An Evaluation of H-CNN Protein Energy Predictions Using Normal Mode Analysis

Sylvie Shaya[1] and Natasha Wozniak[2]

[1]Department of Physics, Wellesley College
[2]Department of Physics, Skidmore College

August 2024

Abstract

Proteins are a critical component of biological systems, yet predicting protein function from structure is a major outstanding challenge in biophysics. Protein function is tied to protein dynamics and motion. The Holographic Convolutional Neural Network (H-CNN) is a physically motivated machine learning model developed to predict amino acid identities based on the surrounding protein structure. The outputs of H-CNN appear to behave similarly to potential energies. It is possible to predict oscillatory motions of a system from the potential energy function using normal mode analysis (NMA). We apply NMA to both H-CNN and an elastic network model to benchmark H-CNN's ability to predict the oscillatory modes of motion of proteins. Although H-CNN is an effective tool for predicting the impact of mutations on protein structure, it does not appear to realistically predict to motions of proteins.

1 Introduction

Proteins are molecules that form the underpinnings of biological function, driving enzymatic reactions, molecular transport, signaling, and immune protection. The arrangement of twenty amino acids in various configurations produces thousands of unique proteins that carry out biological functions. The sequence of amino acids determines the microscopic structure which in turn determines the macroscopic properties and ultimately the function and utility of a given protein(Alberts et al.).
Chains of amino acids fold into complex protein structures due to atomic interactions. The amino acid sequence is called the primary structure of the protein and forms the backbone of the protein. Secondary structure emerges due to interactions backbone amino acids, while tertiary structure is overall three-dimensional structure of the protein typically determined by interactions involving R groups, or amino acid side chains. Some proteins consist of multiple polypeptide chains, or long amino acid chains, which

Figure 1: Protein G, the protein used throughout this paper, represented using BioPython visualization tools

interact to form a quaternary structure(Alberts et al.). The relationship between amino acid sequence and structure has been well studied. A number of machine learning algorithms have been trained to predict protein structure from sequence, most famously AlphaFold. This research works to learn more about the relationship between structure and function in proteins using a neural network(Pun et al.).
Protein dynamics play a significant role in protein function. Learning how structure impacts dynamics helps bridge the gap in understanding the structure-function relationship (Dykeman and Sankey). A deeper understanding of this relationship would allow for biological advances, including reverse engineering proteins for desired functions.
(Bahar et al.) uses a simple model to predict the energy of a molecule. It assumes each bond to be a spring in an elastic network. With spring constants and positions of the atoms, the energy of each bond can be calculated using Hooke's law: $PE_s = \frac{1}{2}k\Delta x^2$. This energy is then summed over each connection in the molecule. This approach has been found to be largely successful in predicting the potential energy and lowest energy positions of molecules. The Hessian matrix for the energy of a protein provides valuable information about the motion of the protein. The eigenvectors of the Hessian provide normal modes of motion, with frequencies represented by eigenvalues(Bahar et al.).
The Holographic Convolutional Neural Network (H-CNN) is a model developed by the Nourmohammad group that predicts an amino acid's identity based on its surroundings. The predictions have been found to mimic the behavior of the free energy contribution of the amino acid. When the free energies of all residues in a protein are summed, the neural network behaves like a potential energy function. This has been tested by perturbing each residue and finding the potential energy map near the wildtype of the protein(Pun et al.). The extent to which this physical interpretation of the neural network holds can be tested through normal mode analysis (NMA).
NMA uses the potential energy of the system to define a new coordinate system made up of the normal oscillating modes of the system. NMA assumes that low-energy movements will follow oscillating patterns and that the lowest frequency modes will describe common protein behaviors(Dykeman and Sankey). 6LYZ, a protein in egg whites, has been found to have a prominent hinging motion that can be predicted by the lowest energy normal mode (Bauer et al.).

1.1 Elastic Network Models

Protein motion depends directly on structure, as the intermolecular and intramolecular bonds governing protein topology depend on overall 3-D structure. Elastic Network Models (ENMs) reduce the complex structure of proteins to a set of springs and nodes. Fig. 2 shows a toy example of an ENM, where a set of nodes is connected via a set of springs. Each spring has an associated spring constant, and from this informationit is possible to write a potential energy function describing the system (Zanin et al.).
In an ENM that models a protein, nodes can represent various levels of protein structure, including atoms, amino acids, or chains. The connections between nodes are often determined by a distance cut-off, where any two nodes within a given separation are connected. The bond potential energy between nodes in an ENM can be modeled as a harmonic potential around equilibrium or as a spring. The spring constants for each harmonic oscillator might depend on the type of atoms or

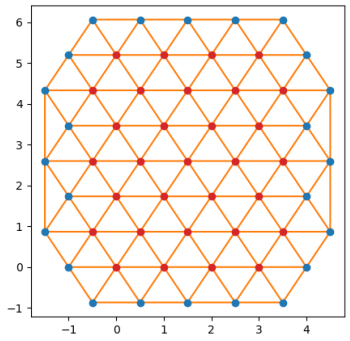

Figure 2: ENM toy example

residues they connect, although there are many models for spring constants.
ENMs are highly effective models for exploring the the vibrational modes of proteins. The global dynamics they predict are comparable to those predicted by models analyzing full atomic force fields (Bahar et al.). Throughout this research, we use an ENM from the ProDy python package to benchmark our predictions of the vibrational modes of proteins.

1.2 Normal Mode Analysis

Normal mode analysis (NMA) is a mathematical tool used to understand oscillatory motions. NMA is used in various fields of physics, including quantum mechanics, optics, and molecular dynamics. In biophysics, NMA is applied to protein models in order to analyze oscillatory motions of proteins. In this work, we apply NMA to an ENM and H-CNN's potential energy-like outputs in order to understand if H-CNN outputs reflect realistic potential energies(Dykeman and Sankey).
NMA begins with a calculation of total energy, which in an ENM looks like a sum of the elastic energy of each bond.

$$U = \sum_{i,j} 1/2 k_{ij}(x_{ij} - x_{0ij})^2 \tag{1}$$

Where k is the spring constant and a sum is taken over the elastic potential energy between every pair of bonded nodes i and j, when $\Delta(x_{ij})$ is the change in bond length from equilibrium. The presence or lack of a bond is decided by a distance cutoff. The Hessian of the energy is taken with respect to each degree of freedom of the system.

$$H = \frac{\partial^2 U}{\partial x_i \partial x_j} \tag{2}$$

The system can be rewritten as an eigenvector eigenvalue problem assuming the system can be well modeled by harmonic potential.

$$H\vec{x} = \omega^2 \vec{x} \tag{3}$$

The eigenvectors of this system provide the normal modes, while the eigenvalues are the squared frequency of oscillations. These oscillatory trajectories along the eigenvectors can be found by plugging in eigenvalues and eigenvectors into the general solution (Blanchard et al.).

$$c(t) = \overrightarrow{x_1} e^{\lambda_1 t} \tag{4a}$$

These solutions were animated to visualize the 3-dimensional movement over time of the proteins.

1.3 Model

H-CNN is a model trained on the task of identifying which residue would fit into a space within a protein. This task allows H-CNN to predict the impact of mutations on protein stability and binding. As input, H-CNN takes a 10$\mathring{A}$ neighborhood around a central amino acid and removes the central residue. It outputs a length 20 vector describing the likelihood that any given amino acid could fit into the empty space(Pun et al.).
In feed forward neural networks, every neuron in one layer is connected to every neuron in the next layer. The strength of each connection is described by a weight, and the level of activation of every neuron in a layer decides the activation of every neuron in the next layer based on these weighted connections. A neural network can be written as a nested function dependent on these weights, where x is some input, W_i is a set of weights, and Φ is a non-linear activation function.

$$f(x) = \Phi(W_k ... \Phi(W_2 \Phi(W_1 x)...)) \tag{5}$$

Neural networks are trained via manipulations of these weights. Given a certain input, the network's output is compared to an expected output, and a function of the difference between the two is used to change the weights so the output better matches the expectation. This process involved propagation, where the gradient of the loss is propagated through the neural network in order to understand how best to change each weight. It is important to note that the neural network must be differentiable in order to do this. Convolutional neural networks (CNNs) are a variety of neural networks developed for classification and image recognition tasks. CNNs are marked by a convolutional layer, where a kernel looking for some feature iterates over the input data and outputs the similarity between the inputs and the feature. One layer may contain multiple kernels in order to effectively break down the inputs into a set of feature maps. CNNs often include multiple convolutional layers that search for features hierarchically, where kernels in later layers track over the outputs of the original kernels to find more complex features.

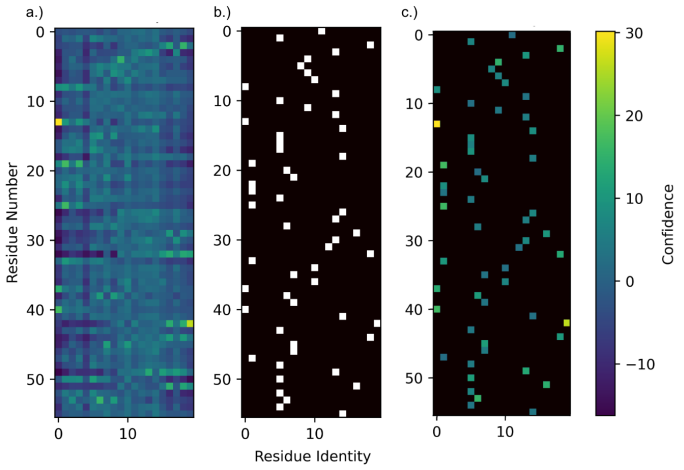

Figure 3: Calculation of pseudo-energy from model outputs. a.) For each residue in a protein, the confidences that that residue is each of the twenty possible amino acids. b.) The correct residue identity for every residue in the protein. c.) A and B layered on top of one another. The sum of these elements is labeled a pseudo-energy.

Traditional CNNs are translationally equivariant, meaning an image that has been translated can be represented as the original image with some linear transformation. In the context of proteins, input residues and neighborhoods can appear in any orientation, so a CNN used to classify proteins must respect rotational symmetry. One solution to respecting rotational symmetry is data augmentation, where a neural network is trained on example inputs in many different rotations, but this becomes computationally costly in three dimensions. H-CNN solves this issue by projecting protein structures into spherical Fourier space as holograms. H-CNN uses 3D Zernike polynomials as spherical basis functions to encode the point clouds as hologram. The angular components of Zernike polynomials are spherical harmonics, which have the benefit of making the inputs rotationally equivarient (Pun et al.).

As inputs, H-CNN takes coordinates, atomic information, and physiochemical information for protein a neighborhood of 10 $\mathring{A}$ from the central amino acid's alpha carbon. Each information type is stored as a separate point cloud, and all are input to the model through different channels and encoding as holograms. The encoded information is then input into a set of convolutional layers, followed by a set of fully connected layers. These layers together return a 20 long vector describing the confidence that a given amino acid could fit into the space in the original neighborhood.

One way to encode this output information is to take the sum of confidences at the correct residues for the entire protein. This process is showing in Fig. 3, where to correct residue identities is mapped onto the full set of confidences, which can then be summed to output a total confidence of

the system.

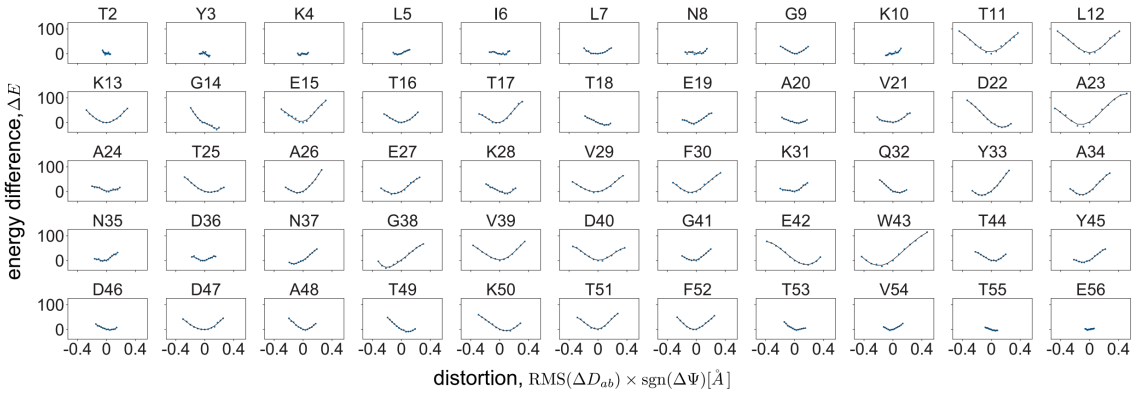

Figure 4: Pseudo-energy behavior modelled by (Pun et al.). For a given perturbation from equillibriums, the change in pseudo-energy from equilibrium is plotted

When run for a complete protein, the sum of confidences at the correct residues appears to behave like a potential energy. With a perturbation about equilibrium, this sum behaves as a parabola, as one would expect from a potential energy perturbed about equilibrium. As can be seeing in Fig. 4 from the original H-CNN publication, perturbations about equilibrium bring about clear harmonic energy-like behavior in the sum of confidences for many of a protein's residues, but not all. This behavior is not exact, and many residues produce curves that flatten far from equilibrium, but overall, there is a distinct similarity between the behavior of the sum of confidences given a perturbation about equilibrium and a potential energy. Due to this similarity, the sum is labeled a pseudo-energy (Pun et al.).

2 Methods

If H-CNN's pseudo-energy outputs are an effective representation of the protein's physical energy, we ought to be able apply NMA these outputs in order to model oscillatory modes of motion in proteins. To do this, we must find the Hessian of the model pseudo-energy with respect to the degrees of freedom of the inputs. We do this in two ways, using automatic differentiation and finite difference.

2.1 Automatic Differentiation

As described in equation 4, a neural network can be thought of as a large function, and must be differentiable to be trained. Automatic differentiation takes advantage of the fact that any computer computation can be understood as a set of basic arithmetic functions. Using tools from the python package PyTorch, we create a modified version of H-CNN where the output pseudo-energy is differentiable with respect to input coordinates.
In order to use NMA on H-CNN, we create a Hessian matrix composed of the second derivatives of the pseudo-energy of the model with respect to the x,y,z degrees of freedom of each atom. Automatic differentiation is computationally costly when applied to a neural network, due to the size of the function that described the neural network. H-CNN takes Hydrogen atoms as part of its input, which many

protein models do not include, including the ENM we compare our results to. The inclusion of these atoms increases the number of degrees of freedom and thus the computational cost of the full Hessian.

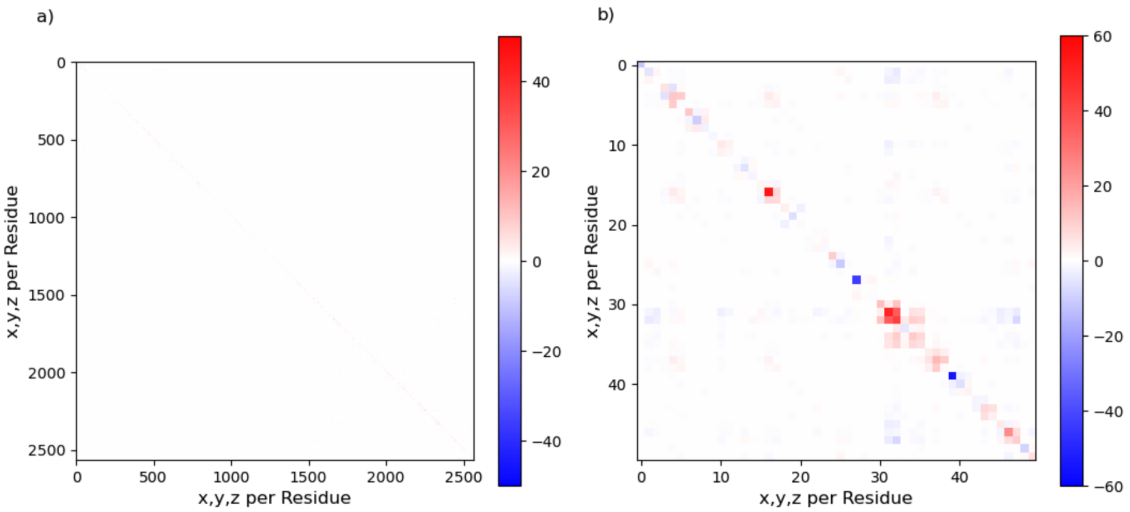

Figure 5: Hessian matrix produced from automatic differentiation. a.) shows the full matrix, with respect to the x,y, and z degrees of freedom of every atom in the protein. At this scale, it is difficult to see the structure of the Hessian, so b.) zooms in on the first 50 degrees of freedom. The matrix has a clear diagonal.

In order to better compare our Hessian to those produced using the ENM and using the finite differences method, we collapse the automatic differentiation Hessian down to only include the components related to the alpha carbons of each amino acid. We confirm the effectiveness of this method by comparing two ENMs, one with respect to atoms and one with respect to residues. We create an alpha carbon Hessian from the atom level ENM in two ways, first by selecting the alpha carbon Hessian components, and second by summing the x,y,z components of each atom within each residue. We compare these results to the residue level ENM and find that the first method performs significantly better. We apply the first method, of selecting alpha carbon Hessian components, to our automatic differentiation hessian in order to create a per residue hessian we can compare to other methods.

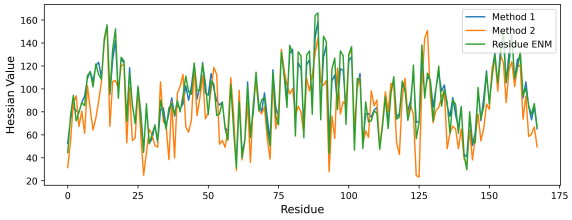

Figure 6: Diagonal values from ENMs. Method 1 collapses the per atom ENM by selecting for alpha carbon elements while Method 2 collapses the per atom ENM by summing over each residue. It is apparent that Method 1 reproduces the per residue ENM more effectively than Method 2.

2.2 Finite Difference

The finite difference method uses a multivariate numerical approximation to the second derivative to approximate the terms in the Hessian. The finite difference formulas are derived from Taylor expansions around a point.

$$f(x+\delta) = f(x) + \delta f'(x) + \frac{\delta^2}{2!}f^{(2)}(x) + \frac{\delta^3}{3!}f^{(3)}(x) + \frac{\delta^4}{4!}f^{(4)}(x) + \mathcal{O}(\delta^5) \tag{6a}$$

$$f(x-\delta) = f(x) - \delta f'(x) + \frac{\delta^2}{2!}f^{(2)}(x) - \frac{\delta^3}{3!}f^{(3)}(x) + \frac{\delta^4}{4!}f^{(4)}(x) + \mathcal{O}(\delta^5) \tag{6b}$$

Adding these equations together an approximation for the second derivative:

$$f''(x) = \frac{f(x+\delta) - 2f(x) + f(x-\delta)}{\delta^2} + \mathcal{O}(\delta^2) \tag{7}$$

The truncation error is on the order of δ^2. The diagonal terms can be expressed as a partial second derivative with respect to the residue. The function $f(x)$ describes the energy function, i.e. H-CNN.

$$H_{ii}(x,y) = \frac{f(x+\delta,y) - 2f(x,y) + f(x-\delta,y)}{\delta^2} \tag{8a}$$

To get the equation for off-diagonal terms in the Hessian:

$$f(x+\delta) = f(x) + \delta f'(x) + \frac{\delta^2}{2!}f^{(2)}(x) + \frac{\delta^3}{3!}f^{(3)}(x) + \frac{\delta^4}{4!}f^{(4)}(x) + \mathcal{O}(\delta^5) \tag{9a}$$

$$f(x-\delta) = f(x) - \delta f'(x) + \frac{\delta^2}{2!}f^{(2)}(x) - \frac{\delta^3}{3!}f^{(3)}(x) + \frac{\delta^4}{4!}f^{(4)}(x) + \mathcal{O}(\delta^5) \tag{9b}$$

Subtracting one from the other gives the central finite difference approximation for the first derivative:

$$f(x+\delta) - f(x-\delta) = 2\delta f'(x) + \mathcal{O}(\delta^3) \tag{10a}$$

$$f'(x) = \frac{f(x+\delta) - f(x-\delta)}{2\delta} + \mathcal{O}(\delta^2) \tag{10b}$$

Using this formula, we can take a partial derivative with respect to each variable, x and y:

$$f_{xy}(x,y) = \frac{f(x+\delta_1,y+\delta_2) - f(x+\delta_1,y-\delta_2) - f(x-\delta_1,y+\delta_2) + f(x-\delta,y-\delta_2)}{4\delta_1\delta_2} + \mathcal{O}\frac{\delta_2^2}{\delta_1} + \mathcal{O}(\delta_1^2) \tag{11a}$$

Since $\delta_1 = \delta_2$, the error will be first order.

$$f_{xy}(x,y) = \frac{f(x+\delta_1,y+\delta_2) - f(x+\delta_1,y-\delta_2) - f(x-\delta_1,y+\delta_2) + f(x-\delta_1,y-\delta_2)}{4\delta_1\delta_2} + \mathcal{O}(\delta_1) \tag{11b}$$

The function f that we want the partial derivative of is the energy function, specifically the entire neural network. The perturbations that the derivative of the energy is taken with respect to are shear perturbations of a single residue. As such, each energy function is the evaluation of the pseudo

energy across the entire protein with the designated residues perturbed.

$$H_{ii} = \frac{f(r_i + \delta_i) - 2f(r) + f(r_i - \delta_i)}{{\delta_i}^2} \tag{12a}$$

$$H_{ij} = \frac{f(r_i + \delta_i, r_j + \delta_j) - f(r_i + \delta_i, r_j - \delta_j) - f(r_i - \delta_i, r_j + \delta_j) + f(r_i - \delta_i, r_j - \delta_j)}{4\delta_i\delta_j} \tag{12b}$$

where:

$$f(r_i + \delta_i) = E(\vec{r}_{\text{protein}} | r_i \to r_i + \delta_i) \tag{13a}$$

$$f(r_i + \delta_i, r_j + \delta_j) = E(\vec{r}_{\text{protein}} | r_i \to r_i + \delta_i, r_j \to r_j + \delta_j) \tag{13b}$$

Equation 12b requires simultaneous perturbation of two residues. The shear perturbation is a natural molecular movement that twists the protein along the backbone. A perturbation of $r_i + \delta_i$ means that the ψ angle on residue i is twisted δ degrees while the ϕ angle on residue $i-1$ is twisted $-\delta$ degrees. This method allows us to limit the degrees of freedom along which the Hessian is evaluated to directions that are physically probable. However, there is an intrinsic error associated with this numerical method. Smaller perturbations reduce this error, but there is a lower bound where the difference in the location of neighborhood atoms is truncated by the PDB writing software. The perturbations ranged from $\delta = [0.1, 10]$, with a δ of 0.1 providing the best results.

3 Results and Discussion

3.1 Hessian Matrices

We produced Hessian matrices of H-CNN's output pseudo-energy using and ENM, finite differences, and automatic differentiation, as seen in Fig. 7. We use an Anisotropic Network Model from the ProDy package as our ENM. Both the ENM and automatic differentiation produce Hessians with strong positive diagonals, while the finite differences Hessian has some diagonal structure, and a number of strong off-diagonal elements.

3.2 Normal Modes Animated

The trajectory of protein G along the lowest normal mode was animated for all three methods, and the animations for the lowest normal mode can be found at the following links: finite difference, automatic differentiation, and ENM. As can be seen in the animations, H-CNN predicts motions that are different from the ENM. The normal modes predicted through the finite difference method and the automatic differentiation method are also different. The noise in the finite difference mode is likely caused by the noise in the Hessian, with little attached physical meaning.
Since ENMs have been shown to accurately predict the motions of proteins using the lowest normal modes, it is unlikely that H-CNN through automatic differentiation is showing physical motions. However, this analysis is more effective using a protein with a known low-frequency motion, such as the hinging motion of the egg white lysosomes discussed in (Dykeman and Sankey). Extending this analysis to a larger protein like this one will confirm the efficacy of using H-CNN to predict the energy of a protein and learn the underlying physics of proteins.

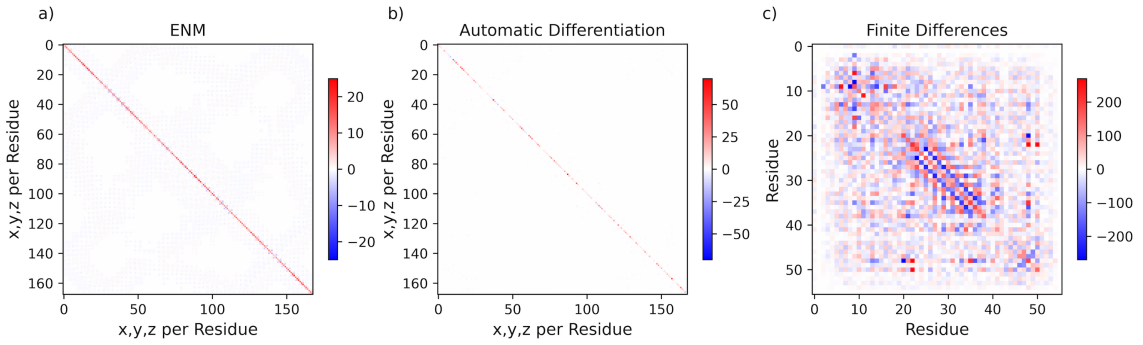

Figure 7: Hessian matrices from different methods. a.) is the Hessian with respect to the x,y and z degrees of freedom of each residue produced using ProDy's ANM, and it features a strong diagonal. b.) is the Hessian produced using automatic differentiation and collapsed to only the x,y, and z degrees of freedom of the alpha carbons in order to be more effectively visualized. It too features a strong diagonal. c.) is the Hessian produced using finite differences with respect to full residues. It features a stronger off-diagonal structure than the other Hessians.

3.3 Automatic Differentiation

In order to better understand the background structure of the automatic differentiation Hessian, we collapse our Hessian once again by taking the norm of the degrees of freedom to produce a residue by residue Hessian. By choosing our colormap such that the background colors are bright, the structure of the off-diagonal Hessian elements becomes apparent.

Using our ENM, we produce a connectivity matrix, which highlights all residues within $20\mathring{A}$. Each neighborhood input to H-CNN has a radius of $10\mathring{A}$, making this cutoff an effective comparison. The background structure of our automatic differentiation Hessian clearly reflects the structure of the connectivity matrix. This means that H-CNN predicts that residues close to one another have a larger impact on one another than residues far apart. Although this does not say much about H-CNN's ability to understand the underlying physics of these systems, it does point to H-CNN's effectiveness in understanding protein structure.

The Hessian produced through automatic differentiation is directly comparable to the Hessian produced using an ENM because both are with respect to the x,y, and z degrees of freedom of each residue in the protein. Although it is apparent that both matrices have a strong positive diagonal, it is not clear if there are other underlying similarities. In order to further compare these matrices, we produce the covariance matrix of each. The covariance matrix is defined as the inverse of the Hessian, here the pseudo-inverse because the automatic differentiation Hessian is not traditionally invertible.

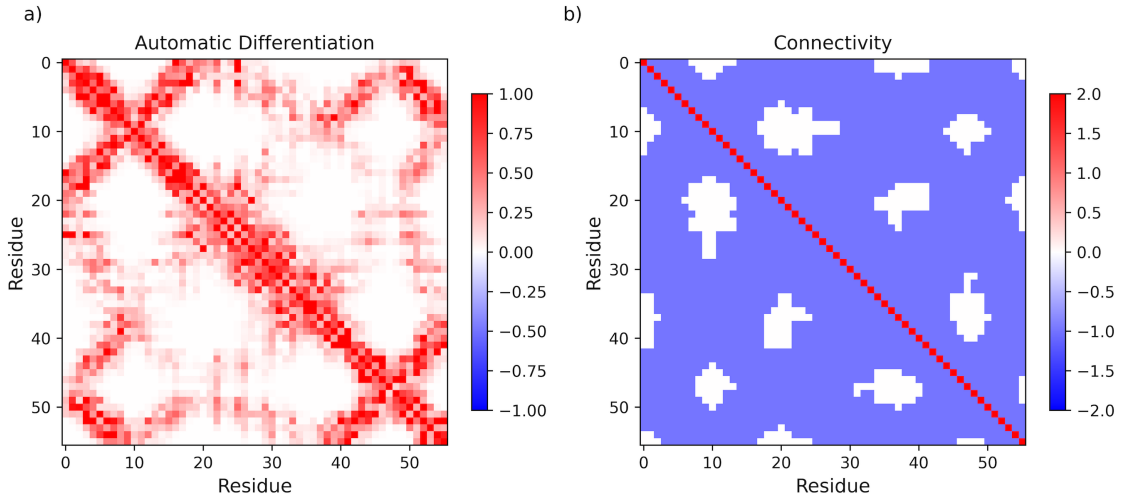

Figure 8: Comparison of Hessian background structure and connectivity. a.) The automatic differentiation hessian collapsed to be per residue, with the colors selected to highlight background structures. b.) The connectivity matrix of the ENM, showing residues within 20$\mathring{A}$ of one another.

The covariance matrices describe the movement of one degree of freedom relative to another. A positive covariance implies there is a direct correlation between motion along one degree of freedom and another, while a negative covariance implies they move opposite one another. Zero covariance implies that the degrees of freedom have little impact on one another. The diagonal elements of the covariance matrix represent individual motions along degrees of freedom. By taking the norm over the x,y, and z degrees of freedom for each residue in these matrices, we produce a number describing the motion of each residue in the protein, that, when combined with a statistical mechanical pre-factor, is comparable to experimental B-factors. B-factors are a measure of the thermal energy of a residue in a protein. They can be experimentally determined through x-ray crystallography, and are a direct way to relate models and experiments. As seen in Fig. 10, the ENM does not exactly predict β-factor, but does a better job than automatic differentiation. The B-factors produced using automatic differentiation have little relationship to the experimental B-factors, implying that H-CNN is not learning the underlying physics of proteins.

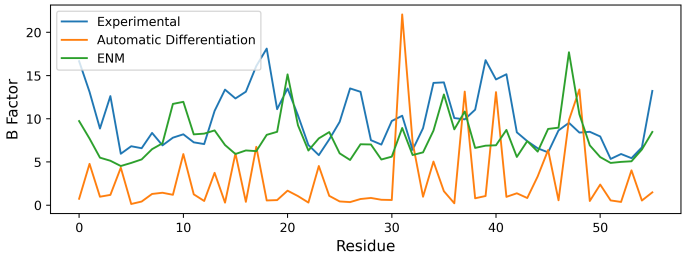

Figure 10: B-factors from the covariance matrices and from the experiment. These numbers have been scaled arbitrarily in order to be effectively compared, as the units of H-CNN pseudo-energy are unclear.

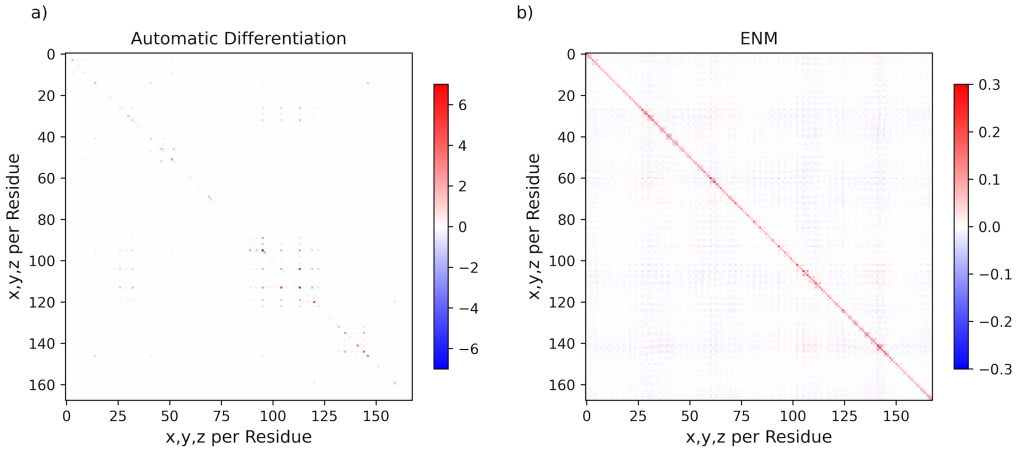

Figure 9: Covariance matrices from automatic differentiation and ENM.

3.4 Finite Difference

In Fig. 11, the finite difference Hessian for each delta perturbation is shown. A $\delta = 0.1$ creates very chaotic, nearly random behavior. The $\delta = 5$ Hessian shows very strong off-diagonal terms and has an error on the order of 25 for diagonal terms. The $\delta = 1$ Hessian appears to be the clearest of the three, showing off-diagonal terms that are on the order of the rest of the matrix. The overpowering of off-diagonal terms compared to diagonal terms is interesting and unexpected. This would indicate that the effect of one residue on another is stronger than the local stability of the backbone. The most prominent off-diagonal terms were the strongly positive terms at (11,22) and (22,48), and the strongly negative terms at (11,9) and (22,19).

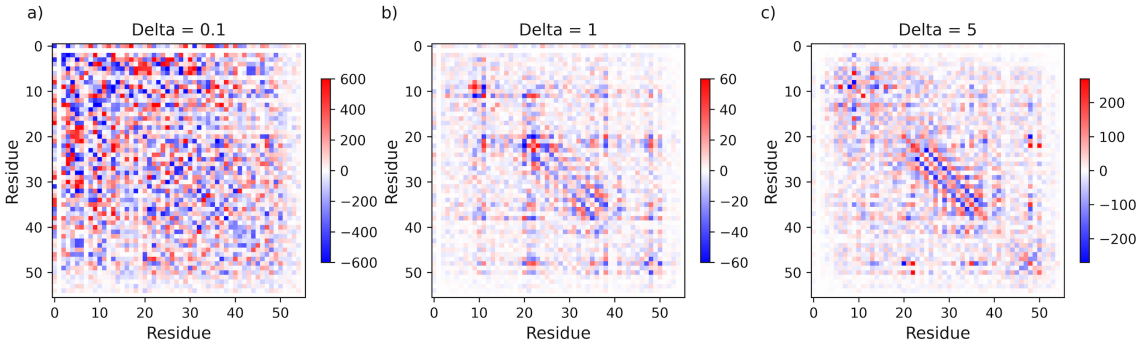

Figure 11: Hessian with Varying δ values

As seen in Fig. 12, these residues of interest appear in different parts of the protein, which does not strongly support the theory that H-CNN is learning specific information about the protein structure as demonstrated through these points of significant stability/instability. Perturbation of these specific external residues may cause steric strain on the inside of the protein, but we would expect stronger relationships between residues that are closer together than residues that are seemingly on opposite sides of the protein. It seems that these strong off-diagonal terms do not provide further physical insight, but instead might result from both error in the model and the intrinsic finite difference error. These Hessian terms are in terms of the shear perturbations, meaning the eigenvectors define an incomplete basis for protein motion. If this Hessian were converted into three-dimensional coordinates for each residue, the motion described by the eigenvectors would be a complete basis. Unfortunately, the process of changing coordinates is not intuitive.

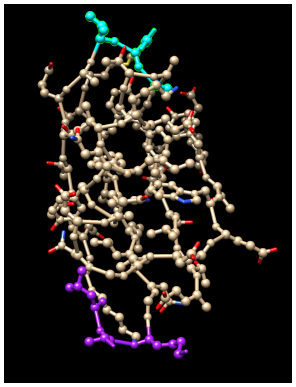

Figure 12: Locations of Residues of Interest

4 Conclusions and Future Work

This project does not support the theory that H-CNN is learning the normal motions of proteins. H-CNN appears to partially recreate the energy predictions of an ENM, but does not provide additional insight into the underlying physics of proteins. It is promising that H-CNN recreates the pattern of the connectivity matrix of an ENM, as seen in Fig. 8, but it does not effectively recreate the covariance matrix. Additionally, B-factors calculated using H-CNN do not recreate experimental B-factors, implying that H-CNN is not learning the underlying physics of the proteins.

In order to complete this analysis, we created workflows for Hessian calculation using both automatic differentiation and finite differences. Future work would apply these workflows to a larger protein with well-known low-frequency motions and could include an analysis of normal modes beyond the first mode. Applying this workflow to a larger protein or to various proteins could also allow us to do accuracy testing on B-factors, as H-CNN's inability to recreate B-factors for protein G could be an artifact of the protein's individual size or structure.

Converting the finite difference Hessian to Cartesian coordinates will allow for direct comparisons between the different methods and different models. The finite difference method could be redone using varying amounts of perturbation to control the RMS of the distances as is shown in (Pun et al.). This may create a more clear energy landscape.

This work shows that the pseudo-energies learned by H-CNN likely do not map to the potential energy of proteins. These conclusions provide insight into H-CNN's applicability and give us a better understanding of the model's outputs as a whole.

References

[Alberts et al.] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. Protein function. In *Molecular Biology of the Cell. 4th edition*. Garland Science.

[Bahar et al.] Bahar, I., Lezon, T. R., Yang, L.-W., and Eyal, E. Global dynamics of proteins: Bridging between structure and function. 39:23–42.

[Bauer et al.] Bauer, J. A., Pavlović, J., and Bauerová-Hlinková, V. Normal mode analysis as a routine part of a structural investigation. 24(18):3293.

[Blanchard et al.] Blanchard, P., Devaney, R., and Hall, G. *Differential Equations*. Cengage Learning.

[Dykeman and Sankey] Dykeman, E. C. and Sankey, O. F. Normal mode analysis and applications in biological physics. 22(42):423202.

[Pun et al.] Pun, M. N., Ivanov, A., Bellamy, Q., Montague, Z., LaMont, C., Bradley, P., Otwinowski, J., and Nourmohammad, A. Learning the shape of protein microenvironments with a holographic convolutional neural network. 121(6):e2300838121. Publisher: Proceedings of the National Academy of Sciences.

[Zanin et al.] Zanin, M., Buldu, J. M., and Boccaletti, S. Network of springs: A practical approach. 20:937–942.