132	AutoDock Vina 1.2.0	SASA Scoring	X	X
133	AutoDock Vina 1.2.0	SEED Scoring	X	X
134	AutoDock Vina 1.2.0	SMINA Scoring	X	X
135	AutoDock Vina 1.2.0	Vinardo Scoring	X	X
136	AutoDock Vina 1.2.0	Zou GB/SA Scoring	X	X
137	AutoDock Vina 1.1.2	AD4 Scoring	X	X
138	AutoDock Vina 1.1.2	ASP Scoring	X	X
139	AutoDock Vina 1.1.2	AutoDock Vina 1.1.2	X	X
140	AutoDock Vina 1.1.2	Bump Filter Scoring	X	X
141	AutoDock Vina 1.1.2	ChemPLP	X	X
142	AutoDock Vina 1.1.2	ChemScore	X	X
143	AutoDock Vina 1.1.2	Contact Scoring	X	X
144	AutoDock Vina 1.1.2	Continuous Scoring	X	X
145	AutoDock Vina 1.1.2	DOCK 3.5 Scoring	X	X
146	AutoDock Vina 1.1.2	DeepAffinity	X	✓
147	AutoDock Vina 1.1.2	DeepBindRG	X	✓
148	AutoDock Vina 1.1.2	Footprint Scoring	X	X
149	AutoDock Vina 1.1.2	GOLD Scoring	X	X
150	AutoDock Vina 1.1.2	Gnina Scoring	X	✓
151	AutoDock Vina 1.1.2	GoldScore	X	X
152	AutoDock Vina 1.1.2	Grid-Based Scoring	X	X
153	AutoDock Vina 1.1.2	Hawkins GB/SA Scoring	X	X
154	AutoDock Vina 1.1.2	MM/GBSA Scoring	X	X
155	AutoDock Vina 1.1.2	MM/PBSA Scoring	X	X
156	AutoDock Vina 1.1.2	MultiGrid FPS Scoring	X	X
157	AutoDock Vina 1.1.2	NNScore 2.0	X	✓
158	AutoDock Vina 1.1.2	OpenBPMD	X	X
159	AutoDock Vina 1.1.2	PB/SA Scoring	X	X
160	AutoDock Vina 1.1.2	PLANTS Scoring	X	X
161	AutoDock Vina 1.1.2	PLP	X	X
162	AutoDock Vina 1.1.2	PLP95	X	X
163	AutoDock Vina 1.1.2	Pharmacophore Matching Similarity Scoring	X	X
164	AutoDock Vina 1.1.2	RF-Score-VS	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
165	AutoDock Vina 1.1.2	Rosetta Scoring	X	X
166	AutoDock Vina 1.1.2	SASA Scoring	X	X
167	AutoDock Vina 1.1.2	SEED Scoring	X	X
168	AutoDock Vina 1.1.2	SMINA Scoring	X	X
169	AutoDock Vina 1.1.2	Vinardo Scoring	X	X
170	AutoDock Vina 1.1.2	Zou GB/SA Scoring	X	X
171	ADFR	AD4 Scoring	X	X
172	ADFR	ADFR	X	X
173	ADFR	ASP Scoring	X	X
174	ADFR	Bump Filter Scoring	X	X
175	ADFR	ChemPLP	X	X
176	ADFR	ChemScore	X	X
177	ADFR	Contact Scoring	X	X
178	ADFR	Continuous Scoring	X	X
179	ADFR	DOCK 3.5 Scoring	X	X
180	ADFR	DeepAffinity	X	✓
181	ADFR	DeepBindRG	X	✓
182	ADFR	Footprint Scoring	X	X
183	ADFR	GOLD Scoring	X	X
184	ADFR	Gnina Scoring	X	✓
185	ADFR	GoldScore	X	X
186	ADFR	Grid-Based Scoring	X	X
187	ADFR	Hawkins GB/SA Scoring	X	X
188	ADFR	MM/GBSA Scoring	X	X
189	ADFR	MM/PBSA Scoring	X	X
190	ADFR	MultiGrid FPS Scoring	X	X
191	ADFR	NNScore 2.0	X	✓
192	ADFR	OpenBPMD	X	X
193	ADFR	PB/SA Scoring	X	X
194	ADFR	PLANTS Scoring	X	X
195	ADFR	PLP	X	X
196	ADFR	PLP95	X	X
197	ADFR	Pharmacophore Matching Similarity Scoring	X	X
198	ADFR	RF-Score-VS	X	X
199	ADFR	Rosetta Scoring	X	X
200	ADFR	SASA Scoring	X	X
201	ADFR	SEED Scoring	X	X
202	ADFR	SMINA Scoring	X	X
203	ADFR	Vina scoring	X	X
204	ADFR	Vinardo Scoring	X	X
205	ADFR	Zou GB/SA Scoring	X	X
206	AutoDock Koto	AD4 Scoring	X	X
207	AutoDock Koto	ASP Scoring	X	X
208	AutoDock Koto	AutoDock Koto	X	X
209	AutoDock Koto	Bump Filter Scoring	X	X
210	AutoDock Koto	ChemPLP	X	X
211	AutoDock Koto	ChemScore	X	X
212	AutoDock Koto	Contact Scoring	X	X
213	AutoDock Koto	Continuous Scoring	X	X
214	AutoDock Koto	DOCK 3.5 Scoring	X	X
215	AutoDock Koto	DeepAffinity	X	✓
216	AutoDock Koto	DeepBindRG	X	✓
217	AutoDock Koto	Footprint Scoring	X	X
218	AutoDock Koto	GOLD Scoring	X	X
219	AutoDock Koto	Gnina Scoring	X	✓

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
220	AutoDock Koto	GoldScore	X	X
221	AutoDock Koto	Grid-Based Scoring	X	X
222	AutoDock Koto	Hawkins GB/SA Scoring	X	X
223	AutoDock Koto	MM/GBSA Scoring	X	X
224	AutoDock Koto	MM/PBSA Scoring	X	X
225	AutoDock Koto	MultiGrid FPS Scoring	X	X
226	AutoDock Koto	NNScore 2.0	X	✓
227	AutoDock Koto	OpenBPMD	X	X
228	AutoDock Koto	PB/SA Scoring	X	X
229	AutoDock Koto	PLANTS Scoring	X	X
230	AutoDock Koto	PLP	X	X
231	AutoDock Koto	PLP95	X	X
232	AutoDock Koto	Pharmacophore Matching Similarity Scoring	X	X
233	AutoDock Koto	RF-Score-VS	X	X
234	AutoDock Koto	Rosetta Scoring	X	X
235	AutoDock Koto	SASA Scoring	X	X
236	AutoDock Koto	SEED Scoring	X	X
237	AutoDock Koto	SMINA Scoring	X	X
238	AutoDock Koto	Vina scoring	X	X
239	AutoDock Koto	Vinardo Scoring	X	X
240	AutoDock Koto	Zou GB/SA Scoring	X	X
241	ATPdock	AD4 Scoring	X	X
242	ATPdock	ASP Scoring	X	X
243	ATPdock	ATPdock	X	X
244	ATPdock	Bump Filter Scoring	X	X
245	ATPdock	ChemPLP	X	X
246	ATPdock	ChemScore	X	X
247	ATPdock	Contact Scoring	X	X
248	ATPdock	Continuous Scoring	X	X
249	ATPdock	DOCK 3.5 Scoring	X	X
250	ATPdock	DeepAffinity	X	✓
251	ATPdock	DeepBindRG	X	✓
252	ATPdock	Footprint Scoring	X	X
253	ATPdock	GOLD Scoring	X	X
254	ATPdock	Gnina Scoring	X	✓
255	ATPdock	GoldScore	X	X
256	ATPdock	Grid-Based Scoring	X	X
257	ATPdock	Hawkins GB/SA Scoring	X	X
258	ATPdock	MM/GBSA Scoring	X	X
259	ATPdock	MM/PBSA Scoring	X	X
260	ATPdock	MultiGrid FPS Scoring	X	X
261	ATPdock	NNScore 2.0	X	✓
262	ATPdock	OpenBPMD	X	X
263	ATPdock	PB/SA Scoring	X	X
264	ATPdock	PLANTS Scoring	X	X
265	ATPdock	PLP	X	X
266	ATPdock	PLP95	X	X
267	ATPdock	Pharmacophore Matching Similarity Scoring	X	X
268	ATPdock	RF-Score-VS	X	X
269	ATPdock	Rosetta Scoring	X	X
270	ATPdock	SASA Scoring	X	X
271	ATPdock	SEED Scoring	X	X
272	ATPdock	SMINA Scoring	X	X
273	ATPdock	Vina scoring	X	X
274	ATPdock	Vinardo Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
275	ATPdock	Zou GB/SA Scoring	X	X
276	CovDock	CovDock	X	X
277	DOCK 6.0	AD4 Scoring	X	X
278	DOCK 6.0	ASP Scoring	X	X
279	DOCK 6.0	Bump Filter Scoring	X	X
280	DOCK 6.0	ChemPLP	X	X
281	DOCK 6.0	ChemScore	X	X
282	DOCK 6.0	Contact Scoring	X	X
283	DOCK 6.0	Continuous Scoring	X	X
284	DOCK 6.0	DOCK 3.5 Scoring	X	X
285	DOCK 6.0	DOCK 6.0	X	X
286	DOCK 6.0	DeepAffinity	X	✓
287	DOCK 6.0	DeepBindRG	X	✓
288	DOCK 6.0	Footprint Scoring	X	X
289	DOCK 6.0	GOLD Scoring	X	X
290	DOCK 6.0	Gnina Scoring	X	✓
291	DOCK 6.0	GoldScore	X	X
292	DOCK 6.0	Grid-Based Scoring	X	X
293	DOCK 6.0	Hawkins GB/SA Scoring	X	X
294	DOCK 6.0	MM/GBSA Scoring	X	X
295	DOCK 6.0	MM/PBSA Scoring	X	X
296	DOCK 6.0	MultiGrid FPS Scoring	X	X
297	DOCK 6.0	NNScore 2.0	X	✓
298	DOCK 6.0	OpenBPMD	X	X
299	DOCK 6.0	PB/SA Scoring	X	X
300	DOCK 6.0	PLANTS Scoring	X	X
301	DOCK 6.0	PLP	X	X
302	DOCK 6.0	PLP95	X	X
303	DOCK 6.0	Pharmacophore Matching Similarity Scoring	X	X
304	DOCK 6.0	RF-Score-VS	X	X
305	DOCK 6.0	Rosetta Scoring	X	X
306	DOCK 6.0	SASA Scoring	X	X
307	DOCK 6.0	SEED Scoring	X	X
308	DOCK 6.0	SMINA Scoring	X	X
309	DOCK 6.0	Vina scoring	X	X
310	DOCK 6.0	Vinardo Scoring	X	X
311	DOCK 6.0	Zou GB/SA Scoring	X	X
312	FlexX	AD4 Scoring	X	X
313	FlexX	ASP Scoring	X	X
314	FlexX	Bump Filter Scoring	X	X
315	FlexX	ChemPLP	X	X
316	FlexX	ChemScore	X	X
317	FlexX	Contact Scoring	X	X
318	FlexX	Continuous Scoring	X	X
319	FlexX	DOCK 3.5 Scoring	X	X
320	FlexX	DeepAffinity	X	✓
321	FlexX	DeepBindRG	X	✓
322	FlexX	FlexX	X	X
323	FlexX	Footprint Scoring	X	X
324	FlexX	GOLD Scoring	X	X
325	FlexX	Gnina Scoring	X	✓
326	FlexX	GoldScore	X	X
327	FlexX	Grid-Based Scoring	X	X
328	FlexX	Hawkins GB/SA Scoring	X	X
329	FlexX	MM/GBSA Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
330	FlexX	MM/PBSA Scoring	X	X
331	FlexX	MultiGrid FPS Scoring	X	X
332	FlexX	NNScore 2.0	X	✓
333	FlexX	OpenBPMD	X	X
334	FlexX	PB/SA Scoring	X	X
335	FlexX	PLANTS Scoring	X	X
336	FlexX	PLP	X	X
337	FlexX	PLP95	X	X
338	FlexX	Pharmacophore Matching Similarity Scoring	X	X
339	FlexX	RF-Score-VS	X	X
340	FlexX	Rosetta Scoring	X	X
341	FlexX	SASA Scoring	X	X
342	FlexX	SEED Scoring	X	X
343	FlexX	SMINA Scoring	X	X
344	FlexX	Vina scoring	X	X
345	FlexX	Vinardo Scoring	X	X
346	FlexX	Zou GB/SA Scoring	X	X
347	FitDock	AD4 Scoring	X	X
348	FitDock	ASP Scoring	X	X
349	FitDock	Bump Filter Scoring	X	X
350	FitDock	ChemPLP	X	X
351	FitDock	ChemScore	X	X
352	FitDock	Contact Scoring	X	X
353	FitDock	Continuous Scoring	X	X
354	FitDock	DOCK 3.5 Scoring	X	X
355	FitDock	DeepAffinity	X	✓
356	FitDock	DeepBindRG	X	✓
357	FitDock	FitDock	X	X
358	FitDock	Footprint Scoring	X	X
359	FitDock	GOLD Scoring	X	X
360	FitDock	Gnina Scoring	X	✓
361	FitDock	GoldScore	X	X
362	FitDock	Grid-Based Scoring	X	X
363	FitDock	Hawkins GB/SA Scoring	X	X
364	FitDock	MM/GBSA Scoring	X	X
365	FitDock	MM/PBSA Scoring	X	X
366	FitDock	MultiGrid FPS Scoring	X	X
367	FitDock	NNScore 2.0	X	✓
368	FitDock	OpenBPMD	X	X
369	FitDock	PB/SA Scoring	X	X
370	FitDock	PLANTS Scoring	X	X
371	FitDock	PLP	X	X
372	FitDock	PLP95	X	X
373	FitDock	Pharmacophore Matching Similarity Scoring	X	X
374	FitDock	RF-Score-VS	X	X
375	FitDock	Rosetta Scoring	X	X
376	FitDock	SASA Scoring	X	X
377	FitDock	SEED Scoring	X	X
378	FitDock	SMINA Scoring	X	X
379	FitDock	Vina scoring	X	X
380	FitDock	Vinardo Scoring	X	X
381	FitDock	Zou GB/SA Scoring	X	X
382	GalaxyDock3	AD4 Scoring	X	X
383	GalaxyDock3	ASP Scoring	X	X
384	GalaxyDock3	Bump Filter Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
385	GalaxyDock3	ChemPLP	X	X
386	GalaxyDock3	ChemScore	X	X
387	GalaxyDock3	Contact Scoring	X	X
388	GalaxyDock3	Continuous Scoring	X	X
389	GalaxyDock3	DOCK 3.5 Scoring	X	X
390	GalaxyDock3	DeepAffinity	X	✓
391	GalaxyDock3	DeepBindRG	X	✓
392	GalaxyDock3	Footprint Scoring	X	X
393	GalaxyDock3	GOLD Scoring	X	X
394	GalaxyDock3	GalaxyDock3	X	X
395	GalaxyDock3	Gnina Scoring	X	✓
396	GalaxyDock3	GoldScore	X	X
397	GalaxyDock3	Grid-Based Scoring	X	X
398	GalaxyDock3	Hawkins GB/SA Scoring	X	X
399	GalaxyDock3	MM/GBSA Scoring	X	X
400	GalaxyDock3	MM/PBSA Scoring	X	X
401	GalaxyDock3	MultiGrid FPS Scoring	X	X
402	GalaxyDock3	NNScore 2.0	X	✓
403	GalaxyDock3	OpenBPMD	X	X
404	GalaxyDock3	PB/SA Scoring	X	X
405	GalaxyDock3	PLANTS Scoring	X	X
406	GalaxyDock3	PLP	X	X
407	GalaxyDock3	PLP95	X	X
408	GalaxyDock3	Pharmacophore Matching Similarity Scoring	X	X
409	GalaxyDock3	RF-Score-VS	X	X
410	GalaxyDock3	Rosetta Scoring	X	X
411	GalaxyDock3	SASA Scoring	X	X
412	GalaxyDock3	SEED Scoring	X	X
413	GalaxyDock3	SMINA Scoring	X	X
414	GalaxyDock3	Vina scoring	X	X
415	GalaxyDock3	Vinardo Scoring	X	X
416	GalaxyDock3	Zou GB/SA Scoring	X	X
417	Glide (HTVS)	AD4 Scoring	X	X
418	Glide (HTVS)	ASP Scoring	X	X
419	Glide (HTVS)	Bump Filter Scoring	X	X
420	Glide (HTVS)	ChemPLP	X	X
421	Glide (HTVS)	ChemScore	X	X
422	Glide (HTVS)	Contact Scoring	X	X
423	Glide (HTVS)	Continuous Scoring	X	X
424	Glide (HTVS)	DOCK 3.5 Scoring	X	X
425	Glide (HTVS)	DeepAffinity	X	✓
426	Glide (HTVS)	DeepBindRG	X	✓
427	Glide (HTVS)	Footprint Scoring	X	X
428	Glide (HTVS)	GOLD Scoring	X	X
429	Glide (HTVS)	Glide (HTVS)	X	X
430	Glide (HTVS)	Gnina Scoring	X	✓
431	Glide (HTVS)	GoldScore	X	X
432	Glide (HTVS)	Grid-Based Scoring	X	X
433	Glide (HTVS)	Hawkins GB/SA Scoring	X	X
434	Glide (HTVS)	MM/GBSA Scoring	X	X
435	Glide (HTVS)	MM/PBSA Scoring	X	X
436	Glide (HTVS)	MultiGrid FPS Scoring	X	X
437	Glide (HTVS)	NNScore 2.0	X	✓
438	Glide (HTVS)	OpenBPMD	X	X
439	Glide (HTVS)	PB/SA Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
440	Glide (HTVS)	PLANTS Scoring	X	X
441	Glide (HTVS)	PLP	X	X
442	Glide (HTVS)	PLP95	X	X
443	Glide (HTVS)	Pharmacophore Matching Similarity Scoring	X	X
444	Glide (HTVS)	RF-Score-VS	X	X
445	Glide (HTVS)	Rosetta Scoring	X	X
446	Glide (HTVS)	SASA Scoring	X	X
447	Glide (HTVS)	SEED Scoring	X	X
448	Glide (HTVS)	SMINA Scoring	X	X
449	Glide (HTVS)	Vina scoring	X	X
450	Glide (HTVS)	Vinardo Scoring	X	X
451	Glide (HTVS)	Zou GB/SA Scoring	X	X
452	Glide (SP)	AD4 Scoring	X	X
453	Glide (SP)	ASP Scoring	X	X
454	Glide (SP)	Bump Filter Scoring	X	X
455	Glide (SP)	ChemPLP	X	X
456	Glide (SP)	ChemScore	X	X
457	Glide (SP)	Contact Scoring	X	X
458	Glide (SP)	Continuous Scoring	X	X
459	Glide (SP)	DOCK 3.5 Scoring	X	X
460	Glide (SP)	DeepAffinity	X	✓
461	Glide (SP)	DeepBindRG	X	✓
462	Glide (SP)	Footprint Scoring	X	X
463	Glide (SP)	GOLD Scoring	X	X
464	Glide (SP)	Glide (SP)	X	X
465	Glide (SP)	Gnina Scoring	X	✓
466	Glide (SP)	GoldScore	X	X
467	Glide (SP)	Grid-Based Scoring	X	X
468	Glide (SP)	Hawkins GB/SA Scoring	X	X
469	Glide (SP)	MM/GBSA Scoring	X	X
470	Glide (SP)	MM/PBSA Scoring	X	X
471	Glide (SP)	MultiGrid FPS Scoring	X	X
472	Glide (SP)	NNScore 2.0	X	✓
473	Glide (SP)	OpenBPM	X	X
474	Glide (SP)	PB/SA Scoring	X	X
475	Glide (SP)	PLANTS Scoring	X	X
476	Glide (SP)	PLP	X	X
477	Glide (SP)	PLP95	X	X
478	Glide (SP)	Pharmacophore Matching Similarity Scoring	X	X
479	Glide (SP)	RF-Score-VS	X	X
480	Glide (SP)	Rosetta Scoring	X	X
481	Glide (SP)	SASA Scoring	X	X
482	Glide (SP)	SEED Scoring	X	X
483	Glide (SP)	SMINA Scoring	X	X
484	Glide (SP)	Vina scoring	X	X
485	Glide (SP)	Vinardo Scoring	X	X
486	Glide (SP)	Zou GB/SA Scoring	X	X
487	Glide (XP)	AD4 Scoring	X	X
488	Glide (XP)	ASP Scoring	X	X
489	Glide (XP)	Bump Filter Scoring	X	X
490	Glide (XP)	ChemPLP	X	X
491	Glide (XP)	ChemScore	X	X
492	Glide (XP)	Contact Scoring	X	X
493	Glide (XP)	Continuous Scoring	X	X
494	Glide (XP)	DOCK 3.5 Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
495	Glide (XP)	DeepAffinity	X	✓
496	Glide (XP)	DeepBindRG	X	✓
497	Glide (XP)	Footprint Scoring	X	X
498	Glide (XP)	GOLD Scoring	X	X
499	Glide (XP)	Glide (XP)	X	X
500	Glide (XP)	Gnina Scoring	X	✓
501	Glide (XP)	GoldScore	X	X
502	Glide (XP)	Grid-Based Scoring	X	X
503	Glide (XP)	Hawkins GB/SA Scoring	X	X
504	Glide (XP)	MM/GBSA Scoring	X	X
505	Glide (XP)	MM/PBSA Scoring	X	X
506	Glide (XP)	MultiGrid FPS Scoring	X	X
507	Glide (XP)	NNScore 2.0	X	✓
508	Glide (XP)	OpenBPMD	X	X
509	Glide (XP)	PB/SA Scoring	X	X
510	Glide (XP)	PLANTS Scoring	X	X
511	Glide (XP)	PLP	X	X
512	Glide (XP)	PLP95	X	X
513	Glide (XP)	Pharmacophore Matching Similarity Scoring	X	X
514	Glide (XP)	RF-Score-VS	X	X
515	Glide (XP)	Rosetta Scoring	X	X
516	Glide (XP)	SASA Scoring	X	X
517	Glide (XP)	SEED Scoring	X	X
518	Glide (XP)	SMINA Scoring	X	X
519	Glide (XP)	Vina scoring	X	X
520	Glide (XP)	Vinardo Scoring	X	X
521	Glide (XP)	Zou GB/SA Scoring	X	X
522	Gnina	AD4 Scoring	X	X
523	Gnina	ASP Scoring	X	X
524	Gnina	Bump Filter Scoring	X	X
525	Gnina	ChemPLP	X	X
526	Gnina	ChemScore	X	X
527	Gnina	Contact Scoring	X	X
528	Gnina	Continuous Scoring	X	X
529	Gnina	DOCK 3.5 Scoring	X	X
530	Gnina	DeepAffinity	X	✓
531	Gnina	DeepBindRG	X	✓
532	Gnina	Footprint Scoring	X	X
533	Gnina	GOLD Scoring	X	X
534	Gnina	Gnina	X	X
535	Gnina	GoldScore	X	X
536	Gnina	Grid-Based Scoring	X	X
537	Gnina	Hawkins GB/SA Scoring	X	X
538	Gnina	MM/GBSA Scoring	X	X
539	Gnina	MM/PBSA Scoring	X	X
540	Gnina	MultiGrid FPS Scoring	X	X
541	Gnina	NNScore 2.0	X	✓
542	Gnina	OpenBPMD	X	X
543	Gnina	PB/SA Scoring	X	X
544	Gnina	PLANTS Scoring	X	X
545	Gnina	PLP	X	X
546	Gnina	PLP95	X	X
547	Gnina	Pharmacophore Matching Similarity Scoring	X	X
548	Gnina	RF-Score-VS	X	X
549	Gnina	Rosetta Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
550	Gnina	SASA Scoring	X	X
551	Gnina	SEED Scoring	X	X
552	Gnina	SMINA Scoring	X	X
553	Gnina	Vina scoring	X	X
554	Gnina	Vinardo Scoring	X	X
555	Gnina	Zou GB/SA Scoring	X	X
556	GOLD	AD4 Scoring	X	X
557	GOLD	ASP Scoring	X	X
558	GOLD	Bump Filter Scoring	X	X
559	GOLD	ChemPLP	X	X
560	GOLD	ChemScore	X	X
561	GOLD	Contact Scoring	X	X
562	GOLD	Continuous Scoring	X	X
563	GOLD	DOCK 3.5 Scoring	X	X
564	GOLD	DeepAffinity	X	✓
565	GOLD	DeepBindRG	X	✓
566	GOLD	Footprint Scoring	X	X
567	GOLD	GOLD	X	X
568	GOLD	Gnina Scoring	X	✓
569	GOLD	GoldScore	X	X
570	GOLD	Grid-Based Scoring	X	X
571	GOLD	Hawkins GB/SA Scoring	X	X
572	GOLD	MM/GBSA Scoring	X	X
573	GOLD	MM/PBSA Scoring	X	X
574	GOLD	MultiGrid FPS Scoring	X	X
575	GOLD	NNScore 2.0	X	✓
576	GOLD	OpenBPM	X	X
577	GOLD	PB/SA Scoring	X	X
578	GOLD	PLANTS Scoring	X	X
579	GOLD	PLP	X	X
580	GOLD	PLP95	X	X
581	GOLD	Pharmacophore Matching Similarity Scoring	X	X
582	GOLD	RF-Score-VS	X	X
583	GOLD	Rosetta Scoring	X	X
584	GOLD	SASA Scoring	X	X
585	GOLD	SEED Scoring	X	X
586	GOLD	SMINA Scoring	X	X
587	GOLD	Vina scoring	X	X
588	GOLD	Vinardo Scoring	X	X
589	GOLD	Zou GB/SA Scoring	X	X
590	GWOVina	AD4 Scoring	X	X
591	GWOVina	ASP Scoring	X	X
592	GWOVina	Bump Filter Scoring	X	X
593	GWOVina	ChemPLP	X	X
594	GWOVina	ChemScore	X	X
595	GWOVina	Contact Scoring	X	X
596	GWOVina	Continuous Scoring	X	X
597	GWOVina	DOCK 3.5 Scoring	X	X
598	GWOVina	DeepAffinity	X	✓
599	GWOVina	DeepBindRG	X	✓
600	GWOVina	Footprint Scoring	X	X
601	GWOVina	GWOVina	X	X
602	GWOVina	Gnina Scoring	X	✓
603	GWOVina	GoldScore	X	X
604	GWOVina	Grid-Based Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
605	GWOVina	Hawkins GB/SA Scoring	X	X
606	GWOVina	MM/GBSA Scoring	X	X
607	GWOVina	MM/PBSA Scoring	X	X
608	GWOVina	MultiGrid FPS Scoring	X	X
609	GWOVina	NNScore 2.0	X	✓
610	GWOVina	OpenBPMD	X	X
611	GWOVina	PB/SA Scoring	X	X
612	GWOVina	PLANTS Scoring	X	X
613	GWOVina	PLP	X	X
614	GWOVina	PLP95	X	X
615	GWOVina	Pharmacophore Matching Similarity Scoring	X	X
616	GWOVina	RF-Score-VS	X	X
617	GWOVina	Rosetta Scoring	X	X
618	GWOVina	SASA Scoring	X	X
619	GWOVina	SEED Scoring	X	X
620	GWOVina	SMINA Scoring	X	X
621	GWOVina	Vina scoring	X	X
622	GWOVina	Vinardo Scoring	X	X
623	GWOVina	Zou GB/SA Scoring	X	X
624	HDOCK	HDOCK	X	X
625	HSYMDOCK	HSYMDOCK	X	X
626	idock	AD4 Scoring	X	X
627	idock	ASP Scoring	X	X
628	idock	Bump Filter Scoring	X	X
629	idock	ChemPLP	X	X
630	idock	ChemScore	X	X
631	idock	Contact Scoring	X	X
632	idock	Continuous Scoring	X	X
633	idock	DOCK 3.5 Scoring	X	X
634	idock	DeepAffinity	X	✓
635	idock	DeepBindRG	X	✓
636	idock	Footprint Scoring	X	X
637	idock	GOLD Scoring	X	X
638	idock	Gnina Scoring	X	✓
639	idock	GoldScore	X	X
640	idock	Grid-Based Scoring	X	X
641	idock	Hawkins GB/SA Scoring	X	X
642	idock	MM/GBSA Scoring	X	X
643	idock	MM/PBSA Scoring	X	X
644	idock	MultiGrid FPS Scoring	X	X
645	idock	NNScore 2.0	X	✓
646	idock	OpenBPMD	X	X
647	idock	PB/SA Scoring	X	X
648	idock	PLANTS Scoring	X	X
649	idock	PLP	X	X
650	idock	PLP95	X	X
651	idock	Pharmacophore Matching Similarity Scoring	X	X
652	idock	RF-Score-VS	X	X
653	idock	Rosetta Scoring	X	X
654	idock	SASA Scoring	X	X
655	idock	SEED Scoring	X	X
656	idock	SMINA Scoring	X	X
657	idock	Vinardo Scoring	X	X
658	idock	Zou GB/SA Scoring	X	X
659	idock	idock	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
660	iGEMDOCK	AD4 Scoring	X	X
661	iGEMDOCK	ASP Scoring	X	X
662	iGEMDOCK	Bump Filter Scoring	X	X
663	iGEMDOCK	ChemPLP	X	X
664	iGEMDOCK	ChemScore	X	X
665	iGEMDOCK	Contact Scoring	X	X
666	iGEMDOCK	Continuous Scoring	X	X
667	iGEMDOCK	DOCK 3.5 Scoring	X	X
668	iGEMDOCK	DeepAffinity	X	✓
669	iGEMDOCK	DeepBindRG	X	✓
670	iGEMDOCK	Footprint Scoring	X	X
671	iGEMDOCK	GOLD Scoring	X	X
672	iGEMDOCK	Gnina Scoring	X	✓
673	iGEMDOCK	GoldScore	X	X
674	iGEMDOCK	Grid-Based Scoring	X	X
675	iGEMDOCK	Hawkins GB/SA Scoring	X	X
676	iGEMDOCK	MM/GBSA Scoring	X	X
677	iGEMDOCK	MM/PBSA Scoring	X	X
678	iGEMDOCK	MultiGrid FPS Scoring	X	X
679	iGEMDOCK	NNScore 2.0	X	✓
680	iGEMDOCK	OpenBPM	X	X
681	iGEMDOCK	PB/SA Scoring	X	X
682	iGEMDOCK	PLANTS Scoring	X	X
683	iGEMDOCK	PLP	X	X
684	iGEMDOCK	PLP95	X	X
685	iGEMDOCK	Pharmacophore Matching Similarity Scoring	X	X
686	iGEMDOCK	RF-Score-VS	X	X
687	iGEMDOCK	Rosetta Scoring	X	X
688	iGEMDOCK	SASA Scoring	X	X
689	iGEMDOCK	SEED Scoring	X	X
690	iGEMDOCK	SMINA Scoring	X	X
691	iGEMDOCK	Vina scoring	X	X
692	iGEMDOCK	Vinardo Scoring	X	X
693	iGEMDOCK	Zou GB/SA Scoring	X	X
694	iGEMDOCK	iGEMDOCK	X	X
695	LeDock	AD4 Scoring	X	X
696	LeDock	ASP Scoring	X	X
697	LeDock	Bump Filter Scoring	X	X
698	LeDock	ChemPLP	X	X
699	LeDock	ChemScore	X	X
700	LeDock	Contact Scoring	X	X
701	LeDock	Continuous Scoring	X	X
702	LeDock	DOCK 3.5 Scoring	X	X
703	LeDock	DeepAffinity	X	✓
704	LeDock	DeepBindRG	X	✓
705	LeDock	Footprint Scoring	X	X
706	LeDock	GOLD Scoring	X	X
707	LeDock	Gnina Scoring	X	✓
708	LeDock	GoldScore	X	X
709	LeDock	Grid-Based Scoring	X	X
710	LeDock	Hawkins GB/SA Scoring	X	X
711	LeDock	LeDock	X	X
712	LeDock	MM/GBSA Scoring	X	X
713	LeDock	MM/PBSA Scoring	X	X
714	LeDock	MultiGrid FPS Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
715	LeDock	NNScore 2.0	X	✓
716	LeDock	OpenBPMD	X	X
717	LeDock	PB/SA Scoring	X	X
718	LeDock	PLANTS Scoring	X	X
719	LeDock	PLP	X	X
720	LeDock	PLP95	X	X
721	LeDock	Pharmacophore Matching Similarity Scoring	X	X
722	LeDock	RF-Score-VS	X	X
723	LeDock	Rosetta Scoring	X	X
724	LeDock	SASA Scoring	X	X
725	LeDock	SEED Scoring	X	X
726	LeDock	SMINA Scoring	X	X
727	LeDock	Vina scoring	X	X
728	LeDock	Vinardo Scoring	X	X
729	LeDock	Zou GB/SA Scoring	X	X
730	LigandFit	AD4 Scoring	X	X
731	LigandFit	ASP Scoring	X	X
732	LigandFit	Bump Filter Scoring	X	X
733	LigandFit	ChemPLP	X	X
734	LigandFit	ChemScore	X	X
735	LigandFit	Contact Scoring	X	X
736	LigandFit	Continuous Scoring	X	X
737	LigandFit	DOCK 3.5 Scoring	X	X
738	LigandFit	DeepAffinity	X	✓
739	LigandFit	DeepBindRG	X	✓
740	LigandFit	Footprint Scoring	X	X
741	LigandFit	GOLD Scoring	X	X
742	LigandFit	Gnina Scoring	X	✓
743	LigandFit	GoldScore	X	X
744	LigandFit	Grid-Based Scoring	X	X
745	LigandFit	Hawkins GB/SA Scoring	X	X
746	LigandFit	LigandFit	X	X
747	LigandFit	MM/GBSA Scoring	X	X
748	LigandFit	MM/PBSA Scoring	X	X
749	LigandFit	MultiGrid FPS Scoring	X	X
750	LigandFit	NNScore 2.0	X	✓
751	LigandFit	OpenBPMD	X	X
752	LigandFit	PB/SA Scoring	X	X
753	LigandFit	PLANTS Scoring	X	X
754	LigandFit	PLP	X	X
755	LigandFit	PLP95	X	X
756	LigandFit	Pharmacophore Matching Similarity Scoring	X	X
757	LigandFit	RF-Score-VS	X	X
758	LigandFit	Rosetta Scoring	X	X
759	LigandFit	SASA Scoring	X	X
760	LigandFit	SEED Scoring	X	X
761	LigandFit	SMINA Scoring	X	X
762	LigandFit	Vina scoring	X	X
763	LigandFit	Vinardo Scoring	X	X
764	LigandFit	Zou GB/SA Scoring	X	X
765	LightDock	LightDock	X	X
766	MDock	AD4 Scoring	X	X
767	MDock	ASP Scoring	X	X
768	MDock	Bump Filter Scoring	X	X
769	MDock	ChemPLP	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
770	MDock	ChemScore	X	X
771	MDock	Contact Scoring	X	X
772	MDock	Continuous Scoring	X	X
773	MDock	DOCK 3.5 Scoring	X	X
774	MDock	DeepAffinity	X	✓
775	MDock	DeepBindRG	X	✓
776	MDock	Footprint Scoring	X	X
777	MDock	GOLD Scoring	X	X
778	MDock	Gnina Scoring	X	✓
779	MDock	GoldScore	X	X
780	MDock	Grid-Based Scoring	X	X
781	MDock	Hawkins GB/SA Scoring	X	X
782	MDock	MDock	X	X
783	MDock	MM/GBSA Scoring	X	X
784	MDock	MM/PBSA Scoring	X	X
785	MDock	MultiGrid FPS Scoring	X	X
786	MDock	NNScore 2.0	X	✓
787	MDock	OpenBPMD	X	X
788	MDock	PB/SA Scoring	X	X
789	MDock	PLANTS Scoring	X	X
790	MDock	PLP	X	X
791	MDock	PLP95	X	X
792	MDock	Pharmacophore Matching Similarity Scoring	X	X
793	MDock	RF-Score-VS	X	X
794	MDock	Rosetta Scoring	X	X
795	MDock	SASA Scoring	X	X
796	MDock	SEED Scoring	X	X
797	MDock	SMINA Scoring	X	X
798	MDock	Vina scoring	X	X
799	MDock	Vinardo Scoring	X	X
800	MDock	Zou GB/SA Scoring	X	X
801	MCDock	AD4 Scoring	X	X
802	MCDock	ASP Scoring	X	X
803	MCDock	Bump Filter Scoring	X	X
804	MCDock	ChemPLP	X	X
805	MCDock	ChemScore	X	X
806	MCDock	Contact Scoring	X	X
807	MCDock	Continuous Scoring	X	X
808	MCDock	DOCK 3.5 Scoring	X	X
809	MCDock	DeepAffinity	X	✓
810	MCDock	DeepBindRG	X	✓
811	MCDock	Footprint Scoring	X	X
812	MCDock	GOLD Scoring	X	X
813	MCDock	Gnina Scoring	X	✓
814	MCDock	GoldScore	X	X
815	MCDock	Grid-Based Scoring	X	X
816	MCDock	Hawkins GB/SA Scoring	X	X
817	MCDock	MCDock	X	X
818	MCDock	MM/GBSA Scoring	X	X
819	MCDock	MM/PBSA Scoring	X	X
820	MCDock	MultiGrid FPS Scoring	X	X
821	MCDock	NNScore 2.0	X	✓
822	MCDock	OpenBPMD	X	X
823	MCDock	PB/SA Scoring	X	X
824	MCDock	PLANTS Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
825	MCDock	PLP	X	X
826	MCDock	PLP95	X	X
827	MCDock	Pharmacophore Matching Similarity Scoring	X	X
828	MCDock	RF-Score-VS	X	X
829	MCDock	Rosetta Scoring	X	X
830	MCDock	SASA Scoring	X	X
831	MCDock	SEED Scoring	X	X
832	MCDock	SMINA Scoring	X	X
833	MCDock	Vina scoring	X	X
834	MCDock	Vinardo Scoring	X	X
835	MCDock	Zou GB/SA Scoring	X	X
836	Molegro	AD4 Scoring	X	X
837	Molegro	ASP Scoring	X	X
838	Molegro	Bump Filter Scoring	X	X
839	Molegro	ChemPLP	X	X
840	Molegro	ChemScore	X	X
841	Molegro	Contact Scoring	X	X
842	Molegro	Continuous Scoring	X	X
843	Molegro	DOCK 3.5 Scoring	X	X
844	Molegro	DeepAffinity	X	✓
845	Molegro	DeepBindRG	X	✓
846	Molegro	Footprint Scoring	X	X
847	Molegro	GOLD Scoring	X	X
848	Molegro	Gnina Scoring	X	✓
849	Molegro	GoldScore	X	X
850	Molegro	Grid-Based Scoring	X	X
851	Molegro	Hawkins GB/SA Scoring	X	X
852	Molegro	MM/GBSA Scoring	X	X
853	Molegro	MM/PBSA Scoring	X	X
854	Molegro	Molegro	X	X
855	Molegro	MultiGrid FPS Scoring	X	X
856	Molegro	NNScore 2.0	X	✓
857	Molegro	OpenBPMD	X	X
858	Molegro	PB/SA Scoring	X	X
859	Molegro	PLANTS Scoring	X	X
860	Molegro	PLP	X	X
861	Molegro	PLP95	X	X
862	Molegro	Pharmacophore Matching Similarity Scoring	X	X
863	Molegro	RF-Score-VS	X	X
864	Molegro	Rosetta Scoring	X	X
865	Molegro	SASA Scoring	X	X
866	Molegro	SEED Scoring	X	X
867	Molegro	SMINA Scoring	X	X
868	Molegro	Vina scoring	X	X
869	Molegro	Vinardo Scoring	X	X
870	Molegro	Zou GB/SA Scoring	X	X
871	MpsDockzn	AD4 Scoring	X	X
872	MpsDockzn	ASP Scoring	X	X
873	MpsDockzn	Bump Filter Scoring	X	X
874	MpsDockzn	ChemPLP	X	X
875	MpsDockzn	ChemScore	X	X
876	MpsDockzn	Contact Scoring	X	X
877	MpsDockzn	Continuous Scoring	X	X
878	MpsDockzn	DOCK 3.5 Scoring	X	X
879	MpsDockzn	DeepAffinity	X	✓

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
880	MpsDockzn	DeepBindRG	X	✓
881	MpsDockzn	Footprint Scoring	X	X
882	MpsDockzn	GOLD Scoring	X	X
883	MpsDockzn	Gnina Scoring	X	✓
884	MpsDockzn	GoldScore	X	X
885	MpsDockzn	Grid-Based Scoring	X	X
886	MpsDockzn	Hawkins GB/SA Scoring	X	X
887	MpsDockzn	MM/GBSA Scoring	X	X
888	MpsDockzn	MM/PBSA Scoring	X	X
889	MpsDockzn	MpsDockzn	X	X
890	MpsDockzn	MultiGrid FPS Scoring	X	X
891	MpsDockzn	NNScore 2.0	X	✓
892	MpsDockzn	OpenBPMD	X	X
893	MpsDockzn	PB/SA Scoring	X	X
894	MpsDockzn	PLANTS Scoring	X	X
895	MpsDockzn	PLP	X	X
896	MpsDockzn	PLP95	X	X
897	MpsDockzn	Pharmacophore Matching Similarity Scoring	X	X
898	MpsDockzn	RF-Score-VS	X	X
899	MpsDockzn	Rosetta Scoring	X	X
900	MpsDockzn	SASA Scoring	X	X
901	MpsDockzn	SEED Scoring	X	X
902	MpsDockzn	SMINA Scoring	X	X
903	MpsDockzn	Vina scoring	X	X
904	MpsDockzn	Vinardo Scoring	X	X
905	MpsDockzn	Zou GB/SA Scoring	X	X
906	PSOVina	AD4 Scoring	X	X
907	PSOVina	ASP Scoring	X	X
908	PSOVina	Bump Filter Scoring	X	X
909	PSOVina	ChemPLP	X	X
910	PSOVina	ChemScore	X	X
911	PSOVina	Contact Scoring	X	X
912	PSOVina	Continuous Scoring	X	X
913	PSOVina	DOCK 3.5 Scoring	X	X
914	PSOVina	DeepAffinity	X	✓
915	PSOVina	DeepBindRG	X	✓
916	PSOVina	Footprint Scoring	X	X
917	PSOVina	GOLD Scoring	X	X
918	PSOVina	Gnina Scoring	X	✓
919	PSOVina	GoldScore	X	X
920	PSOVina	Grid-Based Scoring	X	X
921	PSOVina	Hawkins GB/SA Scoring	X	X
922	PSOVina	MM/GBSA Scoring	X	X
923	PSOVina	MM/PBSA Scoring	X	X
924	PSOVina	MultiGrid FPS Scoring	X	X
925	PSOVina	NNScore 2.0	X	✓
926	PSOVina	OpenBPMD	X	X
927	PSOVina	PB/SA Scoring	X	X
928	PSOVina	PLANTS Scoring	X	X
929	PSOVina	PLP	X	X
930	PSOVina	PLP95	X	X
931	PSOVina	PSOVina	X	X
932	PSOVina	Pharmacophore Matching Similarity Scoring	X	X
933	PSOVina	RF-Score-VS	X	X
934	PSOVina	Rosetta Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
935	PSOVina	SASA Scoring	X	X
936	PSOVina	SEED Scoring	X	X
937	PSOVina	SMINA Scoring	X	X
938	PSOVina	Vina scoring	X	X
939	PSOVina	Vinardo Scoring	X	X
940	PSOVina	Zou GB/SA Scoring	X	X
941	PLANTS	AD4 Scoring	X	X
942	PLANTS	ASP Scoring	X	X
943	PLANTS	Bump Filter Scoring	X	X
944	PLANTS	ChemPLP	X	X
945	PLANTS	ChemScore	X	X
946	PLANTS	Contact Scoring	X	X
947	PLANTS	Continuous Scoring	X	X
948	PLANTS	DOCK 3.5 Scoring	X	X
949	PLANTS	DeepAffinity	X	✓
950	PLANTS	DeepBindRG	X	✓
951	PLANTS	Footprint Scoring	X	X
952	PLANTS	GOLD Scoring	X	X
953	PLANTS	Gnina Scoring	X	✓
954	PLANTS	GoldScore	X	X
955	PLANTS	Grid-Based Scoring	X	X
956	PLANTS	Hawkins GB/SA Scoring	X	X
957	PLANTS	MM/GBSA Scoring	X	X
958	PLANTS	MM/PBSA Scoring	X	X
959	PLANTS	MultiGrid FPS Scoring	X	X
960	PLANTS	NNScore 2.0	X	✓
961	PLANTS	OpenBPMD	X	X
962	PLANTS	PB/SA Scoring	X	X
963	PLANTS	PLANTS	X	X
964	PLANTS	PLANTS Scoring	X	X
965	PLANTS	PLP	X	X
966	PLANTS	PLP95	X	X
967	PLANTS	Pharmacophore Matching Similarity Scoring	X	X
968	PLANTS	RF-Score-VS	X	X
969	PLANTS	Rosetta Scoring	X	X
970	PLANTS	SASA Scoring	X	X
971	PLANTS	SEED Scoring	X	X
972	PLANTS	SMINA Scoring	X	X
973	PLANTS	Vina scoring	X	X
974	PLANTS	Vinardo Scoring	X	X
975	PLANTS	Zou GB/SA Scoring	X	X
976	PI-LZerD	PI-LZerD	X	X
977	PIPER	PIPER	X	X
978	QuickVina 2	AD4 Scoring	X	X
979	QuickVina 2	ASP Scoring	X	X
980	QuickVina 2	Bump Filter Scoring	X	X
981	QuickVina 2	ChemPLP	X	X
982	QuickVina 2	ChemScore	X	X
983	QuickVina 2	Contact Scoring	X	X
984	QuickVina 2	Continuous Scoring	X	X
985	QuickVina 2	DOCK 3.5 Scoring	X	X
986	QuickVina 2	DeepAffinity	X	✓
987	QuickVina 2	DeepBindRG	X	✓
988	QuickVina 2	Footprint Scoring	X	X
989	QuickVina 2	GOLD Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
990	QuickVina 2	Gnina Scoring	X	✓
991	QuickVina 2	GoldScore	X	X
992	QuickVina 2	Grid-Based Scoring	X	X
993	QuickVina 2	Hawkins GB/SA Scoring	X	X
994	QuickVina 2	MM/GBSA Scoring	X	X
995	QuickVina 2	MM/PBSA Scoring	X	X
996	QuickVina 2	MultiGrid FPS Scoring	X	X
997	QuickVina 2	NNScore 2.0	X	✓
998	QuickVina 2	OpenBPMD	X	X
999	QuickVina 2	PB/SA Scoring	X	X
1000	QuickVina 2	PLANTS Scoring	X	X
1001	QuickVina 2	PLP	X	X
1002	QuickVina 2	PLP95	X	X
1003	QuickVina 2	Pharmacophore Matching Similarity Scoring	X	X
1004	QuickVina 2	QuickVina 2	X	X
1005	QuickVina 2	RF-Score-VS	X	X
1006	QuickVina 2	Rosetta Scoring	X	X
1007	QuickVina 2	SASA Scoring	X	X
1008	QuickVina 2	SEED Scoring	X	X
1009	QuickVina 2	SMINA Scoring	X	X
1010	QuickVina 2	Vina scoring	X	X
1011	QuickVina 2	Vinardo Scoring	X	X
1012	QuickVina 2	Zou GB/SA Scoring	X	X
1013	QuickVina-W	AD4 Scoring	X	X
1014	QuickVina-W	ASP Scoring	X	X
1015	QuickVina-W	Bump Filter Scoring	X	X
1016	QuickVina-W	ChemPLP	X	X
1017	QuickVina-W	ChemScore	X	X
1018	QuickVina-W	Contact Scoring	X	X
1019	QuickVina-W	Continuous Scoring	X	X
1020	QuickVina-W	DOCK 3.5 Scoring	X	X
1021	QuickVina-W	DeepAffinity	X	✓
1022	QuickVina-W	DeepBindRG	X	✓
1023	QuickVina-W	Footprint Scoring	X	X
1024	QuickVina-W	GOLD Scoring	X	X
1025	QuickVina-W	Gnina Scoring	X	✓
1026	QuickVina-W	GoldScore	X	X
1027	QuickVina-W	Grid-Based Scoring	X	X
1028	QuickVina-W	Hawkins GB/SA Scoring	X	X
1029	QuickVina-W	MM/GBSA Scoring	X	X
1030	QuickVina-W	MM/PBSA Scoring	X	X
1031	QuickVina-W	MultiGrid FPS Scoring	X	X
1032	QuickVina-W	NNScore 2.0	X	✓
1033	QuickVina-W	OpenBPMD	X	X
1034	QuickVina-W	PB/SA Scoring	X	X
1035	QuickVina-W	PLANTS Scoring	X	X
1036	QuickVina-W	PLP	X	X
1037	QuickVina-W	PLP95	X	X
1038	QuickVina-W	Pharmacophore Matching Similarity Scoring	X	X
1039	QuickVina-W	QuickVina-W	X	X
1040	QuickVina-W	RF-Score-VS	X	X
1041	QuickVina-W	Rosetta Scoring	X	X
1042	QuickVina-W	SASA Scoring	X	X
1043	QuickVina-W	SEED Scoring	X	X
1044	QuickVina-W	SMINA Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1045	QuickVina-W	Vina scoring	X	X
1046	QuickVina-W	Vinardo Scoring	X	X
1047	QuickVina-W	Zou GB/SA Scoring	X	X
1048	QVina2-GPU	AD4 Scoring	X	X
1049	QVina2-GPU	ASP Scoring	X	X
1050	QVina2-GPU	Bump Filter Scoring	X	X
1051	QVina2-GPU	ChemPLP	X	X
1052	QVina2-GPU	ChemScore	X	X
1053	QVina2-GPU	Contact Scoring	X	X
1054	QVina2-GPU	Continuous Scoring	X	X
1055	QVina2-GPU	DOCK 3.5 Scoring	X	X
1056	QVina2-GPU	DeepAffinity	X	✓
1057	QVina2-GPU	DeepBindRG	X	✓
1058	QVina2-GPU	Footprint Scoring	X	X
1059	QVina2-GPU	GOLD Scoring	X	X
1060	QVina2-GPU	Gnina Scoring	X	✓
1061	QVina2-GPU	GoldScore	X	X
1062	QVina2-GPU	Grid-Based Scoring	X	X
1063	QVina2-GPU	Hawkins GB/SA Scoring	X	X
1064	QVina2-GPU	MM/GBSA Scoring	X	X
1065	QVina2-GPU	MM/PBSA Scoring	X	X
1066	QVina2-GPU	MultiGrid FPS Scoring	X	X
1067	QVina2-GPU	NNScore 2.0	X	✓
1068	QVina2-GPU	OpenBPMD	X	X
1069	QVina2-GPU	PB/SA Scoring	X	X
1070	QVina2-GPU	PLANTS Scoring	X	X
1071	QVina2-GPU	PLP	X	X
1072	QVina2-GPU	PLP95	X	X
1073	QVina2-GPU	Pharmacophore Matching Similarity Scoring	X	X
1074	QVina2-GPU	QVina2-GPU	X	X
1075	QVina2-GPU	RF-Score-VS	X	X
1076	QVina2-GPU	Rosetta Scoring	X	X
1077	QVina2-GPU	SASA Scoring	X	X
1078	QVina2-GPU	SEED Scoring	X	X
1079	QVina2-GPU	SMINA Scoring	X	X
1080	QVina2-GPU	Vina scoring	X	X
1081	QVina2-GPU	Vinardo Scoring	X	X
1082	QVina2-GPU	Zou GB/SA Scoring	X	X
1083	QVina2-W-GPU	AD4 Scoring	X	X
1084	QVina2-W-GPU	ASP Scoring	X	X
1085	QVina2-W-GPU	Bump Filter Scoring	X	X
1086	QVina2-W-GPU	ChemPLP	X	X
1087	QVina2-W-GPU	ChemScore	X	X
1088	QVina2-W-GPU	Contact Scoring	X	X
1089	QVina2-W-GPU	Continuous Scoring	X	X
1090	QVina2-W-GPU	DOCK 3.5 Scoring	X	X
1091	QVina2-W-GPU	DeepAffinity	X	✓
1092	QVina2-W-GPU	DeepBindRG	X	✓
1093	QVina2-W-GPU	Footprint Scoring	X	X
1094	QVina2-W-GPU	GOLD Scoring	X	X
1095	QVina2-W-GPU	Gnina Scoring	X	✓
1096	QVina2-W-GPU	GoldScore	X	X
1097	QVina2-W-GPU	Grid-Based Scoring	X	X
1098	QVina2-W-GPU	Hawkins GB/SA Scoring	X	X
1099	QVina2-W-GPU	MM/GBSA Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1100	QVina2-W-GPU	MM/PBSA Scoring	X	X
1101	QVina2-W-GPU	MultiGrid FPS Scoring	X	X
1102	QVina2-W-GPU	NNScore 2.0	X	✓
1103	QVina2-W-GPU	OpenBPMD	X	X
1104	QVina2-W-GPU	PB/SA Scoring	X	X
1105	QVina2-W-GPU	PLANTS Scoring	X	X
1106	QVina2-W-GPU	PLP	X	X
1107	QVina2-W-GPU	PLP95	X	X
1108	QVina2-W-GPU	Pharmacophore Matching Similarity Scoring	X	X
1109	QVina2-W-GPU	QVina2-W-GPU	X	X
1110	QVina2-W-GPU	RF-Score-VS	X	X
1111	QVina2-W-GPU	Rosetta Scoring	X	X
1112	QVina2-W-GPU	SASA Scoring	X	X
1113	QVina2-W-GPU	SEED Scoring	X	X
1114	QVina2-W-GPU	SMINA Scoring	X	X
1115	QVina2-W-GPU	Vina scoring	X	X
1116	QVina2-W-GPU	Vinardo Scoring	X	X
1117	QVina2-W-GPU	Zou GB/SA Scoring	X	X
1118	rDock	AD4 Scoring	X	X
1119	rDock	ASP Scoring	X	X
1120	rDock	Bump Filter Scoring	X	X
1121	rDock	ChemPLP	X	X
1122	rDock	ChemScore	X	X
1123	rDock	Contact Scoring	X	X
1124	rDock	Continuous Scoring	X	X
1125	rDock	DOCK 3.5 Scoring	X	X
1126	rDock	DeepAffinity	X	✓
1127	rDock	DeepBindRG	X	✓
1128	rDock	Footprint Scoring	X	X
1129	rDock	GOLD Scoring	X	X
1130	rDock	Gnina Scoring	X	✓
1131	rDock	GoldScore	X	X
1132	rDock	Grid-Based Scoring	X	X
1133	rDock	Hawkins GB/SA Scoring	X	X
1134	rDock	MM/GBSA Scoring	X	X
1135	rDock	MM/PBSA Scoring	X	X
1136	rDock	MultiGrid FPS Scoring	X	X
1137	rDock	NNScore 2.0	X	✓
1138	rDock	OpenBPMD	X	X
1139	rDock	PB/SA Scoring	X	X
1140	rDock	PLANTS Scoring	X	X
1141	rDock	PLP	X	X
1142	rDock	PLP95	X	X
1143	rDock	Pharmacophore Matching Similarity Scoring	X	X
1144	rDock	RF-Score-VS	X	X
1145	rDock	Rosetta Scoring	X	X
1146	rDock	SASA Scoring	X	X
1147	rDock	SEED Scoring	X	X
1148	rDock	SMINA Scoring	X	X
1149	rDock	Vina scoring	X	X
1150	rDock	Vinardo Scoring	X	X
1151	rDock	Zou GB/SA Scoring	X	X
1152	rDock	rDock	X	X
1153	Rosetta Ligand	AD4 Scoring	X	X
1154	Rosetta Ligand	ASP Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1155	Rosetta Ligand	Bump Filter Scoring	X	X
1156	Rosetta Ligand	ChemPLP	X	X
1157	Rosetta Ligand	ChemScore	X	X
1158	Rosetta Ligand	Contact Scoring	X	X
1159	Rosetta Ligand	Continuous Scoring	X	X
1160	Rosetta Ligand	DOCK 3.5 Scoring	X	X
1161	Rosetta Ligand	DeepAffinity	X	✓
1162	Rosetta Ligand	DeepBindRG	X	✓
1163	Rosetta Ligand	Footprint Scoring	X	X
1164	Rosetta Ligand	GOLD Scoring	X	X
1165	Rosetta Ligand	Gnina Scoring	X	✓
1166	Rosetta Ligand	GoldScore	X	X
1167	Rosetta Ligand	Grid-Based Scoring	X	X
1168	Rosetta Ligand	Hawkins GB/SA Scoring	X	X
1169	Rosetta Ligand	MM/GBSA Scoring	X	X
1170	Rosetta Ligand	MM/PBSA Scoring	X	X
1171	Rosetta Ligand	MultiGrid FPS Scoring	X	X
1172	Rosetta Ligand	NNScore 2.0	X	✓
1173	Rosetta Ligand	OpenBPMD	X	X
1174	Rosetta Ligand	PB/SA Scoring	X	X
1175	Rosetta Ligand	PLANTS Scoring	X	X
1176	Rosetta Ligand	PLP	X	X
1177	Rosetta Ligand	PLP95	X	X
1178	Rosetta Ligand	Pharmacophore Matching Similarity Scoring	X	X
1179	Rosetta Ligand	RF-Score-VS	X	X
1180	Rosetta Ligand	Rosetta Ligand	X	X
1181	Rosetta Ligand	SASA Scoring	X	X
1182	Rosetta Ligand	SEED Scoring	X	X
1183	Rosetta Ligand	SMINA Scoring	X	X
1184	Rosetta Ligand	Vina scoring	X	X
1185	Rosetta Ligand	Vinardo Scoring	X	X
1186	Rosetta Ligand	Zou GB/SA Scoring	X	X
1187	RLDOCK	AD4 Scoring	X	X
1188	RLDOCK	ASP Scoring	X	X
1189	RLDOCK	Bump Filter Scoring	X	X
1190	RLDOCK	ChemPLP	X	X
1191	RLDOCK	ChemScore	X	X
1192	RLDOCK	Contact Scoring	X	X
1193	RLDOCK	Continuous Scoring	X	X
1194	RLDOCK	DOCK 3.5 Scoring	X	X
1195	RLDOCK	DeepAffinity	X	✓
1196	RLDOCK	DeepBindRG	X	✓
1197	RLDOCK	Footprint Scoring	X	X
1198	RLDOCK	GOLD Scoring	X	X
1199	RLDOCK	Gnina Scoring	X	✓
1200	RLDOCK	GoldScore	X	X
1201	RLDOCK	Grid-Based Scoring	X	X
1202	RLDOCK	Hawkins GB/SA Scoring	X	X
1203	RLDOCK	MM/GBSA Scoring	X	X
1204	RLDOCK	MM/PBSA Scoring	X	X
1205	RLDOCK	MultiGrid FPS Scoring	X	X
1206	RLDOCK	NNScore 2.0	X	✓
1207	RLDOCK	OpenBPMD	X	X
1208	RLDOCK	PB/SA Scoring	X	X
1209	RLDOCK	PLANTS Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1210	RLDOCK	PLP	X	X
1211	RLDOCK	PLP95	X	X
1212	RLDOCK	Pharmacophore Matching Similarity Scoring	X	X
1213	RLDOCK	RF-Score-VS	X	X
1214	RLDOCK	RLDOCK	X	X
1215	RLDOCK	Rosetta Scoring	X	X
1216	RLDOCK	SASA Scoring	X	X
1217	RLDOCK	SEED Scoring	X	X
1218	RLDOCK	SMINA Scoring	X	X
1219	RLDOCK	Vina scoring	X	X
1220	RLDOCK	Vinardo Scoring	X	X
1221	RLDOCK	Zou GB/SA Scoring	X	X
1222	SEED	AD4 Scoring	X	X
1223	SEED	ASP Scoring	X	X
1224	SEED	Bump Filter Scoring	X	X
1225	SEED	ChemPLP	X	X
1226	SEED	ChemScore	X	X
1227	SEED	Contact Scoring	X	X
1228	SEED	Continuous Scoring	X	X
1229	SEED	DOCK 3.5 Scoring	X	X
1230	SEED	DeepAffinity	X	✓
1231	SEED	DeepBindRG	X	✓
1232	SEED	Footprint Scoring	X	X
1233	SEED	GOLD Scoring	X	X
1234	SEED	Gnina Scoring	X	✓
1235	SEED	GoldScore	X	X
1236	SEED	Grid-Based Scoring	X	X
1237	SEED	Hawkins GB/SA Scoring	X	X
1238	SEED	MM/GBSA Scoring	X	X
1239	SEED	MM/PBSA Scoring	X	X
1240	SEED	MultiGrid FPS Scoring	X	X
1241	SEED	NNScore 2.0	X	✓
1242	SEED	OpenBPM	X	X
1243	SEED	PB/SA Scoring	X	X
1244	SEED	PLANTS Scoring	X	X
1245	SEED	PLP	X	X
1246	SEED	PLP95	X	X
1247	SEED	Pharmacophore Matching Similarity Scoring	X	X
1248	SEED	RF-Score-VS	X	X
1249	SEED	Rosetta Scoring	X	X
1250	SEED	SASA Scoring	X	X
1251	SEED	SEED	X	X
1252	SEED	SEED Scoring	X	X
1253	SEED	SMINA Scoring	X	X
1254	SEED	Vina scoring	X	X
1255	SEED	Vinardo Scoring	X	X
1256	SEED	Zou GB/SA Scoring	X	X
1257	SMINA	AD4 Scoring	X	X
1258	SMINA	ASP Scoring	X	X
1259	SMINA	Bump Filter Scoring	X	X
1260	SMINA	ChemPLP	X	X
1261	SMINA	ChemScore	X	X
1262	SMINA	Contact Scoring	X	X
1263	SMINA	Continuous Scoring	X	X
1264	SMINA	DOCK 3.5 Scoring	X	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1265	SMINA	DeepAffinity	X	✓
1266	SMINA	DeepBindRG	X	✓
1267	SMINA	Footprint Scoring	X	X
1268	SMINA	GOLD Scoring	X	X
1269	SMINA	Gnina Scoring	X	✓
1270	SMINA	GoldScore	X	X
1271	SMINA	Grid-Based Scoring	X	X
1272	SMINA	Hawkins GB/SA Scoring	X	X
1273	SMINA	MM/GBSA Scoring	X	X
1274	SMINA	MM/PBSA Scoring	X	X
1275	SMINA	MultiGrid FPS Scoring	X	X
1276	SMINA	NNScore 2.0	X	✓
1277	SMINA	OpenBPMD	X	X
1278	SMINA	PB/SA Scoring	X	X
1279	SMINA	PLANTS Scoring	X	X
1280	SMINA	PLP	X	X
1281	SMINA	PLP95	X	X
1282	SMINA	Pharmacophore Matching Similarity Scoring	X	X
1283	SMINA	RF-Score-VS	X	X
1284	SMINA	Rosetta Scoring	X	X
1285	SMINA	SASA Scoring	X	X
1286	SMINA	SEED Scoring	X	X
1287	SMINA	SMINA	X	X
1288	SMINA	SMINA Scoring	X	X
1289	SMINA	Vina scoring	X	X
1290	SMINA	Vinardo Scoring	X	X
1291	SMINA	Zou GB/SA Scoring	X	X
1292	TANKBind	AD4 Scoring	✓	X
1293	TANKBind	ASP Scoring	✓	X
1294	TANKBind	Bump Filter Scoring	✓	X
1295	TANKBind	ChemPLP	✓	X
1296	TANKBind	ChemScore	✓	X
1297	TANKBind	Contact Scoring	✓	X
1298	TANKBind	Continuous Scoring	✓	X
1299	TANKBind	DOCK 3.5 Scoring	✓	X
1300	TANKBind	DeepAffinity	✓	✓
1301	TANKBind	DeepBindRG	✓	✓
1302	TANKBind	Footprint Scoring	✓	X
1303	TANKBind	GOLD Scoring	✓	X
1304	TANKBind	Gnina Scoring	✓	✓
1305	TANKBind	GoldScore	✓	X
1306	TANKBind	Grid-Based Scoring	✓	X
1307	TANKBind	Hawkins GB/SA Scoring	✓	X
1308	TANKBind	MM/GBSA Scoring	✓	X
1309	TANKBind	MM/PBSA Scoring	✓	X
1310	TANKBind	MultiGrid FPS Scoring	✓	X
1311	TANKBind	NNScore 2.0	✓	✓
1312	TANKBind	OpenBPMD	✓	X
1313	TANKBind	PB/SA Scoring	✓	X
1314	TANKBind	PLANTS Scoring	✓	X
1315	TANKBind	PLP	✓	X
1316	TANKBind	PLP95	✓	X
1317	TANKBind	Pharmacophore Matching Similarity Scoring	✓	X
1318	TANKBind	RF-Score-VS	✓	X
1319	TANKBind	Rosetta Scoring	✓	X

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1320	TANKBind	SASA Scoring	✓	✗
1321	TANKBind	SEED Scoring	✓	✗
1322	TANKBind	SMINA Scoring	✓	✗
1323	TANKBind	TANKBind	✓	✗
1324	TANKBind	Vina scoring	✓	✗
1325	TANKBind	Vinardo Scoring	✓	✗
1326	TANKBind	Zou GB/SA Scoring	✓	✗
1327	VinaCarb	AD4 Scoring	✓	✗
1328	VinaCarb	ASP Scoring	✓	✗
1329	VinaCarb	Bump Filter Scoring	✓	✗
1330	VinaCarb	ChemPLP	✓	✗
1331	VinaCarb	ChemScore	✓	✗
1332	VinaCarb	Contact Scoring	✓	✗
1333	VinaCarb	Continuous Scoring	✓	✗
1334	VinaCarb	DOCK 3.5 Scoring	✓	✗
1335	VinaCarb	DeepAffinity	✓	✓
1336	VinaCarb	DeepBindRG	✓	✓
1337	VinaCarb	Footprint Scoring	✓	✗
1338	VinaCarb	GOLD Scoring	✓	✗
1339	VinaCarb	Gnina Scoring	✓	✓
1340	VinaCarb	GoldScore	✓	✗
1341	VinaCarb	Grid-Based Scoring	✓	✗
1342	VinaCarb	Hawkins GB/SA Scoring	✓	✗
1343	VinaCarb	MM/GBSA Scoring	✓	✗
1344	VinaCarb	MM/PBSA Scoring	✓	✗
1345	VinaCarb	MultiGrid FPS Scoring	✓	✗
1346	VinaCarb	NNScore 2.0	✓	✓
1347	VinaCarb	OpenBPMD	✓	✗
1348	VinaCarb	PB/SA Scoring	✓	✗
1349	VinaCarb	PLANTS Scoring	✓	✗
1350	VinaCarb	PLP	✓	✗
1351	VinaCarb	PLP95	✓	✗
1352	VinaCarb	Pharmacophore Matching Similarity Scoring	✓	✗
1353	VinaCarb	RF-Score-VS	✓	✗
1354	VinaCarb	Rosetta Scoring	✓	✗
1355	VinaCarb	SASA Scoring	✓	✗
1356	VinaCarb	SEED Scoring	✓	✗
1357	VinaCarb	SMINA Scoring	✓	✗
1358	VinaCarb	Vina scoring	✓	✗
1359	VinaCarb	VinaCarb	✓	✗
1360	VinaCarb	Vinardo Scoring	✓	✗
1361	VinaCarb	Zou GB/SA Scoring	✓	✗
1362	VinaXB16	AD4 Scoring	✓	✗
1363	VinaXB16	ASP Scoring	✓	✗
1364	VinaXB16	Bump Filter Scoring	✓	✗
1365	VinaXB16	ChemPLP	✓	✗
1366	VinaXB16	ChemScore	✓	✗
1367	VinaXB16	Contact Scoring	✓	✗
1368	VinaXB16	Continuous Scoring	✓	✗
1369	VinaXB16	DOCK 3.5 Scoring	✓	✗
1370	VinaXB16	DeepAffinity	✓	✓
1371	VinaXB16	DeepBindRG	✓	✓
1372	VinaXB16	Footprint Scoring	✓	✗
1373	VinaXB16	GOLD Scoring	✓	✗
1374	VinaXB16	Gnina Scoring	✓	✓

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1375	VinaXB16	GoldScore	✓	✗
1376	VinaXB16	Grid-Based Scoring	✓	✗
1377	VinaXB16	Hawkins GB/SA Scoring	✓	✗
1378	VinaXB16	MM/GBSA Scoring	✓	✗
1379	VinaXB16	MM/PBSA Scoring	✓	✗
1380	VinaXB16	MultiGrid FPS Scoring	✓	✗
1381	VinaXB16	NNScore 2.0	✓	✓
1382	VinaXB16	OpenBPMD	✓	✗
1383	VinaXB16	PB/SA Scoring	✓	✗
1384	VinaXB16	PLANTS Scoring	✓	✗
1385	VinaXB16	PLP	✓	✗
1386	VinaXB16	PLP95	✓	✗
1387	VinaXB16	Pharmacophore Matching Similarity Scoring	✓	✗
1388	VinaXB16	RF-Score-VS	✓	✗
1389	VinaXB16	Rosetta Scoring	✓	✗
1390	VinaXB16	SASA Scoring	✓	✗
1391	VinaXB16	SEED Scoring	✓	✗
1392	VinaXB16	SMINA Scoring	✓	✗
1393	VinaXB16	Vina scoring	✓	✗
1394	VinaXB16	VinaXB16	✓	✗
1395	VinaXB16	Vinardo Scoring	✓	✗
1396	VinaXB16	Zou GB/SA Scoring	✓	✗
1397	Vina-GPU	AD4 Scoring	✓	✗
1398	Vina-GPU	ASP Scoring	✓	✗
1399	Vina-GPU	Bump Filter Scoring	✓	✗
1400	Vina-GPU	ChemPLP	✓	✗
1401	Vina-GPU	ChemScore	✓	✗
1402	Vina-GPU	Contact Scoring	✓	✗
1403	Vina-GPU	Continuous Scoring	✓	✗
1404	Vina-GPU	DOCK 3.5 Scoring	✓	✗
1405	Vina-GPU	DeepAffinity	✓	✓
1406	Vina-GPU	DeepBindRG	✓	✓
1407	Vina-GPU	Footprint Scoring	✓	✗
1408	Vina-GPU	GOLD Scoring	✓	✗
1409	Vina-GPU	Gnina Scoring	✓	✓
1410	Vina-GPU	GoldScore	✓	✗
1411	Vina-GPU	Grid-Based Scoring	✓	✗
1412	Vina-GPU	Hawkins GB/SA Scoring	✓	✗
1413	Vina-GPU	MM/GBSA Scoring	✓	✗
1414	Vina-GPU	MM/PBSA Scoring	✓	✗
1415	Vina-GPU	MultiGrid FPS Scoring	✓	✗
1416	Vina-GPU	NNScore 2.0	✓	✓
1417	Vina-GPU	OpenBPMD	✓	✗
1418	Vina-GPU	PB/SA Scoring	✓	✗
1419	Vina-GPU	PLANTS Scoring	✓	✗
1420	Vina-GPU	PLP	✓	✗
1421	Vina-GPU	PLP95	✓	✗
1422	Vina-GPU	Pharmacophore Matching Similarity Scoring	✓	✗
1423	Vina-GPU	RF-Score-VS	✓	✗
1424	Vina-GPU	Rosetta Scoring	✓	✗
1425	Vina-GPU	SASA Scoring	✓	✗
1426	Vina-GPU	SEED Scoring	✓	✗
1427	Vina-GPU	SMINA Scoring	✓	✗
1428	Vina-GPU	Vina scoring	✓	✗
1429	Vina-GPU	Vina-GPU	✓	✗

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1430	Vina-GPU	Vinardo Scoring	✓	✗
1431	Vina-GPU	Zou GB/SA Scoring	✓	✗
1432	Vina-GPU-2.0	AD4 Scoring	✓	✗
1433	Vina-GPU-2.0	ASP Scoring	✓	✗
1434	Vina-GPU-2.0	Bump Filter Scoring	✓	✗
1435	Vina-GPU-2.0	ChemPLP	✓	✗
1436	Vina-GPU-2.0	ChemScore	✓	✗
1437	Vina-GPU-2.0	Contact Scoring	✓	✗
1438	Vina-GPU-2.0	Continuous Scoring	✓	✗
1439	Vina-GPU-2.0	DOCK 3.5 Scoring	✓	✗
1440	Vina-GPU-2.0	DeepAffinity	✓	✓
1441	Vina-GPU-2.0	DeepBindRG	✓	✓
1442	Vina-GPU-2.0	Footprint Scoring	✓	✗
1443	Vina-GPU-2.0	GOLD Scoring	✓	✗
1444	Vina-GPU-2.0	Gnina Scoring	✓	✓
1445	Vina-GPU-2.0	GoldScore	✓	✗
1446	Vina-GPU-2.0	Grid-Based Scoring	✓	✗
1447	Vina-GPU-2.0	Hawkins GB/SA Scoring	✓	✗
1448	Vina-GPU-2.0	MM/GBSA Scoring	✓	✗
1449	Vina-GPU-2.0	MM/PBSA Scoring	✓	✗
1450	Vina-GPU-2.0	MultiGrid FPS Scoring	✓	✗
1451	Vina-GPU-2.0	NNScore 2.0	✓	✓
1452	Vina-GPU-2.0	OpenBPM	✓	✗
1453	Vina-GPU-2.0	PB/SA Scoring	✓	✗
1454	Vina-GPU-2.0	PLANTS Scoring	✓	✗
1455	Vina-GPU-2.0	PLP	✓	✗
1456	Vina-GPU-2.0	PLP95	✓	✗
1457	Vina-GPU-2.0	Pharmacophore Matching Similarity Scoring	✓	✗
1458	Vina-GPU-2.0	RF-Score-VS	✓	✗
1459	Vina-GPU-2.0	Rosetta Scoring	✓	✗
1460	Vina-GPU-2.0	SASA Scoring	✓	✗
1461	Vina-GPU-2.0	SEED Scoring	✓	✗
1462	Vina-GPU-2.0	SMINA Scoring	✓	✗
1463	Vina-GPU-2.0	Vina scoring	✓	✗
1464	Vina-GPU-2.0	Vina-GPU-2.0	✓	✗
1465	Vina-GPU-2.0	Vinardo Scoring	✓	✗
1466	Vina-GPU-2.0	Zou GB/SA Scoring	✓	✗
1467	EnzyDock	AD4 Scoring	✓	✗
1468	EnzyDock	ASP Scoring	✓	✗
1469	EnzyDock	Bump Filter Scoring	✓	✗
1470	EnzyDock	ChemPLP	✓	✗
1471	EnzyDock	ChemScore	✓	✗
1472	EnzyDock	Contact Scoring	✓	✗
1473	EnzyDock	Continuous Scoring	✓	✗
1474	EnzyDock	DOCK 3.5 Scoring	✓	✗
1475	EnzyDock	DeepAffinity	✓	✓
1476	EnzyDock	DeepBindRG	✓	✓
1477	EnzyDock	Footprint Scoring	✓	✗
1478	EnzyDock	GOLD Scoring	✓	✗
1479	EnzyDock	Gnina Scoring	✓	✓
1480	EnzyDock	GoldScore	✓	✗
1481	EnzyDock	Grid-Based Scoring	✓	✗
1482	EnzyDock	Hawkins GB/SA Scoring	✓	✗
1483	EnzyDock	MM/GBSA Scoring	✓	✗

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not. (Continued on next page.)

Index	Pose Prediction/Sampling Method	Scoring Function	DL-Based Sampling	DL-Based Scoring
1484	EnzyDock	MM/PBSA Scoring	✓	✗
1485	EnzyDock	MultiGrid FPS Scoring	✓	✗
1486	EnzyDock	NNScore 2.0	✓	✓
1487	EnzyDock	OpenBPMD	✓	✗
1488	EnzyDock	PB/SA Scoring	✓	✗
1489	EnzyDock	PLANTS Scoring	✓	✗
1490	EnzyDock	PLP	✓	✗
1491	EnzyDock	PLP95	✓	✗
1492	EnzyDock	Pharmacophore Matching Similarity Scoring	✓	✗
1493	EnzyDock	RF-Score-VS	✓	✗
1494	EnzyDock	Rosetta Scoring	✓	✗
1495	EnzyDock	SASA Scoring	✓	✗
1496	EnzyDock	SEED Scoring	✓	✗
1497	EnzyDock	SMINA Scoring	✓	✗
1498	EnzyDock	Vina scoring	✓	✗
1499	EnzyDock	Vinardo Scoring	✓	✗
1500	EnzyDock	Zou GB/SA Scoring	✓	✗

Supplementary Table 8 (Continued). docking protocols Supported by the AFVS and AFU Modules. Each docking protocol is determined by the combination of a pose prediction method and a scoring function. For both of these components, it is indicated whether they are based on deep learning (DL) or not.

Index	File Extension	Proper Name	Preparation Method
1	acesin	ACES input format	Open Babel
2	adf	ADF cartesian input format	Open Babel
3	alc	Alchemy format	Open Babel
4	ascii	ASCII format	Open Babel
5	bgf	MSI BGF format	Open Babel
6	box	Dock 3.5 Box format	Open Babel
7	bs	Ball and Stick format	Open Babel
8	c3d1	Chem3D Cartesian 1 format	Open Babel
9	c3d2	Chem3D Cartesian 2 format	Open Babel
10	cac	CAChe MolStruct format	Open Babel
11	cacrt	Cacao Cartesian format	Open Babel
12	cache	CAChe MolStruct format	Open Babel
13	cacint	Cacao Internal format	Open Babel
14	can	Canonical SMILES format	Open Babel
15	cdjson	ChemDoodle JSON	Open Babel
16	cdxml	ChemDraw CDXML format	Open Babel
17	cht	Chemtool format	Open Babel
18	cif	Crystallographic Information File	Open Babel
19	cml	Chemical Markup Language	Open Babel
20	cmlr	CML Reaction format	Open Babel
21	com	Gaussian 98/03 Input	Open Babel
22	CONFIG	DL-POLY CONFIG	Open Babel
23	CONTCAR	VASP format	Open Babel
24	CONTFE	MDFE format	Open Babel
25	crk2d	Chemical Resource Kit diagram(2D)	Open Babel
26	crk3d	Chemical Resource Kit 3D format	Open Babel
27	csr	Accelrys/MSI Quanta CSR format	Open Babel
28	cssr	CSD CSSR format	Open Babel
29	ct	ChemDraw Connection Table format	Open Babel
30	dalmol	DALTON input format	Open Babel
31	dmol	DMol3 coordinates format	Open Babel
32	ent	Protein Data Bank format	Open Babel
33	exyz	Extended XYZ cartesian coordinates format	Open Babel
34	fa	FASTA format	Open Babel
35	fasta	FASTA format	Open Babel
36	feat	Feature format	Open Babel
37	fh	Fenske-Hall Z-Matrix format	Open Babel
38	fhiaims	FHiaims XYZ format	Open Babel
39	fix	SMILES FIX format	Open Babel
40	fps	FPS text fingerprint format (Dalke)	Open Babel
41	fpt	Fingerprint format	Open Babel
42	fract	Free Form Fractional format	Open Babel
43	fs	Fastsearch format	Open Babel
44	fsa	FASTA format	Open Babel
45	gamin	GAMESS Input	Open Babel
46	gau	Gaussian 98/03 Input	Open Babel
47	gjc	Gaussian 98/03 Input	Open Babel
48	gjf	Gaussian 98/03 Input	Open Babel
49	gpr	Ghemical format	Open Babel
50	gr96	GROMOS96 format	Open Babel
51	gro	GRO format	Open Babel
52	gukin	GAMESS-UK Input	Open Babel
53	gzmat	Gaussian Z-Matrix Input	Open Babel
54	hin	HyperChem HIN format	Open Babel
55	inchi	InChI format	Open Babel

Supplementary Table 9. Supported Chemical File Formats by AFLP for Prepared Output Ligand Library. AFLP can store the prepared output ligand library in one or more output formats. The table lists the supported formats as well as the preparation method that AFLP uses.

Index	File Extension	Proper Name	Preparation Method
56	inchikey	InChIKey	Open Babel
57	inp	GAMESS Input	Open Babel
58	jin	Jaguar input format	Open Babel
59	Impdat	The LAMMPS data format	Open Babel
60	mcdl	MCDL format	Open Babel
61	mcif	Macromolecular Crystallographic Info	Open Babel
62	MDFF	MDFF format	Open Babel
63	mdl	MDL MOL format	Open Babel
64	ml2	Sybyl Mol2 format	Open Babel
65	mmcif	Macromolecular Crystallographic Info	Open Babel
66	mmd	MacroModel format	Open Babel
67	mmod	MacroModel format	Open Babel
68	mna	Multilevel Neighborhoods of Atoms (MNA)	Open Babel
69	mol	MDL MOL format	Open Babel
70	mol2	Sybyl Mol2 format	Open Babel
71	mold	Molden format	Open Babel
72	molden	Molden format	Open Babel
73	molf	Molden format	Open Babel
74	molreport	Open Babel molecule report	Open Babel
75	mop	MOPAC Cartesian format	Open Babel
76	mopert	MOPAC Cartesian format	Open Babel
77	mopin	MOPAC Internal	Open Babel
78	mp	Molpro input format	Open Babel
79	mpc	MOPAC Cartesian format	Open Babel
80	mpd	MolPrint2D format	Open Babel
81	mpqcin	MPQC simplified input format	Open Babel
82	mrsv	Chemical Markup Language	Open Babel
83	msms	M.F. Sanner's MSMS input format	Open Babel
84	nw	NWChem input format	Open Babel
85	orcainp	ORCA input format	Open Babel
86	outmol	DMol3 coordinates format	Open Babel
87	paint	Painter format	Open Babel
88	pcjson	PubChem JSON	Open Babel
89	pcm	PCModel Format	Open Babel
90	pdb	Protein Data Bank format	Open Babel
91	pdqt	AutoDock PDBQT format	Open Babel
92	png	PNG 2D depiction	Open Babel
93	pointcloud	Point cloud on VDW surface	Open Babel
94	POSCAR	VASP format	Open Babel
95	POSFF	MDFF format	Open Babel
96	pov	POV-Ray input format	Open Babel
97	pqr	PQR format	Open Babel
98	pqs	Parallel Quantum Solutions format	Open Babel
99	qcin	Q-Chem input format	Open Babel
100	report	Open Babel report format	Open Babel
101	rsmi	Reaction SMILES format	Open Babel
102	rxn	MDL RXN format	Open Babel
103	sd	MDL MOL format	Open Babel
104	sdf	MDL MOL format	Open Babel
105	selfies	SELFIES format	SELFIES Package
106	smi	SMILES format	Open Babel
107	smi	SMILES format	AdaptiveFlow internal
108	stl	STL 3D-printing format	Open Babel
109	svg	SVG 2D depiction	Open Babel
110	sy2	Sybyl Mol2 format	Open Babel

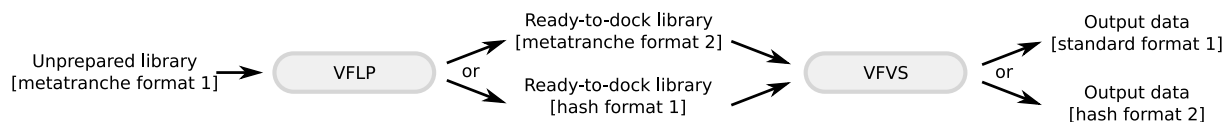
Supplementary Table 9 (Continued). Supported Chemical File Formats by AFLP for Prepared Output Ligand Library. AFLP can store the prepared output ligand library in one or more output formats. The table lists the supported formats as well as the preparation method that AFLP uses.

Index	File Extension	Proper Name	Preparation Method
111	therm	Thermo format	Open Babel
112	tmol	TurboMole Coordinate format	Open Babel
113	txt	Title format	Open Babel
114	txyz	Tinker XYZ format	Open Babel
115	unixyz	UniChem XYZ format	Open Babel
116	vmol	ViewMol format	Open Babel
117	xed	XED format	Open Babel
118	xyz	XYZ cartesian coordinates format	Open Babel
119	yob	YASARA.org YOB format	Open Babel
120	zin	ZINDO input format	Open Babel

Supplementary Table 9 (Continued). Supported Chemical File Formats by AFLP for Prepared Output Ligand Libraries. AFLP can store the prepared output ligand library in one or more output formats. The table lists the supported formats as well as the preparation method that AFLP uses.

C AdaptiveFlow File and Folder Structures

The AFLP and AFVS modules of AdaptiveFlow require input libraries and generate output data/libraries. An overview of the file structures and associated relations is shown in Supplementary Fig. 3.

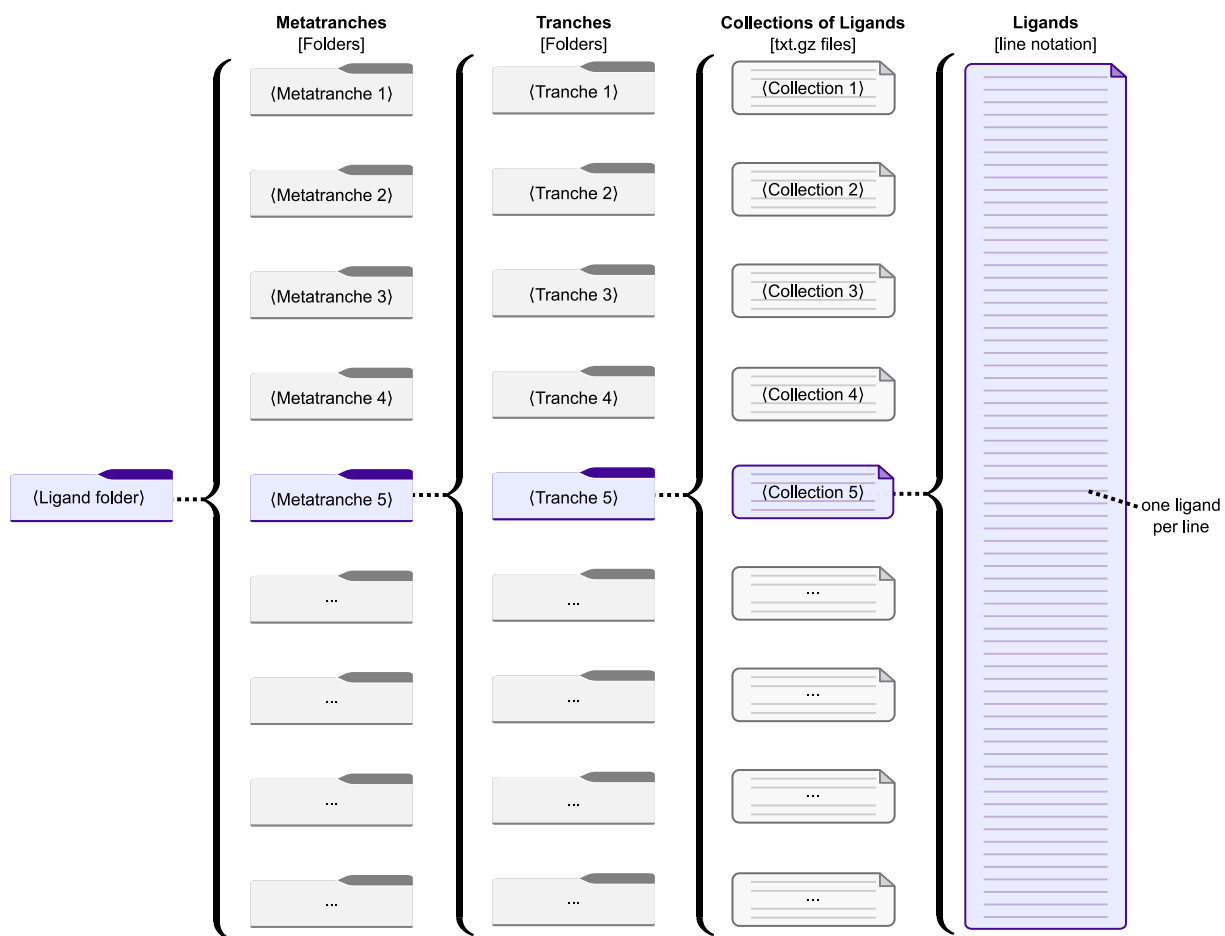


Supplementary Fig. 3. Overview of Input and Output Library Formats within AdaptiveFlow. AFLP requires that the processed input ligand library is stored in the *metatranche format 1*. AFLP can then store the output library in one of two types of formats, either the *metatranche format 2* or the *hash format 1*. These two formats for the ligand libraries can be used by AFVS, and the output data of AFVS can be stored in the *standard format 1* or the *hash format 2*.

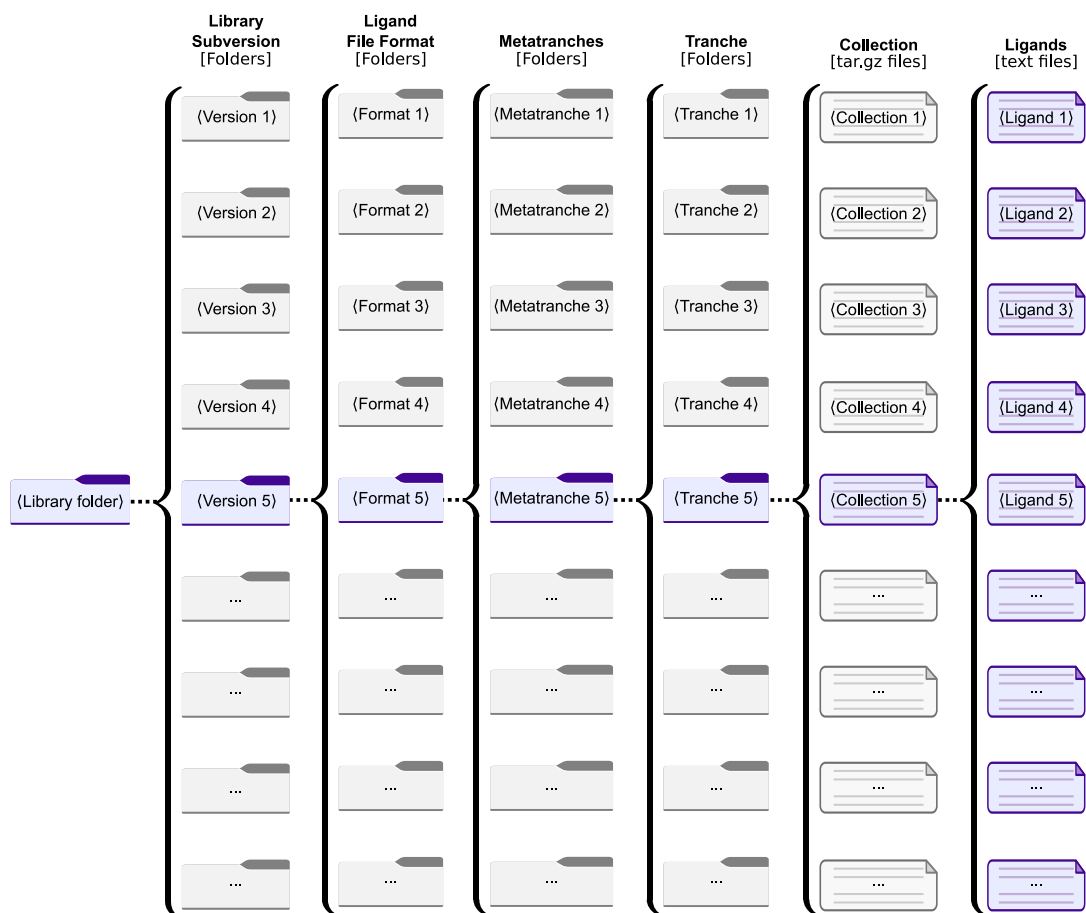
The input-library format for AFLP is the *metatranche format 1*. In the root library folder are subfolders called meta-tranches. The metatranche folders contain subfolders that are called tranches. In the tranche folders are gzipped text files, called collections, where each collection file contains any number of ligands (typically up to several thousand ligands) in line notation. Supported line notations are SMILES, SELFIES, and amino acid sequences. The *metatranche format 1* library structure is illustrated in Supplementary Fig. 4. AFLP supports two library output formats, the *metatranche format 2* (Supplementary Fig. 5), as well as the *hash format 1* (Supplementary Fig. 6). These two formats can be used by AFVS for the input ligand libraries. *Hash format 1* includes two hash prefixes/directory levels, which allow a more equal distribution of the I/O to a moderate number of different folders. This can prevent I/O issues when running massively parallel workflows.

The REAL Space library (enumerated, version 2022q1-2) that we provide is stored in the *hash format 1*. Available subversions are the complete REAL Space library, as well as a sparse version that is used by the ATG-VS method. Available ligand file formats are PDB, PDBQT, SDF, MOL2, SMILES, and SELFIES. The library structure is illustrated in Supplementary Fig. 7.

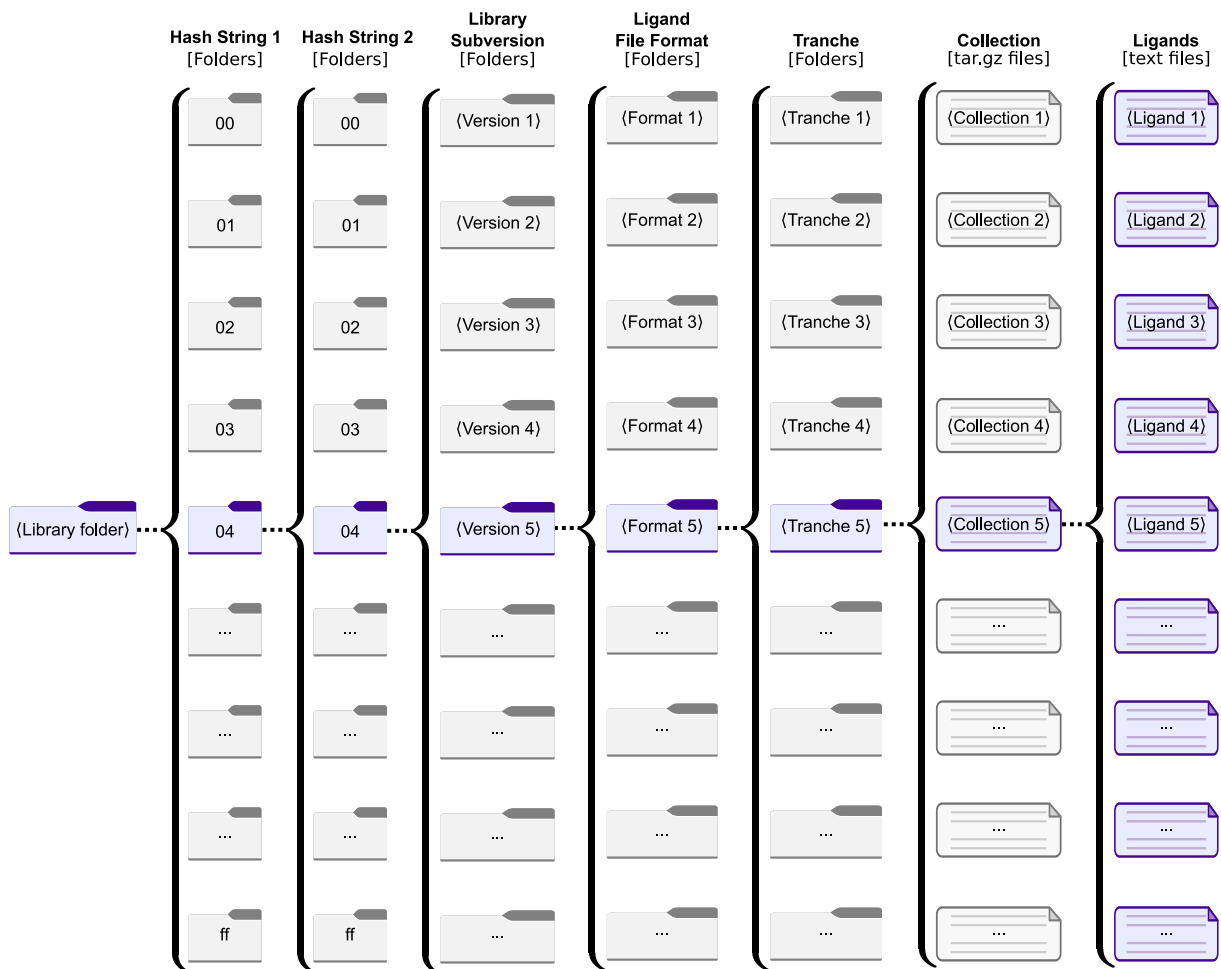
The same two formats that can be generated by AFLP can also be used by AFVS, i.e. the *metatranche format 2* and the *hash format 1*. AFVS itself can produce two types of output formats in which the output data can be stored, the *standard format* (Supplementary Fig. 8) and the *hash format 2* (Supplementary Fig. 9).



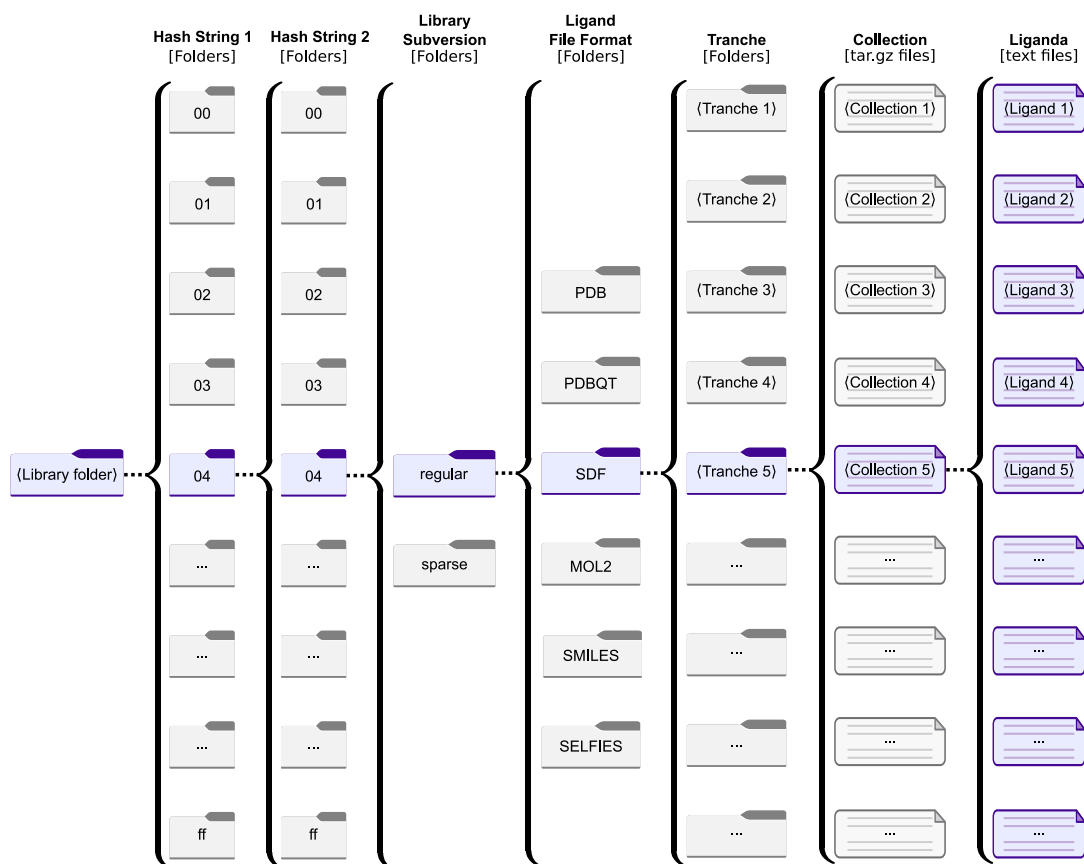
Supplementary Fig. 4. AFLP Input Library Structure (Metatranche Format 1). The *metatranche format 1* is used by AFLP as the input library format. In the library root folder are subfolders called metatranches, which contain subfolders called tranches. Each tranche folder contains one or more (ligand) collections, where each collection is a gzipped text file containing ligands in line notation (one ligand per line). Currently supported line notations are SMILES, SELFIES; and amino acid sequences.



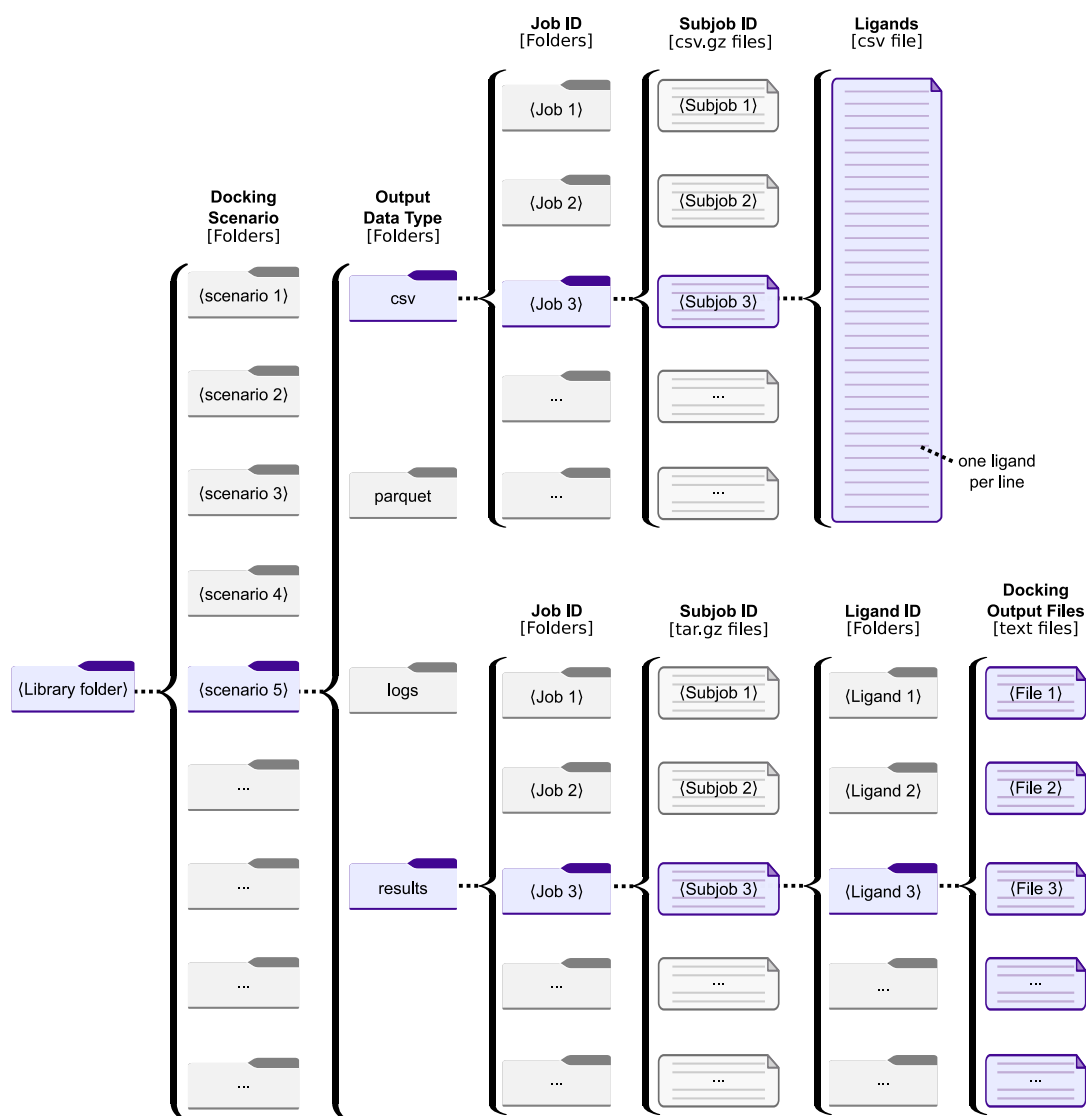
Supplementary Fig. 5. AFLP Output Library Structure and AFVS Input Library Structure (Metatranch Format 2). The *metatranch format 2* is one of the two file formats that AFLP can use for the output data ligand library and AFVS for the input ligand library. The library root folder contains subfolders for different library subversions (e.g. the full library, sparse versions, etc.). Each library subversion folder contains subfolders for the ligand file formats in which the ligands are stored. Each file format folder contains subfolders called metatranches, which themselves contain subfolders called tranches. Each tranch folder contains one or more (ligand) collections, where each collection is a gzipped tar archive containing the ligands in individual text files (one ligand per file).



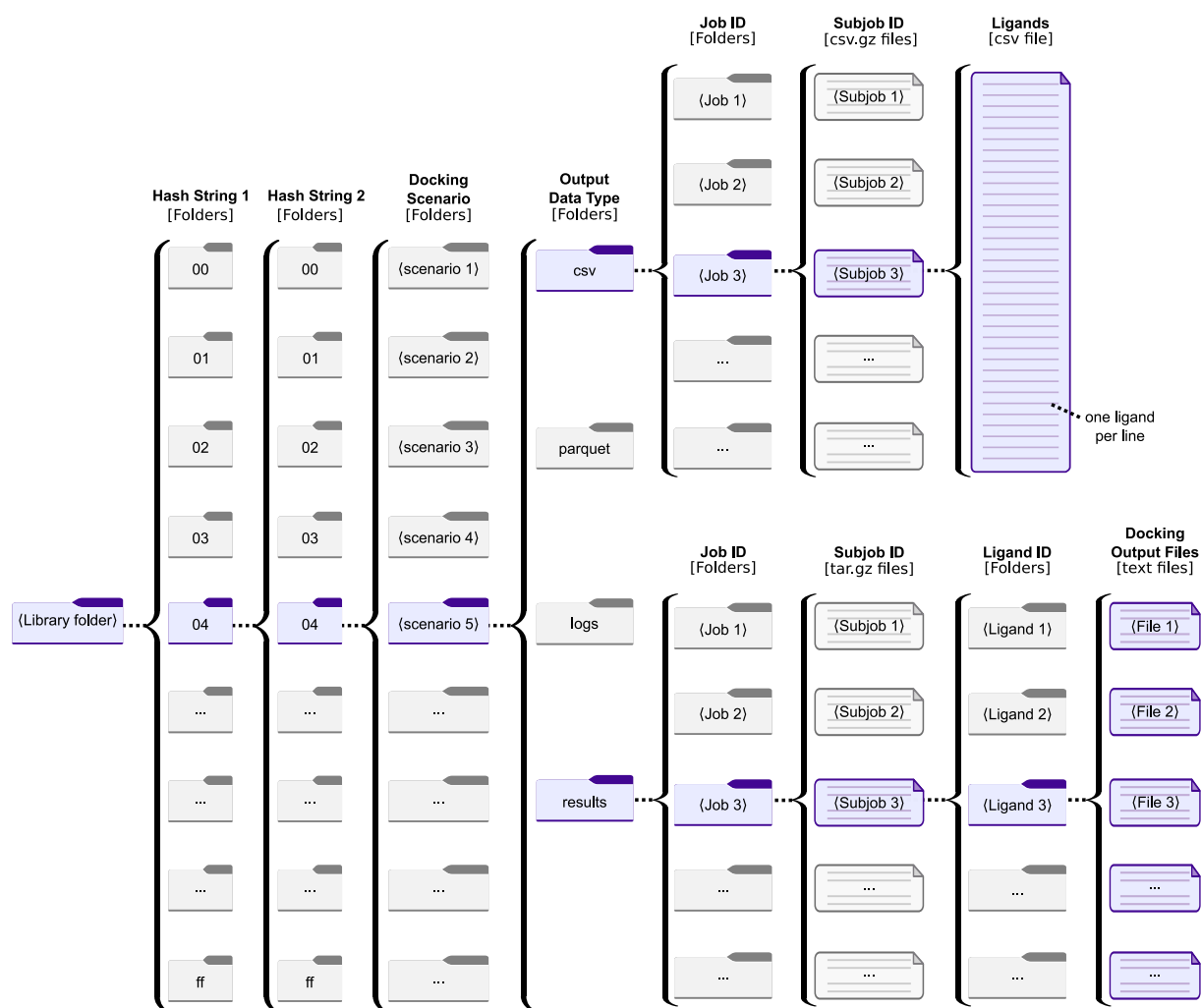
Supplementary Fig. 6. AFLP Output Library Structure and AFVS Input Library Structure (Hash Format 1). The hash format 1 is one of the two file formats that AFLP can use for the output data ligand library and AFVS for the input ligand library. The format is the same as the metatranche format 2, except that two hash subfolders are used directly within the library root folder. The hash folders each consist of two alphanumeric characters, and allow for a balanced distribution of the I/O load during highly parallel workflows, as well as allow to reduce the number of collections files per tranche folder. The hash prefixes are based on the collection and tranche IDs.



Supplementary Fig. 7. Structure of the REAL Space in AdaptiveFlow's Hash Format 1. The REAL Space that was prepared in this work was stored in hash format 1. The general hash format 1 was described in Supplementary Fig. 6. In the case of the REAL Space, there are two subversions of the library available, the regular (full) REAL Space, as well as a sparse version. Regarding the ligand file formats, six file formats are available (PDB, PDBQT, SDF, MOL2, SMILES, and SELFIES).



Supplementary Fig. 8. AFVS Output Library Structure (Standard Format 1). One of the two output data formats of AFVS is the *standard format 1*. Within the library root folder, each docking scenario has its own folder, containing subfolders for the different types of output files. The results subfolder contains the docking poses and other output files generated by the docking program. The csv and parquet subfolders contain the summary information on the dockings, such as the docking scores, in the csv and parquet formats respectively. The logs subfolder (if enabled) contains log files. Each of these folders contains subfolders for each batchsystem job, and each of them subsequently contains a file for each subjob. In the case of the parquet and the csv folders for the output data type, these files are parquet and gzipped csv files respectively. In the case of the log and the results files, these files are tar.gz files that contain folders for each ligand processed by the subjob.



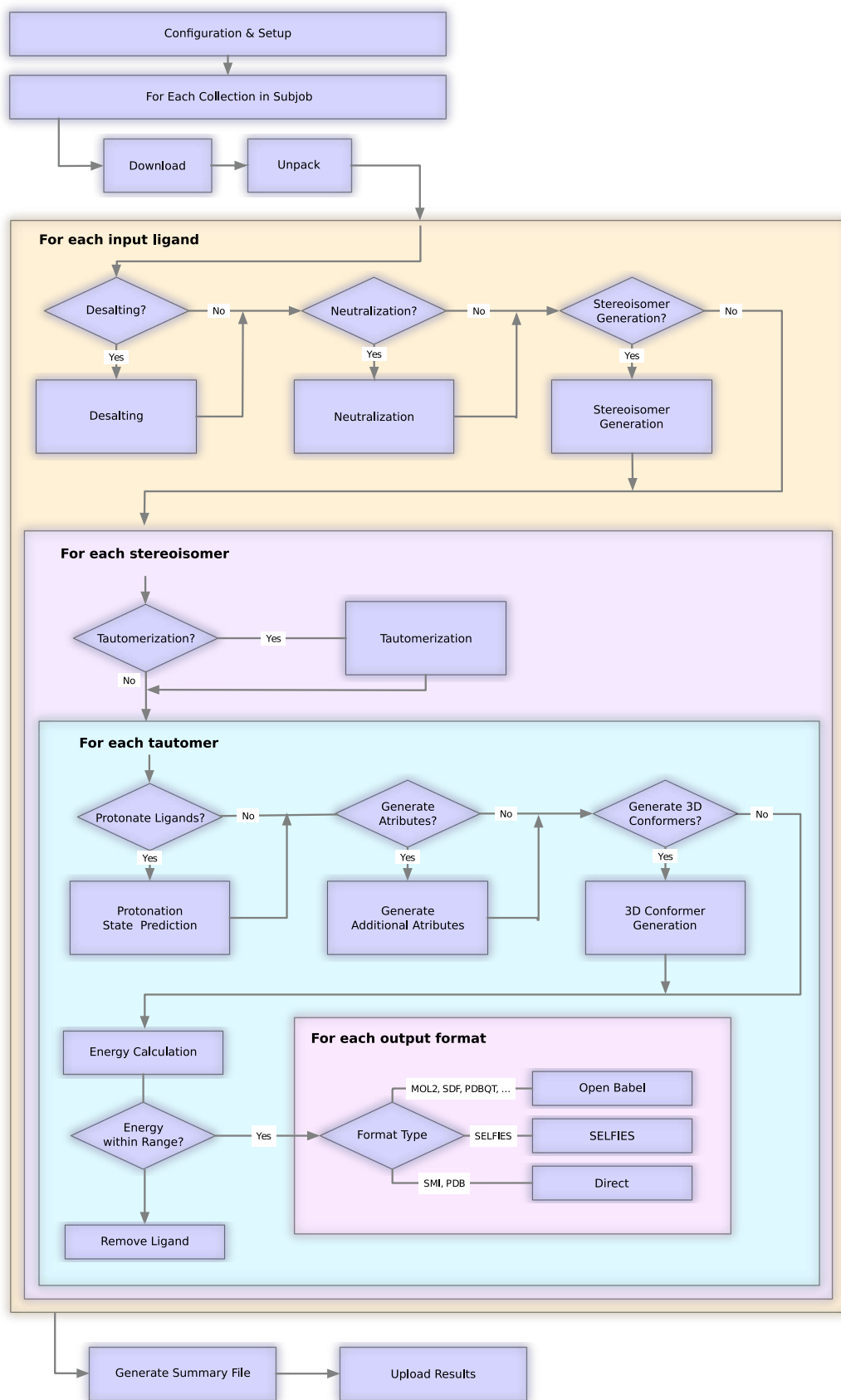
Supplementary Fig. 9. AFVS Output Library Structure (Hash Format 2). One of the two output data formats of AFVS is the *hash format 2*. The hash format 2 is principally the same format as the standard format 1, except that two hash subfolders are used directly within the library root folder. The hash folders each consist of two alphanumeric characters, and allow for a balanced distribution of the I/O load during highly parallel workflows, as well as allow to reduce the number of collections files per tranche folder. The hash prefixes are based on the collection and tranche IDs.

D AFLP Workflow

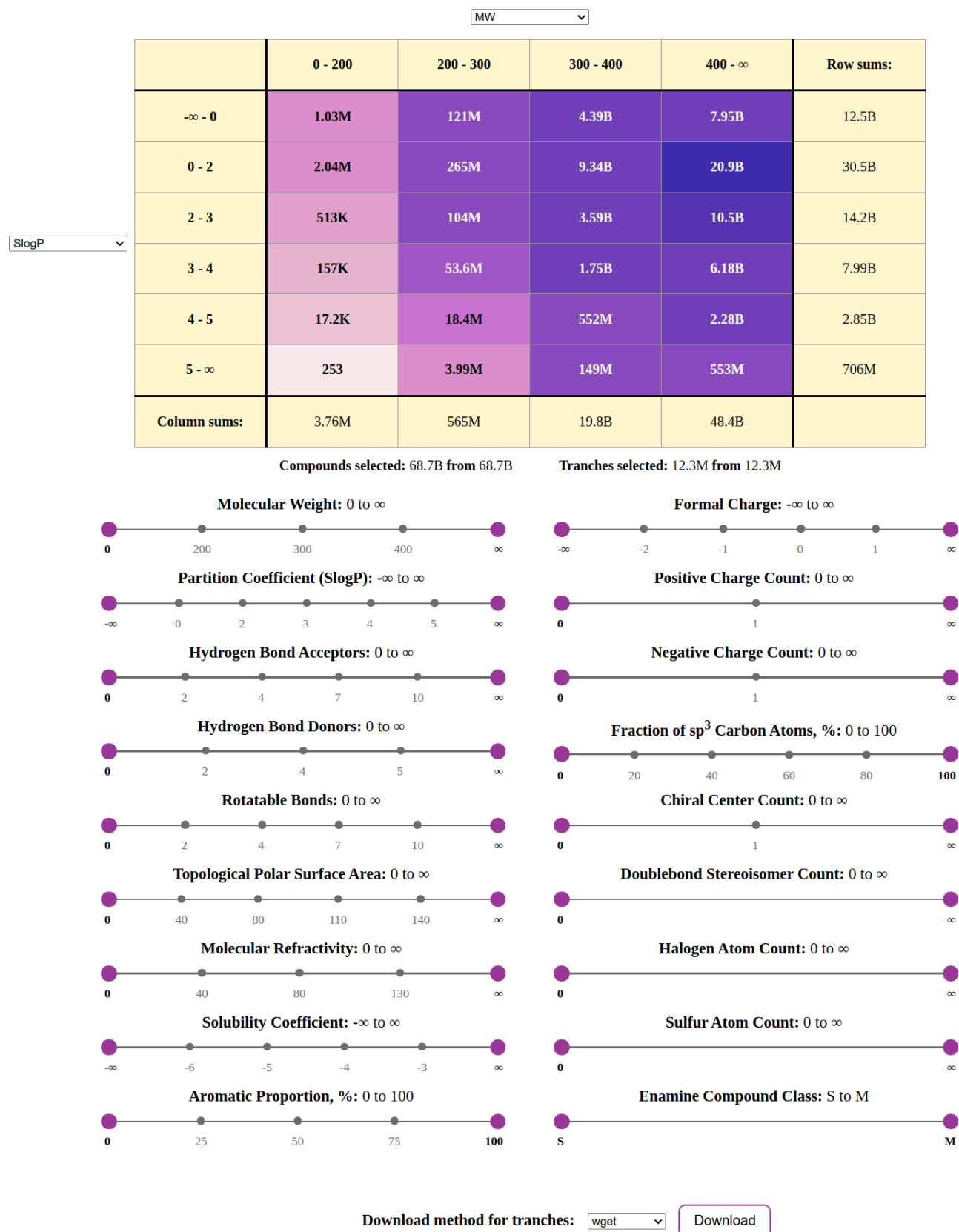
A single subjob for AFVS is defined by a set of *collections* to process and the input parameters, which describe the processing steps that should be followed for each ligand that is part of the input collections. A collection contains generally approximately 1,000 ligands and a single subjob will contain one or more of these collections. Each ligand proceeds through several steps based on the configuration that was provided for the job. Supplementary Fig. 3 shows the primary steps, with additional descriptions of each step provided below.

- **Desalting.**
- **Neutralization.**
- **Stereoisomer enumeration.** This process can generate multiple chemical species, so each of these species will process the next steps individually. In the output ligand names, a `_S<index>` will be appended to the original ligand name.
- **Tautomer enumeration.** This process can generate multiple chemical species, so each of these species will process the next steps individually. In the output ligand names, a `_T<index>` will be appended to the ligand name.
- **Attribute generation.** If requested as part of the configuration, various attributes of a ligand can be calculated within AFLP. These attributes can be generated from Open Babel, CxCalc, and RDKit.
- **Protonation state prediction.**
- **3D coordinate generation.** Up until this step, the SMILES format of the ligand has been used for processing. This step will generate a PDB file that will be used for the subsequent steps and target format generation.
- **Energy Check.** The potential energy of the ligand is calculated using Open Babel, and any ligands that exceed the configurable max energy value are discarded.
- **Target format generation.** Each target format is generated by Open Babel from the PDB that was generated as part of the previous step. Any attributes that were calculated as part of the process will be provided as comments in the PDB, PDBQT, SDF, and MOL2 formats.

After all of the ligands within a collection have been processed, AFLP will generate a summary JSON file, and archives for each of the target formats. The summary file provides information on computational timing, logging of decisions made by AFLP, and a summary of the attributes that were calculated for each of the output ligands. It also has the ability to capture intermediate logs for additional analysis. A workflow diagram of AFVS is shown in Supplementary Data Fig. 10.



Supplementary Fig. 10. Overview of the AFLP Subjob Procedure. AdaptiveFlow uses a multi-process model for each subjob where work is processed via a set of submodules (implemented as processes) through Python queues. This model allows for parallelism within Python, while avoiding the Python Global Interpreter Lock (GIL) that can otherwise prevent efficient parallelism.



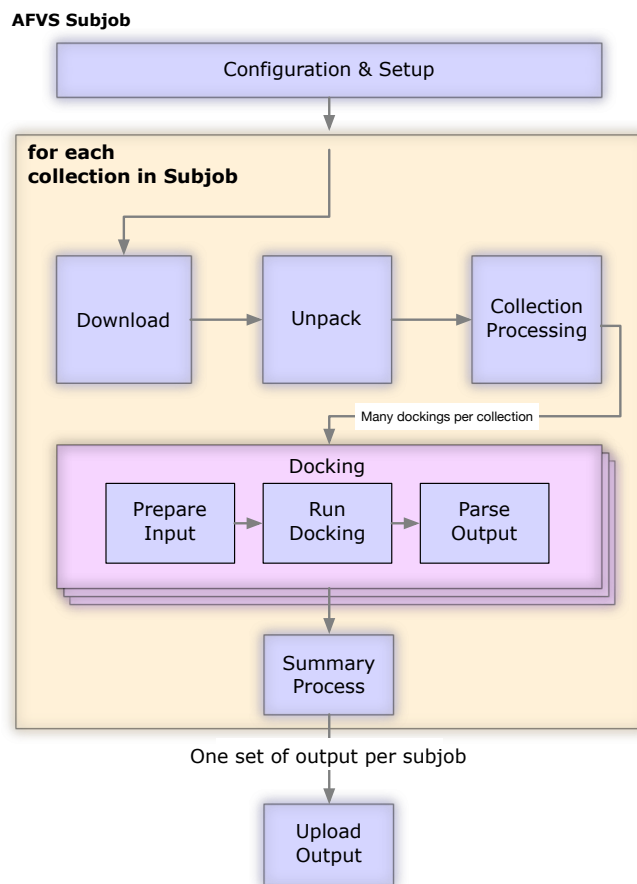
Supplementary Fig. 11. Webinterface (version 1) of the REAL Space. Shown is an interactive web interface that is made available on the AdaptiveFlow homepage (<https://adaptive-flow.ai/real-space-2022q12-interface-2>) that allows selecting subsets of the REAL Space partitioned into an 18-dimensional matrix. Each dimension corresponds to a molecular property and has a slider that can be adjusted to select the desired property ranges. The number of selected molecules is updated automatically in the web interface. Effectively, the sliders allow the selection of multidimensional rectangles within the 18-dimensional matrix.

1122 E AFVS

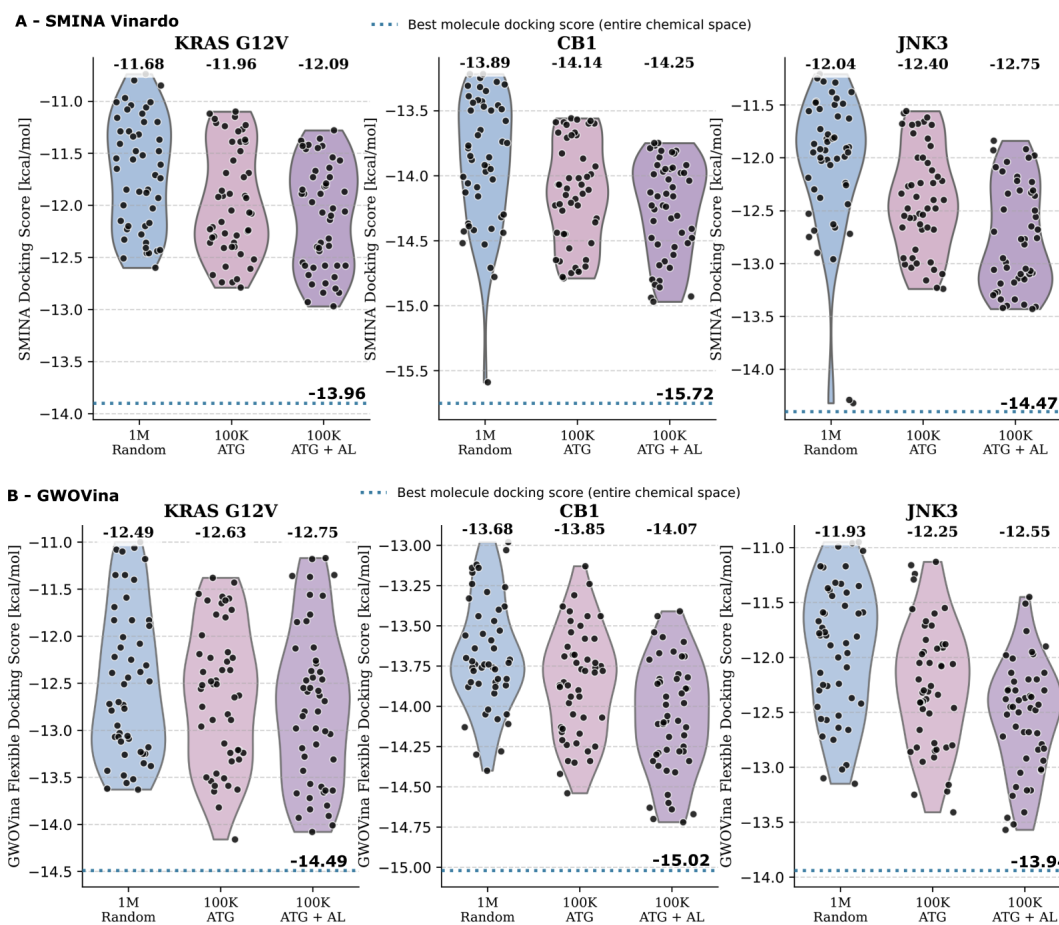
1123 A single subjob for AFVS is defined by a set of *collections* to process and the input parameters, which describe the docking
1124 steps that should be applied for each ligand that is part of the input collections. A collection is generally approximately 1,000
1125 ligands and a single subjob will contain one or more of these collections. Supplementary Fig. 12 shows the subjob modules.

- 1126 • **Download.** This module downloads the candidate ligand collections from either a shared filesystem or from object
1127 storage.
- 1128 • **Unpack.** This module expands the compressed archive to split out the individual candidate ligands within the collection
- 1129 • **Collection Processing.** This module will determine which ligands of a collection should be processed. In the case of an
1130 ATG-VS, it will remove ligands that are not needed. Additionally, this module will optionally remove ligands that may
1131 be invalid for the scenario that is being processed (e.g. the presence of specific elements). This module will then submit
1132 each candidate ligand and docking scenario to the 'Docking' module.
- 1133 • **Docking.** Processing of each individual docking (candidate ligand and receptor). Based on the docking program being
1134 used the module will construct the appropriate command line arguments and parse the output and send it along to the
1135 summary module. As there are many concurrent processes executing this is where most of the computational time is
1136 spent.
- 1137 • **Summary.** Collects the individual docking scores and data and stores in memory until all dockings for the subjob are
1138 complete (single process)
- 1139 • **Upload.** Once all of the dockings are complete, this module will upload the summary files to the results storage. (few
1140 processes)

1141 All of these are distinct pipeline stages, so they can be overlapped. For example, if a subjob includes multiple collections, the
1142 downloading of the second collection can occur while the first collection is being docked.



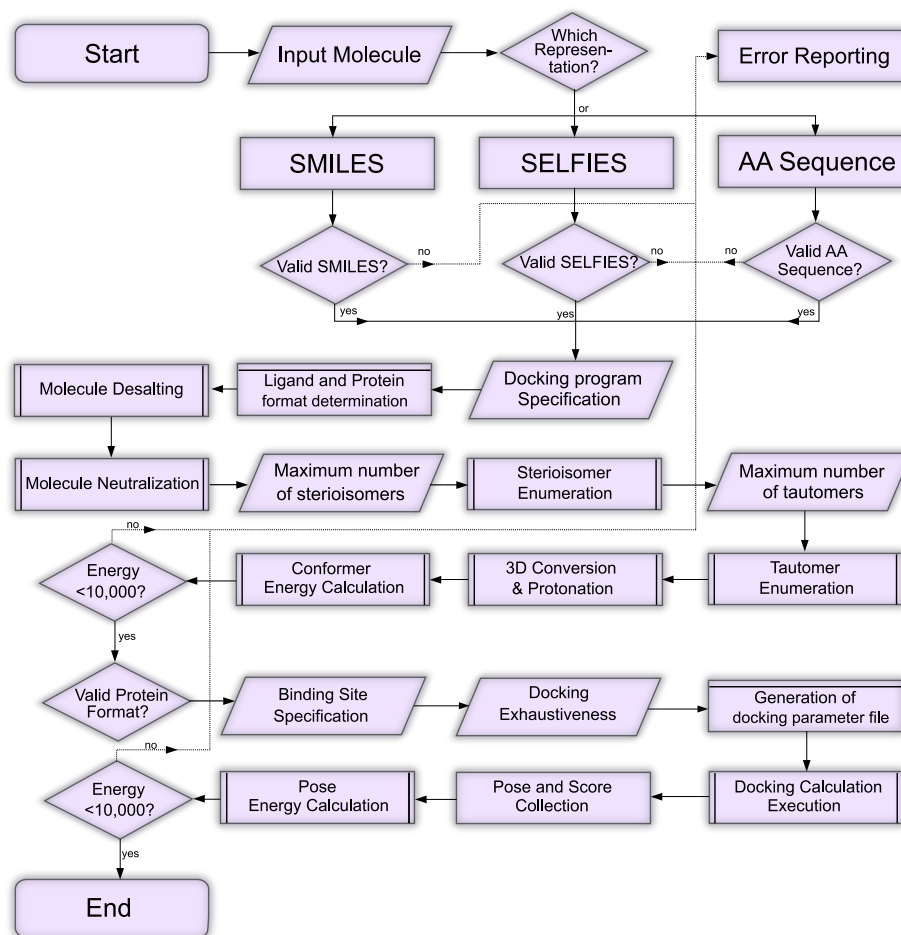
Supplementary Fig. 12. Overview of the AFVS Subjob Procedure. AdaptiveFlow uses a multi-process model for each subjob where work is processed via a set of submodules (implemented as processes) through Python queues. This model allows for parallelism within Python, while avoiding the Python Global Interpreter Lock (GIL) that can otherwise prevent efficient parallelism.



Supplementary Fig. 13. Robustness of the Adaptive Target-Guided (ATG) strategy across different docking engines. Performance comparison of the standard ATG and ATG with active learning step (ATG+AL) workflows applied to three distinct protein targets: KRAS G12V, CB1, and JNK3. Violin plots of the docking scores (the more negative, the better) for the top 50 hits obtained using (A) Smina (employing the Vinardo scoring function) and (B) GWOVina (incorporating flexible side-chain docking). The plots compare three approaches: (i) standard ultra-large virtual screening (ULVS) of 1 million random compounds (blue), (ii) adaptive target-guided (ATG) screening with 100,000 molecules (rose), and (iii) ATG screening augmented with active learning (purple). Black dots represent individual data points, and the numbers above the plots indicate the mean docking scores. The horizontal dashed lines indicate the best docking score obtained from a brute-force screen of the entire library, serving as a reference for the absolute best possible hit.

F AdaptiveFlow Unity Workflow

AFU begins by processing the ligand: desalting and neutralizing the molecule, followed by enumerating stereoisomers and enantiomers. Subsequently, OpenBabel (126) is used to convert the molecule into a 3D file format (pdb, pdbt, mol2, sdf, etc.) based on the user's specified docking software. Once the ligands are prepared, the availability of any docking parameterization file (to be provided by the user) is checked, and the calculation is executed. As output, the program returns the docked ligand pose, along with the docking score (if available) of the protein-ligand complex. A step-by-step workflow for the program is described within Fig. 14.



Supplementary Fig. 14. UML Workflow Diagram of AdaptiveFlow Unity. Initially, based on a user's specified choice of representation (SMILES, SELFIES, or amino-acid sequence), a molecule is checked for validity (by conversion back to SMILES and conversion to an RDKit mol object). Subsequently, based on the docking program specification, the supported ligand format is determined, allowing for the processing of the molecule by desalting, neutralizing, and enumerating tautomers and enantiomers. The user can control the maximum number of generated tautomers and enantiomers. The molecule is then converted into a 3D format (using OpenBabel) supported by the docking software choice at a specific protonation state (default pH 7.4). To check the successful conversion of the molecule, the energy of each 3D molecule is calculated using the obenergy program (molecules with an energy greater than 10,000 are unstable, resulting in their removal). After the successful processing of the ligand, based on the user's specified docking program, the availability of the required input file is checked. If there is no error, the docking calculation is executed, and the docking energy and the co-complexed ligand pose are returned.

G FSP1 Ligand Discovery with AdaptiveFlow

Experimental Validation Studies Involving FSP1

Materials and methods used in the experimental validation of the molecules for FSP1 are described below.

Chemicals. All chemicals and proteins employed for binding and enzymatic assays were purchased from Sigma Aldrich, except for 2,3,5-trimethylcyclohexa-2,5-diene-1,4-dione (α -tocopherolquinone hydrophilic head) purchased from Synthonix and menaquinone (Vitamin K2) from Supelco Analytical. FSP1 inhibitor candidates were purchased from Enamine; iFSP1 from Medchemexpress; viFSP1 and FSEN1 from Cayman Chemical. Inhibitors were analyzed by NMR (supplementary data). Samples were prepared by dissolving the compounds in deuterated DMSO (d6-DMSO) to 1 mM final concentration. Spectra were acquired with 100 scans on a Bruker Avance III 400 MHz spectrometer and processed using TopSpin version 4.5.0.

1159 **Phylogenetic Inference and Ancestral Sequence Reconstruction.** Phylogenetic analysis and ancestral sequence recon-
1160 struction were carried out following the methods described in (127). Briefly, human FSP1 sequence (UniprotID: Q9BRQ8) was
1161 employed as query for homology searches with BLASTP. Datasets were constructed vetting all chordate classes according to
1162 TimeTree (128) by mining at least two species with fully sequenced genomes. Multiple sequence alignments were constructed
1163 in MAFFT v7 (129) and manually trimmed for single sequence insertions/terminal extensions. Neighbour-joining guide trees
1164 were constructed in MEGA v10.2 to assess the quality of datasets under construction. Once the working multiple-sequence
1165 alignments were obtained, best-fit substitution models and gamma distribution values (α) were calculated using ProtTest (data
1166 for each protein phylogeny are shown in Supplementary Fig. 15B). Maximum likelihood phylogenies were inferred using
1167 RaxML v8.2.10 (HPC-PTHREADS module (130)), using rapid bootstrap analysis and searching for the best-scoring ML tree,
1168 with 500 bootstraps replicates and the given best-fit model under gamma distribution. When required, a species tree was used to
1169 constrain the phylogeny, this was constructed using TimeTree (timetree.org). Once the phylogenies were inferred, the bootstrap
1170 values were subjected to transfer bootstrap expectation values using BOOSTER online (131). Figtree v1.4.2 was employed for
1171 analysing and visualizing the trees. Ancestral sequence reconstruction was performed employing PAML v4.9a (CODEML mod-
1172 ule) as marginal reconstruction, using the phylogenies obtained previously, empirical amino acid substitution model (model=2)
1173 and JTT substitution matrix, four gamma categories and re-estimation of gamma shape parameter (132, 133). The distribution
1174 of the posterior probabilities for each of the ancestral states was analysed at the node corresponding to the tetrapod ancestor in
1175 the phylogeny. Sites that displayed posterior probabilities <0.8 were considered ambiguously reconstructed when alternative
1176 states displayed posterior probabilities >0.2 (134). The length of the targeted nodes was treated by Fitch's parsimony. To
1177 visualize the degree of sequence conservation across tetrapod organisms, multiple sequence alignments were generated using
1178 ESPrnt 3 (135).

1179 **Construct design and expression of FSP1.** The gene encoding for human full-length FSP1-twinSTREP-eGFP cloned
1180 into pcDNA3 was purchased from Genscript. HEK293-EBNA1-6E cells were used for the transient transfection. Cells were
1181 maintained in suspension with FreeStyle medium (Invitrogen). The day before transfection, cells were seeded at a concentration
1182 of 0.5×10^5 cells/mL. The day after (1×10^6 cells/mL), cells were transiently transfected using polyethyleneimine with a ratio
1183 of 1:3 DNA:polyethyleneimine. After 48 h, cells were harvested by centrifugation at 1,200 g for 5 min at 20 °C. Trypan
1184 Blue exclusion assay (Trypan Blue solution w/v, Sigma) was used to evaluate cell viability. The genes encoding for His₆-
1185 SUMO-ancestral FSP1, His₆-SUMO-ancestral L323A FSP1, His₆-SUMO-ancestral F15A FSP1, His₆-SUMO-ancestral F354A
1186 FSP1, lacking the Gly-2 myristoylation site, were cloned in a pET-24a(+) vector with kanamycin resistance as obtained from
1187 Genscript. Plasmids were then transformed by heat shock into *E. coli* BL21(DE3) cells (25 s at 42 °C). Cells from a single
1188 colony were picked and pre-inoculated into 100 mL LB broth containing 100 $\mu\text{g mL}^{-1}$ kanamycin and grown overnight at 37
1189 °C. 10 mL of pre-cultures were inoculated into 1 liter of Terrific Broth and grown at 37 °C, 200 r.p.m. for 3 h until the optical
1190 density (OD₆₀₀) reached 0.5-0.7. Protein expression was induced with isopropyl β -D-1-thiogalactopyranoside (0.1 mM final)
1191 and incubated overnight at 30 °C while shaking 200 r.p.m.. Cells were harvested by centrifugation (5,000 g, 15 min, 4 °C),
1192 flash frozen in liquid nitrogen, and stored at -20 °C.

1193 **Protein purification.** Cells (ca. 29 g) were re-suspended 1:5 (grams:milliliters) in Buffer A (50 mM HEPES pH 7.2, 250 mM
1194 NaCl, 10% (v/v) glycerol) supplemented with 1 mM FAD and protease inhibitors (1 mM phenylmethylsulphonyl fluoride, 10
1195 μM leupeptin, 10 μM pepstatin) and 5 μg DNase I /g cells. Cell resuspensions were stirred at 4 °C for 25 min before cell
1196 lysis, which was performed with a high-pressure homogeniser (Emulsiflex c-3, ATA Scientific) in three cycles. Cell lysates
1197 were centrifuged (4 °C, 56,000 g, 1 h) using an Avanti J-26 XP centrifuge equipped with a JA-25.15 rotor (Beckman Coulter).
1198 The resulting supernatant was filtered through a 0.45 μm filter and loaded on a gravity column containing Nickel Sepharose
1199 HP IMAC resin (Cytiva) pre-equilibrated with Buffer A. The crude extract was passed onto the column twice. The resin
1200 was then washed with at least five column volumes of Buffer A. The wash up of the resin was performed adding 5 column
1201 volumes of Buffer A supplemented with increasing concentrations of imidazole, 5 to 30 mM. The protein was then eluted with
1202 at least ten column volumes of Buffer B (Buffer A supplemented with 300 mM imidazole). The purified protein sample was
1203 concentrated using an Amicon Ultra (Merck) with a 30 kDa cut-off up to 2.5 mL. The imidazole was then removed with a
1204 PD-10 batch desalting column (Cytiva) pre-equilibrated with Buffer A. To cleave the His₆-SUMO tag, the desalted sample was
1205 concentrated down to 800 μL and then added of a His₆-tagged SUMO protease (1.2 mg mL^{-1}) to a volume ratio of 100:1 and
1206 incubated overnight at 4 °C on a rotating wheel. The sample was then loaded onto an Äkta Pure system (Cytiva) equipped
1207 with a Ni HiTrap-HP 5 mL column (Cytiva) pre-equilibrated in Buffer A and proteins were eluted out using an imidazole
1208 concentration gradient of 2% Buffer B corresponding to 6 mM imidazole. The elution of the tag-less protein was monitored
1209 with a multiwavelength detector following the absorbance at 280 nm and 450 nm. The sample was loaded onto an Äkta Pure
1210 system (Cytiva) equipped with a Superdex 200 10/300 GL (Cytiva) pre-equilibrated in 100 mM NaCl, 50 mM HEPES pH
1211 7.2. The purified protein was then concentrated down to about 1 mM, measuring the concentration using a NanoDrop ND-
1212 100 UV/Vis spectrophotometer (Thermo Scientific) based on the absorption at 280 nm and at 458 nm ($\epsilon_{280\text{ nm}}$ predicted by
1213 ProtParam tool - ExPASy). Purity of the sample was evaluated by SDS-PAGE. The purified protein was finally flash frozen in
1214 liquid nitrogen and stored at -80 °C.

Enzyme kinetics. The activity of FSP1 and mutants was monitored using 0.2-1 μM protein (estimated by 458 nm absorbance) in a 150 μL final volume of Buffer A. Reactions were followed in 10.00 mm quartz cuvettes (Hellma) and a Cary Eclipse Fluorescence Spectrophotometer (Agilent) or a Cary 100 UV-vis spectrophotometer (Varian) equipped with a thermo-stated cell holder ($T = 25^\circ\text{C}$). Reactions were started by adding NADH and rates were determined by following NADH consumption, caused by the oxidation to NAD^+ (excitation 340 nm, emission 460 nm or monitoring the absorbance at 340 nm). A calibration line was built by measuring the fluorescence at known NADH concentrations. GraphPad Prism 9 was used to perform both linear and non-linear regression.

Superoxide assay. Superoxide radicals generated by oxidation of NADH were detected by the reduction of nitroblue tetrazolium to nitroblue diformazane using Tris-HCl buffer (16 mM, pH 8.0) 50 μM nitroblue tetrazolium, and 0.2 μM final FSP1 (final volume 100 μL). The reaction was started by adding various concentrations of NADH. The absorbance was measured at 560 nm using a ClarioStar plate reader.

Enzyme inhibition. K_i were measured following NADH oxidation by monitoring the absorbance 340 nm with Cary 100 UV-vis spectrophotometer (Varian). 0.2-1 μM purified FSP1 was added to a final volume of 150 μL of 50 mM HEPES buffer pH 7.2. Inhibitors were tested with concentrations varying from micromolar to low nanomolar.

Resazurin assay. The activity of FSP1 and mutants was monitored using 0.2-0.1 μM (final, absorbance at 458 nm) protein in a 100 μL 50 mM HEPES buffer pH 7.2 and 50 μM resazurin. The activity was monitored by the formation of resorufin (excitation, 572 nm, emission 583 nm) using varying concentrations of iFSP1 and viFSP1 (10, 44, 45), afi-FSP1-1, afi-FSP1-2, afi-FSP1-3, afi-FSP1-4, afi-FSP1-5 and afi-FSP1-6. The reaction was started by adding 200-100 μM NADH. The fluorescence was read using a ClarioStar plate reader, with a gain of 800.

Nano Differential Scanning Fluorimetry. Thermo-stability analyses were carried out using a TychoTMNT.6 system (NanoTemper GmbH). The protein was diluted to 1 mg mL^{-1} in Buffer A and pre-incubated with ligands at room temperature for 15 minutes before experiments. Melting curves were obtained following the intrinsic fluorescence of tryptophan and tyrosine residues (emission 350 nm and 330 nm, respectively) applying a temperature gradient from 35°C to 95°C for a time range of three minutes. Data were analysed as F_{350}/F_{330} ratio and derived to determine the inflection temperature (T_m). All measurements were performed in duplicate. The following condition were tested: FSP1 in presence of 1 mM FAD, NAD^+ or 1 mM CoQ_1 or 1 mM α -tocopherolquinone hydrophilic head or 1 mM vitamin K3 or 10 μM inhibitors or 5% DMSO v/v as a control.

ThermoFAD. Experiments were performed with 20 μM FSP1 after 10 min incubation with inhibitors (10 μM) or 5% v/v DMSO in HEPES buffer (50 mM, pH 7.2) (136). The temperature gradient was set to $30-90^\circ\text{C}$ with fluorescence detection at a constant rate 0.1°C/s at 450 ± 30 nm excitation and 535 ± 30 nm emission (BioRad MiniOpticon Real-Time PCR System). Data were normalized and fitted with sigmoidal non-linear regression using GraphPad Prism 9.

Cellular thermal shift assays. Inhibitor afi-FSP1-1: HEK293-EBNA1-6E cells were seeded at a density of 3.0×10^7 cells/mL and incubated with either 5% v/v DMSO or 10 μM of 1 for 1 h at 37°C in 2 ml of culture medium. Cells were collected by centrifugation at $1,200 \times g$ for 5 min at room temperature, washed twice with PBS, and resuspended at the initial cell concentration. Aliquots of 50 μL were distributed into PCR strip tubes. Samples were subjected to a heat treatment at different temperatures (ranging from 38.7 to 86.2°C) using a Bio-Rad T100 Thermal Cycler. Following heat exposure, cells were lysed by adding 1% Triton X-100, protease inhibitors (see Protein purification section), and DNase I (5 μg per g of cells). Lysis was performed for 1 h at 4°C on a rotation wheel. Lysates were centrifuged at $20,000 \times g$ for 20 min at 4°C , and 15 μL of each supernatant was loaded onto Bio-Rad Mini-PROTEAN TGX SDS-PAGE gels. Proteins were transferred to 0.2 μm PVDF membranes (Bio-Rad) using the Trans-Blot Turbo system. Membranes were exposed with an anti-AIFM2 antibody (clone 1115 ZooMAb®, rabbit monoclonal, Sigma-Aldrich), followed by an HRP-conjugated anti-rabbit secondary antibody (Millipore). Chemiluminescent signals were detected with the Bio-Rad ECL detection kit on a ChemiDoc MP imaging system. For data analysis, band intensities at each temperature were quantified and normalized by setting the strongest and weakest signals to 100% and 0%, respectively. Melting curves were fitted using a Boltzmann sigmoidal model in GraphPad Prism 9. Aggregation temperatures were calculated from two independent biological replicates, and results are reported as single replicates.

Inhibitors afi-FSP1-2, 3, 4: 1.5×10^7 HEK293-EBNA1-6E cells/mL expressing human FSP1-twinSTREP-eGFP were treated with 1 μM compound 2, or 10 μM compound 3, or 10 μM compound 4 or 5% v/v DMSO following the above-described protocol. The heat treatment ranged from 31.6 to 78.1°C . PVDF membranes containing proteins were incubated with Precision Protein™ Strep-Tactin® HRP-conjugate (BioRad). The detection and data analysis were described before.

Immunoblot analysis. Jurkat T-lymphoid cell line was obtained from ATCC and cultured in RPMI 1640 with 10% FBS, 100 units/mL penicillin and streptomycin. The cells were lysed using sonication in M-PER buffer (cat:78501 from Thermo Scientific) and 50 μg of total protein per lane was loaded on the gel. FSP1 antibody is a Rabbit-derived cat. # HY-P80679 from MCE; GPX4 antibody is mouse-derived cat. # 67763-1-1g from Protein tech; GAPDH antibody (6C5) is mouse-derived, cat. # AM4300 from Invitrogen

Cell viability assays. Trypan blue dye exclusion was used to determine cell viability. Logarithmic phase cells were seeded at 1×10^5 cells/mL in a 12-well plate. Cells were then treated with either DMSO (control) or the test compounds solubilized in DMSO, gently mixed and incubated for 24 hours under culture conditions as described. Viability was assessed by removing an aliquot (10 μ L) of the cell suspension and mixing with an equal volume of 0.4% Trypan Blue solution. Live cells were then enumerated by light microscopy. Viability was expressed as a percentage of the DMSO control number.

Statistical Analysis and Reproducibility. Significance levels are indicated in legends of each figure as follows: * $P < 0.05$, ** $P < 0.01$, *** $P < 0.001$, **** $P < 0.0001$ (Student's t test). Tests used to derive significance are indicated in legends. Individual data points are indicated in graphs. In the latter case all replicates are independent and not multiple measurements taken from a single sample. Analysis of significance was done using unpaired Student's t -test, two-sided.

Crystallisation and structural determination of FSP1. Crystallisation was performed using 13 g/mL in 50 mM HEPES pH 7.2, 1 mM FAD, 1 mM NAD^+ with or without 1 mM coenzyme Q_1 . Small crystals or crystalline precipitate appeared in a few days under several conditions of the Mopheus screening kit (Molecular Dimensions) using the vapour diffusion sitting drop methods. The best-looking crystals grew in HEPES/MOPS 0.1 M pH 7.5, 12.5% PEG 1000, 12.5% PEG 3350, 12.5% MPD, 0.06 M MgCl_2 , 0.06 M CaCl_2 at 20 $^\circ\text{C}$ with a ratio of 1:1 in a sitting drop. After harvesting, they were smashed to generate small crystals, and used for microseeding. Specifically, the drops were composed of 0.15 μ L protein, 0.05 μ L seeds, and 0.2 μ L reservoir from the following commercial screens.

- FSP1-FAD structure; 0.06 M MgCl_2 , 0.06 M CaCl_2 , 0.1 M HEPES/MOPS (acid) pH 7.5; 20% v/v PEG500*MME, 10% PEG20000.
- FSP1-FAD- NAD^+ structure; 0.06 M MgCl_2 , 0.06 CaCl_2 , 0.1 M HEPES/MOPS (acid) pH 7.5, 12.5% PEG1000, 12.5% PEG3350, 12.5% MPD.
- FSP1-FAD- NAD^+ -coenzyme Q_1 structure; 0.06 M MgCl_2 , 0.06 CaCl_2 , 0.1 M imidazole/MES monohydrate (acid) pH 6.5, 20.0% PEG500* MME, 10% PEG20000. Crystals were then soaked with 20 mM coenzyme Q_1 added to the crystallisation drop.
- FSP1-FAD- NAD^+ -afi-FSP1-1; 0.06 M MgCl_2 , CaCl_2 , 0.1M imidazole/MES monohydrate (acid) pH 6.5, 12.5% MPD, 12.5% PEG1000, 12.5% PEG3350. Crystals were then soaked with 1 mM afi-FSP1-1 added to the crystallisation drop.
- FSP1-FAD- NAD^+ -afi-FSP1-3; 0.12 M diethylene glycol, 0.12 M triethylene glycol, 0.12 tetraethylene glycol, 0.12 pentaethylene glycol, 0.1 M tris (base)/bicine pH 8.5, 12.5% MPD, 12.5% PEG1000, 12.5% PEG3350. Crystals were then soaked with 10 mM afi-FSP1-3 added to the crystallisation drop.
- FSP1-FAD- NAD^+ -afi-FSP1-4; 0.06 M MgCl_2 , 0.06 M CaCl_2 , 0.1 M imidazole/MES monohydrate (acid) pH 6.5, 12.5% MPD, 12.5% PEG1000, 12.5% PEG3350. Crystals were then soaked with 10 mM afi-FSP1-4 added to the crystallisation drop.
- FSP1(L323A)-FAD- NAD^+ -afi-FSP1-2; 0.06 M MgCl_2 , 0.06 M CaCl_2 , 0.1 M HEPES/MOPS (acid) pH 7.5, 20% ethylene glycol, 20% PEG8000. Crystals were then soaked with 1 mM afi-FSP1-2 added to the crystallisation drop.

Electron densities from co-crystallisations with previously reported inhibitors (Supp. Table 12) did not show any ligand binding. Data were collected at the European Synchrotron Radiation Facility (Grenoble, France) and processed with the XDS and CCP4 packages (137, 138). The phase problem was solved by molecular replacement using an Alpha Fold model and CCP4 programs. Model building and refinement were conducted using COOT and Refmac5 (139). Figures were then generated using ChimeraX (140), PyMOL (DeLano Scientific; www.pymol.org).

SPR Binding Assays. SPR binding assays were carried out using the Biacore 8K SPR system (Cytiva) with the running buffer (50 mM HEPES, pH 7.2, 150 mM NaCl, 0.05% Tween-20, and 2% DMSO) under controlled conditions at 25 $^\circ\text{C}$. The His-Sumo-FSP1 protein was immobilized on a preconditioned and activated Series S NTA sensor chip (Cytiva). Pre-conditioning involved treatment with 350 mM EDTA, followed by activation with 0.5 mM NiCl_2 . To enhance stability and minimize signal drift, the chip was further activated using EDC (1-ethyl-3-(3-dimethylaminopropyl) carbodiimide) and NHS (N-hydroxysuccinimide) before coupling the protein to the surface. The protein was immobilized to a response level of 7500 RU.

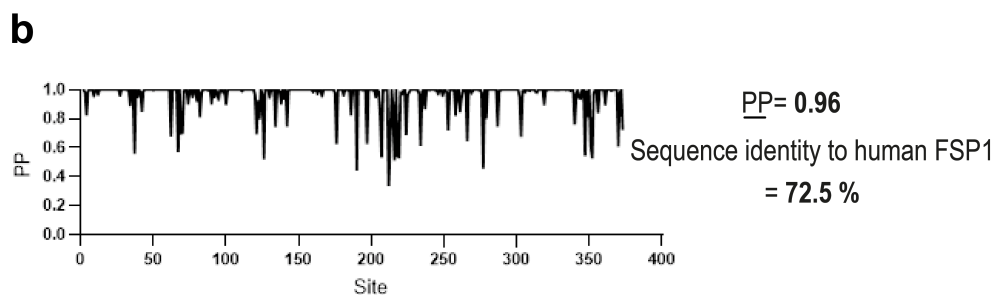
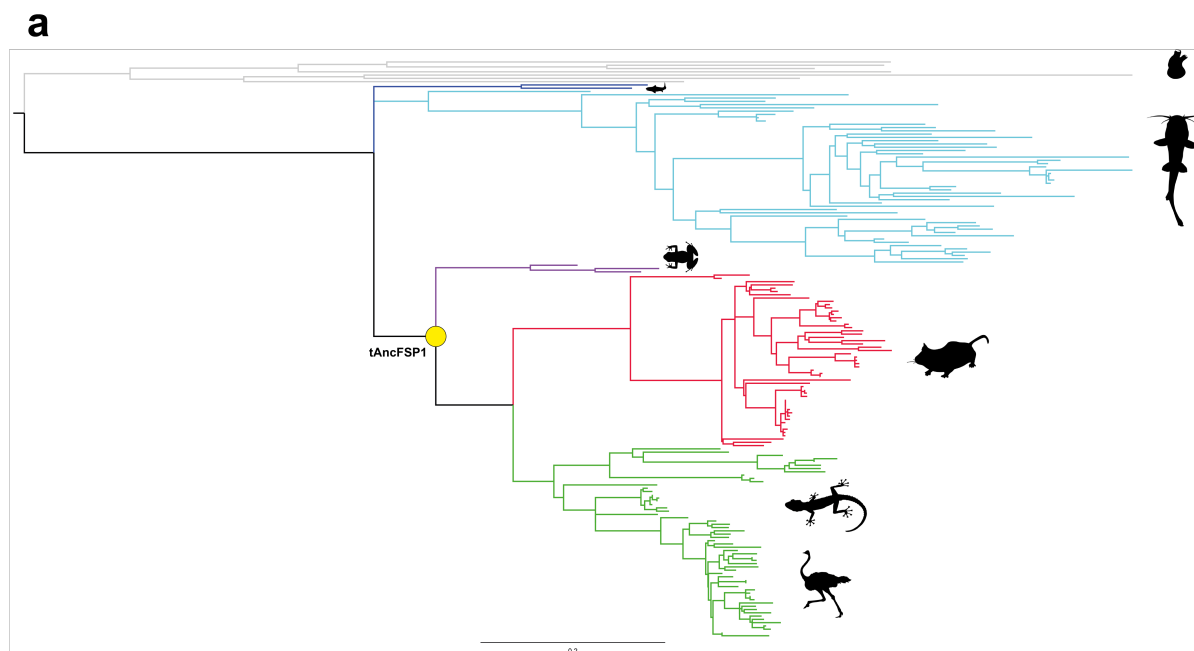
Compounds were applied in a three-fold serial dilution series (6, 18.9, 60, 189, 600, 1897, 6000, 18973, and 60000 nM) using a single-cycle kinetics (SCK) setup. Compound plating and DMSO normalization were performed using the Tecan D-300e digital dispenser (Tecan Life Sciences). Binding was monitored for 60 s during the association phase and 120 s during the dissociation phase. Reference channel signals and buffer blanks were used to correct the raw binding signals (RU).

Binding data were collected from three independent experiments; however, only a single representative dataset is shown for each compound in the manuscript. Data processing and binding analysis were performed using Biacore Insight software

1319 (Cytiva). Steady-state binding levels (R_{eq}) were plotted against compound concentrations and fitted to a one-site binding model
1320 using the Levenberg–Marquardt algorithm, enabling accurate determination of binding affinities for each compound.

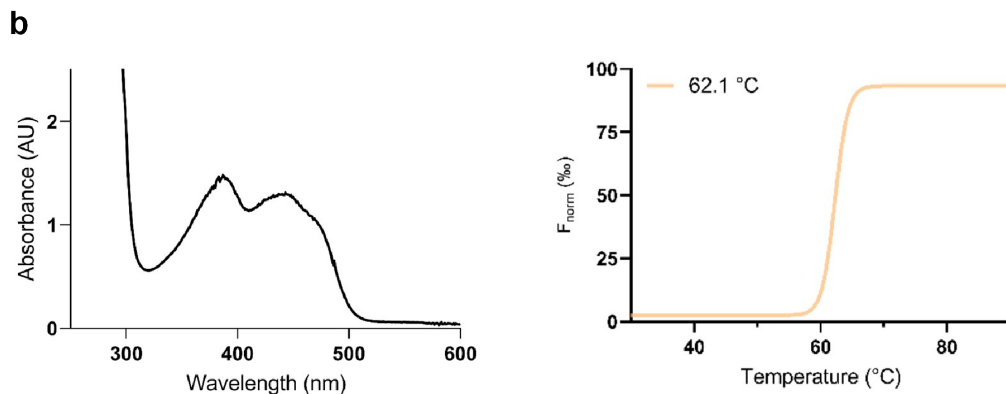
1321 **Virtual Screens.** The AdaptiveFlow open-source drug-discovery platform was used to carry out the virtual screen. The size
1322 of the docking box was set to 14 Å × 18 Å × 16 Å. The FSP1-FAD-NAD⁺ complex structure was held rigid during the
1323 virtual screens (ATG prescreen and primary screen). QuickVina 2 (111) using a hybrid empirical and knowledge-based scoring
1324 function was used as the docking program. The exhaustiveness value was set to 1. The number of replicas was set to 1. The
1325 virtual screen was run in Amazon Web Services (AWS). Postprocessing of the results including clustering with the similarity
1326 limit set to 0.8 was done with DataWarrior (v6.0.0) (141). In detail: For each target, the virtual library was docked and the
1327 top ~2,000 compounds by docking score were retained for further processing. Compounds flagged by DataWarrior “Nasty
1328 Functions” and PAINS alerts were removed to exclude known assay-interfering or chemically problematic substructures (e.g.,
1329 strong Michael acceptors, redox-active quinones, rhodanines, catechols/hydroquinones, azo dyes). The remaining compounds
1330 were filtered by simple physicochemical criteria, requiring predicted aqueous solubility $\geq 10 \mu\text{M}$, overall aromaticity $\leq 75\%$,
1331 and fraction of sp^3 carbons ≥ 0.1 . This filtered set was then clustered by 2D similarity, and from each cluster one or more
1332 representatives were chosen, prioritizing the best docking scores, to yield a final diverse set of ~80 compounds per target
1333 for experimental testing. Apart from these automated substructure and physicochemical filters and the docking scores, no
1334 additional interaction-based or other structural criteria were applied.

1335 **Discovery of Inhibitor Analogs.** Analog Hunter and Scaffold Hopper modules in the infiniSee (142) module from BioSolveIT
1336 were used to screen analogs of compound afi-FSP1-1. Top-ranking 5000 compounds from each module were exported to
1337 SeeSAR (?) from BioSolveIT and docked with the FlexX algorithm in the general docking mode to the coenzyme Q binding
1338 site of the FSP1/FAD/NAD/afi-FSP1-1 complex structure (PDB code: 9IFU) with compound afi-FSP1-1 removed. The docked
1339 poses were scored with the HYDE scoring function. Top-ranking 168 compounds with a similar binding pose as compound
1340 afi-FSP1-1 were tested with 100-ns-long molecular dynamics (MD) simulations with the PMEMD module (CUDA version) of
1341 the AMBER 22 package and protein.ff19SB force field for protein (143). The Molecular Mechanics/Generalized Born Surface
1342 Area (MM/GBSA) binding free energies between protein and the ligands that remained in the binding pocket were calculated
1343 by the MMPBSA.py script in AmberTools 23 (144). The top-ranking 20 analogs with the lowest MM/GBSA binding free
1344 energies were ordered from Enamine for testing.

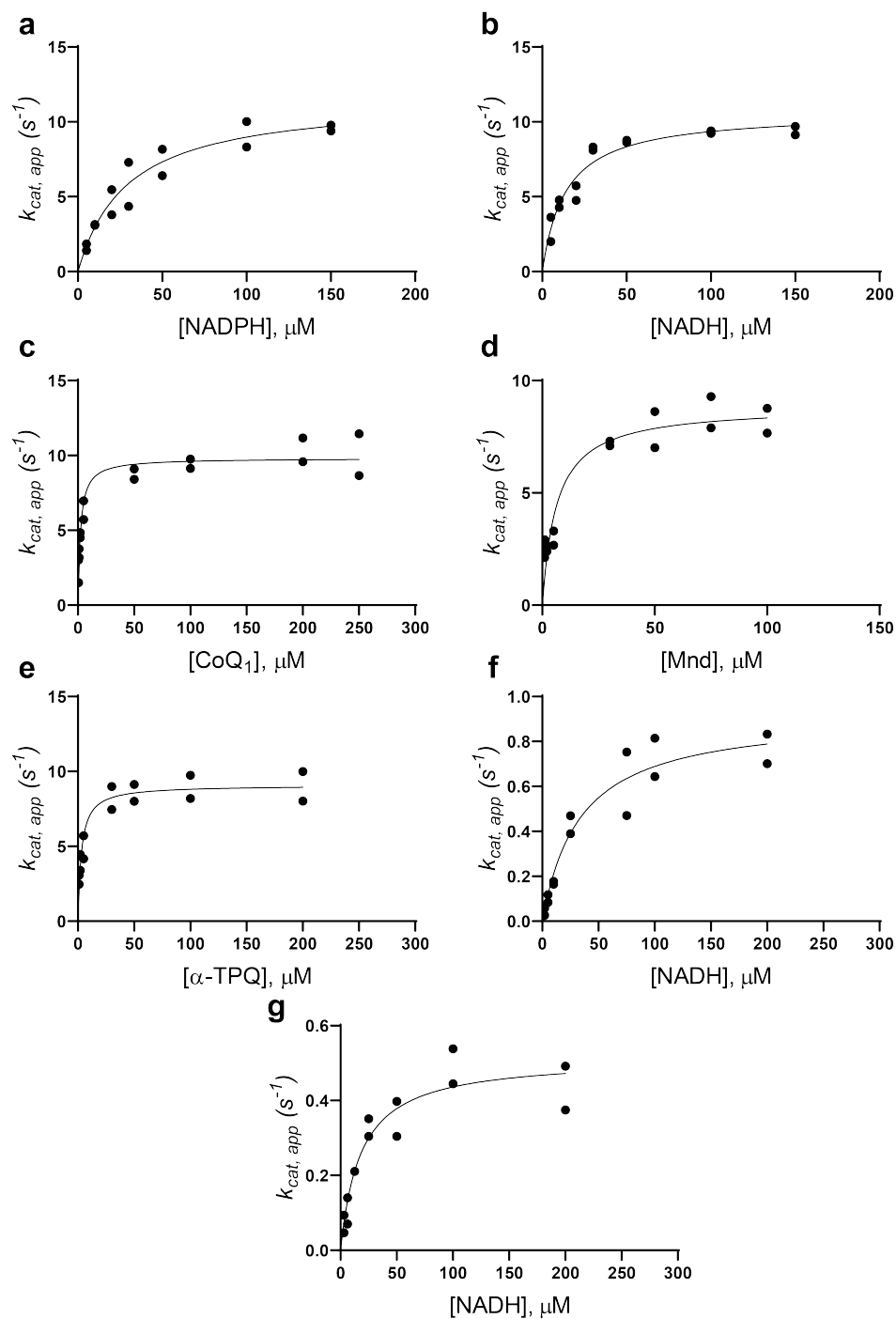


>MGSKLSVDDSVHVVIIVGGGFGGIIAAASQLKSWGVPFVLVDMKDAFHNNVAALRASVESGFAKKTFFPYAATFGDSFKQG
KVVGIDLEKQHVLLLEDGEELHFSLILATGSDGPFPGKFNQPVSMETAIQMYEDMVKEVQKAQRIVVVGGSAGVEMAAE
IKTDYPDKEVTLIHISKIALADVELLPVSRQGVKEILLQKGVQLLLGQRVSNLQELTLNQVQENMKVKTDKGTEITADLVI
CCTGIKVNSSAYSSAFGDKLAENGALKVNEHLQVEGYDNVYAI GDCADVKEPKMAYHAGLHANVAVTNI INSLTNKPLKS
YKPGSLTMLLSMGRNDGVGQLNGYYVGRFVVRIAKSRDLFVGKSWKEMGQTMPI

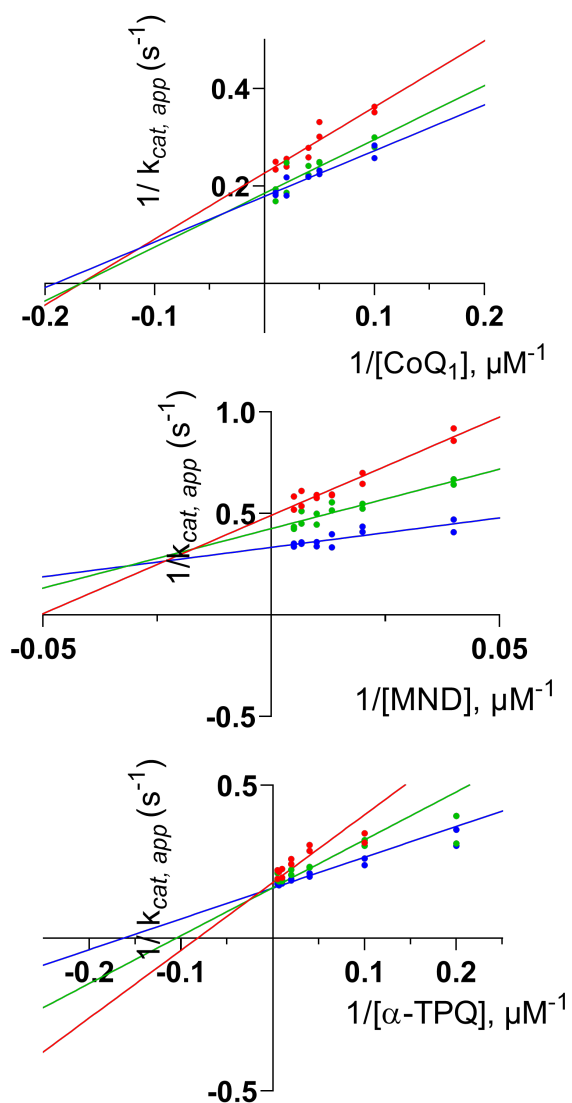
Supplementary Fig. 15. Ancestral Sequence Reconstruction of FSP1. (a) Condensed maximum likelihood phylogenetic tree of Chordata FSP1 proteins. Taxonomic groups are color-coded: Echinodermata, Hemichordata, and Ascidiacea (grey); Chondrichthyes (blue); Actinopteri (cyan); Amphibia (violet); Mammalia (red); and Sauria (dark green). Silhouettes were obtained from <https://www.phylopic.org/>. The yellow circle highlights the reconstructed and experimentally characterized ancestral FSP1 corresponding to the tetrapodal ancestor. (b) Amino acid sequence of the reconstructed FSP1 used in this study, along with the posterior probability distribution per residue. Residues highlighted in red were excluded from the biochemical construct, as they correspond to the N-terminal region involved in myristoylation.



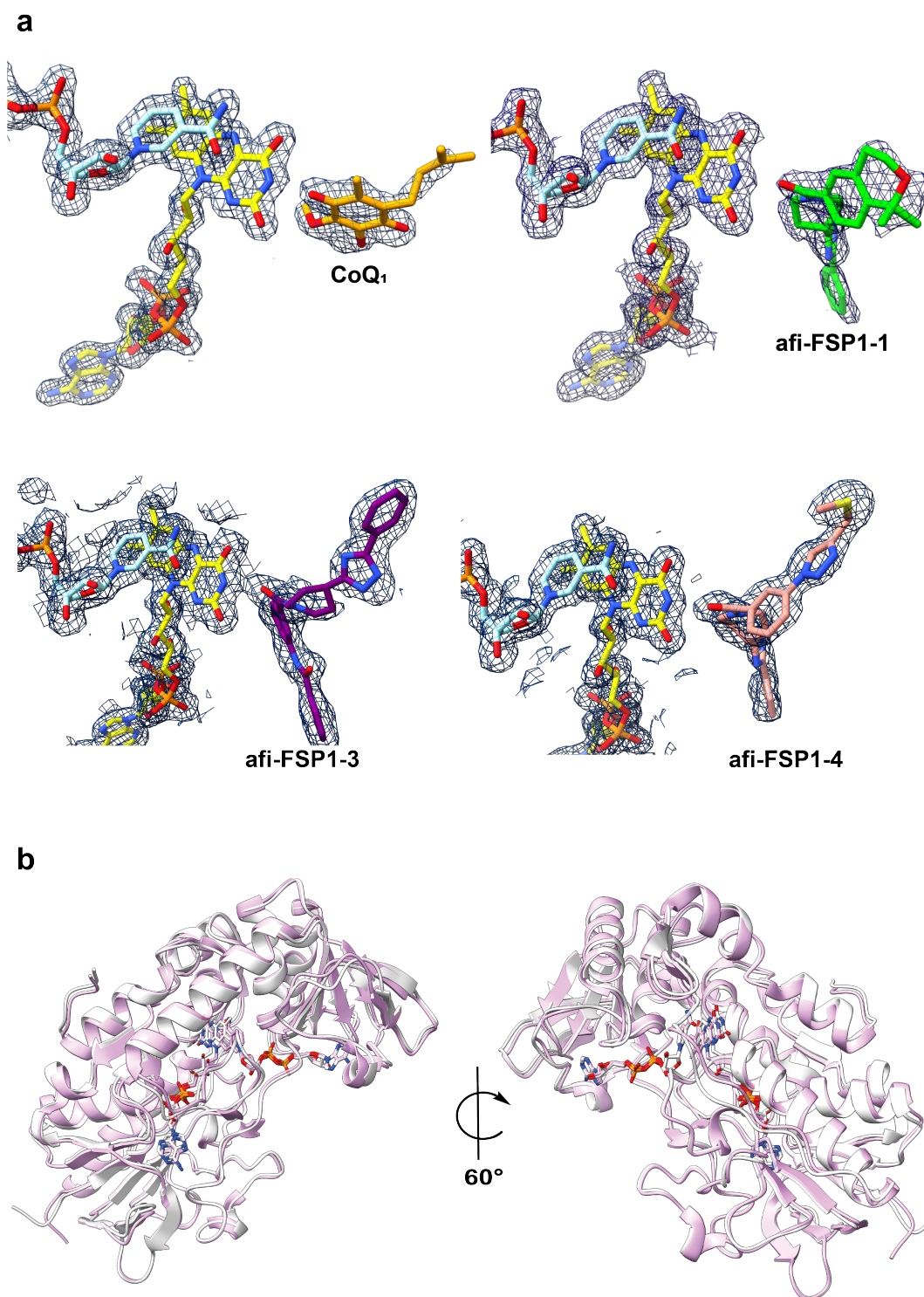
Supplementary Fig. 16. (a) Sequence alignment of human FSP1 with representative orthologs from amphibian (*Microcaecilia unicolor*), bird (*Gallus gallus*), and the reconstructed ancestral FSP1. (b) UV-visible absorbance spectrum and ThermoFAD unfolding profile of the ancestral FSP1.



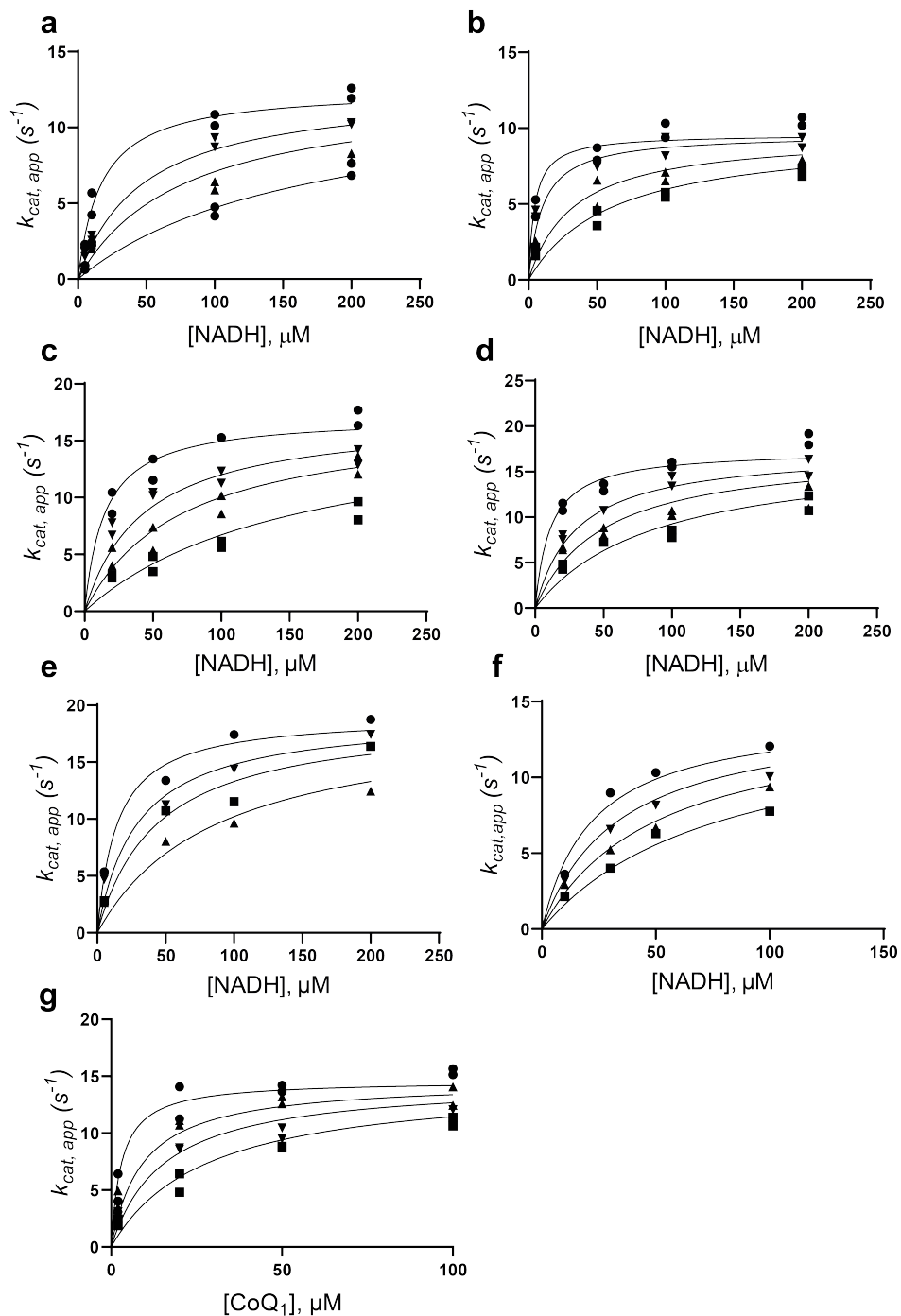
Supplementary Fig. 17. Steady-state Kinetic Analysis of FSP1 Activity. (a) NADPH titration at fixed coenzyme Q_1 (100 μM). (b) NADH titration at fixed coenzyme Q_1 (100 μM). (c) Coenzyme Q_1 titration at fixed NADH (100 μM). (d) Menadione titration at fixed NADH (100 μM). (e) α -tocopherol (quinonic hydrophilic head, α -TPQ) titration at fixed NADH (100 μM). (f) NADH titration monitored by superoxide formation using the nitroblue tetrazolium (NBT) assay. Reactions in (a–e) were monitored via fluorescence-based NAD(P)H consumption, while (f–g) used absorbance-based and superoxide detection methods, respectively. Protein concentrations were 0.2 μM for panels (a–e) and 1 μM for panels (f–g). Each curve represents a fit to the Michaelis–Menten model; individual replicates ($n = 2$) are plotted as dots. MND = menadione; α -TPQ = α -tocopherol quinonic hydrophilic head.



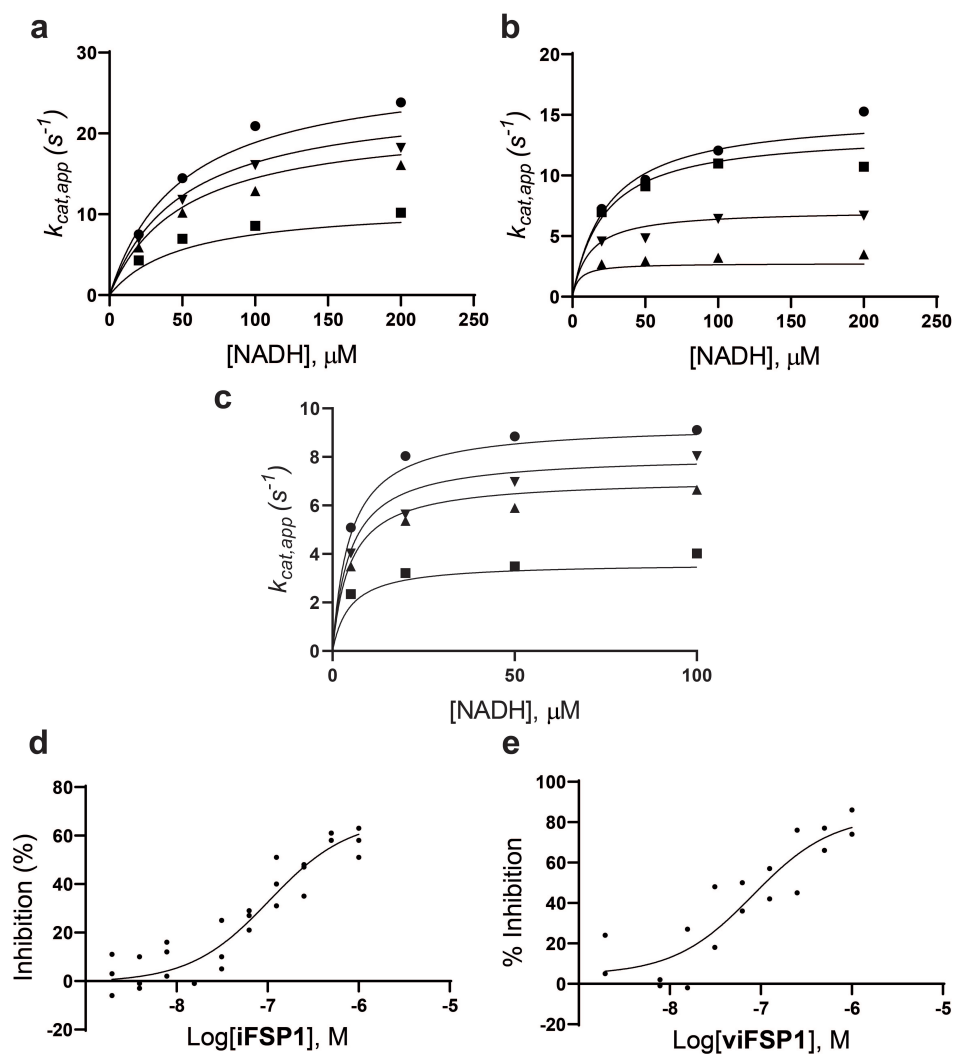
Supplementary Fig. 18. Lineweaver–Burk Plots for Different Substrate Pairs. (Top) NADH/coenzyme Q₁, (Middle) NADH/menadione (MND), and (Bottom) NADH/ α -tocopherol (α -TPQ; quinonic hydrophilic head group). Experiments were performed with varying concentrations of the electron acceptor substrate at fixed NADH concentrations (25, 50, and 100 μM), shown in red, green, and blue lines, respectively. The convergence of lines to the left of the y-axis indicates a ternary complex mechanism of substrate binding. Individual replicates ($n = 2$) are plotted as dots.



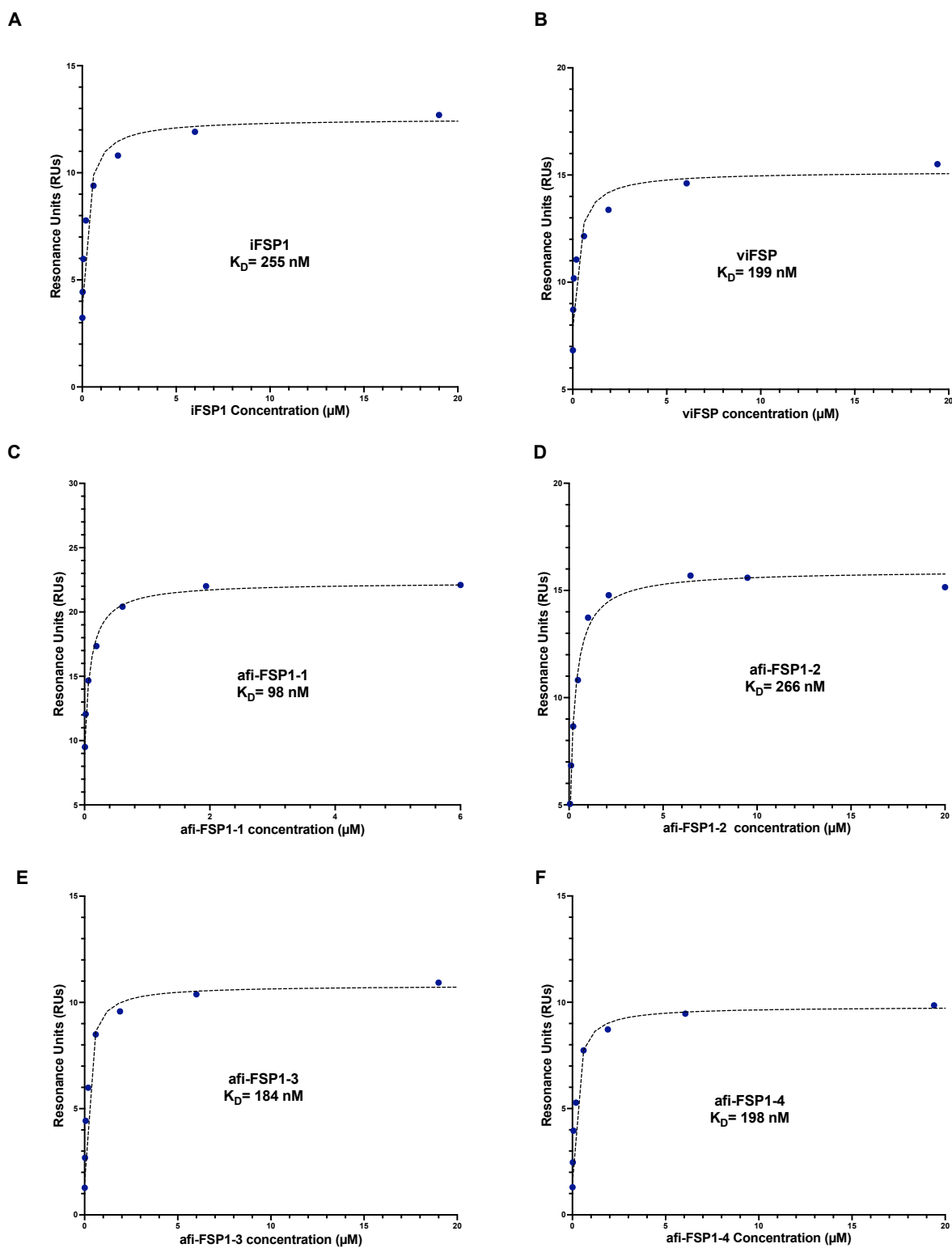
Supplementary Fig. 19. Quality of Ligand Electron Densities and Structural Similarity of FSP1. (a) Crystal structures of FSP1 in complex with NAD⁺ and coenzyme Q₁ (orange), NAD⁺ and compound afi-FSP1-1, NAD⁺ and compound afi-FSP1-3, and NAD⁺ and compound afi-FSP1-4 (inhibitors shown in green, purple and pink). FAD carbon atoms are colored yellow, and NAD⁺ carbons in light cyan. Final 2F_o–F_c electron density maps are contoured at 1.3–1.4 σ . (b) Structural superposition of ancestral FSP1 (white) and human FSP1 (pink; PDB 1WIK). The root-mean-square deviation (RMSD) is 1.1 Å across 373 C α atoms.



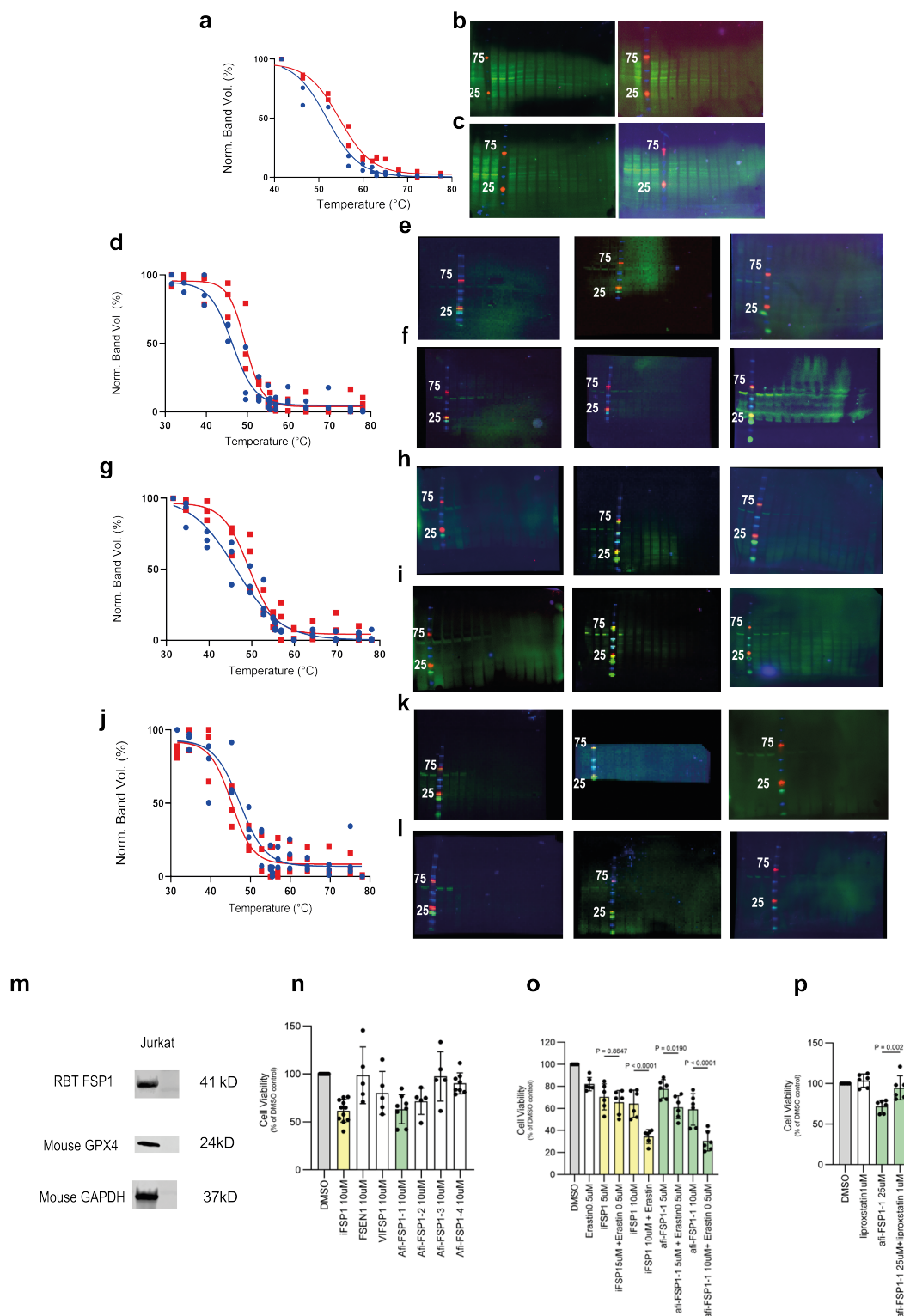
Supplementary Fig. 20. K_i measurements of FSP1 inhibitors Under Competitive Inhibition Conditions. (a) K_i determination for compound afi-FSP1-1 by varying NADH concentrations in the presence of coenzyme Q_1 (100 μ M). Data points (circles, downward arrows, upward arrows, squares) correspond to 0, 1.0, 2.0, and 5.0 μ M inhibitor concentrations. Individual replicates ($n = 2$) are plotted as dots. (b) K_i for compound afi-FSP1-2, tested at 0, 1.0, 5.0, and 10.0 μ M ($n = 2$). (c) K_i for compound afi-FSP1-3, tested at 0, 1.0, 2.0, and 5.0 μ M. (d) K_i for compound afi-FSP1-4, tested at 0, 0.5, 1.0, and 2.0 μ M ($n = 2$). (e) K_i for compound afi-FSP1-5, tested at 0, 1.0, 2.0, and 5.0 μ M ($n = 1$). (f) K_i for compound afi-FSP1-6, tested at 0, 2.0, 5.0, and 10.0 μ M ($n = 1$). (g) K_i for compound afi-FSP1-1 by varying coenzyme Q_1 at fixed NADH (100 μ M), with inhibitor concentrations of 0, 0.5, 1.0, and 2.0 μ M ($n = 2$). All panels show individual replicates plotted as dots, with non-linear regression fits.



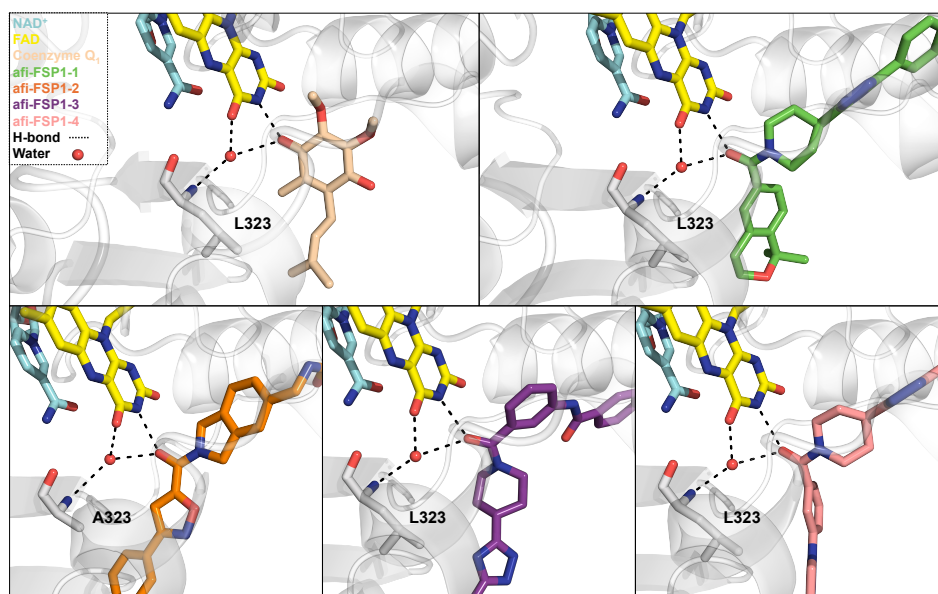
Supplementary Fig. 21. K_i and IC_{50} Measurements for Previously Reported FSP1 Inhibitors.(a) K_i determination for iFSP1 by varying NADH concentrations in the presence of coenzyme Q₁ (100 μM). Data points (circles, downward arrows, upward arrows, and squares) correspond to 0, 0.5, 1.0, and 5.0 μM inhibitor concentrations. (b) K_i determination for viFSP1 under the same conditions, using 0, 0.05, 0.5, and 2.0 μM inhibitor concentrations. (c) K_i determination for FSEN1, tested at 0, 0.05, 0.1, and 0.5 μM inhibitor concentrations. All K_i measurements were performed with $n = 1$. (d–e) IC_{50} values for iFSP1 and viFSP1 determined using the resazurin assay (100 μM resazurin, 200 μM NADH) with $n = 3$ and $n = 2$ replicates, respectively.



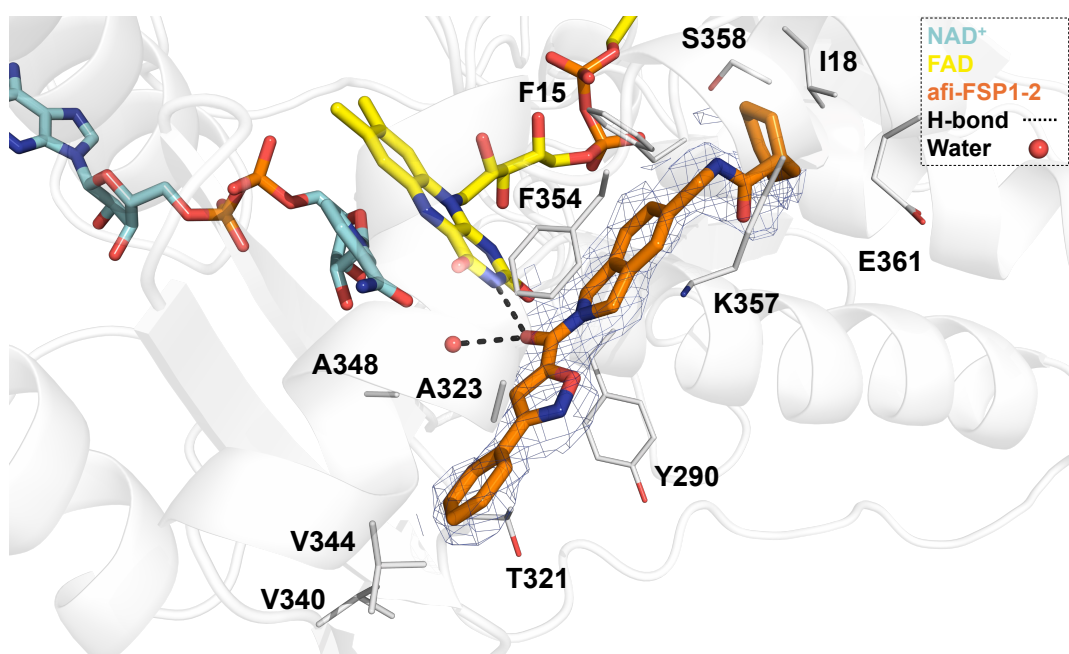
Supplementary Fig. 22. Steady-State Binding Curves and Corresponding K_D Values. Steady-state binding curves and corresponding K_D values based on surface plasmon resonance (SPR) for control compounds (A, B) and the top four inhibitors (C-F). Each curve represents binding responses measured across a range of compound concentrations, with data collected from three independent experiments. A single representative dataset is shown for each compound, illustrating the consistency and reproducibility of the observed binding interactions.



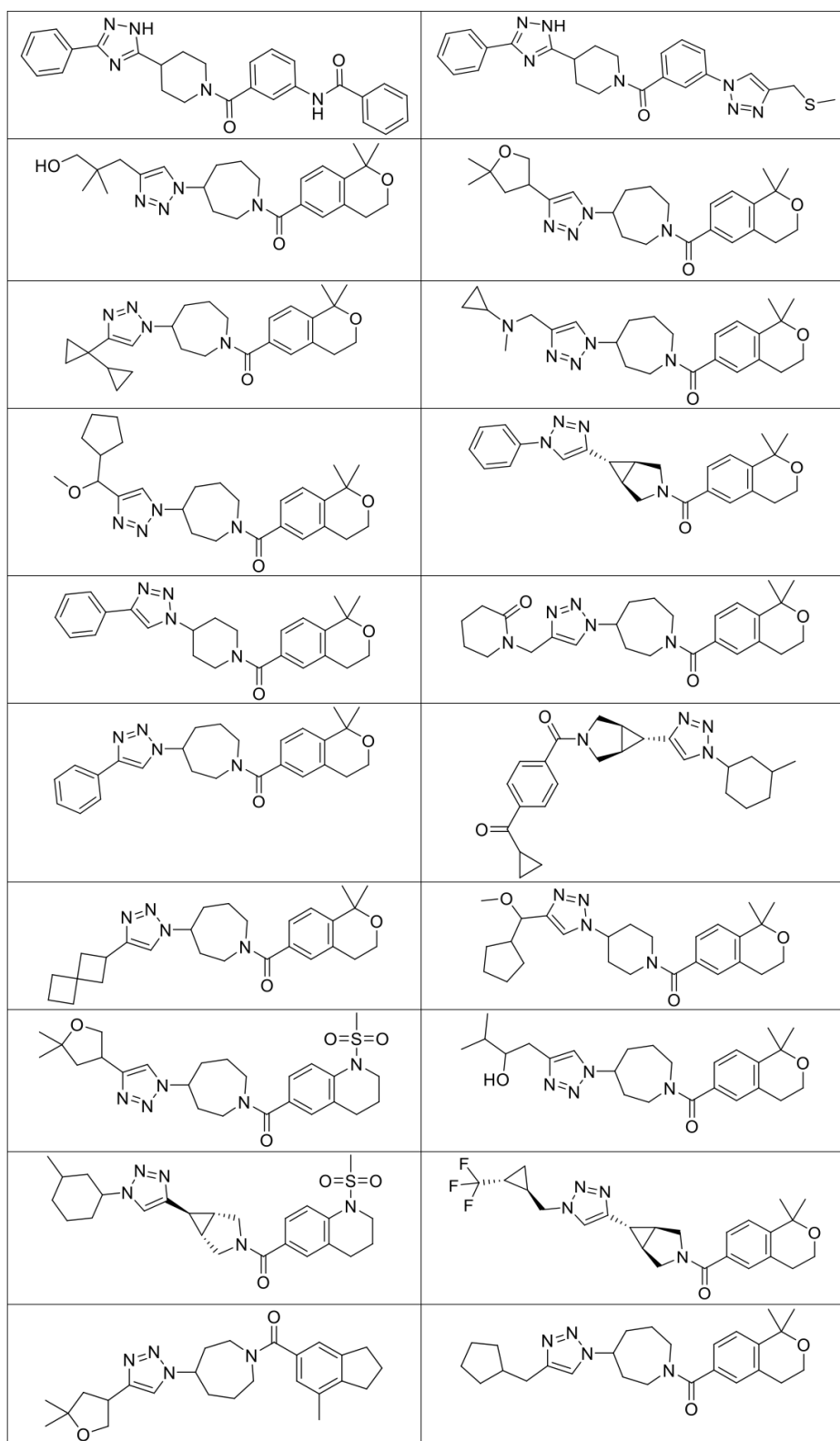
Supplementary Fig. 23. Cellular Thermal Shift Assay (CETSA) and activity. (a) 10 μM ari-FSP1-1 (red) engages endogenous human FSP1 (40.53 kDa) in HEK293 cells. The thermal shift is 2.8°C. Data are given as $n = 2$ independent experiments. DMSO (blue) was used as a control (b-c) Uncropped western blot gels representing protein quantification after exposure to increasing temperatures without (b) and with ari-FSP1-1 (c). (d) 1 μM ari-FSP1-2 (red) engages recombinant humanFSP1-twinSTREP-eGFP (71.5 kDa). The thermal shift is 3.1°C. Data are given as $n = 3$ independent experiments. (e-f) Uncropped western blot gels representing protein quantification after exposure to increasing temperatures without (e) and with ari-FSP1-2 (f). (g) 10 μM ari-FSP1-3 (red) engages human FSP1-twinSTREP-eGFP (71.5 kDa). The thermal shift is 3.2°C. Data are given as $n = 3$ independent experiments. (h-i) Uncropped western blot gels representing protein quantification after exposure to increasing temperatures without (h) and with ari-FSP1-3 (i). (j) No positive thermal shift was detected for 10 μM ari-FSP1-4 (red) possibly because of poor cell permeability or cell penetration. Data are given as $n = 3$ independent experiments. (k-l) Uncropped western blot gels representing protein quantification after exposure to increasing temperatures without (k) and with ari-FSP1-4 (l). For all the Western Blots, the molecular weight markers are in red and blue. (m) Jurkat cells express FSP1 and GPX4 (glutathione peroxidase 4) that is implicated in ferroptosis as it protects lipids from oxidation. (n) ari-FSP1 inhibitors (10 μM) are only minimally toxic as single agents. iFSP1, FSEN1, and viFSP1 are known FSP1 inhibitors. (o) Cotreatment of Jurkat cells with the combination of ari-FSP1-1 and erastin, a ferroptosis inducer, decreases cell viability to the same extent as the iFSP1-erastin cotreatment. (p) Liproxstatin, a ferroptosis inhibitor, fully rescues the effect of high 25 μM ari-FSP1-1.



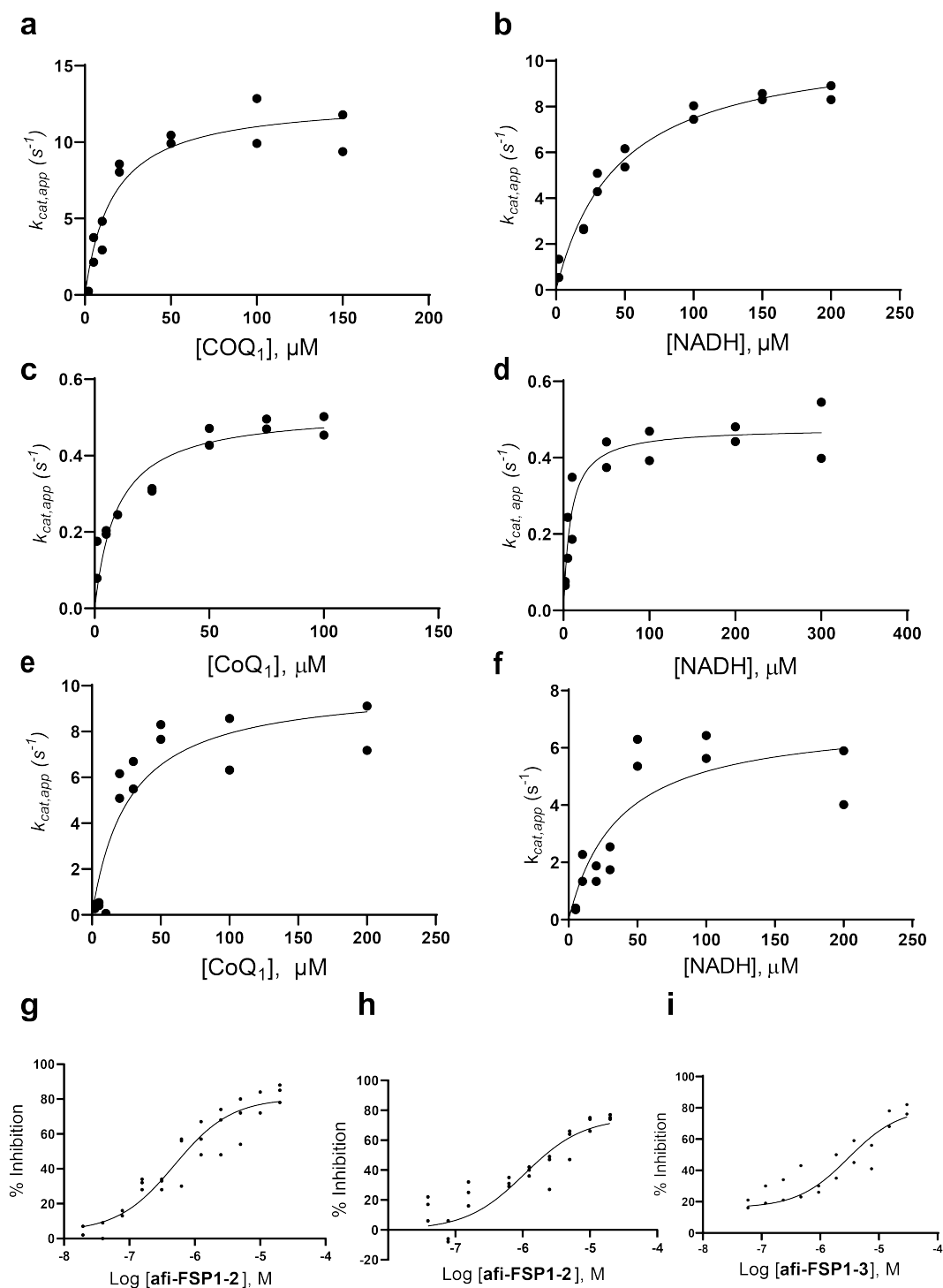
Supplementary Fig. 24. Water-mediated hydrogen bonds between the flavin, the active-site ligands, and the protein.



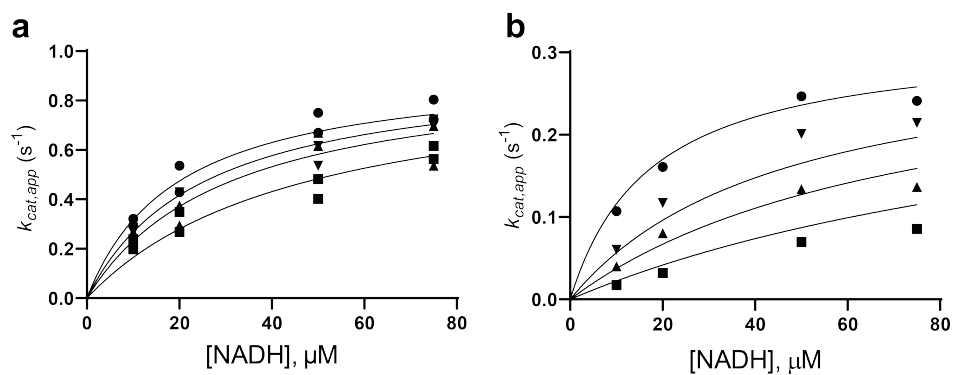
Supplementary Fig. 25. Co-Crystal Structure of Inhibitor afi-FSP1-2 binding to FSP1 mutant L323A. The polder omit map for the inhibitor is contoured at the 3.0 σ level. Close-up views of the binding site with afi-FSP1-2 (orange) bound in the region normally occupied by coenzyme Q₁, adjacent to FAD (yellow) and NAD⁺ (cyan). The inhibitor displaces coenzyme Q₁ and interacts directly with the catalytic cavity.



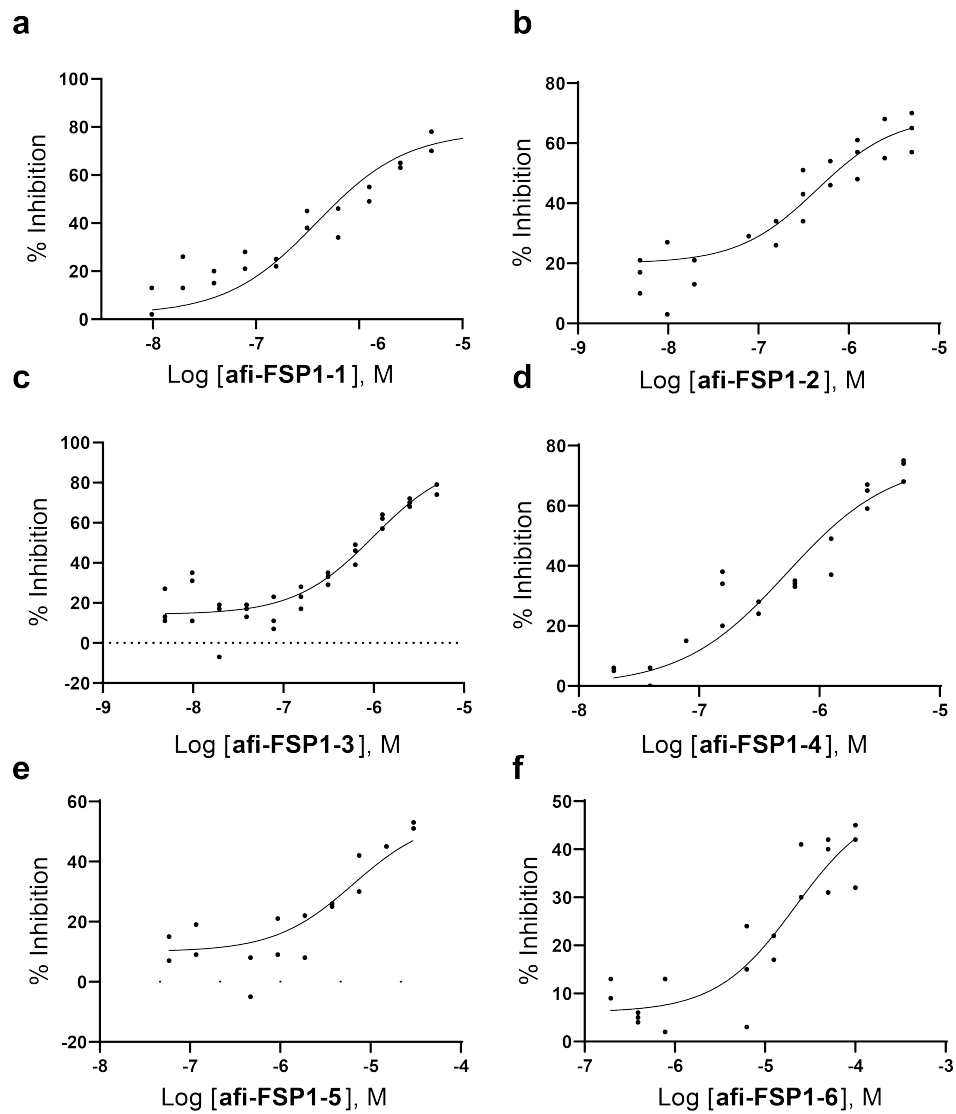
Supplementary Fig. 26. Additional FSP1 Inhibitors. Additional FSP1 inhibitors identified from the analog screening of afi-FSP1-1 and validated experimentally as a mixture if the analog has stereoisomers. Compounds showing >50% inhibition at 10 μ M are highlighted in the top two rows and are listed in Supplementary Table 11.



Supplementary Fig. 27. Steady-State Kinetics and IC_{50} Measurements for F15A and L323A FSP1 Mutants. (a–b) Michaelis–Menten kinetics of L323A measured by varying coenzyme Q_1 at fixed NADH (100 μM) (a), or varying NADH at fixed coenzyme Q_1 (100 μM) (b). Protein concentration: 0.1 μM . (c–d) Michaelis–Menten kinetics of F15A under the same conditions. Protein concentration: 1 μM . Each data point represents an independent replicate ($n = 2$); dots show individual measurements, and curves represent non-linear regression fits. (e–f) Michaelis–Menten kinetics of F354A under the same conditions. Protein concentration: 0.2 μM . Each data point represents an independent replicate ($n = 2$); dots show individual measurements, and curves represent non-linear regression fits. None of the compounds developed in this study inhibits this mutant. (g) IC_{50} determination for compound afi-FSP1-2 against L323A (0.1 μM) using the resazurin assay (50 μM resazurin, 200 μM NADH, $n = 3$). (h–i) IC_{50} determination for compounds afi-FSP1-2 ($n = 3$) and afi-FSP1-3 ($n = 2$) against F15A (0.5 μM).



Supplementary Fig. 28. K_i Determination from NADH Oxidase Activity Assays under Competitive Inhibition Conditions. (a) K_i for compound afi-FSP1-1, measured by varying NADH concentrations in the absence of coenzyme Q₁. Data points (circles, downward arrows, upward arrows, and squares) represent inhibitor concentrations of 0, 1.0, 2.0, and 5.0 μM, respectively ($n = 2$). (b) K_i for compound afi-FSP1-2, measured under the same conditions. Data points represent inhibitor concentrations of 0, 5.0, 10.0, and 20.0 μM, respectively ($n = 1$).



Supplementary Fig. 29. IC_{50} Values (μM) of the Most Potent FSP1 Inhibitors. Panels (a–f) show dose–response curves obtained using the resazurin assay (100 μM resazurin, 200 μM NADH). Each curve represents mean values from independent experiments (a,e: $n = 2$; b,c,d,f: $n = 3$).

Condition	k_{cat} (s ⁻¹)	K_M (μM)	$\frac{k_{\text{cat}}}{K_M}$ (s ⁻¹ μM ⁻¹)
Oxidoreductase activity^a			
Varying NADPH, fixed coenzyme Q ₁	11.60 ± 0.88	29.27 ± 6.26	0.40
Varying NADH, fixed coenzyme Q ₁	10.65 ± 0.50	13.89 ± 2.39	0.76
Varying coenzyme Q ₁ , fixed NADH	9.81 ± 0.33	2.15 ± 0.36	4.56
Varying menadione, fixed NADH	8.86 ± 0.45	6.33 ± 0.74	1.40
Varying α-tocopherol ^b , fixed NADH	9.07 ± 0.35	3.00 ± 0.60	3.02
NADH oxidase activity at atmospheric O₂ concentration			
Varying NADH ^c	0.92 ± 0.08	33.96 ± 9.53	0.03
Varying NADH ^d	0.51 ± 0.04	18.04 ± 5.60	0.03

Supplementary Table 10. Steady-state Kinetic Parameters for FSP1 with Different Electron Donors and Acceptors. k_{cat} and K_M values were determined from Michaelis–Menten fits. ^aFixed concentrations were 100 μM. Data shown in Supplementary Fig. 17. ^bQuinone hydrophilic head group. ^cMeasured via NADH consumption assay. ^dMeasured using nitroblue tetrazolium assay (superoxide detection).

Parameter	FAD	FAD + NAD ⁺	FAD + NAD ⁺ + CoQ ₁	FAD + NAD ⁺ + afi-FSP1-1	FAD + NAD ⁺ + afi-FSP1-2 (L323A)	FAD + NAD ⁺ + afi-FSP1-3	FAD + NAD ⁺ + afi-FSP1-4
Space group	P2 ₁	P2 ₁	P2 ₁	P2 ₁	P2 ₁	P2 ₁	P2 ₁
Unit cell a/b/c (Å)	43.09	43.95	43.51	43.57	44.45	43.85	43.66
	81.54	80.54	79.15	79.68	78.06	79.33	79.51
	57.69	60.92	112.07	113.06	115.08	113.37	112.78
Resolution (Å)	2.10	1.45	1.70	1.55	2.40	1.94	1.74
PDB code	9IFZ	9IFT	9IFY	9IFU	9QRT	9IG9	9IFW
$R_{\text{sym}}^{\text{a,b}}$ (%)	19.4 (98.3)	8.9 (171.0)	4.4 (53.6)	10.6 (39.3)	12.0 (71.0)	11.9 (84.6)	8.0 (74.8)
$CC_{1/2}$ (%)	99.2 (70.2)	99.9 (45.5)	99.8 (70.9)	97.9 (76.5)	97.3 (36.1)	95.5 (32.7)	99.4 (27.7)
Completeness ^b (%)	99.9 (99.8)	99.0 (99.5)	95.8 (96.8)	83.7 (50.6) ^e	96.1 (98.1)	93.5 (81.6)	97.1 (96.5)
Unique reflections	23,156	72,942	79,096	46,979	29,339	53,297	75,055
Redundancy	6.9 (7.1)	6.9 (6.7)	2.7 (2.6)	2.5 (2.6)	2.7 (2.6)	2.0 (2.0)	2.4 (2.4)
I/σ^{b}	7.2 (2.1)	10 (1.0)	8.9 (1.1)	5.1 (2.2)	7.4 (1.6)	4.7 (0.8)	5.0 (0.6)
Non-H atoms (protein / FAD)	2770 / 53	2804 / 53	5594 / 2×53	5612 / 2×53	5534 / 2×53	5548 / 2×53	5573 / 2×53
NAD ⁺	–	44	2×44	2×44	2×44	2×44	2×44
Coenzyme Q ₁	–	–	2×17	–	–	–	–
Inhibitor	–	–	–	2×30	33	2×34	2×32
Water molecules	118	294	364	556	161	182	273
B-factor (protein / ligands) (Å ²)	27.6 / 28.0	22.1 / 18.3	30.94 / 31.42	21.38 / 24.51	29.97 / 29.65	29.15 / 30.17	25.9 / 29.16
$R_{\text{cryst}}^{\text{b,c}}$ (%)	19.76 (24.0)	17.5 (32.0)	18.81 (34.0)	17.9 (18.7)	21.6 (22.9)	16.3 (32.8)	20.5 (36.0)
$R_{\text{free}}^{\text{b,c}}$ (%)	22.74 (31.2)	20.3 (33.0)	22.58 (37.9)	24.25 (22.0)	28.0 (24.0)	15.9 (26.6)	20.5 (21.57)
RMS bond length (Å)	0.064	0.010	0.008	0.006	0.0070	0.0069	0.0076
RMS bond angles (°)	1.37	1.63	1.47	1.34	1.47	1.46	1.53

Supplementary Table 11. Data Collection and Refinement Statistics for FSP1 Crystal Structures. $R_{\text{sym}} = \sum (|I_i - \langle I \rangle|) / \sum I_i$, where I_i is the intensity of the i^{th} observation and $\langle I \rangle$ the mean intensity of the reflection. Values in parentheses correspond to the highest resolution shell. $R_{\text{cryst}} = \sum (|F_{\text{obs}} - F_{\text{calc}}|) / \sum |F_{\text{obs}}|$. R_{free} was calculated using a separate test set. Anisotropic datasets were processed with *STARANISO* (<http://staraniso.globalphasing.org/cgi-bin/staraniso.cgi>).

Compound	Structure	K_i (μM)	K_d (μM)
iFSP1		3.29 ± 0.48^b	0.255
viFSP1		0.446 ± 0.07^c	0.199
FSEN1		0.315 ± 0.03^d	0.319
afi-FSP1-1		0.283 ± 0.05	0.098
afi-FSP1-2		0.777 ± 0.22	0.266
afi-FSP1-3		0.520 ± 0.11	0.184
afi-FSP1-4		0.284 ± 0.07	0.198

Supplementary Table 12. K_i Values (μM) and K_d Values (μM) of FSP1 Inhibitors. K_i values were measured by monitoring NADH oxidation via absorbance at 340 nm (pH 7.2), using fixed concentrations of 100 μM coenzyme Q_1 and 0.1–0.2 μM FSP1. Compounds were tested from micromolar to low nanomolar concentrations. Data are shown in Supplementary Fig. 21a–c. The reported IC_{50} for iFSP1 against human FSP1 in the resazurin assay is 0.103 μM (10). We measured 0.104 μM for the ancestral FSP1 using the same assay (Supplementary Fig. 21d). For viFSP1, the reported IC_{50} is 0.083 μM (45); we measured 0.081 μM for the ancestral protein (Supplementary Fig. 21e). For FSEN1, the reported IC_{50} against human FSP1 is 0.313 μM (44).

Mutant	Condition	k_{cat} (s^{-1})	K_M (μM)	$\frac{k_{\text{cat}}}{K_M}$ ($\text{s}^{-1} \mu\text{M}^{-1}$)
F15A	Varying NADH, fixed coenzyme Q_1	0.48 ± 0.02	8.35 ± 2.15	0.06
	Varying coenzyme Q_1 , fixed NADH	0.52 ± 0.03	9.48 ± 2.55	0.05
L323A	Varying NADH, fixed coenzyme Q_1	10.88 ± 0.57	45.23 ± 6.90	0.24
	Varying coenzyme Q_1 , fixed NADH	12.80 ± 0.95	16.20 ± 4.23	0.79
F354A	Varying NADH, fixed coenzyme Q_1	7.06 ± 1.25	36.59 ± 17.58	0.19
	Varying coenzyme Q_1 , fixed NADH	10.05 ± 1.34	26.85 ± 10.62	0.37

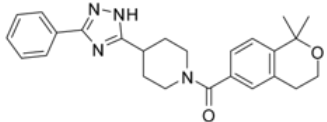
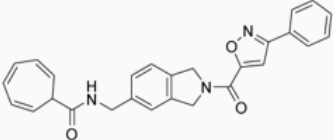
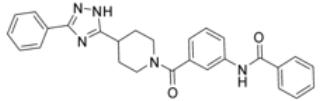
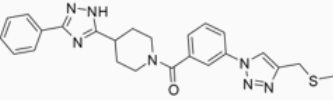
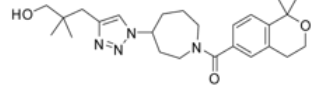
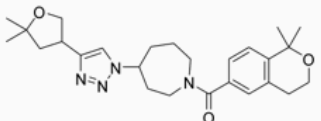
Supplementary Table 13. Biochemical Analysis of F15A, L323A and F354A FSP1 Mutants: Steady-State Kinetics. NADH oxidation was monitored by absorbance at 340 nm (pH 7.2) using 0.2–0.5 μM FSP1. Fixed concentrations for each assay were 100 μM NADH or coenzyme Q_1 , depending on the variable substrate. Both mutants retain enzymatic activity, though F15A exhibits reduced k_{cat} and L323A and F354A show increased K_M .

Compound	Structure	F15A IC ₅₀ (μM)	L323A IC ₅₀ (μM)
afi-FSP1-1		No inhibition	No inhibition
afi-FSP1-2		1.07 ± 0.07	0.490 ± 0.03
afi-FSP1-3		3.03 ± 0.08	No inhibition
afi-FSP1-4		No inhibition	No inhibition
afi-FSP1-5		No inhibition	No inhibition
afi-FSP1-6		No inhibition	No inhibition

Supplementary Table 14. F15A and L323A FSP1 Mutants IC₅₀. F15A and L323A mutants target residues involved in inhibitor binding, as revealed by crystal structures with compounds afi-FSP1-1, afi-FSP1-3, and afi-FSP1-4. Mutations abolish binding in most cases, except for afi-FSP1-3 (F15A retains binding) and compound afi-FSP1-2, which inhibits both mutants, suggesting a distinct binding mode or reduced reliance on the mutated side chains. Activities were measured by fluorescence-based resorufin formation at pH 7.2, using 100–200 μM NADH, 50 μM resazurin, and 0.1–0.5 μM FSP1. Compounds were tested across micromolar to low nanomolar concentrations. Experiments were performed with the racemic mixture of compound afi-FSP1-5 and the diastereomeric mixture of compound afi-FSP1-6. None of the compounds inhibited F354A.

Compound	Structure	K _i (μM)
afi-FSP1-1		3.56 ± 0.82
afi-FSP1-2		3.02 ± 0.78
afi-FSP1-3		~ 5.41 ± 2
afi-FSP1-4		No inhibition

Supplementary Table 15. K_i values (μM) of selected FSP1 inhibitors against NADH oxidase activity. Measurements were performed by monitoring NADH oxidation at 340 nm (pH 7.2), using inhibitor concentrations in the micromolar range and 1 μM FSP1. Data are shown in Supplementary Fig. 28. The K_i value for compound afi-FSP1-3 is an estimate due to poor solubility above 20–50 μM.

Compound	Structure	IC ₅₀ (μM)
afi-FSP1-1		0.382 ± 0.07
afi-FSP1-2		0.443 ± 0.22
afi-FSP1-3		0.912 ± 0.13
afi-FSP1-4		0.534 ± 0.06
afi-FSP1-5		6.43 ± 0.10
afi-FSP1-6		21.5 ± 0.10

Supplementary Table 16. IC₅₀ Values (μM) of the Most Potent FSP1 Inhibitors. The experiments were conducted using the racemic mixture of compound afi-FSP1-5 and the diastereomeric mixture of compound afi-FSP1-6. Activities were measured by monitoring fluorescence-based resorufin formation at pH 7.2, with fixed NADH concentrations (100–200 μM), 50 μM resazurin, and 0.05–0.1 μM FSP1. Inhibitors were tested in the low nanomolar to micromolar range. Full data are shown in Supplementary Fig. 29.

H ADME analysis of afi-FSP1-1, afi-FSP1-2, afi-FSP1-3, afi-FSP1-4

We performed a standard panel of in vitro Absorption, Distribution, Metabolism, and Excretion (ADME)/developability assays to assess oral bioavailability potential, including aqueous solubility, PAMPA permeability, buffer stability, and metabolic stability in mouse and human liver microsomes. Afi-FSP-1 and afi-FSP-4 were highly soluble (>75%), afi-FSP-3 showed modest solubility (32%), and afi-FSP-2 was poorly soluble. Consistent with these results, afi-FSP-1, -3, and -4 showed moderate PAMPA permeability (40–55% recovery), whereas afi-FSP-2 showed no measurable permeability (Supplementary Table 18). All four compounds were stable in buffer ($t_{1/2}$ >48 h). In mouse microsomes, afi-FSP-2 and afi-FSP-3 showed moderate metabolic stability ($t_{1/2}$ = 0.39 and 0.29 h, respectively), while afi-FSP-1 and afi-FSP-4 cleared rapidly ($t_{1/2}$ = 0.02 h). In human liver microsomes, afi-FSP-3 showed excellent stability ($t_{1/2}$ >2 h; scaled intrinsic clearance = 594.6 mL/min/kg) (Supplementary Table 19). Overall, afi-FSP-3 is the most promising starting point for a medicinal chemistry campaign, combining modest solubility with moderate permeability and excellent human metabolic stability. Afi-FSP-1 is a secondary candidate, offering superior solubility and permeability but requiring substantial improvements in metabolic stability to support preclinical development.

ADME Methods

Aqueous Solubility Assay

Solubility assays were performed using a Biomek FX lab automation workstation (Beckman Coulter, Inc., Fullerton, CA) and μ SOL Evolution software (pION Inc., Woburn, MA). In a 96-well microplate (Cat. No. 3363, Corning Incorporated, Salt Lake, UT), 10 μ L of a 1 or 10 mM test compound stock in DMSO was added to 190 μ L of 1-propanol (Thermo Scientific, spectroscopy grade, Cat. No. 434360010, Fair Lawn, NJ) to create a reference stock plate. From this reference stock plate, 5 μ L aliquots were mixed with 70 μ L of 1-propanol and 75 μ L of Dulbecco's Phosphate-Buffered Saline (DPBS, 1 $\times$; GibcoTM, Cat. No. 14190-144, Thermo Fisher Scientific, Waltham, MA) to generate the reference plate.

In a 96-well storage plate (Cat. No. 201276-100, Agilent, Santa Clara, CA), 6 μ L of the test compound stock was added to 600 μ L of DPBS, mixed, sealed, and incubated at room temperature for 18 hours. Solutions were then filtered through a 96-well filter plate (Cat. No. 110037, pION Inc., Woburn, MA). Fractions collected from the filtered sample plate were diluted with 1-propanol at a 1:1 (v/v) ratio to generate the sample plate. Both the reference and sample plates were analyzed by UV spectrometry (SpectraMax Plus 384 Microplate Reader, Molecular Devices, LLC, San Jose, CA). Concentration calculations and solubility values (μ g/mL) were determined using μ SOL Evolution software. All compounds were tested in triplicate wells.

Concentrations in the reference and sample plates were also assessed using UPLC-SQD-MS (Waters, Milford, MA). Solubility (μ M) was determined using the following equation:

$$\text{Solubility} = \frac{\text{Peak Area}_{\text{sample}}}{3} \times \frac{C_0}{\text{Peak Area}_{\text{reference}}}$$

where C_0 is the concentration of the DMSO stock solution divided by 100. All compounds were tested in triplicate wells.

Parallel Artificial Membrane Permeability Assay (PAMPA)

Six microliters of 1 or 10 mM compound in DMSO were dispensed into each well of a storage plate. The compounds were diluted 200-fold in DPBS to generate 600 μ L spiked solution, and 180 μ L of that was transferred to the donor plate (Preloaded stirwell PAMPA sandwich, Cat. No. 120551, pION Inc.). An artificial lipid (GIT-0 lipid solution, Cat. No. 110669)–coated acceptor plate containing 200 μ L of acceptor sink buffer (Cat. No. 110139, pION Inc.) was then placed over the donor plate to form the PAMPA sandwich. Plates were placed on the Gut-Box and stirred at room temperature for 30 min. Fractions were collected from both the donor and acceptor plates. Compound concentrations were quantified by UV spectrometry. All compounds were tested in triplicate wells.

Liver Microsomes Stability Assay

NADPH regenerating agent solutions A and B (Corning GentestTM, Cat. No: 451220) were purchased from Discovery Labware (Woburn, MA). Liver microsomes of mouse (CD-1, pooled, GibcoTM, Cat. No: MSMCPL) and human (mixed gender, pool of 50, Cat. No. H0630, Xenotech) were obtained from Thermo Fisher Scientific. Stock solutions of test compounds were prepared at 1 or 10 mM in DMSO. Sample preparation involved transferring 50 nL of stock solutions into six 96-well microplates (0, 0.25, 0.5, 1, and 2 hours) using a Labcyte Echo 650 acoustic dispenser. All compounds were tested in triplicate per time point. The reaction solution contained mouse liver microsomes (20 mg/mL), EDTA, NADPH regenerating agent, and 0.1 M potassium phosphate buffer (pH 7.4). Final concentrations applied: microsome protein 0.5 mg/mL, compound 1 or 5 μ M, NADPH A 1.3 mM, NADPH B 0.4 U/mL. Reaction plates were incubated at 37°C on an orbital shaker (100 rpm) and quenched with cold acetonitrile with internal standard. Supernatants were analyzed using UPLC–MS/MS. Metabolic stability was determined via first-order kinetics.

PBS Stability Assay

Compound degradation in DPBS was monitored at 0, 1, 3, 6, 24, and 48 hours. Stock solutions were dispensed into triplicate wells of a 96-well microplate using the Labcyte Echo 650 dispenser. DPBS was added to achieve a final compound concentration of 2 μ M. Plates were incubated at 37°C on an orbital shaker and quenched with cold acetonitrile with internal standard. Supernatants were analyzed using UPLC-MS/MS. PBS stability was evaluated by first-order kinetics.

UPLC/UV/MS System

Chromatographic separation was conducted using a Waters Acquity UPLC system coupled to both UV detector and single quadrupole mass spectrometer. Conditions were set as reported previously.

UHPLC-MS/MS System

Separation was performed on an Acquity UPLC BEH C18 1.7 μ m, 2.1 x 50 mm column (Waters Corporation, Milford, MA) using a Sciex ExionLC™ with 6500+ Qtrap system. Data were acquired using Analyst v1.7 and analyzed using OS software. Column temperature: 55 °C. Mobile phase A: 0.1% formic acid in MilliQ H₂O; Mobile phase B: 0.1% formic acid in acetonitrile. Flow rate: 0.9 mL/min with gradient: 0–0.2 min, B 1–1%; 0.2–0.5 min, B 1–50%; 0.5–1.6 min, B 50–95%; 1.6–1.95 min, B 95–95%; 1.95–1.96 min, B 95–1%; 1.96–2.2 min, B 1–1%. Mass spectrometer operated in positive-ion mode; MRM transitions for liver microsome and PBS stability assays were applied.

Compound	m/z (Q1 > Q3)	DP (V)	CE (V)	CXP (V)
SJ001127367-1	411 > 300	120	32	9
SJ001127368-1	411 > 300	110	31	9
SJ001127369-1	408 > 297	130	35	20
SJ001127371-1	408 > 297	110	33	9

Supplementary Table 17. MRM transitions used for analysis of the 4 FSP1 hits.

Compound	Avg. Solubility (UV, μ M)	% Solubility	Avg. Solubility (LC/MS, μ M)	Avg. Permeability (PAMPA, 10^{-6} cm/s)	% Recovery
afi-FSP1-1	76.5 $\pm$ 2.1	77.3	44.2 $\pm$ 0.7	885.7 $\pm$ 138.7	54.4 $\pm$ 1.6
afi-FSP1-2	1.02 $\pm$ 0.3	1.0	0.4 $\pm$ 0.1	0	0
afi-FSP1-3	31.1 $\pm$ 0.9	31.4	25.6 $\pm$ 0.2	197 $\pm$ 81.6	46.7 $\pm$ 17.6
afi-FSP1-4	75.42 $\pm$ 4.7	76.2	64.2 $\pm$ 2.9	240.1 $\pm$ 115.1	42.7 $\pm$ 17.4
Albendazole	2.4 $\pm$ 0.2	2.5	1.9	NA	NA
Carbamazepine	92.5 $\pm$ 6.6	93.4	64.3 $\pm$ 1.5	68.9 $\pm$ 40.6	39.2 $\pm$ 5.9
Verapamil	80.3 $\pm$ 6	81.1	42.1 $\pm$ 1.1	1251.2	64.1 $\pm$ 1.1

Supplementary Table 18. Solubility and PAMPA Permeability of afi-FSP1 compounds: Aqueous kinetic solubility was determined by UV spectrophotometry in Biomek automated 96 well format and by LC–MS–based quantitative analysis; values are reported as mean solubility (μ M) $\pm$ SD from $n = 3$ independent determinations and as % solubility. PAMPA permeability (Pe, 10^{-6} cm/s) was measured across an artificial lipid membrane, and data are shown as mean $\pm$ SD ($n = 3$); % recovery is also calculated. Albendazole, carbamazepine, and verapamil were included as reference compounds to benchmark low, moderate, and high permeability/solubility behavior.

Compound	PBS Stability $t_{1/2}$ (hr)	Mouse Metabolic $t_{1/2}$ (hr)	Mouse Metabolic CL_{int} (mL/min/kg)	Human Metabolic $t_{1/2}$ (hr)	Human Metabolic CL_{int} (mL/min/kg)
afi-FSP1-1	>48	0.02	2320.40	0.08	267.4
afi-FSP1-2	>48	0.39	253.90	NA	NA
afi-FSP1-3	>48	0.29	340.36	>2	594.6
afi-FSP1-4	>48	0.02	4343.87	0.07	409.5
Verapamil	NA	0.01	381.65	0.17	111.8

Supplementary Table 19. In vitro metabolic stability of afi-FSP1 compounds in Mouse and Human liver microsomes: Apparent in vitro half life ($t_{1/2}$, hr) and intrinsic clearance (CL_{int} , mL/min/kg) were determined from the depletion of parent compound in pooled human (HLM) and mouse liver microsomes (MLM) in the presence of NADPH. Data are reported as mean $t_{1/2}$ in hours from $n = 3$ incubations. Reference compound Verapamil is used as a high-clearance control.

I Experimental Validation of AdaptiveFlow using PARP1

I.1 Targeting PARP1. To evaluate the effectiveness of the AdaptiveFlow platform and the methodology presented, we have conducted a thorough experimental validation using Poly(ADP-ribose) polymerase 1 (PARP1) as our target. PARP1 is a nuclear enzyme critical in the DNA repair process, particularly in the base excision repair pathway. PARP1 is vital for maintaining genomic integrity by identifying and repairing DNA damage, such as single-strand breaks (145). Structurally, PARP1 is composed of distinct domains: a) a DNA-binding domain that enables it to recognize and bind to damaged DNA, a pivotal step for initiating the repair process; b) an auto-modification domain where it adds poly(ADP-ribose) chains to itself, thus regulating its activity and influencing interactions with other cellular components; and c) a catalytic domain that includes an alpha-helical subdomain (HD) and an ADP ribosyl transferase subdomain (CAT), the site of enzymatic activity (Fig. 36(a)) (146, 147). PARP1 uses NAD⁺ to transfer ADP-ribose units to target proteins, a critical process for DNA repair and various cellular responses (146). In recent years, PARP1 has gained significant attention in drug development and cancer therapy. Pharmacological inhibitors of PARP1 have been developed, mainly for treating cancers with deficiencies in homologous recombination DNA repair, such as those with BRCA mutations (11, 148). Clinical trials have confirmed the efficacy of PARP inhibitors in treating several types of cancer, including ovarian, breast, and prostate cancer. FDA-approved inhibitors, such as olaparib, rucaparib, and niraparib, exemplify this. Current research is extending their potential use in other types of cancer and in combination therapies (149).

I.2 Applying AdaptiveFlow to PARP1. To assess the effectiveness of AdaptiveFlow, our analysis was centered on two distinct crystal structures of the PARP1 enzyme in complex with different inhibitors localized within its catalytic domain. Our *in silico* screening approach specifically harnessed the crystal structure of PARP1 conjugated to olaparib (PDB 7AAD) (147) in the first docking scenario. This included both the active site and the regulatory helical domain (HD). Conversely, the second scenario employed the crystal structure of PARP1 complexed with the inhibitor UTT57 (PDB 6NRG) (150), where the HD subdomain was excluded. For docking processes, QuickVina 2 (65) served as the computational tool, alongside the application of the ATG-VS method for both scenarios. Following the *in silico* screening, we established rigorous selection criteria, enabling the identification of 80 promising candidates from each scenario for further experimental verification, as detailed in the Methods section (*Methods: Prioritizing Hits for Experimental Validation*). The preliminary evaluation comprised a PARP1 activity-based assay, which was then complemented by validation of selected hits through protein NMR, protein crystallography, and an examination of cellular activity via a colony formation assay (Fig. 30a).

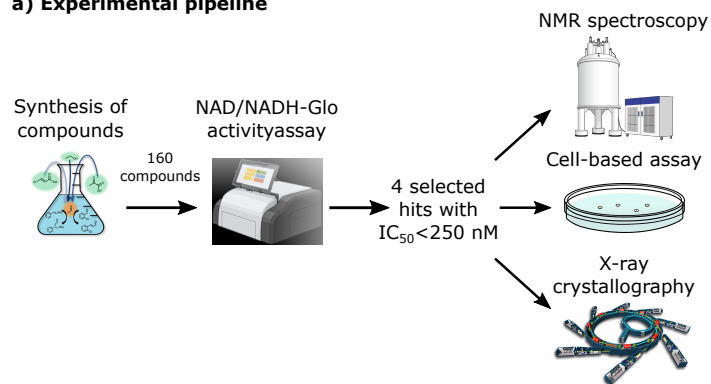
I.3 Selection and Characterization of Initial PARP1 Inhibitor Compounds. The initial phase of our experimental validation involved a dual-concentration enzymatic assay, conducted at 5 μ M and 50 μ M, to measure the inhibitory activity of 160 synthesized compounds (Fig. 30a). Through this assay, we performed duplicate experiments to ensure reproducibility and identified seven compounds that exhibited a notable inhibition of PARP1 activity, with an inhibition percentage exceeding 50% at the lower concentration 5 μ M. Notably, three compounds were associated with the first docking scenario, while the remaining four were aligned with the second scenario. Importantly, all seven compounds (Fig. 30b) demonstrated compatibility with the assay conditions, showing no signs of interference. To quantify the potency of these inhibitors, we proceeded with determining the half-maximum inhibitory concentration (IC₅₀) for each compound. The calculated IC₅₀ values are the result of three independent experiments and range from as low as 0.84 nM to approximately 1500 nM, indicating varying degrees of inhibitory efficacy between compounds. To further investigate iParp1 as a strong PARP1 inhibitor against the reference compound olaparib, we performed a complementary intracellular NanoBRET target-engagement assay and derived apparent K_i values by competitive tracer displacement, yielding $K_i = 5.49 \pm 1.68$ nM for iParp1 and $K_i = 6.73 \pm 0.35$ nM for olaparib (Supplementary Fig. 30b, Supplementary Table 20). Representative multi-tracer dose-response curves and the corresponding linear analysis of IC₅₀ versus tracer concentration used to extract $K_{i,app}$ are provided in Supplementary Fig. S27.

Ligand	Docking Model (PDB)	IC ₅₀ Exp 1 (nM)	IC ₅₀ Exp 2 (nM)	IC ₅₀ Exp 3 (nM)	Mean IC ₅₀ $\pm$ SD (nM)	K _i $\pm$ SD (nM)	SMILES
iParp1	6NRG	0.544	0.530	1.44	0.84 $\pm$ 0.52	5.49 $\pm$ 1.68	FC=1C=CC(CC2=NNC(=O)C=3C=CC=CC23)=CC1C(=O)N4CC5CC4C(=O)O5
iParp2	6NRG	75.5	42.6	67.0	61.7 $\pm$ 17.1	—	CC1=C(NC(=O)C=2C=CC=CC12)C(=O)NCC3=NC=4C(=CN=C5C=CC=CC45)N3
iParp3	7AAD	780	837	700	772 $\pm$ 68.8	—	O=C(N1CC2CC1CCN2C(=O)C3=NNC(=O)C=4C=CC=CC34)C=5C=CC=6NC=CC6C5
iParp4	7AAD	1000	793	992	928 $\pm$ 117	—	CC1CC(CN1C(=O)C2=NNC(=O)C3=CC=CC=CC23)NC(=O)C4=NNC=5C=CC(=O)CC45
iParp5	6NRG	1790	692	1360	1281 $\pm$ 553	—	O=C(NC1CC2(C1)CN(C2)C(=O)C3=CC=4C=CC=CC4C(=O)N3)C5=CC=CC5
iParp6	7AAD	1430	1470	1350	1417 $\pm$ 61.1	—	O=C(N1CC2CC1CCN2C(=O)C=3C=CC=4C=CC=CC4C3)C5=NNC(=O)C=6C=CC=CC56
iParp7	6NRG	809	808	766	794 $\pm$ 24.5	—	CC=1NN=CC1C(=O)N2CCC(C)(C2)NC(=O)C3=CNC(=O)C=4C=CC=CC34
Olaparib	-	0.597	0.377	0.607	0.53 $\pm$ 0.13	6.73 $\pm$ 0.35	O=C(N1CCN(C1)C(=O)C1CC(CCC1F)CC1[nH]c(=O)c2c1cccc2)C1CC1

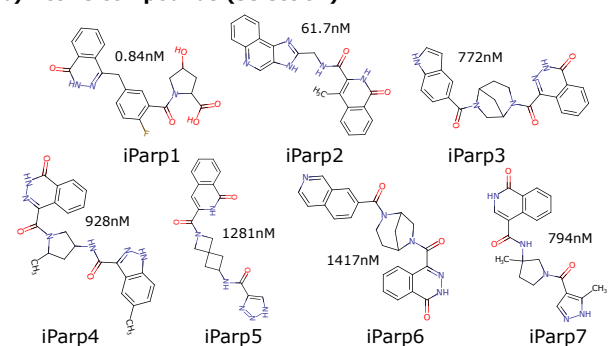
Supplementary Table 20. Independent IC₅₀ measurements (n=3), mean IC₅₀ $\pm$ SD, K_i $\pm$ SD values for iParp1 and olaparib, corresponding docking model (PDB ID), and SMILES for PARP1 inhibitors.

Subsequently, based on the IC₅₀ values determined from triplicate dose-response experiments, we selected four inhibitors with reproducible submicromolar potency (IC₅₀ < 1 μ M), comprising two representatives for each of the two docking scenarios/PDB structures used, for in-depth biophysical characterization (Supplementary Fig. 30a). Based on this criteria, four compounds (iParp1, iParp2, iParp3, and iParp4) were distinguished for further investigation studies. These inhibitors not only exhibited strong PARP1 inhibition in preliminary assays but also showed promise for significant biological activity, thus warranting additional validation through subsequent NMR spectroscopy, X-ray crystallography, and cell-based assays. Interestingly, hits resulting from computational screening using PDB 6NRG where the HD domain was absent, yielded the highest

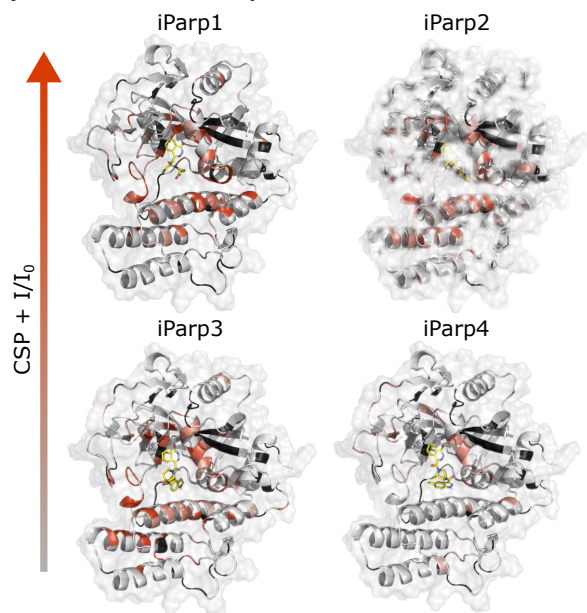
a) Experimental pipeline



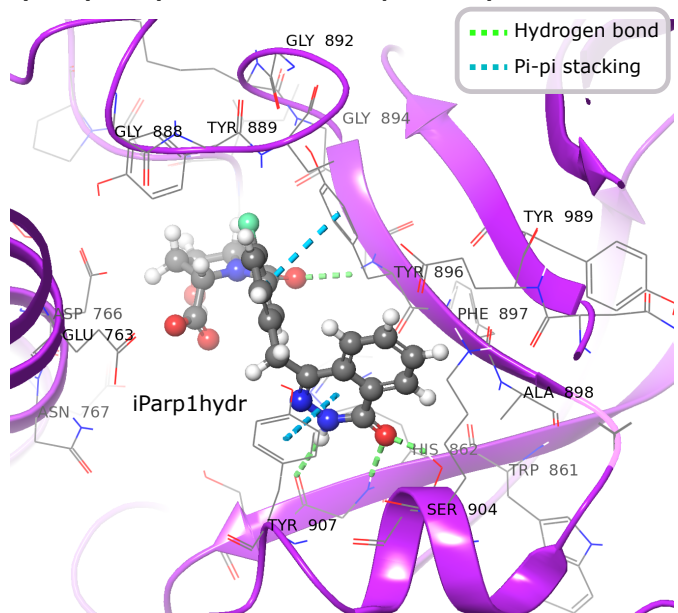
b) Active compounds (selection)



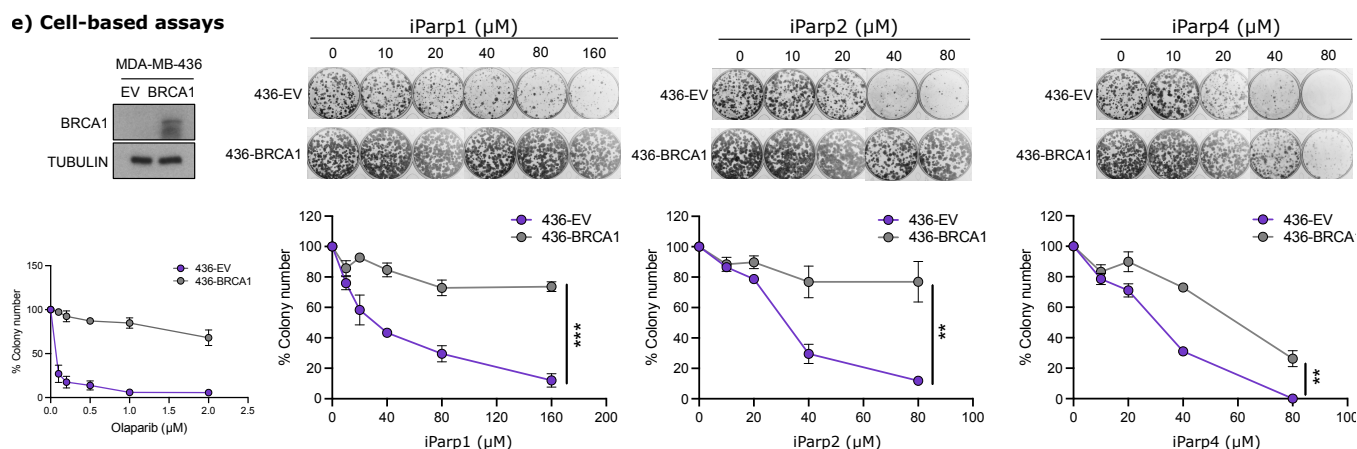
c) NMR - Chemical shift perturbations



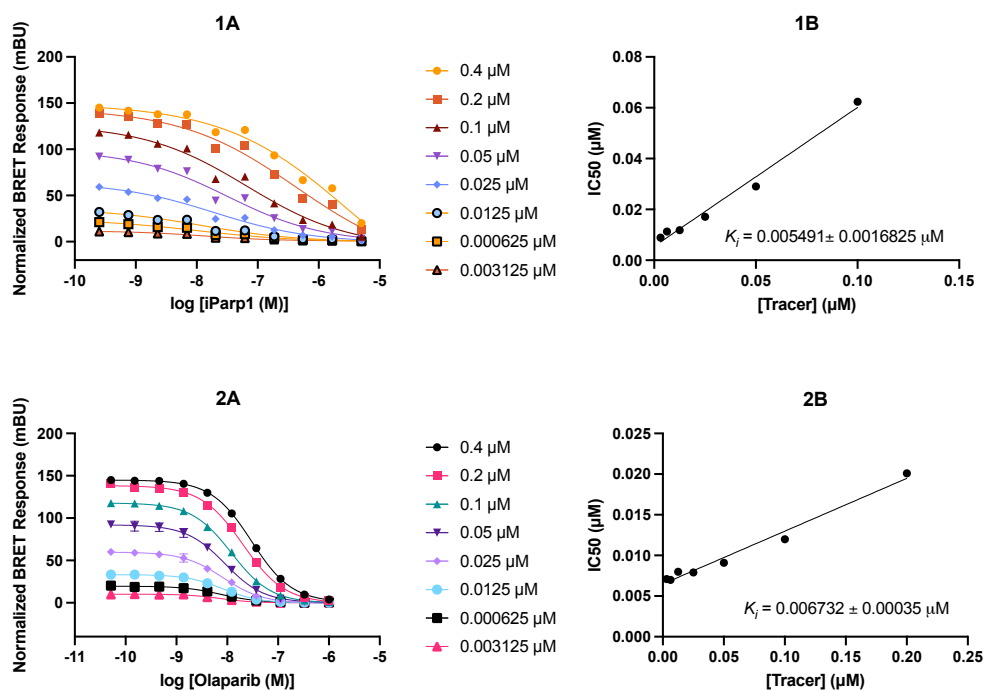
d) X-ray co-crystal structure of iParp1 in its hydrolized form



e) Cell-based assays



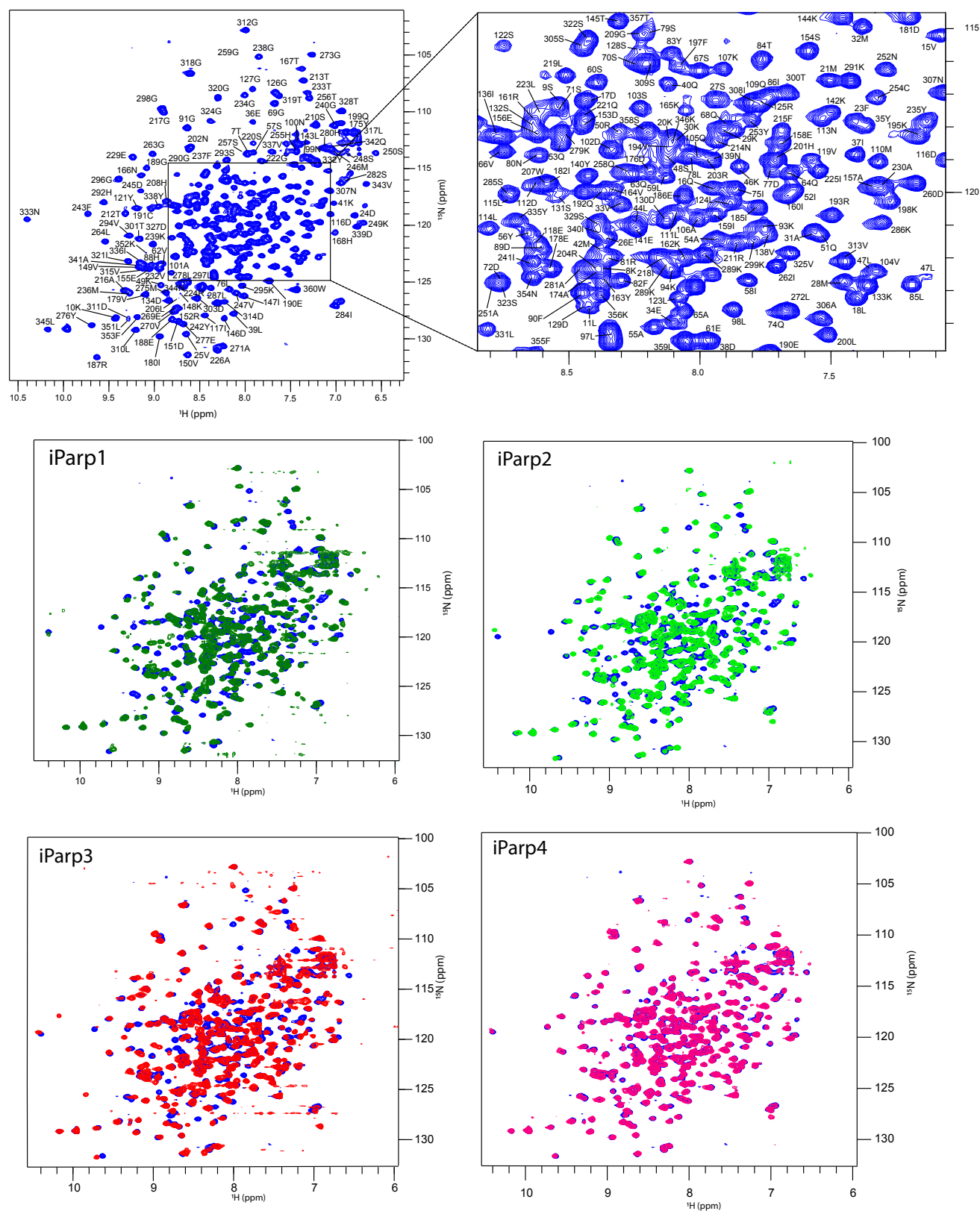
Supplementary Fig. 30. Experimental Validation of AdaptiveFlow with Human PARP1. (a) Schematic representation of the experimental workflow utilized to validate hits from virtual screening targeting the PARP1 enzyme. The pipeline integrates a suite of biochemical assays, biophysical methods, and cellular-based evaluations. (b) A selection of the most potent inhibitors confirmed through experimental assays, with their respective IC_{50} values. (c) Docking poses of PARP1 with inhibitors tested by NMR spectroscopy. Residues in red showed higher chemical shift perturbations (CSPs), or stronger broadening, than the mean plus one standard deviation (S1). Shades of red depict the combined size of peak broadening and CSPs. (d) Ribbon representation of the co-crystal structure of iParp1 bound to PARP1, detailing the molecular interactions within the catalytic site. Ball representation of iParp1, with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively. Hydrogen bonds and pi-pi stacking are represented as dashed blue and dashed green lines. (e) Cellular assays conducted on MDA-MB-436, a triple-negative breast cancer cell line, demonstrate the differential cytotoxic effects of the ligands, corroborating the targeted inhibition activity of the compounds. BRCA1 and tubulin protein levels were analyzed by western blot in BRCA1-mutated MDA-MB-436 expressing empty vector (EV) or BRCA1. Clonogenic assay of MDA-MB-436 control (436-EV) and its isogenic BRCA1-reconstituted pair (436-BRCA1) treated with the indicated compounds. Relative quantifications of colonies at the endpoint are shown at the top ($n=3$, mean $\pm$ SEM from three independent experiments) and representative images of the colonies are shown in the line plots.



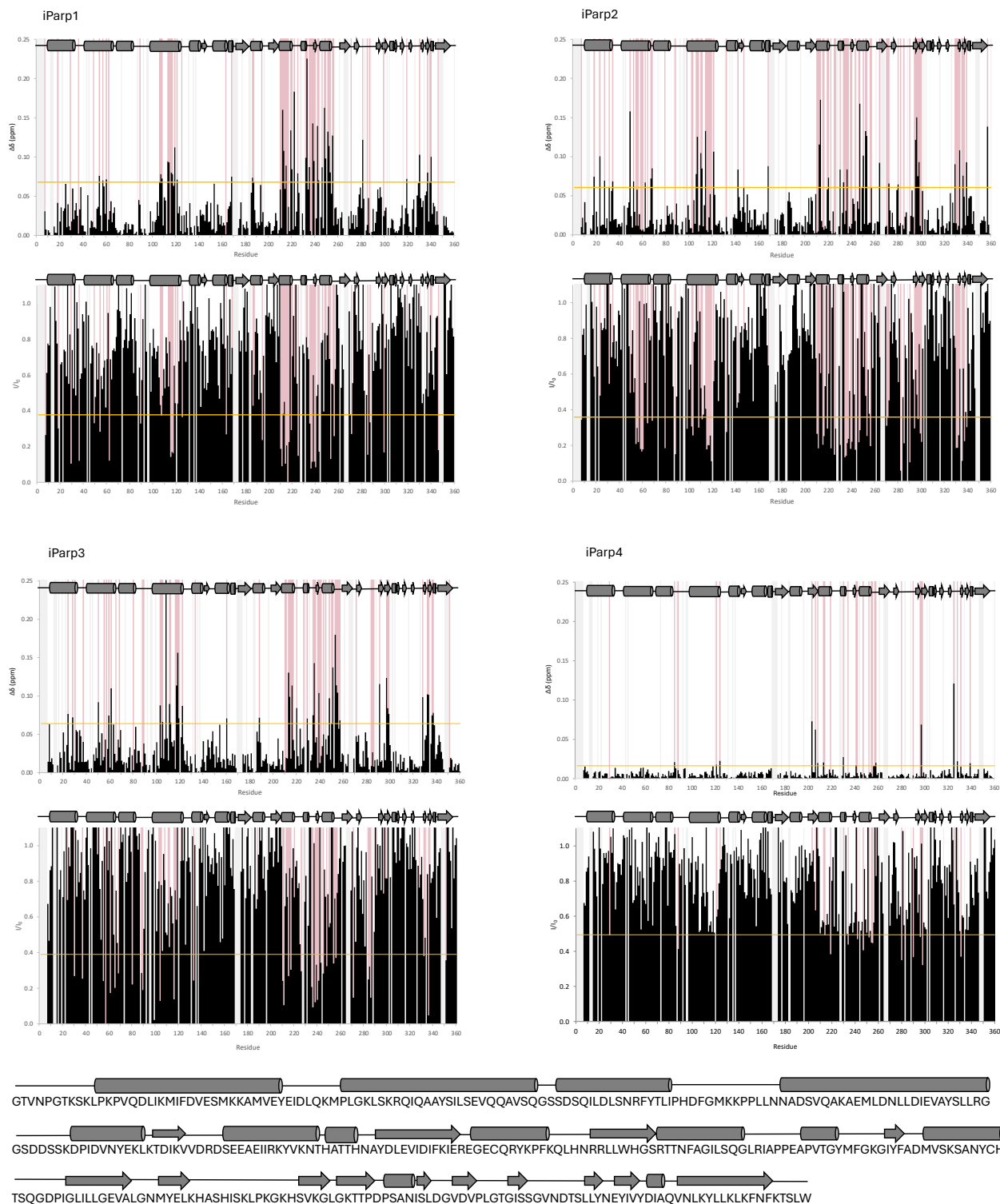
Supplementary Fig. 31. NanoBRET target engagement for PARP1. Representative multi-tracer dose–response curves for iParp1 and olaparib, and linear fits of IC₅₀ versus tracer concentration used to obtain $K_{i,\text{app}}$. Reported values: iParp1 $K_{i,\text{app}} = 5.49 \pm 1.68 \text{ nM}$; olaparib $K_{i,\text{app}} = 6.73 \pm 0.35 \text{ nM}$. $n = 1$

affinity molecules (iParp1, iParp2).

I.4 Characterization of PARP1-Inhibitor Interactions via Protein NMR Spectroscopy. The binding interactions between PARP1 and selected inhibitory compounds were comprehensively elucidated using protein NMR spectroscopy, revealing conclusive evidence of complex formation. We observed significant chemical shift perturbations (CSPs) and peak broadening in the PARP1 NMR spectra upon addition of each inhibitor, indicative of binding events. These CSPs were notable at both 1:1 and 1:2 molar ratios for inhibitors iParp2, iParp4, iParp1, and iParp3, as illustrated in Supplementary Fig. 30c. The perturbations predominantly occurred in regions proximal to PARP1’s active site, aligning with the binding locale forecasted by our computational docking simulations (refer to Supplementary Fig. 36a and 30d). Supplementary Fig. 32 and 33 detail the specific residues impacted and the extent of chemical shift perturbations consequent to ligand association.

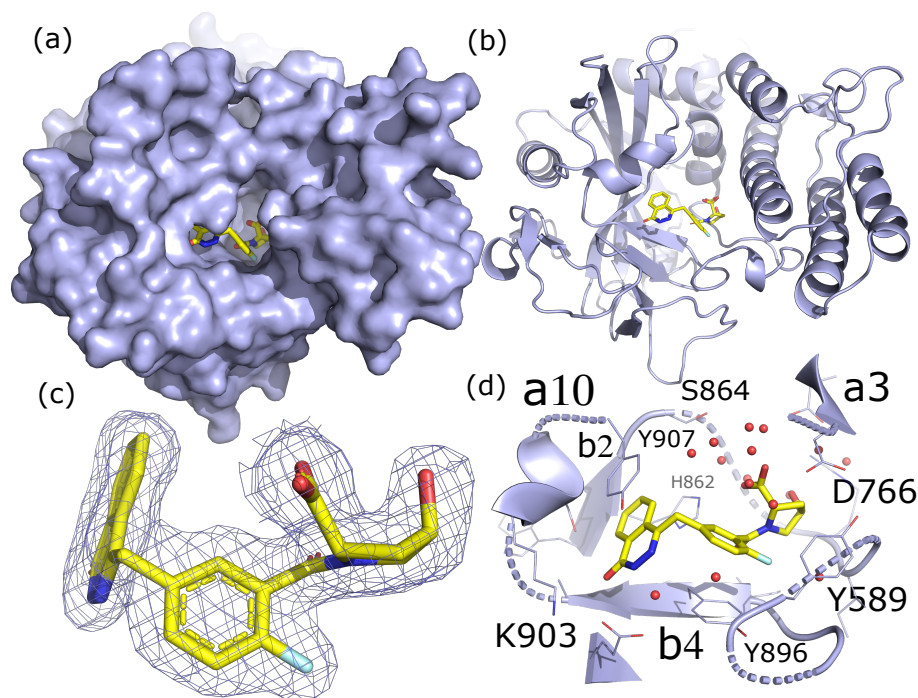


Supplementary Fig. 32. TROSY NMR spectra illustrating the interaction of PARP1 with various inhibitors. The assignment of resonances was based on BMRB entry 50454. Spectra are color-coded to indicate different conditions: Blue represents the apoprotein spectrum with 1% DMSO; red indicates the spectrum in the presence of inhibitor iParp3 at a 1:2 protein-to-inhibitor molar ratio; magenta corresponds to inhibitor iParp4 at a 1:1 ratio; dark green shows inhibitor iParp1 at a 1:2 ratio; and light green depicts inhibitor iParp2 at a 1:1 ratio. The shifts in resonance peaks reflect the binding and perturbation effects of the inhibitors on the PARP1 structure.



Supplementary Fig. 33. Nuclear Magnetic Resonance (NMR) Spectral Analysis of PARP1 Protein Interaction with Distinct Inhibitors. The graphs display the chemical shift perturbation (CSP) profiles and peak broadening profiles for the PARP1 protein upon binding with four different inhibitors: iParp1, iParp2, iParp3, and iParp4. For each inhibitor, the top panel shows the CSPs of backbone resonances in parts per million (ppm) and the bottom panel shows peak broadening (I/I_0). The horizontal yellow lines indicate the threshold for significant chemical shift changes, that is mean + 1 STD. Residues that exhibit CSPs above this threshold or signal intensity lower than the threshold are highlighted in red, suggesting regions of PARP1 that are affected by inhibitor binding. Grey background signifies missing assignment or proline residues. The secondary structure of PARP1 is depicted above each graph, with α -helices represented by cylinders and β -strands by arrows. The sequence below the graphs corresponds to the PARP1 regions shown.

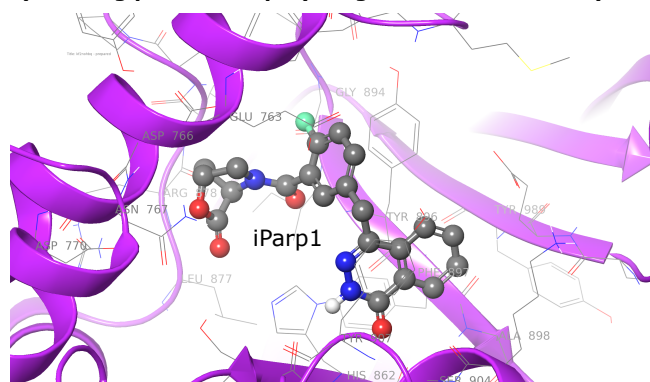
I.5 Co-Crystallization Experiments.. Our crystallization experiments resulted in the determination of the crystal structure of the isolated catalytic domain of human PARP1 (PDB:8VYH) in complex with the hydrolyzed version of iParp1 at a 2.05 Å resolution (Supplementary Fig. 30d). Data collection and refinement statistics can be found in Supplementary Table 21. The unbiased electron density allowed us to model the inhibitor molecule in the enzyme active site and unambiguously show the protein complex with the hydrolyzed form of iParp1 (iParp1hydr, Supplementary Fig. 34). The observed hydrolysis can be explained by the inherent ring strain of the azabicyclo moiety. Fig. Supplementary 35 shows in detail the chemical structures and corresponding 3D structures of iParp1 and its hydrolyzed form iParp1hydr. iParp1hydr binds at the center of the active site of PARP1 performing pi-pi stacking interactions with Tyr896 and Tyr907 (Supplementary Fig. 30d). Furthermore, a hydrogen bond is established between Tyr896 and the oxygen atom close to the hydroxypyrrolidine group. An important hydrogen bond interaction with Glu988 is also observed. To ascertain if AdaptiveFlow could have identified the iParp1hydr had it been present in the library, we conducted docking studies of both iParp1 and iParp1hydr to the PARP1 crystal structure obtained. Remarkably, the docking scores for iParp1 (-12.7 kcal/mol) and iParp1hydr (-12.5 kcal/mol) were comparable, indicating potential efficacy despite structural alterations induced by hydrolysis.



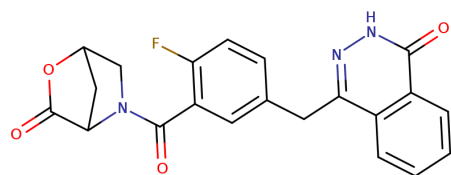
Supplementary Fig. 34. The Crystal Structure of the PARP1 Catalytic Domain in Complex with iParp1hydr. a, surface display of human PARP1 and the bound (carbons in yellow) inhibitor, represented in sticks. Oxygen, nitrogen, and fluorine atoms are colored in red, blue, and cyan, respectively. b, ribbon diagram (purple) of PARP1 in complex with iParp1hydr. The compound is in stick representation with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively. c, Representation of the electron density of iParp1hydr highlighting the 4-hydroxypyrrolidine-2-carboxylic acid moiety. The compound is in stick representation with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively. d, Representation of iParp1hydr and the interactions with its neighboring residues within the catalytic domain active site. The compound is in stick representation with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively.

I.6 Cellular Activity of Identified Hits. The cellular impact of the identified hits was further delineated through clonogenic assays conducted on MDA-MB-436 cells, a triple-negative breast cancer (TNBC) line with a known BRCA1 5396 + 1G>A mutation. This particular mutation diminishes BRCA1 protein levels, leading to increased sensitivity to PARP1 inhibitors (151, 152) (Supplementary Fig. 30e). To ascertain the specificity of our compounds towards PARP1 in cancer cells, we employed MDA-MB-436 cells harboring an empty vector (control) and their isogenic counterpart expressing wildtype BRCA1. Our findings revealed that the control MDA-MB-436 cells exhibited pronounced sensitivity to iParp1 and iParp2, suggesting these compounds' efficacy in targeting PARP1 within tumor cells. Conversely, cells reintroduced with wildtype BRCA1 exhibited resistance to these inhibitors, underscoring the specificity of iParp1 and iParp2 for PARP1-mediated pathways in the context of BRCA1 deficiency (Supplementary Fig. 30e). iParp4 presented a concentration-dependent inhibitory effect on 436-EV cells, with significant cytotoxicity observed even in 436-BRCA1 cells at higher doses, suggesting a broader range of activity that may transcend PARP1 inhibition. Notably, iParp3 did not exert significant cytotoxicity in either cell line, aligning with the hypothesis of its limited utility in targeting PARP1. Overall, the observed effect of the tested inhibitors on the MDA-MB-436 cells was less significant than the affinity observed with our biochemical assay, probably due to the molecules' limited cell penetration profiles. Such is the case of iParp1, which in its hydrolyzed version bears a carboxylic acid, which

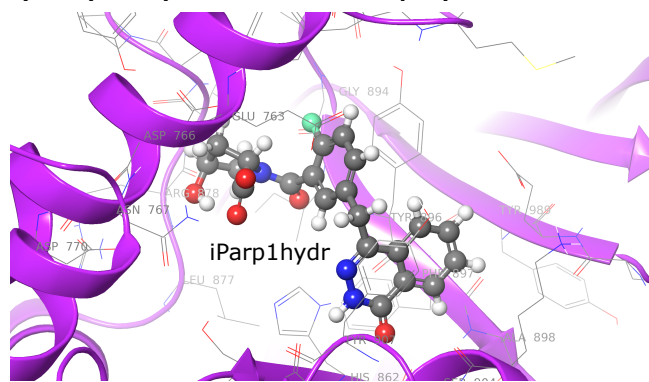
a) Docking pose of iParp1 (using PDB structure 7AAD)



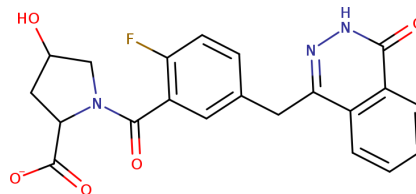
c) 2D representation of iParp1



b) X-ray co-crystal structure of iParp1hydr



d) 2D representation of iParp1hydr



Supplementary Fig. 35. Chemical and Structural Characterization of iParp1 and its Hydrolyzed Form iParp1hydr. a, Ribbon representation of the docking pose of the virtual screened iParp1 bound to PARP1. The in-silico screening was performed using the PDB structure 7AAD. Ball representation of iParp1, with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively. Hydrogen bonds and pi-pi stacking are represented as dashed blue and dashed green lines. b, Ribbon representation of the co-crystal structure of iParp1hydr with PARP1. Ball representation of iParp1, with carbon, nitrogen, oxygen, and fluorine atoms colored in yellow, blue, red, and cyan, respectively. Hydrogen bonds and pi-pi stacking are represented as dashed blue and dashed green lines. c, Chemical structure of iParp1. d, Chemical structure of iParp1hydr..

consequently leads to poor cell permeability.

Statistical Analysis. Colony formation data are represented as mean $\pm$ SEM from three independent experiments, and significance was determined by unpaired, two-tailed Student's t-test.

I.7 Comparison to Approved Drugs. A noteworthy outcome of the *in silico* screening conducted using AdaptiveFlow was the identification of a few hits resembling olaparib, an FDA-approved PARP inhibitor. Despite no prior information on known inhibitors being used by AdaptiveFlow during the screens, VirtualFlow A2I was able to discover a few hits (iParp1, iParp3, and iParp6) that are similar to olaparib in structure and also potency. iParp1 has a similar IC₅₀ value than olaparib (single digit nanomolar). Olaparib has undergone extensive classical medicinal chemistry during its development, while our compounds come directly from a virtual screen without any optimization, highlighting the efficiency of our approach and increasing the validity of the discovered ligands. All our compounds exhibit distinctiveness in comparison to previously reported inhibitors, and the inhibitors discovered herein through AdaptiveFlow are not covered by prior patents. Utilizing Morgan fingerprints (153) to assess structural similarity, it was observed that among our discovered inhibitors, iParp1, iParp3, and iParp6 exhibited Morgan fingerprint similarities of 0.58, 0.40, and 0.39, respectively. Conversely, iParp2, iParp4, iParp5, and iParp7 are rather different from olaparib, highlighting that our platform can discover novel and diverse sets of molecules with a range of structural features. Olaparib, characterized by its N-acylpiperazine structure with a cyclopropane group and a monofluoro-benzene on an oxo-phthalazine scaffold, shares the oxo-phthalazine core with the first docking scenario series hits, iParp1 and iParp5. This core is altered to a hydroxyisoquinoline in iParp2 and iParp7, while the cyclopropyl N-acylpiperazine feature is replaced with acylated 2,6-diazabicyclo[3.2.1]octane in iParp6 and iParp3. Notable is the modification in iParp1, where the N-acylpiperazine is substituted with a bicyclic lactone prone to hydrolysis into a stable hydroxyproline derivative, a transformation confirmed through NMR and mass spectroscopy and observable in the crystal structure. Remarkably, the docking scores for iParp1 (-12.7 kcal/mol) and iParp1hydr (-12.5 kcal/mol) were comparable, indicating potential efficacy despite structural alterations induced by hydrolysis.

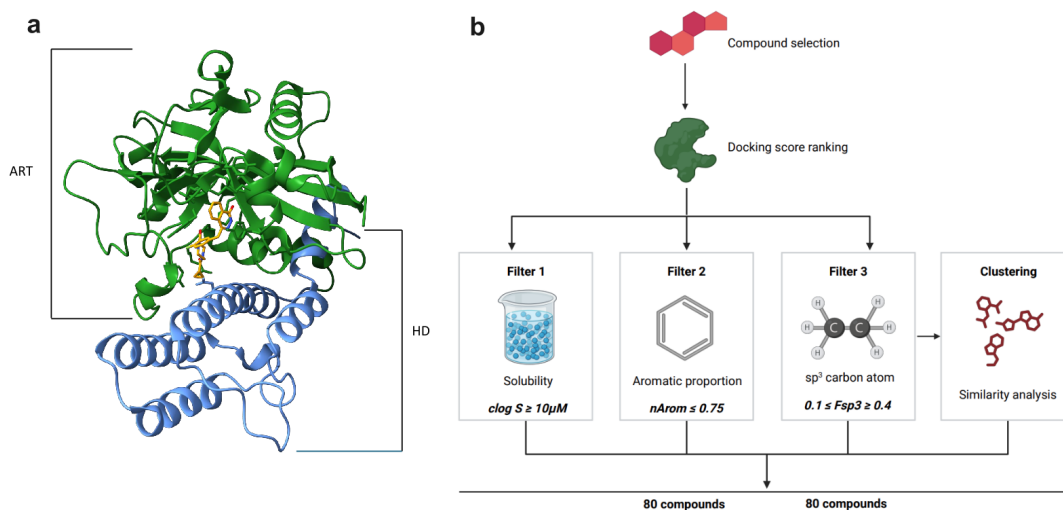
I.8 Reagents. Sodium chloride, hepes buffer, phenylmethylsulfonyl fluoride, glycerol, SigmaFast™ Protease Inhibitor Cocktail Tablets EDTA-Free, Ethylenediaminetetraacetic acid (EDTA), tris(2-carboxyethyl) phosphine (TCEP), and dithiothreitol (DTT) were purchased from Sigma-Aldrich. All tested compounds were purchased from the ChemSpace/Enamine supplier. NAD/NADH-Glo assay kit was purchased from Promega (Madison, WI). [¹⁵N] ammonium chloride, U-¹³C₆D-glucose, alpha-ketobutyric acid-¹³C₄,3,3-D₂ and alpha-ketoisovaleric acid-U-¹³C₅ were purchased from Cambridge Isotope Laboratories.

1521 alpha/beta-Tubulin Antibody was purchased from Cell Signaling Technology (# 2148S).

1522 **I.9 Receptor Preparation.** The receptor structures for the first and second docking scenarios have PDB ID 7AAD and 6NRG,
1523 respectively. The HD domain was present in the first docking scenario, and was absent in the second docking scenario. The
1524 structures were prepared with the Protein Preparation Wizard of Maestro by Schrödinger (hydrogens, protonation states, bond
1525 orders) (154). MGLTools with AutoDockTools was used to convert the receptor PDB files into PDBQT files, and to set up the
1526 docking boxes (155).

1527 **Docking Scenario Setup.** The protein was set up to be rigid during the docking, and QuickVina 2 was used as the
1528 docking program for both docking scenarios and both screening stages (ATG prescreen and ATG primary screen). The
1529 catalytic domain was targeted in both docking scenarios. In the first docking scenario (with HD domain), the docking
1530 box had a size of $20.0 \times 16.0 \times 16.0$ Å, while in the second docking scenario (without HD domain) the docking box
1531 had a size of $16.0 \times 24.0 \times 20.0$ Å. In the ATG prescreen, one representative molecule was chosen per tranche, resulting in
1532 approximately 12 million compounds in total. During the ATG primary screen, a total of 100 million compounds were screened.
1533

1534 **I.10 Prioritizing Hits for Experimental Validation.** Postprocessing of the results was done with DataWarrior. Basic filters
1535 have been applied, such as $\log P < 5$, and compounds with problematic functional groups and predicted toxicity (mutagenic
1536 compounds, compounds with reproductive effects, tumorigenic compounds) were removed. In addition to the ranking by
1537 docking score (compounds displayed docking scores as high as -15.1 kcal/mol), we implemented a filtering and prioritization
1538 routine to identify a set of diverse drug-like compounds within the hit list (Supplementary Fig. 30b) (156). The hits were filtered
1539 by their aqueous solubility, setting a clogS of at least -5 (10 μM). Many studies have suggested that the aromatic proportion of
1540 a molecule has a detrimental impact on solubility (157, 158); for this reason, we filtered out compounds displaying aromaticity
1541 higher than 75%. In parallel, the flatness of the library was further assessed by leveraging the importance of an increased number
1542 of sp^3 hybridized carbons in the increase of positive receptor-ligand interactions, as well as leading to increased solubility (28).
1543 As such, the degree of saturation (F_{sp^3}) was calculated (28), and compounds with values ranging from 0.1–0.4 were given
1544 preference (159–161). Finally, clustering was performed using the DataWarrior software to obtain a diverse set of compounds
1545 for experimental validation. Supplementary Fig. 36b summarizes the filtering pipeline implemented in this study. All the
1546 compounds from the study were provided by Enamine Ltd. (Kyiv, Ukraine).



Supplementary Fig. 36. a. Model of PARP1 catalytic domain highlighting the alpha-helical subdomain (HD) in blue and the ADP ribosyl transferase subdomain (CAT), in green. The structure is displayed with the inhibitor olaparib (in yellow) bound at the active site. b. Graphic representation of the pipeline used for compound filtering – besides docking score ranking, water solubility, and molecule flatness were considered. The compounds were filtered for nasty functional groups and pan-assay interference compounds (PAINS). Clustering analysis was applied using the DataWarrior software.

1547 **I.11 Protein Expression and Purification.** PARP1 catalytic domain (residues 661–1014, full-length CAT) was expressed as
1548 described (162, 163). Unlabeled and ^{15}N -labeled PARP1 was expressed using *E. coli* BL21(DE3) cells in Lysogeny Broth (LB)
1549 or M9 medium, respectively, supplemented with 50 $\mu\text{g}/\text{ml}$ kanamycin. Protein expression was induced with 0.2 mM isopropyl
1550 β -D-1-thiogalactopyranoside (IPTG) and performed at 16°C overnight.

1551 The labeled and unlabeled PARP1 catalytic domain was purified by incubation with nickel-nitriloacetic acid (Ni-NTA) agarose
1552 resin (Qiagen) for 3h at 4 °C. The resin was washed with 50 ml HEPES, pH 8.0, 500 mM NaCl, and 0.5 mM TCEP. The protein
1553 was eluted with 50 ml HEPES, pH 8.0, 100 mM NaCl, 300 mM imidazole, and 0.5 mM TCEP and injected into a Resource Q
1554 column for DNA separation. Unlabeled protein was purified by size exclusion chromatography in 50 mM HEPES, pH 8.0, 150

1555 mM NaCl, and 0.5 mM TCEP. PARP1 catalytic domain purified for NMR labeled purposes was purified using 10 mM sodium
1556 phosphate, 222 mM sodium chloride, 2.7 mM potassium chloride, and 2 mM DTT.

1557 **I.12 NADH/NAD⁺ Activity-Based Assay.** NAD⁺/NADH was measured using the NADH/NAD-Glo Assay from Promega
1558 according to the manufacturer's recommendations. Briefly, the enzyme/histones substrate mix prepared in the reaction buffer
1559 was added to the reaction plate wells (Corning 3572, Non-Treated). Compounds in 100% DMSO were added into the enzyme
1560 mixture by acoustic technology (Echo550; nanoliter range). 100% DMSO was delivered to no compound/DMSO control wells.
1561 The reaction plate was centrifuged following a pre-incubation of 20 min at room temperature. The reaction was initiated
1562 by adding NAD⁺ followed by an incubation of 2 hours at room temperature. NAD/NADH-Glo™ prepared as per the kit
1563 instructions were added to all assay wells. After incubation of 30 minutes at room temperature in the dark, luminescence with
1564 an endpoint read at 30 minutes was recorded. Data were compared with DMSO-treated cells and expressed as % control. Data
1565 obtained is the result of three independent experiments (n=3).

1566 **NanoBRET Target Engagement (Intracellular PARP1)**

1567 NanoBRET® Target Engagement assays were performed essentially following the Promega protocol with minor modifications.
1568 HEK293 cells were transiently transfected with a PARP1–NanoLuc® fusion construct using FuGENE® HD (Promega). Test
1569 compounds were pre-spotted into 384-well assay plates using an Echo 550 acoustic dispenser (Labcyte). Transfected cells were
1570 harvested, mixed with the cell-permeable NanoBRET TE PARP Tracer-01, and dispensed into the plates. Plates were incubated
1571 for 1 h at 37°C in 5% CO₂. NanoBRET® Nano-Glo® Substrate plus Extracellular NanoLuc® Inhibitor solution was then added,
1572 followed by a 20 min incubation at room temperature. Donor (460 nm) and acceptor (600 nm) emissions were recorded on an
1573 EnVision 2104 multimode plate reader (PerkinElmer). The NanoBRET ratio was computed as:

$$\text{BRET Ratio} = \frac{\text{Acceptor}_{\text{sample}}}{\text{Donor}_{\text{sample}}} - \frac{\text{Acceptor}_{\text{no tracer}}}{\text{Donor}_{\text{no tracer}}}.$$

1574 Concentration–response curves were fitted in GraphPad Prism using a four-parameter logistic model to obtain apparent IC₅₀
1575 values. Apparent K_i values were then obtained by competitive tracer-displacement analysis, correcting each IC₅₀ for the tracer
1576 concentration and its apparent affinity for PARP1 (i.e., $K_i \approx \text{IC}_{50} / [1 + [\text{tracer}] / K_{D,\text{app}}]$). Assay conditions were set to support
1577 accurate affinity estimation by keeping tracer occupancy modest, controlling expression levels, minimizing ligand depletion and
1578 nonspecific binding, and fitting a single-site competition model.

1579 **I.13 NMR Spectroscopy.** The NMR titration samples were prepared by adding the unlabeled inhibitors (iParp1, iParp2, iParp3
1580 and iParp4) into solutions of U-[¹⁵N, D]-labeled PARP1 to reach a ratio of 1:2 (50 μM protein and 100 μM olaparib, iParp1
1581 or iParp3) or 1:1 for inhibitors with limited water solubility (25 μM protein, and 25 μM iParp2 or iParp4). 500 μL of each
1582 sample was transferred into a 5 mm NMR tube and 2D-¹⁵N-TROSY spectra were acquired at 298 K on a Bruker Avance III
1583 HD spectrometer equipped with a TCI cryoprobe and z-shielded gradients, operating at a ¹H frequency of 800 MHz. Data
1584 were processed using NmrPipe (164) and analyzed with CCPNmr (165). To transfer PARP1 CAT-domain assignment, HNCA
1585 and HNCACB spectra were acquired and the obtained frequencies were compared to BMRB entry 50454. Chemical shift
1586 perturbations and peak broadening were extracted using CCPNmr and their combined normalized value was plotted on the
1587 PARP1 docking poses.

1588 **I.14 Crystallography.** PARP1 was co-crystallized with iParp1, the most potent of our inhibitors. The details are described
1589 below.

1590 **Protein Expression and Purification.** The human PARP1 construct (residues 662-1011) contains an N-terminal GST tag
1591 and 3C cleavage site, was overexpressed in *E. coli* BL21 (DE3), and purified using affinity chromatography and size-exclusion
1592 chromatography. Briefly, cells were grown at 37°C in TB medium in the presence of 100 μg/mL of ampicillin to an OD of
1593 0.8, cooled to 17°C, induced with 400 μM isopropyl-1-thio-D-galactopyranoside (IPTG), incubated overnight for 20 hrs at
1594 17°C, collected by centrifugation, and stored at -80°C. Cell pellets were lysed in buffer A (25 mM HEPES, pH 7.5, 200 mM
1595 NaCl, and 1 mM TCEP) using a Microfluidizer (Microfluidics), and the resulting lysate was centrifuged at 30,000g for 40 min.
1596 Glutathione beads (Cytiva) were mixed with cleared lysate for 90 min and washed with buffer A. Beads were transferred to
1597 an FPLC-compatible column, and the bound protein was washed further with buffer A supplemented with 1 M NaCl for 10
1598 column volumes, followed by buffer A for 10 column volumes. The GST-tag was cleaved on column by adding 3C protease to
1599 the washed beads and incubating overnight in the cold room. The cleaved PARP1 was eluted from the column, then concentrated
1600 and purified further using a Superdex 75 16/600 column (Cytiva) in buffer A. The fractions containing cleaved PARP1 were
1601 concentrated to 6.5 mg/mL and stored at -80°C.

1602 **Crystallization.** A solution of 6.5 mg/mL human PARP1 was crystallized in 30% PEG-4000, 200 mM Lithium sulfate, and 100
1603 mM Tris, pH 8.5 by sitting-drop vapor diffusion at 20°C. Crystals were transferred briefly into crystallization buffer containing
1604 25% glycerol prior to flash-freezing in liquid nitrogen.

Structure Determination. Diffraction data were collected at beamline 17-ID-2 of the National Synchrotron Light Source II (Brookhaven National Laboratory). Datasets were integrated and scaled using XDS (137). Structures were solved by molecular replacement using the program Phaser (166) and the search model (PDB code 7KK2). Iterative manual model building and refinement using Phenix (167) and Coot (168) led to a model with excellent statistics summarized in Supplementary Table 21.

Parameter	Value
PDB Code	8VYH
Resolution range	46.28 - 2.05 (2.09 - 2.05)
Space group	P 21 21 21
Unit cell	48.92 93.14 165.55 90 90 90
Total reflections	189094 (12142)
Unique reflections	45179 (2880)
Multiplicity	4.2 (4.2)
Completeness (%)	94.54 (95.96)
Mean I/sigma(I)	4.12 (70.0)
Wilson B-factor	30.65
R-merge	0.2088 (1.529)
R-pim	0.1064 (0.773)
CC1/2	0.987 (0.323)
CC*	0.997 (0.699)
Reflections used in refinement	45134 (2853)
Reflections used for R-free	2275 (147)
R-work	0.1971 (0.3110)
R-free	0.2504 (0.3614)
CC(work)	
CC(free)	
Number of non-hydrogen atoms	6123
macromolecules	5506
ligands	90
solvent	527
Protein residues	700
Nucleic acid bases	
RMS(bonds)	0.003
RMS(angles)	0.55
Ramachandran favored (%)	99.14
Ramachandran allowed (%)	0.86
Ramachandran outliers (%)	0.00
Rotamer outliers (%)	0.16
Clashscore	12.26
Average B-factor	
macromolecules	35.77
ligands	35.49
solvent	41.46
Number of TLS groups	

Supplementary Table 21. Crystallographic Data and Refinement Statistics for PARP1-iParp1 complex structure.

I.15 Colony Formation Assays. MDA-MB-436 cells (1000/well) were seeded in 6-well plates, treated with graded concentrations of the indicated compounds the next day, and grown in the presence of drugs for 15 days. Colonies were fixed with ice-cold methanol for 10 minutes and then stained with 1% crystal violet for 30 minutes. Plates were imaged with GelCount (Oxford Optronix), and quantification of the colony numbers was performed with ImageJ.

1613 J Computational Overhead

1614 AdaptiveFlow employs a hierarchical orchestration approach specifically designed to minimize computational overheads at the
1615 scale required for large-scale molecular docking operations. At the fundamental level of this system, individual ligand docking
1616 operations require between several seconds to several minutes of computation time on a single CPU core, with this duration
1617 varying considerably based on the complexity of the molecular structures involved and the specific parameters employed in the
1618 docking calculation. When executing distributed docking operations at scale, one will encounter multiple sources of overhead
1619 that, while individually small, can compound to create significant inefficiencies. These overhead sources include the workload
1620 submission scheduling through systems such as Slurm or AWS Batch, job startup communication protocols, the transfer of input
1621 data files containing ligand information, runtime initialization overhead particularly from Java Virtual Machine initialization
1622 in Java-based docking programs, the storage and upload of computational results, and finally the job completion overhead
1623 associated with scheduler communication. To illustrate the significance of these overheads, consider a conservative scenario
1624 where each non-docking operation requires merely 0.1 seconds of computation time, an estimate that underestimates actual
1625 overhead in many real-world cases. For a docking operation requiring 10 seconds of computation, these six overhead sources
1626 would collectively contribute approximately 0.6 seconds, representing a 6% performance penalty per ligand. When this analysis
1627 is extrapolated to operations involving one million ligands, the cumulative overhead becomes substantial and would compound
1628 significantly without appropriate mitigation strategies.

1629 **J.1 Hierarchical Strategy.** AdaptiveFlow addresses this scaling challenge through a three-layer hierarchical architecture that
1630 systematically amortizes overhead costs across multiple docking operations.

1631 **Data Layer.** At the foundation of this hierarchy lies the data layer, where ligands are assembled into collections of approx-
1632 imately 1,000 molecules each. These collections are subsequently compressed into archive files, a strategy that serves dual
1633 purposes: it substantially reduces the file and object count that must be managed by the storage system, and it amortizes
1634 input/output costs across the many docking operations that will be performed on ligands within each collection.

1635 **Compute Layer Architecture.** The compute layer operates through independent computational tasks termed "subjobs," each
1636 of which processes one or more ligand collections. These subjobs are typically dimensioned to execute for 45 to 90 minutes
1637 of runtime on computational nodes equipped with 16 virtual CPUs, corresponding to 8 physical processor cores (this sizing is
1638 configurable). This particular sizing rationale reflects optimization for cloud-based spot instances and preemptible instances
1639 available on high-performance computing platforms, where this configuration typically yields cost savings of 50-60% while
1640 maintaining compatibility with HPC scheduler backfill efficiency. Importantly, these parameters remain configurable to ac-
1641 commodate the specific requirements and constraints of different computational environments. Each subjob executes a custom
1642 Python program, referred to as the AdaptiveFlow Runner, which leverages standard Python libraries. Notably, the implemen-
1643 tation employs Python's multiprocessing package to circumvent the Python Global Interpreter Lock (GIL), thereby enabling
1644 true parallel execution. The Runner implements a processing pipeline comprising several connected stages: the initial down-
1645 load or copy of ligand collections from storage, decompression of these archives, execution of docking calculations for each
1646 individual ligand, aggregation and summarization of computational results, and finally upload of compressed result files back
1647 to persistent storage. For docking programs that use Java, each Runner will share a single JVM through Nailgun, removing the
1648 traditional JVM startup time from each docking process. This queue-based architecture incorporating dynamic worker threads
1649 provides several critical advantages. First, it enables overlapping of input/output operations with computational work, allowing
1650 the next collection to download while docking calculations proceed on the current dataset. Second, the architecture supports
1651 configurable oversubscription, wherein more docking threads than physical cores can be instantiated, which has empirically
1652 demonstrated throughput improvements as the overhead itself is reduced rather than increased by this approach. Third, the
1653 system performs batch compression and storage of results, further amortizing input/output costs across multiple ligands.

1654 **Scheduler Layer.** At the highest level of the hierarchy, the scheduler layer addresses the rate limits imposed by HPC schedulers
1655 on job submission. AdaptiveFlow introduces the concept of "workunits," configurable bundles of subjobs that map to scheduler
1656 array jobs. Rather than submitting thousands of individual jobs to the scheduler, which would incur substantial overhead, the
1657 system submits a single array job containing thousands of array elements. For instance, instead of submitting 1,000 separate job
1658 requests, AdaptiveFlow submits one array job with 1,000 constituent elements, thereby dramatically reducing scheduler over-
1659 head and the associated job management costs. AdaptiveFlow can submit to either Slurm or AWS Batch using their respective
1660 syntax for array jobs. While generic orchestration frameworks exist within the scientific computing ecosystem, AdaptiveFlow
1661 implements a purpose-built model designed specifically for the molecular docking workload pattern. This design decision
1662 serves three primary objectives: minimizing software dependencies and deployment complexity, optimizing specifically for the
1663 characteristic patterns of docking workflows, and maintaining portability across both traditional HPC environments and cloud
1664 computing platforms while optimizing for the specific characteristics of each. The effectiveness of this hierarchical approach is
1665 demonstrated through empirical analysis of the system's scaling behavior. Linear scaling observed in deployments utilizing up
1666 to 5.6 million virtual CPUs provides compelling evidence that orchestration overhead remains negligible across this range. If
1667 Python-based task management imposed significant overhead on the system, two characteristic failure patterns would manifest:

first, sublinear scaling as communication costs begin to dominate computational work, and second, performance plateaus at high core counts where task management overhead saturates available resources. Instead, the sustained linear scaling behavior demonstrates that computational time remains dominated by the actual docking calculations, which require seconds to minutes per ligand, rather than by task management operations, which require only milliseconds. This scaling characteristic provides strong empirical evidence that the hierarchical orchestration approach successfully achieves its design objective of minimizing overhead at scale. To directly validate that orchestration overhead is negligible at the single-run level (in addition to the large-scale scaling results), we benchmarked the same set of 579 ligands using Quick Vina 2 in two modes: (i) a pure serial bash loop that executes one docking at a time, and (ii) execution through the AFVS workunit runner under identical docking parameters ($exhaustiveness = 1$, $num_modes = 1$, $cpu = 1$, $seed = 42$). As summarized in Supplementary Table 22, the per-ligand docking times are on the order of a few seconds in both cases, while the AFVS Python overhead remains at the millisecond level (~ 5 ms per docking on average), contributing a negligible fraction of the total per-ligand runtime. Moreover, AFVS reduces the end-to-end elapsed time (wall-clock span) by executing many independent dockings concurrently, without altering docking outcomes, reinforcing that overall runtime is dominated by docking computation rather than workflow management.

J.2 File Usage Analysis for One Million Molecules. The hierarchical collection-based architecture described in the previous response was designed not only to minimize computational overhead but also to explicitly address file/inode limitations the reviewer correctly notes that are characteristic of traditional HPC filesystems. We also note that inode constraints are specifically relevant for classical POSIX filesystems; when deploying AdaptiveFlow on cloud infrastructure utilizing AWS S3, these limitations do not apply as object storage architectures do not impose inode quotas. For deployments on traditional HPC filesystems, the collection strategy provides substantial mitigation of inode consumption. Rather than generating individual files for each ligand molecule, an approach that would rapidly exhaust available inodes during large-scale screening campaigns, AdaptiveFlow organizes both input data and output results into aggregated file structures. The input data structure consists of approximately 1,000 compressed collection files, where each collection contains roughly 1,000 ligand molecules. We can also assume one workunit input configuration file (assume 1 collection per subjob, 1000 subjobs to a workunit). On the output side, results are aggregated at the subjob level by the Runner rather than at the individual ligand level, generating approximately 1,000 result files (in Parquet or CSV format containing structured docking results) plus approximately 1,000 log archives for diagnostic purposes. This architecture yields a total file count of approximately 3,001 files per million molecules. The collection and subjob-based strategy serves multiple complementary objectives within the system architecture: it amortizes I/O costs across multiple docking operations as discussed previously, reduces network transfer overhead through compressed archives, and maintains compatibility with HPC inode quotas that would otherwise constrain large-scale screening operations. For cloud-based deployments, AdaptiveFlow provides native support for S3-compatible object storage. While the inode limitations are not there, the same aggregated approach is still used to reduce the overheads and costs that would be associated with per-docking objects.

Cloud Scalability Benchmark. The benchmark was defined as a real-world stress test involving the preparation and docking of the 69 billion compound Enamine REAL Space library. The primary metric was scalability efficiency, defined as the ability to maintain constant processing throughput per core as the total number of concurrent vCPUs increased.

Characterization. The 5.6 million vCPU run was characterized by the simultaneous utilization of Spot Instances (Intel Xeon Scalable processors) across multiple AWS regions, orchestrated via AWS Batch. This scale refers to concurrent virtual CPUs executing the workflow in parallel.

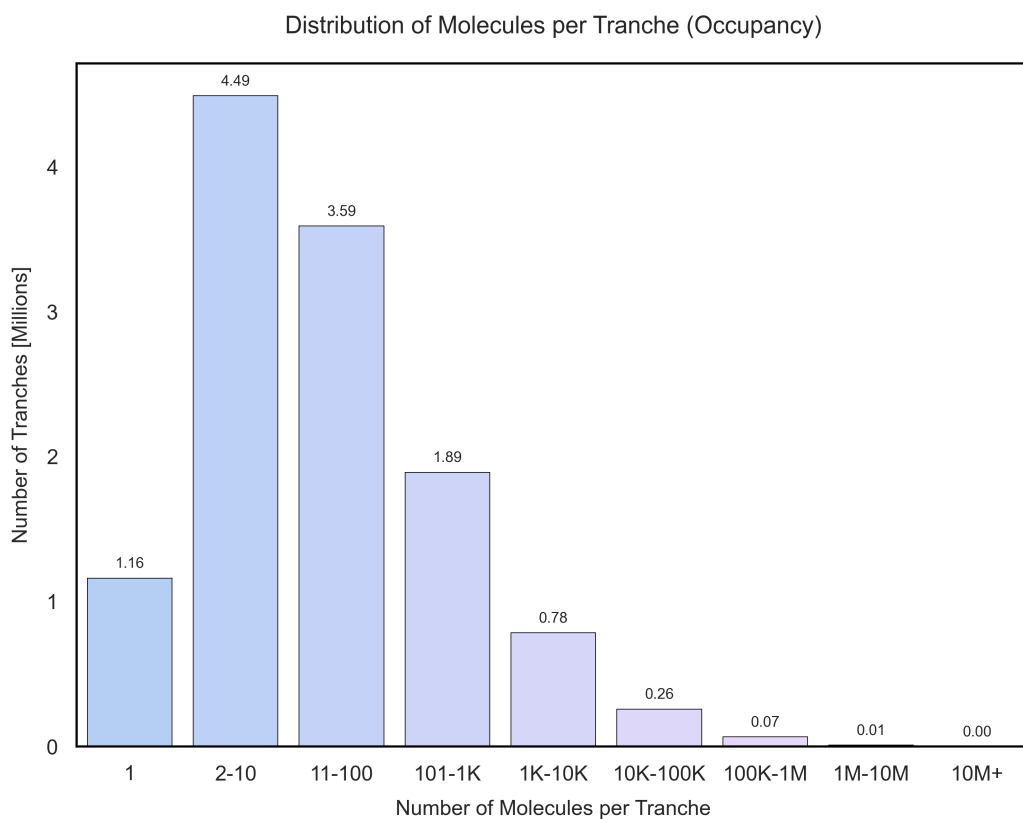
Overhead Management. Overhead was minimized via a hierarchical batching architecture. Ligands were grouped into "collections" ($\sim 1,000$ molecules) and further packed into "subjobs" (atomic units of work processing multiple collections). This ensured that the compute duration for each task (approx. 45–90 minutes) significantly exceeded the millisecond-scale scheduling overhead.

Scaling Behavior. The system demonstrated near-linear weak scaling, where throughput (molecules processed per minute) increased linearly with the number of vCPUs up to the 5.6 million limit. This effectively translates to linear strong scaling for the fixed, ultra-large workload (69 billion compounds), enabling the total run time to be compressed from weeks to hours.

We are not aware of any other publicly known examples that show scaling to this level (5.6 million concurrent vCPUs). e.g. other publicly known large scale benchmarks include [Clemson public safety video processing](#) at 2.1M vCPUs and 3.1M vCPUs on AWS using [Yellowdog](#).

Timing metric	QVina02 via AFVS (s)	QVina02 serial (s)
Records analyzed (count)	579	579
Total docking time	2311.86	2366.44
Average docking time	3.99	4.09
Wall-clock span (min-max)	145.86	2394.00
Total Python overhead	3.11	0.00
Average Python/bash overhead	0.01	0.00
Total time incl. overhead	2314.96	2366.44
Average time incl. overhead	4.00	4.09

Supplementary Table 22. Benchmark timing comparison of Quick Vina 2 docking via AFVS vs. a serial Quick Vina 2 loop. Per-ligand docking durations were summed to compute total docking time, and mean values report average per-ligand time. The wall-clock span corresponds to the elapsed time between the earliest docking start and the latest docking end. For the serial run, Python/bash overhead is set to zero because AFVS orchestration is not used. Quick Vina 2 settings: *exhaustiveness* = 1, *num_modes* = 1, *cpu* = 1, *seed* = 42.



Supplementary Fig. 37. Distribution of Compound Occupancy Across Tranches in the Enamine REAL Space. Histogram displaying the number of tranches (y-axis) populated by a given number of molecules (x-axis) within the 18-dimensional property grid. The distribution spans from single-molecule tranches to highly populated tranches containing over 10 million compounds.

1716 K Configuration File of AFLP

1717 All options of AFLP are specified in a central configuration file. A sample configuration file is shown in Supplementary Listing
1718 1.

Supplementary Listing 1. AFLP Example Configuration File.

```
1719 # Each line that starts with '#' is a comment/description of the variable above the comment
1720 # Each line that starts with '*' belongs to a section heading
1721
1722 ***** Job Resource Configuration
1723
1724 job_name=testing
1725 # alphabetic characters (i.e. letters from a-z or A-Z)
1726 # Used to describe distinct runs (using the same name will
1727 # overwrite data if using S3!)
1728
1729 threads_to_use=384
1730 # This sets how many processes the main execution loop should be using
1731 # to process. This is generally 2x the number of vCPUs or hyperthreads
1732 # available on the system it is being run on
1733
1734
1735 *****
1736 ** Batch system configuration
1737 *****
1738
1739 batchsystem=slurm
1740 # Possible values: awsbatch, slurm
1741
1742
1743 ***** AWS Batch Options (if batchsystem=awsbatch)
1744
1745 ### To use AWS Batch you must first complete the steps outlined
1746 ### in the user guide for AWS Batch
1747
1748 aws_batch_prefix=af
1749 # Prefix for the name of the AWS Batch queues. This is normally 'af'
1750 # if you used the CloudFormation template
1751
1752 aws_batch_number_of_queues=2
1753 # Should be set to the number of queues that are setup for AWS Batch.
1754 # Generally this number is 2 unless you have a large-scale (100K+ vCPUs)
1755 # setup
1756
1757 aws_batch_jobdef=af-jobdef-AFLP
1758 # Generally this is [aws_batch_prefix]-jobdef-AFLP
1759 # (e.g. if aws_batch_prefix=af, then aws_batch_jobdef=af-jobdef-AFLP
1760
1761 aws_batch_array_job_size=2
1762 # Target for the number of jobs that should be in a single array job for AWS Batch.
1763
1764 aws_ecr_repository_name=af-AFLP-ecr
1765 # Set it to the name of the Elastic Container Registry (ECR)
1766 # repository (e.g. af-AFLP-ecr) in your AWS account
1767 # (If you used the template it is generally af-AFLP-ecr)
1768
1769 aws_region=us-east-1
1770 # Set to the AWS location code where you are running AWS Batch
1771 # (e.g. us-east-1 for North America, Northern Virginia)
1772
1773 aws_batch_subjob_vcpus=8
1774 # Set to the number of vCPUs that should be launched per subjob.
1775 # 'threads_to_use' above should be >= to this value.
1776
1777 aws_batch_subjob_memory=15000
1778 # Memory per subjob to setup for the container in MB
1779
1780 aws_batch_subjob_timeout=10800
1781 # Maximum amount of time (in seconds) that a single AWS Batch job should
1782 # ever run before being terminated.
1783
1784 ***** Slurm Options (if batchsystem=slurm)
1785
1786 slurm_template=./templates/template1.slurm.sh
1787 # Template for the slurm job
1788 # Additional slurm attributes can be added directly to this
1789 # template file if they are not available as pass throughs from
1790 # AFVS
1791
1792 slurm_array_job_throttle=1
1793 # Maximum number of jobs running within a single slurm array job
1794
1795 slurm_partition=standard96
1796 # Partition to submit the job
1797
1798
```



```

1799 slurm_cpus=192
1800 # Number of CPUs that are being used
1801
1802 slurm_array_job_size=1
1803 # Maximum number of concurrent jobs from a single array job
1804 # that should be run
1805
1806
1807 *****
1808 ** Storage configuration
1809 *****
1810
1811 job_storage_mode=sharedfs
1812 # This mode determines where data is retrieved and stored from as part of
1813 # AFVS. Valid modes:
1814 # * s3: Job data is stored on S3 object store, which is the required
1815 # mode if using AWS Batch. Items under the "S3 Object Store"
1816 # heading in the configuration are required if this mode is used
1817 # * sharedfs: This mode requires that all running jobs have access to the
1818 # same shared filesystem that will allow for both input and output
1819 # of data. This is required if using Slurm
1820
1821 job_storage_output_addressing=standard
1822 # If input is placed with the hash addressing mode, then use 'hash'.
1823 # otherwise use "standard" for the classic addressing mode
1824 # 'hash' will place data in:
1825 # [0-f][0-f]/[0-f][0-f]/complete/<output_type>/<metatranche>/<tranche>/<collection>
1826 # 'standard' will place data in:
1827 # complete/<output_type>/<metatranche>/<tranche>/<collection>
1828
1829
1830 ***** Object Store Settings (S3)
1831
1832 object_store_input_ligand_library_bucket=adaptiveflow-data
1833 # Bucket name for the collection data
1834 # Trailing slashes should be used
1835
1836 object_store_input_ligand_library_prefix=AFLP/data/Enamine_2021_test
1837 # Prefix used within the object store to address the collections
1838 # Trailing slashes should be used
1839
1840 object_store_job_output_data_bucket=adaptiveflow-data
1841 # Bucket name for the job data
1842 # Trailing slashes should be used
1843
1844 object_store_job_output_data_prefix=jobs/AFLP
1845 # Where to place job-specific data. This includes where AdaptiveFlow will place
1846 # the input data needed for jobs as well as the output files. Data be be placed
1847 # in object_store_job_output_data_prefix/<job_letter>
1848 # Trailing slashes should be used
1849
1850 ***** Shared Filesystem Settings
1851
1852 collection_folder=./input-files/collections
1853 # Path to where the collection file are stored
1854 # * This is used when job_storage_mode=sharedfs or
1855 # when the uploader helper script is being used
1856 #
1857 # Slash at the end is not required (optional)
1858 # Either pathname is required w.r.t. the folder tools/
1859 # or absolute path (e.g. /home/afuser/collections)
1860
1861 # Output of processed ligands will be placed in
1862 # ../workflow/complete (where ../workflow is relative to the tools
1863 # directory of the AFVS installation
1864
1865 ***** Workflow Options
1866
1867 ligands_todo_per_queue=1000
1868 # Used as a limit of ligands for the to-do lists.
1869 # A reasonable number for this is generally 1000. The length of time
1870 # to process will depend on the docking scenarios run
1871
1872 tempdir_default=/dev/shm
1873 # The directory which is used for the temporary workflow files which need a normal performance
1874 # Is normally a local SSD or HDD. A temporary directory will be created underneath this dir
1875
1876 tempdir_fast=/dev/shm
1877 # The directory which is used for the temporary workflow files which need a fast performance
1878 # Should be a a local ram filesystem/ramdisk. A temporary directory will be created underneath this dir
1879
1880 store_all_intermediate_logs=false
1881 # This determines if the intermediate files (all intermediate steps) are
1882 # retained for review after a run. This is not recommended unless you
1883 # are debugging as it increases output by over 2x
1884 # Valid values:

```

```

1885 # * true: Keep the logs. They will be placed in intermediate/
1886 # * false: Do not retain the logs
1887
1888
1889 *****
1890 ** Ligand Preparation Options
1891 *****
1892
1893
1894 ***** Desalting
1895
1896 desalting=true
1897 # If true, extracts the largest organic part of the molecule
1898 # Possible values:
1899 # * false
1900 # * true
1901
1902 desalting_obligatory=false
1903 # Setting only required if desalting=true
1904 # Possible values:
1905 # * true: Successful desalting of the ligand is mandatory (unsuccessful desalting leads to the omission of this ligand
1906 # ).
1907 # * false: The ligand will continue to be processed even if the desalting step fails.
1908
1909 ***** Neutralization
1910
1911 neutralization=true
1912 # Neutralizes molecules using JChem's Standardizer of ChemAxon
1913 # Possible values:
1914 # * false
1915 # * true
1916
1917 neutralization_program_1=obabel
1918 neutralization_program_2=none
1919 # Program 1 is used at first for each ligand, and if it fails program 2 is used instead. If the second program also
1920 # fails, then the ligand is skipped.
1921 # Setting only required if neutralization=true
1922 # Possible values:
1923 # * obabel
1924 # * standardizer
1925 # * none (only for neutralization_program_2)
1926
1927 chemaxon_neutralization_timeout=15
1928 # Time in seconds to wait to timeout for neutralization with chemaxon
1929
1930 obabel_neutralization_timeout=15
1931 # Time in seconds to wait to timeout for neutralization with obabel
1932
1933 neutralization_mode=only_genuine_desalting_and_if_charged
1934 ###neutralization_mode=after_desalting_if_charged
1935 # Only relevant if neutralization=true
1936 # Possible values:
1937 # * always: always neutralize the molecule
1938 # * only_genuine_desalting: only neutralize the molecule if the input structure
1939 # contained more than one component
1940 # * only_genuine_desalting_and_if_charged: only neutralize if the input structure
1941 # contained more than one component and if the smallest component contained an ion
1942
1943
1944 neutralization_obligatory=false
1945 # Setting only required if neutralization=true
1946 # Possible values:
1947 # * true: Successful neutralization of the ligand is mandatory (unsuccessful neutralization leads to the omission of
1948 # this ligand).
1949 # * false: The ligand will continue to be processed even if the neutralization step fails.
1950 # Settable via range control files: Yes
1951
1952 ***** Stereoisomer Generation
1953
1954 stereoisomer_generation=false
1955 # Possible values:
1956 # * false
1957 # * true
1958
1959 stereoisomer_generation_program_1=rdkit
1960 stereoisomer_generation_program_2=none
1961 # Program 1 is used at first for each ligand, and if it fails program 2 is used instead. If the second program also
1962 # fails, then the ligand is skipped.
1963 # Setting only required if stereoisomer_generation=true
1964 # Possible values:
1965 # * rdkit
1966 # * cxcalc
1967
1968 stereoisomer_obligatory=false
1969 # Setting only required if stereoisomer_generation=true
1970 # Possible values:

```

```

1971 # * true: Successful stereoisomer generation of the ligand is mandatory
1972 # (unsuccessful stereoisomer generation leads to the omission of this ligand).
1973 # * false: The ligand will continue to be processed even if the stereoisomer generation
1974 # step fails.
1975
1976 cxcalc_stereoisomer_generation_options=-T -f smiles
1977 # any options which should be passed to the stereoisomer plugin of cxcalc
1978 # This setting has to contain the "-f smiles" option
1979 # The "-T" option for instance specifies that that chiral carbons should be preserved.
1980 # Documentation on the options is available here:
1981 # https://docs.chemaxon.com/display/docs/cxcalc-calculator-functions.md#src-1806682-cxcalccalculatorfunctions-
1982 # stereoisomers
1983
1984 rdkit_stereoisomer_generation_unique_correction=false
1985 # Enumeration of stereoisomers by RDKit can result in redundant stereoisomers
1986 # This can be resolved by setting this option to true. However, can be time consuming
1987
1988 cxcalc_stereoisomer_timeout=15
1989 # Time in seconds to wait to timeout for cxcalc
1990
1991 obabel_stereoisomer_timeout=30
1992 # Time in seconds to wait to timeout for obabel
1993
1994 ***** Tautomerization
1995
1996 tautomerization=true
1997 # Possible values:
1998 # * false
1999 # * true
2000
2001 tautomerization_program_1=obabel
2002 tautomerization_program_2=none
2003 # Program 1 is used at first for each ligand, and if it fails program 2 is used instead. If the second program also
2004 # fails, then the ligand is skipped.
2005 # Setting only required if tautomerization=true
2006 # Possible values:
2007 # * obabel
2008 # * cxcalc
2009 # * none (only for tautomerization_program_2)
2010
2011 tautomerization_obligatory=false
2012 # Setting only required if tautomerization=true
2013 # Possible values:
2014 # * true: Successful tautomerization of the ligand is mandatory (unsuccessful tautomerization leads to the omission of
2015 # this ligand).
2016 # * false: The ligand will continue to be processed even if the tautomerization step fails.
2017
2018 cxcalc_tautomerization_options=
2019 # any options which should be passed to the tautomerization plugin of cxcalc
2020
2021 cxcalc_tautomerization_timeout=15
2022 # Time in seconds to wait to timeout for tautomerization with cxcalc
2023
2024 obabel_tautomerization_timeout=30
2025 # Time in seconds to wait to timeout for tautomerization with obabel
2026
2027 ***** Protonation State Generation
2028
2029 protonation_state_generation=true
2030 # Possible values:
2031 # * false
2032 # * true
2033
2034 protonation_program_1=obabel
2035 protonation_program_2=none
2036 # Program 1 is used at first for each ligand, and if it fails program 2 is used instead. If the second program also
2037 # fails, then the ligand is skipped.
2038 # Setting only required if protonation_state_generation=true
2039 # Possible values:
2040 # * obabel
2041 # * cxcalc
2042 # * none (only for protonation_program_2)
2043
2044 cxcalc_protonation_timeout=15
2045 # Time in seconds to wait to timeout
2046
2047 obabel_protonation_timeout=15
2048 # Time in seconds to wait to timeout
2049
2050 protonation_obligatory=false
2051 # Setting only required if protonation_state_generation=true
2052 # Possible values:
2053 # * true: Successful protonation of the ligand is mandatory (unsuccessful
2054 # protonation leads to the omission of this ligand).
2055 # * false: The ligand will continue to be processed even if the protonation step fails.
2056 # This might light to protonation states which are unphysiological.

```

```

2057
2058 protonation_pH_value=7.4
2059 # Setting only required if protonation_state_generation=true
2060 # Possible values: floating point number between 0.0 and 14.0
2061
2062
2063 ***** Conformation Generation
2064
2065 conformation_generation=true
2066 # Generation of 3D conformation/coordinates of the ligand
2067 # Possible values:
2068 #   * false
2069 #   * true
2070
2071 conformation_program_1=obabel
2072 conformation_program_2=none
2073 # Setting only required if conformation_generation=true
2074 # Program 1 is used at first for each ligand, and if it fails program 2 is used instead.
2075 # If the second program also fails, then the ligand is skipped.
2076 #
2077 # Possible values:
2078 #   * obabel
2079 #   * molconvert
2080 #   * none (only possible for conformation_program_2)
2081
2082 molconvert_3D_options=-3:{fine}
2083 # Setting only required if conformation_generation=true and one of the programs used is molconvert
2084 # 3D conformation generation options for molconvert.
2085 # See also the help text printed by molconvert for additional information
2086 # Possible values:
2087 #   * -3           Defaults to value 3{fast}
2088 #   * 3{fine}      Find low energy conformer Leave failed fragments intact
2089 #   * 3{fast}      Fast clean, if failed, perform fine clean, accept any generated structure (default)
2090 #   * 3{nofaulty} Same as S{fast}, but leave failed fragments intact.
2091
2092 molconvert_conformation_timeout=15
2093 # Time in seconds to wait to timeout
2094
2095 obabel_conformation_timeout=15
2096 # Time in seconds to wait to timeout
2097
2098 conformation_obligatory=true
2099 # Setting only required if conformation_generation=true
2100 # Possible values:
2101 #   * true: Successful 3D conformation generation of the ligand is mandatory
2102 #           (unsuccessful conformation generation leads to the omission of this ligand).
2103 #   * false: The ligand will continue to be processed even if the conformation
2104 #            generation step fails.
2105
2106
2107 ***** Target Format Generation
2108
2109 targetformats=pdb:pdibt:sdf:mol2:smi:selfies
2110 # Possible values:
2111 #   * selfies (https://github.com/aspuru-guzik-group/selfies)
2112 #   * Any format supported by the Open Babel, using the file format identifiers used by Open Babel.
2113 #   * A complete list can be obtained by running the command "obabel -L formats"
2114 # Multiple target formats can be specified by separating them with colons, e.g. pdb:sdf:pdibt
2115
2116
2117 ***** Open Babel
2118
2119 obabel_memory_limit=1000000
2120 # In KB
2121 # Recommended value: >= 500000
2122
2123
2124 ***** Energy Check
2125
2126 energy_check=true
2127 # Determines whether the potential energy is checked by obenergy (Open Babel Energy). This can be useful to filter out
2128 # compounds with unrealistic predicted 3D geometry
2129 # Possible values:
2130 #   * true
2131 #   * false
2132
2133 energy_max=10000
2134 # Maximum allowed energy value. Recommended: 10000
2135 # Possible values: Positive integer
2136
2137
2138 ***** Tranche Assignments
2139
2140 tranche_assignments=true
2141 # Should each ligand be assigned a new tranche based based on molecular properties of the ligand?
2142 # This is useful in particular if the tranches of the input ligands are pseudo-tranches (i.e. are arbitrary or have no

```

```

2143     meaning).
2144
2145     tranche_types=hbd_obabel
2146     # These variables are only needed if tranche_assignments=true
2147     # Multiple values are separated by colons
2148     # Possible values:
2149     # * mw_jchem: molecular weight
2150     # * mw_obabel: molecular weight
2151     # * logp_jchem: octanol water partition coefficient by JChem's cxcalc
2152     # * logp_obabel: octanol water partition coefficient by Open Babel
2153     # * hba_jchem: hydrogen bond acceptor count
2154     # * hba_obabel: hydrogen bond acceptor count
2155     # * hbd_jchem: hydrogen bond donor count
2156     # * hbd_obabel: hydrogen bond donor count
2157     # * rotb_jchem: rotatable bond count
2158     # * tpsa_jchem: topological polar surface area by JChem's cxcalc
2159     # * tpsa_obabel: topological polar surface area by Open Babel
2160     # * logd: octanol water partition coefficient
2161     # * logs: aqueous solubility in mol/L
2162     # * atomcount_jchem: number of atoms (including hydrogen)
2163     # * atomcount_obabel: number of atoms (including hydrogen)
2164     # * bondcount_jchem: number of bonds (including hydrogen-heavy atom bonds)
2165     # * bondcount_obabel: number of bonds (including hydrogen-heavy atom bonds)
2166     # * ringcount: number of non-aromatic rings
2167     # * aromaticringcount: number of aromatic rings
2168     # * mr_jchem: molecular refractivity by JChem's cxcalc
2169     # * mr_obabel: molecular refractivity by Open Babel
2170     # * formalcharge: total (formal) charge of the molecule
2171     # * positivechargecount: number of positive charged functional groups/atoms
2172     # * negativechargecount: number of negatively charged functional groups/atoms
2173     # * fsp3: fraction of sp3 hybridized carbon atoms
2174     # * chiralcentercount: number of chiral centers
2175     # * halogencount: number of halogen atoms (F, Br, Cl, I)
2176     # * sulfurcount: number of sulfur atoms (S)
2177     # * NOcount: number of oxygen and nitrogen atoms (N, O)
2178     # * electronegativeatomcount: number of electronegative atoms (N, O, S, P, F, Br, Cl, I)
2179     # * mw_file: molecular weight
2180     # * logp_file: octanol water partition coefficient by JChem's cxcalc
2181     # * hba_file: hydrogen bond acceptor count
2182     # * hbd_file: hydrogen bond donor count
2183     # * rotb_file: rotatable bond count
2184     # * tpsa_file: topological polar surface area by JChem's cxcalc
2185     # * logd_file: octanol water partition coefficient
2186     # * logs_file: aqueous solubility in mol/L
2187     # * heavyatomcount_file: number of atoms (including hydrogen)
2188     # * bondcount_file: number of bonds (including hydrogen-heavy atom bonds)
2189     # * ringcount_file: number of non-aromatic rings
2190     # * aromaticringcount_file: number of aromatic rings
2191     # * mr_file: molecular refractivity by Open Babel
2192     # * formalcharge_file: total (formal) charge of the molecule
2193     # * positivechargecount_file: number of positive charged functional groups/atoms
2194     # * negativechargecount_file: number of negatively charged functional groups/atoms
2195     # * fsp3_file: fraction of sp3 hybridized carbon atoms
2196     # * chiralcentercount_file: number of chiral centers
2197     # * halogencount_file: number of halogen atoms (F, Br, Cl, I)
2198     # * sulfurcount_file: number of sulfur atoms (S)
2199     # * NOcount_file: number of oxygen and nitrogen atoms (N, O)
2200     # * electronegativeatomcount_file: number of electronegative atoms (N, O, S, P, F, Br, Cl, I)
2201     # * enamine_type: Enamine type category
2202     # * doublebondstereoisomercount_jchem: number of double-bond stereoisomers from JChem
2203     # * aromaticproportion_jchem: aromatic proportion generated from JChem
2204     # * qed_rdkit: QED generated by RDKit
2205
2206
2207     file_fieldnames=smi:ligand-name:hbd_obabel
2208     # For each tranche type which reads the value from a file, those values must be defined.
2209     # At a minimum 'smi' and 'ligand-name' are REQUIRED (and are generally expected to be the first
2210     # two items per line
2211     #
2212     # e.g. file_fieldnames=smi:ligand-name:-:-:-:-mw:atomcount
2213     # This means that the first column should be the SMILES string, the second
2214     # column is the 'friendly name', columns 3-7 are values not used,
2215     # column 8 is the value that should be used for mw_file in tranche assignment,
2216     # column 9 is the value that should be used for atomcount_file in tranche assignment
2217     # Other valid options include: enamine (for enamine_type), mw, heavyatomcount,
2218     # logp, hba, hbd, rotb, fsp3, tpsa
2219
2220
2221     # For each tranche_type X in the variable tranche_types, one additional variable "tranche_<X>_partition" has to be
2222     # specified.
2223     # This variable has to be a set of N >= 1 values (at least one), separated by colons, which partitions the value range
2224     # of the molecular property of tranche <X> into N+1 intervals.
2225     # For example, the partitions '-0:1:2' would result in the 4 intervals: (-∞,0], (0,1], (1,2], (2,∞). The maximum
2226     # number of allowed intervals is 27 (corresponding to 26 values of the alphabet). Each value has to be an integer or
2227     # float, and the values have to be in ascending order.
2228     # These variables are only needed if tranche_assignments=true

```

```

2229 tranche_mw_jchem_partition=200:300:400:450
2230 tranche_mw_obabel_partition=200:300:400:450
2231 tranche_mw_file_partition=200:300:400:450
2232 tranche_logp_jchem_partition=0:2:3:4:5
2233 tranche_logp_obabel_partition=0:2:3:4:5
2234 tranche_logp_file_partition=0:2:3:4:5
2235 tranche_hba_jchem_partition=2:4:7:10
2236 tranche_hba_obabel_partition=2:4:7:10
2237 tranche_hba_file_partition=2:4:7:10
2238 tranche_hbd_jchem_partition=0:2:4:5
2239 tranche_hbd_obabel_partition=0:2:4:5
2240 tranche_hbd_file_partition=0:2:4:5
2241 tranche_rotb_jchem_partition=0:2:4:7:10
2242 tranche_rotb_obabel_partition=0:2:4:7:10
2243 tranche_tpsa_jchem_partition=40:80:110:140
2244 tranche_tpsa_obabel_partition=40:80:110:140
2245 tranche_tpsa_file_partition=40:80:110:140
2246 tranche_logd_partition=0:2:3:4:5
2247 tranche_logd_file_partition=-1:0:1:2:2.5:3:3.5:4:4.5:5
2248 tranche_logs_partition=-6:-5:-4:-3
2249 tranche_logs_file_partition=-6:-5:-4:-3
2250 tranche_atomcount_jchem_partition=10:20:30:40:50
2251 tranche_atomcount_obabel_partition=10:20:30:40:50
2252 tranche_heavyatomcount_file_partition=10:20:30:40:50
2253 tranche_bondcount_jchem_partition=10:20:30:40:50
2254 tranche_bondcount_obabel_partition=10:20:30:40:50
2255 tranche_bondcount_file_partition=10:20:30:40:50
2256 tranche_ringcount_partition=0:1:2:3:4:5
2257 tranche_ringcount_file_partition=0:1:2:3:4:5
2258 tranche_aromaticringcount_partition=0:1:2:3:4:5
2259 tranche_aromaticringcount_file_partition=0:1:2:3:4:5
2260 tranche_formalcharge_partition=-2:-1:0:1
2261 tranche_formalcharge_file_partition=-2:-1:0:1
2262 tranche_mr_jchem_partition=40:80:130
2263 tranche_mr_obabel_partition=40:80:130
2264 tranche_mr_file_partition=40:80:130
2265 tranche_positivechargecount_partition=0:1
2266 tranche_positivechargecount_file_partition=-2:-1:0:1:2
2267 tranche_negativechargecount_partition=0:1
2268 tranche_negativechargecount_file_partition=-2:-1:0:1:2
2269 tranche_fsp3_partition=0.2:0.4:0.6:0.8
2270 tranche_fsp3_file_partition=0.2:0.4:0.6:0.8
2271 tranche_chiralcentercount_partition=0:1
2272 tranche_chiralcentercount_file_partition=0:1:2:3:4:5
2273 tranche_halogencount_partition=0
2274 tranche_halogencount_file_partition=0:1:2:3:4:5
2275 tranche_sulfurcount_partition=0
2276 tranche_sulfurcount_file_partition=0:1:2:3:4:5
2277 tranche_NOcount_partition=0:1:2:3:4:5:6:7:8:9:10
2278 tranche_NOcount_file_partition=0:1:2:3:4:5:6:7:8:9:10
2279 tranche_electronegativeatomcount_partition=0:1:2:3:4:5:6:7:8:9:10
2280 tranche_electronegativeatomcount_file_partition=0:1:2:3:4:5:6:7:8:9:10
2281 tranche_rotb_file_partition=0:2:4:7:10
2282 tranche_aromaticproportion_jchem_partition=0:0.25:0.5:0.75
2283 tranche_doublebondstereoisomercount_jchem_partition=0
2284 tranche_qed_rdkit_partition=0.2:0.4:0.6:0.8:1.0
2285
2286 # For tranche types that are characters or strings (non-numeric)
2287 # they should be in the format tranche_(tranche_type)_mapping=M:A,C:B
2288 # -- the above sample shows that a string value of "M" should
2289 #    map to tranche of A, "C" to a tranche of B
2290
2291 tranche_enamine_type_mapping=M:A,S:B
2292
2293
2294 ***** Attributes
2295
2296 # In addition to the tranche_types above, attributes can be calculated and placed by AFLP
2297 # as remarks in pdb, pdbqt, and mol2 files.
2298
2299 attributes_to_generate=scaffold_rdkit
2300 # Multiple values are separated by colons
2301 # Valid values:
2302 #   * (any of the tranche_types above). e.g. mr_jchem
2303 #   * scaffold_rdkit: Canonical SMI of Murcko scaffold
2304
2305
2306 ***** JChem-Related Packages
2307
2308 use_cxcalc_helper=0
2309 # If using AWS Batch, then set this to 1. If not, then set to 0 currently. This option
2310 # provides an optimization to improve performance of the chemaxon properties
2311
2312 jchem_package_filename=None
2313 # Required only if cxcalc or molconvert (both of ChemAxon) are used in the preparation steps, and if tranche_assignments
2314 =true

```

```

2315 # The filename of the JChem package located in the folder tools/packages/ in the tar.gz format (available on the
2316 # ChemAxon homepage)
2317 # The root folder in the archive has to have the name jchemsuite (normally distributed by ChemAxon in this way)
2318 # Possible values:
2319 # * <filename>
2320 # * none
2321
2322 chemaxon_license_filename=none
2323 # Required only if cxcalc or molconvert (both of ChemAxon) are used in the preparation steps.
2324 # The filename of the license file, which has to be located in the folder tools/packages/
2325 # Possible values:
2326 # * <filename>
2327 # * none
2328
2329 java_package_filename=none
2330 # Required only if cxcalc or molconvert (both of ChemAxon) are used in the preparation steps.
2331 # Any JRE binary distribution of version of at least version 8.
2332 # If java is provided by the system (e.g. by loading a module), then no Java package needs to be provided.
2333 # This has to be a file in the tar.gz format, which has to be located in the folder tools/packages/
2334 # The root folder in the archive has to have the name "java" (which will be used for the JAVA_HOME variable). This
2335 # normally needs to be manually changed in the Java package after downloading a JRE.
2336 # Possible values:
2337 # * <filename>
2338 # * none
2339
2340 ng_package_filename=none
2341 # Required only if cxcalc or molconvert (both of ChemAxon) are used in the preparation steps.
2342 # Nailgun package filename.
2343 # This has to be a file in the tar.gz format, which has to be located in the folder tools/packages/
2344 # The root folder in the archive has to have the name "nailgun"
2345 # Possible values:
2346 # * <filename>
2347 # * none
2348
2349 java_max_heap_size=4
2350 # Size in GB
2351 # Recommended: >= 1 GB * queues_per_step
2352 # The required memory depends mainly on how many queues are run per step (and thus per JVM/NG server), since one JVM is
2353 # used per step

```

2355 L Configuration File of AFVS

2356 All options of AFVS are specified in a central configuration file. A sample configuration file is shown in Supplementary Listing
2357 2.

Supplementary Listing 2. AFVS Example Configuration File.

```
2358 # Each line that starts with '#' is a comment/description of the variable above the comment
2359 # Each line that starts with '*' belongs to a section heading
2360
2361 ***** Job Resource Configuration
2362
2363 job_name=testing
2364 # alphabetic characters (i.e. letters from a-z or A-Z)
2365 # Used to describe distinct runs (using the same name will
2366 # overwrite data if using S3!)
2367
2368 threads_per_docking=1
2369 # How many threads should be used for each docking program.
2370
2371 threads_to_use=8
2372 # This sets how many processes the main execution loop should be using
2373 # to process. This is generally 2x the number of vCPUs or hyperthreads
2374 # available on the system it is being run on
2375
2376 program_timeout=90
2377 # How many seconds to wait for each ligand to be processed by a program
2378
2379 ***** Batch system configuration
2380
2381 ** Batch system configuration
2382
2383 *****
2384
2385 batchsystem=awsbatch
2386 # Possible values: awsbatch, slurm
2387
2388 ***** AWS Batch Options (if batchsystem=awsbatch)
2389
2390 ### To use AWS Batch you must first complete the steps outlined
2391 ### in the user guide for AWS Batch
2392
2393 aws_batch_prefix=af
2394 # Prefix for the name of the AWS Batch queues. This is normally 'af'
2395 # if you used the CloudFormation template
2396
2397 aws_batch_number_of_queues=2
2398 # Should be set to the number of queues that are setup for AWS Batch.
2399 # Generally this number is 2 unless you have a large-scale (100K+ vCPUs)
2400 # setup
2401
2402 aws_batch_jobdef=af-jobdef-AFVS
2403 # Generally this is [aws_batch_prefix]-jobdef-AFVS
2404 # (e.g. if aws_batch_prefix=af, then aws_batch_jobdef=af-jobdef-AFVS)
2405
2406 aws_batch_array_job_size=200
2407 # Target for the number of jobs that should be in a single array job for AWS Batch.
2408
2409 aws_ecr_repository_name=af-AFVS-ecr
2410 # Set it to the name of the Elastic Container Registry (ECR)
2411 # repository (e.g. af-AFVS-ecr) in your AWS account
2412 # (If you used the template it is generally af-AFVS-ecr)
2413
2414 aws_region=us-east-1
2415 # Set to the AWS location code where you are running AWS Batch
2416 # (e.g. us-east-1 for North America, Northern Virginia)
2417
2418 aws_batch_subjob_vcpus=8
2419 # Set to the number of vCPUs that should be launched per subjob.
2420 # 'threads_to_use' above should be >= to this value.
2421
2422 aws_batch_subjob_memory=15000
2423 # Memory per subjob to setup for the container in MB
2424
2425 aws_batch_subjob_timeout=10800
2426 # Maximum amount of time (in seconds) that a single AWS Batch job should
2427 # ever run before being terminated.
2428
2429 ***** Slurm Options (if batchsystem=slurm)
2430
2431 slurm_template=./templates/template1.slurm.sh
2432 # Template for the slurm job
2433 # Additional slurm attributes can be added directly to this
2434 # template file if they are not available as pass throughs from
2435 # AFVS
2436
2437 slurm_array_job_throttle=100
```



```

2438 # Maximum number of jobs running within a single slurm array job
2439
2440 slurm_partition=partition
2441 # Partition to submit the job
2442
2443 slurm_cpus=18
2444 # Number of CPUs that are being used
2445
2446 slurm_array_job_size=100
2447 # Maximum number of concurrent jobs from a single array job
2448 # that should be run
2449
2450
2451 *****
2452 ** Storage configuration
2453 *****
2454
2455 job_storage_mode=s3
2456 # This mode determines where data is retrieved and stored from as part of
2457 # AFVS. Valid modes:
2458 # * s3: Job data is stored on S3 object store, which is the required
2459 #       mode if using AWS Batch. Items under the "S3 Object Store"
2460 #       heading in the configuration are required if this mode is used
2461 # * sharedfs: This mode requires that all running jobs have access to the
2462 #             same shared filesystem that will allow for both input and output
2463 #             of data. This is required if using Slurm
2464
2465
2466 data_collection_addressing_mode=metatranche
2467 # If input is placed with the hash addressing mode, then use 'hash'.
2468 # otherwise use "metatranche" for the classic addressing mode
2469
2470 data_collection_identifer=
2471 # This is only used if object_store_data_collection_addressing_mode=hash
2472 # Generally this is the dataset name (e.g. Enamine_REAL_Space_2021q12)
2473
2474 job_addressing_mode=metatranche
2475 # If job output is to be placed with the hash addressing mode, then use 'hash'.
2476 # otherwise use "metatranche" for the classic addressing mode
2477
2478
2479 ***** Object Store Settings (S3)
2480
2481 object_store_job_bucket=
2482 # Bucket name for the job data (output and job files)
2483
2484 object_store_job_prefix=jobs
2485 # Where to place job-specific data. This includes where AdaptiveFlow will place
2486 # the input data needed for jobs as well as the output files.
2487 #
2488 # Data be be placed:
2489 # if object_store_job_addressing_mode=hash
2490 #     in object_store_job_prefix/XX/YY/<job_letter>
2491 #     (where 'XX', and 'YY' are hash values that will vary for
2492 #     different files)
2493 # else
2494 #     in object_store_job_prefix/<job_letter>
2495
2496 object_store_data_bucket=
2497 # Bucket name for the input collection data (often the same as the job one)
2498
2499 object_store_data_collection_prefix=
2500 # Prefix used within the object store to address the collections
2501
2502
2503
2504
2505 ***** Shared Filesystem Settings
2506
2507 collection_folder=/home/ec2-user/collections
2508 # Path to where the collection file are stored
2509 # * This is used when job_storage_mode=sharedfs or
2510 #   when the uploader helper script is being used
2511 #
2512 # Slash at the end is not required (optional)
2513 # Either pathname is required w.r.t. the folder tools/
2514 #     or absolute path (e.g. /home/afuser/collections)
2515
2516
2517 *****
2518 ** Run configuration
2519 *****
2520
2521 ***** Output information
2522
2523 summary_formats=parquet

```

```

2524 # Format for summary files that are generated with the score data.
2525 # Supported values:
2526 # * txt.gz (space delimited files)
2527 # * parquet
2528 # Multiple formats can be generated by placing a comma
2529 # (e.g. summary_formats=parquet,txt.gz)
2530
2531 print_smi_in_summary=1
2532 # Whether or not the SMILES string should be printed in
2533 # the summary file (works for pdbqt and mol2)
2534 # Supported values:
2535 # * 0 : Do not provide
2536 # * 1 : SMILES string in the summary file
2537
2538
2539 ***** Workflow Options
2540
2541 ligands_todo_per_queue=1000
2542 # Used as a limit of ligands for the to-do lists.
2543 # A reasonable number for this is generally 1000. The length of time
2544 # to process will depend on the docking scenarios run
2545
2546 ligand_library_format=pdbqt
2547 # Supported values:
2548 # * pdbqt
2549 # * mol2
2550 # This value is case sensitive
2551 # All AutoDock based docking programs require the library to
2552 # be in the pdbqt format.
2553 # When the docking program PLANTS is used, both libraries in the
2554 # pdbqt and the mol2 format are supported.
2555
2556 tmpdir_default=/dev/shm
2557 # The directory which is used for the temporary workflow files which need a normal performance
2558 # Is normally a local SSD or HDD. A temporary directory will be created underneath this dir
2559
2560
2561 tmpdir_fast=/dev/shm
2562 # The directory which is used for the temporary workflow files which need a fast performance
2563 # Should be a a local ram filesystem/ramdisk. A temporary directory will be created underneath this dir
2564
2565 ***** Virtual Screening Options
2566
2567 docking_scenario_names=qvina02_rigid_receptor1
2568 # Names for the docking scenarios, separated by colons
2569 # Each docking scenario has one value. Multiple docking scenarios/names have
2570 # to be separated by colons ":" and without spaces
2571 #
2572 # Example: docking_scenario_names=receptor1_vina_rigid:receptor1_smina_flexible
2573 # The docking scenario names are used for the folder names in which the output files are stored
2574
2575 docking_scenario_programs=qvina02
2576 # For each docking scenario name, a docking protocol has to be specified
2577 # Values have to be separated by colons ":" and without spaces, e.g: docking_scenario_programs=vina:smina
2578 # Possible values:
2579 #   AutoDock-Koto,   AutodockVina_1.2,   AutodockZN,   EquiBind,   FRED
2580 #   FitDock,        GalaxyDock3,        LigandFit,    LightDock,   M-Dock
2581 #   MCDock,         MM-GBSA,        PLANTS,      PSOVina,    RLDock
2582 #   SEED,           adfr,          autodock_cpu, autodock_gpu, autodock_vina
2583 #   dock6,          flexx,          glide,        gnina,       gnina-scoring
2584 #   gold,           gwovina,        iGemDock,     idock,       ledock
2585 #   molegro,        nnscore2,       qvina,        qvina-w,     rDock
2586 #   rf-score,       rosetta-ligand, smina,        smina-scoring, vina
2587 #   vina_carb,      vina_xb
2588 # Please note: different pose prediction/docking protocols can be combined with scoring functions.
2589 # For example: 'qvina+nnscore2'.
2590 # For supported choices/combinations please see the AdaptiveFlow homepage.
2591
2592
2593 docking_scenario_replicas=1
2594 # Series of integers separated by colons ":"
2595 # The number of values has to equal the number of docking programs
2596 # specified in the variable "docking_programs"
2597 # The values are in the same order as the docking programs specified in the
2598 # variable "docking_scenario_programs
2599 # e.g.: docking_scenario_replicas=1:1
2600 # possible range: 1-99999 per field/docking program
2601 # The docking scenario is comprised of all the docking types and their replicas
2602
2603 docking_scenario_basefolder=../input-files
2604 # Relative path to tools directory
2605 # Base directory for where the docking scenarios are held. Nothing other
2606 # than the required files for the docking scenario should be placed here
2607
2608 docking_scenario_inputfolders=qvina02_rigid_receptor1
2609 # folder names inside 'docking_scenario_basefolder'

```

```
2610 # In each input folder must be the file config.txt which is used by the
2611 #   docking program to specify its options
2612 # If other input files are required by the docking type, usually
2613 #   specified in the config.txt file, they have to be in the same folder
```

2615 M Configuration File of AF Unity

2616 All options of AF Unity are specified in a central configuration file when used in standalone mode. A sample configuration file
2617 is shown in Supplementary Listing 3.

Supplementary Listing 3. AdaptiveFlow Unity (AFU) Example Configuration File.

```
2618 # The choice of the docking protocol
2619 # Possible choices:
2620 #   AutoDock-Koto,   AutodockVina_1.2,   AutodockZN,   EquiBind,   FRED
2621 #   FitDock,         GalaxyDock3,        LigandFit,    LightDock,  M-Dock
2622 #   MCDock,          MM-GBSA,             PLANTS,      PSOVina,    RLDock
2623 #   SEED,            adfr,             autodock_cpu, autodock_gpu, autodock_vina
2624 #   dock6,           flexx,             glide,       gnina,      gnina-scoring
2625 #   gold,            gwovina,           iGemDock,    idock,      ledock
2626 #   molegro,         nnscore2,          qvina,       qvina-w,    rDock
2627 #   rf-score,        rosetta-ligand,    smina,       smina-scoring, vina
2628 #   vina_carb,       vina_xb
2629 # Please note: different pose prediction/docking protocols can be combined with scoring functions.
2630 #               For example: 'qvina+nnscore2'.
2631 # For supported choices/combinations please see the AdaptiveFlow homepage.
2632
2633 program_choice=qvina
2634
2635
2636 # The x,y&z coordinates of the center of the docking space. The binding space describes the location where a molecule is
2637 #   allowed to bind.
2638 center_x=10
2639 center_y=10
2640 center_z=10
2641
2642 #
2643
2644 # The size (in Angstroms) of the docking space in the x,y&z directions.
2645 size_x=10
2646 size_y=10
2647 size_z=10
2648
2649 # How many poses to search for, providing a limit to the maximum number of iterations that a docking program performs in
2650 #   search for good poses
2651 # Large exhaustiveness settings lead to increased computational costs.
2652 exhaustiveness=10
2653
2654 # Molecule (either a string in smiles, selfies or amino-acid sequence)
2655 # If the is_selfies=True, or is_peptide=True, a conversion from selfies->smiles
2656 # and aa-sequence->smiles is performed.
2657 smi=CCCCCCCCCCCC
2658 is_selfies=False
2659 is_peptide=False
2660
2661 # Location to the prepared receptor file
2662 # The receptor needs to be in the correct format, supported by the user's selected docking program.
2663 # Additionally, the file needs to be present in the config directory.
2664 receptor=./config/5wiu_test.pdbqt
2665
2666
```