A Probabilistic Programming Approach to Protein Structure Superposition

Lys Sanz Moreta[1*], Ahmad Salim Al-Sibahi[1,2*], 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] 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, Denmark

[*] Corresponding authors.
moreta@di.ku.dk (Lys Sanz Moreta),
ahmad@di.ku.dk (Ahmad Salim Al-Sibahi),
thamelry@bio.ku.dk (Thomas Hamelryck)

***Abstract*—Optimal superposition of protein structures is crucial for understanding their structure, function, dynamics and evolution. 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 deep probabilistic programming language Pyro. Unlike conventional methods that minimize the sum of the squared distances, THESEUS takes into account correlated atom positions and heteroscedasticity (i.e., 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 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 (homoscedasticity) and are uncorrelated. This is problematic in the case of proteins with flexible loops or flexible terminal regions, where the atoms can posit high variance. Here we present a Bayesian model that is based on the previously reported THESEUS model [4]–[6]. THESEUS is a probabilistic model of protein superposition that allows for regions with low and high variance (heteroscedasticity), 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 $\mathbf{M}$.

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 (MAP) parameter estimation. We provide suitable prior distributions over the rotation, the translations, the variances and the latent, mean model. We implemented the entire model in the deep probabilistic programming language Pyro [7], using its automatic 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 used as a likelihood function in Bayesian protein structure prediction using deep probabilistic programming.

II. Methods

A. Overall model

According to the THESEUS model [4], each observed protein structure $\mathbf{X}_n$ is a noisy observation of a rotated and

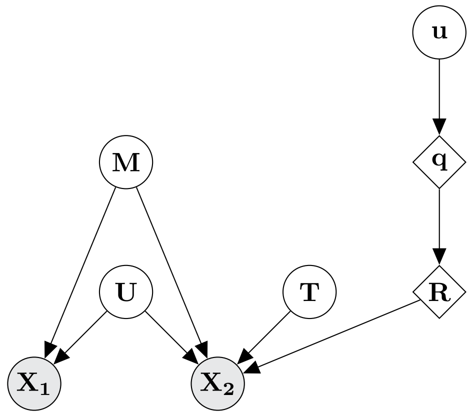


Fig. 1: The THESEUS-PP model as a Bayesian graphical model. $\mathbf{M}$ is the latent, mean structure, which is an N-by-3 coordinate matrix, where N is the number of atoms. $\mathbf{T}$ is the translation. $\mathbf{q}$ is a unit quaternion calculated from three random variables $\mathbf{u}$ sampled from the unit interval. $\mathbf{R}$ is the corresponding rotation matrix. $\mathbf{U}$ is the among-row variance matrix of a matrix-normal distribution. $\mathbf{X}_1$ and $\mathbf{X}_2$ are N-by-3 coordinate matrices representing the proteins to be superimposed. Circles denote random variables. A lozenge denotes a deterministic transformation of a random variable. Shaded circles denote observed variables. Bold capital and bold small letters represent matrices and vectors, respectively.

translated latent, mean structure $\mathbf{M}$ with noise $\mathbf{E}_n$,

$$\mathbf{X}_n = (\mathbf{M} + \mathbf{E}_n)\mathbf{R}_n - \mathbf{1}_N \mathbf{T}_n \quad (1)$$

where n is an index that identifies the protein, $\mathbf{R}$ is a rotation matrix, $\mathbf{T}$ is a three-dimensional translation, $\mathbf{E}$ is the error and $\mathbf{M}$ and $\mathbf{X}$ are matrices with the atomic coordinate vectors along the rows,

$$\mathbf{M} = \begin{bmatrix} \mathbf{m}_0 \\ \dots \\ \mathbf{m}_{N-1} \end{bmatrix}, \qquad \mathbf{X}_n = \begin{bmatrix} \mathbf{x}_{n,0} \\ \dots \\ \mathbf{x}_{n,N-1} \end{bmatrix}. \quad (2)$$

Another way of representing the model is seeing $\mathbf{X}_n$ as distributed according to a matrix-normal distribution with mean $\mathbf{M}$ and covariance matrices $\mathbf{U}$ and $\mathbf{V}$ - 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 $\mathbf{X}$ distributed according to a matrix-normal distribution with mean $\mathbf{M}$, 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 $\mathbf{U}$ and a 3×3 column covariance $\mathbf{V}$. Then, $\mathbf{X} \sim \mathcal{MN}(\mathbf{M}, \mathbf{U}, \mathbf{V})$ will be equal to

$$\mathbf{X} = \mathbf{M} + \sqrt{\mathbf{U}}\mathbf{Q}\sqrt{\mathbf{V}}, \quad (3)$$

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

To ensure identifiability, one (arbitrary) protein $\mathbf{X}_1$ is assumed to be a fixed noisy observation of the structure $\mathbf{M}$:

$$\mathbf{X}_1 \sim \mathcal{MN}(\mathbf{M}, \mathbf{U}, \mathbf{V}). \quad (4)$$

The other protein $\mathbf{X}_2$ is assumed to be a noisy observation of the rotated as well as translated mean structure $\mathbf{M}$:

$$\mathbf{X}_2 \sim \mathcal{MN}(\mathbf{M}\mathbf{R} - \mathbf{1}_N\mathbf{T}, \mathbf{U}, \mathbf{V}). \quad (5)$$

Thus, the model uses the same covariance matrices $\mathbf{U}$ and $\mathbf{V}$ for the matrix-normal distributions of both $\mathbf{X}_1$ and $\mathbf{X}_2$.

B. Bayesian posterior

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

$$\begin{aligned} &p(\mathbf{R}, \mathbf{T}, \mathbf{M}, \mathbf{U} | \mathbf{X}_1, \mathbf{X}_2) \propto \\ &\quad p(\mathbf{X}_1, \mathbf{X}_2 | \mathbf{M}, \mathbf{R}, \mathbf{T}, \mathbf{U}) p(\mathbf{M}) p(\mathbf{T}) p(\mathbf{R}) p(\mathbf{U}) = \\ &\quad\quad p(\mathbf{X}_1 | \mathbf{M}, \mathbf{U}) p(\mathbf{X}_2 | \mathbf{M}\mathbf{R} - \mathbf{1}_N\mathbf{T}, \mathbf{U}) \\ &\quad\quad\quad p(\mathbf{M}) p(\mathbf{T}) p(\mathbf{R}) p(\mathbf{U}). \end{aligned} \quad (6)$$

Below, we specify how each of the priors and the likelihood function is formulated and implemented.

C. Prior for the mean structure

Recall that according to the THESEUS-PP model, the atoms of the structures to be superimposed are noisy observations of a mean structure $\mathbf{M}$. 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 $\mathbf{M}$.

We use an uninformative prior for $\mathbf{M}$. Each element of $\mathbf{M}$ is sampled from a Student's t-distribution with degrees of freedom ($\nu = 1$), mean ($\boldsymbol{\mu} = 0$) and a uniform diagonal variance ($\boldsymbol{\sigma}^2 = 3$). The Student's t-distribution is chosen over the normal distribution for reasons of numerical stability: the fatter tails of the Student's t-distribution avoid numerical problems associated with highly variable regions.

D. Prior over the rotation

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 [8]. 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 $\mathbf{q} = (w, x, y, z)$ is sampled in the following way. First, three independent random variables are sampled from the unit interval,

$$u_0, u_1, u_2 \sim U(0, 1). \quad (7)$$

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

$$\theta_1 = 2\pi u_1, \tag{8a}$$
$$\theta_2 = 2\pi u_2 \tag{8b}$$
$$r_1 = \sqrt{1 - u_0}, \tag{8c}$$
$$r_2 = \sqrt{u_0}. \tag{8d}$$

The unit quaternion $\mathbf{q}$ is then obtained in the following way,

$$\begin{aligned}\mathbf{q} &= (w, x, y, z) \\ &= (r_2 \cos\theta_2, r_1 \sin\theta_1, r_1 \cos\theta_1, r_2 \sin\theta_2).\end{aligned} \tag{9}$$

Finally, the unit quaternion $\mathbf{q}$ is transformed into its corresponding rotation matrix $\mathbf{R}$ as follows,

$$\mathbf{R} = \begin{bmatrix} w^2+x^2-y^2-z^2 & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & w^2-x^2+y^2-z^2 & 2(yz-wx) \\ 2(xz-wy) & 2(yz+wx) & w^2-x^2-y^2+z^2 \end{bmatrix} \tag{10}$$

E. Prior over the translation

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

$$\mathbf{T} \sim \mathcal{N}(0, \mathbf{I}_3) \tag{11}$$

where $\mathbf{I}_3$ is the three-dimensional identity matrix.

F. Prior over U

The Student's t-distribution variance over the rows is sampled from the half-normal distribution with standard deviation set to 1.

$$\sigma_i \sim \mathcal{N}_+(1). \tag{12}$$

G. Likelihood

In our case, the matrix-normal likelihood of THESEUS reduces to a product of univariate Student's t-distributions. Again, we use the Student's t-distribution rather than the normal distribution (as in THESEUS) for reasons of numerical stability. Below, we have used trivariate Student's t-distributions with diagonal covariance matrices for ease of notation. The likelihood can thus be written as

$$\begin{aligned} p(\mathbf{X}_1, \mathbf{X}_2 \mid \mathbf{M}, \mathbf{T}, \mathbf{R}, \mathbf{U}) \\ &= p(\mathbf{X}_1 \mid \mathbf{M}, \mathbf{U}) p(\mathbf{X}_2 \mid \mathbf{M}, \mathbf{T}, \mathbf{R}, \mathbf{U}) \\ &= \prod_{i=1}^{N} t_1(\mathbf{x}_{1,i} \mid \mathbf{m}_i, \sigma_i \mathbf{I}_3) \\ &\quad \times t_1(\mathbf{x}_{2,i} \mid [\mathbf{M}\mathbf{R} - \mathbf{1}_N \mathbf{T}]_i, \sigma_i \mathbf{I}_3), \end{aligned} \tag{13}$$

where the product runs over the matrix rows that contain the x, y, z coordinates of $\mathbf{X}_1, \mathbf{X}_2$ and the rotated and translated latent, mean structure $\mathbf{M}$.

H. Algorithm

Algorithm 1 The Theseus-PP model.

```
▷ Prior over the elements of $\mathbf{M}$
$\mathbf{m}_i \sim t_1(\mathbf{0}, \sigma_M \mathbf{I}_3)$, where $i$ indicates the atom position
▷ Prior over the translation
$\mathbf{T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_3)$
▷ Prior over rotation
$\mathbf{u}_j \sim U[0,1]$, for $j$ from 0 to 2
$\mathbf{q} \leftarrow \text{Quaternion}(\mathbf{u})$
$\mathbf{R} \leftarrow \text{RotationMatrix}(\mathbf{q})$
▷ Prior over diagonal covariance matrix $\mathbf{U}$
$\sigma_i \sim \mathcal{N}_+(1)$
▷ Likelihood over the $N$ atom coordinates
$\mathbf{x}_{1,i} \sim t_1(\mathbf{m}_i, \sigma_i \mathbf{I}_3))$
$\mathbf{x}_{2,i} \sim t_1([\mathbf{R}\mathbf{M} - \mathbf{1}_N \mathbf{T}]_i, \sigma_i \mathbf{I}_3)$
```

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 $\mathbf{X}_1$ and $\mathbf{X}_2$ in a suitable way. First, we pre-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 $\mathbf{u}$ to $(0.9, 0.1, 0.9)$.

We further improve performance by initializing the mean structure $\mathbf{M}$ to the average of the two pre-superimposed structures $\mathbf{X}_1$ and $\mathbf{X}_2$.

J. Maximum a-posteriori optimization

We performed MAP estimation using Pyro's AutoDelta guide. For optimization, we used AdagradRMSProp [9], [10] with the default parameters for the learning rate (1.0), momentum (0.1) and step size modulator (1.0×10^{-16}). A fragment of the model implementation in Pyro can be seen in Figure 3 in the Appendix.

Convergence was detected using Earlystop from Pytorch's Ignite library (version 0.2.0) [11]. 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 25.

III. Materials

Proteins

The algorithm was tested on several proteins from the RCSB protein database [12] 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, 2CPD, 2KHI, 2LKL and 2YS9.

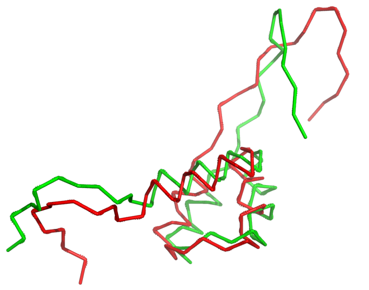
IV. Results

The algorithm was executed 15 times on each protein (see TABLE I) with different seeds. The computations where carried on a Intel Core i7-8750H CPU 2.20GHz processor.

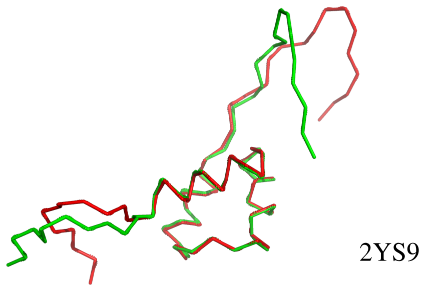
TABLE I: Results of applying THESEUS-PP to the test structures. First column: PDB identifier. Second column: the number of C_α atoms used in the superposition. Third column: the model identifiers. Fourth column: mean convergence time and standard deviation. Last column: Number of epochs.

PBD ID	**Length (Amino Acids)**	**Protein Models**	**Average Computational Time (seconds)**	**Epochs**
1ADZ	71	0 and 1	0.64± 0.15	210±45
1AHL	49	0 and 2	0.53± 0.119	166±36
1AK7	174	0 and 1	0.74± 0.23	222±67
2CPD	99	0 and 2	0.51± 0.0.093	161±30
2KHI	95	0 and 1	0.47±0.11	149±39
2LKL	81	0 and 8	0.59±0.092	187±30
2YS9	70	0 and 3	0.47±0.11	144±33

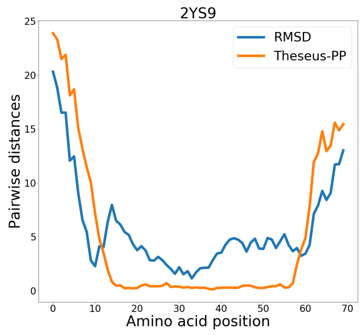
An example of a pair of superimposed structures is shown in Figure 2. For comparison, the superposition resulting from the conventional RMSD method, as calculated using Biopython [13], is shown on the left (Figure 2a). The THESEUS-PP superposition is shown on the right (Figure 2b). Note how the former fails to adequately distinguish regions with high from regions with low variance, resulting in poor matching of conserved regions. Additional, similar figures of superimposed structures can be found in the Appendix.



(a) Kabsch-RMSD



(b) Theseus-PP



(c)

Fig. 2: Protein superposition for two conformations of protein 2YS9 obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows 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.

The original THESEUS algorithm, which makes use of maximum likelihood estimation using iterative expectation maximization, is considerably faster with an average execution time under 0.1 s. Although some of the longer execution time in THESEUS-PP is due to the use of variational inference and priors, it is clear that the flexibility and productivity of a probabilistic programming language can come with a speed penalty.

Recently, end-to-end protein protein structure prediction using deep learning methods has become possible [15]. 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 $\mathbf{M}$ as the predicted structure and a single rotated and translated $\mathbf{X}$ as the observed protein structure. During training of the probabilistic model, regions in $\mathbf{M}$ 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 [16].

Contributions and Acknowledgements

Implemented algorithm in Pyro: *LSM*. Contributed code: *ASA*, *AM*. Wrote article: *LSM*, *TH*, *ASA*. Prototyped algorithm in the probabilistic programming language PyMC3 [17]: *TH*, *WB*, *BNR*. Performed experiments: *LSM*. Designed experiments: *TH*, *DT*. *LSM* and *ASA* acknowledge support from the Independent Research Fund Denmark (grant: "Resurrecting ancestral proteins in silico to understand how cancer drugs work") and Innovationsfonden/Skanned.com (grant: "Intelligent accounting document management using probabilistic programming"), respectively. We thank Jotun Hein, Michael Golden, Kanti Mardia and Wouter Boomsma for suggestions and discussions.

Data and Software Availability

Pyro [7] and PyTorch [11] based code is a available at https://github.com/LysSanzMoreta/Theseus-PP

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] K. Mardia and I. Dryden, "The statistical analysis of shape data," *Biometrika*, vol. 76, no. 2, pp. 271–281, 1989.

[7] 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.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] 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.

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

[15] M. AlQuraishi, "End-to-end differentiable learning of protein structure," *Cell Systems*, 2019.

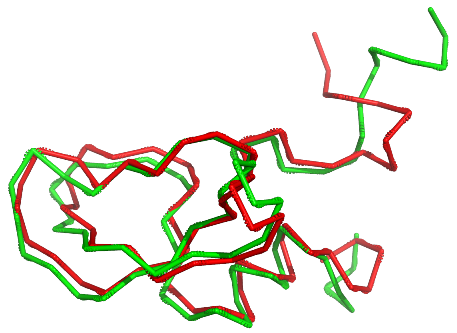
[16] I. Kufareva and R. Abagyan, "Methods of protein structure comparison," in *Homology Modeling*. Springer, 2011, pp. 231–257.

[17] J. Salvatier, T. V. Wiecki, and C. Fonnesbeck, "Probabilistic programming in python using PyMC3," *PeerJ Computer Science*, vol. 2, p. e55, 2016.

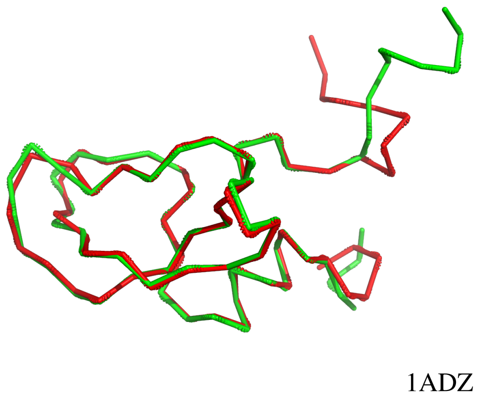
VI. Appendix

```
 # Prior over mean M, with N=number of atoms
 M = pyro.sample("M", dist.StudentT(1,0,3).expand_by([N,3]).to_event(2))
 # Prior over variances U
 U = pyro.sample("U", dist.HalfNormal(1).expand_by([N]).to_event(1))
 U =  U.reshape(N,1).repeat(1,3).view(-1)
 # Prior over translation T
 T = pyro.sample("T", dist.Normal(0,1).expand_by([3]).to_event(1))
 # Prior over rotation R
 u = pyro.sample("u",dist.Uniform(0,1).expand_by([3]).to_event(1))
 # Transformation: turn u via a unit quaternion into a rotation R
 R = u_to_quat_to_R(u)
 # Transformation: rotate and translate M for X2
 M_RT = M @ R + T
 # Likelihood
with pyro.plate("plate_students", N*3,dim= -1):
     pyro.sample("X1", dist.StudentT(1, M.view(-1), U),obs=X1)
     pyro.sample("X2", dist.StudentT(1, M_RT.view(-1), U), obs=X2)
```

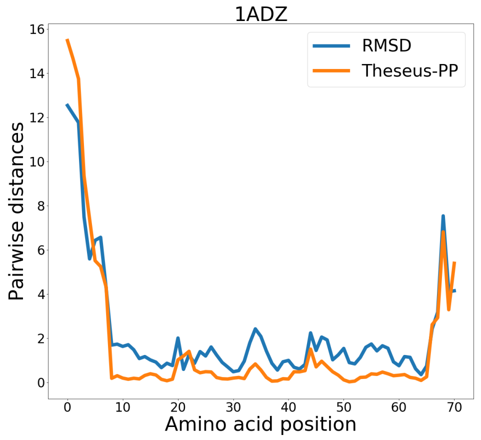
Fig. 3: Code fragment from the THESEUS-PP implementation in Pyro. *pyro.sample* calls a primitive stochastic function from which a named sample is drawn. *expand_by* specifies the shape of the batch that is to be drawn from the distribution. *pyro.plate* declares the variables within a tensor dimension as conditionally independent, while *to_event* declares them as dependent.



(a) Kabsch-RMSD

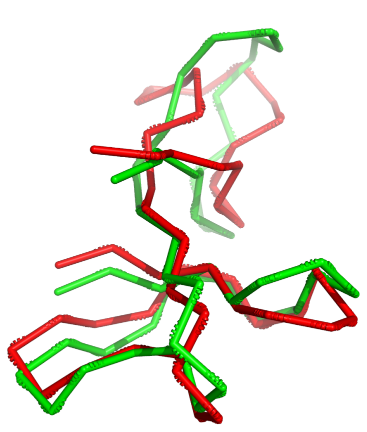


(b) Theseus-PP

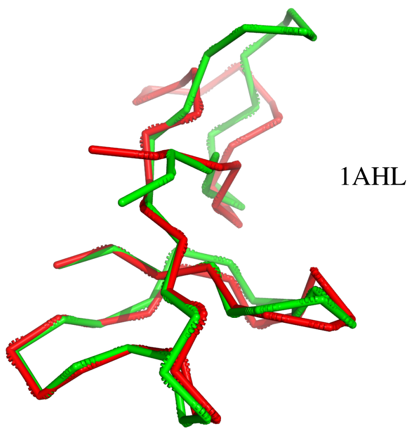


(c)

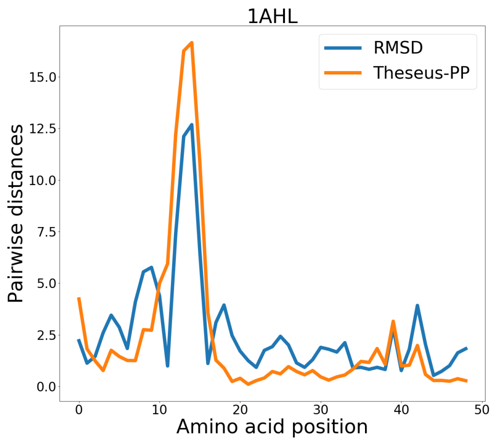
Fig. 4: Protein superposition for two conformations of protein 1ADZ obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows the pairwise distances (in Å) between the C_α coordinates of the structure pairs. The blue and orange lines represent RMSD and THESEUS-PP superposition, respectively.



(a) Kabsch-RMSD

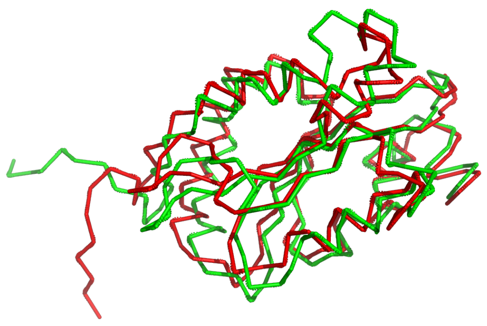


(b) Theseus-PP

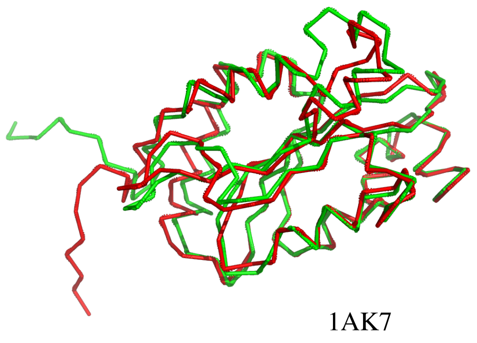


(c)

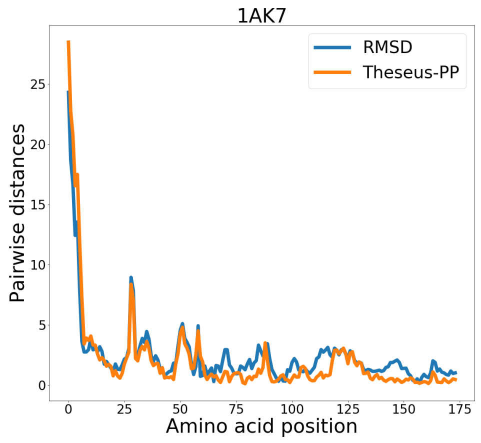
Fig. 5: Protein superposition for two conformations of protein 1AHL obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows the pairwise distances (in Å) between the C_α coordinates of the structure pairs. The blue and orange lines represent RMSD and THESEUS-PP superposition, respectively.



(a) Kabsch-RMSD

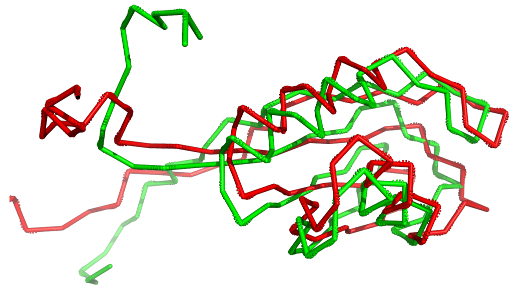


(b) Theseus-PP

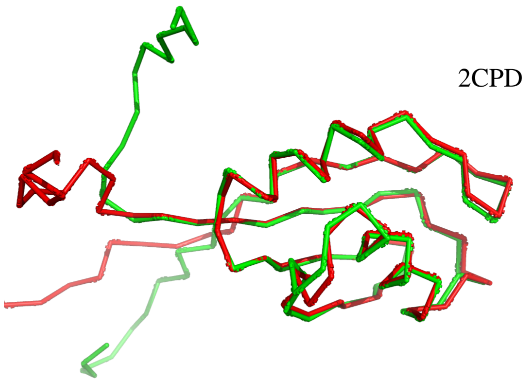


(c)

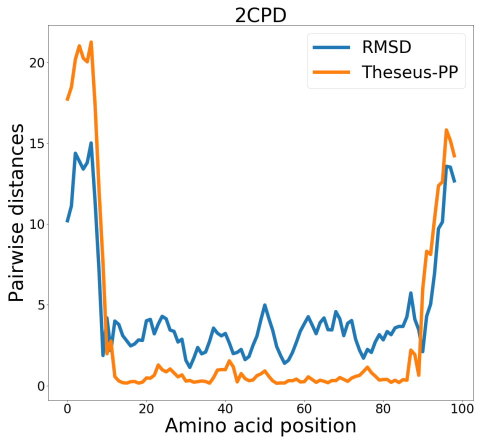
Fig. 6: Protein superposition for two conformations of protein 1AK7 obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows the pairwise distances (in Å) between the C_α coordinates of the structure pairs. The blue and orange lines represent RMSD and THESEUS-PP superposition, respectively.



(a) Kabsch-RMSD



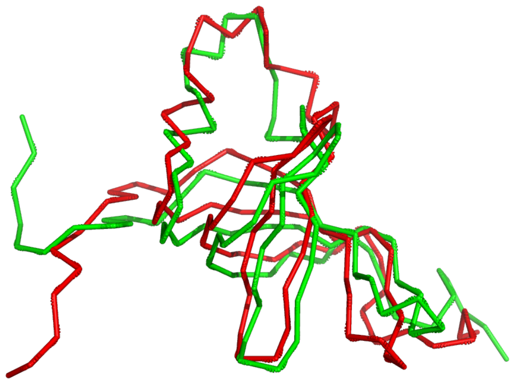
(b) Theseus-PP



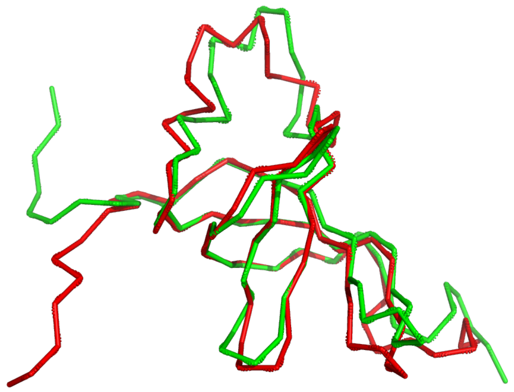
(c)

Fig. 7: Protein superposition for two conformations of protein 2CPD obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows 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.

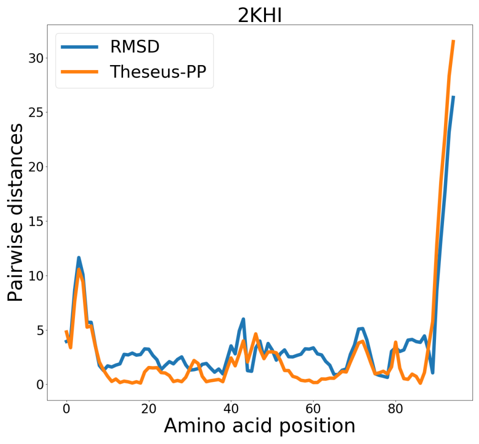
2KHI



(a) Kabsch-RMSD

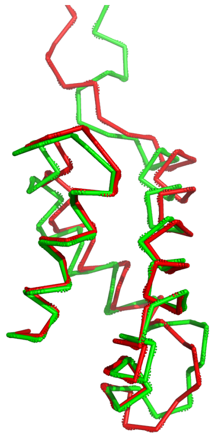


(b) Theseus-PP

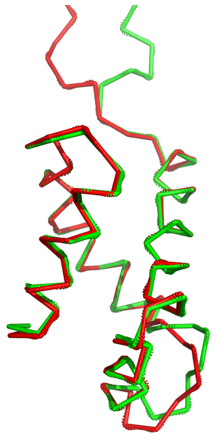


(c) Theseus-PP

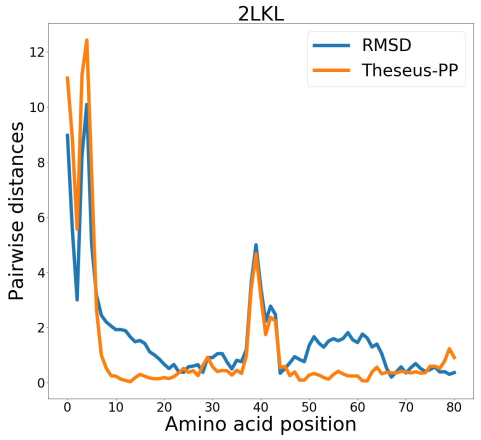
Fig. 8: Protein superposition for two conformations of protein 2KHI obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows the pairwise distances (in Å) between the C_α coordinates of the structure pairs. The blue and orange lines represent RMSD and THESEUS-PP superposition, respectively.



(a) Kabsch-RMSD



(b) Theseus-PP



(c)

Fig. 9: Protein superposition for two conformations of protein 2LKL obtained from (a) conventional RMSD superimposition and (b) THESEUS-PP. The protein in green is rotated (X_2). The images are generated with PyMOL [14]. Graph (c) shows 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.