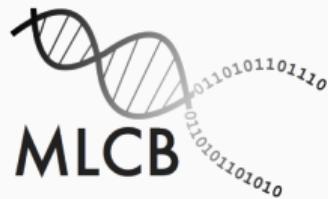


# Building a library to evaluate protein generative models

---

Philip Hartout

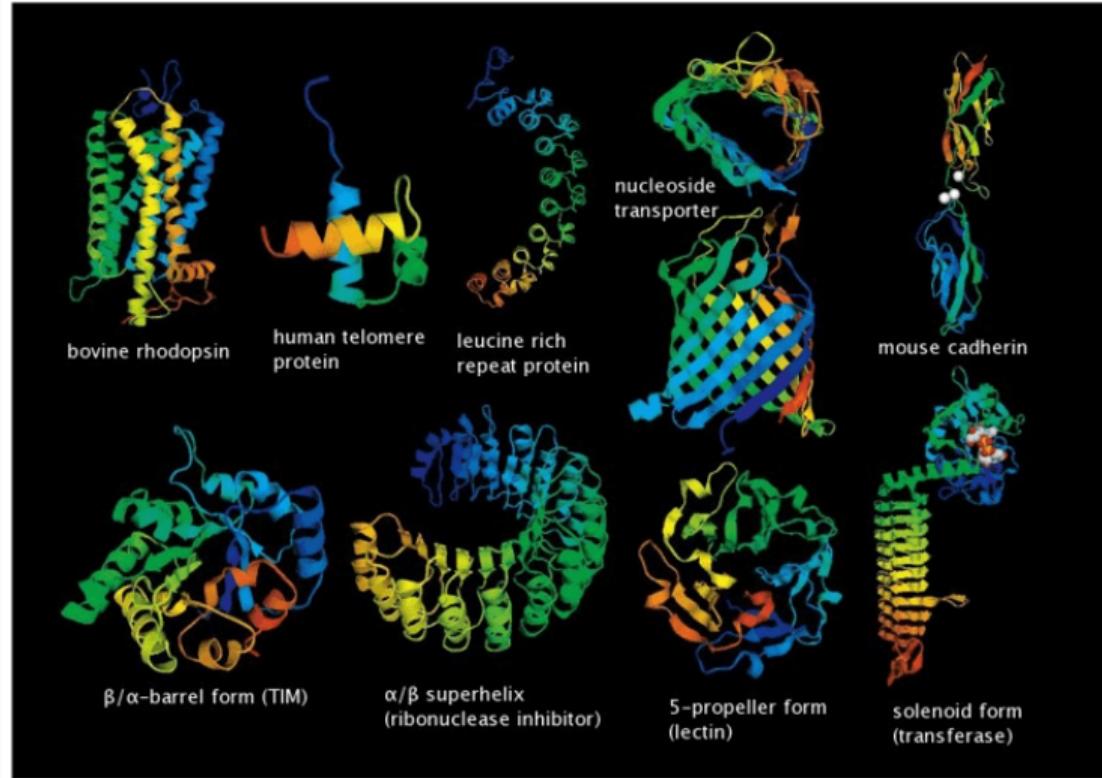
May 5, 2022



DBSSE

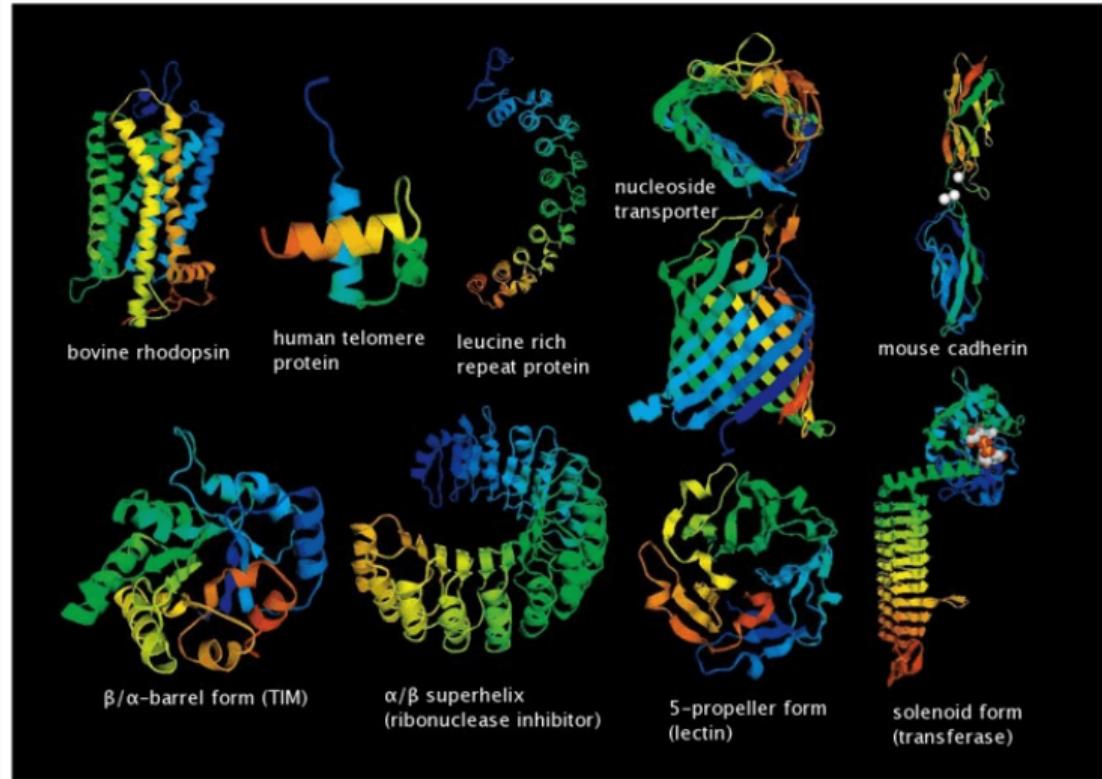
ETH zürich

# Introduction



Proteins are diverse.

# Introduction



Proteins are diverse.  
Support all functions for  
life.

# Generative Protein Modelling

Proteins

# Generative Protein Modelling

## Proteins

- can be engineered

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$



# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$



# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$

## Evaluation Problem



# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$



## Evaluation Problem

- Are the generated and empirical data distributions the same?

# Generative Protein Modelling

## Proteins

- can be engineered
- are well-defined

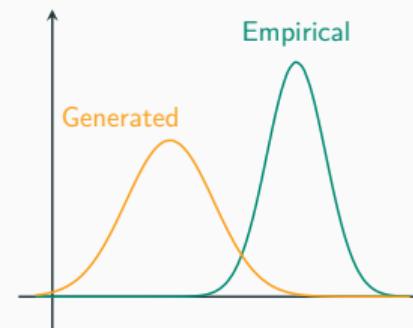
## Generative Model

- captures  $P(X)$
- generate samples following  $P(X)$



## Evaluation Problem

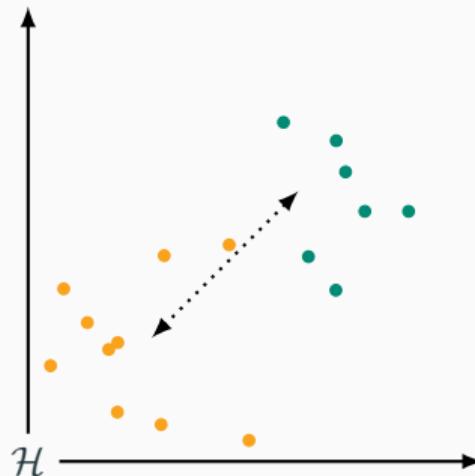
- Are the generated and empirical data distributions the same?



**What makes a good protein?**

# Maximum Mean Discrepancy (MMD)

$$\text{MMD}(X, Y) := \frac{1}{n^2} \sum_{i,j=1}^n k(x_i, x_j) + \frac{1}{m^2} \sum_{i,j=1}^m k(y_i, y_j) - \frac{2}{nm} \sum_{i=1}^n \sum_{j=1}^m k(x_i, y_j)$$



where:

- $\mathcal{X}$  is some non-empty set.
- $x_i, x_j \subseteq \mathcal{X}$ ,  $n$  is the number of samples in  $\mathbf{x}$ ;
- $y_i, y_j \subseteq \mathcal{X}$ ,  $m$  is the number of samples in  $\mathbf{y}$ ;
- $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  is a valid kernel.

MMD captures the distance between 2 sets on *any* RKHS  $\mathcal{H}$ .

## Maximum Mean Discrepancy (MMD) – continued

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

1. It's possible to leverage decades of kernel research!

Both a **blessing** and a **curse**:

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

1. It's possible to leverage decades of kernel research!

Both a **blessing** and a **curse**:

**Blessing** Flexibility, Computation on multiple representations

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

1. It's possible to leverage decades of kernel research!

Both a **blessing** and a **curse**:

**Blessing** Flexibility, Computation on multiple representations

**Curse** Instability (see Leslie's ICLR work), hyperparameter tuning.

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

1. It's possible to leverage decades of kernel research!

Both a **blessing** and a **curse**:

**Blessing** Flexibility, Computation on multiple representations

**Curse** Instability (see Leslie's ICLR work), hyperparameter tuning.

2. Possibility of statistical testing.

## Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate GNNs.

Advantages:

1. It's possible to leverage decades of kernel research!

Both a **blessing** and a **curse**:

**Blessing** Flexibility, Computation on multiple representations

**Curse** Instability (see Leslie's ICLR work), hyperparameter tuning.

2. Possibility of statistical testing.
3. Possibility of leveraging multiple representations.

– Thesis Goal –

Build a library to evaluate protein generative models

# Overview

Files

PDB Files

# Overview

Representations

$\varepsilon$  graphs

$k$ -NN graphs

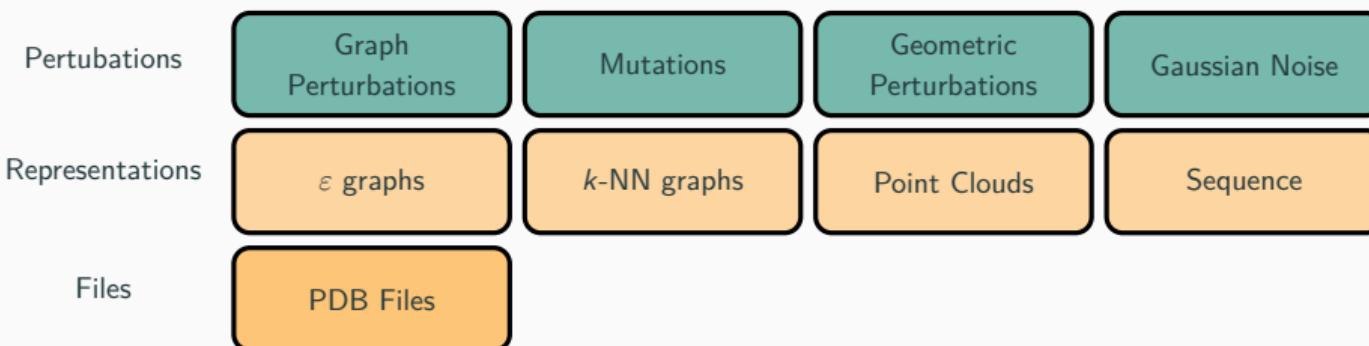
Point Clouds

Sequence

Files

PDB Files

# Overview



# Overview

Descriptors	Graph Descriptors	TDA Descriptors	Sequence Embeddings	Protein descriptors
Perturbations	Graph Perturbations	Mutations	Geometric Perturbations	Gaussian Noise
Representations	$\varepsilon$ graphs	$k$ -NN graphs	Point Clouds	Sequence
Files	PDB Files			

# Overview

Kernels	Graph Kernels	Vector Kernels	TDA Kernels	Kernel Composition
Descriptors	Graph Descriptors	TDA Descriptors	Sequence Embeddings	Protein descriptors
Perturbations	Graph Perturbations	Mutations	Geometric Perturbations	Gaussian Noise
Representations	$\varepsilon$ graphs	$k$ -NN graphs	Point Clouds	Sequence
Files	PDB Files			

# Overview

Measures	MMD			
Kernels	Graph Kernels	Vector Kernels	TDA Kernels	Kernel Composition
Descriptors	Graph Descriptors	TDA Descriptors	Sequence Embeddings	Protein descriptors
Perturbations	Graph Perturbations	Mutations	Geometric Perturbations	Gaussian Noise
Representations	$\varepsilon$ graphs	$k$ -NN graphs	Point Clouds	Sequence
Files	PDB Files			

## Experimental setup

We will study the following settings:

## Experimental setup

We will study the following settings:

1. Add gaussian noise to the point cloud and use graph kernels

## Experimental setup

We will study the following settings:

1. Add gaussian noise to the point cloud and use graph kernels
2. Add twist to proteins to the point cloud and see the results on different kernels

## Experimental setup

We will study the following settings:

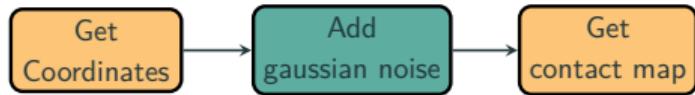
1. Add gaussian noise to the point cloud and use graph kernels
2. Add twist to proteins to the point cloud and see the results on different kernels
3. Add mutations to proteins to the point cloud and see the results on different kernels

## Experiment 1 – Gaussian Noise



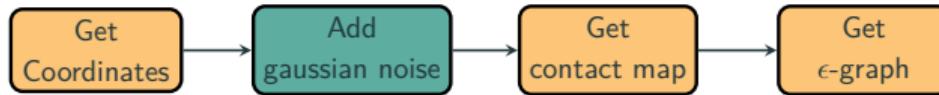
**Figure 1:** Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

## Experiment 1 – Gaussian Noise



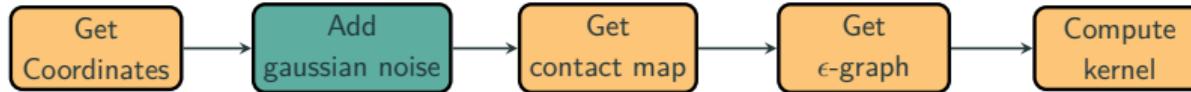
**Figure 1:** Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

## Experiment 1 – Gaussian Noise



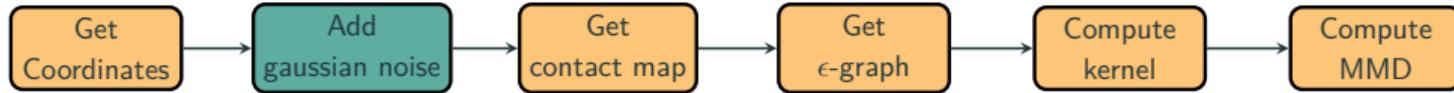
**Figure 1:** Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

## Experiment 1 – Gaussian Noise



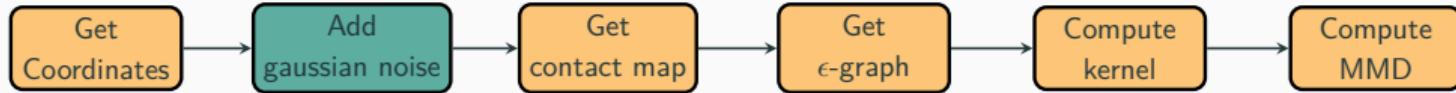
**Figure 1:** Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

## Experiment 1 – Gaussian Noise

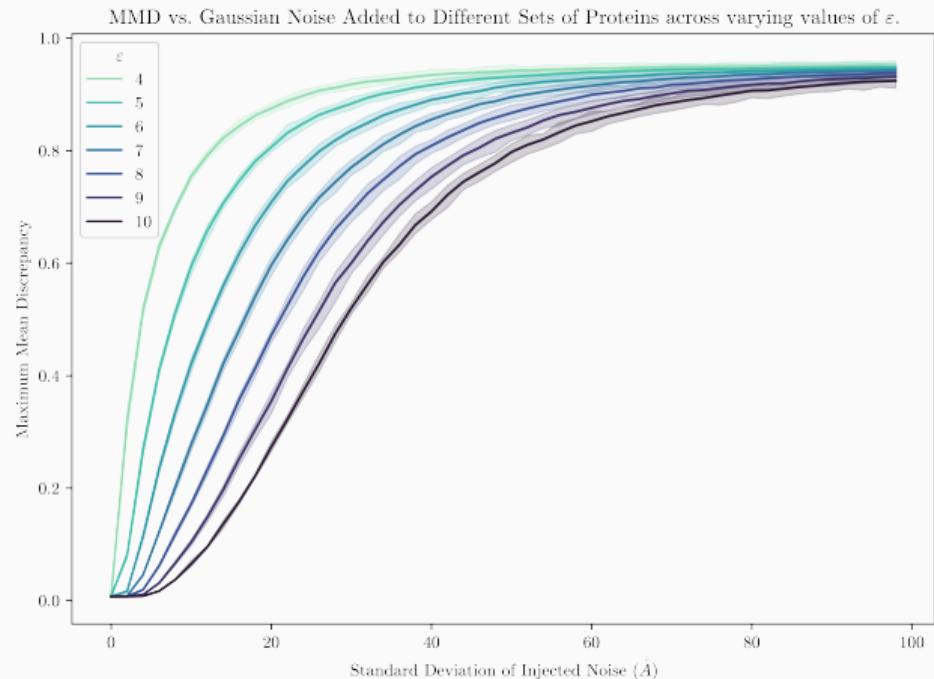
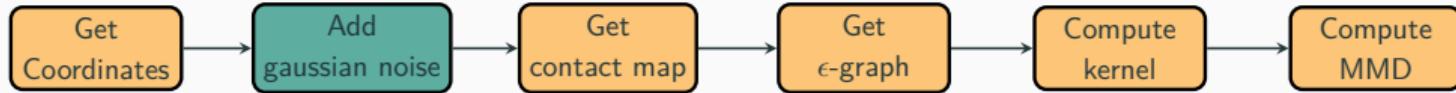


**Figure 1:** Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

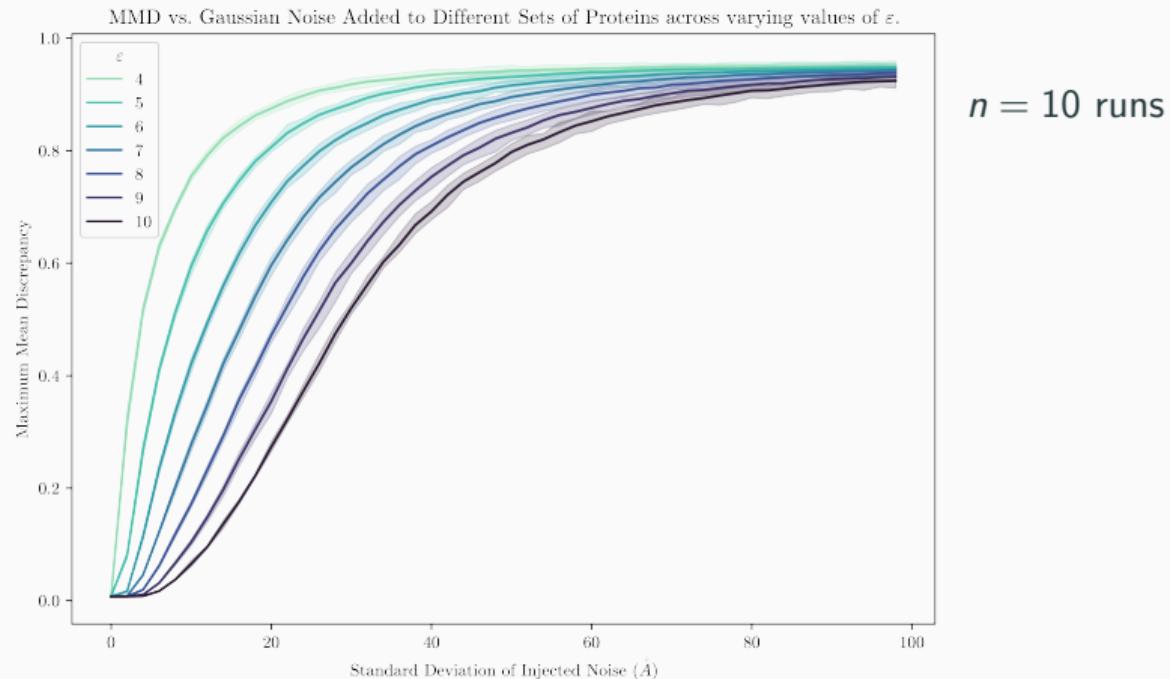
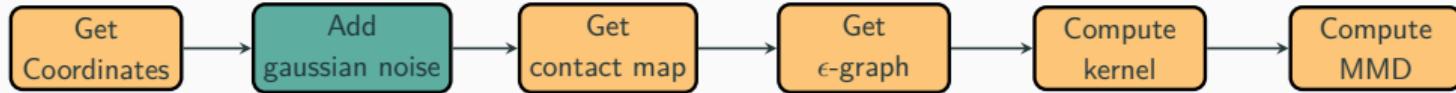
## Experiment 1 – Gaussian Noise



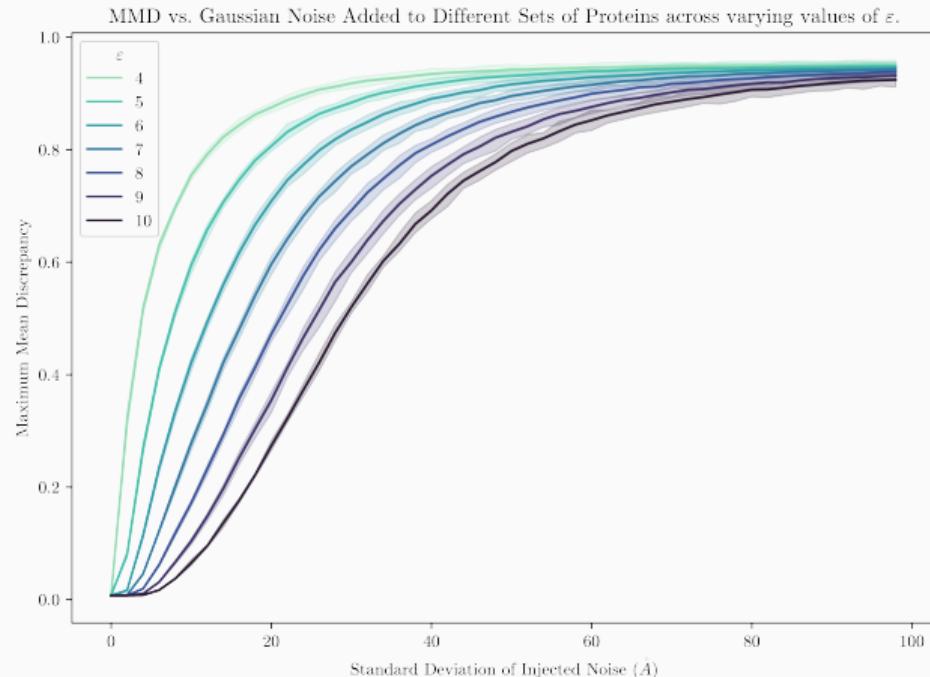
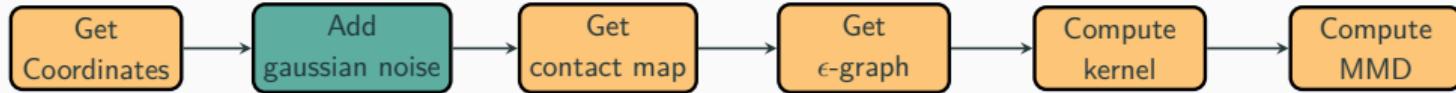
# Experiment 1 – Gaussian Noise



# Experiment 1 – Gaussian Noise



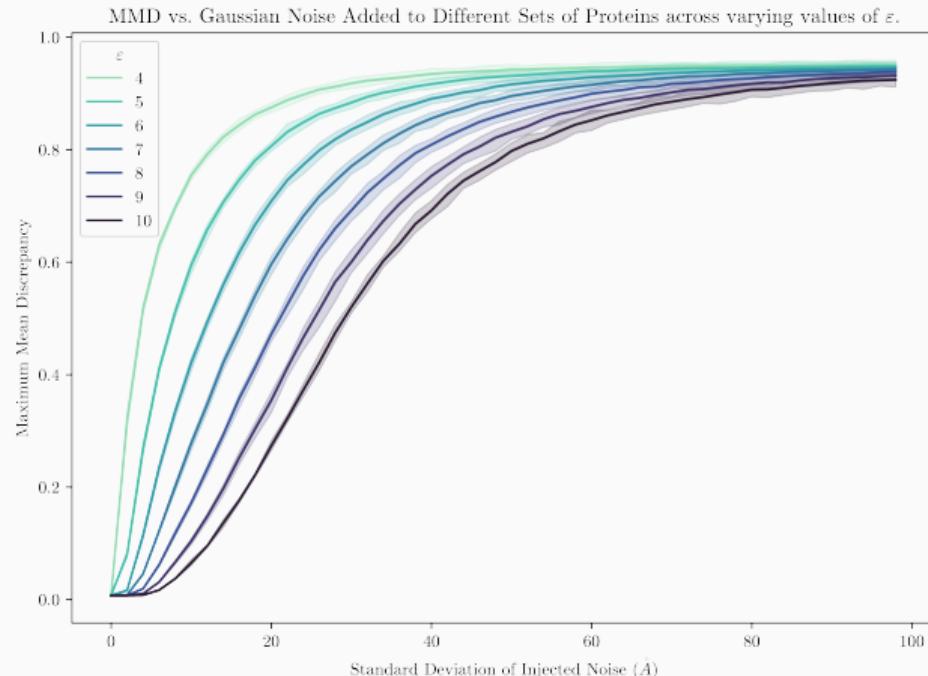
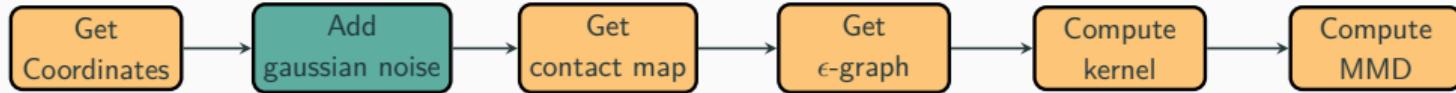
# Experiment 1 – Gaussian Noise



$n = 10$  runs

2 sources of variance:

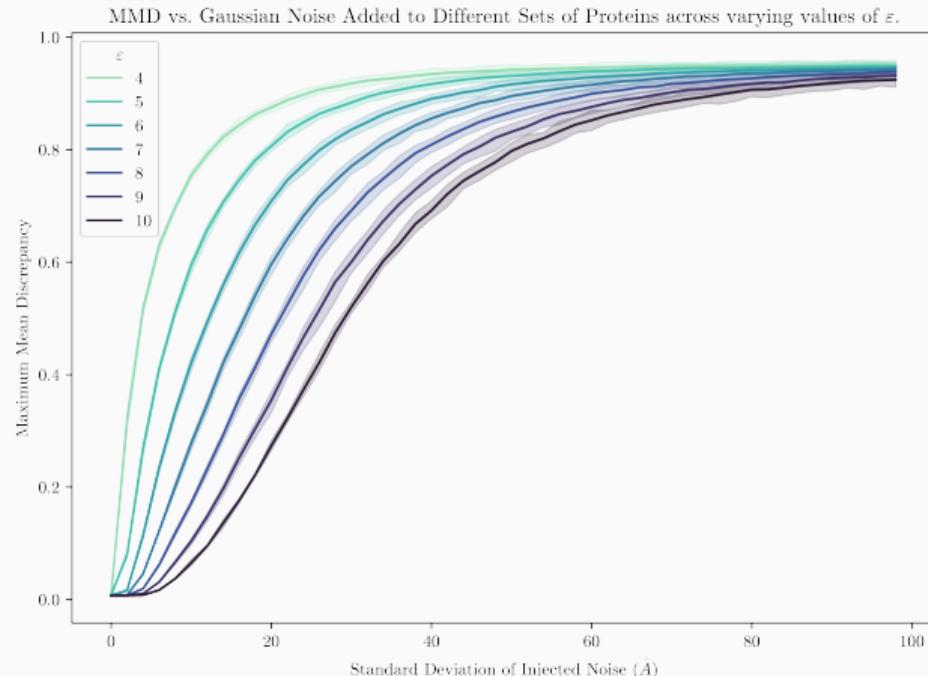
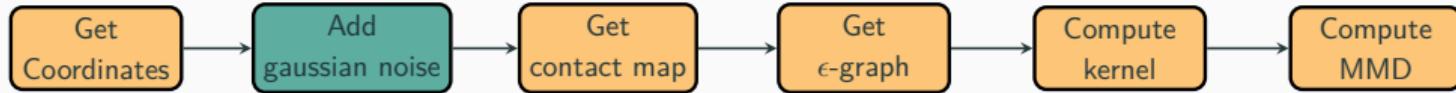
# Experiment 1 – Gaussian Noise



$n = 10$  runs  
2 sources of variance:

- Data

# Experiment 1 – Gaussian Noise



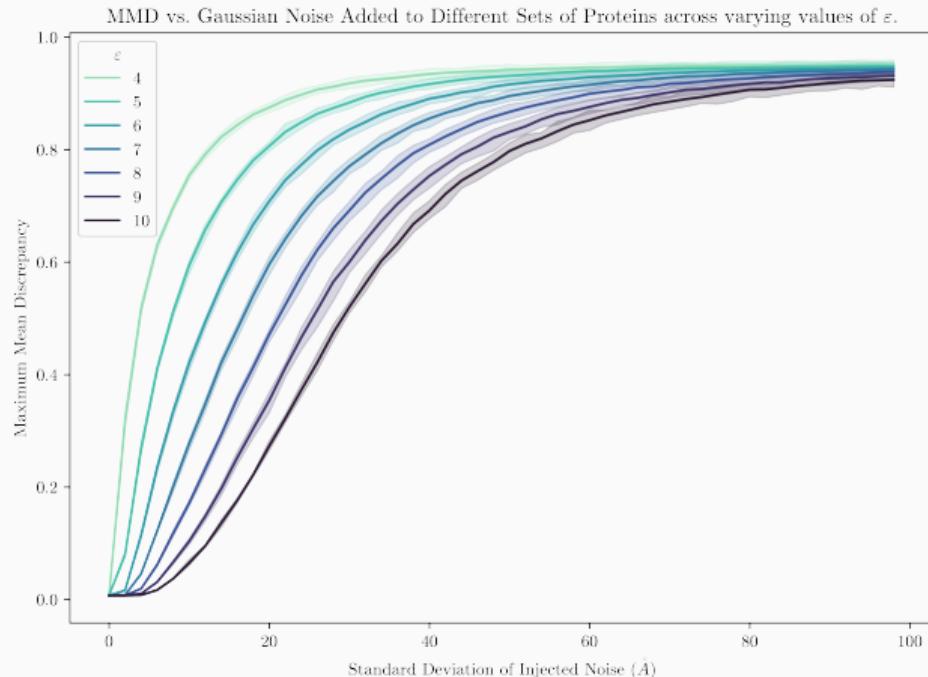
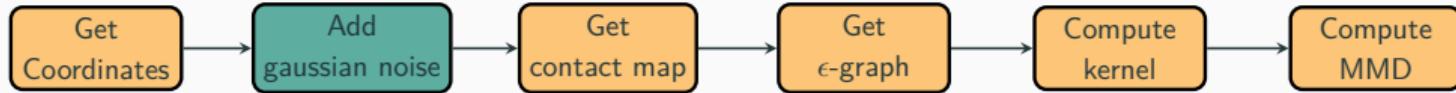
$n = 10$  runs

2 sources of variance:

- Data
- Noise

Conclusions

# Experiment 1 – Gaussian Noise



$n = 10$  runs

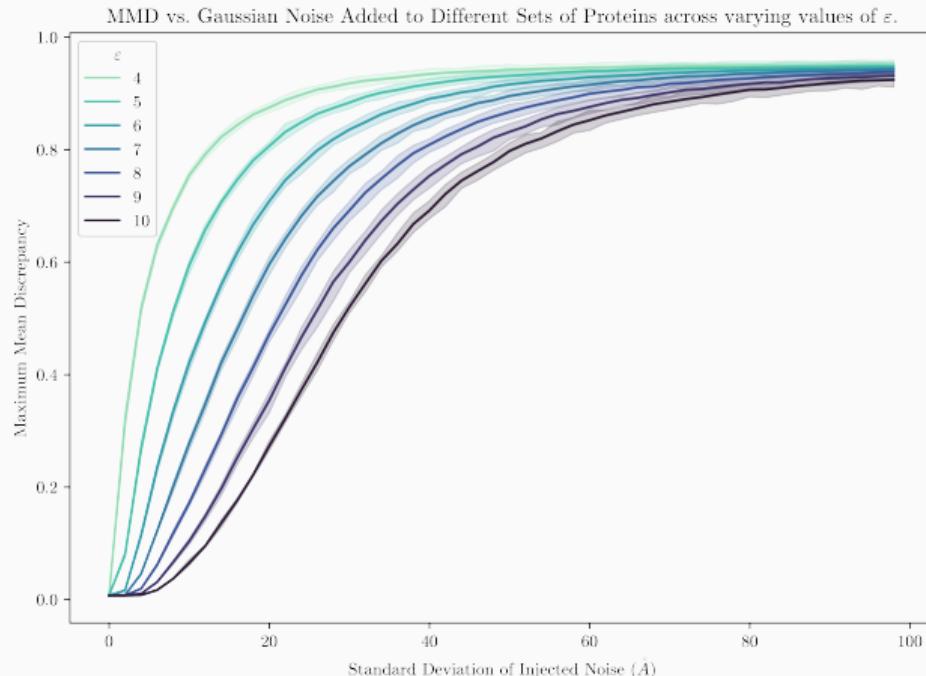
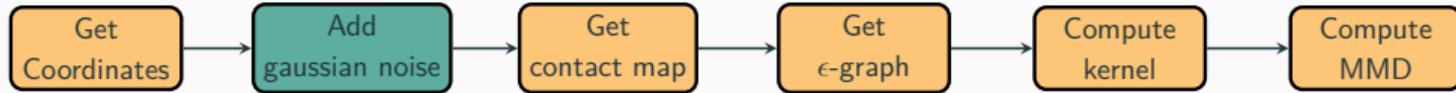
2 sources of variance:

- Data
- Noise

## Conclusions

1. MMD is stable using the Weisfeiler-Lehman kernel

# Experiment 1 – Gaussian Noise



$n = 10$  runs

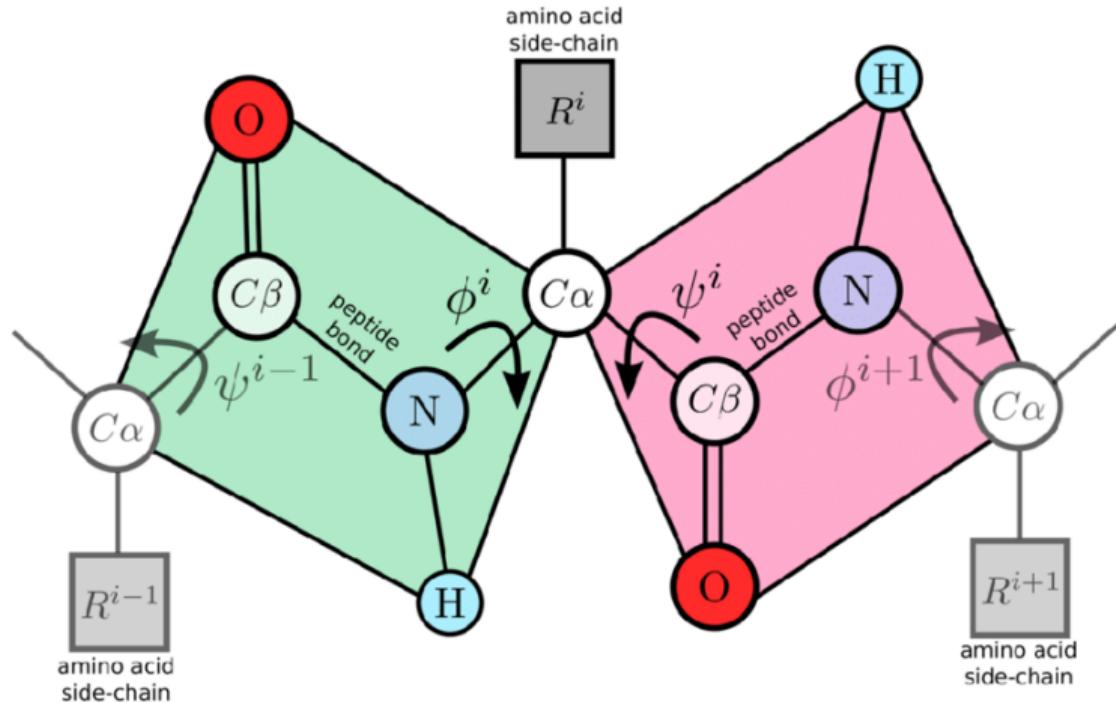
2 sources of variance:

- Data
- Noise

## Conclusions

1. MMD is stable using the Weisfeiler-Lehman kernel
2. Choice of representation influences MMD

# Introduction

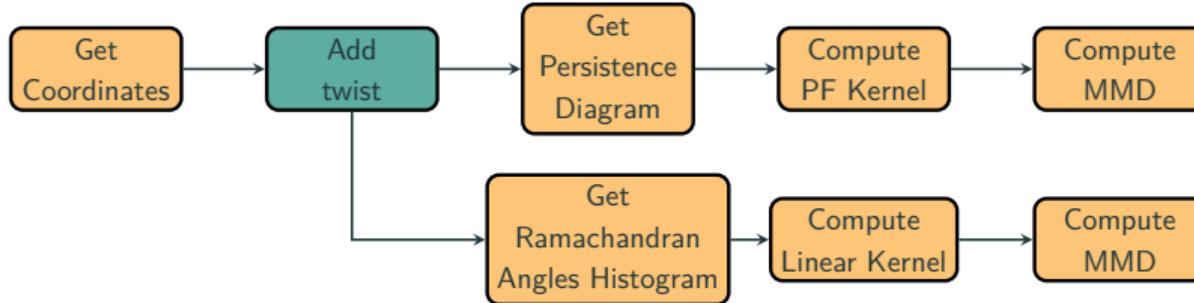


These angles *define* the shape of the protein

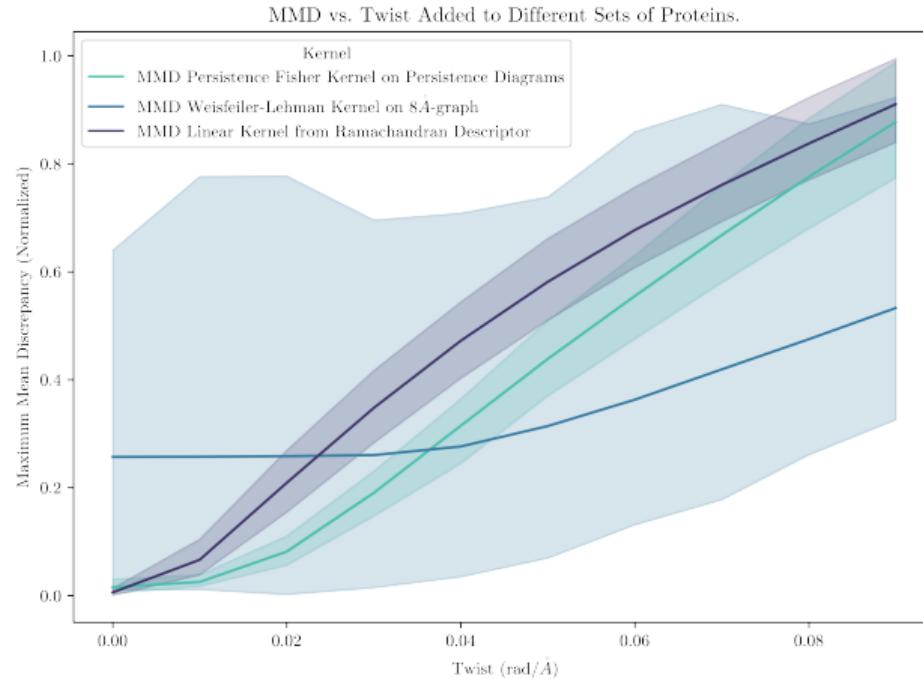
## Čech filtrations

Čech filtrations captures connected components, cycles and holes by varying  $\varepsilon$ .

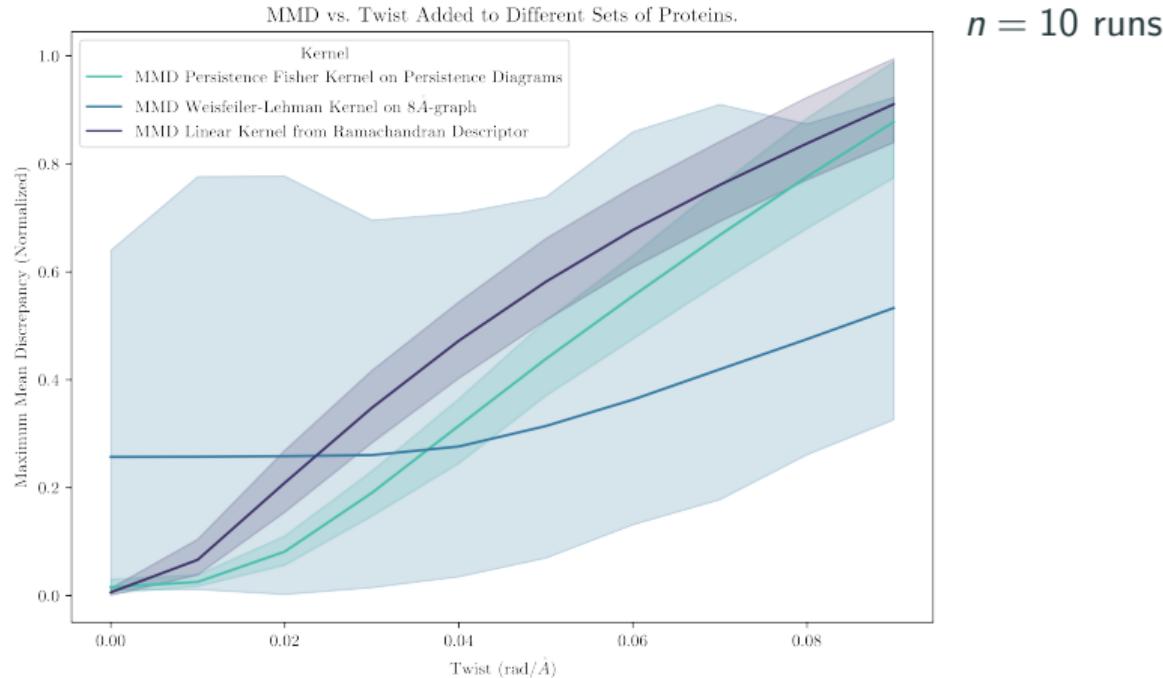
## Experiment 2 – Twist



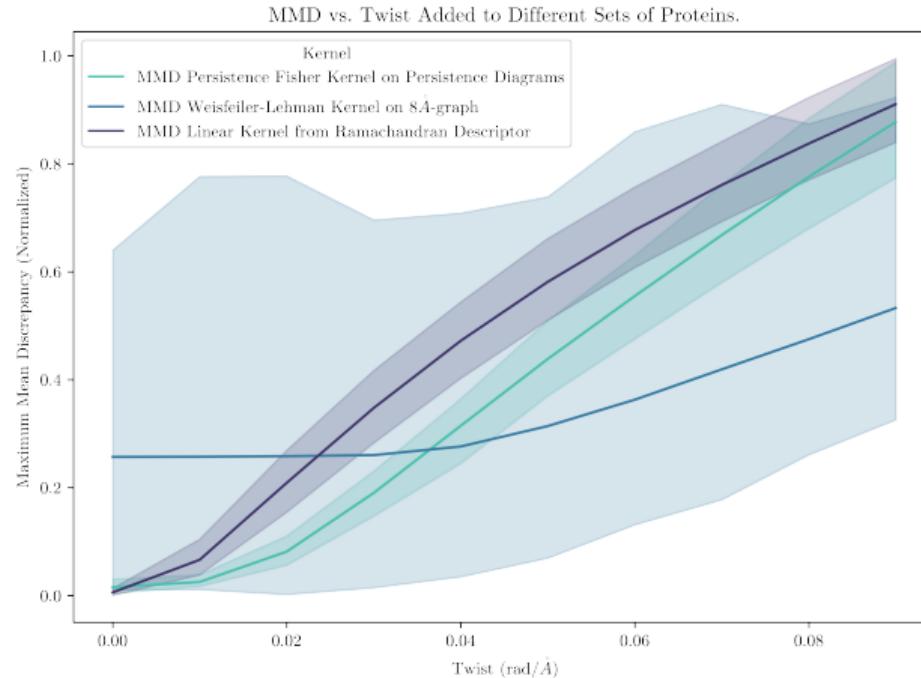
## Experiment 2 – Twist



## Experiment 2 – Twist



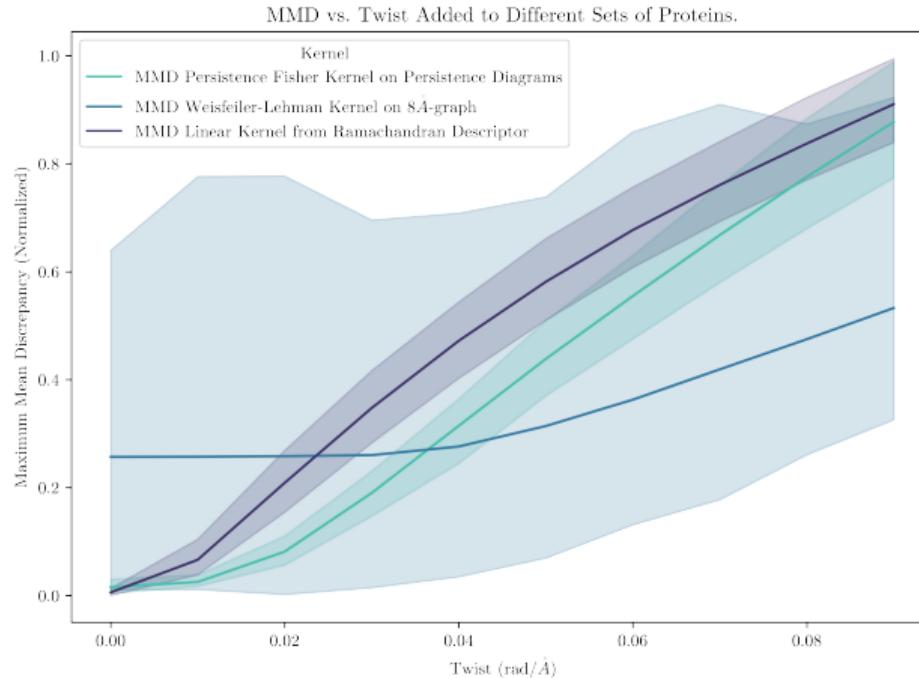
## Experiment 2 – Twist



$n = 10$  runs

1 source of variance:

## Experiment 2 – Twist



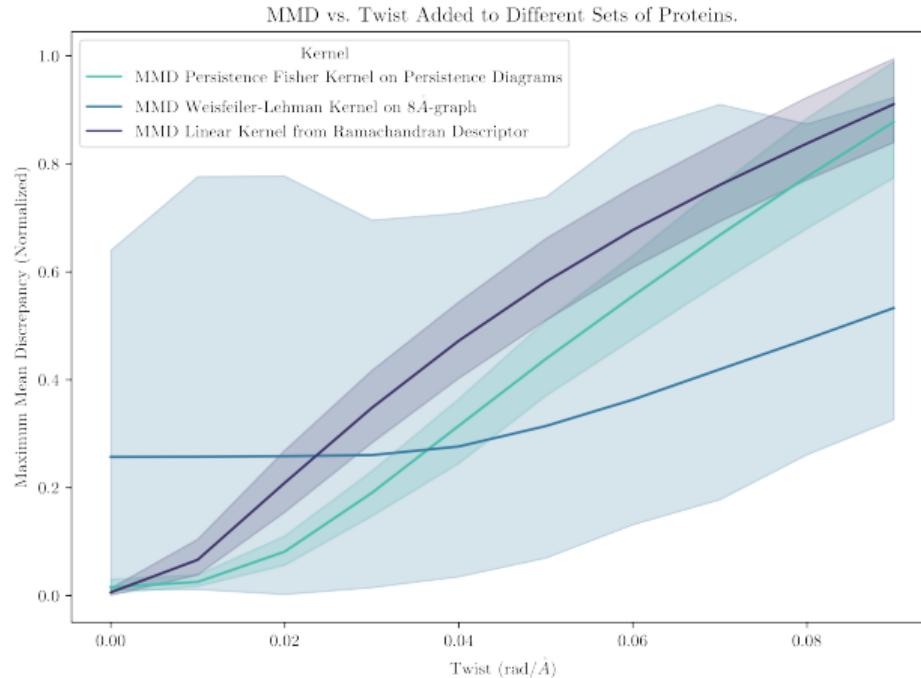
$n = 10$  runs

1 source of variance:

- Data

Conclusions

# Experiment 2 – Twist



$n = 10$  runs

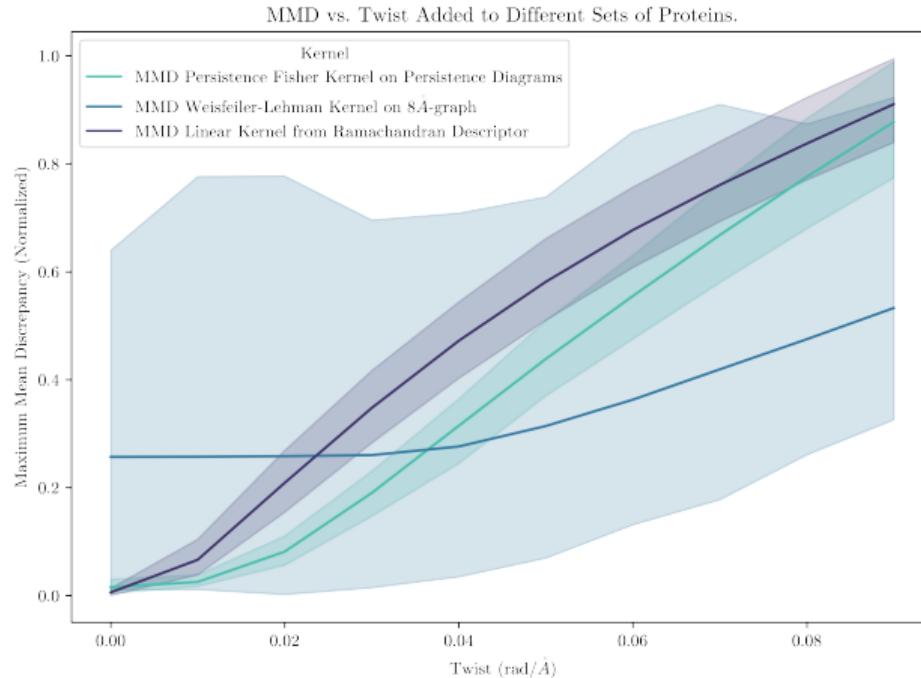
1 source of variance:

- Data

## Conclusions

1. Weisfeiler-Lehman kernel is not best to capture geometrical perturbations

## Experiment 2 – Twist



$n = 10$  runs

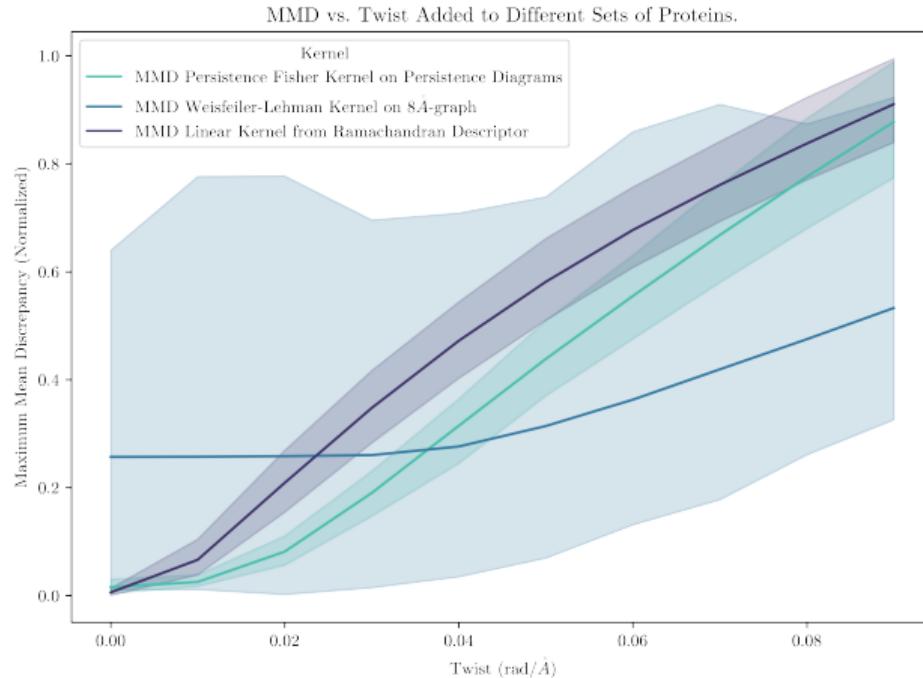
1 source of variance:

- Data

### Conclusions

1. Weisfeiler-Lehman kernel is not best to capture geometrical perturbations
2. TDA was difficult to compute but behaves very well

# Experiment 2 – Twist



$n = 10$  runs

1 source of variance:

- Data

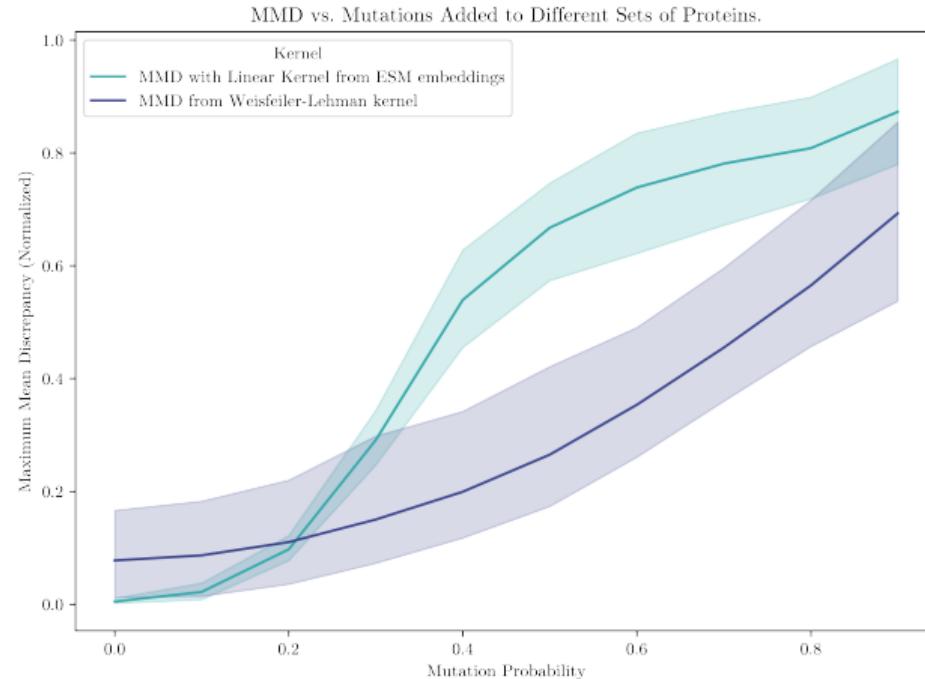
## Conclusions

1. Weisfeiler-Lehman kernel is not best to capture geometrical perturbations
2. TDA was difficult to compute but behaves very well
3. Ramachandran descriptors capture geometric perturbations very well

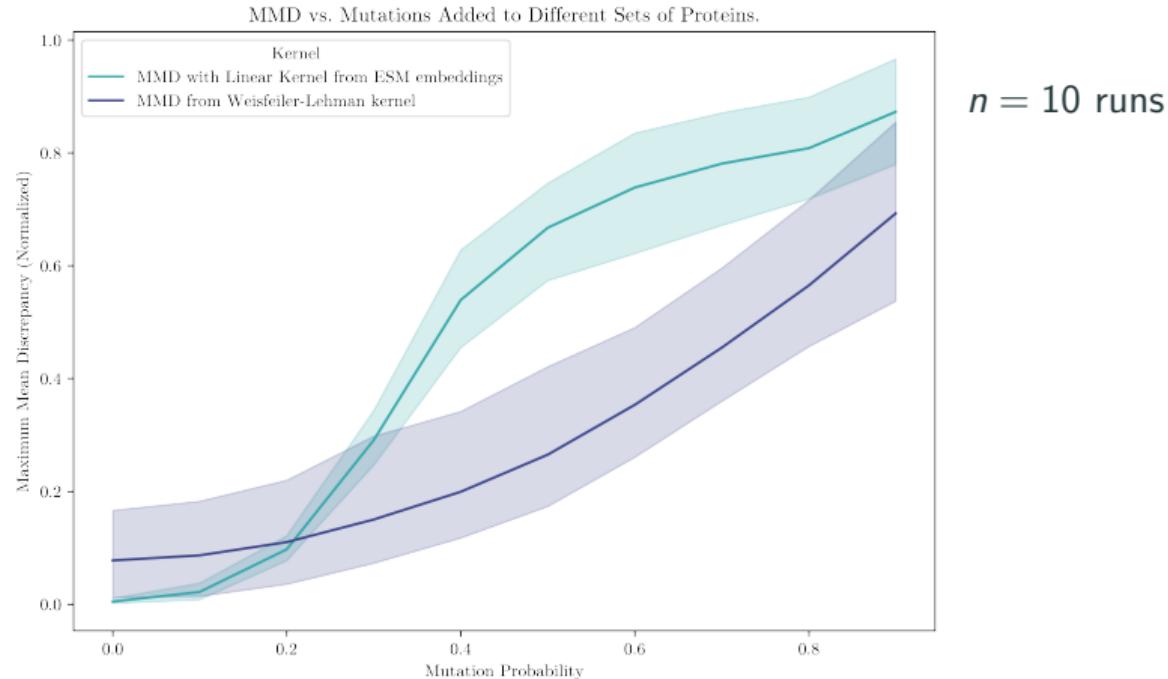
## Experiment 3 – Mutate



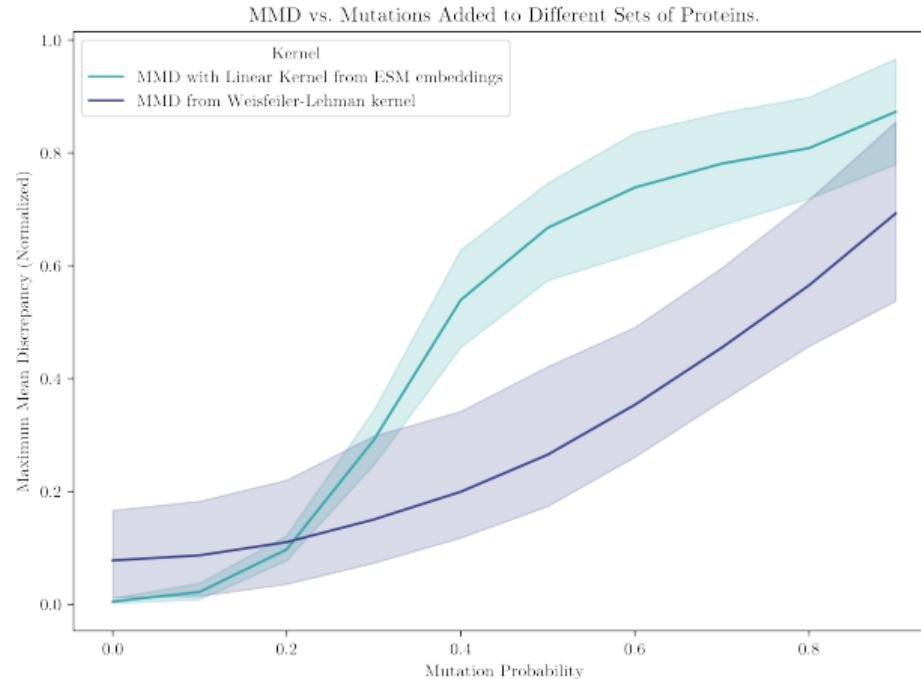
# Experiment 3 – Mutate



# Experiment 3 – Mutate



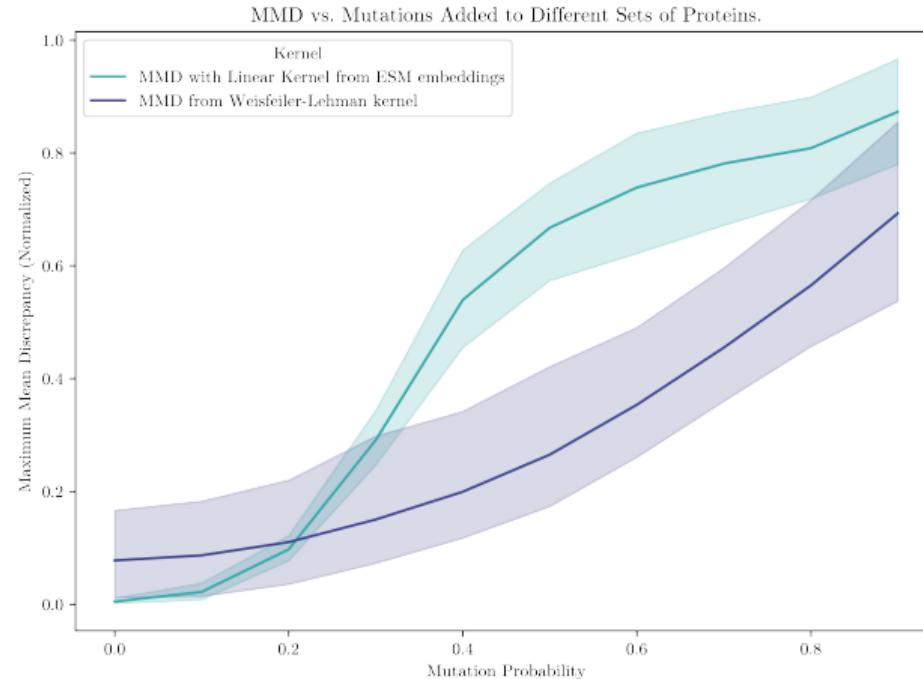
# Experiment 3 – Mutate



$n = 10$  runs

2 sources of variance:

# Experiment 3 – Mutate

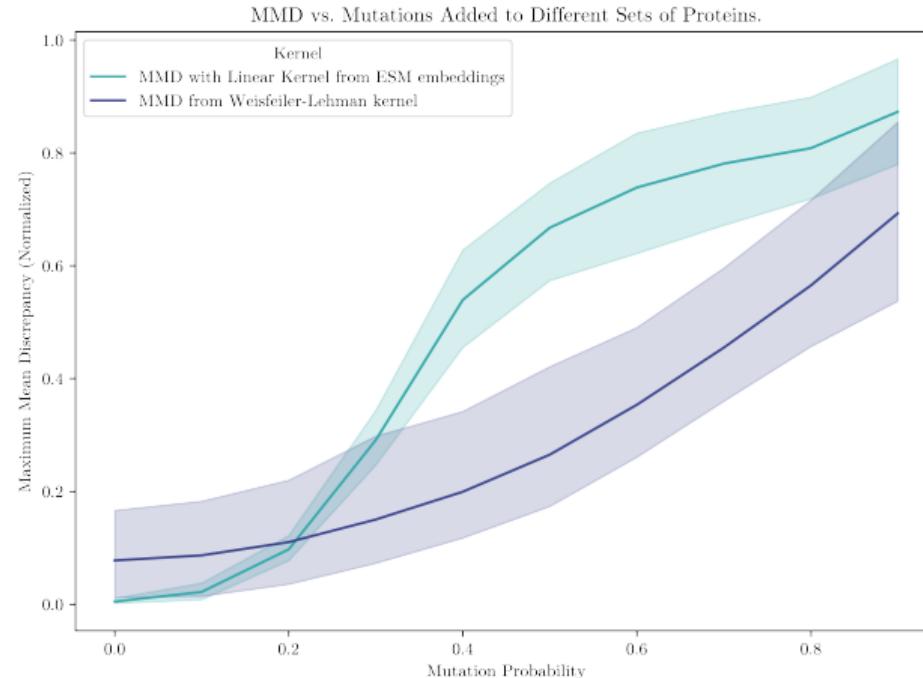


$n = 10$  runs

2 sources of variance:

- Data

# Experiment 3 – Mutate



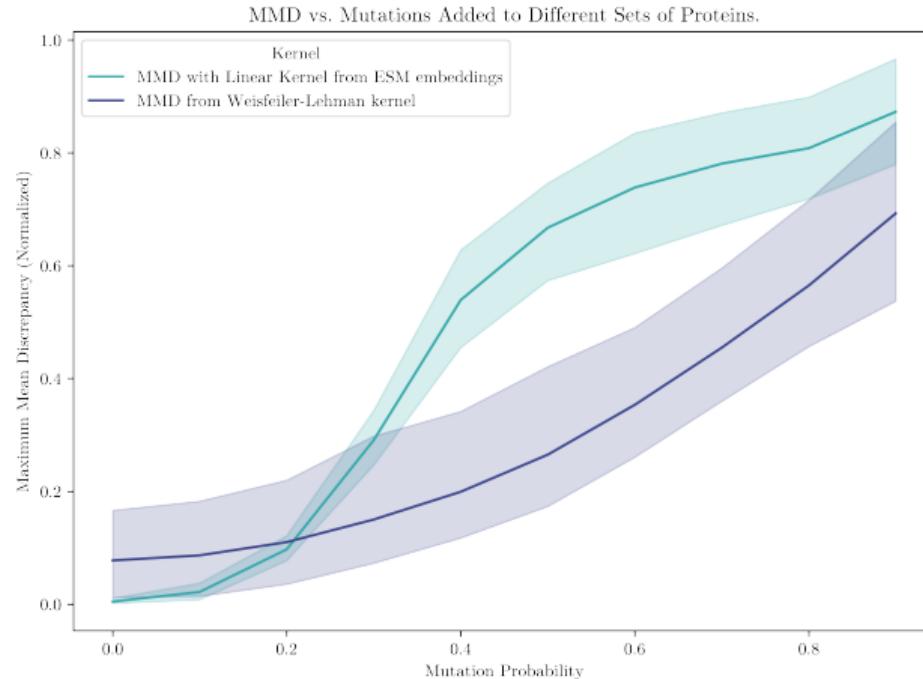
$n = 10$  runs

2 sources of variance:

- Data
- Mutation seed

Conclusions

# Experiment 3 – Mutate



$n = 10$  runs

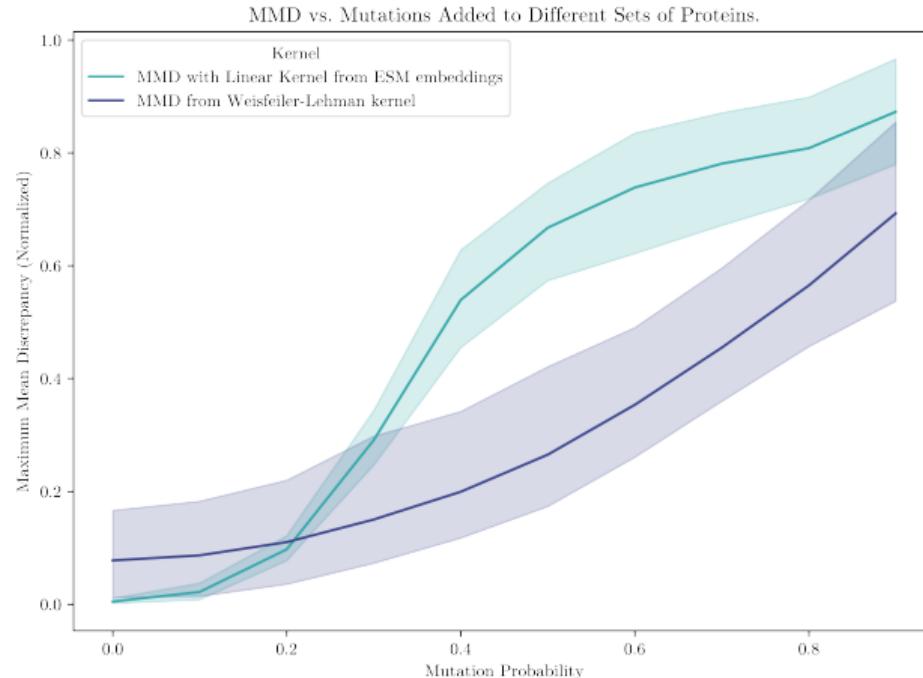
2 sources of variance:

- Data
- Mutation seed

## Conclusions

1. Weisfeiler-Lehman kernel does capture node label changes well

# Experiment 3 – Mutate



$n = 10$  runs

2 sources of variance:

- Data
- Mutation seed

## Conclusions

1. Weisfeiler-Lehman kernel does capture node label changes well
2. ESM also looks ok, but might carry bias from the data.