Pharmacy and Applied Health Science Journal
Volume 2, Number 1 (2023) Knowledge University

DEVELOPMENT AND APPLICATION OF STRUCTURE-ASSISTED ATOM-BASED DE NOVO MOLECULAR DESIGN (SAAD) APPROACH USING MATLAB

MOHAMMED NOORALDEEN MAHMOD AL-QATTAN [1]*, MOHD NIZAM MORDI[2]

[1]Department of pharmacy, College of Pharmacy, Knowledge University, Erbil 44001, Iraq
[1]College of Pharmacy, Ninevah University, Mosul, Iraq
[2]Centre for Drug Research, University sains Malaysia, 11800 USM, Pulau Pinang, Malaysia

DOI: https://doi.org/10.59480/phahs.v1i2.15

How to cite this article:. Mohammed Nooraldeen Mahmod Al-Qattan, Mohd Nizam Mordi, Development and Application of Structure Assisted Atom Based de Novo Molecular Design (SAAD) Approach using matlab. PHARM. APPL. H. SCI. [Internet]. 2023;2(1):9-22. Available from: https://phahs.knu.edu.iq/index.php/phahs/article/view/15 This is an open access article distributed under the Creative Commons Attribution License: Attribution-Non Commercial-No Derivatives 4.0 International (CC BY-NC-ND 4.0)

Corresponding Author:
MOHAMMED NOORALDEEN MAHMOD AL-QATTAN *

Department of pharmacy, College of Pharmacy, Knowledge University, Erbil 44001, Iraq, College of Pharmacy, Ninevah University, Mosul, Iraq

Email: mohammed.n.alqattan@gmail.com

Received: 14 November 2022
Accepted: 20 April 2023
Published: 30 June 2023

Abstract

In the context of drug design and development, phytochemical and synthetic libraries are starting places for searching leads. However, knowing crystal structure of receptor may help scratching molecules that complements available interaction sites using *de novo* molecular design approaches. This research describes the implementation of genetic algorithm optimization technique for construction of molecular scaffold composed from sp2 carbon, oxygen and nitrogen atoms. Using Matlab, functions for molecule construction, crossover, mutation and fitness measurement were individually programmed. Fitness function was derived from AutoDock3.05 function which calculates molecular interaction energy with protein using grid map. The approach was successfully applied to design glutathione-S-transferase inhibitory molecule at sub-site of the binding pocket that is originally occupied by γ-glutamyl and cysteinyl moieties of glutathione substrate. Starting from scratches, the approach was able to design molecular scaffolds complementarily filling the pocket.

***Keywords*:** Keywords: De novo design, atom-based, matlab, docking, Glutathione-S-transferase, Computational molecular design

1. Introduction

Drug design and discovery usually starts by searching lead compounds to satisfy receptor binding site. Although phytochemical and synthetic libraries are starting places for searching leads, by virtual and/or high throughput screening, however, knowing crystal structure of binding pocket may aid scratching molecules that complements the available interaction sites using *de novo* molecular design approaches. De novo molecular design is the process of automatically proposing novel chemical structures that optimally satisfy a desired molecular profile (Meyers et al., 2021).

Over years, many *de novo* molecular design approaches are being developed which frequently

implement newer molecular representations, scoring functions as well as optimization and adaptation algorithms (Schneider, 2013). Although variation in molecular representations frequently follow size (i.e. atoms or fragments) and/or type (graph, SMILES code, etc.), variation in optimization and adaptation algorithms is steadily increasing such as genetic algorithm and neural networks, respectively. Scoring functions are used to measure molecular fitness either against binding pocket (structure-based) using docking score (Spiegel and Durrant, 2020), by calculating simple physicochemical properties (Herring and Eden, 2015) or by using trained neural network (Olivecrona et al., 2017). For example, atom-based de novo design may use SMILES code representation for atoms, variational autoencoder neural network as adaptation method and drug-likeness and/or docking as scoring functions (Kusner et al., 2017; Yoshikawa et al., 2018). While docking scores are used if receptor structure is known (structure-based), a database of molecules can be used to train neural network to generate SMILES codes of relevant structures (Segler et al., 2018).

Graph-based method for molecular representation and genetic algorithm was used to develop several fragment-based de novo molecular design programs (Pegg et al., 2001; Devi et al., 2014; Chu and He, 2019) which have some advantages over SMILES representation (Leguy et al., 2020). Structure-based Atom-based de novo molecular design (SAAD) is developed in this study to use Matlab and free AutoGrid software and made open source to be a good starting point for beginner drug designers. Unlike previous work which use merely sp3 carbon atoms (Rotstein and Murcko, 1993), SAAD is programmed with Matlab (© 1994-2021 The MathWorks, Inc.) to use genetic algorithm to evolve molecules composed from sp2 atoms for all of carbon, oxygen and nitrogen. In order to efficiently include most of the available protein interaction sites, grid map for interaction field is used to guide the *de novo* design.

1.2. Methodology

Under Global Optimization Toolbox, Matlab R2017b offers a set of algorithms which can be employed to solve optimization problems. Of the available algorithms; genetic and simulated annealing algorithms are most commonly used. The *de novo* design of lead molecules was translated into an optimization problem to be solved for atomic arrangement in a molecule that fits the binding pocket (i.e. structure-assisted or based). The Structure-Assisted Atom-based *De novo* molecular design (SAAD) approach was programmed in Matlab and functionally uses AUTOGRID to generate grid maps for receptor and OBABEL to calculate partial atomic charge for the generated molecules.

1.2.1. Preparing component functions for genetic algorithm in Matlab R2017b

The genetic algorithm (GA) solver in Matlab R2017b has been programmed to relay on separate functions that perform solution creation, crossover, mutation, and selection operations. However, the GA solver assumes that the fitness function takes a single vector (i.e. single row matrix) as an input (i.e. a solution). The number of elements in the input vector equals to the number of variables in the fitness function. Therefore, the creation, crossover, mutation and selection functions available in Matlab R2017b can only handle numerical vectors.

In order to apply GA for molecular design, the fitness function should be able to take input as molecular structure. Similarly, creation, crossover, and mutation functions should be able to perform operations on molecular structure. To take over the process in Matlab, molecules were represented as cell array called gene (**Error! Reference source not found.**). The gene represents a single molecular solution and is composed from three sub matrices each of type double. The first matrix stores the atomic descriptors of the molecule (i.e. atomic XYZ coordinates, type, charge, connectivity, etc.). The second and third matrices store the XYZ

coordinates and quaternion angles for molecular center of geometry, respectively.

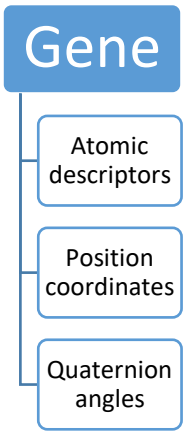


Figure 1: Schematic representation of gene composites that encodes full molecular structure. The gene is composed from three matrices, namely: atomic descriptors, position coordinates, and quaternion angles

Most of the functions required for solution creation, crossover, mutation and fitness measurement were individually designed (**Error! Reference source not found.**). For compatibility purposes, a fitness function has been designed to accept the gene, and then translate it to molecular phenotype before evaluating molecule-protein interaction energy. To further improve computational speed, the fitness function has been designed to accept an array of genes (a chromosome) and produce an array of energy values.

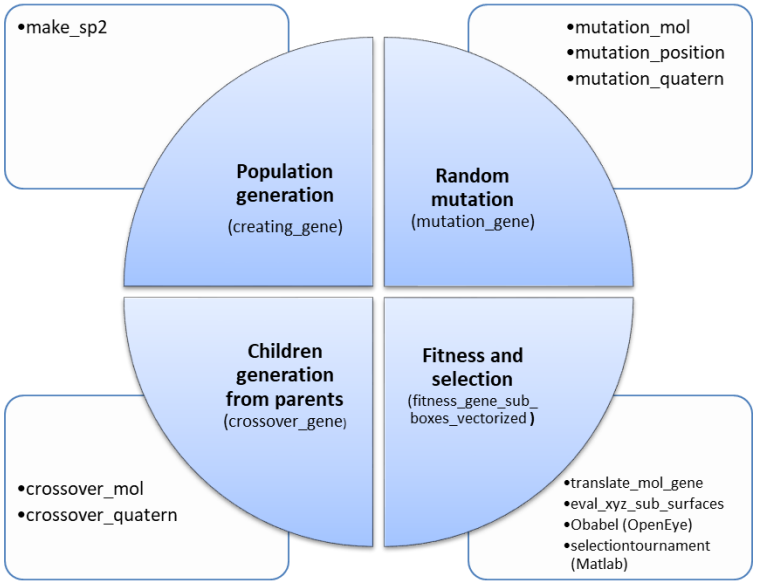


Figure 2: A schematic representation of the four stages as well as the main and sub functions used in Matlab genetic algorithm for Structure-Assisted Atom-based De novo molecular design (SAAD).

1.2.1.1. Molecule creation functions (population generation)

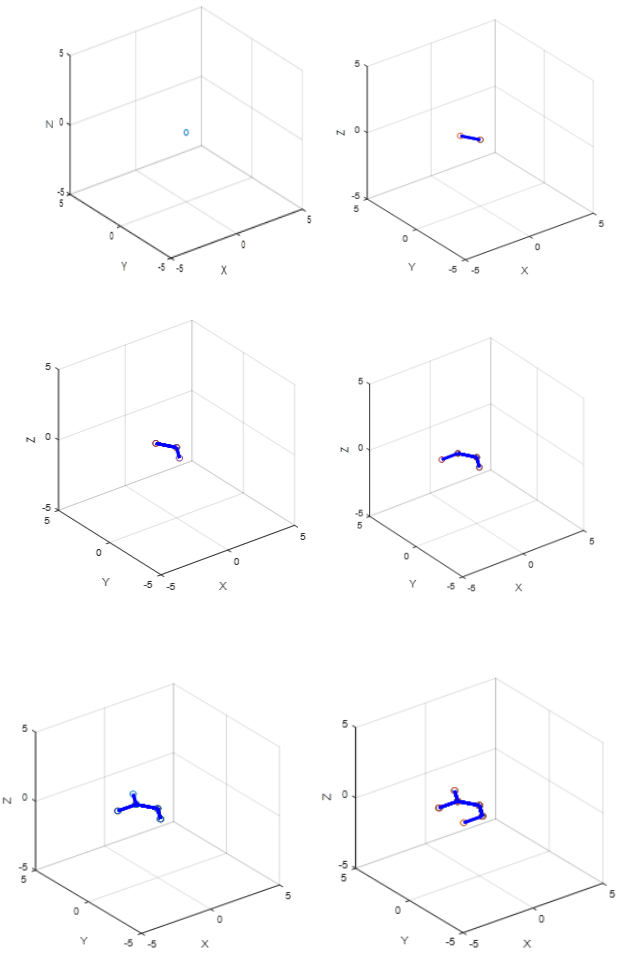
The main function which controls the creation of molecular solutions is `creating_gene3` which creates chromosome of different gene solutions. Each gene is created by `creating_gene(num_atoms,position,lb, ub, O_num,N_num)` function. The `creating_gene` function takes total number of atoms (`num_atoms`) and xyz `position` of created molecule, otherwise, the created molecule will be positioned randomly between lower bound (`lb`) and upper bound (`ub`) coordination limits. The created molecular skeleton is composed from carbon, oxygen and nitrogen atoms when the number of oxygen (`O_num`), and nitrogen (`N_num`) heteroatoms are defined, otherwise, it is composed absolutely from carbon atoms. The arrangement of heteroatoms is absolutely random within the carbon skeleton, however, avoids three connections for oxygen atom and existence of directly connected heteroatoms.

The `creating_gene` function is linked to `make_sp2` function which creates atomic descriptor matrix for the molecule. The latter function creates random 2D molecule fully from sp2 hybridized atoms by using defined total number of atoms as well as oxygen and nitrogen percentages. The function use mathematical equation to generate sp2 connected atomic geometry. It is compulsory for each generated molecule to contain one atom at (0,0,0) coordinate to avoid possible disconnection during molecular crossover. Each molecule is generated as n by m matrix, where n equals the number of atoms and m number of descriptor for each atom which is equal to 15 in this version of the function (**Error! Reference source not found.**).

Table 1: The arrangement of atomic descriptor matrix generated by make_sp2_7ed function.

Column(s)	Atomic descriptor
1	Atom number
2-4	XYZ coordinates
5	Number of atoms linked to current atoms
6-9	Rank numbers for the atoms linked to current atom
10-13	Bond type (1,2, and 3 for single, double and triple bonds)
14	Atom type in ASCII codes: 67 for C, 79 for O, 78 for N, and 72 for H.
15	Partial atomic charge

The `make_sp` function starts by creating initial (master) atom at XYZ of (0,0,0). Subsequently, the function search for atoms in the molecule to be used as origins for newer atoms creation. The origins are principally atoms having value of less than 3 in column 5 i.e. less than 3 connections. Random atom is selected from the available origins to create a new atom at randomly signed ±120 degree (sp2 hybridization geometry) via duplicating and rotating coordinates of a parent atom of the selected origin. The newly created atom is checked for overlapping with other atoms before being accepted. The iteration of new atom creation is repeated until the required number of atoms within the molecule is satisfied (Figure 1). The final step includes adjusting the contents of connection Table (columns 6-9) according to linked atom rank. Using the same parameters in make_sp2 function, different molecules are obtained in each function-call (Figure 2).



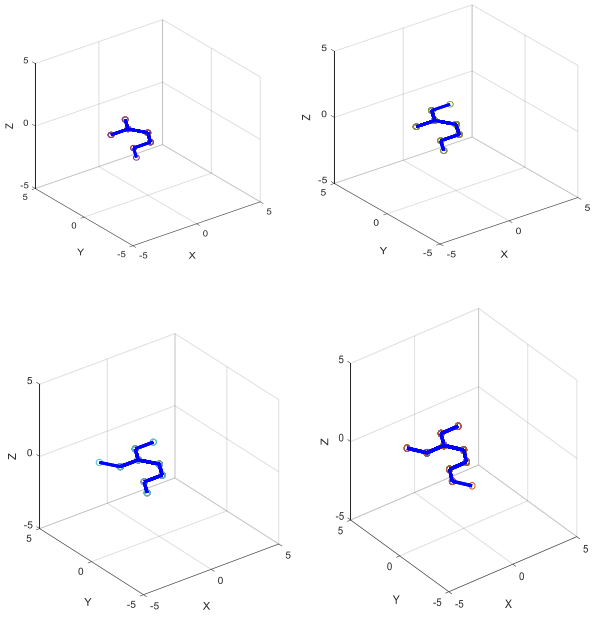


Figure 1: The progress of atom generation using make_sp2 function

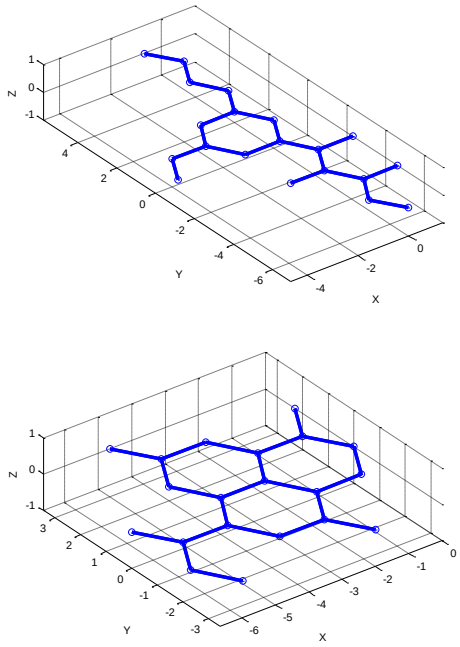
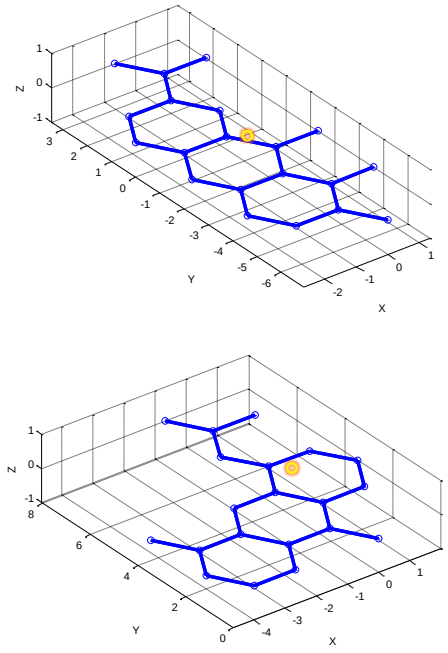


Figure 2: Example of molecules created by four calls of `make_sp2_7ed` *function using total number of atoms equal to 20. Note that all molecules are flat, composed from almost sp^2 hybridized carbon atoms geometry, and have master atom at 0,0,0 coordinates (encircled).*



1.2.1.2. Molecules crossover functions (children generation from parents)

The crossover_gene function performs crossover between the relevant matrices of two genes (i.e. parents). The crossover between atomic descriptor matrices is performed by crossover_mol_2ed function, while crossovers for position and quaternion matrices are performed by crossover_quatern function.

The `crossover_mol(mol1,mol2)` takes two molecular coordinates matrices (parents) as input and give single molecular coordinate matrix (child) as an output. The child is generated from two parent molecules by random selection of atoms from the two parents that not lead to disconnection in the child molecule (Figure 3). Each parent participates at different extent in creating the child gene. The extent of participation is randomly ranged from 25% (minimum) to 75% (maximum) for each parent. The receptor

interaction energy for each atom of the two parents is measured before cross over, then the probability of passing the atoms to child can be optionally related to the energy value. In other word, parents' atoms of good interaction energy have higher probability to be inherited to child if weight value is adjusted in the crossover function to be more than zero. The crossover function takes care of ensuring similar number of heteroatoms in child as in either parents. For the generated child, the atom types of both parents is compared for particular child atom, then priority selection rank is followed. If parents have heteroatoms at particular child atom location, higher priority is given for that child atom to follow either of two parents' atom types while satisfying number of bonds for oxygen (less than 3) and nitrogen (less than 4), more explanation is given in Figure 4.

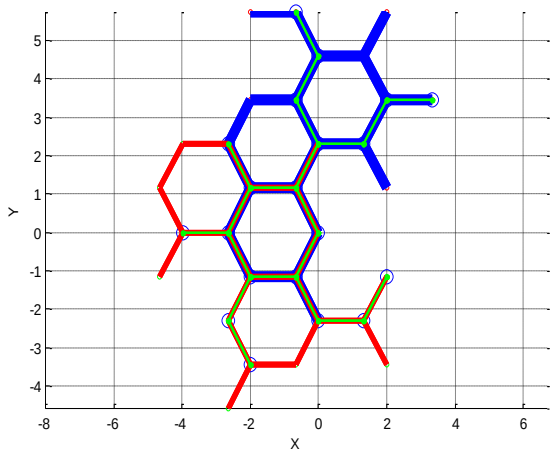
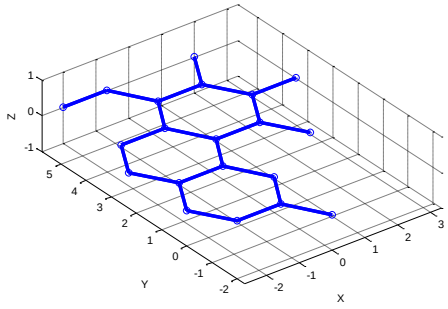
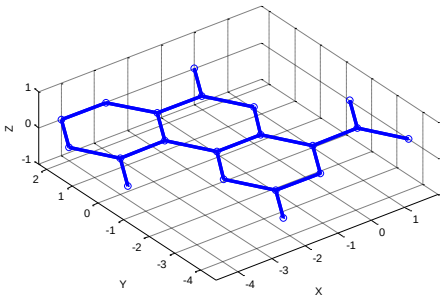


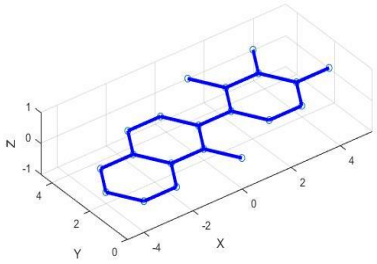
Figure 3: Crossover between Mol1 and Mol2. The `crossover_mol` function randomly connects atoms from Mol1 (blue stick) and Mol2 (red stick) to generate child molecule (green stick).

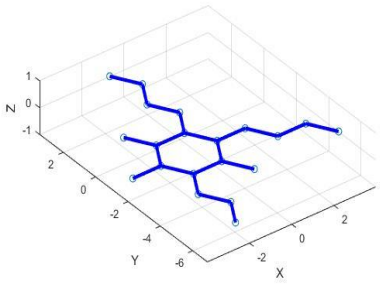


Mol1



Mol2





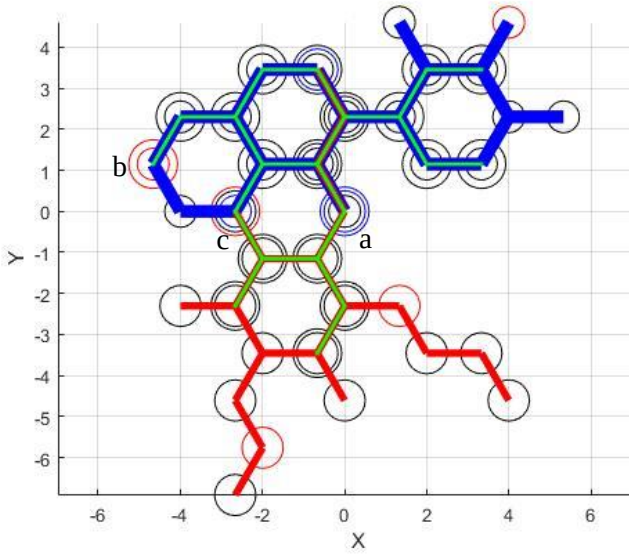


Figure 4: Cross over between two molecules showing the atom of both parents (smaller circles) and child (largest circle) colored by atom types of red for oxygen, blue for nitrogen and black for carbon. The atoms of child inherited from two parents have three co-centered circles, while those inherited from one parent have two co-centered circles. Note atom (a) is inherited from two parents, one of them has nitrogen, so child has nitrogen. For atom (b), it is inherited from one parent which has oxygen, so child's atom also oxygen. For atom (c) it is inherited from two parents where one has carbon and the other has nitrogen, however, since total number of oxygen atoms in child is not satisfied yet, the child atom is considered oxygen as it share one parent in being a heteroatom.

1.2.1.3. Molecules mutation functions (random mutation)

The `mutation_gene4` function accepts a selected group of molecular solutions (genes) for mutation. The mutation for each gene is performed by three functions. The atomic descriptor matrix is mutated by `mutation_mol` function, while position matrix is mutated by `mutation_position` function, and quaternion matrix is mutated by `mutation_quatern` function.

The `mutation_mol_2ed` function first looks for atoms that have less than 3 connections thus able to accept new atomic connection (excluding oxygen atom). This step is equivalent to that performed by `make_sp2` function. The next step involves random removal of an atom (any atom other than the master and the newly added atom), yet, not leads to molecule fragmentation (Figure 5). The type of the newly added atom is adjusted to be similar to the removed atom while avoiding formation of bonds between heteroatoms. The final step includes adjustment of connection table within the atomic descriptor matrix. The extent of mutation can be optionally adjusted by setting the number of mutation cycles to be performed on each molecule.

The `mutation_position` function mutates the XYZ coordinates by three sequenced random steps. The first step selects randomly a number of dimensions to be mutated (mutation range) from the three available dimensions. The second step selects randomly new dimensions to satisfy the mutation range (i.e. dimension(s) to be mutated). For each selected dimension, the third step choses a random coordinate between upper and lower bounds of that particular dimension. The difference between random and original coordinates represents the random displacement value for that dimension. Finally, the random displacement value is multiplied by optional mutation weight (0 to 1) before being summed to original coordinate. Therefore, the mutation weight controls the severity of the change produced by the mutation function. The mutation_quatern function mutates the three quaternion angles in a way similar to mutation_position function.

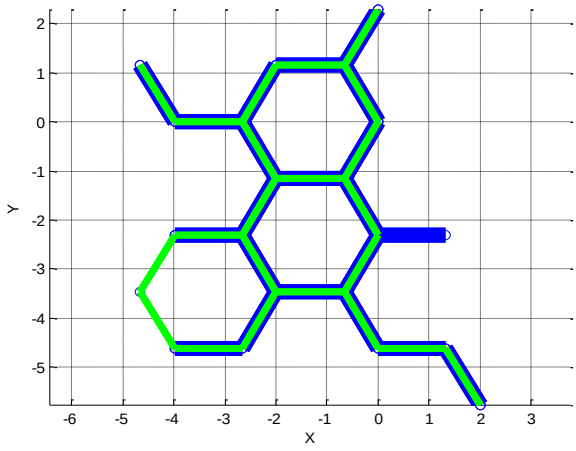


Figure 5: The parent molecule (blue stick) was mutated by `mutation_mol_2ed` function to give mutated molecule (green stick).

1.2.1.4. Molecules selection functions

The selection function needs to accept a list of solution IDs and fitness values to select parents for next generation. Therefore, the internal selectiontournament function in Matlab R2017b was used for selecting molecules to survive for next generation by sole dependence on energy value as provided by the fitness function. A set of 10 molecules are randomly selected, then two molecules of best score are chosen for crossover.

1.2.1.5. Molecules fitness function

The grid map is used to evaluate molecular interaction energy score. It is the grid box generated by Autogrid4 which determines the dimensions of the receptor pocket involved in the *de novo* design. In order to improve computational speed, a vectorized form of fitness function has been used. The fitness_gene_sub_boxes_vectorized (chromosome,C,N,O,H,e,P_C,P_O,P_N,keep_original_charge) function was used to evaluate the fitness for a vector of solutions or genes (a chromosome), by using the carbon (C), nitrogen (N), oxygen (O), hydrogen (H), electrostatic (e) maps from Autogrid4 (Morris et al., 2008). Optionally, the function can accept fixed charge value for carbon (P_C), oxygen (P_O), and nitrogen (P_N) by setting the Boolean "keep_original_charge" to true.

The fitness_gene_sub_boxes_vectorized function depends internally on two functions. The first function is translate_mol_gene which translates the gene into molecular phenotype. While the second function is eval_xyz_sub_surfaces which calculate the protein interaction energy for the particular molecular phenotype according to grid maps.

The eval_xyz_sub_surfaces(mol,Atom_Type,C,O,N,H,e, P_C,P_O,P_N,keep_original_charge)function uses the atomic interaction maps of Autogrid4 to evaluate the Van der Waals interaction energy of molecular phenotype. If "keep_original_charge" Boolean is set to false, Obabel (OpenEye) is used to calculate the partial atomic Gasteiger charges for each atom in the molecular phenotype. The partial atomic charges were then multiplied by the values from electrostatic interaction map (e) to give the electrostatic interaction energy of the molecular phenotype. The final energy equals to the summation of Van der Waals and electrostatic interaction energies. It is important to note that the final energy score includes only Inter- and no Intra- molecular interaction energies

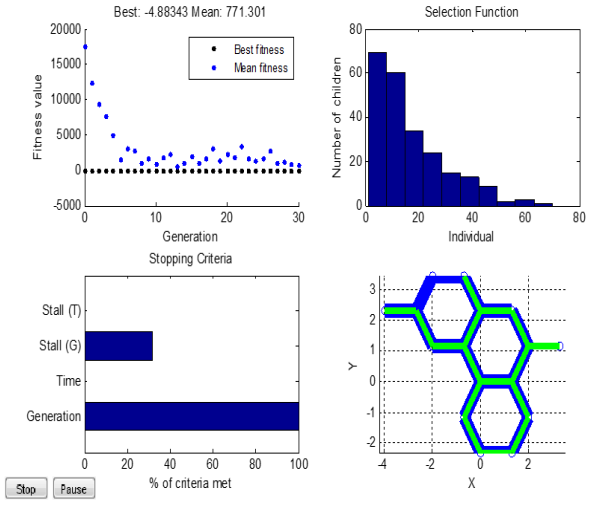
In order to reduce memory demands, the get_map_point function has been used to read the interaction energy values directly from the .map file of Autogrid4. The eight neighboring grid points were read and used to calculate the value at the desired point coordinates using trilinear interpolation routine.

1.2.1.6. Main controller function

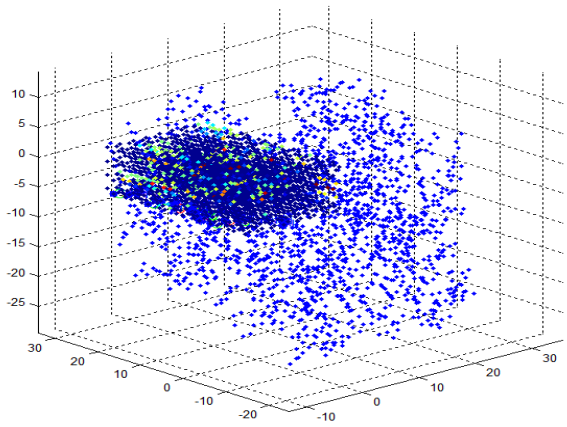
Due to large amount of data encoded in each molecular gene, premature convergence may occur upon increasing number of atoms and/or grid box dimensions. Similar outcome may result from using smaller population and/or generation numbers in GA. Therefore, ga_make function which controls the whole processes has been designed to split the genetic algorithm job into desired number of sub-jobs. Each sub-job performs full GA molecular design within defined sub-box of the full grid box.

Accordingly, the ga_make2 function takes a vector of three numbers representing number of divisions required for each of X, Y, and Z edges of the grid box. For example, num_sub_boxes can be assigned as [2,1,2] to split the grid box into four sub-boxes by equally divides X and Z edges. The ga_make function returns translated molecular phenotype (mol) and the associated protein interaction score (score) i.e. [mol,score]= ga_make_2(num_atoms, num_sub_boxes,C_map,O_map,N_map,H_map,e_ma

p. In order to simplify SAAD for users, a configuration file is prepared using text editor which is passed ga_maker function. The graphical interface for SAAD is shown in **Error! Reference source not found.**a.



a)



b)

Figure 8: a) The graphical interface of SAAD program showing the progress of molecular evolution.

As shown, the fitness value (binding energy) is reduced to lower value within the first few generations

due to progression in molecular structure optimization. b) An atomic representation of PfGST (light blue dots) with grid box (dark blue dots) covers the binding pocket where color of the dot encodes the electrostatic potential for probe atom of charge=+1.

1.1.2. Application of SAAD for PfGST inhibitor design

The SAAD program has been applied to design PfGST inhibitor. The receptor protein includes only chain A of 2AAW (PDB ID). The grid box has been created by AUTODOCK tools to have dimensions of 22, 20, and 26 grid points for X, Y, and Z edges, respectively, and grid spacing of 0.375 Å. The grid box covered the γ-glutamyl and cysteinyl moieties of GSH. The Autogrid4 was used to generate grid map files. The atom types in grid parameter file was set to C for aliphatic carbon, OS, for oxygen, NS for nitrogen, and HD for hydrogen. The "S" prefix to atom type indicates spherical acceptor of hydrogen bonds, while "D" prefix indicates hydrogen bond donator.

The parameters used for SAAD program were as follow: number of atoms was 10, percentages of each of O and N atoms were 10, mutation rate was 0.2, population size was 100, number of generations was 30, number of elites to survive for next generation was 30, and fraction of next generation to be generated by crossover was 0.8.

1.3 Results and discussion

1.3.1 Development and application of SAAD for PfGST inhibitor design

In order to better suit the binding site requisites, molecules can be assembled at atomic rather than fragment levels. Of the available software that use atom-based approach for inhibitor design, GenStar can generate molecules absolutely composed from sp3 hybridized carbon atoms (Rotstein andMurcko, 1993) . However, Genstar only uses carbon atoms to build molecules, which was reasoned by avoiding search space explosion if heteroatoms are included. SAAD

program was developed to generate flat molecules composed absolutely from sp2 hybridized atoms geometry while uses mixed carbon, oxygen and nitrogen heteroatoms to fulfill the mixed polar characteristic of active site.

In Glutathione-S-transferases (GSTs) including PfGST, the active site within the thioredoxin domain which recognizes GSH (G site) and particularly γ-glutamyl moiety binding site is conserved and less prone to mutation across GSTs isoforms compared to substrate binding site (H-site) (Armstrong, 1997). Without prior knowledge regarding natural ligand, SAAD was able to generate molecule of structure close to that of γ-glutamyl moiety of GSH with respect to oxygen and nitrogen atom distribution within the carbon backbone (Figure 9). The carbon skeleton of generated molecule overlapped with that of the moiety. Two oxygen atoms were present in the generated molecule and occupy almost equivalent positions to the carboxylic oxygen atoms of the moiety. Similarly, single nitrogen atom was present at position parallel to that of α amino group of the moiety.

Generally,increasing number of population size most probably lead to molecules of higher of affinity using constant number of carbons and heteroatoms. Using SAAD with grid box covering binding sites of both γ-glutamyl and cysteinyl moieties of GSH, variable number of carbon atoms, heteroatoms and population size, gives results shown in Figure 10 . Using building blocks of 10 carbon atoms, one oxygen atom and 1 nitrogen atoms, SAAD was able to generate heterocyclic molecules which fits the sub-pocket of GSH with energy score ranging from -3 to -7 kcal/mol according to population size. Increasing number of heteroatoms as building blocks, provides molecules with better energy scores due to enhanced polar interactions with the pocket. The use of 8 carbon atoms and 2 atoms of each of oxygen and nitrogen atoms generate molecules of energy score from -3 to almost -17kcal/mol. Population size larger than 200 usually provides better results in SAAD.

The number of variables to be optimized during the design of a molecule composed from 12 atoms equals to 66 variables. Each atom associates with five variables (the X, Y, and Z coordinates, as well as the atomic type and charge). Moreover, the whole molecule associates with three coordinates for center of geometry and three angles for quaternion orientation. Population for the first generation is generated from random assembly of these 66 variables, then it is evolved by following selection according to fitness. In order to simplify the optimization problem and circumvent the convergence problem, a single GA job can be divided into many sub-jobs (Figure 11). SAAD can divide the three edges of grid box equally to give sub-boxes, subsequently generates a molecule within each sub-box.The sub-boxes approach has an advantage of improving the generated molecules by dividing the search space into smaller sub-spaces where separated molecules composed of less number of atoms can be more efficiently optimized. However, connecting the resultant molecules without disrupting the original binding could be another challenge.

The main advantage of SAAD over fragment-based approach is enhancement of structural diversity of the generated molecule. Although molecules generated by SAAD are absolutely composed from sp 2 geometry, it favorably provides more rigid carbon skeleton with distributed oxygen and nitrogen atoms that satisfy receptor interaction sites. However, these features add challenge to synthetic accessibility of the designed molecules. According to scoring function employed, entropy term and solvation energy is not included. Therefore, the generated molecule may have larger water accessible surface area within the pocket.

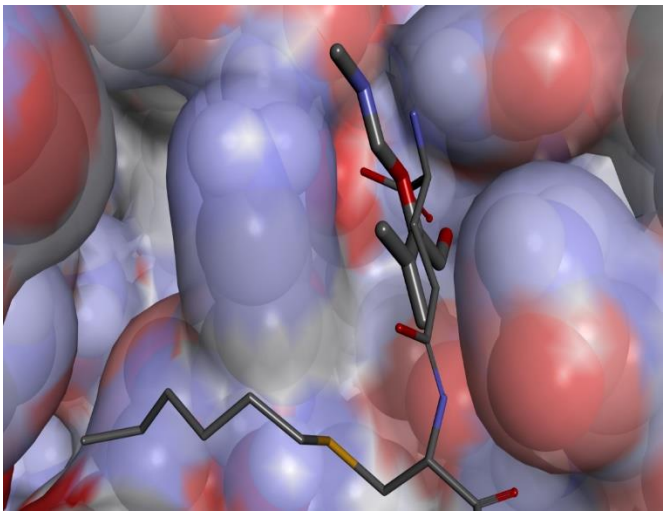
1.4. Conclusion

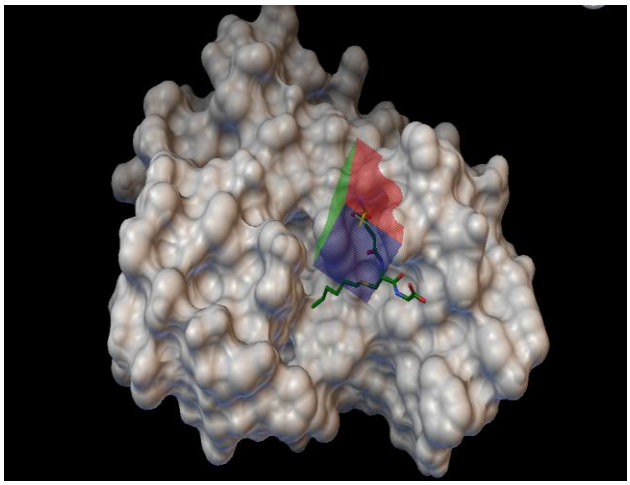
Using the available free access programs (Autodock3.05, OpenEye (toolkit) and Matlab2019) we developed a set of functions (SAAD) to handle the process of atom-based de novo molecular design.

The application of SAAD for pfGST binding pocket provides molecules that can fit the pocket by steric,

electrostatic, and H-bond complementarity using hydrogen, carbon, oxygen and nitrogen atoms only.

Increasing number of polar atoms and/or number of generations within SAAD provides molecule of higher affinity toward the pocket. Although the molecules generated are fully composed of sp2 heavy atoms, the molecules can later be manually adjusted to fulfill synthetic accessibility.Further development are required to connect SAAD to trained artificial neural network in order to guide the atomic connectivity within the generated molecules toward known synthetically accessible active molecules.





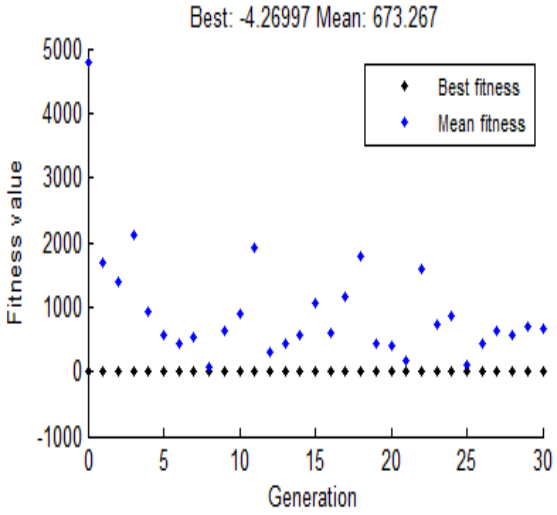


Figure 6: The application of SAAD for designing molecule composed from 10 atoms using grid box at γ-glutamyl part of GSH (a). Through 30 generations, the algorithm ended with molecules of -4.3 kcal/mol (b). The generated molecule (thick stick) composed from atoms where types and coordinates are relative to γ-glutamyl moiety of GSH (thin stick) (c).

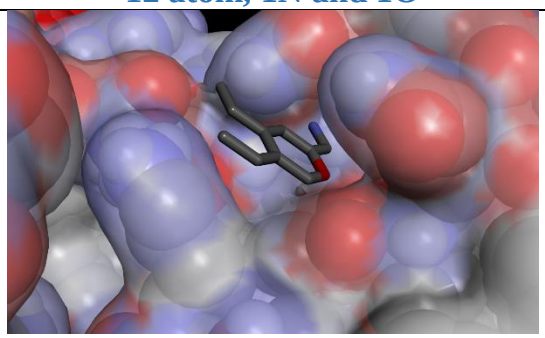
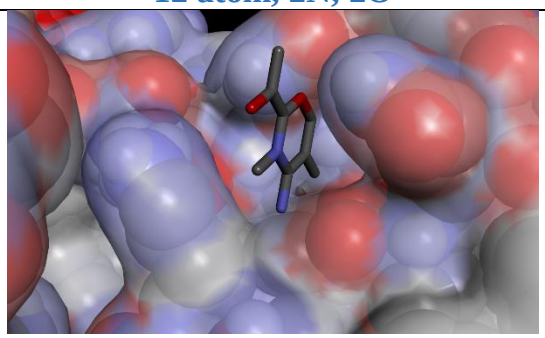
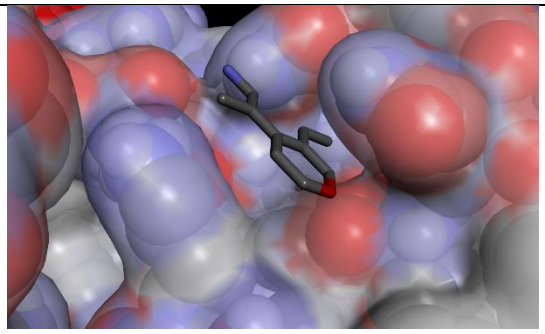
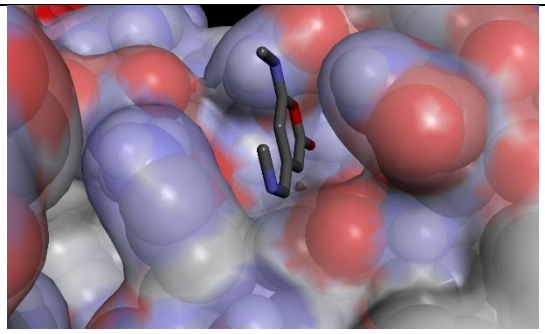
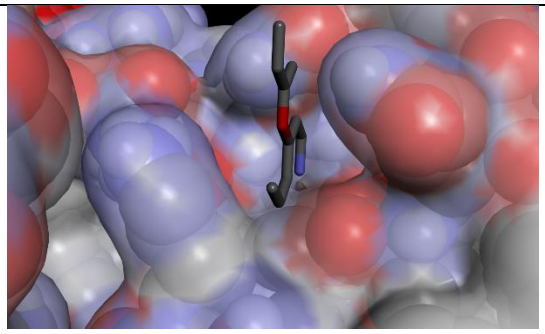
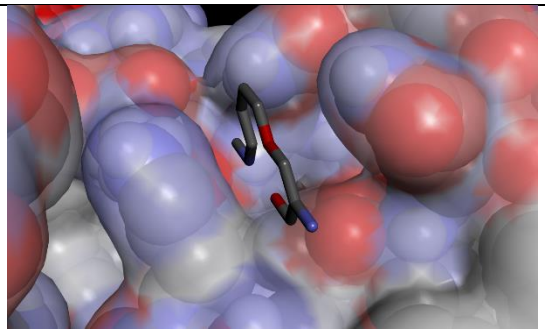
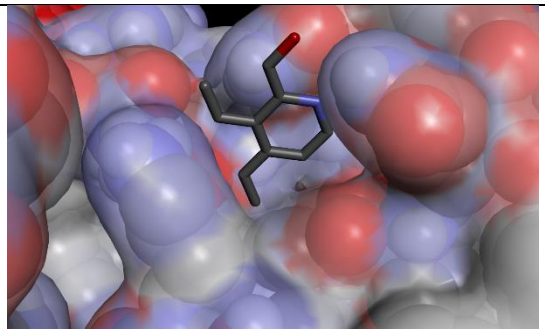
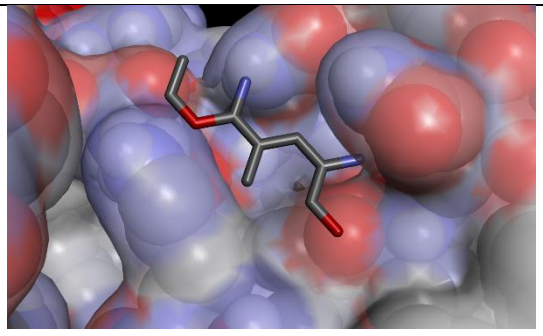
P 0	12 atom, 1N and 1O	12 atom, 2N, 2O
50	 -3.25 Kcal/mol	 -3.14 Kcal/mol
100	 -3.42 Kcal/mol	 -3.37 Kcal/mol
150	 -3.53 Kcal/mol	 -5.26 Kcal/mol
200	 -7.10 Kcal/mol	 -16.83 Kcal/mol

Figure 7: The effect of population size on the SAAD outcome upon fixing other parameters as Generations=30, Mutation rate=0.2, Crossover fraction=0.8, Crossover E weight=0, Survived elites=30, O atoms =2, and percentage of N atoms=2. Increasing the population size usually lead to molecules of better interaction energy of similar atomic composition.

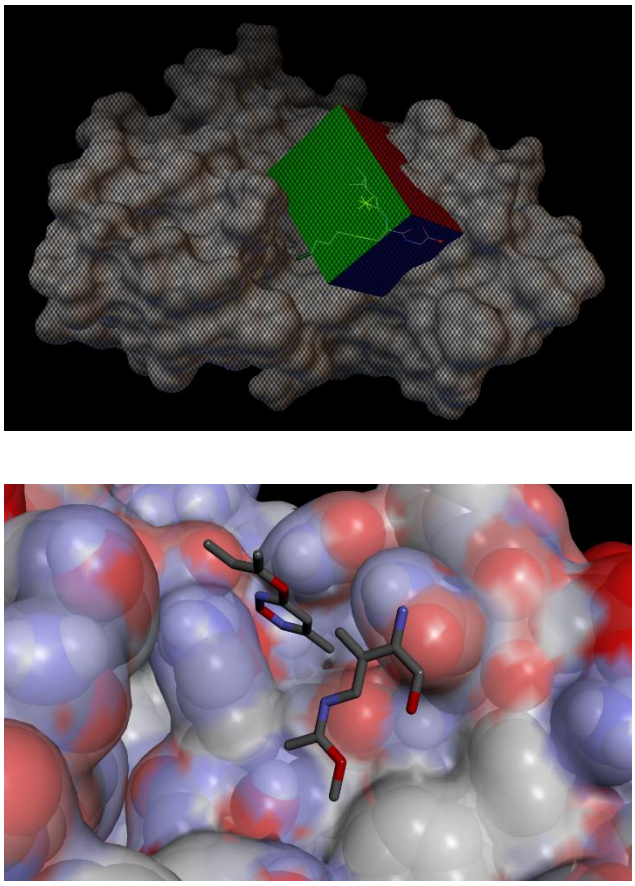


Figure 8: The application of grid sub-boxes approach in SAAD. The binding site of GSX is a) encompassed by a grid box (X, Y and Z dimensions are colored red, green and blue, respectively) which was computationally divided into two sub-boxes through the Z dimension during molecular design process in order to generate b) two molecules where each is composed of 12 atoms including 2 oxygen and 2 nitrogen atoms. The upper molecule has -4 kcal/mol and the lower has -19 kal/mol.

1.5. Reference

ARMSTRONG, R. N. 1997. Structure, catalytic mechanism, and evolution of the glutathione transferases. *Chem Res Toxicol,* **10,** 2-18.

CHU, Y. & HE, X. 2019. Molegear: A java-based platform for evolutionary de novo molecular design. *Molecules,* **24,** 1444.

DEVI, R. V., SATHYA, S. S. & COUMAR, M. S. 2014. Gamol: Genetic algorithm based de novo molecule generator.

HERRING, R. H. & EDEN, M. R. 2015. Evolutionary algorithm for de novo molecular design with multi-dimensional constraints. *Computers & Chemical Engineering,* **83,** 267-277.

KUSNER, M. J., PAIGE, B. & HERNÁNDEZ-LOBATO, J. M. 2017. Grammar variational autoencoder. *In:* DOINA, P. & YEE WHYE, T. (eds.) *Proceedings of the 34th International Conference on Machine Learning*. Proceedings of Machine Learning Research: PMLR.

LEGUY, J., CAUCHY, T., GLAVATSKIKH, M., DUVAL, B. & DA MOTA, B. 2020. Evomol: A flexible and interpretable evolutionary algorithm for unbiased de novo molecular generation. *Journal of Cheminformatics,* **12,** 55.

MEYERS, J., FABIAN, B. & BROWN, N. 2021. De novo molecular design and generative models. *Drug Discovery Today*.

MORRIS, G. M., HUEY, R. & OLSON, A. J. 2008. Using autodock for ligand-receptor docking. *Current Protocols in Bioinformatics,* **24,** 8.14.1-8.14.40.

OLIVECRONA, M., BLASCHKE, T., ENGKVIST, O. & CHEN, H. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics,* **9,** 48.

PEGG, S. C., HARESCO, J. J. & KUNTZ, I. D. 2001. A genetic algorithm for structure-based de novo design. *Journal of computer-aided molecular design,* **15,** 911-33.

ROTSTEIN, S. & MURCKO, M. 1993. Genstar: A method for de novo drug design. *J. Comput. Aided Mol. Des.,* **7,** 23-43.

SCHNEIDER, G. 2013. *De novo molecular design*, John Wiley & Sons.

SEGLER, M. H. S., KOGEJ, T., TYRCHAN, C. & WALLER, M. P. 2018. Generating focused molecule libraries for drug discovery with

recurrent neural networks. *ACS Central Science*, **4,** 120-131.

SPIEGEL, J. O. & DURRANT, J. D. 2020. Autogrow4: An open-source genetic algorithm for de novo drug design and lead optimization. *Journal of Cheminformatics*, **12,** 25.

YOSHIKAWA, N., TERAYAMA, K., SUMITA, M., HOMMA, T., OONO, K. & TSUDA, K. 2018. Population-based de novo molecule generation, using grammatical evolution. *Chemistry Letters*, **47,** 1431-1434.