EasyChair Preprint

№ 8695

Using a Graph Transformer Network to Predict 3D Coordinates of Proteins via Geometric Algebra Modelling

Alberto Pepe, Joan Lasenby and Pablo Chacon

August 22, 2022

Using a Graph Transformer network to predict 3D coordinates of proteins via Geometric Algebra modelling

Alberto Pepe[1][0000−0001−8775−4427], Joan Lasenby[1][0000−0002−0571−0218], and Pablo Chacón[2][0000−0002−3168−4826]

[1] Signal Processing and Communications Group
Cambridge University Engineering Department
Trumpington Street, Cambridge CB2 1PZ, UK
{ap2219,jl221}@cam.ac.uk
[2] Chacon Lab
Rocasolano Physical Chemistry Institute
28006 Madrid, Spain

Abstract. The state of the art in protein structure prediction (PSP) is currently achieved by complex deep learning pipelines that require several input features. In this paper, we demonstrate the relevance of Geometric Algebra (GA) for modelling protein features in PSP. We do so by proposing a novel GA metric based on the relative orientations of amino acid residues. We then employ this metric as an additional input feature to a Graph Transformer (GT) to aid the prediction of the 3D coordinates of a protein. Adding this GA-based orientational information improves the accuracy of the predicted coordinates even after few learning iterations and on a small dataset.

Keywords: protein structure prediction · 3D modelling · geometric algebra · graph transformer

1 Introduction

The last Critical Assessment of Protein Structure Prediction (CASP14) was won by AlphaFold 2, reaching an unprecedented global distance test (GDT) score of above 90% in almost 70% of the proteins in the CASP dataset [1]-[3]. AlphaFold 2 confirmed that deep learning (DL) is the most successful approach for PSP, and significantly cheaper and faster than experimental techniques [4]-[6].

A typical DL-based PSP pipeline is generally composed of several cascaded neural networks, whose end goal is the prediction of 3D coordinates of some of the atoms in the protein backbone [7]. In recent literature, Transformer networks have been proven to be particularly suitable for this task [1, 7, 12]. Transformer networks are sequence-to-sequence models first introduced in [8], and have found widespread application in fields including speech synthesis [9], semantic correspondence [10] and trajectory forecasting [11].

In PSP, for example, two of the seven networks employed in [7] are Transformer networks to predict and refine the coordinates of the backbone atoms, respectively. Similarly, a multiple sequence alignment (MSA) Transformer followed by a GT has been employed to predict 3D coordinates starting from the protein's sequence of amino acids in [12].

The 3D coordinates are predicted by training the network on several biological and chemical features of the protein. These features are extracted starting from its amino acid sequence (or primary structure) [14]. It has been shown that the interresidue distances (e.g. distance between amino acid pairs), the secondary structures of the proteins (e.g. the folding patterns such as helices, sheets or turns), as well as some measure of the orientation between amino acids (e.g. angle maps) are among the most relevant features when learning accurate 3D coordinates [7, 13, 14].

GA is a suitable candidate to represent the features mentioned above due to its intuitive handling of geometrical objects and operations on them [15, 16]. GA has already found some applications in protein modelling, especially in the molecular distance problem [17, 18], but to the best of our knowledge there has not been an effort to employ GA modelling for PSP.

The goal of this paper is hence to (1) employ GA to model a protein and capture information about the orientation of the amino acids and (2) use this information as a feature in a GT network. The motivations of using GA are that: (1) GA easily deals with geometrical objects such as planes, which naturally occur in the protein geometry (2) our GA feature is more compact compared with torsion and valence angles, which also grasp orientational information, but are more than one and asymmetrical, as seen in [13] and (3) it has a clear physical meaning, since it is related to secondary structures (see Section 2.1).

The rest of the paper is structured as follows: in Section 2 protein modelling through GA and graphs are presented, in Section 3 the learning architecture is introduced, in Section 4 results are presented and in Section 5 conclusions are drawn.

2 Modelling Proteins

2.1 Proteins as Rigid Bodies

The atoms in the protein backbone determine its overall shape. Each amino acid is bonded to an α-carbon (C_α), which is preceded by a nitrogen (N) atom and followed by a carbon (C) atom. Hence, there is a one-to-one correspondence between an amino acid i and a triplet $\{N, C_\alpha, C\}_i$.

Each $\{N, C_\alpha, C\}$ triplet lies on a plane. We can take advantage of this information and associate each triplet i with a plane Π_i in Conformal Geometric Algebra (CGA): let A_i, B_i and C_i be the CGA representations of the Euclidean coordinates of the atoms $\{N, C_\alpha, C\}_i$. Π_i can be then computed as the 4-blade:

$$\Pi_i = A_i \wedge B_i \wedge C_i \wedge n_\infty \tag{1}$$

where $n_\infty = e + \bar{e}$, with $e^2 = +1, \bar{e} = -1$ being two basis vector of $\mathcal{G}_{4,1,0}$ and $\wedge$ denoting the outer product.

In this way, a protein is modelled as a collection of planes not too dissimar to the gas of 3D rigid bodies of AlphaFold 2 [19] (see Fig. 1).

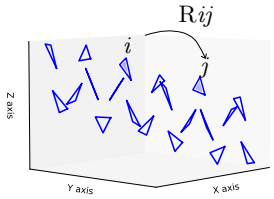

Fig. 1. A toy helix protein as a collection of planes. $\Pi_j = R_{ij}\Pi_i\tilde{R}_{ij}$

For each pair of planes Π_i, Π_j we can then form a rotor that rotates Π_i into Π_j as presented in [20]:

$$R_{ij} = \frac{1}{\sqrt{\langle \xi \rangle_0}}(1 - \Pi_i\Pi_j) \tag{2}$$

where $\xi = 2 - (\Pi_i\Pi_j + \Pi_j\Pi_i)$ and $\langle\cdot\rangle$ is the grade projector operator.

We now use the cost function $\mathscr{C}_\lambda(R)$ as defined in [21] that quantifies the variation of R from the identity. $\mathscr{C}_\lambda(R)$ is a weighted sum of a translational and a rotational term:

$$\mathscr{C}_{\lambda_1\lambda_2}(R) = \lambda_1\langle R_\parallel\tilde{R}_\parallel\rangle_0 + \lambda_2\langle(R_\perp - 1)(\tilde{R}_\perp - 1)\rangle_0 \tag{3}$$

in which the translational error is represented by $R_\parallel = R \cdot e$, and the rotational error by $\langle(R_\perp - 1)(\tilde{R}_\perp - 1)\rangle_0 = \langle(R - 1)(\tilde{R} - 1)\rangle_0$. As we are interested in an orientational feature, we will focus exclusively on the rotational part (i.e. $\lambda_1 = 0, \lambda_2 = 1$).

Since each amino acid can be associated with a plane, and each pair of planes can be associated with a rotor and eventually to a cost, we can then build an $N \times N$ matrix $\mathbf{M}$ as follows:

$$\mathbf{M}_{ij} = \begin{cases} \mathrm{C}_{\lambda_1\lambda_2}(R_{ij}) & \text{if } d_{ij} < 15 \text{ Å} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where N is the amino acid sequence length and d_{ij} is the Euclidean distance between the C_α of residues i, j measured in Å. We call $\mathbf{M}$ a "cost map". An example of a cost map is given in Fig. 2.

It is possible to establish a relationship between the secondary structure and the patterns in the cost maps. By secondary structure we refer to local folding

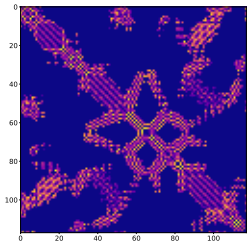

Fig. 2. Cost map for protein 2hc5 from the PDB database [24]

patterns of a protein, including α-helices, β-sheets or turns. We illustrate this relationship by assigning an arbitrary colour to each secondary structure: red to α-helices, green to β-sheets, blue to turns and white to all the others. In Fig. 3 we see how the same colour patches have almost identical cost map patterns.

To the best of our knowledge, this is the first example of a single orientational map based on GA that matches the secondary structures.

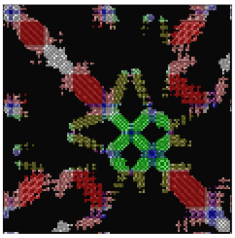

Fig. 3. Colour coded secondary structures overlapping the cost map of protein 2hc5.

2.2 Proteins as Graphs

It is also possible to represent a protein as a heterogeneous graph $\mathscr{G}(V, E)$ with V and E being its set of nodes and edges, respectively. By heterogeneous graph we refer to a graph with different types of nodes and edges. If $|V| = N$ is the total number of nodes, the graph can be described as a set of adjacency matrices for each of the K edge types, i.e. $\{A_k\}_{k=1}^{K}$, where $A_k \in \mathbb{R}^{N\times N}$, or in tensor form $\mathbf{A} \in \mathbb{R}^{N\times N\times K}$. Along with $\mathbf{A}$, we can also define a feature matrix $X \in \mathbb{R}^{N\times D}$, where D is the dimensionality of the features, or equivalently we can say there are D node types.

For our experiment, we employed the PDNET dataset and recast it in graph form [14]. PDNET is composed of a stack of 57 $N \times N$ channels for each of its proteins. We can hence associate each pairwise feature with an edge type and each per-amino acid feature with a node type. Of the 57 channels, 3 of them

correspond to 3 pairwise features (FreeCon, CCMPred and potential). To these 3 we added distance maps (defined as $\mathbf{D}_{ij} = d_{ij}$, where $d_{ij} = \|T_i - T_j\|_2$, with $T \in \mathbb{R}^{N\times 3}$ being the ground truth coordinates of the C_α atoms of the protein) and cost maps $\mathbf{M}$, to obtain a total of $K = 5$ pairwise maps of size $N \times N$, with N being the protein chain length. They correspond to the edges of the protein graph, i.e. the adjacency matrices $\mathbf{A} \in \mathbb{R}^{N\times N\times K=5}$. The remaining 54 channels are the matrices and their transposes of the remaining 4 features (SA, PSSM, SS, entropy), which are associated with a single amino acid. Ignoring the transposed matrices, we are left with 27 channels, which can be manipulated and arranged in a feature matrix $X \in \mathbb{R}^{N\times D=27}$.

The input to the architecture is then given by the pair of tensors $\{\mathbf{A}, X\}^{(i)}$ for each protein i in the dataset.

3 Architecture

The end-to-end architecture, derived from [12], is composed of two parts: (1) a GT and (2) 3D projector. A summary of the architecture is shown in Fig. 4. We omitted the MSA Transformer of [12] as the employed dataset allows us to directly perform node and edge embedding on its features.

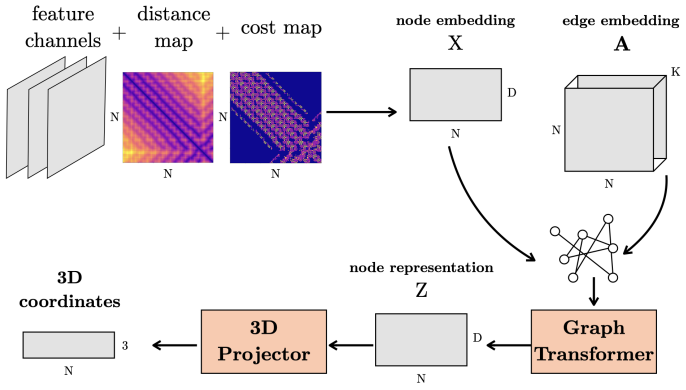

Fig. 4. The employed architecture

3.1 Graph Transformer

The GT has been implemented as described in [22]. The goal of a GT is to learn informative meta-path within the graph, i.e. an ordered sequence of node types and edge types. The output of the l-th layer of a GT with C attention heads is a node representation with same dimensionality as X, i.e. $Z \in \mathbb{R}^{N\times D}$ which can

be written as

$$Z^{(l)} = \bigoplus_{i=1}^{C} \sigma(\tilde{\Delta}_i^{-1} \tilde{A}_i^{(l)} XW) \tag{5}$$

where $\bigoplus$ is the concatenation operator, $\sigma(\cdot)$ is the sigmoid function, $\tilde{\Delta}_i$ is the degree matrix of $\tilde{A}_i^{(l)}$ (defined as $\Delta_{mm} = \sum_n A_{mn}$), X is the feature matrix, $W \in \mathbb{R}^{D \times D}$ is a trainable weight matrix and $\tilde{A}_i^{(l)} = A_i^{(l)} + I$, in which $A_i^{(l)}$ is the adjacency matrix from the i-th channel of the metapath tensor $\mathbf{A}^{(l)} \in \mathbb{R}^{N \times N \times C}$. $\mathbf{A}^{(l)}$ is evaluated as $\mathbf{A}^{(l)} = \Delta^{-1}\mathbf{Q}_1\mathbf{Q}_2$. $\mathbf{Q}_1$ and $\mathbf{Q}_2$, both $\in \mathbb{R}^{N \times N \times C}$, are two adjacency tensors selected according to $\mathbf{Q} = \varphi[\mathbf{A}; \zeta(\mathbf{W}_\varphi)]$, where $\mathbf{A} \in \mathbb{R}^{N \times N \times K}$ is the adjacency tensor, $\varphi(\cdot)$ is the convolution operator, $\zeta(\cdot)$ is the softmax function and $\mathbf{W}_\varphi \in \mathbb{R}^{C \times C \times K}$ are the weights of φ. Z contains the node representations from C different meta-path graphs.

3.2 3D Projector

The 3D projector is a simple fully connected layer obeying $P = Z^{(L)}W_P$, where $Z^{(L)}$ is the output of the L-th layer of the GT, $W_P \in \mathbb{R}^{D \times 3}$ is the weight matrix of the projector and $P \in \mathbb{R}^{N \times 3}$ are the 3D coordinates of the N C_α atoms in the protein chain.

To train the model, a distance map is evaluated for each protein from the predicted coordinates P as $\tilde{\mathbf{D}}_{ij} = d_{ij}$, where $d_{ij} = \|P_i - P_j\|_2$ is the Euclidean distance between the 3D coordinates of the i-th and j-th amino acid in P.

The total loss to minimize is equal to $\mathscr{L} = \mathscr{L}_1 + \mathscr{L}_2$. The first term *minimizes* the L_1 loss between $\mathbf{D}$ (the ground truth distance map) and $\tilde{\mathbf{D}}$, as $\mathscr{L}_1 = \frac{1}{N^2}\sum_i^N \sum_j^N \|\tilde{\mathbf{D}}_{ij} - \mathbf{D}_{ij}\|_1$. The second term *maximizes* the structural similarity index (SSIM) between $\mathbf{D}$ and $\tilde{\mathbf{D}}$ weighted by an arbitrary coefficient $\alpha = 10$ to make $\mathscr{L}_2$ of the same order of magnitude of $\mathscr{L}_1$, namely $\mathscr{L}_2 = \alpha\left(1 - SSIM\{\mathbf{D}, \tilde{\mathbf{D}}\}\right)$. The loss is measured over distance maps and not over 3D coordinates as 3D coordinates depend on a reference frame, while distances are rotationally and translationally invariant.

3.3 Training Details

We trained the model consisting of the GT and 3D projector on the PDNET dataset. The model consists of 108813 trainable parameters, of which 108648 of the GT and 165 of the projector. The train and validation sets are subsets of PDNET composed of 200 proteins each, while the test set contains 150 proteins. The optimizer has been set to Adam with exponentially decaying learning rate, with initial learning rate $\eta_0 = 1 \times 10^{-2}$ and decay rate per epoch $\gamma = 0.9$. The GT has $C = 4$ attention heads and $L = 3$ layers. The batch size has been fixed to $B = 1$ and the network has been trained for $E = 5$ epochs, for a total of 1000 training iterations.

Combinations of $\eta \in \{1 \times 10^{-1}, 1 \times 10^{-2}, 1 \times 10^{-3}, 3 \times 10^{-4}\}$, $E \in \{3, 5, 10\}$, $B \in \{1, 50, 100\}$, $L \in \{3, 6, 10\}$, $C \in \{1, 4, 5\}$ have also been implemented and tested, but the hyperparameters above were found to be optimal for our problem.

The code has been written as a Jupyter Notebook on Google Colaboratory, run on an NVIDIA Tesla K80 GPU and it uses PyTorch for the DL architecture, the Clifford library for GA operations [23] and the PDB Module of Biopython for handling protein data. The GT was derived from [12]. Scripts and datasets are available upon request to the authors.

4 Results

We trained the architecture and collected results for two cases: (1) *with* cost maps ($D = 27, K = 5$) and (2) *without* cost maps ($D = 27, K = 4$), to verify whether adding a single additional GA-based adjacency matrix A_k in our graph could provide an improvement.

From the predicted coordinates $P \in \mathbb{R}^{N \times 3}$ we constructed distance maps $\tilde{\mathbf{D}} \in \mathbb{R}^{N \times N}$, and we then measured the mean absolute error (MAE) and SSIM between $\mathbf{D}$ and $\tilde{\mathbf{D}}$. The MAE and SSIM distributions are presented in Table 1, while the distributions and percentiles over the test set are visualized in Figs. 5-6 for the MAE and the SSIM, respectively.

Table 1. Metric between original and predicted distance maps. Results without costs are in parenthesis.

Set	Metric	Max	Mean	Min	Std
Train	SSIM	0.98 (0.90)	0.88 (0.43)	0.12 (-0.10)	0.10 (0.22)
Test	SSIM	0.99 (0.86)	0.88 (0.43)	0.38 (-0.14)	0.10 (0.25)
Train	MAE (Å)	27.9 (23.3)	6.09 (7.38)	2.16 (3.32)	2.69 (3.05)
Test	MAE (Å)	10.3 (12.8)	5.99 (7.02)	2.49 (3.58)	2.35 (1.91)

Note in Table 1 how the average SSIM doubles from 0.43 when coordinates are predicted without cost maps to 0.88 when coordinates are predicted with cost maps. Similarly, the average MAE decreases by 1.29 Å and 1.03 Å on the train and test sets, respectively, when we include cost maps. From Fig. 5 it can be seen that the median MAE of the test set is found to be at about 5 Å with costs maps and at about 7 Å without cost. The improvement introduced with cost maps is even more evident in Fig. 6, in which the median SSIM of the test set is > 0.4 without cost maps and > 0.8 with cost maps.

We then aligned P and T via singular value decomposition (SVD) (see Appendix A) and performed the GDT, and evaluated the GDT_TS (total score) and GDT_HA (half size) between predicted coordinates P and ground truth coordinates T, obtained from the Protein Data Bank (PDB) [24].

$$\text{GDT\_TS} = \frac{p_{<1\text{Å}} + p_{<2\text{Å}} + p_{<4\text{Å}} + p_{<8\text{Å}}}{4} \tag{6}$$

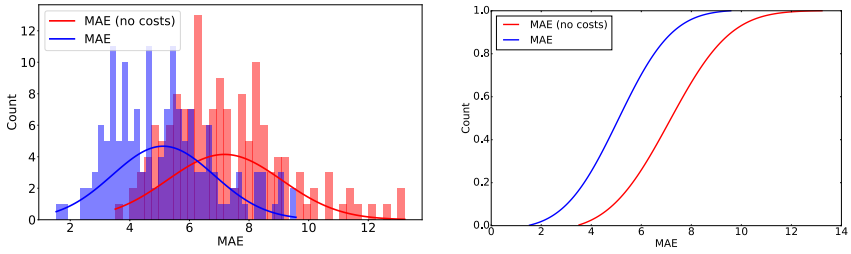

Fig. 5. MAE measured over the testing set. Distribution (left) and cumulative probability (right).

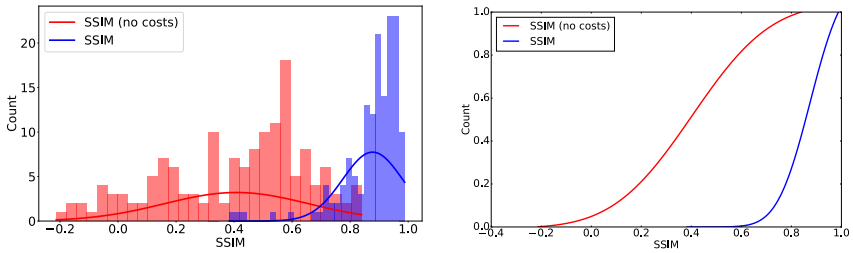

Fig. 6. SSIM measured over the testing set. Distribution (left) and cumulative probability (right).

$$\text{GDT\_HA} = \frac{p_{<0.5\text{Å}} + p_{<1\text{Å}} + p_{<2\text{Å}} + p_{<4\text{Å}}}{4} \tag{7}$$

where $p_{<n\text{Å}}$ indicates the percentage of an amino acid's coordinates in P whose distance from the corresponding amino acid's coordinates in T is below n Å.

Results for selected proteins are shown in Table 2. Note how both the GDT_TS and the GDT_HA generally increase by at least a factor of 2 when adding cost maps as an additional feature. Examples of the predicted coordinates and relative distance maps are given in Figs. 7-10.

5 Conclusions

In this paper we introduced a measure of the orientation between amino acids based on GA. We presented the ideas behind the modelling of a protein as a collection of planes, we introduced a measure of the "distance" between each pair of planes and arranged it in matrix form, i.e. a cost map.

We then employed these matrices as an additional feature in a GT + 3D projector pipeline to predict 3D coordinates of C_α atoms in proteins. We did so by adapting in graph form a dataset comprising several biochemical features already available in the literature, to which we added cost maps. Eventually, we

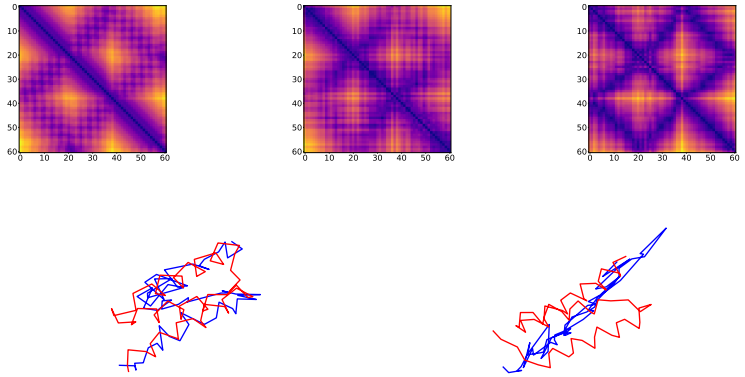

Fig. 7. Top row, from left to right: original, predicted and predicted (without cost maps) distogram for protein 2gomA. Bottom row: original (red) and predicted (blue) C_α coordinates. Left: with costs, right: without costs

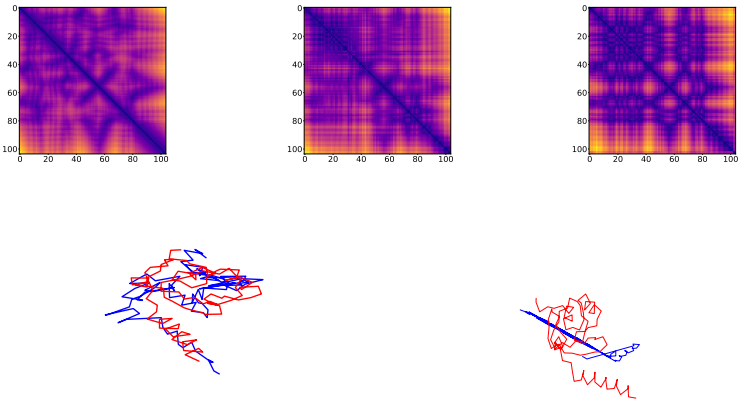

Fig. 8. Top row, from left to right: original, predicted and predicted (without cost maps) distogram for protein 1dm9A. Bottom row: original (red) and predicted (blue) C_α coordinates. Left: with costs, right: without costs

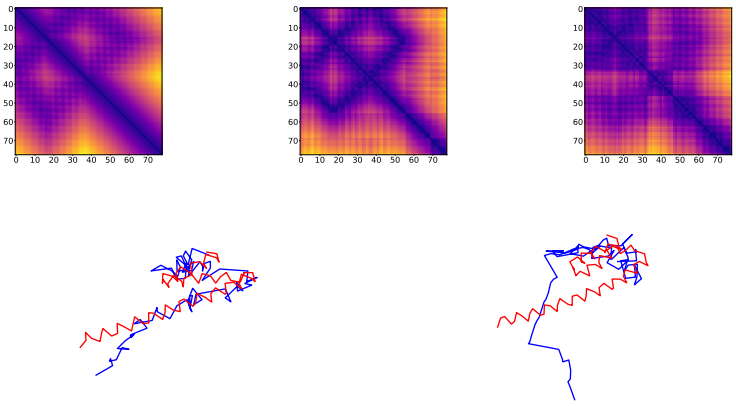

Fig. 9. Top row, from left to right: original, predicted and predicted (without cost maps) distogram for protein 2fztA. Bottom row: original (red) and predicted (blue) C_α coordinates. Left: with costs, right: without costs

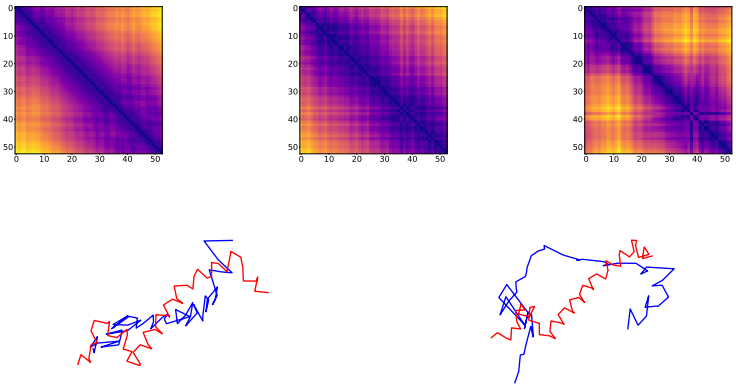

Fig. 10. Top row, from left to right: original, predicted and predicted (without cost maps) distogram for protein 2fyuK. Bottom row: original (red) and predicted (blue) C_α coordinates. Left: with costs, right: without costs

Table 2. Metrics between original and predicted coordinates with and without (-) cost maps after SVD alignment. MAE and SSIM are measured on distance maps.

Protein	MAE	SSIM	GDT_TS	GDT_HA
2gomA	2.49	0.86	31.2	9.84
2gomA (-)	5.31	0.53	13.9	4.10
1zv1A	2.83	0.96	28.8	10.2
1zv1A (-)	4.78	0.46	14.0	3.39
1dm9A	3.58	0.95	25.5	7.93
1dm9A (-)	6.94	0.36	6.25	0.48
1m8nA	3.59	0.92	22.0	7.92
1m8nA (-)	5.04	0.49	11.0	2.08
2fztA	3.65	0.96	18.6	4.49
2fztA (-)	6.81	0.55	8.96	1.60
2fyuK	3.44	0.98	15.1	3.30
2fyuK (-)	6.69	0.82	4.72	1.41

compared the 3D coordinates predicted including cost maps with coordinates predicted without them.

We showed that our GA-based cost maps aids the convergence of the model and the prediction of more accurate coordinates in terms of GDT_TS and GDT_HA scores with respect to ground truth. In addition, the distance maps constructed from the coordinates predicted including costs are closer to the original distance maps in terms of both MAE and SSIM.

Despite training the model on a dataset of only 200 short proteins and for few iterations, we managed to obtain reasonable protein structures. We are confident that including cost maps on a larger scale problem (e.g. larger training set, more learning iterations, higher dimensionality of node and edge embeddings, etc.) can constitute an asset in PSP by increasing prediction accuracy with a minimal amount of additional information.

References

1. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., ... Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. Nature, 596(7873), 583-589.
2. Thornton, J. M., Laskowski, R. A., Borkakoti, N. (2021). AlphaFold heralds a data-driven revolution in biology and medicine. Nature Medicine, 27(10), 1666-1669.
3. Perrakis, A., Sixma, T. K. (2021). AI revolutions in biology: The joys and perils of AlphaFold. EMBO reports, 22(11), e54046.
4. Torrisi, M., Pollastri, G., Le, Q. (2020). Deep learning methods in protein structure prediction. Computational and Structural Biotechnology Journal, 18, 1301-1310.
5. Kandathil, S. M., Greener, J. G., Jones, D. T. (2019). Recent developments in deep learning applied to protein structure prediction. Proteins: Structure, Function, and Bioinformatics, 87(12), 1179-1189.
6. Pakhrin, S. C., Shrestha, B., Adhikari, B., Kc, D. B. (2021). Deep learning-based advances in protein structure prediction. International Journal of Molecular Sciences, 22(11), 5553.

7. Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., ... Baker, D. (2021). Accurate prediction of protein structures and interactions using a three-track neural network. Science, 373(6557), 871-876.
8. Jaderberg, M., Simonyan, K., Zisserman, A. (2015). Spatial transformer networks. Advances in neural information processing systems, 28.
9. Li, N., Liu, S., Liu, Y., Zhao, S., Liu, M. (2019, July). Neural speech synthesis with transformer network. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 6706-6713).
10. Kim, S., Lin, S., Jeon, S. R., Min, D., Sohn, K. (2018). Recurrent transformer networks for semantic correspondence. Advances in neural information processing systems, 31.
11. Giuliari, F., Hasan, I., Cristani, M., Galasso, F. (2021, January). Transformer networks for trajectory forecasting. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 10335-10342). IEEE.
12. Costa, A., Ponnapati, M., Jacobson, J. M., Chatterjee, P. (2021). Distillation of MSA Embeddings to Folded Protein Structures with Graph Transformers. bioRxiv.
13. Yang, J., Anishchenko, I., Park, H., Peng, Z., Ovchinnikov, S., Baker, D. (2020). Improved protein structure prediction using predicted interresidue orientations. Proceedings of the National Academy of Sciences, 117(3), 1496-1503.
14. Adhikari, B. (2020). A fully open-source framework for deep learning protein real-valued distances. Scientific reports, 10(1), 1-10.
15. Doran, C., Gullans, S. R., Lasenby, A., Lasenby, J., Fitzgerald, W. (2003). Geometric algebra for physicists. Cambridge University Press.
16. Dorst, L., Doran, C., Lasenby, J. (Eds.). (2012). Applications of geometric algebra in computer science and engineering. Springer Science Business Media.
17. Lavor, C., Alves, R. (2019). Oriented conformal geometric algebra and the molecular distance geometry problem. Advances in Applied Clifford Algebras, 29(1), 1-15.
18. Alves, R., Lavor, C. (2017). Geometric algebra to model uncertainties in the discretizable molecular distance geometry problem. Advances in Applied Clifford Algebras, 27(1), 439-452.
19. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Tunyasuvunakool, K., ... Hassabis, D. (2020). AlphaFold 2.
20. Lasenby, J., Hadfield, H., Lasenby, A. (2019). Calculating the rotor between conformal objects. Advances in Applied Clifford Algebras, 29(5), 1-9.
21. Eide, E.R., Master's Degree Thesis, University of Cambridge, Camera Calibration using Conformal Geometric Algebra, 2018
22. Yun, S., Jeong, M., Kim, R., Kang, J., Kim, H. J. (2019). Graph transformer networks. Advances in neural information processing systems, 32.
23. Hadfield H., Wieser E., Arsenovic A., Kern R., The Pygae Team. pygae/clifford.
24. Burley, S. K., Berman, H. M., Kleywegt, G. J., Markley, J. L., Nakamura, H., Velankar, S. (2017). Protein Data Bank (PDB): the single global macromolecular structure archive. Protein Crystallography, 627-641.