A Probabilistic Programming Approach to Protein Structure Superposition

Lys Sanz Moreta[1*], Ahmad Salim Al-Sibahi[2*], Douglas Theobald[3], William Bullock[4], Basile Nicolas Rommes[4], Andreas Manoukian[4], and Thomas Hamelryck[1,4*]

[1] Department of Computer Science. University of Copenhagen, Denmark
[2] Department of Computer Science. University of Copenhagen/Skanned.com, Denmark
[3] Department of Biochemistry. Brandeis University. Waltham, MA 02452, USA
[4] The Bioinformatics Centre. Section for Computational and RNA Biology. University of Copenhagen. Copenhagen, Denmark

[*] Corresponding authors.
moreta@di.ku.dk (Lys Sanz Moreta),
ahmad@di.ku.dk (Ahmad Salim Al-Sibahi),
theobal@brandeis.edu (Douglas Theobald),
thamelry@binf.ku.dk (Thomas Hamelryck)

***Abstract*—Optimal superposition of protein structures or other biological molecules is crucial for understanding their structure, function, dynamics and evolution. Here, we investigate the use of probabilistic programming to superimpose protein structures guided by a Bayesian model. Our model THESEUS-PP is based on the THESEUS model, a probabilistic model of protein superposition based on rotation, translation and perturbation of an underlying, latent mean structure. The model was implemented in the probabilistic programming language Pyro. Unlike conventional methods that minimize the sum of the squared distances, THESEUS takes into account correlated atom positions and heteroscedasticity (ie. atom positions can feature different variances). THESEUS performs maximum likelihood estimation using iterative expectation-maximization. In contrast, THESEUS-PP allows automated maximum a-posteriori (MAP) estimation using suitable priors over rotation, translation, variances and latent mean structure. The results indicate that probabilistic programming is a powerful new paradigm for the formulation of Bayesian probabilistic models concerning biomolecular structure. Specifically, we envision the use of the THESEUS-PP model as a suitable error model or likelihood in Bayesian protein structure prediction using deep probabilistic programming.**

***Index Terms*—protein superposition, Bayesian modelling, deep probabilistic programming, protein structure prediction**

I. Introduction

In order to compare biomolecular structures, it is typically necessary to superimpose them onto each other in an optimal way. The standard method minimizes the sum of the squared distances (root mean square deviation, RMSD) between the matching atom pairs. This can be easily accomplished by shifting the centre of mass of the two proteins to the origin and obtaining the optimal rotation using singular value decomposition [1] or quaternion algebra [2], [3]. These methods however typically assume that all atoms have equal variance (heteroscedasticity) and are uncorrelated. This is potentially problematic, for example in the case of proteins with flexible loops or flexible terminal regions, which correspond to atoms positions with high variance. Here we present a Bayesian model that is based on the previously reported THESEUS model [4], [5]. The THESEUS model is a probabilistic model of protein superposition that allows for regions with low and high variance, corresponding respectively to conserved and variable regions [4], [5]. THESEUS assumes that the structures which are to be superimposed are translated, rotated and perturbed observations of an underlying latent, mean structure M.

In contrast to the THESEUS model which features maximum likelihood parameter estimation using iterative expectation maximization, we formulate a Bayesian model (THESEUS-PP) and perform maximum a-posteriori parameter estimation. We provide suitable prior distributions over the rotation, the translations, the variances and the latent, mean model. We implement the entire model in the probabilistic programming language Pyro [6], and make use of its automated inference features.

The results indicate that deep probabilistic programming readily allows the implementation, estimation and deployment of advanced non-Euclidean models relevant to structural bioinformatics. Specifically, we envision that THESEUS-PP can be adapted for use as a likelihood function in Bayesian protein structure prediction using deep probabilistic programming, due to its support for heteroscedasticity.

II. Methods

A. Overall model

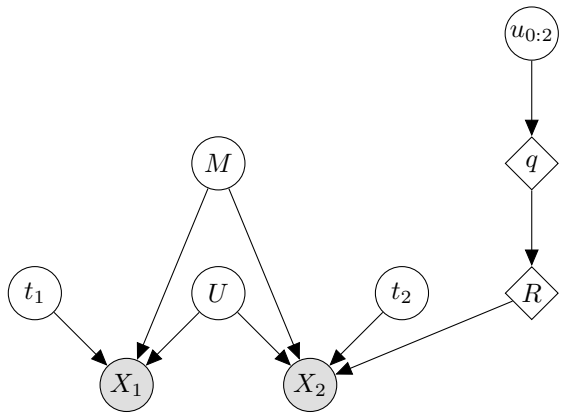


Fig. 1: The THESEUS-PP model as a Bayesian graphical model. M is the latent, mean structure, which is an N-by-3 coordinate matrix, where N is the number of atoms. t_1 and t_2 are the translations. q is a unit quaternion calculated from three random variables $u_{0:2}$ sampled from the unit interval and R is the corresponding rotation matrix. U is the among-row variance matrix of a matrix-normal distribution; X_1 and X_2 are N-by-3 coordinate matrices representing the proteins to be superimposed. Circles denote random variables; squares denote deterministic transformations of random variables. Shaded circles denote observed variables. Capital and small letters represent matrices and vectors respectively.

According to the THESEUS model, each protein structure X_i is a noisy observation of a rotated and translated latent, mean structure M,

$$X_i = R_i M + t_i + \epsilon_i, \tag{1}$$

where R_i is a rotation matrix, t_i is a three-dimensional translation and ϵ_i is the error. Another way of representing the model is seeing X_i as distributed according to a matrix-normal distribution with mean M and covariance matrices U and V - one concerning the rows and the other the columns.

The matrix-normal distribution can be considered as an extension of the standard multivariate normal distribution from vector-valued to matrix-valued random variables. Consider a random variable X distributed according to a matrix-normal distribution with mean M, which in our case is an $N \times 3$ matrix where N is the number of atoms. In this case, the matrix-normal distribution is further characterized by an $N \times N$ row covariance matrix U and a 3×3 column covariance V. Then, $X \sim \mathcal{MN}(M, U, V)$ will be distributed according to

$$X \sim M + \sqrt{U} Q \sqrt{V}, \tag{2}$$

where Q is an $N \times 3$ matrix with elements distributed according to the standard normal distribution.

To ensure identifiability, one (arbitrary) protein X_1 is assumed to be a noisy observation of the translated - but not rotated - mean structure M:

$$X_1 \sim \mathcal{MN}(M + t_1, U, V). \tag{3}$$

The other protein X_2 is assumed to be a noisy observation of the rotated as well as translated mean structure M:

$$X_2 \sim \mathcal{MN}(RM + t_2, U, V). \tag{4}$$

Thus, the model uses the same covariance matrices U and V for the matrix-normal distributions of both X_1 and X_2.

B. Bayesian posterior

The graphical model of THESEUS-PP is shown in Figure 1. The corresponding Bayesian posterior distribution is

$$\begin{aligned} &p(R, t_1, t_2, M, U | X_1, X_2) \propto \\ &p(X_1, X_2 | M, R, t_1, t_2, U) p(M) P(t_1) p(t_2) P(R) P(U) = \\ &\quad p(X_1 | M + t_1, U, I_3) p(X_2 | RM + t_2, U, I_3) P(M) \\ &\quad p(t_1) p(t_2) p(R) p(U), \end{aligned} \tag{5}$$

where I_3 is the three-dimensional identity matrix. Below, we specify how each of the priors and the likelihood function is formulated and implemented.

C. Prior for the mean structure M

Recall that according to the THESEUS-PP model, the atoms of the structures to be superimposed are noisy observations of a mean structure, $M = \{m_0, \ldots, m_{N-1}\}$, where N is the number of atoms considered in the superposition. Typically, only C_α atoms are considered and in that case, N corresponds to the number of amino acids. Hence, we need to formulate a prior distribution over the latent, mean structure M. We use an uninformative prior for M. Each element of M is sampled from a univariate normal distribution with mean 0 and variance σ_M. We set σ_M to the maximum distance of a C_α atom to the center of its structure. Finally, M is translated so its center-of-mass lies at the origin.

D. Prior over the rotation R

In the general case, we have no *a priori* information on the optimal rotation. Hence, we use a uniform prior over the space of rotations. There are several ways to construct such a uniform prior. We have chosen a method that makes use of quaternions [7]. Quaternions are the 4-dimensional extensions of the better known 2-dimensional complex numbers. Unit quaternions form a convenient way to represent rotation matrices. For our goal, the overall idea is to sample uniformly from the space of unit quaternions. Subsequently, the sampled unit quaternions are transformed into the corresponding rotation matrices, which establishes a uniform prior over rotations.

A unit quaternion $q = (w, x, y, z)$ is sampled in the following way. First, three independent random variables are sampled from the unit interval,

$$u_0, u_1, u_2 \sim U(0, 1). \tag{6}$$

Then, four auxiliary deterministic variables $(\theta_1, \theta_2, r_1, r_2)$ are calculated from u_1, u_2, u_3,

$$\theta_1 = 2\pi u_1, \tag{4a}$$
$$\theta_2 = 2\pi u_2 \tag{4b}$$
$$r_1 = \sqrt{1 - u_0}, \tag{4c}$$
$$r_2 = \sqrt{u_0}. \tag{4d}$$

The unit quaternion q is then obtained in the following way,

$$q = (w, x, y, z) = (r_2 \cos\theta_2, r_1 \sin\theta_1, r_1 \cos\theta_1, r_2 \sin\theta_2). \tag{7}$$

Finally, the unit quaternion q is transformed into its corresponding rotation matrix R as follows,

$$R = \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2(xy - wz) & 2(xz + wy) \\ 2(xy + wz) & w^2 - x^2 + y^2 - z^2 & 2(yz - wx) \\ 2(xz - wy) & 2(yz + wx) & w^2 - x^2 - y^2 + z^2 \end{bmatrix}. \tag{8}$$

E. Prior over the translations t_1 and t_2

For the translations, we use a standard trivariate normal distribution,

$$t_1, t_2 \sim \mathcal{N}_3(\mathbf{0}, I_3). \tag{9}$$

where I_3 is the identity matrix. This prior assumes that X_1, X_2 and M are centered at the origin before the translations are applied (see below).

F. Prior over U

The matrix-normal distribution has two covariance matrices, one concerning the rows (U) and another concerning the columns (V). V can be set to the identity matrix I_3. The N diagonal elements of U are sampled from the half-normal distribution with standard deviation set to 0.01,

$$\sigma_i \sim \mathcal{N}_+(0.01).$$

G. Likelihood

In our case, the matrix-normal likelihood can be formulated as a product of univariate normal distributions. Below, we have used trivariate Gaussian distributions with diagonal covariance matrices for ease of notation. The likelihood can thus be written as

$$\begin{aligned} p(X_1, X_2 \mid M, t_1, t_2, R, U) \\ &= p(X_1 \mid M, t_1, U) p(X_2 \mid M, t_2, R, U) \\ &= \prod_{i=1}^{N} \mathcal{N}_3(X_{1,i} \mid M_i + t_1, \sigma_i I_3) \\ &\qquad \times \mathcal{N}_3(X_{2,i} \mid RM_i + t_2, \sigma_i I_3), \end{aligned} \tag{10}$$

where the product runs over the matrix rows that contain the x, y, z coordinates of X_1, X_2 and the rotated and translated latent, mean structure M.

H. Algorithm

Algorithm 1 The Theseus-PP model.

▷ *Normal prior over the elements of M*
$M_{i,j} \sim \mathcal{N}(0, \sigma_M)$
▷ *Center M at origin*
$M_0 \leftarrow M - \overline{M}$
▷ *Priors over translations*
$t_1 \sim \mathcal{N}_3(\mathbf{0}, \mathbf{1})$
$t_2 \sim \mathcal{N}_3(\mathbf{0}, \mathbf{1})$
▷ *Prior over rotation*
$u_i \sim U[0, 1]$, for i from 0 to 2
$q \leftarrow \text{Quaternion}(u)$
$R \leftarrow \text{RotationMatrix}(q)$
▷ *Prior over diagonal covariance matrix U*
$U_{i,i} \sim \mathcal{N}_+(0.01)$, for i from 0 to $N - 1$
▷ *Matrix-Normal likelihood*
$X_1 \sim \mathcal{MN}(M_0 + t_1, U, I_3)$
$X_2 \sim \mathcal{MN}(RM_0 + t_2, U, I_3)$

I. Initialization

Convergence of the MAP estimation can be greatly improved by selecting suitable starting values for certain variables and by transforming the two structures in a suitable way. First, we superimpose the two structures using conventional least-squares superposition. Therefore, the starting rotation can be initialized close to the identity matrix (ie., no rotation). This is done by setting the vector u to (0.9,0.1,0.9).

Optimization can be further enhanced by initializing the mean structure M to the average of the two input structures X_1 and X_2.

J. Maximum a-posteriori optimization

We performed MAP estimation using Pyro's AutoDelta guide. For optimization, we used AdagradRMSProp [8], [9] with the default parameters for the learning rate, momentum and step size modulator. For some proteins, optimization sometimes fails to converge due to numerical instabilities (data not shown). We are investigating means to reparameterise the model in order to avoid this.

Convergence was detected using Earlystop from Pytorch's Ignite library (version 0.2.0) [10]. This method evaluates the stabilization of the error loss and stops the optimization according to the value of the *patience* parameter. The *patience* value was set to 100.

III. Materials

Proteins

The algorithm was tested on several proteins from the RCSB protein database [11] that were obtained from Nuclear Magnetic Resonance (NMR) experiments. Such structures typically contain several models of the same protein. These models represent the structural dynamics of the protein in an aqueous medium and thus typically contain both conserved and variable regions. This makes them challenging targets

for conventional RMSD superposition. We used the following structures: 1ADZ, 1AHL, 1AK7, 2KHI and 2LKL.

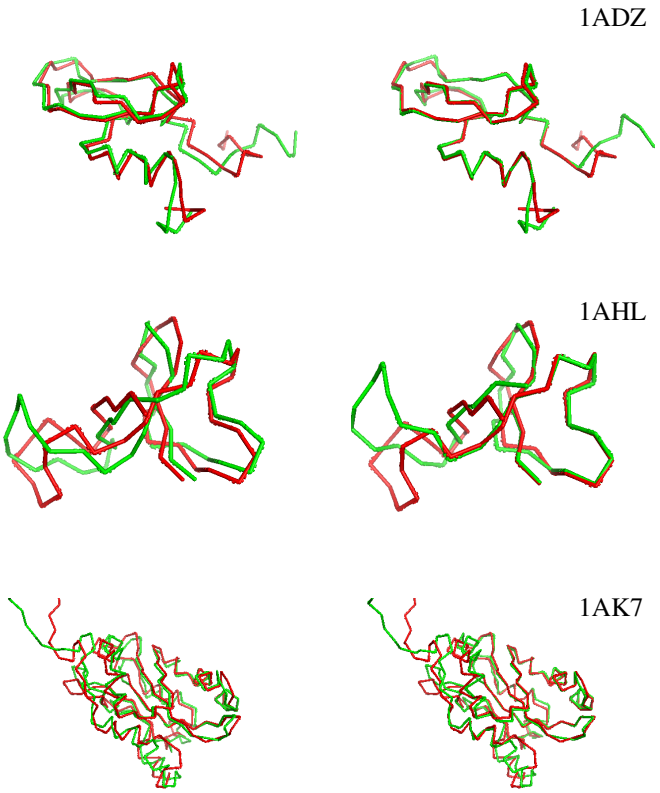
IV. Results

The algorithm was executed 10 times on each protein (see Table I).

TABLE I: Results of applying THESEUS-PP to the test structures. First column: PDB identifier. Second column: the number of C_α atoms used in the superposition. Third column: the model identifiers. Fourth column: mean convergence time and standard deviation. Last column: Number of iterations.

PBD ID	**Length (Amino Acids)**	**Protein Models**	**Average Computational Time (seconds)**	**Iterations**
1ADZ	71	0 and 1	3.3 $\pm$ 0.95	663 $\pm$121
1AHL	49	0 and 2	3.75$\pm$ 0.42	626 $\pm$59
1AK7	174	0 and 1	4.2$\pm$ 0.35	626 $\pm$109
2KHI	95	0 and 1	8.95$\pm$9.59	952$\pm$1121
2LKL	81	0 and 8	8.05$\pm$ 7.13	1403$\pm$1354

The resulting superimposed structures are shown in Figure 2. For comparison, conventional RMSD superpositions, obtained using Biopython [12], are shown on the left. THESEUS-PP superpositions are shown on the right. Note how the former fail to adequately distinguish regions with high from regions with low variance, resulting in poor matching of conserved regions.



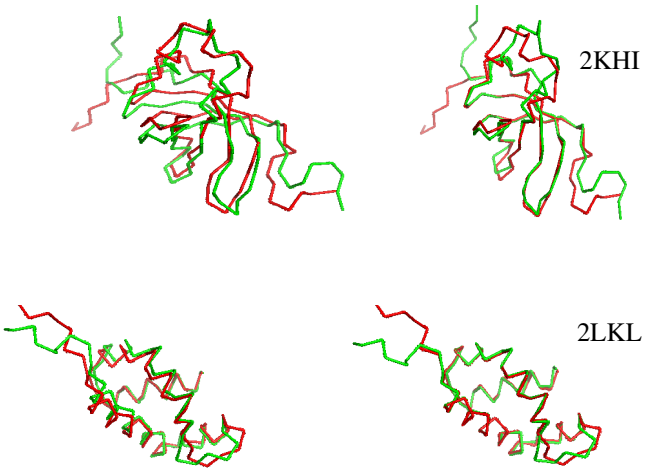
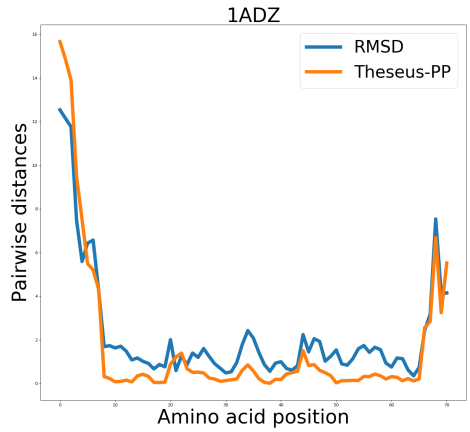
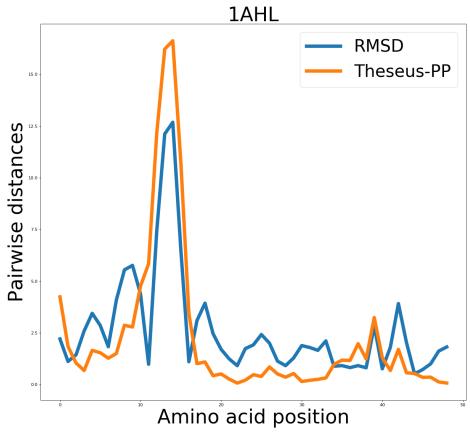


Fig. 2: Protein pairs obtained with conventional RMSD superimposition (left) and with THESEUS-PP (right).The protein in green is rotated (X_2). The images are generated with PyMOL [13].





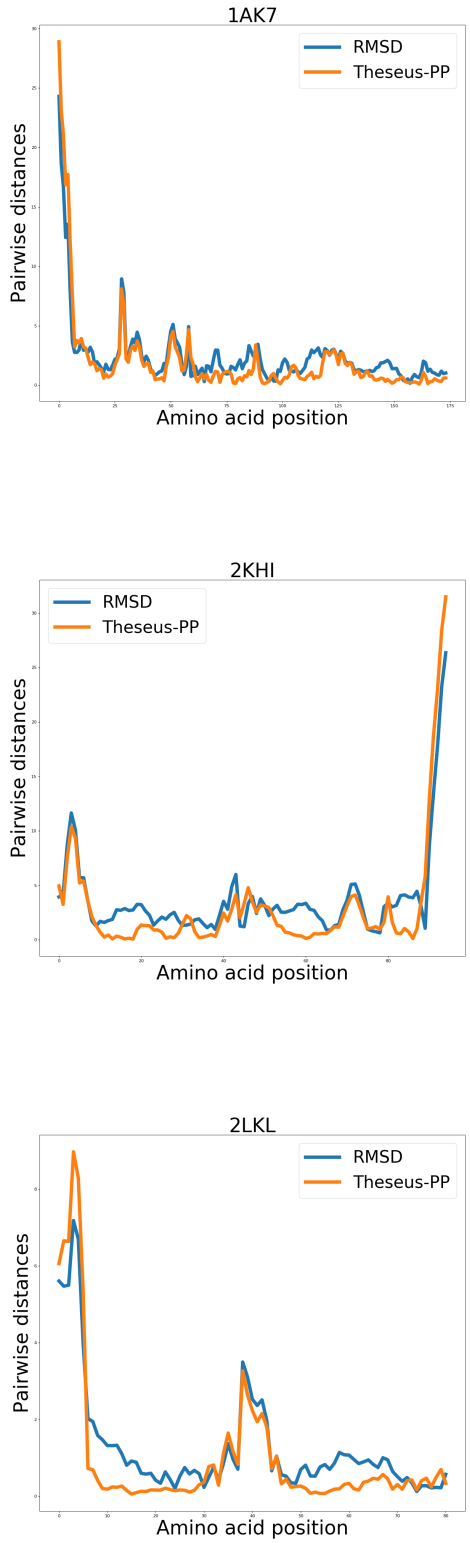


Fig. 3: Graphs showing the pairwise distances (in Å) between the C_α coordinates of the structure pairs. The blue and orange lines represent RMSD and THESEUS-PP superposition, respectively.

V. Conclusion

Probabilistic programming is a powerful, emerging paradigm for probabilistic protein structure analysis, prediction and design. Here, we present a Bayesian model for protein structure superposition implemented in the deep probabilistic programming language Pyro and building on the previously reported THESEUS maximum likelihood model. MAP estimates of its parameters are readily obtained using Pyro's automated inference engine. Recently, end-to-end protein protein structure prediction using deep learning methods has become possible [14]. We envision that Bayesian protein structure prediction will soon be possible using a deep probabilistic programming approach, which will lead to protein structure predictions with associated statistical uncertainties. In order to achieve this goal, suitable error models and likelihood functions need to be developed and incorporated in these models. The THESEUS-PP model can potentially serve as such an error model, by interpreting M as the predicted structure and a single rotated and translated X as the observed protein structure. During training of the probabilistic model, regions in M that are wrongly predicted can be assigned high variance, while correctly predicted regions can be assigned low variance. Thus, it can be expected that an error model based on THESEUS-PP will make estimation of these models easier, as the error function can more readily distinguish between partly correct and entirely wrong predictions, which is notoriously difficult for RMSD-based methods [15].

Contributions and Acknowledgements

Implemented algorithm in Pyro: *LSM*. Contributed code: *ASA*, *AM*. Wrote article: *LSM*, *TH*, *ASA*. Prototyped algorithm in pyMC3: *TH*, *WB*, *BNR*. Performed experiments: *LSM*. Designed experiments: *TH*, *DT*. *LSM* and *ASA* acknowledge funding from the Independent Research Fund Denmark (DFF-FTP) and Innovationsfonden, respectively.

References

[1] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Cryst. A*, vol. 34, pp. 827–828, 1978.

[2] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, pp. 629–642, 1987.

[3] E. Coutsias, C. Seok, and K. Dill, "Using quaternions to calculate rmsd," *J. Comp. Chem.*, vol. 25, pp. 1849–1857, 2004.

[4] D. L. Theobald and D. S. Wuttke, "Empirical Bayes hierarchical models for regularizing maximum likelihood estimation in the matrix Gaussian Procrustes problem," *PNAS*, vol. 103, pp. 18 521–18 527, 2006.

[5] D. L. Theobald and P. A. Steindel, "Optimal simultaneous superpositioning of multiple structures with missing data," *Bioinformatics*, vol. 28, pp. 1972–1979, 2012.

[6] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman, "Pyro: Deep Universal Probabilistic Programming," *Journal of Machine Learning Research*, 2018.

[7] X. Perez-Sala, L. Igual, S. Escalera, and C. Angulo, "Uniform sampling of rotations for discrete and continuous learning of 2D shape models," in *Robotic Vision: Technologies for Machine Learning and Vision Applications*. IGI Global, 2013, pp. 23–42.

[8] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[9] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[10] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[11] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The protein data bank," *Nucleic acids research*, vol. 28, no. 1, pp. 235–242, 2000.

[12] P. J. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski *et al.*, "Biopython: freely available python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, 2009.

[13] Schrödinger, LLC, "The PyMOL molecular graphics system, version 1.8," November 2015.

[14] M. AlQuraishi, "End-to-end differentiable learning of protein structure," *Available at SSRN 3239970*, 2018.

[15] I. Kufareva and R. Abagyan, "Methods of protein structure comparison," in *Homology Modeling*. Springer, 2011, pp. 231–257.