Journal of Molecular Graphics and Modelling 44 (2013) 104–112

Contents lists available at ScienceDirect

Journal of Molecular Graphics and Modelling

journal homepage: www.elsevier.com/locate/JMGM

AutoGrow 3.0: An improved algorithm for chemically tractable, semi-automated protein inhibitor design

Jacob D. Durrant[a,*], Steffen Lindert[b], J. Andrew McCammon[b,c,d]

[a] Department of Chemistry & Biochemistry, University of California San Diego, La Jolla, CA 92093, United States
[b] Department of Pharmacology, University of California San Diego, La Jolla, CA 92093, United States
[c] Department of Chemistry & Biochemistry, NSF Center for Theoretical Biological Physics, National Biomedical Computation Resource, University of California San Diego, La Jolla, CA 92093, United States
[d] Howard Hughes Medical Institute, University of California San Diego, La Jolla, CA, United States

ARTICLE INFO

Article history:
Received 27 March 2013
Received in revised form 7 May 2013
Accepted 11 May 2013
Available online 23 May 2013

Keywords:
Drug design
Click chemistry
Autogrow
Computational chemistry

ABSTRACT

We here present an improved version of AutoGrow (version 3.0), an evolutionary algorithm that works in conjunction with existing open-source software to automatically optimize candidate ligands for predicted binding affinity and other druglike properties. Though no substitute for the medicinal chemist, AutoGrow 3.0, unlike its predecessors, attempts to introduce some chemical intuition into the automated optimization process. AutoGrow 3.0 uses the rules of click chemistry to guide optimization, greatly enhancing synthesizability. Additionally, the program discards any growing ligand whose physical and chemical properties are not druglike. By carefully crafting chemically feasible druglike molecules, we hope that AutoGrow 3.0 will help supplement the chemist's efforts.

To demonstrate the utility of the program, we use AutoGrow 3.0 to generate predicted inhibitors of three important drug targets: *Trypanosoma brucei* RNA editing ligase 1, peroxisome proliferator-activated receptor γ, and dihydrofolate reductase. In all cases, AutoGrow generates druglike molecules with high predicted binding affinities.

AutoGrow 3.0 is available free of charge (http://autogrow.ucsd.edu) under the terms of the GNU General Public License and has been tested on Linux and Mac OS X.

1. Introduction

Ligand identification and optimization are challenging tasks. In recent years, computational algorithms have played increasingly prominent roles in assisting the medicinal chemist. Driven by the exponential growth of computer power and the ever expanding number of experimentally derived, atomistic structures of receptors and ligand–receptor complexes [1], these programs have been applied at almost every stage of the drug-discovery process. Computational algorithms have assisted in the development of many drugs, including dorzolamide, zanamivir, oseltamivir, nelfinavir, raltegravir, aliskiren, and boceprevir [2–4].

Despite advances in computer-aided drug discovery, the processes of ligand identification and optimization are still largely medicinal-chemist driven. While computers lack the insight and intuition that chemists have, recent efforts have sought to improve automation; the AutoGrow algorithm [5], among others [6–14], has been developed to aid the identification and optimization of predicted ligands. The initial version, released in 2009, uses an evolutionary algorithm in conjunction with existing docking software to add interacting moieties to models of known inhibitors in order to optimize their predicted binding affinities. At the time of its initial release, the main advantage of the program was its degree of automation; beyond the initial setup of fragment libraries and docking parameters, no user interaction is required until the final compounds are presented for evaluation.

However, in the absence of the chemist's insight, AutoGrow versions 1.0 and 2.0 often produce compounds that are neither druglike nor easily synthesizable. These programs are useful for providing chemists with insights into possible ligand–receptor interactions, but if a compound cannot be synthesized and/or lacks the necessary physical properties characteristic of approved drugs [15,16], clinical success is unlikely.

In the current paper, we present AutoGrow 3.0, an improved algorithm that attempts to introduce some chemical intuition into the automated identification/optimization process. Though no substitute for the medicinal chemist, AutoGrow 3.0 can produce

* Corresponding author at: Department of Chemistry & Biochemistry, University of California San Diego, 9500 Gilman Drive, Mail Code 0365, La Jolla, CA 92093-0365, United States. Tel.: +1 858 822 0169; fax: +1 858 534 4974.
E-mail address: jdurrant@ucsd.edu (J.D. Durrant).

1093-3263
http://dx.doi.org/10.1016/j.jmgm.2013.05.006

chemically synthesizable, druglike molecules that may supplement the chemist's efforts. Version 3.0 is significantly improved over previous versions. Additionally, as the new implementation is written in python rather than java, editing and expanding the code is easier than ever.

AutoGrow 3.0 is dependent on several other free software packages, including MGLTools [17], Open Babel [18], AutoDock Vina [19], and NumPy/SciPy [20]. We are hopeful that AutoGrow 3.0 will be even more useful to the drug-design community than previous implementations. A copy can be downloaded free of charge from http://autogrow.ucsd.edu/.

2. Design and implementation

2.1. The AutoGrow 3.0 algorithm

As an evolutionary algorithm, AutoGrow 3.0 deals not with a single ligand, but with populations of ligands. These populations are divided into "generations." Each generation is subject to three operators, called mutation, crossover, and selection [5].

To derive a novel compound *via* mutation, AutoGrow 3.0 first randomly selects one of the many click-chemistry reactions programmed into AutoClickChem [21]. A fragment that can participate in this reaction is then selected at random from a user-specified database and added to the known or suspected ligands by simulating the reaction *in silico*.

AutoClickChem performs two kinds of virtual reactions. Modification reactions involve replacing certain moieties with chemically reactive groups. For example, a halide atom can be replaced with an azide group. In contrast, joining reactions involve combining two distinct molecular models into one *via* simulated click-chemistry reactions. For example, a molecule containing an azide group can be joined to a molecule containing an alkyne group *via* a simulated azide-alkyne Huisgen cycloaddition. AutoGrow 3.0 allows the user to specify whether "mutant" ligands should be derived using both modification and joining reactions, or if joining reactions alone should be permitted.

The AutoGrow 3.0 crossover operator is based on the LigMerge algorithm [22]. First, two "parent" molecules are aligned by superimposing the maximum (largest) substructure common to both. Novel compounds are then generated by systematically mixing and matching the distinct fragments attached to the respective aligned substructures. In this way, "child" molecules can be generated that are topologically similar to but nevertheless distinct from their two "parents."

Once a generation of compounds has been created using the mutation and crossover operators, the selection operator is used to identify the ligands that are the most "fit." A number of criteria are used in selecting the top ligands. First, each ligand is evaluated for druglike properties using Open Babel [18] and python definitions built with the *pymolecule* framework [21]. Compounds that are not druglike are discarded. The user can select whether generated compounds must satisfy Lipinski's Rule of Fives [15] with no violations, Lipinski's Rule of Fives with at most one violation, or the criteria described by Ghose et al. [16].

The user can also instruct AutoGrow to discard any compounds that do not contain specific, key moieties. For example, suppose previous research has identified ten inhibitors that all contain a single carboxylate group. As the carboxylate group may be critical for binding, the user may wish to use AutoGrow to generate novel compounds from these ten that preserve this key moiety. However, AutoClickChem considers carboxylate groups to be reactive and tends to convert them into esters, amides, *etc.* Additionally, LigMerge could potentially generate compounds that do not contain the carboxylate group. To preserve this key moiety, the user can "mark" the two oxygen atoms of the carboxylate group by editing the PDB files of the ten known inhibitors and in every case appending an exclamation point to the atom names of the two carboxylate oxygen atoms. AutoGrow can then be instructed to discard all generated compounds that do not contain at least two marked atoms, thus preserving the key moiety.

Finally, those ligands that remain are subsequently docked into the receptor of interest using AutoDock Vina [19]. Optionally, the docked poses can be reevaluated with NNScore 1.0 [23] or NNScore 2.0 [24]. The best-scoring ligands are then selected to be the founders of the next generation. The compounds of this new generation are again created *via* mutation and crossover operators, this time applied to the best ligands of the previous generation, and the process begins anew, repeating until the user-specified number of generations has been completed.

2.2. Fragment libraries

The mutation (AutoClickChem) operator draws upon a user-specified library of molecular fragments. In the absence of a user-generated fragment library, one of the default libraries that ship with AutoGrow 3.0 can be used. These default libraries were generated by performing sub-structure searches of the compounds in the ZINC database [25] to identify fragments that could potentially participate in any of the many reactions of click chemistry [21]. Molecules containing acid anhydride, acyl halide, alcohol, thiol, alkene, alkyne, amine, azide, carbonochloridate, carboxylate, epoxide, ester, halide, isocyanate, isothiocyanate, sulfonylazide, and thio acid moieties were included. The structures of these compounds were optimized using Schrodinger's LigPrep program in conjunction with the OPLS 2005 forcefield [26,27]. The LigPrep models were then filtered according to molecular weight and categorized by the specific reactive moiety identified. Libraries of fragments with molecular weights less than 150, 200, and 250 Da, containing 2264, 8772, and 21,010 fragments, respectively, are provided with the program.

2.3. AutoGrow runs

To demonstrate the utility of the AutoGrow algorithm, crystal structures of RNA editing ligase 1 (REL1), peroxisome proliferator-activated receptor γ (PPARγ), and dihydrofolate reductase (DHFR) were obtained form the Protein Data Bank (PDB) [1] (PDB IDs: 1XDN [28], 1FM9 [29], and 3DFR [30], respectively). In all three cases, crystallographic water molecules and co-crystallized ligands were removed. We note, however, that AutoGrow does not require that water molecules be removed, and in some cases these molecules may in fact mediate important ligand–receptor interactions. Hydrogen atoms were subsequently added to the protein using PDB2PQR [31,32]. In the case of DHFR, the NDP cofactor was retained, with hydrogen atoms assigned according to those present in the DUD database [33].

For REL1, AutoGrow created 20 mutants per generation and advanced the top ten to the subsequent generation. AutoClickChem, serving as the mutation operator, performed both modification and joining click-chemistry reactions. No crossover operations were permitted. For PPARγ, AutoGrow generated ten mutants and five crossovers for each generation, advancing the top five to the subsequent generation. For DHFR, AutoGrow generated ten mutants and ten crossovers for each generation, advancing the top fifteen to the subsequent generation.

In all three cases, AutoDock Vina [19] was used for docking and scoring. Docking boxes centered on the REL1, PPARγ, and DHFR actives sites measured 30 Å × 30 Å × 20 Å, 40.912 Å × 44.262 Å × 46.769 Å, and 42.851 Å × 44.835 Å × 44.012 Å, respectively. In all three cases, compounds were filtered according

to Lipinski's Rule of Fives (no violations allowed), and fragment addition drew upon the default AutoGrow 3.0 fragment library of compounds with molecular weights less than 250 Da.

3. Results and discussion

A number of ligand-optimization schemes based on evolutionary algorithms have been proposed [34]. Among these, AutoGrow [5] has been used to provide insight into the design of influenza neuraminidase [35,36] and *Trypanosoma brucei* RNA editing ligase 1 inhibitors [37]. AutoGrow 3.0 includes significant improvements over previous versions that will make the algorithm even more useful for drug discovery.

3.1. AutoGrow improvements

The new AutoGrow mutation operator helps ensure chemical synthesizability. In versions 1.0 and 2.0, the replacement of a single hydrogen atom with a molecular fragment chosen at random from a database constitutes a "mutation." While useful as a means of identifying potential interactions and fragments that might aid drug design and optimization, the compounds generated by AutoGrow 1.0 and 2.0 are not necessarily druglike or synthesizable. The mutation operator simply replaces hydrogen atoms with molecular fragments, without regard for the chemistry required to actually generate the compounds *ex silico*. AutoGrow 3.0 addresses these deficits by adding fragments according to the rules of click chemistry [21]. Consequently, the resulting products can be easily synthesized for subsequent experimental testing.

The AutoGrow crossover operator, which generates novel compounds by mixing and matching moieties from two "parents," is also much improved. Based on LigMerge [22], this enhanced operator allows for crossovers between even structurally distinct ligands. Use of the crossover operator is likely to increase the diversity present in each AutoGrow generation, permitting the algorithm to ultimately generate more potent predicted ligands. However, LigMerge can in principle generate compounds that are not easily synthesizable. The user must decide if the potential for enhanced potency justifies the risk of generating chemically intractable compounds. If users are particularly concerned about synthesizability, the crossover operator can be easily deactivated, and predicted ligands can be generated using the mutation operator alone. We do note, however, that LigMerge-generated compounds are still more likely to be synthesizable than compounds derived using the crossover operators of previous AutoGrow versions because the LigMerge compounds are themselves derived from presumably synthesizable molecules (e.g. known inhibitors) and so contain druglike substructures.

Significant improvements have also been made to the selection operator. Previous versions of AutoGrow ignored the druglike properties of the ligands generated; in the current implementation, compounds that do not meet key criteria are discarded immediately. Ligand creation proceeds until the current generation contains the required number of druglike mutants and crossovers. Specifically, ligands are evaluated using the criteria presented by Lipinski et al. [15] and Ghose et al. [16]. Lipinski's Rule of Fives states that an orally active, druglike molecule typically has no more than one violation of the following criteria: the number of hydrogen-bond donors is less than or equal to five, the number of hydrogen-bond acceptors is less than or equal to ten, the molecular weight is less than 500 Da, and the octanol–water partition coefficient (log*P* value) is less than or equal to five. Similarly, Ghose et al. suggest that druglike molecules generally satisfy five criteria: the log*P* is between −0.4 and 5, the molar refractivity is between 40 and 130, the molecular weight is between 160 and 500 Da, the number of atoms is between 20 and 70, and the polar surface area is less than or equal to 140 $Å^2$. In AutoGrow 3.0, the user selects whether generated compounds must satisfy Lipinski's Rule of Fives with no violations, Lipinski's Rule of Fives with at most one violation (as originally proposed), or the criteria described by Ghose and coworkers.

Like previous implementations, AutoGrow 3.0 also considers docking scores when identifying fit ligands. Each ligand is docked into the receptor of interest using AutoDock Vina [19]. AutoGrow 3.0 adds the option to reevaluate the docked poses with NNScore 1.0 [23] or NNScore 2.0 [24], neural-network-based scoring functions that for some systems are more accurate than the Vina scoring function [24,38]. As the performance of different scoring functions is highly dependent on the protein receptor being studied, we recommend using known inhibitors (positive controls) to validate a given scoring function prior to starting an AutoGrow run.

3.2. Examples of usage

AutoGrow 3.0 can generate entirely novel predicted inhibitors *ex nihilo* from very basic starting structures. Alternatively, it can optimize existing inhibitors in order to improve the predicted binding affinity. To demonstrate these two techniques, we provide three examples of use. First, we generate predicted inhibitors of *T. brucei* RNA editing ligase 1 (REL1) by building entirely novel molecules from very basic brominated benzenes, without regard for the structures of known inhibitors. We similarly generate predicted inhibitors of peroxisome proliferator-activated receptor γ (PPARγ) from a small library of slightly more diverse starting molecules. Finally, we demonstrate ligand optimization by generating predicted inhibitors of dihydrofolate reductase (DHFR) using known inhibitors as the starting structures.

3.3. Example: inhibitors of T. brucei RNA editing ligase 1

To demonstrate the utility of AutoGrow 3.0 in creating novel ligands *ex nihilo*, we first designed predicted inhibitors of RNA editing ligase 1 (REL1) from *T. brucei*, the etiological agent of African sleeping sickness. As *Tb*REL1 is critical for the survival of the *T. brucei* parasite [39], it has been the target of several drug-discovery efforts in recent years [40–43].

AutoGrow created the initial generation of compounds from 12 simple halogenated benzenes: bromobenzene, perbromobenzene, pentabromobenzene, and all possible dibromobenzenes, tribromobenzenes, and tetrabromobenzenes. Benzene rings are common drug moieties, and AutoClickChem can convert the attached bromide atoms into other reactive groups, particularly azides. Running on a workstation with eight cores, AutoGrow iterated through 17 generations in 2 days, 10 h. In all, AutoGrow generated 89 synthesizable compounds with docking scores equal to or better than −10 kcal/mol.

Among the top compounds of the 17th generation, compound **1** (Fig. 1) seemed promising. Examination revealed that this compound was produced by multiple AutoClickChem reactions, including two reactions that led to fragment additions. The predicted binding pose of compound **1** (Fig. 2A) is reminiscent of the known binding pose of ATP [28], the natural substrate, in that aromatic rings are buried deep within the adenine-binding pocket where they form π-stacking interactions with F209. Additionally, compound **1** is predicted to form two hydrogen bonds with the receptor residues H89 and R288 and to participate in possible cation-π interactions with R111 at the periphery.

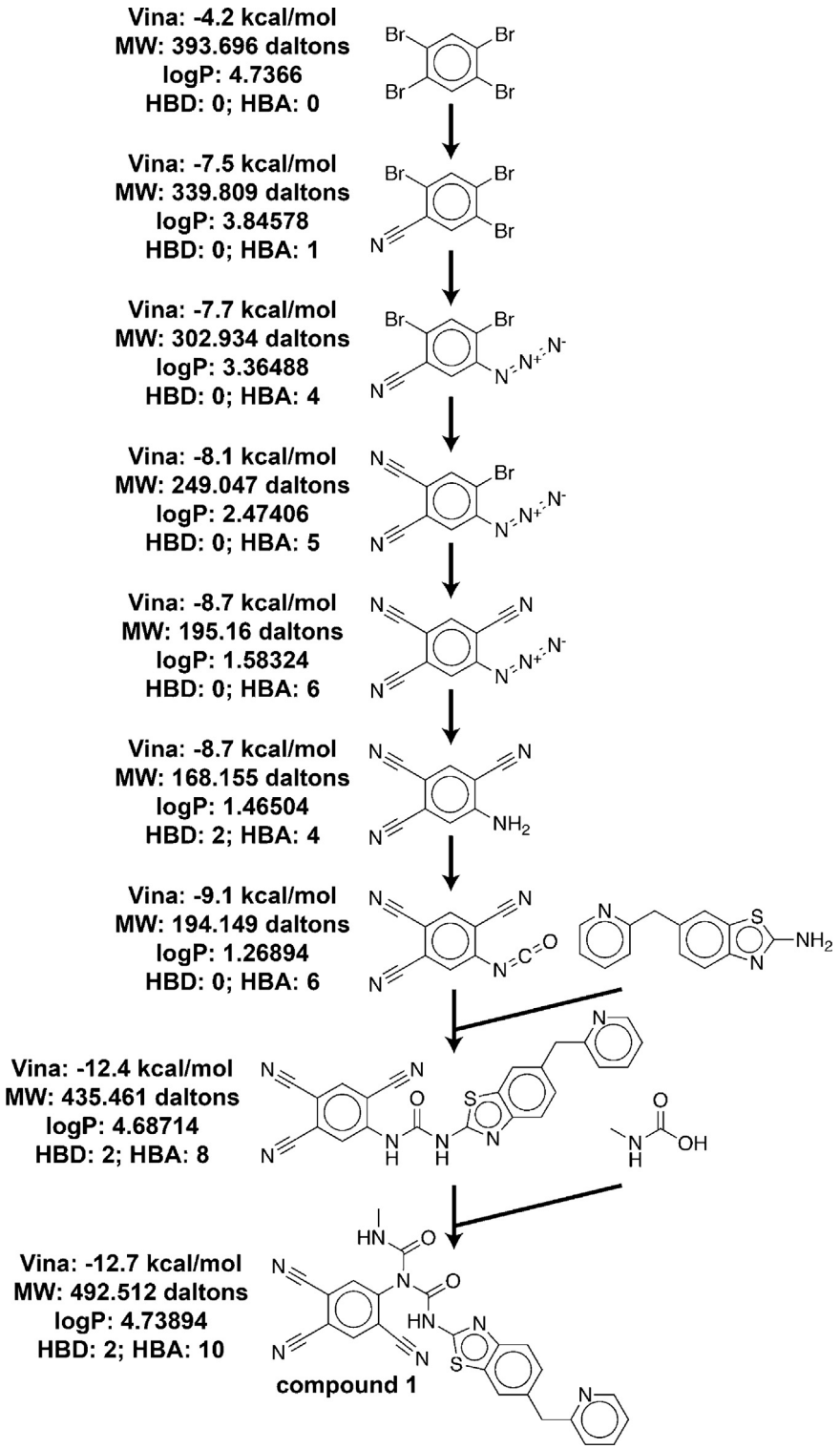

Fig. 1. The evolutionary steps taken to generate a predicted *Tb*REL1 inhibitor.

Fig. 3 shows the average Vina score of the top-five ranked compounds of each generation. After 17 generations, the best-scoring AutoGrow compound had a predicted binding affinity of −12.7 kcal/mol, comparable to the docking scores of known *Tb*REL1 inhibitors. This inhibitor was ultimately derived from a tetrabromobenzene fragment whose predicted *Tb*REL1 binding affinity was only −4.2 kcal/mol, demonstrating significant optimization (Fig. 1).

3.4. Example: inhibitors of peroxisome proliferator-activated receptor γ

As an additional example of creating predicted ligands *ex nihilo*, we next applied AutoGrow to peroxisome proliferator-activated receptor γ (PPARγ), the molecular target of several FDA-approved treatments for diabetes.

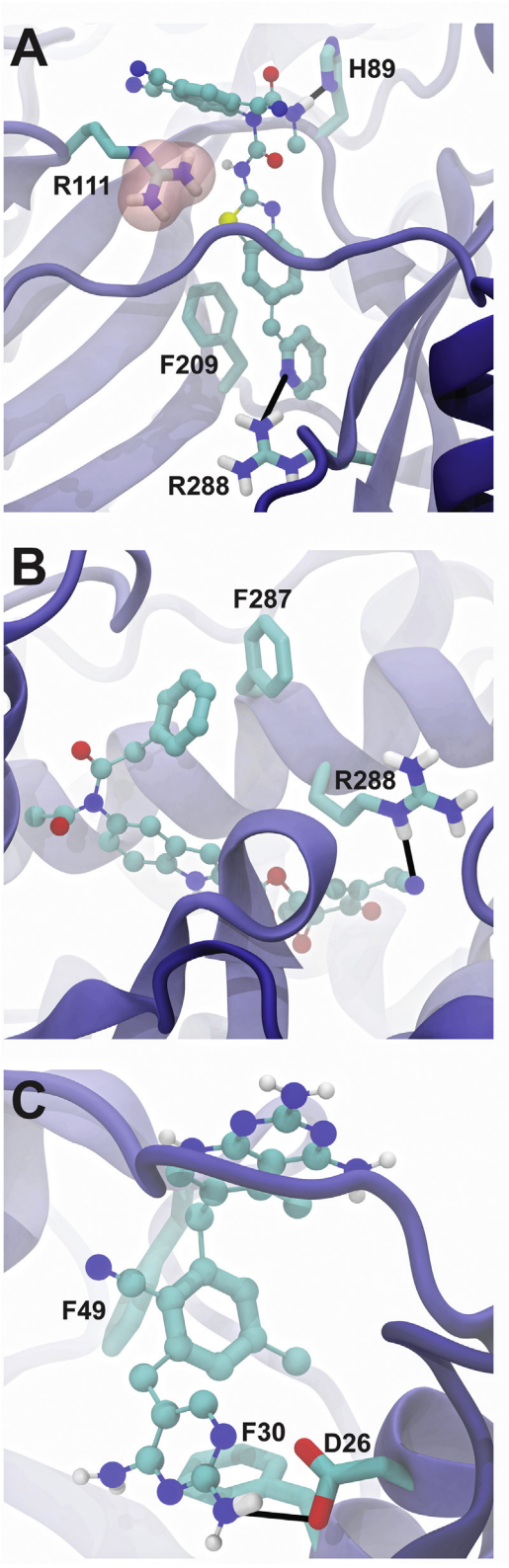

Fig. 2. The predicted binding poses of top AutoGrow-generated compounds. (A) *Tb*REL1; (B) PPARγ; (C) DHFR.

AutoGrow created the first generation of compounds from a series of small bromine-containing fragments that are somewhat more diverse than the benzene derivatives used in the REL1 runs: 2-phenylacetyl bromide, (bromomethyl)benzene, (R)-bromo(phenyl)methanol, benzoyl bromide, benzimidoyl bromide,

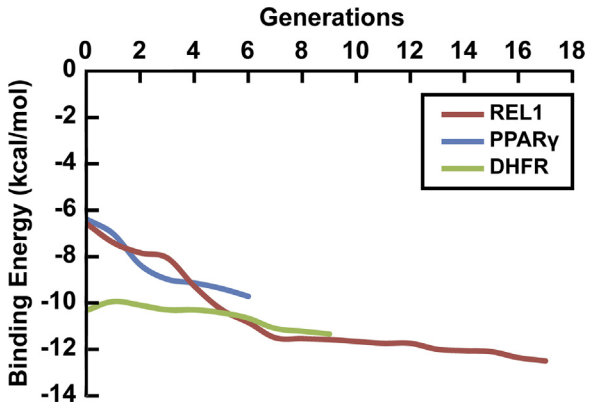

Fig. 3. The average Vina score of the top-five ranked compounds of each generation.

N-bromobenzenaminium, benzoyl bromide, phenyl carbonobromidate, and *N*-bromoaniline. As before, brominated compounds were selected because AutoClickChem can easily convert them into other reactive groups, particularly azides.

Running on 8 cores, AutoGrow iterated through 6 generations in 4 h and 6 min. The program was manually terminated after spending nearly 24 h trying to generate the 15 ligands of the seventh generation. Thus we see that it is possible for AutoGrow to take evolutionary paths that are ultimately cul-de-sacs; in some situations, AutoGrow may not be able to generate a sufficient number of novel ligands to complete a generation. This is especially true when many constraints are placed on ligand growth (e.g. too few mutant or crossover products per generation, too few initial ligands from which to derive the ligands of the first generation, too strict criteria for determining whether a compound is druglike, *etc.*).

If the user is reluctant to manually monitor the AutoGrow output for this cul-de-sac scenario, the program can also be instructed to automatically terminate if any generation takes more than a user-specified number of seconds. Additionally, if an evolutionary cul-de-sac occurs early in an AutoGrow run, one need only restart the program. AutoGrow 3.0, like its predecessors, is non-deterministic. Each run produces different potential ligands by following different evolutionary paths.

Compound **2** (Fig. 4) of the sixth generation had the best predicted binding energy while maintaining druglike properties. Examination revealed that this compound was produced by multiple AutoClickChem and LigMerge operations. While one should not place too much confidence in a docked pose, it is nevertheless noteworthy that the predicted pose of compound **2** is credible (Fig. 2B). Specifically, aside from having excellent shape complementarity, the compound is also predicted to form a hydrogen bond with R288 and a π-stacking interaction with F287.

Fig. 3 shows the average Vina score of the top-five ranked compounds of each generation. After 6 generations, the best-scoring AutoGrow compound had a predicted binding affinity of −10.1 kcal/mol. This inhibitor was ultimately derived from 2-phenylacetyl bromide, whose predicted PPARγ binding affinity was only −7.0 kcal/mol, demonstrating significant optimization. In all, AutoGrow generated 2 easily synthesizable compounds with docking scores equal to or better than −10 kcal/mol.

3.5. Example: inhibitors of dihydrofolate reductase

Having shown that AutoGrow 3.0 can be used to generate predicted ligands *ex nihilo*, we next used AutoGrow to optimize known ligands of dihydrofolate reductase (DHFR), the molecular target of several anti-bacterial and anti-cancer agents [44,45].

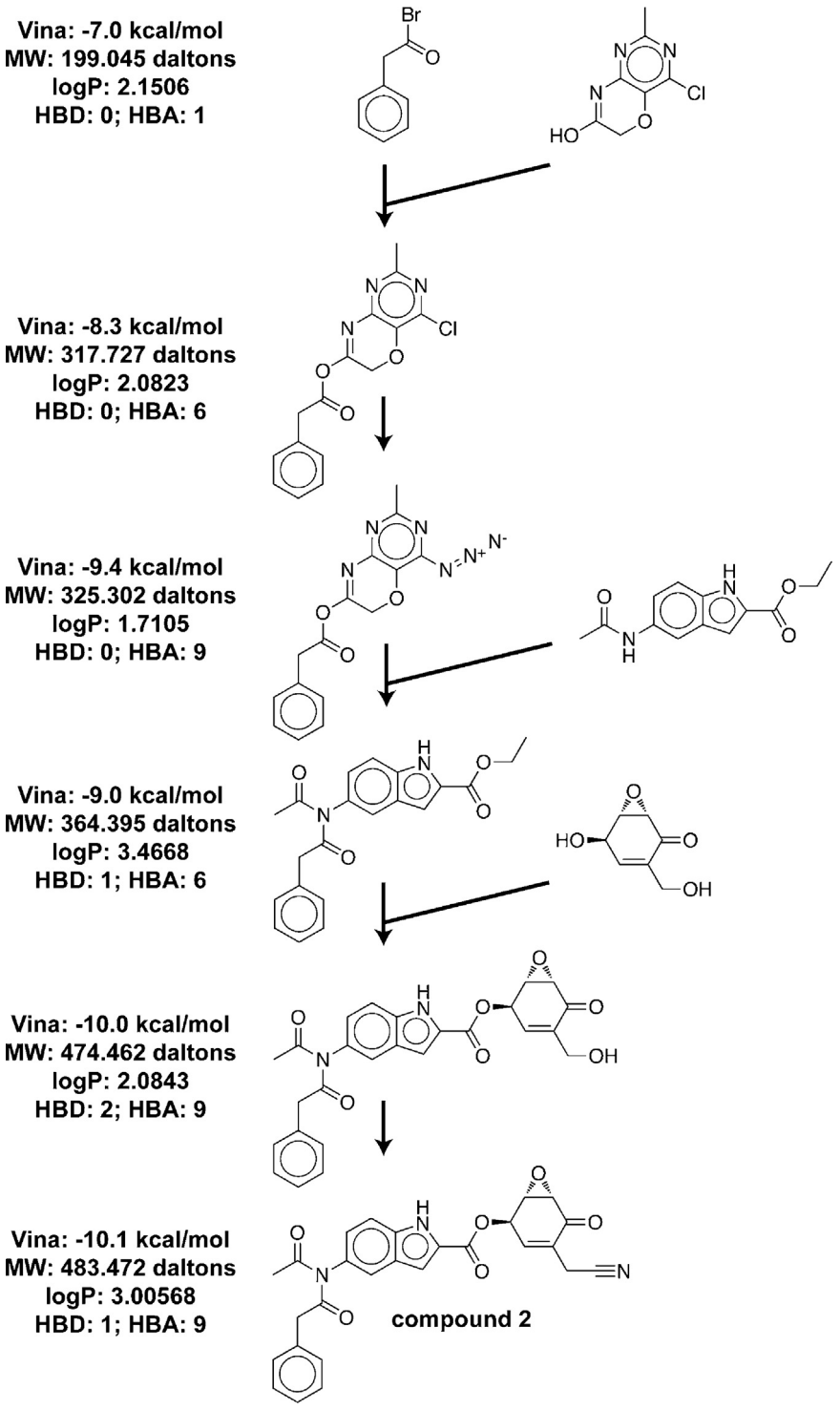

Fig. 4. The evolutionary steps taken to generate a predicted PPARγ inhibitor.

The initial AutoGrow generation was created from 66 models of known DHFR inhibitors. These models were derived from the top 30 DHFR inhibitors with the lowest IC_{50} values listed in the BindingDB database [46] using Schrodinger's LigPrep program. Each of these ligands contained a pyrimidine-2,4-diamine substructure that binds deep in the DHFR folate-binding site. As this moiety is likely critical to binding, AutoGrow was instructed to only consider compounds that preserved it.

Running on a workstation with eight cores, AutoGrow iterated through 9 generations in 6 h, 50 min. Among the top compounds of the 9th generation, compound **3** (Fig. 5) had the highest Vina score. Compound **3** was produced by multiple AutoClickChem reactions and LigMerge crossovers. The predicted binding pose (Fig. 2C) is similar to that of known inhibitors in that it positions the pyrimidine-2,4-diamine substructure deep within the folate-binding pocket [30]. Aside from hydrophobic contacts, compound

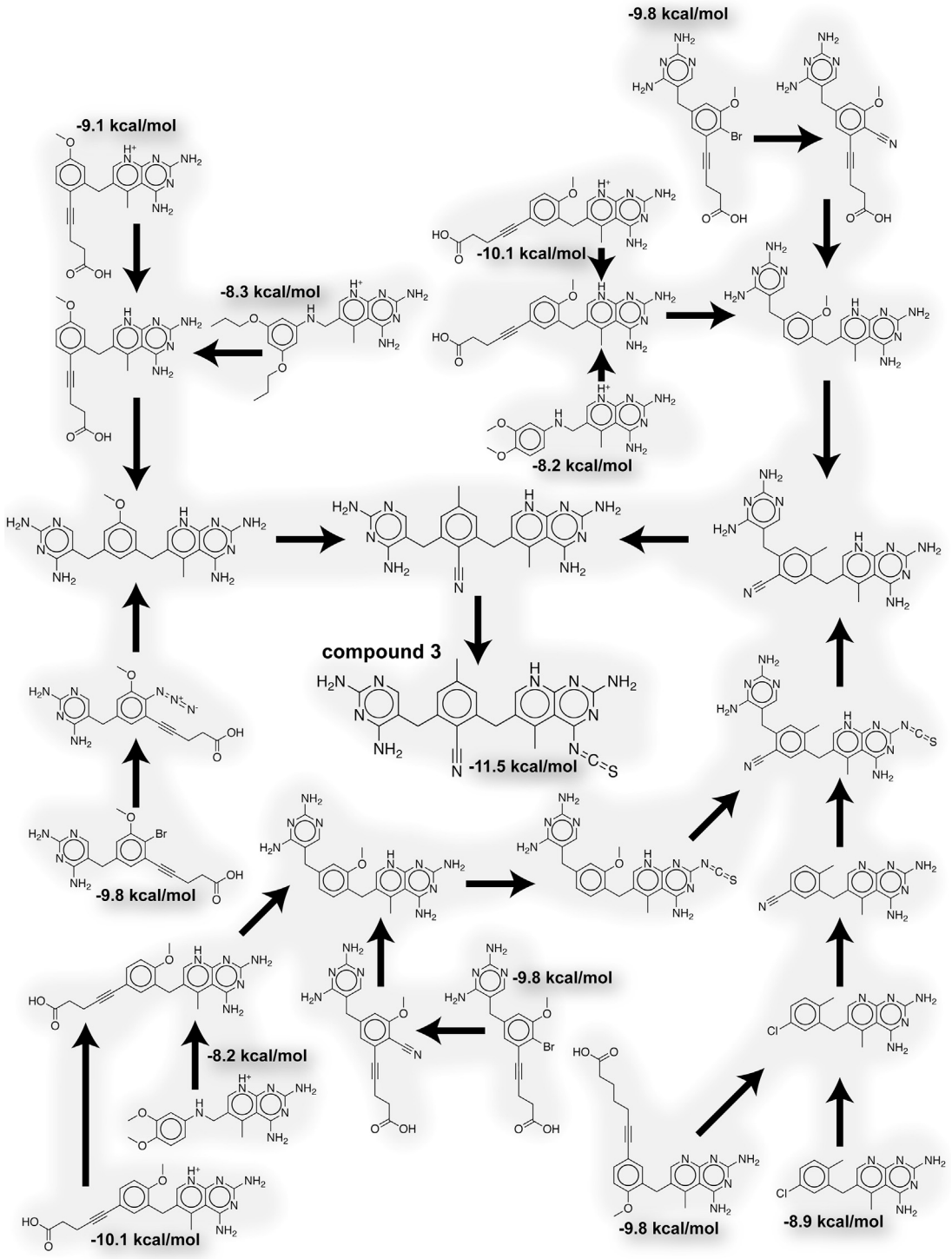

Fig. 5. The evolutionary steps taken to generate a predicted DHFR inhibitor. A subtle gray background is included to facilitate visualization. The evolutionary paths depicted are admittedly convoluted; the primary purpose of this figure is to demonstrate the complexity of the AutoGrow evolutionary process rather than to provide specific chemical details.

3 is predicted to form a hydrogen bond with the D26 carboxylate side chain and two T-shaped aromatic interactions with F30 and F49 [47].

Fig. 3 shows the average Vina score of the top-five ranked compounds of each generation. After 9 generations, the best-scoring AutoGrow compound had a predicted binding affinity of −11.5 kcal/mol. This compound was ultimately derived from seven known DHFR inhibitors with docking scores ranging from −8.2 to −10.1 kcal/mol. This improvement in predicted binding affinity is more modest than that of the two previous examples, as expected given that the original compounds were known inhibitors that had presumably already been subject to optimization.

In all, AutoGrow generated 18 easily synthesizable compounds with docking scores equal to or better than −10 kcal/mol.

4. Conclusions/availability

In the current work, we built upon our previous experience with virtual-screening and compound-library design to create a new version of AutoGrow (3.0) that is significantly improved over previous releases. While no substitute for the medicinal chemist, AutoGrow 3.0 does incorporate some chemical intuition into its ligand-growing strategy. Unlike previous versions of AutoGrow, version 3.0 rejects any compounds that are not druglike. Additionally, the program now adds fragments to core scaffolds by simulating the reactions of click chemistry rather than merely replacing hydrogen atoms with molecular fragments without regard for synthesizability. AutoGrow 3.0 has been specifically tested on Ubuntu 12.10 with python 2.7.3, NumPy 1.6.2, SciPy 0.10.1, MGLTools 1.5.6rc3, Open Babel 2.3.2, and AutoDock Vina 1.1.2; Scientific Linux 6.2 with python 2.6.6, NumPy 1.6.2, SciPy 0.11.0, MGLTools 1.5.4, Open Babel 2.3.1, and AutoDock Vina 1.1.2; and Mac OS X 10.8.1 with python 2.7.2, NumPy 1.6.1, SciPy 0.12.0dev, MGLTools 1.5.6, Open Babel 2.3.1, and AutoDock Vina 1.1.2. We are hopeful that AutoGrow 3.0 will be a helpful tool for the drug-discovery community.

Acknowledgments

We would like to thank Mr. Rajeev Gangal of Sai Life Sciences Ltd. India for helpful discussions regarding program improvements and features. Funding for this work was provided by NIH GM31749, NSF MCB-1020765, and MCA93S013 to JAM. Support from the Howard Hughes Medical Institute, the NSF Supercomputer Centers, the San Diego Supercomputer Center, the W.M. Keck Foundation, the National Biomedical Computational Resource, and the Center for Theoretical Biological Physics is also gratefully acknowledged.

References

[1] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, et al., The protein data bank, Nucleic Acids Research 28 (2000) 235–242.
[2] T.T. Talele, S.A. Khedkar, A.C. Rigby, Successful applications of computer aided drug discovery: moving drugs from concept to the clinic, Current Topics in Medicinal Chemistry 10 (2010) 127–141.
[3] G. Schneider, U. Fechner, Computer-based de novo design of drug-like molecules, Nature Reviews Drug Discovery 4 (2005) 649.
[4] J.A. McCammon, Computer-aided drug discovery: two antiviral drugs for HIV/AIDS, in: T. Schlick (Ed.), Innovations in Biomolecular Modeling and Simulations, vol. 2, Royal Society of Chemistry, 2012, pp. 316–319.
[5] J.D. Durrant, R.E. Amaro, J.A. McCammon, AutoGrow: a novel algorithm for protein inhibitor design, Chemical Biology and Drug Design 73 (2009) 168–178.
[6] V. Gillet, A.P. Johnson, P. Mata, S. Sike, P. Williams, SPROUT: a program for structure generation, Journal of Computer-Aided Molecular Design 7 (1993) 127.
[7] J.B. Moon, W.J. Howe, Computer design of bioactive molecules: a method for receptor-based de novo ligand design, Proteins: Structure, Function, and Genetics 11 (1991) 314.
[8] H.-J. Bohm, The computer program LUDI: a new method for the de novo design of enzyme inhibitors, Journal of Computer-Aided Molecular Design 6 (1992) 61.
[9] V.A. Gillet, A.P. Johnson, P. Mata, S. Sike, Automated structure design in 3D, Tetrahedron 3 (1990) 681.
[10] S.H. Rotstein, M.A. Murcko, GroupBuild: a fragment-based method for de novo drug design, Journal of Medicinal Chemistry 36 (1993) 1700.
[11] R.S. Bohacek, C. McMartin, Multiple highly diverse structures complementary to enzyme binding-sites—results of extensive application of a de-novo design method incorporating combinatorial growth, Journal of the American Chemical Society 116 (1994) 5560.
[12] D.E. Clark, D. Frenkel, S.A. Levy, J. Li, C.W. Murray, B. Robson, et al., PRO_LIGAND: an approach to de novo molecular design. 1. Application to the design of organic molecules, Journal of Computer-Aided Molecular Design 9 (1995) 13.
[13] D.R. Westhead, D.E. Clark, D. Frenkel, J. Li, C.W. Murray, B. Robson, et al., PRO_LIGAND: an approach to de novo molecular design. 1. Application to the design of organic molecules, Journal of Computer-Aided Molecular Design 9 (1995) 139.
[14] R.S. DeWitte, E.I. Shakhnovich, SMoG: de novo design method based on simple, fast, and accurate free energy estimates. 1. Methodology and supporting evidence, Journal of the American Chemical Society 118 (1996) 11733.
[15] C.A. Lipinski, F. Lombardo, B.W. Dominy, P.J. Feeney, Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings, Advanced Drug Delivery Reviews 46 (2001) 3–26.
[16] A.K. Ghose, V.N. Viswanadhan, J.J. Wendoloski, A knowledge-based approach in designing combinatorial or medicinal chemistry libraries for drug discovery. 1. A qualitative and quantitative characterization of known drug databases, Journal of Combinatorial Chemistry 1 (1999) 55–68.
[17] M.F. Sanner, Python: a programming language for software integration and development, Journal of Molecular Graphics and Modelling 17 (1999) 57–61.
[18] N.M. O'Boyle, M. Banck, C.A. James, C. Morley, T. Vandermeersch, G.R. Hutchison, Open Babel: an open chemical toolbox, Journal of Cheminformatics 3 (2011) 33.
[19] O. Trott, A.J. Olson, AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading, Journal of Computational Chemistry 31 (2009) 455–461.
[20] E. Jones, T. Oliphant, P. Peterson, SciPy: Open Source Scientific Tools for Python, 2001.
[21] J.D. Durrant, J.A. McCammon, AutoClickChem: click chemistry in silico, PLoS Computational Biology 8 (2012) e1002397.
[22] S. Lindert, J.D. Durrant, J.A. McCammon, LigMerge: a fast algorithm to generate models of novel potential ligands from sets of known binders, Chemical Biology and Drug Design 80 (2012) 358–365.
[23] J.D. Durrant, J.A. McCammon, NNScore: a neural-network-based scoring function for the characterization of protein–ligand complexes, Journal of Chemical Information and Modeling 50 (2010) 1865–1871.
[24] J.D. Durrant, J.A. McCammon, NNScore 2.0: a neural-network receptor–ligand scoring function, Journal of Chemical Information and Modeling 51 (2011) 2897–2903.
[25] J.J. Irwin, B.K. Shoichet, ZINC—a free database of commercially available compounds for virtual screening, Journal of Chemical Information and Modeling 45 (2005) 177–182.
[26] W.L. Jorgensen, D.S. Maxwell, J. TiradoRives, Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids, Journal of the American Chemical Society 118 (1996) 11225–11236.
[27] G.A. Kaminski, R.A. Friesner, J. Tirado-Rives, W.L. Jorgensen, Evaluation and reparametrization of the OPLS-AA force field for proteins via comparison with accurate quantum chemical calculations on peptides, Journal of Physical Chemistry B 105 (2001) 6474–6487.
[28] J. Deng, A. Schnaufer, R. Salavati, K.D. Stuart, W.G. Hol, High resolution crystal structure of a key editosome enzyme from *Trypanosoma brucei*: RNA editing ligase 1, Journal of Molecular Biology 343 (2004) 601–613.
[29] R.T. Gampe, V.G. Montana, M.H. Lambert, A.B. Miller, R.K. Bledsoe, M.V. Milburn, et al., Asymmetry in the PPAR gamma/RXR alpha crystal structure reveals the molecular basis of heterodimerization among nuclear receptors, Molecular Cell 5 (2000) 545–555.
[30] J.T. Bolin, D.J. Filman, D.A. Matthews, R.C. Hamlin, J. Kraut, Crystal-structures of *Escherichia coli* and *Lactobacillus casei* dihydrofolate-reductase refined at 1.7 a resolution.1. General features and binding of methotrexate, Journal of Biological Chemistry 257 (1982) 3650–3662.
[31] T.J. Dolinsky, P. Czodrowski, H. Li, J.E. Nielsen, J.H. Jensen, G. Klebe, et al., PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations, Nucleic Acids Research 35 (2007) W5–W522.
[32] T.J. Dolinsky, J.E. Nielsen, J.A. McCammon, N.A. Baker, PDB2PQR: an automated pipeline for the setup of Poisson–Boltzmann electrostatics calculations, Nucleic Acids Research 32 (2004) W7–W665.
[33] N. Huang, B.K. Shoichet, J.J. Irwin, Benchmarking sets for molecular docking, Journal of Medicinal Chemistry 49 (2006) 6789–6801.
[34] J.A. Hiss, M. Hartenfeller, G. Schneider, Concepts and applications of "natural computing" techniques in de novo drug and peptide design, Current Pharmaceutical Design 16 (2010) 1656–1665.
[35] S.Q. Wang, X.C. Cheng, W.L. Dong, R.L. Wang, K.C. Chou, Three new powerful oseltamivir derivatives for inhibiting the neuraminidase of influenza virus, Biochemical and Biophysical Research Communications 401 (2010) 188–191.
[36] J.D. Durrant, J.A. McCammon, Potential drug-like inhibitors of group 1 influenza neuraminidase identified through computer-aided drug design, Computational Biology and Chemistry 34 (2010) 97–105.
[37] J.D. Durrant, J.A. McCammon, Towards the development of novel *Trypanosoma brucei* RNA editing ligase 1 inhibitors, BMC Pharmacology 11 (2011) 9.
[38] S. Lindert, W. Zhu, Y.L. Liu, R. Pang, E. Oldfield, J.A. McCammon, Farnesyl diphosphate synthase inhibitors from in silico screening, Chemical Biology and Drug Design 81 (6) (2013) 742–748.
[39] A. Schnaufer, A.K. Panigrahi, B. Panicucci, R.P. Igo Jr., R. Salavati, K. Stuart, An RNA ligase essential for RNA editing and survival of the bloodstream form of *Trypanosoma brucei*, Science 291 (2001) 2159–2162.
[40] R.E. Amaro, A. Schnaufer, H. Interthal, W. Hol, K.D. Stuart, J.A. McCammon, Discovery of drug-like inhibitors of an essential RNA-editing ligase in *Trypanosoma brucei*, Proceedings of the National Academy of Sciences 105 (2008) 17278–17283.
[41] R.E. Amaro, R.V. Swift, J.A. McCammon, Functional and structural insights revealed by molecular dynamics simulations of an essential RNA editing ligase in *Trypanosoma brucei*, PLOS Neglected Tropical Diseases 1 (2007) e68.

[42] J.D. Durrant, R.E. Amaro, L. Xie, M.D. Urbaniak, M.A. Ferguson, A. Haapalainen, et al., A multidimensional strategy to detect polypharmacological targets in the absence of structural and sequence homology, PLoS Computational Biology 6 (2010) e1000648.
[43] J.D. Durrant, L. Hall, R.V. Swift, M. Landon, A. Schnaufer, R.E. Amaro, Novel naphthalene-based inhibitors of *Trypanosoma brucei* RNA editing ligase 1, PLOS Neglected Tropical Diseases 4 (2010) e803.
[44] R.L. Then, Antimicrobial dihydrofolate reductase inhibitors—achievements and future options: review, Journal of Chemotherapy 16 (2004) 3–12.
[45] A.A. Adjei, A review of the pharmacology and clinical activity of new chemotherapy agents for the treatment of colorectal cancer, British Journal of Clinical Pharmacology 48 (1999) 265–277.
[46] T. Liu, Y. Lin, X. Wen, R.N. Jorissen, M.K. Gilson, BindingDB: a web-accessible database of experimentally determined protein–ligand binding affinities, Nucleic Acids Research 35 (2007) D198–D201.
[47] M.O. Sinnokrot, E.F. Valeev, C.D. Sherrill, Estimates of the ab initio limit for pi-pi interactions: the benzene dimer, Journal of the American Chemical Society 124 (2002) 10887–10893.