Supplementary Information

Structure Prediction of Cyclic Peptides by Molecular Dynamics + Machine Learning

Jiayuan Miao, Marc L. Descoteaux, and Yu-Shan Lin*

Department of Chemistry, Tufts University, Medford, Massachusetts, 02155, United States

*Email: yu-shan.lin@tufts.edu

1. Molecular dynamics (MD) simulations

Two parallel bias-exchange metadynamics (BE-META) simulations starting from two different initial structures were performed for each cyclic peptide. The two initial structures were prepared using the UCSF Chimera package,[1] and the backbone RMSD between the two structures was ensured to be larger than 1.3 Å. The initial structure was solvated in a water box. The minimum distance between the atoms of the peptide and the walls of the box was 1.0 nm. Counter ions were added to neutralize the total charge of the system. Energy minimization was then performed on the solvated system using the steepest descent algorithm to remove bad contacts. The solvated system underwent two stages of equilibrations. In the first stage, the solvent molecules were equilibrated while restraining the heavy atoms of the cyclic peptide using a harmonic potential with a force constant of 1,000 $\text{kJ}\cdot\text{mol}^{-1}\cdot\text{nm}^{-2}$. This stage of equilibration consisted of a 50-ps simulation at 300 K in an NVT ensemble and a following 50-ps simulation at 300 K and 1 bar in an NPT ensemble. The second stage of equilibration was performed without restraints and consisted of a 100-ps simulation at 300 K in an NVT ensemble, followed by a 100-ps simulation at 300 K and 1 bar in an NPT ensemble. The production simulations were performed at 300 K and 1 bar in an NPT ensemble. The equations of motion were integrated by the leapfrog algorithm with a time step of 2 fs. Bonds involving hydrogen were constrained with the LINCS algorithm. Electrostatic interactions, van der Waals interactions, and neighbor searching were truncated at 1.0 nm. Long-range electrostatics were treated using the particle mesh Ewald method with a Fourier grid spacing of 0.12 nm and an order of 4. A long-range dispersion correction for energy and pressure was applied to account for the 1.0 nm cut-off of the Lennard-Jones interactions. Five extra improper dihedrals related to the H, N, C, O atoms of the peptide bonds were applied to suppress the formation of *cis* bonds. It was ensured the data used in the analysis were free of *cis* peptide bonds.

BE-META simulations were performed using GROMACS 2018.6[2] patched by PLUMED 2.5.1 plugin.[3] In each BE-META simulation, there were 10 biased replicas, with five biasing the 2D collective variables (ϕ_i, ψ_i) and five biasing the 2D collective variables (ψ_i, ϕ_{i+1}). These collective variables were chosen according to the observation that cyclic peptides usually switch conformations through coupled changes of two dihedrals involving (ϕ_i, ψ_i) or (ψ_i, ϕ_{i+1}).[4] In addition, five neutral replicas (i.e., replicas with no bias) were used to obtain the unbiased structural ensemble for later analysis. Dihedral principal component analysis was used to analyze the trajectories. Normalized integrated product (NIP)[5] between the two parallel simulations of each cyclic peptide was calculated in the 3D space spanned by the top three principal components to monitor the convergence of the simulations. The lengths of the BE-META simulations were 100 ns for most of the cyclic peptides and were extended for some peptides until the NIPs were larger than 0.9 (an NIP value of 1.0 would suggest perfect similarity). Trajectories in the last 50 ns of the neutral replicas of both parallel simulations were combined for each cyclic peptide and used for further structural analysis.

2. The StrEAMM models

2.1 StrEAMM (1,2)/sys: Optimizing (1,2) interaction weights to predict populations of cyclic peptide structures

In this version of StrEAMM models, we considered how the interactions between the nearest neighbors, i.e., the (1,2) interactions, impact the structural preferences of a cyclic peptide, as the first-order approximation. The population of cyclo-($X_1X_2X_3X_4X_5$) adopting a certain structure $S_1S_2S_3S_4S_5$, $p_{S_1S_2S_3S_4S_5}^{X_1X_2X_3X_4X_5}$, was related to these (1,2) interactions as:

$$p_{S_1S_2S_3S_4S_5}^{X_1X_2X_3X_4X_5} \propto \exp\left(w_{S_1S_2}^{X_1X_2} + w_{S_2S_3}^{X_2X_3} + w_{S_3S_4}^{X_3X_4} + w_{S_4S_5}^{X_4X_5} + w_{S_5S_1}^{X_5X_1}\right), \quad (1)$$

where $w_{S_iS_{i+1}}^{X_iX_{i+1}}$ was the weight assigned to a sequential 2-residue section of the cyclic peptides when residues X_iX_{i+1} adopted structure S_iS_{i+1}, X_i was one of the 15 amino acids (G, A, V, F, N, S, D, R, a, v, f, n, s, d, and r; lowercase letters denote D-amino acids), and S_i was one of the 10 structural digits (Β, Π, Γ, Λ, Ζ, β, π, γ, λ, and ζ). The expression is illustrated in **Fig. 2b**. The weights were presumed additive, sharing a similar property with energies. Since energies appear in the exponential of Boltzmann factors when related to populations, an exponential operation was also introduced here to relate the sum of the five weights to the predicted population. The operation also helped prevent the predicted populations from adopting values <0.

To obtain the exact population of cyclo-($X_1X_2X_3X_4X_5$) adopting a certain structure $S_1S_2S_3S_4S_5$, the partition function (Q) needed to be considered:

$$Q = \sum_{\substack{\text{all} \\ \text{structures}}} \exp\left(w_{S_1S_2}^{X_1X_2} + w_{S_2S_3}^{X_2X_3} + w_{S_3S_4}^{X_3X_4} + w_{S_4S_5}^{X_4X_5} + w_{S_5S_1}^{X_5X_1}\right). \quad (2)$$

$$p_{S_1S_2S_3S_4S_5}^{X_1X_2X_3X_4X_5} = \exp\left(w_{S_1S_2}^{X_1X_2} + w_{S_2S_3}^{X_2X_3} + w_{S_3S_4}^{X_3X_4} + w_{S_4S_5}^{X_4X_5} + w_{S_5S_1}^{X_5X_1}\right)/Q, \quad (3)$$

which could also be written as:

$$\ln\left(p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}}\right) = w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} - \ln Q. \quad (4)$$

However, Eq. (2) broke the linearity of Eq. (4), making it difficult to reach convergence when solving a set of Eq. (4)'s. Hence, we introduced another independent weight for each cyclic peptide in the training set:

$$w_Q = -\ln Q \quad (5)$$

and

$$\ln\left(p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}}\right) = w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} + w_Q. \quad (6)$$

Each structure of each cyclic peptide in the training set contributed an Eq. (6). Together, these equations formed a nonhomogeneous linear equation group, which could be rewritten in the matrix format:

$$\ln \boldsymbol{p} = \boldsymbol{A}\boldsymbol{w}. \quad (7)$$

The logarithms of populations were arranged into an $N\times 1$ column vector, where N was the summation of the number of structure types of each cyclic peptide in the training set. Different weights were arranged into an $M\times 1$ column vector, where M was the number of weights. The coefficient matrix $\boldsymbol{A}$ controlled which weights were used to compute the population of a specific cyclic-peptide sequence adopting a specific structure. See **Fig. S1** for detailed illustration of the matrix. The weights were determined by weighted least square fitting, i.e., by minimizing the following loss function with respect to the weights $\boldsymbol{w}$:

$$L(\boldsymbol{w}) = \sum_{i=1}^{N} p_i \left|\sum_{j=1}^{M} A_{ij} w_j - \ln p_i\right|^2 \quad (8)$$

To predict the populations of a new cyclic peptide, Eq. (3) was used, with partition function Q calculated by Eq. (2). In theory, Eq. (2) required exhaustively counting the contributions of all possible structures. In practice, we only accounted for structures that had a population larger than 0.1% (500 frames) in at least one of the cyclic peptides in the training set (Datasets 1–3; see "Datasets" in the Methods section). See **List S1** for the resulting structure pool that included 550 structures. Due to the incompleteness of the structure pool, we introduced a compensation factor f when computing Q. To estimate f, we computed the sum of the populations of these 550 structures for each cyclic peptide in the training set. The mean value of these summations was 0.996 and was used as the compensation factor f. The partition function used was:

$$Q = \sum_{\substack{\text{structures}\\ \text{in the pool}}} \exp\left(w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}}\right)/f. \quad (9)$$

The predicted population was then calculated using Eq. (3) with partition function calculated using Eq. (9).

When calculating the populations using Eq. (3) for a new cyclic peptide, it was possible to encounter some weights that did not exist in the training set. The absence of these weights in the training set suggested the amino acid sequences had little tendency to adopt the corresponding structures, and these weights were thus assigned to a very negative number (–20 was used, which was small enough to bring the final predicted population to essentially zero).

The dataset used in the training for **StrEAMM (1,2)/sys** was Dataset 1 (see "Datasets" in the Methods section). The matrix equation (7) contained 131,779 linear equations and 6,101 independent weights; weights that were mirror images of each other were treated as one independent weight because $w_{\mathrm{S}_i\mathrm{S}_{i+1}}^{\mathrm{X}_i\mathrm{X}_{i+1}} = w_{\mathrm{s}_i\mathrm{s}_{i+1}}^{\mathrm{x}_i\mathrm{x}_{i+1}}$ with capital and lowercase letter pairs representing enantiomers of amino acids and structures. The distribution of the weights is shown in **Fig. S3**.

2.2 **StrEAMM (1,2)+(1,3)/sys** and **StrEAMM (1,2)+(1,3)/random**: Including both (1,2) and (1,3) interaction weights

In **StrEAMM (1,2)+(1,3)/sys** and **StrEAMM (1,2)+(1,3)/random**, we considered interactions between the nearest neighbors and between next-nearest neighbors, i.e., both (1,2) interactions and (1,3) interactions. The population of cyclo-($\mathrm{X_1X_2X_3X_4X_5}$) adopting a certain structure $\mathrm{S_1S_2S_3S_4S_5}$, $p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}}$, was related to the (1,2) and (1,3) interactions as:

$$p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}} \propto \exp\left(w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} + w_{\mathrm{S_1S_2S_3}}^{\mathrm{X_1\_X_3}} + w_{\mathrm{S_2S_3S_4}}^{\mathrm{X_2\_X_4}} + w_{\mathrm{S_3S_4S_5}}^{\mathrm{X_3\_X_5}} + w_{\mathrm{S_4S_5S_1}}^{\mathrm{X_4\_X_1}} + w_{\mathrm{S_5S_1S_2}}^{\mathrm{X_5\_X_2}}\right), \quad (10)$$

where $w_{\mathrm{S}_i\mathrm{S}_{i+1}\mathrm{S}_{i+2}}^{\mathrm{X}_i\_\mathrm{X}_{i+2}}$ was the weight assigned to the interactions between residues X_i and X_{i+2} when residues $\mathrm{X}_i\mathrm{X}_{i+1}\mathrm{X}_{i+2}$ adopted the structure $\mathrm{S}_i\mathrm{S}_{i+1}\mathrm{S}_{i+2}$. Note that while describing (1,3) interactions, we also included the structure of the middle residue, considering that the ϕ and ψ dihedrals of residue X_{i+1} would affect the relative distance and orientation of residues X_i and X_{i+2}. However, the middle residue X_{i+1} can be any amino acid. The expression is illustrated in **Fig. 2c**. Similar to what was done in **StrEAMM (1,2)/sys**, exact populations could be obtained by introducing the partition function Q:

$$p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}} = \exp\left(w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} + w_{\mathrm{S_1S_2S_3}}^{\mathrm{X_1\_X_3}} + w_{\mathrm{S_2S_3S_4}}^{\mathrm{X_2\_X_4}} + w_{\mathrm{S_3S_4S_5}}^{\mathrm{X_3\_X_5}} + w_{\mathrm{S_4S_5S_1}}^{\mathrm{X_4\_X_1}} + w_{\mathrm{S_5S_1S_2}}^{\mathrm{X_5\_X_2}}\right)/Q, \quad (11)$$

and

$$Q = \sum_{\substack{\text{structures}\\ \text{in the pool}}} \exp\left(w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} + w_{\mathrm{S_1S_2S_3}}^{\mathrm{X_1\_X_3}} + w_{\mathrm{S_2S_3S_4}}^{\mathrm{X_2\_X_4}} + w_{\mathrm{S_3S_4S_5}}^{\mathrm{X_3\_X_5}} + w_{\mathrm{S_4S_5S_1}}^{\mathrm{X_4\_X_1}} + w_{\mathrm{S_5S_1S_2}}^{\mathrm{X_5\_X_2}}\right)/f \quad (12)$$

with f being the compensation factor to account for the incompleteness of the structure pool. Again, we applied Eq. (5) when fitting for the weights with the following linear equation:

$$\ln\left(p_{\mathrm{S_1S_2S_3S_4S_5}}^{\mathrm{X_1X_2X_3X_4X_5}}\right) = w_{\mathrm{S_1S_2}}^{\mathrm{X_1X_2}} + w_{\mathrm{S_2S_3}}^{\mathrm{X_2X_3}} + w_{\mathrm{S_3S_4}}^{\mathrm{X_3X_4}} + w_{\mathrm{S_4S_5}}^{\mathrm{X_4X_5}} + w_{\mathrm{S_5S_1}}^{\mathrm{X_5X_1}} + w_{\mathrm{S_1S_2S_3}}^{\mathrm{X_1\_X_3}} + w_{\mathrm{S_2S_3S_4}}^{\mathrm{X_2\_X_4}} + w_{\mathrm{S_3S_4S_5}}^{\mathrm{X_3\_X_5}} + w_{\mathrm{S_4S_5S_1}}^{\mathrm{X_4\_X_1}} + w_{\mathrm{S_5S_1S_2}}^{\mathrm{X_5\_X_2}} + w_Q. \quad (13)$$

Each structure of each cyclic peptide in the training set contributed an Eq. (13). Together, these equations formed a matrix equation (7). The optimized weights were obtained by minimizing the loss function (8). The predicted population of a new cyclic peptide adopting a specific structure was calculated by Eq. (11) with Q calculated via Eq. (12).

We used the SciPy package of Python language to build the matrix and calculate the weights. The loss function in Eq. (8) was minimized by the scipy.sparse.linalg.lsqr function of the package.

2.2.1 ***StrEAMM (1,2)+(1,3)/sys****: Training with Dataset 2 (see "Datasets" in the Methods section)*

The matrix equation (7) contained 251,120 linear equations and 34,100 independent weights, including 6,123 (1,2) interaction weights and 27,977 (1,3) interaction weights. The distributions of the weights are shown in **Fig. S5**.

2.2.2 ***StrEAMM (1,2)+(1,3)/random****: Training with Dataset 3 (see "Datasets" in the Methods section)*

The matrix equation (7) contained 465,728 linear equations and 44,439 independent weights, including 7,626 (1,2) interaction weights and 36,813 (1,3) interaction weights. The distributions of weights related to (1,2) interactions and (1,3) interactions are shown in **Fig. S7**. To avoid large errors in the weight estimates, if a weight occurred fewer than 10 times in the training set, it was assigned a very negative number (–20 was used, which was small enough to bring the final predicted population to essentially zero) when calculating a population.

2.2.3 ***StrEAMM (1,2)+(1,3)/sys37****: Training with Dataset 5*

Dataset 5 was an extension of Dataset 2 by including the basic amino acids in L- or D-configurations except Pro (37 amino acids total). The reason we excluded Pro is that it increases the likelihood of observing a *cis* peptide bond, and we believe the current force fields are not trained to and are unable to predict *cis*/*trans* configurations correctly. The new training dataset (Dataset 5) included 1,315 systematic sequences: Cyclo-(GGGGG), cyclo-($\mathtt{X_1GGGG}$), cyclo-($\mathtt{X_1X_2GGG}$), cyclo-($\mathtt{X_1x_2GGG}$), cyclo-($\mathtt{X_1GX_2GG}$), and cyclo-($\mathtt{X_1Gx_2GG}$), with $\mathtt{X}_i$ being one of the 18 L-amino acids and $\mathtt{x}_i$ being one of the 18 D-amino acids. Each sequence contained one unique nearest-neighbor or next-nearest-neighbor pair with the rest of the sequence filled by Gly's. Again, the enantiomers of these cyclic peptides were not simulated, and their structural ensembles were inferred from the 1,314 simulated cyclic peptides.

A new test dataset of 75 sequences that contained the 37 types of amino acids was built (Dataset 6, **List S4**). The performance of the model is shown in **Fig. S9**.

2.3 Extendibility of StrEAMM: Graph neural networks (GNNs) and amino-acid fingerprints

More advanced neural networks and amino-acid representations can be introduced to the StrEAMM model. Here, we provide such an example and show the extendibility of the model. In this example, we trained a GNN (message passing network) to predict structural ensembles of cyclic pentapeptides while encoding the peptides as a graph.[6] GNNs have recently been applied to chemical systems due to their potential to handle inputs of diverse graph structures. In general, molecular systems naturally lend themselves to graph representations.[7]

Neural network training and graph creation were done using Pytorch 1.9.0[8] and Pytorch Geometric 1.7.2.[9] Amino acids were encoded using circular topological molecular fingerprints, specifically the Morgan Fingerprints[10] generated with RDKit version 2021.03.05,[11] using a radius of three and a fingerprint length of 2,048 bits; amino acids were input with NH_2 and COOH termini, and sidechain charges matched the charges used in the MD simulations. We chose this encoding because recent work has found that neural network models trained with amino acids as circular fingerprint encodings[12] were able to predict the properties of miniproteins composed of amino acids not found in the training dataset.[13] With this encoding, every amino acid in a cyclic-peptide sequence can be represented by a 2,048-bit fingerprint. To represent the structural ensemble of a cyclic peptide, we used an array of 2,742 populations where each population in the array corresponded to a structure or a cyclic permutation of a structure in the structure pool (**List S1**). We note that there are fewer than 2,750 (550×5) populations because "ΛΛΛΛΛ" and "λλλλλ" in the structure pool are cyclic invariant.

In preparation for the use of a GNN, we represented a cyclic pentapeptide as a graph with one node for each amino acid in the sequence and the initial node representation given by an amino acid's molecular fingerprint. Nodes were connected by four types of directed edges. Two types of edges (forward and backward with respect to peptide sequence) connected (1,2) neighbor nodes, and two types of edges connected (1,3) neighbor nodes. The edges must be directed to prevent a sequence and its retroisomer (reverse ordering sequence) from being encoded as identical graphs. Thus, a cyclic pentapeptide is represented by a graph with 5 nodes and 20 edges.

We constructed a GNN that converted a cyclic-pentapeptide graph into an array of structure populations. The network performed the following sequence of operations. From the input graph, we performed one message passing operation using the RGCNConv operator through Pytorch Geometric.[14] This operator updated a node representation in the graph by summing up the node's transformed initial representation and transformed representations of the node's (1,2) and (1,3) neighbors. Each different edge type had a unique learned transformation. A rectified linear unit (ReLU) activation function was then applied to the node representations. Next, the node representations were concatenated and transformed by one densely connected layer with 2,048 nodes using a ReLU activation function, and an output layer with 2,742 nodes to represent a structural ensemble as an array of 2,742 populations. A SoftMax activation function was used for the second layer to ensure the output structural ensemble was normalized.

The models were trained using the Adam optimizer[15] and summation of the squared errors loss function ($L = \sum_{i=1}^{N}\left(p_{i,\mathrm{learned}} - p_i\right)^2$, where N is the number of populations in the training dataset, $p_{i,\mathrm{learned}}$ is the learned population by the network, and p_i is the actual population observed in MD simulations) for 1,000 epochs with a learning rate of 0.000005 and a batch size of

50. To account for the non-cyclic permutation invariant operation of node concatenation, we trained on all cyclic permutations of a sequence, as well as the corresponding enantiomer sequences, whose data we constructed from the initial simulation results of a sequence by cyclically permuting structural digits or flipping them across the centro-symmetric structural map for the two different cases, respectively. By doing this, we aimed to train the model to be invariant to cyclic permutations of the input sequence. The first model, **StrEAMM GNN/random**, was trained on the semi-randomly generated Dataset 3 containing 15 types of representative amino acids, as well as their cyclically permuted sequences and enantiomer sequences (7,050 input graphs). The second model, **StrEAMM GNN/random37**, was trained on Dataset 3 and 50 additional random sequences containing 37 types of amino acids (Dataset 6.1, **List S5**), as well as their cyclically permuted sequences and enantiomers (7,550 input graphs).

To evaluate the performance of the models, we tested **StrEAMM GNN/random** and **StrEAMM GNN/random37** on the 50 sequences of Dataset 4 that contain 15 types of representative amino acids (**List S2**) and on the 25 sequences of Dataset 6.2 that contain 37 types of amino acids (**List S5**). The results for **StrEAMM GNN/random** and **StrEAMM GNN/random37** are shown in **Figs. S10** and **S11**, respectively.

3. Binning the Ramachandran plot

The Ramachandran plot of cyclo-(GGGGG) was first divided into 100×100 grids, and the probability density of each grid was calculated (**Fig. S12a**). Cluster analysis was only performed on the grids with a probability density larger than 0.00001 (**Fig. S12b**) using a grid-based and density peak-based method.[16] **Fig. S12c** shows the resulting 10 clusters. The centroid of each cluster was determined as the grid point with the smallest average of distances weighted by probability density to the remaining grids of the cluster (**Fig. S12c**, black dots). All the other grid points in the Ramachandran plot were then assigned to their closest centroid (**Fig. S12d**) to obtain the final map. To verify the applicability of the binning map to non-Gly residues, **Fig. S13** shows the Ramachandran plot of the first residue in cyclo-(GGGGG), cyclo-(AGGGG), cyclo-(VGGGG), cyclo-(FGGGG), cyclo-(NGGGG), cyclo-(SGGGG), cyclo-(RGGGG), and cyclo-(DGGGG), with the boundaries of the map shown. We see that in general, the binning map is capable of separating the major peaks in these Ramachandran plots as well.

List S1. The structure pool used in the analysis. The pool includes 550 structures (275 enantiomer pairs) whose populations (either one structure or its enantiomer, or both) were larger than 0.1% (500 frames) in at least one of the cyclic peptides in Datasets 1–3.

λΓλλζ ΛγΛΛΖ λπΓλλ ΛΠγΛΛ ΓγΒλΖ γΓβΛζ γΖΛΠΠ Γζλππ γΒζπΒ ΓβΖΠβ γΖΠγΠ ΓζπΓπ λλβΓπ ΛΛΒγΠ ΠλλζΒ
πΛΛΖΒ λΓλΛΓ ΛγΛλγ βΠβΓζ ΒπΒγΖ ΛΓλΛΛ λγΛλλ ΛΓζλλ λγΖΛΛ ΒβΓγΛ βΒγΓλ ΒβΠλγ βΒπΛΓ ΠλΒπΖ πΛβΠζ
ΠπΒβΖ πΠβΒζ πΛΒγΖ ΠλβΓζ πΠζπΒ ΠπΖΠβ πΒζλζ ΠβΖΛΖ ΛΠγΒβ λπΓβΒ ΠλΖΛΒ πΛζλβ βΓγΛλ ΒγΓλΛ πΛΒβΒ
ΠλβΒβ πΒβΠζ ΠβΒπΖ ΖΓγΠπ ζγΓπΠ ζβΓγΓ ΖΒγΓγ πΓΒπΒ ΠγβΠβ λλΛΓζ ΛΛΛγΖ πΛΛβΓ ΠλλΒγ λΓζγΛ ΛγΖΓλ
λλβΛΠ ΛΛΒλπ ΛΓΖΓβ λγζγΒ ΒζπΛΛ βΖΠλλ βΛγΓζ ΒλΓγΖ βΒλλΓ ΒβΛΛγ ζλβΛζ ΖΛΒλΖ ΛΛΒζβ λλβΖΒ γγΒλζ
ΓΓβΛΖ ζλγΛλ ΖΛΓλΛ ΖΓγΓβ ζγΓγΒ γγΒπΠ ΓΓβΠπ Γλλζβ γΛΛΖΒ ΓγΖΠβ γΓζπΒ ΖΓλΓλ ζγΛγΛ λΛΒγΛ ΛλβΓλ
ΓγΓβπ γΓγΒΠ ΠλΖΠβ πΛζπΒ ΠγΓγΛ πΓγΓλ Βγζλβ βΓΖΛΒ ΠλβΠζ πΛΒπΖ ΓλΖΓβ γΛζγΒ βΠπΓζ ΒπΠγΖ ζγΓγΠ
ΖΓγΓπ ΒλζλΖ βΛΖΛζ ΠλζγΒ πΛΖΓβ ΖΓγΠγ ζγΓπΓ λβΛλλ ΛΒλΛΛ πΓβΓβ ΠγΒγΒ λγΛγΠ ΛΓλΓπ ΠπΖΒπ πΠζβΠ
γΛβΠλ ΓλΒπΛ ΛΛΖΛβ λλζλΒ ζλΛΛλ ΖΛΛΛΛ ζλΖΛΒ ΖΛζλβ λΓγΖΠ ΛγΓζπ βΠλγΠ ΒπΛΓπ λλπΓζ ΛΛΠγΖ ζγΒπΒ
ΖΓβΠβ γΛγΖΓ ΓλΓζγ Πζζλβ πΖΖΛΒ ΓπΠλπ γΠπΛΠ γΒλΓλ ΓβΛγΛ ΓγΓππ γΓγΠΠ λγΖΠζ ΛΓζπΖ ΛΠβΛΖ λπΒλζ
ΓβΖΛΛ γΒζλλ λΛΓγΛ ΛλγΓλ γζλλλ ΓΖΛΛΛ λλΛγΒ ΛΛλΓβ ΓβΒγΖ γΒβΓζ λλζβΒ ΛΛΖΒβ λΓλΓλ ΛγΛγΛ ΖΠπΓγ
ζπΠγΓ ΓπΓγΛ γΠγΓλ ΖΛΓγΖ ζλγΓζ Γλλζπ γΛΛΖΠ πΛβΓζ ΠλΒγΖ γΓβΓΓ ΓγΒγγ ΛΛΛΠπ λλλπΠ ΓλζγΖ γΛΖΓζ
ΠγζπΖ πΓΖΠζ γΒζλζ ΓβΖΛΖ λγΓγΛ ΛΓγΓλ λλζγΠ ΛΛΖΓπ ΓλΖΓγ γΛζγΓ πΠζλζ ΠπΖΛΖ γΛΛζλ ΓλλΖΛ λπΠπΓ
ΛΠπΠγ ΖΓΓγΛ ζγγΓλ λζβΛζ ΛΖΒλΖ ΛΓζλΖ λγΖΛζ πΠπΒζ ΠπΠβΖ ΛΛγΒπ λλΓβΠ πΒβΓζ ΠβΒγΖ λγΒλΒ ΛΓβΛβ
λλΒπΒ ΛΛβΠβ λππΛΒ ΛΠΠλβ πΠγΒζ ΠπΓβΖ γΛΒπΒ ΓλβΠβ γΓΖΛΒ Γγζλβ ΒγΒπΖ βΓβΠζ πΛζπΠ ΠλΖΠπ ΛΛΒπΖ
λλβΠζ λΓβΛΒ ΛγΒλβ πΠβΠζ ΠπΒπΖ ΠλβΛβ πΛΒλΒ ΖΛΒγΖ ζλβΓζ γΛζλζ ΓλΖΛΖ ΠγβΓπ πΓΒγΠ λλλλΒ ΛΛΛΛβ
λγΒγζ ΛΓβΓΖ ΠγΖΓπ πΓζγΠ λπΓβΓ ΛΠγΒγ ΠπΖΓβ πΠζγΒ ΓγΛΛγ γΓλλΓ πΓζπΒ ΠγΖΠβ ζπΓγΓ ΖΠγΓγ ΓβΖΓβ
γΒζγΒ λπΛΓζ ΛΠλγΖ ΠζβΛΖ πΖΒλζ ΠλλΒπ πΛΛβΠ λγΒγΠ ΛΓβΓπ λπΓλζ ΛΠγΛΖ ΓζγΛγ γΖΓλΓ ΓλζπΓ γΛΖΠγ
ΛΒζλγ λβΖΛΓ γΓβΓζ ΓγΒγΖ ΠγΛΒγ πΓλβΓ ΛΠγΓβ λπΓγΒ γΠλλλ ΓπΛΛΛ γΖΒλζ ΓζβΛΖ ΓβΠγΛ γΒπΓλ ΖΓλλγ
ζγΛΛΓ γΠβΛΒ ΓπΒλβ ΠγζγΖ πΓΖΓζ λζζγΒ ΛΖΖΓβ λβΛΛΠ ΛΒλλπ ΠγβΛΖ πΓΒλζ ΒλλΓπ βΛΛγΠ γΖΠλλ ΓζπΛΛ
βΠπΠγ ΒπΠπΓ γΖΒλλ ΓζβΛΛ ΛΒζγΖ λβΖΓζ ΛΛΖΓΖ λλζγζ πΓλγΠ ΠγΛΓπ βΓβΓζ ΒγΒγΖ ΛΖΖΛΖ λζζλζ λγζλζ
ΛΓΖΛΖ πΓΖΛΠ Πγζλπ λλλΓλ ΛΛΛγΛ γΛΒλΛ ΓλβΛλ λβΠγΠ ΛΒπΓπ ΓλβΠγ γΛΒπΓ ΛΓλζβ λγΛΖΒ πΠπΠζ ΠπΠπΖ
ΓγΛΛΖ γΓλλζ ΓγΖΓβ γΓζγΒ ΠγΖΛΒ πΓζλβ ΓλγΛλ γΛΓλΛ ΛΓλΓβ λγΛγΒ ΒβΛΛΛ βΒλλλ πΠβΓζ ΠπΒγΖ βΛΓγΒ
ΒλγΓβ ΛΓλζγ λγΛΖΓ γΛβΛΒ ΓλΒλβ ΛΖΒγΖ λζβΓζ ΓγΠγΛ γΓπΓλ λλΖΓζ ΛΛζγΖ ΠλΖΓβ πΛζγΒ βΠλπΓ ΒπΛΠγ
πΠλλζ ΠπΛΛΖ γΠπΠγ ΓπΠπΓ ΠλζπΖ πΛΖΠζ πΓλβΠ ΠγΛΒπ βΒπΠλ ΒβΠπΛ ΒπΛΛΛ βΠλλλ ΓγΓγΖ γΓγΓζ λγΓπΒ
ΛΓγΠβ λβΒπΒ ΛΒβΠβ πΛβΛΒ ΠλΒλβ ΛΓγΛβ λγΓλΒ ΓβΓγΛ γΒγΓλ γΛΒλΓ ΓλβΛγ ΠγβΓβ πΓΒγΒ πΠπΓζ ΠπΠγΖ
ΠγΖΓβ πΓζγΒ πΓβΠζ ΠγΒπΖ ΓγΖΛΖ γΓζλζ ΓλΖΛΛ γΛζλλ πΠπΠλ ΠπΠπΛ λλζπΒ ΛΛΖΠβ γΓβΓλ ΓγΒγΛ πΠπΒλ
ΠπΠβΛ γΠπΒλ ΓπΠβΛ ΒγΛγΛ βΓλΓλ λγΖΓζ ΛΓζγΖ ΛΖΛΓπ λζλγΠ λλλλλ ΛΛΛΛΛ λζπΛΓ ΛΖΠλγ ΠγΖΛΖ πΓζλζ
ΓλγΛγ γΛΓλΓ πΛΒλβ ΠλβΛΒ ΓπΛΒγ γΠλβΓ βΓλλλ ΒγΛΛΛ ΓγΛγΛ γΓλΓλ γΛΓζλ ΓλγΖΛ ΓγΠπΓ γΓπΠγ λλβΛζ
ΛΛΒλΖ ΓγΒπΓ γΓβΠγ λλγΛΓ ΛΛΓλγ ΠλζγΖ πΛΖΓζ ΛΛγΓπ λλΓγΠ ΛΓβΠβ λγΒπΒ πΛΛγΠ ΠλλΓπ ΠβΒλβ πΒβΛΒ
ΠλΖΛΖ πΛζλζ ΓβΠπΛ γΒπΠλ πΓβΓζ ΠγΒγΖ γΛΒλζ ΓλβΛΖ λλΒλζ ΛΛβΛΖ ΒλγΛΛ βΛΓλλ ΠλβΠβ πΛΒπΒ ζλλγΓ
ΖΛΛΓγ λβΒγΒ ΛΒβΓβ ΓλβΓβ γΛΒγΒ ΛΛβΓβ λλΒγΒ ΛΒβΛΖ λβΒλζ πΓβΛΠ ΠγΒλπ ΛΒγΓγ λβΓγΓ ΛΛΖΛΛ λλζλλ

πΓβΛΓ ΠγΒλγ ΠπΛΓπ πΠλγΠ γΒβΛΒ ΓβΒλβ ΓπΛΓγ γΠλγΓ λλλγΒ ΛΛΛΓβ λβΖΛΒ ΛΒζλβ Γλζλβ γΛΖΛΒ λλλγΓ
ΛΛΛΓγ λλγΛΒ ΛΛΓλβ ΠπΛΒγ πΠλβΓ ΛΛζλβ λλΖΛΒ ΓγΓγΛ γΓγΓλ Γλζλλ γΛΖΛΛ λλβΓζ ΛΛΒγΖ Πγζλβ πΓΖΛΒ
λζλπΠ ΛΖΛΠπ λλζγΒ ΛΛΖΓβ ΛΓγΒπ λγΓβΠ ΛΛΖΛΖ λλζλζ ΒλβΛΖ βΛΒλζ γΓβΛΒ ΓγΒλβ βΛΛγΓ ΒλΛΓγ ΓγΛΖΛ
γΓλζλ ΠπΛΓγ πΠλγΓ ΠπΒλβ πΠβΛΒ ΠλβΠπ πΛΒπΠ ΛΓβΓβ λγΒγΒ λγΒλζ ΛΓβΛΖ ΠλβΛΖ πΛΒλζ ΓβΛΓγ γΒλγΓ
ΛΓζλβ λγΖΛΒ πΛΒγΒ ΠλβΓβ λλβΛΒ ΛΛΒλβ πΓβΛΒ ΠγΒλβ Πλζλβ πΛΖΛΒ

List S2. 50 random cyclic peptide sequences in the test dataset (Dataset 4).

```
rNDsF vrdvA RfdNr SVFdR ANDnA DdFVs RAvRs fffaa SSrsA nvDnF nsaaF DAvsD snGAa avNSd
SdFrS avVrr dVsrf NAnNS RsFDS arFGa AsNAr fNvAA RADaS RDAvs FAdNA SrGnR nrRAv SaARF
FvrFR vfDsR VssSN RaRDR nARRD FsGna DfaNv NNDas DdSrD fnDSd VSadn dfRNd FdNfA VaAVn
sDAsF SVFAa AsNVs dNsGV GNsrn sdSfd vsVAG FaanA
```

List S3. 705 semi-random cyclic peptide sequences in the training dataset (Dataset 3) for StrEAMM (1,2)+(1,3)/random.

```
SnSVa nNnGA NfNdv fFfDV FaFrG aAaRd AvAsD vVvSr VGVnR GdGNs dDdfS DrDFn rRraN RsRAf sSsvF
drrFV DRRaG rssAd RSSvD snnVr SNNGR nffds faarn avvsf vGGna VddNA GDDfv AGsVN vdSGf VDndF
GrNDa dRfrA DsFRv rSasV RnASG sNvnd SfVND nFGfr NadFR fADas FvrAS aVRvn VNrSS GfRnn dFsNN
DaSff rAnFF RvNaa sVfAA SGFvv ndaVV NDAGG frvdd asGrr ASdRR vnDss sFdns SaDNS nArfn NvRFN
fVsaf FGSAF adnva vrfGv VRFdV GsaDG dSArd DnvRD rNVsr RfGSR nddRr NDDsR frrSs FRRnS assNn
ASSfN vnnFf VNNaF GffAa dFFvA DaaVv rAAGV RvvdG sVVDd SGGrD RRRGr sssdR SSSDs nnnrS NNNRn
fffsN FFFSf aaanF AAANa vvvfA VVVFv GGGaV dddAG DDDvd rrrVD dfRFv DFsaV raSAG RAnvd svNVD
SVfGr nGFdR NdaDs fDArS FrvRn aRVsN AsGSf vSdnF VnDNa GNrfA VnsRD GNSsr dfnSR DFNns rafNS
RAFfn svaFN SVAaf nGvAF NdVva fDGVA FrdGv aRDdV AsrDG vSRrd DnFSA rNanv RfANV sFvfG SaVFd
nAGaD NvdAr fVDvR FGrVs adRGS ADsdn vrSDN VRnrf GsNRF dSfsa dnGnR DNdNs rfDfS RFrFn saRaN
SAsAf nvSvF NVnVa fGNGA Fdfdv aDFDV ArarG vRARd VsvsD GSVSr vVdDG VGDrd GdrRD dDRsr DrsSR
rRSns RsnNS sSNfn SnfFN nNFaf NfaAF fFAva FavVA aAVGv AvGdV sRRFV SssaG nSSAd NnnvD fNNVr
FffGR aFFds AaaDS vAArn VvvRN GVVsf dGGSF Dddna rDDNA Rrrfv DaFDs rAarS RvARn sVvsN SGVSf
ndGnF NDdNa frDfA FRrFv asRaV ASsAG vnSvd VNnVD GfNGr dFfdR vDdRa VrDsA GRrSv dsRnV DSsNG
rnSfd RNnFD sfNar SFfAR naFvs NAaVS fvAGn FVvdN aGVDf AdGrF GdsGn dDSdN DrnDf rRNrF RsfRa
sSFsA SnaSv nNAnV NfvNG fFVfd FaGFD aAdar AvDAR vVrvs VGRVS NGdFd fdDaD FDrAr arRvR ARsVs
vsSGS VSndn GnNDN dNfrf DfFRF rFasa RaASA sAvnv SvVNV nVGfG DDGFs rrdaS RRDAn ssrvN SSRVf
nnsGF NNSda ffnDA FFNrv aafRV AAFsG vvaSd VVAnD GGvNr ddVfR DnrAF rNRva RfsVA sFSGv SandV
nANDG Nvfrd fVFRD FGasr adASR ADvns vrVNS VRGfn GsdFN dSDaf ARDan vsrAN VSRvf GnsVF dNSGa
DfndA rFNDv RafrV sAFRG Svasd nVASD NGvnr fdVNR FDGfs ardFS RaaGN sAAdf SvvDF nVVra NGGRA
fddsv arrnG ARRNd vssfD VSSFr GnnaR dNNAs DffvS rFFVn GdvDf dDVrF DrGRa rRdsA RsDSv sSrnV
SnRNG nNsfd NfSFD fFnar FaNAR aAfvs AvFVS vVaGn VGAdN NAASv fvvnV FVVNG aGGfd AddFD vDDar
VrrAR GRRvs dssVS DSSGn rnndN RNNDf sffrF SFFRa naasA AsaNR vSAfs VnvFS GNVan dfGAN DFdvf
raDVF RArGa svRdA SVsDv nGSrV NdnRG fDNsd FrfSD aRFnr vaGRD VAdsr GvDSR dVrns DGRNS rdsfn
srnaf nsfva fnaGv FNAdV afvDG AFVrd DvGnr rVdNR RGDfs sdrFS SDRan nrsAN NRSvf fsnVF FSNGa
anfdA ANFDv vfarV VFARG Gavsd dAVSD vNNFN Vffaf GFFAF daava rvvGv RVVdV sGGDG Sddrd nDDRD
Nrrsr fRRSR Fssns aSSNS Annfn aFdra AaDRA vArsv VvRSV GVsnG dGSNd DdnfD rDNFr RrfaR sRFAs
SsavS nSAVn NnvGN fNVdf FfGDF GVvFv dGVaV DdGAG rDdvd RrDVD sRrGr SsRdR nSsDs NnSrS fNnRn
FfNsN aFfSf AaFnF vAaNa VvAfA asvrR ASVRs vnGsS VNdSn GfDnN dFrNf DaRfF rAsFa RvSaA sVnAv
SGNvV ndfVG NDFGd fradD FRADr SsVGd nSGdD NndDr fNDrR FfrRs aFRsS AasSn vASnN VvnNf GVNfF
dGfFa DdFaA rDaAv RrAvV sRvVG vdVda VDGDA Grdrv dRDRV DsrsG rSRSd RnsnD sNSNr SfnfR nFNFs
NafaS fAFAn FvavN aVAVf AGvGF ASrdf vnRDF VNsra GfSRA dFnsv DaNSV rAfnG RvFNd sVafD SGAFr
ndvaR NDVAs frGvS FRdVn asDGN vVsdd VGSDD Gdnrr dDNRR Drfss rRFSS Rsann sSANN Snvff nNVFF
NfGaa fFdAA FaDvv aArVV AvRGG nadda fvrrv aGssG AdSSd vDnnD VrNNr GRffR dsFFs DSaaS rnAAn
RNvvN sfVVf SFGGF snrvS SNRVn nfsGN NFSdf fanDF FANra avfRA AVFsv vGaSV VdAnG GDvNd drVfD
DRGFr rsdaR RSDAs RfvFf sFVaF SaGAa nAdvA NvDVv fVrGV FGRdG adsDd ADSrD vrnRr VRNsR GsfSs
dSFnS DnaNn rNAfN RFRfD sasFr SASaR nvnAs NVNvS fGfVn FdFGN aDadf ArADF vRvra VsVRA GSGsv
dndSV DNDnG rfrNd VAGFf GvdaF dVDAa DGrvA rdRVv RDsGV srSdG SRnDd nsNrD NSfRr fnFsR FNaSs
afAnS AFvNn vaVfN vNSAA Vfnvv GFNVV dafGG DAFdd rvaDD RVArr sGvRR SdVss nDGSS Nrdnn fRDNN
Fsrff aSRFF Ansaa sAaAN SvAvf nVvVF NGVGa fdGdA FDdDv arDrV ARrRG vsRsd VSsSD GnSnr dNnNR
DfNfs rFfFS RaFan rVFaV RGaAG sdAvd SDvVD nrVGr NRGdR fsdDs FSDrS anrRn ANRsN vfsSf VFSnF
GanNa dANfA DvfFv SrsVd nRSGD Nsndr fSNDR Fnfrs aNFRS Afasn vFASN Vavnf GAVNF dvGfa DVdFA
rGDav RdrAV sDRvG DNVAD FVGNA AdDFV DSnGR RNfDS SFaRN NAvSF aDAAV NFAVA FsVDD AADAN VSFAf
FDSNs ARNFn NvGAD DFRVF FFDDS DVRRV SVFVR NVfrn Fdasf Arvna VsGfv dnDaG rfRvD saSGR SAnds
VNaDS VrRAV nNdsD fFrnR aAsfS vVnaN GdfvF sSVrG DaVsv GfAra dFvRA FRnvn rAGSG Rvdnd NDsas
frSAS NADRV VadrG dvrsD SrNFF RdSNF SfGFR VDNrA sFAAR AFSRv DDASa DnfSn FGDVr FNRSr GSdFD
GsnRv NRDRd RNANS RfaNn RsvRV SavFf VVSFV dRFsS fVdvv nAVAA nvNDG DraAr sNVSf GrfAn nFdad
```

List S4. 75 random cyclic peptide sequences in the test dataset (Dataset 6) for StrEAMM (1,2)+(1,3)/sys37, including 37 types of amino acids.

```
AaGGr AewYN AHCVE ANiRQ AqWLw ASFsY caTNv DeSMA dhKYA dyqDk EekVh emfAR FdFKF FlcsH fLWSl
FrMRM FVnTK GcEQd GqqDE GyATS GYEtS HDckk HDlkt hlMKy hlstI HSfCT hwtiD IEKst INfWm IyyKA
kKEct kqRef LlHHK LMdCW lWrHh MsMTK NhfwK nhSYY NHyFI nSLtc qaqQH qcyAf qIqrV QlSFV qTySW
```

qwHca QWRNH raNaK rddwn rICAm RQwsv rtiGN SdNTl sDyeT sSnvf SvVid Syhww taEVR TddDk Teslq
tIqIL tIWci TqGdv vccVe vQNGk VTrDl WFFqN wHseh WtYdE YciGr YdFAm ynWvy ySmmw YvrNC YwnlK

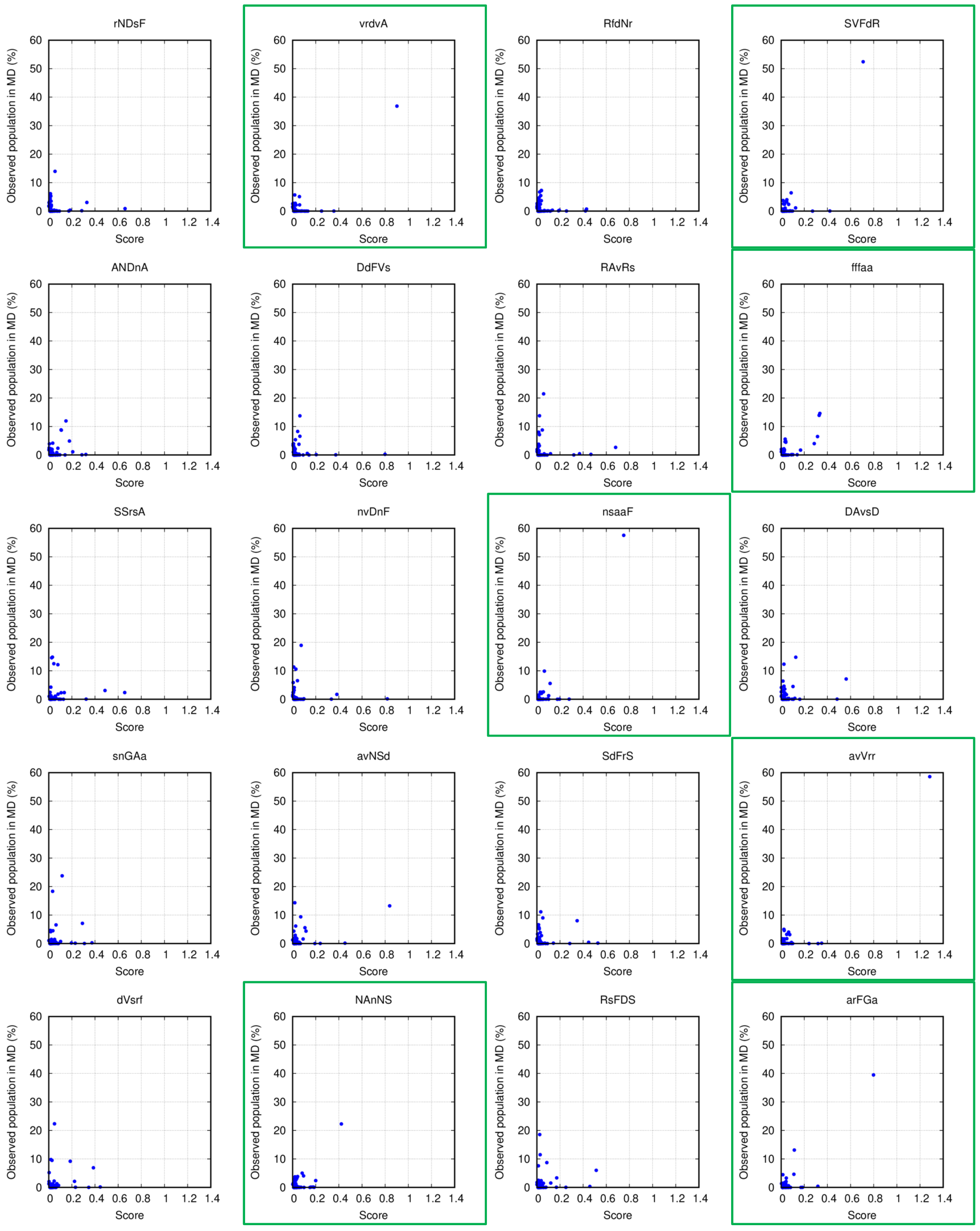
List S5. Dataset 6 in List S4 was divided into two sub datasets, Dataset 6.1 and Dataset 6.2. Dataset 6.1 was used in the training of StrEAMM GNN/random37; Dataset 6.2 was used for testing both StrEAMM GNN/random and StrEAMM GNN/random37.

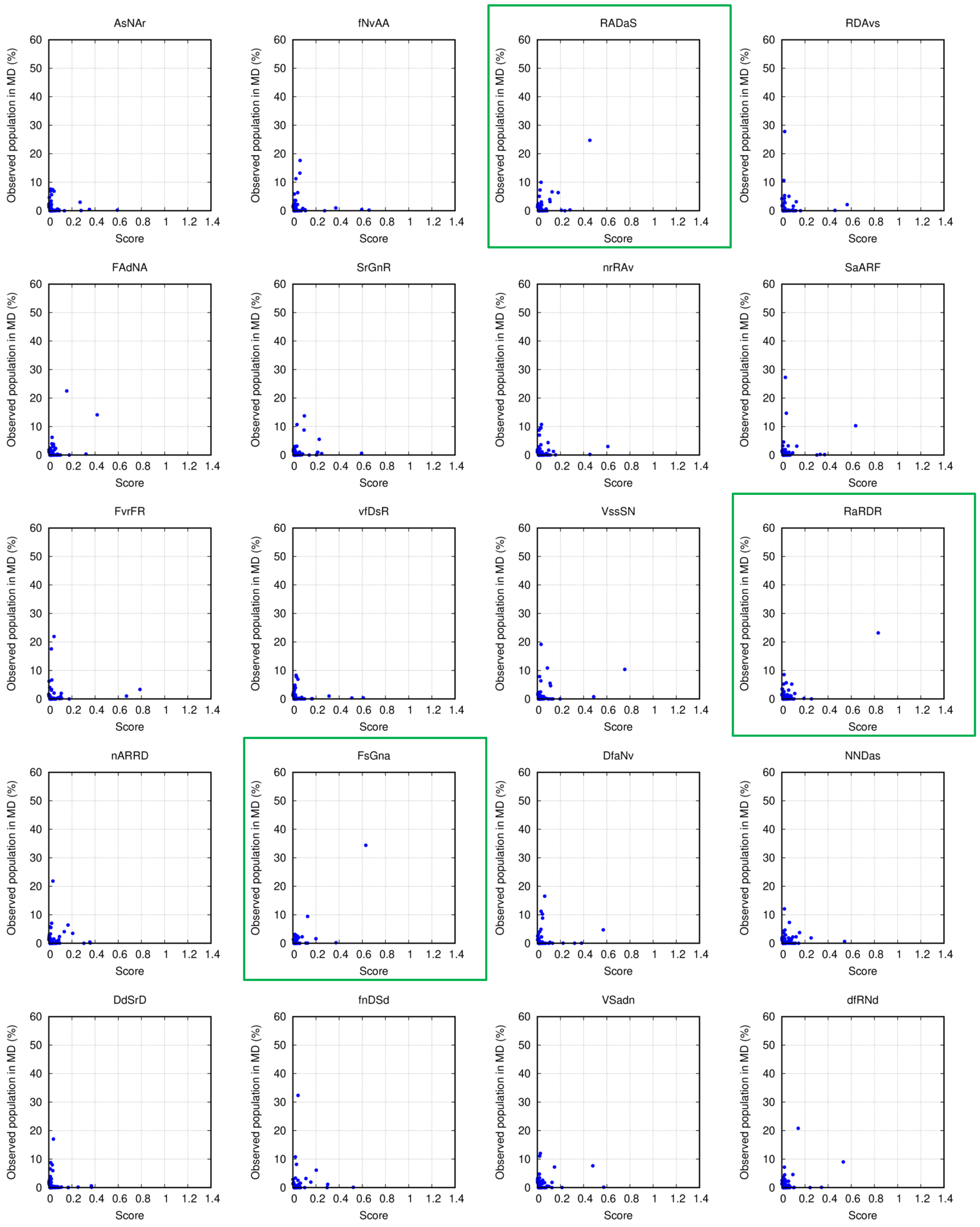
Dataset 6.1:
AaGGr AewYN AHCVE ANiRQ AqWLw ASFsY caTNv DeSMA dhKYA dyqDk EekVh emfAR FdFKF FlcsH fLWSl
FrMRM FVnTK GcEQd GqqDE GyATS GYEtS HDckk HDlkt hlMKy hlstI HSfCT hwtiD IEKst INfWm IyyKA
kKEct kqRef LlHHK LMdCW lWrHh MsMTK NhfwK nhSYY NHyFI nSLtc qaqQH qcyAf qIqrV QlSFV qTySW
qwHca QWRNH raNaK rddwn rICAm
Dataset 6.2:
RQwsv rtiGN SdNTl sDyeT sSnvf SvVid Syhww taEVR TddDk Teslq tIqIL tIWci TqGdv vccVe vQNGk
VTrDl WFFqN wHseh WtYdE YciGr YdFAm ynWvy ySmmw YvrNC YwnlK

$$\begin{gathered}
\ln p^{\mathrm{GGGGG}}_{\Lambda\Lambda\Lambda\Lambda\Lambda} = 5\times w^{\mathrm{GG}}_{\Lambda\Lambda} + 5\times w^{\mathrm{G\_G}}_{\Lambda\Lambda\Lambda} + w^{\mathrm{GGGGG}}_{\mathrm{Q}} \\
\ln p^{\mathrm{GGGGG}}_{\lambda\lambda\lambda\lambda\lambda} = 5\times w^{\mathrm{GG}}_{\lambda\lambda} + 5\times w^{\mathrm{G\_G}}_{\lambda\lambda\lambda} + w^{\mathrm{GGGGG}}_{\mathrm{Q}} \\
\ln p^{\mathrm{GGGGG}}_{\Lambda\lambda\Lambda\lambda\Lambda} = w^{\mathrm{GG}}_{\Lambda\Lambda} + 2\times w^{\mathrm{GG}}_{\Lambda\lambda} + 2\times w^{\mathrm{GG}}_{\lambda\Lambda} + w^{\mathrm{G\_G}}_{\Lambda\Lambda\lambda} + 2\times w^{\mathrm{G\_G}}_{\Lambda\lambda\Lambda} + w^{\mathrm{G\_G}}_{\lambda\Lambda\lambda} + w^{\mathrm{G\_G}}_{\lambda\Lambda\Lambda} + w^{\mathrm{GGGGG}}_{\mathrm{Q}} \\
\vdots \\
\ln p^{\mathrm{VGGGG}}_{\Lambda\Lambda\Lambda\Lambda\Lambda} = 3\times w^{\mathrm{GG}}_{\Lambda\Lambda} + w^{\mathrm{VG}}_{\Lambda\Lambda} + w^{\mathrm{GV}}_{\Lambda\Lambda} + 3\times w^{\mathrm{G\_G}}_{\Lambda\Lambda\Lambda} + w^{\mathrm{V\_G}}_{\Lambda\Lambda\Lambda} + w^{\mathrm{G\_V}}_{\Lambda\Lambda\Lambda} + w^{\mathrm{VGGGG}}_{\mathrm{Q}} \\
\ln p^{\mathrm{VGGGG}}_{\lambda\lambda\lambda\lambda\lambda} = 3\times w^{\mathrm{GG}}_{\lambda\lambda} + w^{\mathrm{VG}}_{\lambda\lambda} + w^{\mathrm{GV}}_{\lambda\lambda} + 3\times w^{\mathrm{G\_G}}_{\lambda\lambda\lambda} + w^{\mathrm{V\_G}}_{\lambda\lambda\lambda} + w^{\mathrm{G\_V}}_{\lambda\lambda\lambda} + w^{\mathrm{VGGGG}}_{\mathrm{Q}} \\
\vdots \\
\Downarrow
\end{gathered}$$

$$\begin{array}{cccc}
\begin{pmatrix}
\ln p^{\mathrm{GGGGG}}_{\Lambda\Lambda\Lambda\Lambda\Lambda} \\
\ln p^{\mathrm{GGGGG}}_{\lambda\lambda\lambda\lambda\lambda} \\
\ln p^{\mathrm{GGGGG}}_{\Lambda\lambda\Lambda\lambda\Lambda} \\
\vdots \\
\ln p^{\mathrm{VGGGG}}_{\Lambda\Lambda\Lambda\Lambda\Lambda} \\
\ln p^{\mathrm{VGGGG}}_{\lambda\lambda\lambda\lambda\lambda} \\
\vdots
\end{pmatrix}
= &
\begin{pmatrix}
5 & 0 & 0 & 0 & 0 & \cdots & 5 & 0 & 0 & 0 & 0 & \ldots & 1 & \cdots & 0 & \cdots \\
5 & 0 & 0 & 0 & 0 & \cdots & 5 & 0 & 0 & 0 & 0 & \cdots & 1 & \cdots & 0 & \cdots \\
1 & 4 & 0 & 0 & 0 & \cdots & 0 & 1 & 3 & 1 & 0 & \cdots & 1 & \cdots & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \cdots \\
3 & 0 & 0 & 0 & 0 & \cdots & 3 & 0 & 0 & 0 & 0 & \cdots & 0 & \cdots & 1 & \cdots \\
3 & 0 & 0 & 0 & 0 & \cdots & 3 & 0 & 0 & 0 & 0 & \cdots & 0 & \cdots & 1 & \cdots \\
 & & & & \vdots & & & & & & & & \vdots & & &
\end{pmatrix}
\cdot &
\begin{pmatrix}
w^{\mathrm{GG}}_{\Lambda\Lambda}(\text{or } w^{\mathrm{GG}}_{\lambda\lambda}) \\
w^{\mathrm{GG}}_{\Lambda\lambda}(\text{or } w^{\mathrm{GG}}_{\lambda\Lambda}) \\
w^{\mathrm{GG}}_{\Lambda\mathrm{B}}(\text{or } w^{\mathrm{GG}}_{\lambda\beta}) \\
w^{\mathrm{GG}}_{\mathrm{B}\Lambda}(\text{or } w^{\mathrm{GG}}_{\beta\lambda}) \\
w^{\mathrm{GG}}_{\Lambda\beta}(\text{or } w^{\mathrm{GG}}_{\lambda\mathrm{B}}) \\
\vdots \\
w^{\mathrm{G\_G}}_{\Lambda\Lambda\Lambda}(\text{or } w^{\mathrm{G\_G}}_{\lambda\lambda\lambda}) \\
w^{\mathrm{G\_G}}_{\Lambda\Lambda\lambda}(\text{or } w^{\mathrm{G\_G}}_{\lambda\lambda\Lambda}) \\
w^{\mathrm{G\_G}}_{\Lambda\lambda\Lambda}(\text{or } w^{\mathrm{G\_G}}_{\lambda\Lambda\lambda}) \\
w^{\mathrm{G\_G}}_{\lambda\Lambda\Lambda}(\text{or } w^{\mathrm{G\_G}}_{\Lambda\lambda\lambda}) \\
w^{\mathrm{G\_G}}_{\Lambda\Lambda\mathrm{B}}(\text{or } w^{\mathrm{G\_G}}_{\lambda\lambda\beta}) \\
\vdots \\
w^{\mathrm{GGGGG}}_{\mathrm{Q}} \\
\vdots \\
w^{\mathrm{VGGGG}}_{\mathrm{Q}}(\text{or } w^{\mathrm{vGGGG}}_{\mathrm{Q}}) \\
\vdots
\end{pmatrix} \\
N\times 1 \text{ vector} & N\times M \text{ matrix} & M\times 1 \text{ vector}
\end{array}$$

Figure S1. Illustration of the matrix equation (7) $\ln \boldsymbol{p} = \boldsymbol{A}\boldsymbol{w}$. The logarithms of populations are arranged into a column vector of size *N*, where *N* is the summation of the number of structure types of each cyclic peptide in the training set. Different weights are arranged into a column vector of size *M*, where *M* is the number of weights. Weights that are mirror images of each other are treated as equal, for example, $w^{\mathrm{X}_i\mathrm{X}_{i+1}}_{\mathrm{S}_i\mathrm{S}_{i+1}} = w^{\mathrm{x}_i\mathrm{x}_{i+1}}_{\mathrm{s}_i\mathrm{s}_{i+1}}$, $w^{\mathrm{X}_i\_\mathrm{X}_{i+2}}_{\mathrm{S}_i\mathrm{S}_{i+1}\mathrm{S}_{i+2}} = w^{\mathrm{x}_i\_\mathrm{x}_{i+2}}_{\mathrm{s}_i\mathrm{s}_{i+1}\mathrm{s}_{i+2}}$, and $w_Q^{\mathrm{X}_1\mathrm{X}_2\mathrm{X}_3\mathrm{X}_4\mathrm{X}_5} = w_Q^{\mathrm{x}_1\mathrm{x}_2\mathrm{x}_3\mathrm{x}_4\mathrm{x}_5}$, with capital and lowercase letter pairs representing enantiomers of amino acids and structures. The coefficient matrix $\boldsymbol{A}$ controls which weights are used to compute the population of a specific cyclic-peptide sequence adopting a specific structure.


rNDsF
vrdvA
RfdNr
SVFdR
ANDnA
DdFVs
RAvRs
fffaa
SSrsA
nvDnF
nsaaF
DAvsD
snGAa
avNSd
SdFrS
avVrr
dVsrf
NAnNS
RsFDS
arFGa
Observed population in MD (%)
Score
0
10
20
30
40
50
60
0 0.2 0.4 0.6 0.8 1 1.2 1.4


AsNAr
fNvAA
RADaS
RDAvs
FAdNA
SrGnR
nrRAv
SaARF
FvrFR
vfDsR
VssSN
RaRDR
nARRD
FsGna
DfaNv
NNDas
DdSrD
fnDSd
VSadn
dfRNd
Observed population in MD (%)
Score

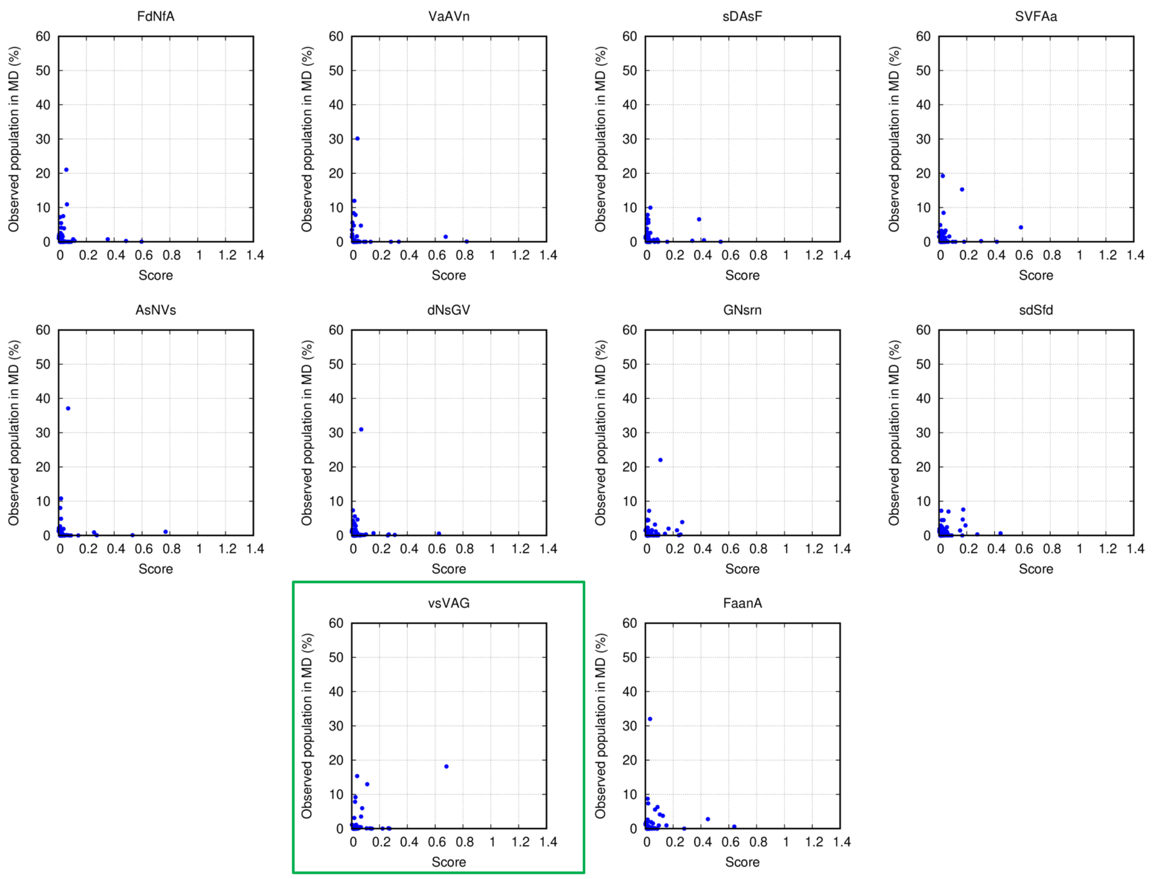


Figure S2. Performance of Scoring Function 1.0 on the test Dataset 4. Subplots show comparison between scores predicted by Scoring Function 1.0 and the actual populations of various structures observed in the MD simulations for 50 random sequences. Only structures whose observed populations are >1% or whose predicted scores are above 0.01 are shown. Green boxes show cyclic peptides whose top structures were predicted correctly by the scoring function.

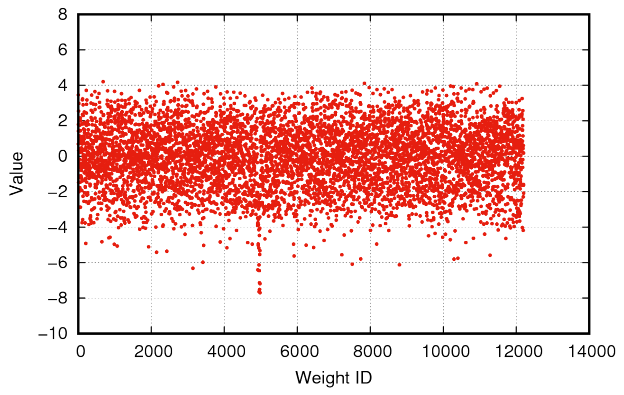
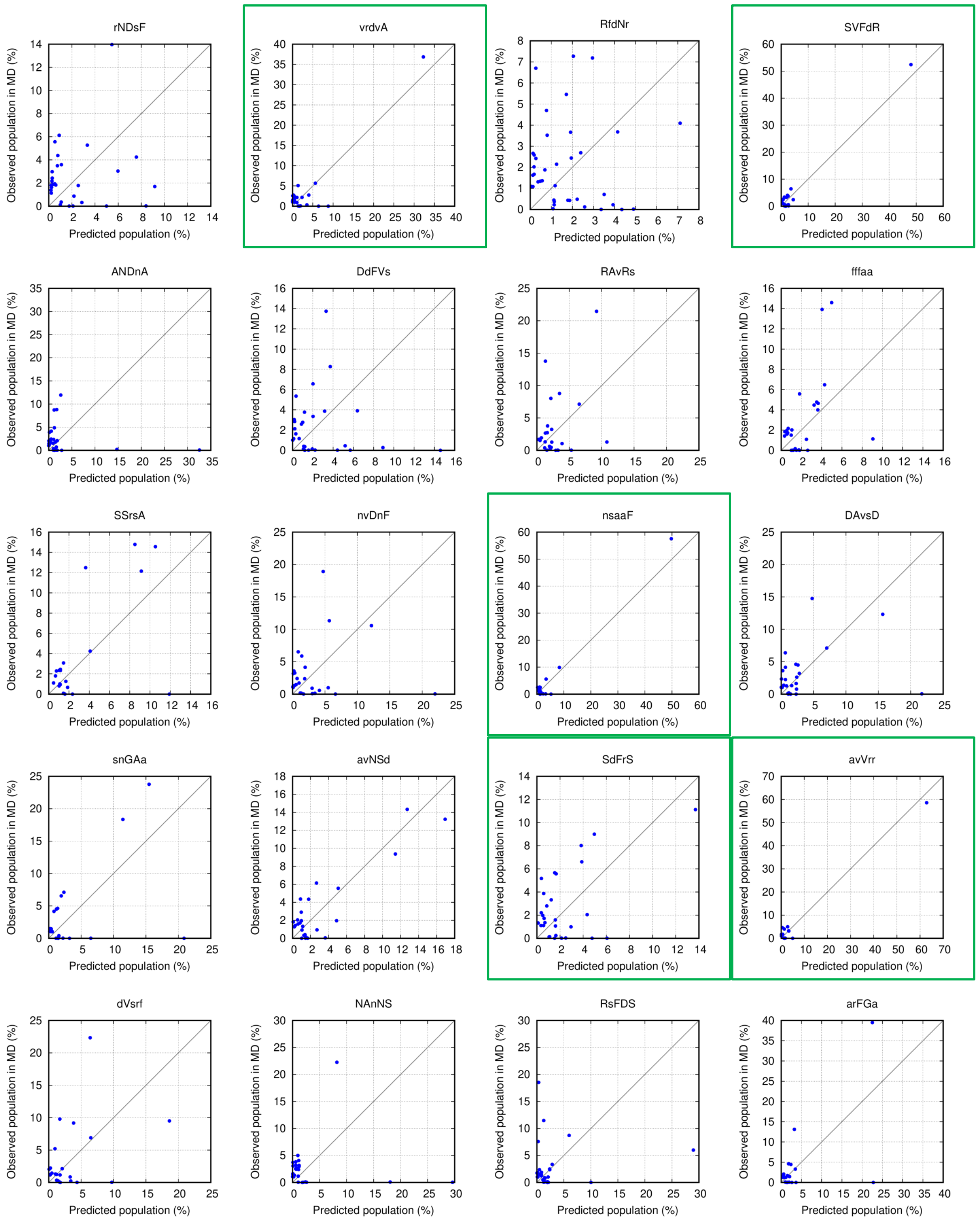
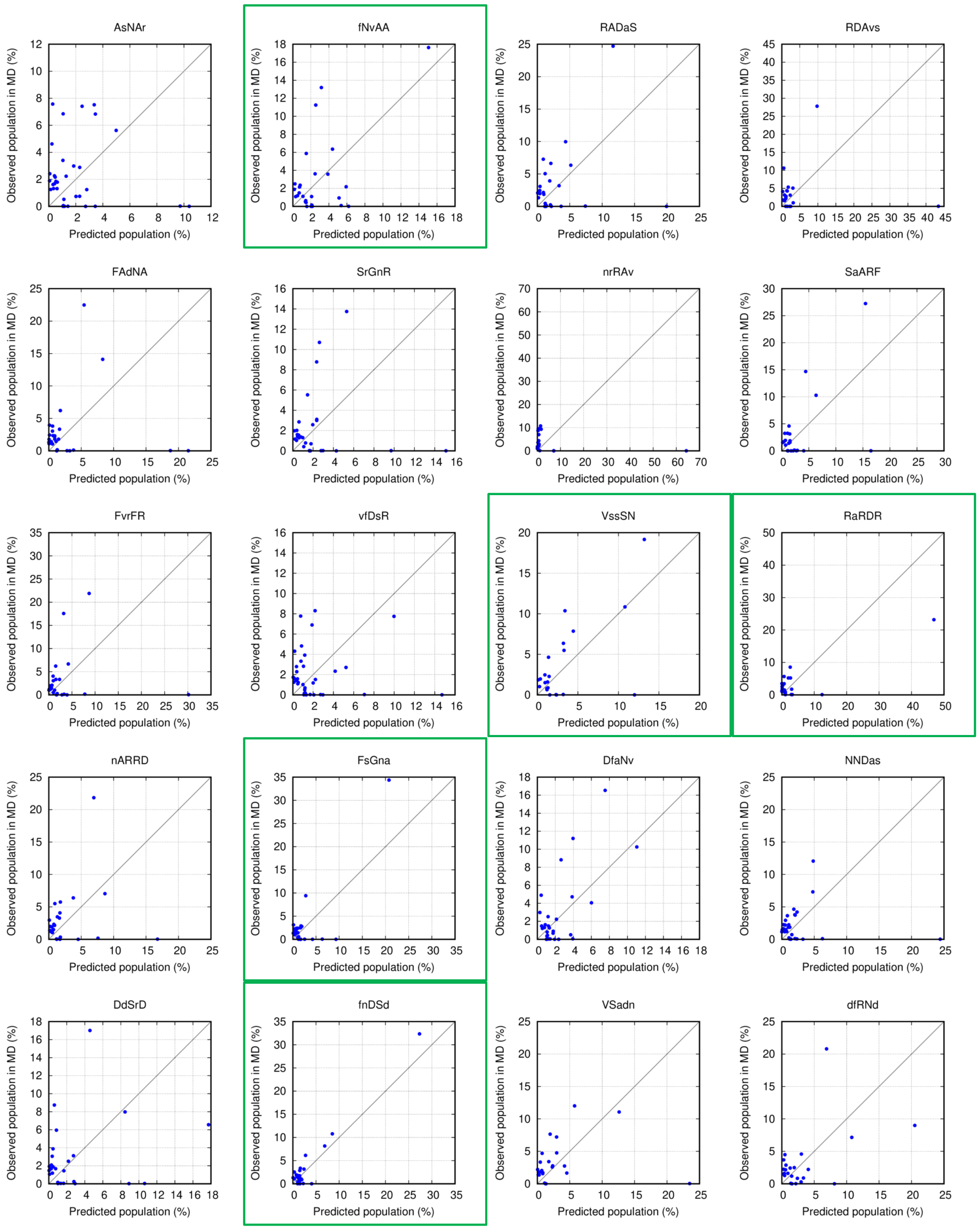


Figure S3. Distribution of weights for StrEAMM (1,2)/sys. The weights are related to (1,2) interactions. Both enantiomers of a weight are shown.


rNDsF
vrdvA
RfdNr
SVFdR
ANDnA
DdFVs
RAvRs
fffaa
SSrsA
nvDnF
nsaaF
DAvsD
snGAa
avNSd
SdFrS
avVrr
dVsrf
NAnNS
RsFDS
arFGa
Observed population in MD (%)
Predicted population (%)


AsNAr
fNvAA
RADaS
RDAvs
FAdNA
SrGnR
nrRAv
SaARF
FvrFR
vfDsR
VssSN
RaRDR
nARRD
FsGna
DfaNv
NNDas
DdSrD
fnDSd
VSadn
dfRNd
Observed population in MD (%)
Predicted population (%)

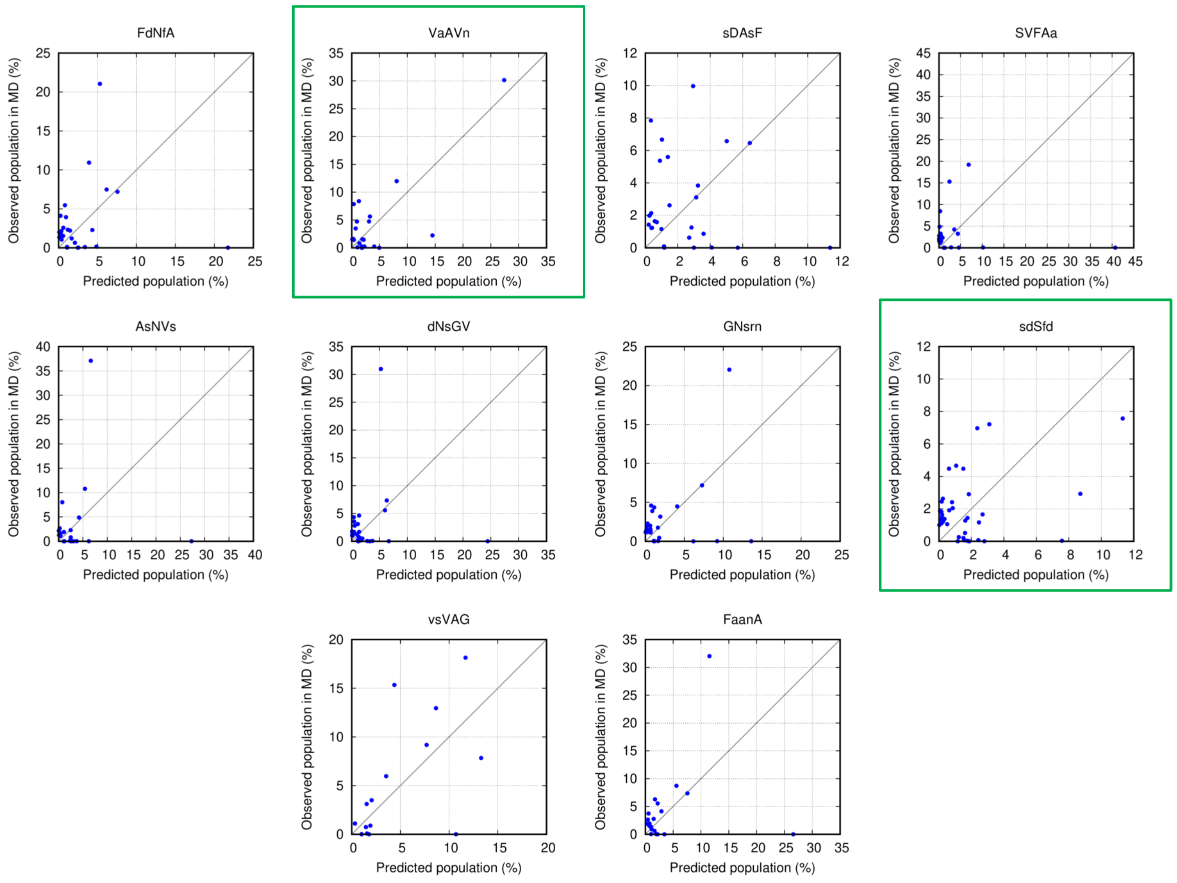


Figure S4. Performance of StrEAMM (1,2)/sys on the test Dataset 4. Subplots show comparison between populations predicted by **StrEAMM (1,2)/sys** and the actual populations of various structures observed in the MD simulations for 50 random sequences. Only structures with observed populations or predicted populations >1% are shown. Gray lines show where the predicted populations equal real populations. Green boxes show cyclic peptides whose top structures were predicted correctly by the StrEAMM model.

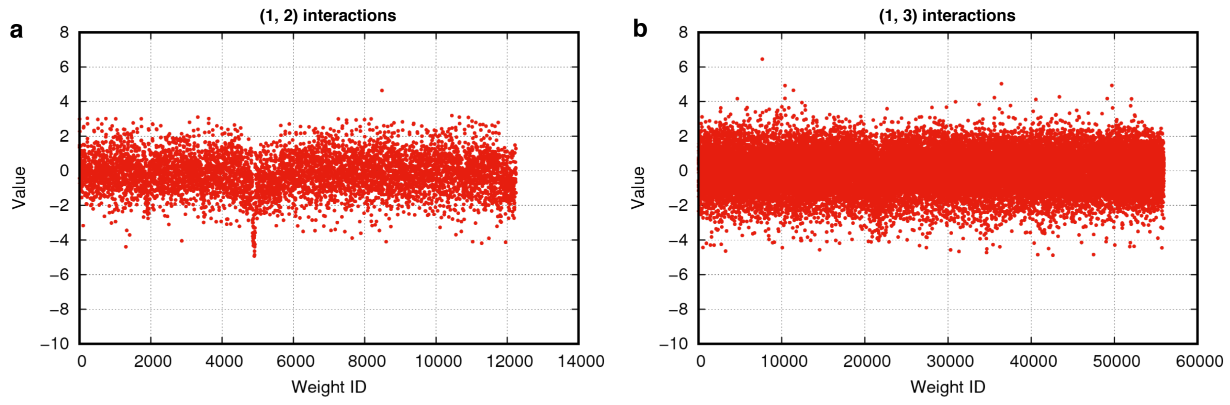
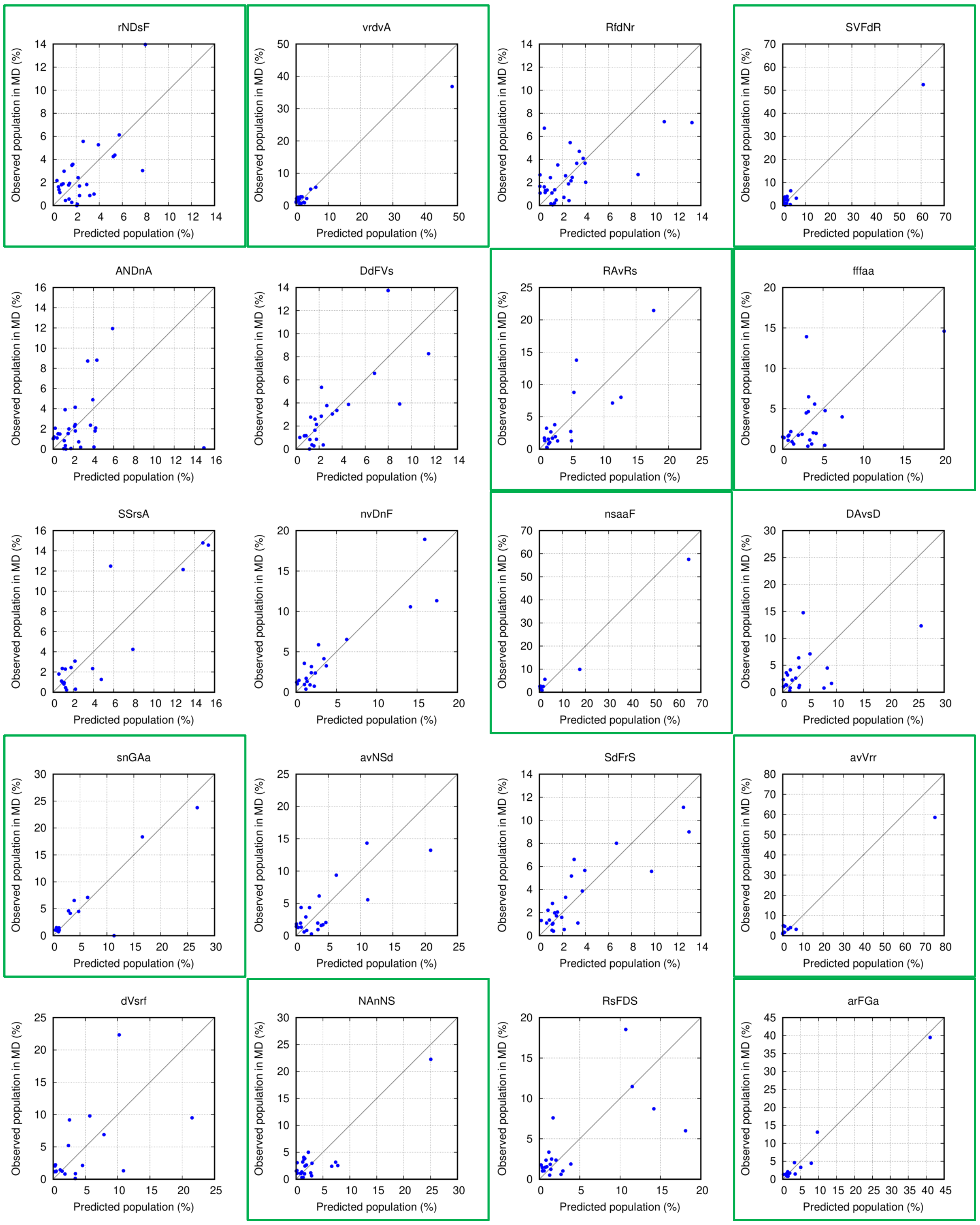
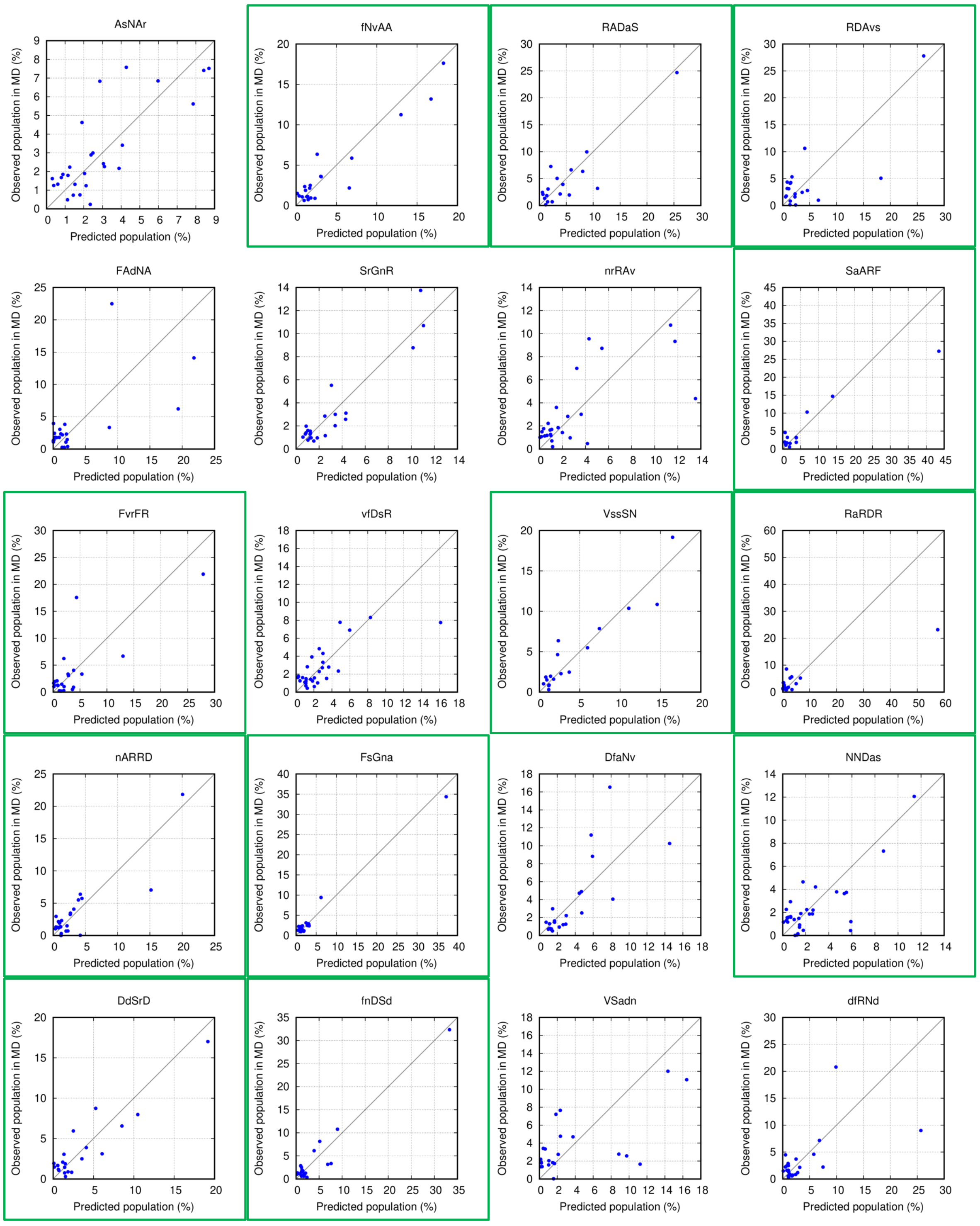


Figure S5. Distributions of weights for StrEAMM (1,2)+(1,3)/sys. a, Distribution of the weights related to (1,2) interactions. Both enantiomers of a weight are shown. **b,** Distribution of the weights related to (1,3) interactions. Both enantiomers of a weight are shown.


rNDsF
vrdvA
RfdNr
SVFdR
ANDnA
DdFVs
RAvRs
fffaa
SSrsA
nvDnF
nsaaF
DAvsD
snGAa
avNSd
SdFrS
avVrr
dVsrf
NAnNS
RsFDS
arFGa
Observed population in MD (%)
Predicted population (%)


AsNAr
fNvAA
RADaS
RDAvs
FAdNA
SrGnR
nrRAv
SaARF
FvrFR
vfDsR
VssSN
RaRDR
nARRD
FsGna
DfaNv
NNDas
DdSrD
fnDSd
VSadn
dfRNd
Observed population in MD (%)
Predicted population (%)

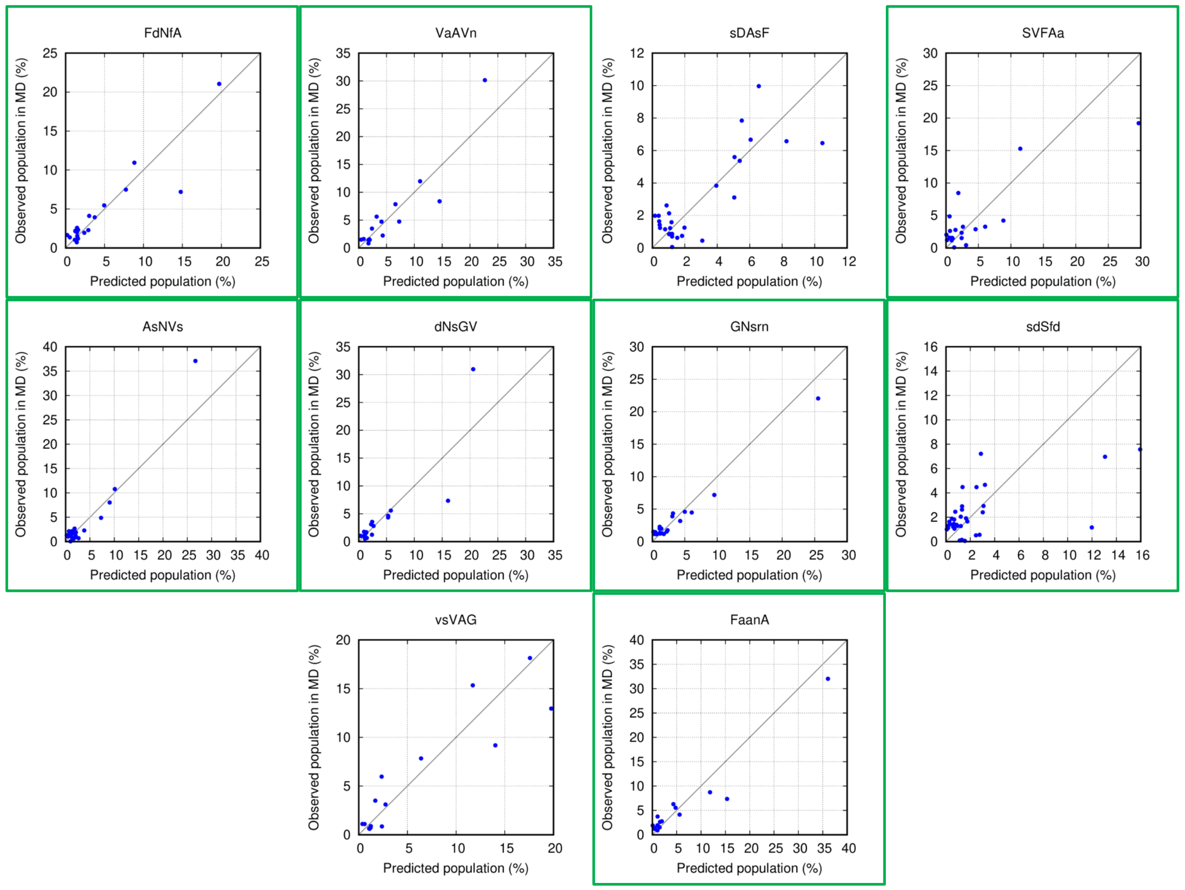


Figure S6. Performance of StrEAMM (1,2)+(1,3)/sys on the test Dataset 4. Subplots show comparison between populations predicted by **StrEAMM (1,2)+(1,3)/sys** and the actual populations of various structures observed in the MD simulations for 50 random sequences. Only structures with observed populations or predicted populations >1% are shown. Gray lines show where the predicted populations equal real populations. Green boxes show cyclic peptides whose top structures were predicted correctly by the StrEAMM model.

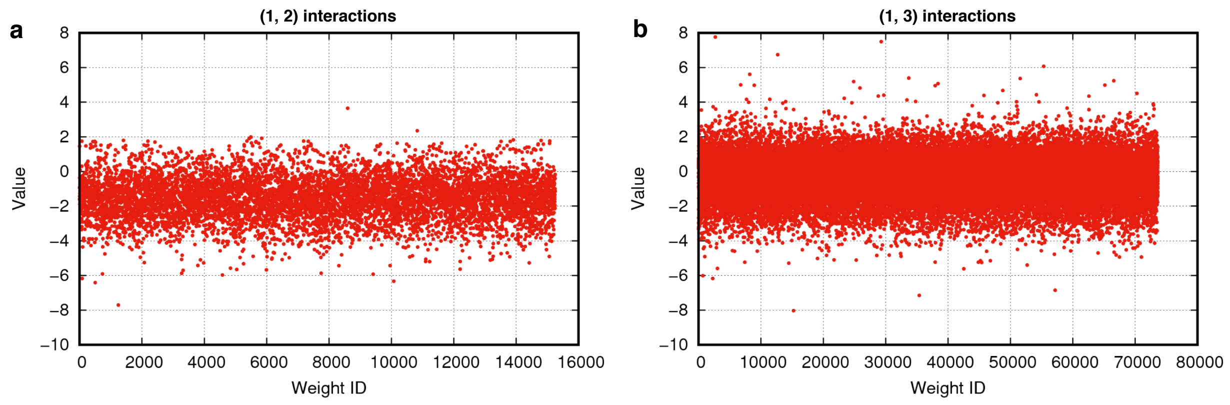
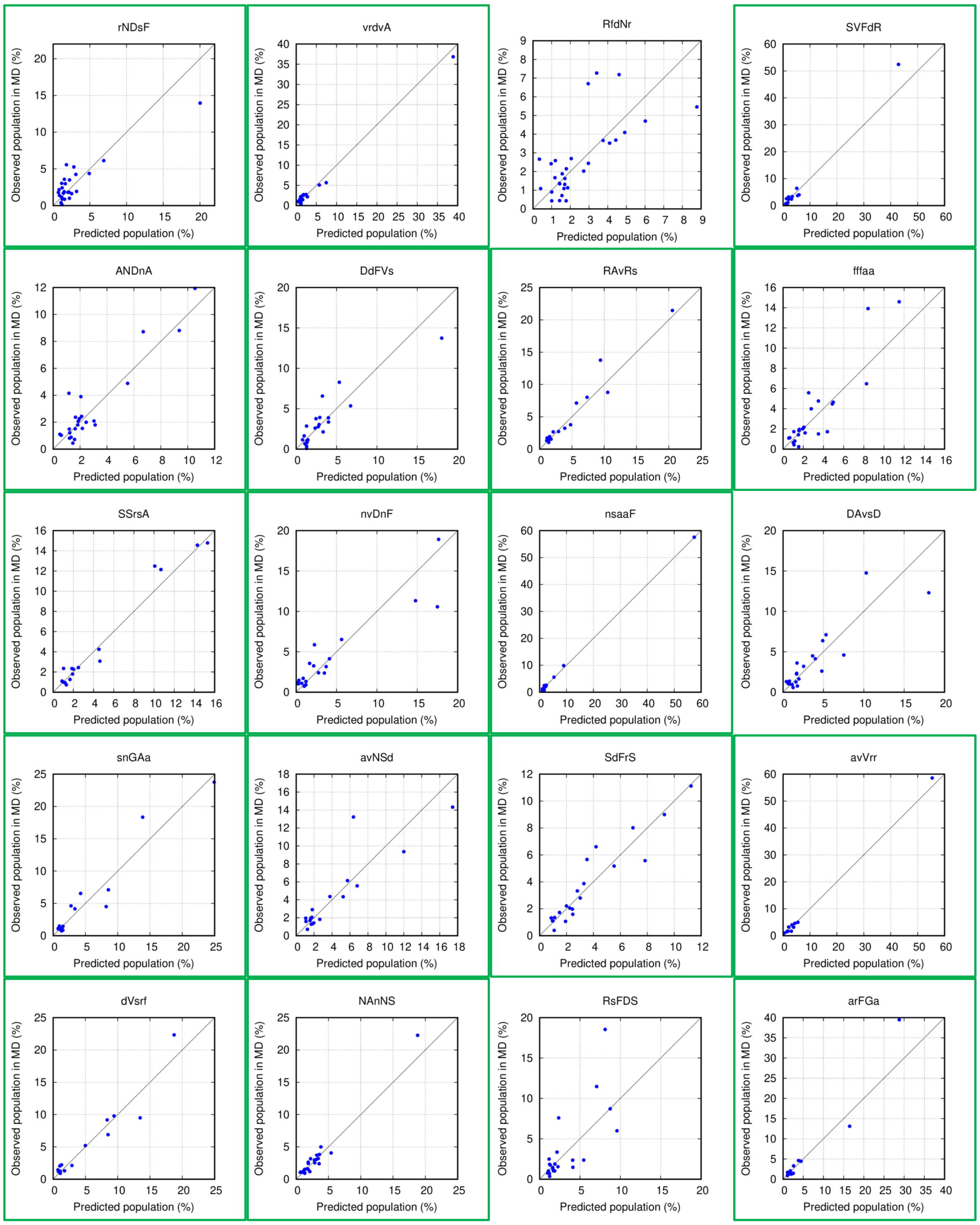
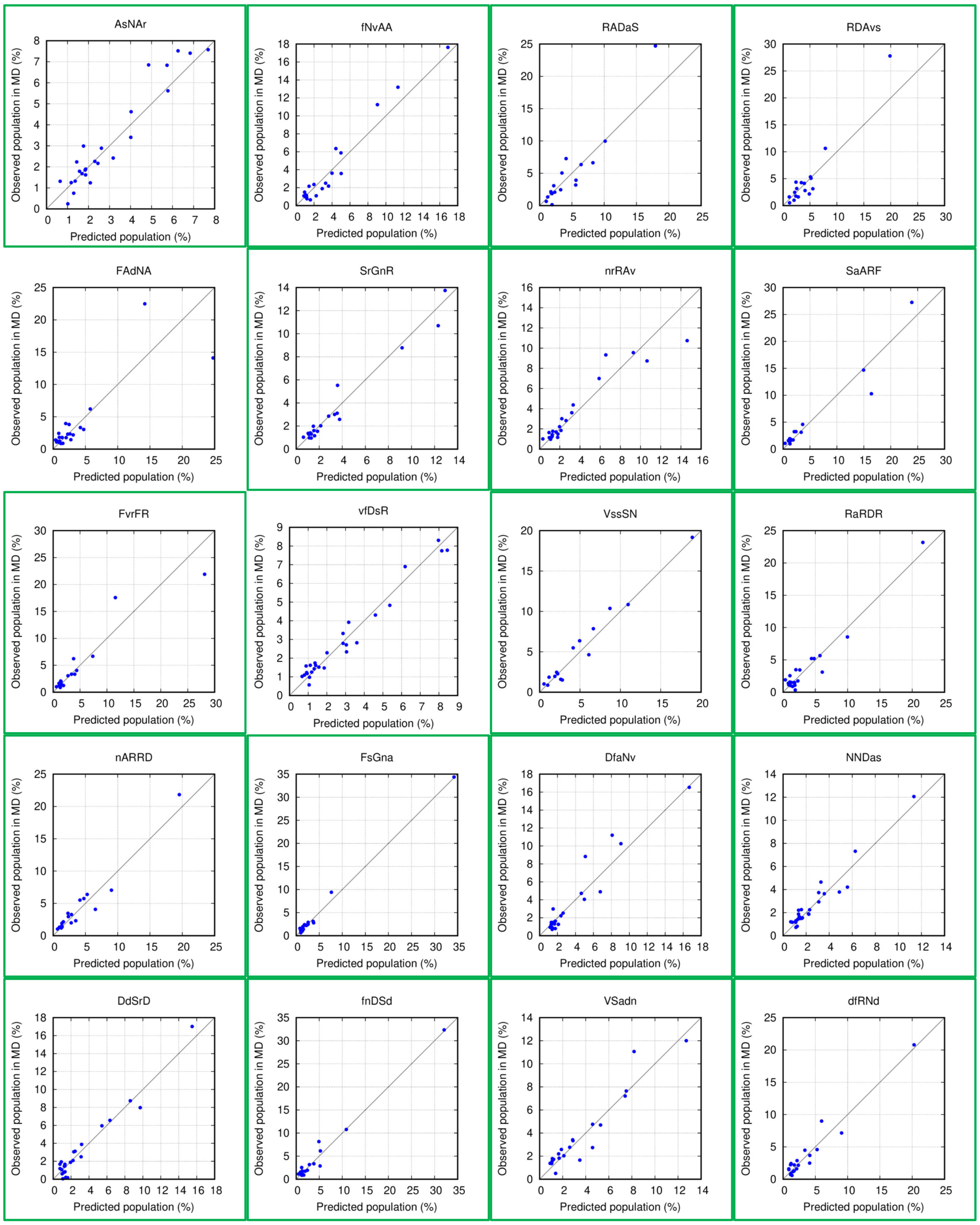


Figure S7. Distributions of weights for StrEAMM (1,2)+(1,3)/random. a, Distribution of the weights related to (1,2) interactions. Both enantiomers of a weight are shown. **b,** Distribution of the weights related to (1,3) interactions. Both enantiomers of a weight are shown.


rNDsF
vrdvA
RfdNr
SVFdR
ANDnA
DdFVs
RAvRs
fffaa
SSrsA
nvDnF
nsaaF
DAvsD
snGAa
avNSd
SdFrS
avVrr
dVsrf
NAnNS
RsFDS
arFGa
Observed population in MD (%)
Predicted population (%)


AsNAr
fNvAA
RADaS
RDAvs
FAdNA
SrGnR
nrRAv
SaARF
FvrFR
vfDsR
VssSN
RaRDR
nARRD
FsGna
DfaNv
NNDas
DdSrD
fnDSd
VSadn
dfRNd
Observed population in MD (%)
Predicted population (%)

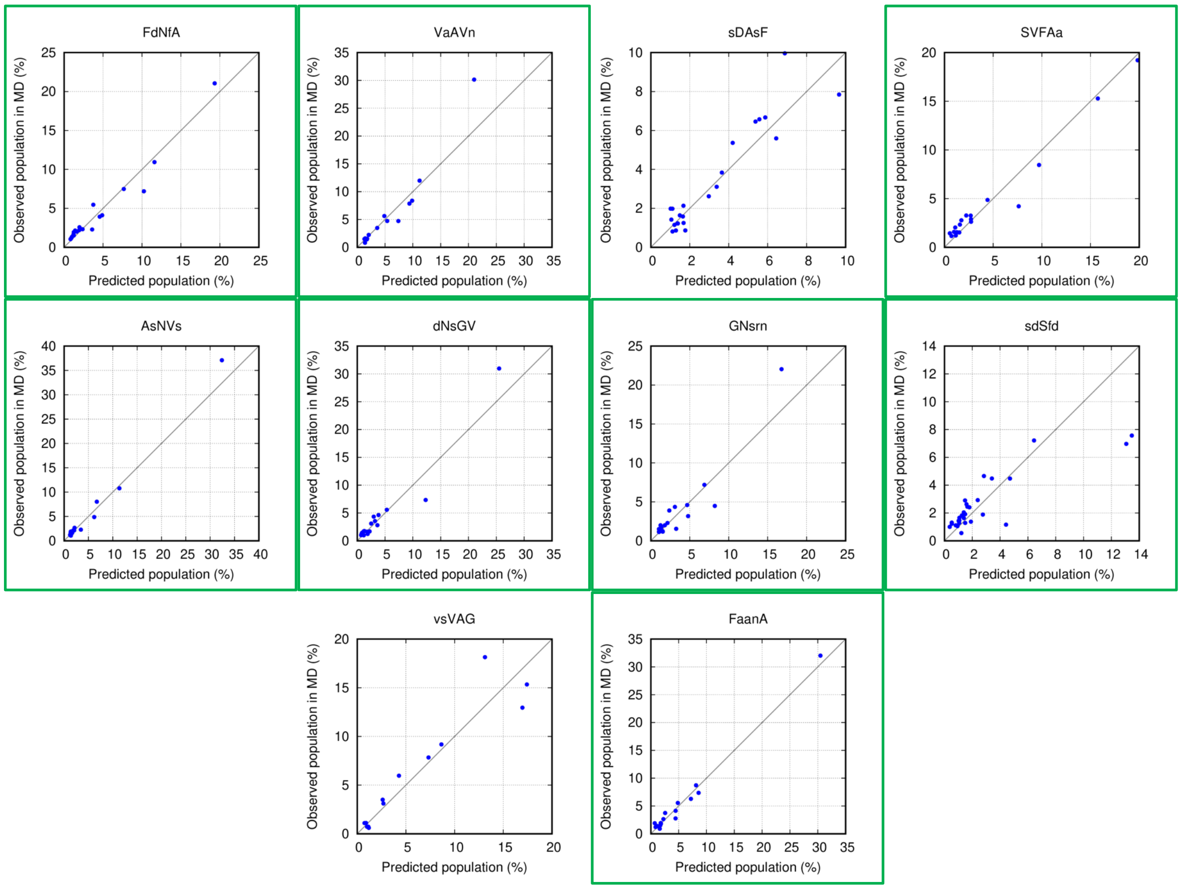


Figure S8. Performance of StrEAMM (1,2)+(1,3)/random on the test Dataset 4. Subplots show comparison between populations predicted by **StrEAMM (1,2)+(1,3)/random** and the actual populations of various structures observed in the MD simulations for 50 random sequences. Only structures with observed populations or predicted populations >1% are shown. Gray lines show where the predicted populations equal real populations. Green boxes show cyclic peptides whose top structures were predicted correctly by the StrEAMM model.

StrEAMM (1,2)+(1,3)/sys37

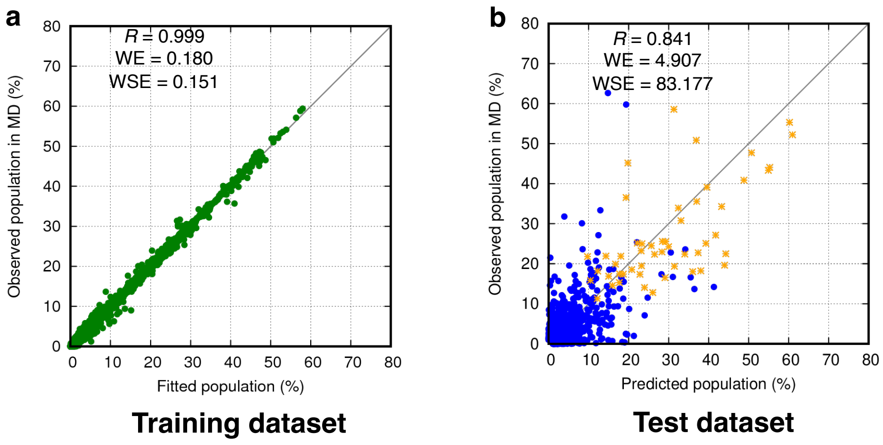


Figure S9. Performance of StrEAMM (1,2)+(1,3)/sys37. a, Comparison between the fitted populations and the actual populations of various structures observed in the MD simulations of the training dataset. **b,** Comparison between the populations predicted by **StrEAMM (1,2)+(1,3)/sys37** and the actual populations of various structures observed in the MD simulations of 75 random test sequences (**List S4**); only structures with observed populations or predicted populations >1% are shown. Pearson correlation coefficient (*R*), weighted error (WE $= \frac{\sum_i p_{i,\text{observed}}|p_{i,\text{observed}} - p_{i,\text{theory}}|}{\sum_i p_{i,\text{observed}}}$, where $p_{i,\text{theory}}$ is the fitted population or the predicted population), and weighted squared error (WSE $= \frac{\sum_i p_{i,\text{observed}}(p_{i,\text{observed}} - p_{i,\text{theory}})^2}{\sum_i p_{i,\text{observed}}}$) were calculated. Gray lines show where the fitted/predicted populations equal the observed populations in MD simulations. **StrEAMM (1,2)+(1,3)/sys37** successfully predicts the most-populated structures of 51 out of the 75 cyclic peptides in the test dataset, and these structures are shown as orange stars.

StrEAMM GNN/random

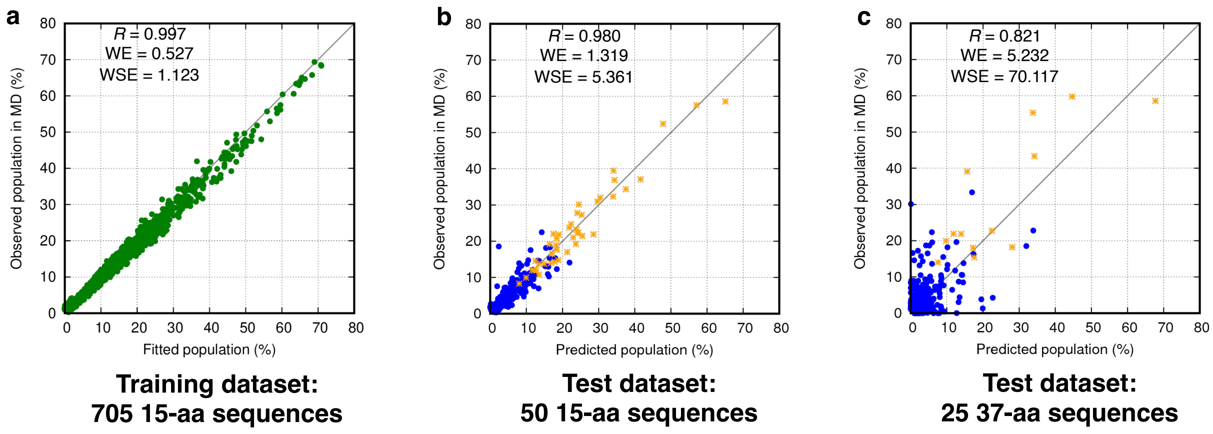


Figure S10. Performance of StrEAMM GNN/random. a, Comparison between the fitted populations and the actual populations of various structures observed in the MD simulations of the training dataset (Dataset 3). Data for enantiomers and cyclically permuted sequences are not shown, and only structures with observed populations or fitted populations >1% are shown. **b,** Comparison between the populations predicted by **StrEAMM GNN/random** and the actual populations of various structures observed in the MD simulations of 50 random test sequences (Dataset 4, **List S2**); only structures with observed populations or predicted populations >1% are shown. The model successfully predicts the most-populated structures of 42 out of the 50 cyclic peptides in the test dataset, and these structures are shown as orange stars. **c,** Comparison between the populations predicted by **StrEAMM GNN/random** and the actual populations of various structures observed in the MD simulations of another 25 random test sequences including 37 amino acids (Dataset 6.2, **List S5**); only structures with observed populations or predicted populations >1% are shown. The model successfully predicts the most-populated structures of 13 out of the 25 cyclic peptides in the test dataset, and these structures are shown as orange stars. Pearson correlation coefficient (*R*), weighted error (WE), and weighted squared error (WSE) were calculated. Gray lines show where the fitted/predicted populations equal the observed populations in MD simulations.

StrEAMM GNN/random37

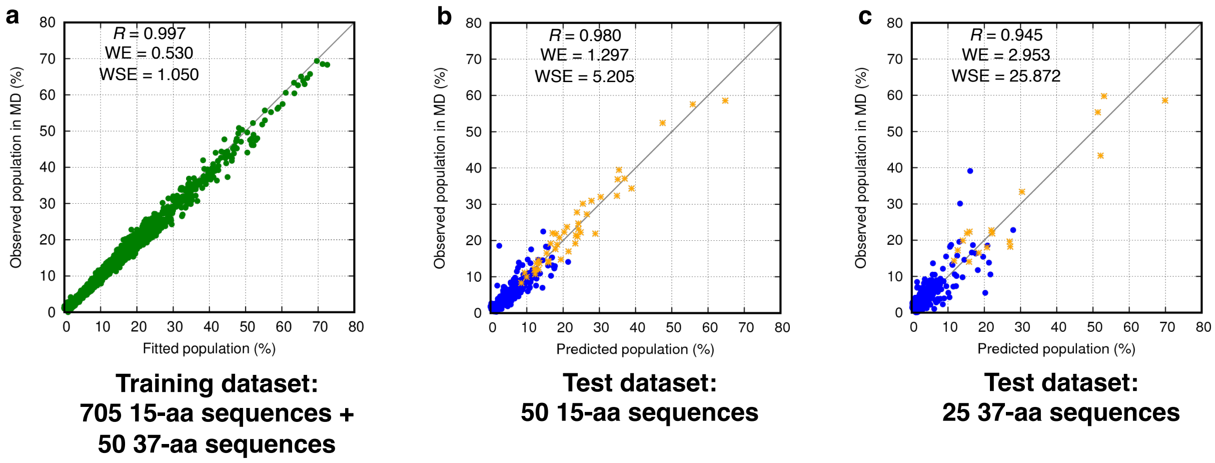


Figure S11. Performance of StrEAMM GNN/random37. a, Comparison between the fitted populations and the actual populations of various structures observed in the MD simulations of the training dataset (705 sequences in Dataset 3 including 15 amino acids, plus another 50 random sequences in Dataset 6.1 (**List S5**) including 37 amino acids). Data for enantiomers and cyclically permuted sequences are not shown, and only structures with observed populations or fitted populations >1% are shown. **b,** Comparison between the populations predicted by **StrEAMM GNN/random37** and the actual populations of various structures observed in the MD simulations of 50 random test sequences (Dataset 4, **List S2**); only structures with observed populations or predicted populations >1% are shown. The model successfully predicts the most-populated structures of 43 out of the 50 cyclic peptides in the test dataset, and these structures are shown as orange stars. **c,** Comparison between the populations predicted by **StrEAMM GNN/random37** and the actual populations of various structures observed in the MD simulations of another 25 random test sequences including 37 amino acids (Dataset 6.2, **List S5**); only structures with observed populations or predicted populations >1% are shown. The model successfully predicts the most-populated structures of 17 out of the 25 cyclic peptides in the test dataset, and these structures are shown as orange stars. Pearson correlation coefficient (*R*), weighted error (WE), and weighted squared error (WSE) were calculated. Gray lines show where the fitted/predicted populations equal the observed populations in MD simulations.

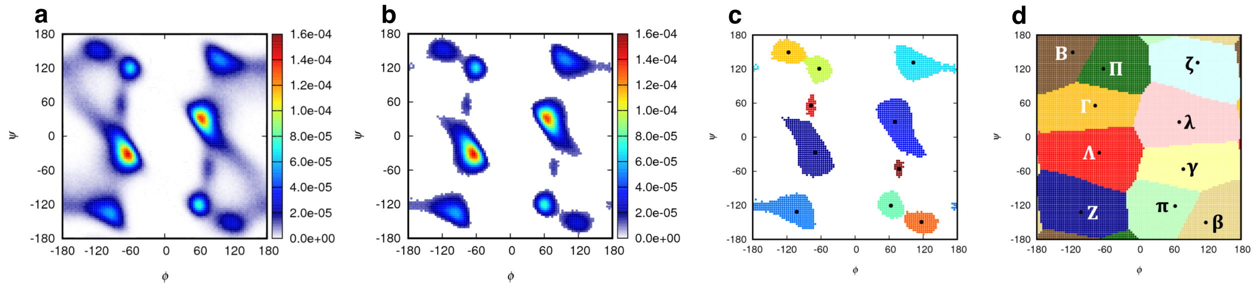


Figure S12. The Ramachandran plot is divided into 10 regions for structural description. a, The total probability distribution of (ϕ, ψ) of cyclo-(GGGGG). The plot is the same as **Fig. 6a** of the main text except that grids with the lowest densities are colored white. **b,** Only grid points with a probability density larger than 0.00001 are shown and used for further cluster analysis. **c,** The grids in **b** are grouped into 10 clusters. The centroid of each cluster is marked by black dots. **d,** All the grid points in the Ramachandran plot are assigned to their closest centroid, forming 10 regions: Λ, λ, Γ, γ, B, β, Π, π, Z, and ζ.

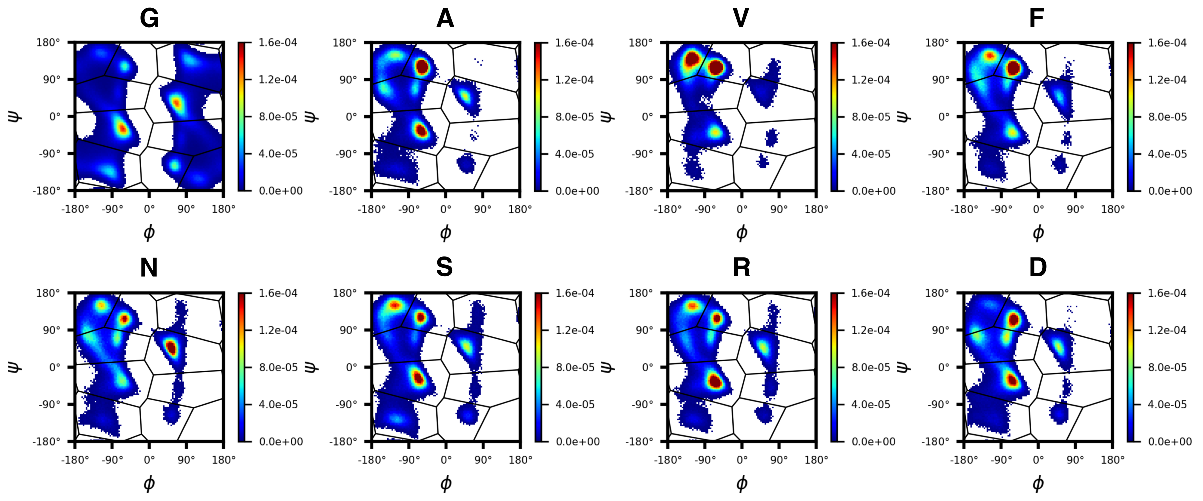


Figure S13. Universality of the binning map in Fig. S12d. The (ϕ, ψ) distributions for G, A, V, F, N, S, R, and D are from cyclo-(GGGGG), cyclo-(AGGGG), cyclo-(VGGGG), cyclo-(FGGGG), cyclo-(NGGGG), cyclo-(SGGGG), cyclo-(RGGGG), and cyclo-(DGGGG), respectively. The boundaries of the binning map are overlaid on each Ramachandran plot. Ramachandran plots of D-amino acids are not shown because their distribution is center-symmetric with that of the corresponding L-amino acids about origin (0°, 0°).

Cyclo-(GNSRV)

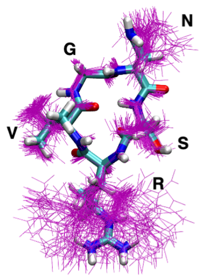
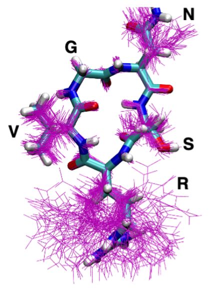
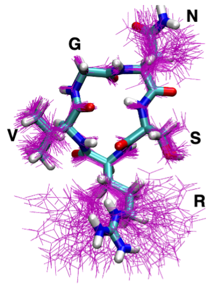
Structure			
Structural digits	πΛΖΛΒ	πΓΖΛΒ	πΛΒγΒ
Actual population in MD	50.5%	7.8%	5.1%
Scoring Function 1.0	0.755	0.057	0.014
StrEAMM (1,2)/sys	35.5%	8.9%	2.2%
StrEAMM (1,2)+(1,3)/sys	49.4%	17.4%	1.6%
StrEAMM (1,2)+(1,3)/random	41.2%	10.5%	6.2%

Figure S14. Comparison of performance of Scoring Function 1.0 and the StrEAMM models on cyclo-(GNSRV). Cyclo-(GNSRV) is a well-structured cyclic peptide predicted by Slough *et al.*[17] The three most-populated structures are shown, with a representative conformation shown in sticks and 100 randomly selected conformations shown in magenta lines. The actual populations observed in the MD simulations are given and compared to the predictions made by Scoring Function 1.0 and **StrEAMM (1,2)/sys**, **StrEAMM (1,2)+(1,3)/sys**, and **StrEAMM (1,2)+(1,3)/random**.

References

1. E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng and T. E. Ferrin, *J. Comput. Chem.*, 2004, **25**, 1605-1612.
2. M. J. Abraham, T. Murtola, R. Schulz, S. Páll, J. C. Smith, B. Hess and E. Lindahl, *SoftwareX*, 2015, **1-2**, 19-25.
3. G. A. Tribello, M. Bonomi, D. Branduardi, C. Camilloni and G. Bussi, *Comput. Phys. Commun.*, 2014, **185**, 604-613.
4. S. M. McHugh, J. R. Rogers, H. Yu and Y. S. Lin, *J. Chem. Theory Comput.*, 2016, **12**, 2480-2488.
5. J. M. Damas, L. C. Filipe, S. R. Campos, D. Lousa, B. L. Victor, A. M. Baptista and C. M. Soares, *J. Chem. Theory Comput.*, 2013, **9**, 5148-5157.
6. F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, *IEEE Trans. Neural Networks*, 2009, **20**, 61-80.
7. J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, Neural Message Passing for Quantum Chemistry, Sydney, Australia, 2017.
8. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. M. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. J. Bai and S. Chintala, PyTorch: An Imperative Style, High-Performance Deep Learning Library, Vancouver, Canada, 2019.
9. M. Fey and J. E. Lenssen, 2019, arXiv:1903.02428.
10. H. L. Morgan, *J. Chem. Doc.*, 1965, **5**, 107-113.
11. G. Landrum, RDKit: Open-source cheminformatics, http://www.rdkit.org, 2021.
12. D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742-754.
13. C. K. Schissel, S. Mohapatra, J. M. Wolfe, C. M. Fadzen, K. Bellovoda, C. L. Wu, J. A. Wood, A. B. Malmberg, A. Loas, R. Gomez-Bombarelli and B. L. Pentelute, *Nat. Chem.*, 2021, **13**, 992-1000.
14. M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov and M. Welling, Modeling Relational Data with Graph Convolutional Networks, Heraklion, Crete, Greece, 2018.
15. D. P. Kingma and J. Ba, 2014, arXiv:1412.6980.
16. A. Rodriguez and A. Laio, *Science*, 2014, **344**, 1492-1496.
17. D. P. Slough, S. M. McHugh, A. E. Cummings, P. Dai, B. L. Pentelute, J. A. Kritzer and Y. S. Lin, *J. Phys. Chem. B*, 2018, **122**, 3908-3919.