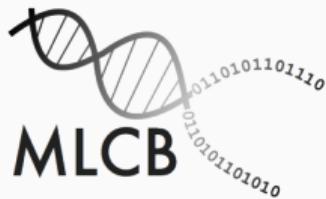


Building a library to evaluate generative protein models

Philip Hartout

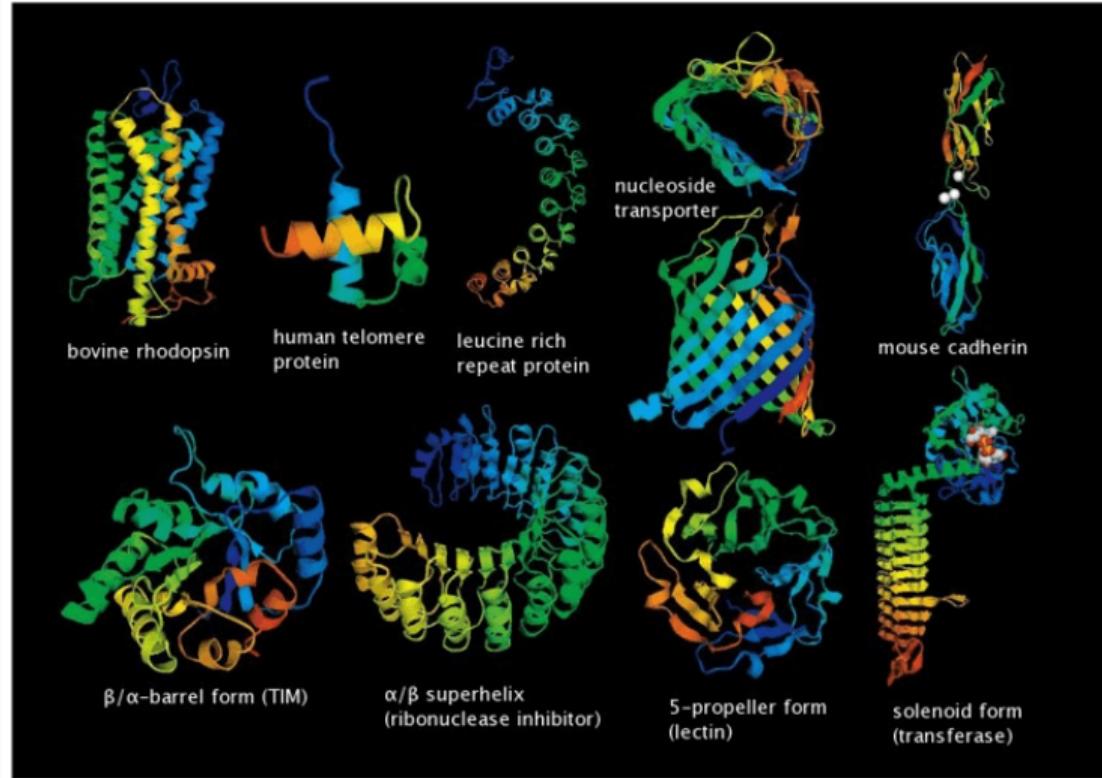
May 6, 2022



DBSSE

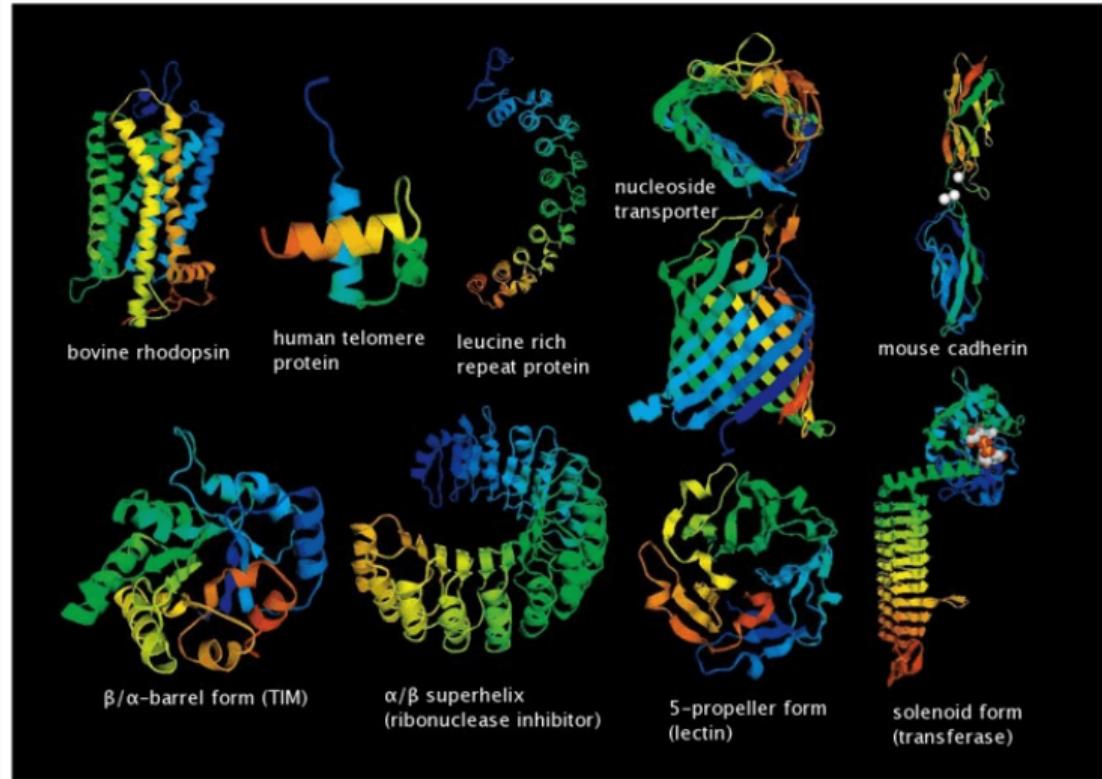
ETH zürich

Introduction



Proteins are diverse.

Introduction



Proteins are diverse.
Support all functions for
life.

Generative Protein Modelling

Proteins

Generative Protein Modelling

Proteins

- well-defined (sequence)

Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Model

Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Model

- captures $P(X)$

Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Model

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

Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Model

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



Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

Generative Model

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

Evaluation Problem



Generative Protein Modelling

Proteins

- well-defined (sequence)
- large databases

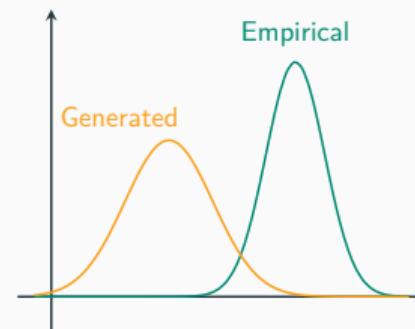
Generative Model

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

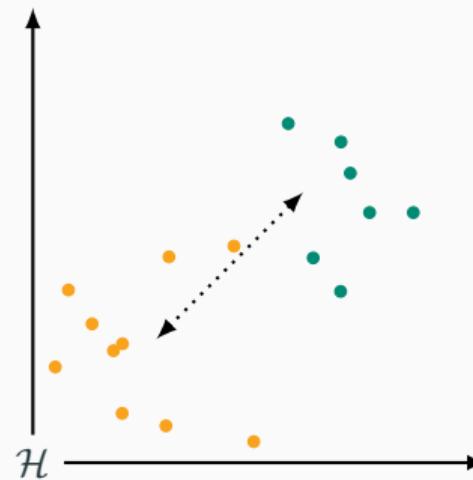


Evaluation Problem

- Are the generated and empirical data distributions the same?



Maximum Mean Discrepancy (MMD)



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

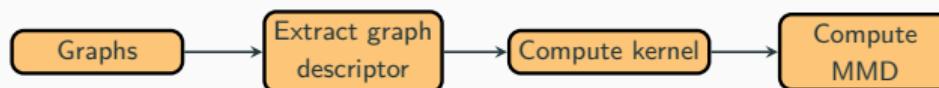
Maximum Mean Discrepancy (MMD) – continued

Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate generative GNNs.

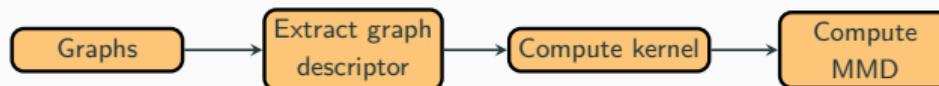
Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate generative GNNs.



Maximum Mean Discrepancy (MMD) – continued

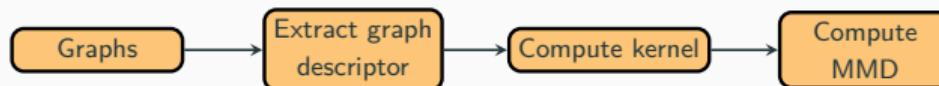
Currently accepted method to evaluate generative GNNs.



It's possible to leverage decades of kernel research!

Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate generative GNNs.

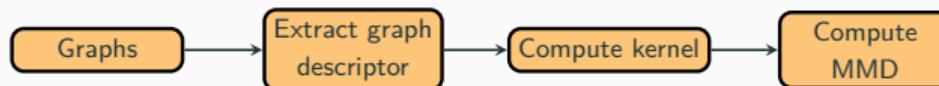


It's possible to leverage decades of kernel research!

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

Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate generative GNNs.



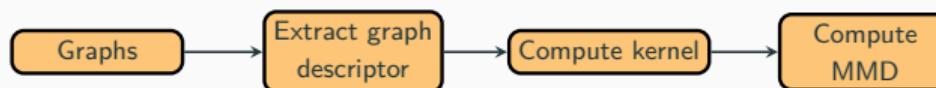
It's possible to leverage decades of kernel research!

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

Blessing Flexibility, Computation on multiple representations

Maximum Mean Discrepancy (MMD) – continued

Currently accepted method to evaluate generative GNNs.



It's possible to leverage decades of kernel research!

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

Blessing Flexibility, Computation on multiple representations

Curse Instability (see Obray et al (2021)), hyperparameter tuning.

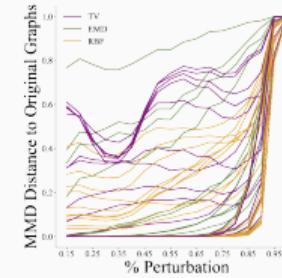


Figure 1: MMD computed from a clustering coefficient on synthetic graphs. TV: total variation kernel, RBF: radial basis function, EMD: earth mover's distance.

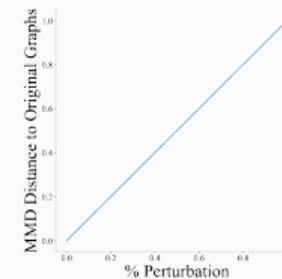


Figure 2: Ideal MMD behaviour.

– Thesis Goal –

Build a library to evaluate generative protein models

Experimental setup

Take 10 sets of 100 proteins from Alphafold DB (easy to work with)

Experimental setup

Take 10 sets of 100 proteins from Alphafold DB (easy to work with)

We will study the following settings:

Experimental setup

Take 10 sets of 100 proteins from Alphafold DB (easy to work with)

We will study the following settings:

1. Add gaussian noise

Experimental setup

Take 10 sets of 100 proteins from Alphafold DB (easy to work with)

We will study the following settings:

1. Add gaussian noise
2. Add twist

Experimental setup

Take 10 sets of 100 proteins from Alphafold DB (easy to work with)

We will study the following settings:

1. Add gaussian noise
2. Add twist
3. Add mutations

Experiment 1 – Gaussian Noise

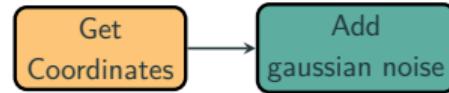
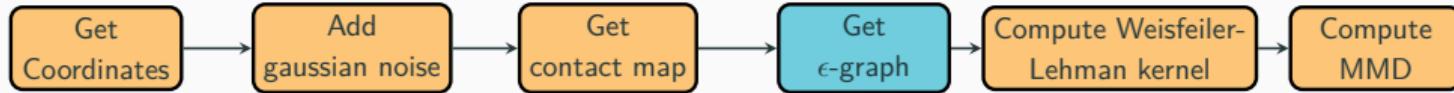
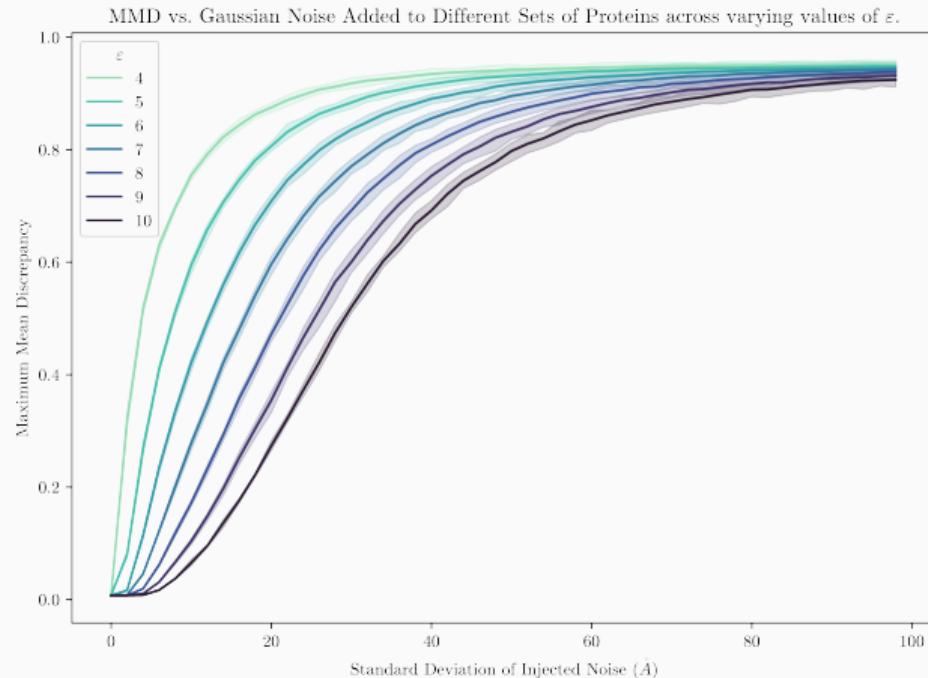
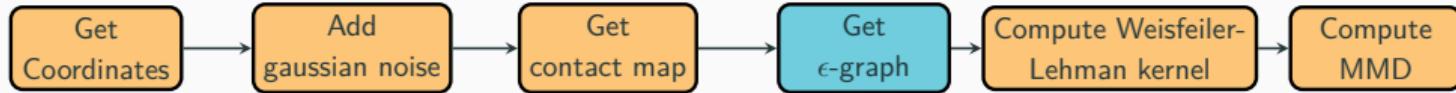


Figure 3: Adding Gaussian Noise. Protein is color-coded according to the index of each amino acid.

Experiment 1 – Gaussian Noise

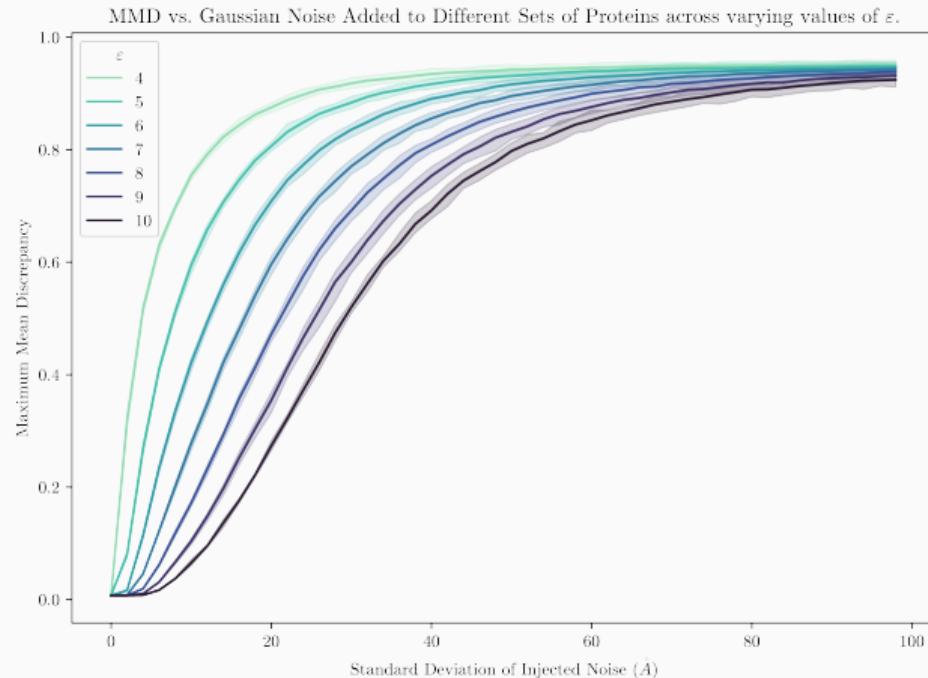
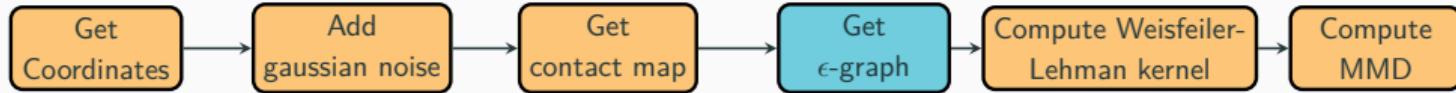


Experiment 1 – Gaussian Noise



Each curve: different ε .

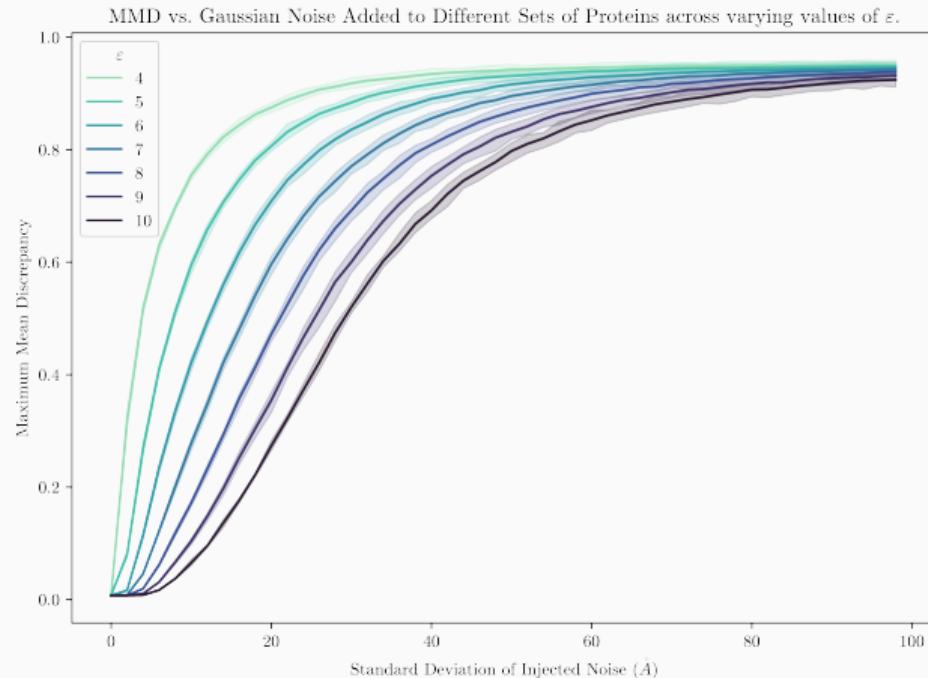
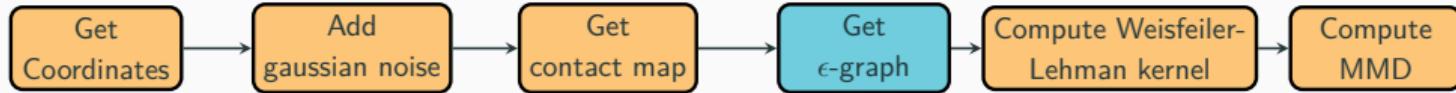
Experiment 1 – Gaussian Noise



Each curve: different ε .

2 sources of variance:

Experiment 1 – Gaussian Noise

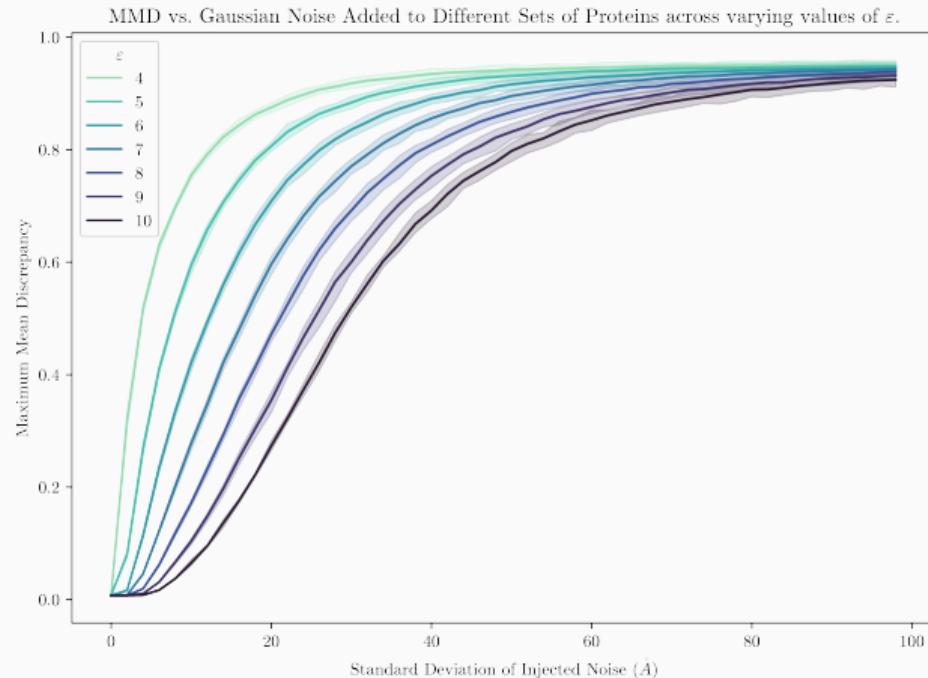
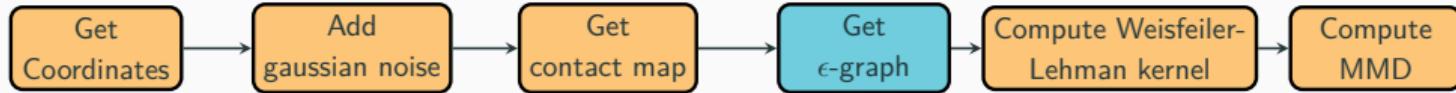


Each curve: different ε .

2 sources of variance:

- Data

Experiment 1 – Gaussian Noise



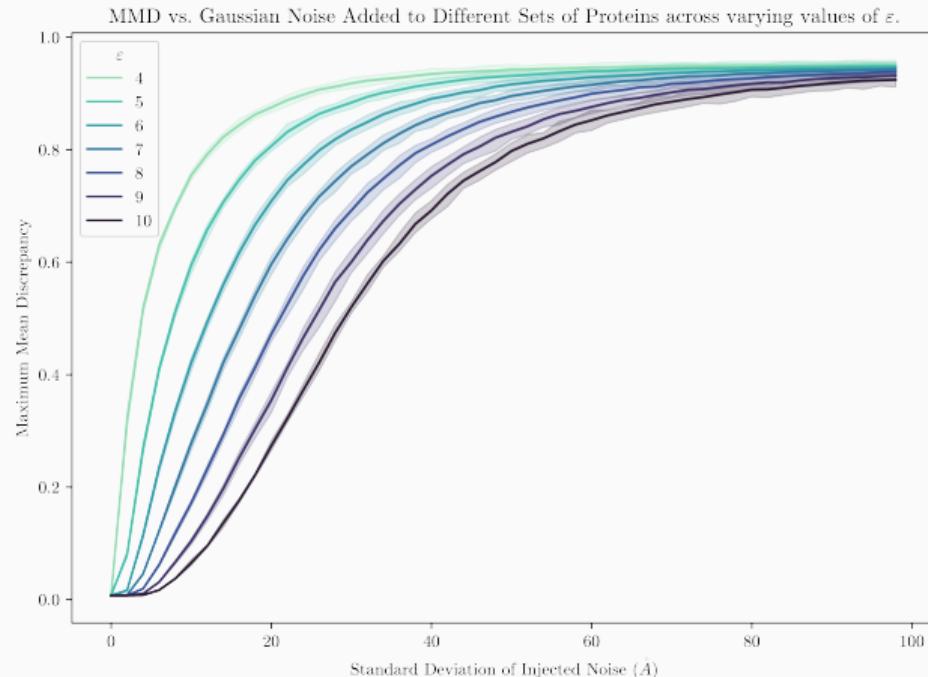
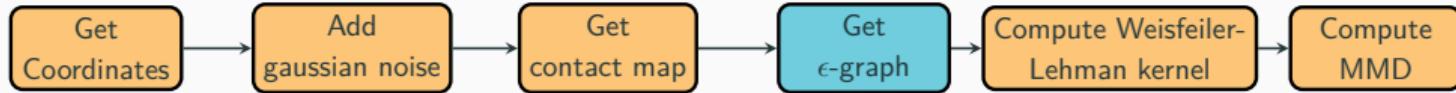
Each curve: different ε .

2 sources of variance:

- Data
- Noise

Conclusions

Experiment 1 – Gaussian Noise



Each curve: different ε .

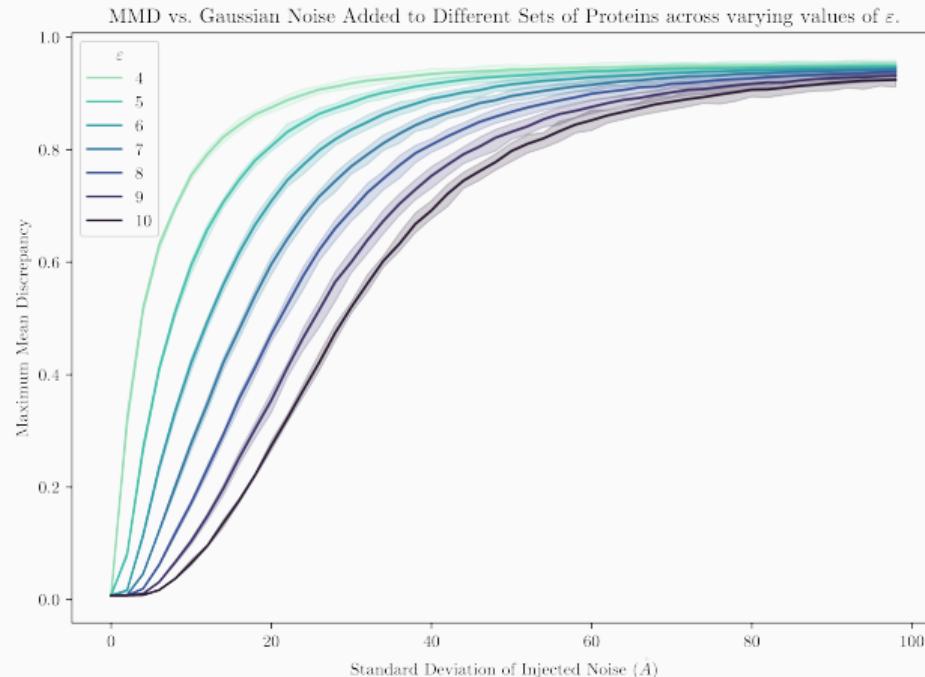
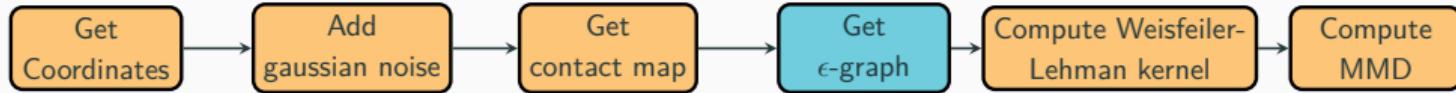
2 sources of variance:

- Data
- Noise

Conclusions

1. MMD is stable using the Weisfeiler-Lehman kernel

Experiment 1 – Gaussian Noise



Each curve: different ε .

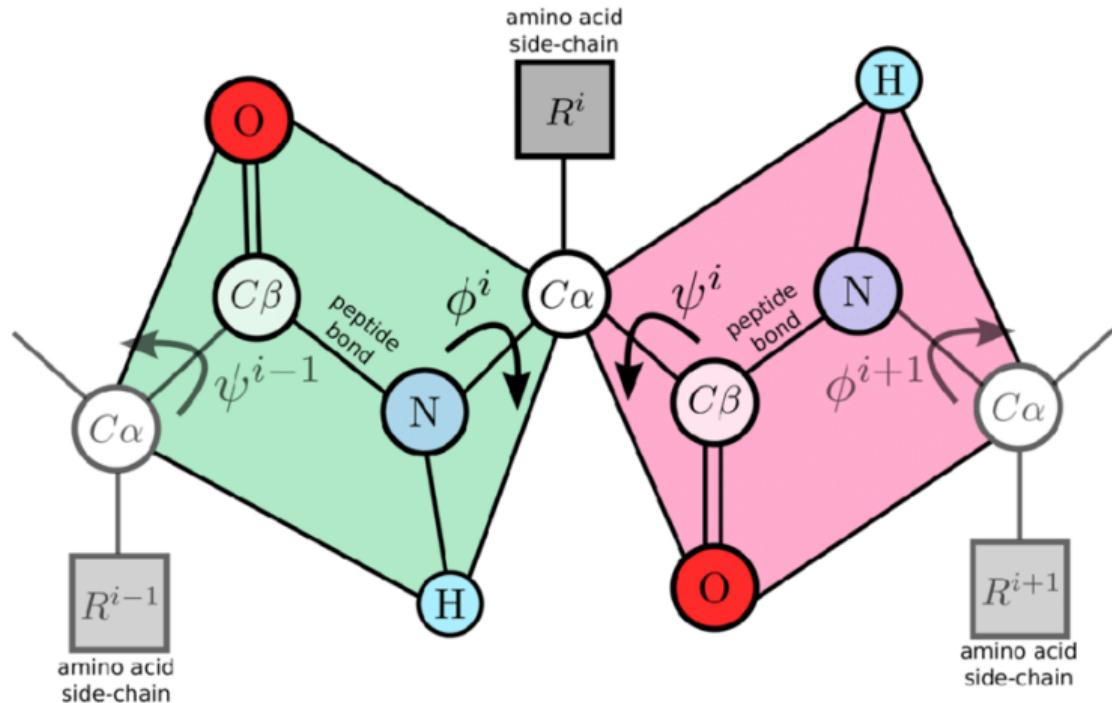
2 sources of variance:

- Data
- Noise

Conclusions

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

Dihedral angles

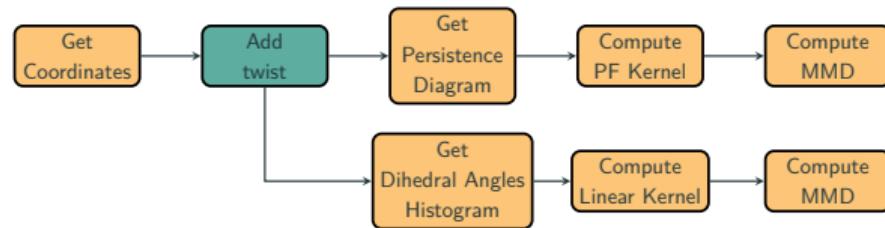


These angles *define* the shape of the protein.

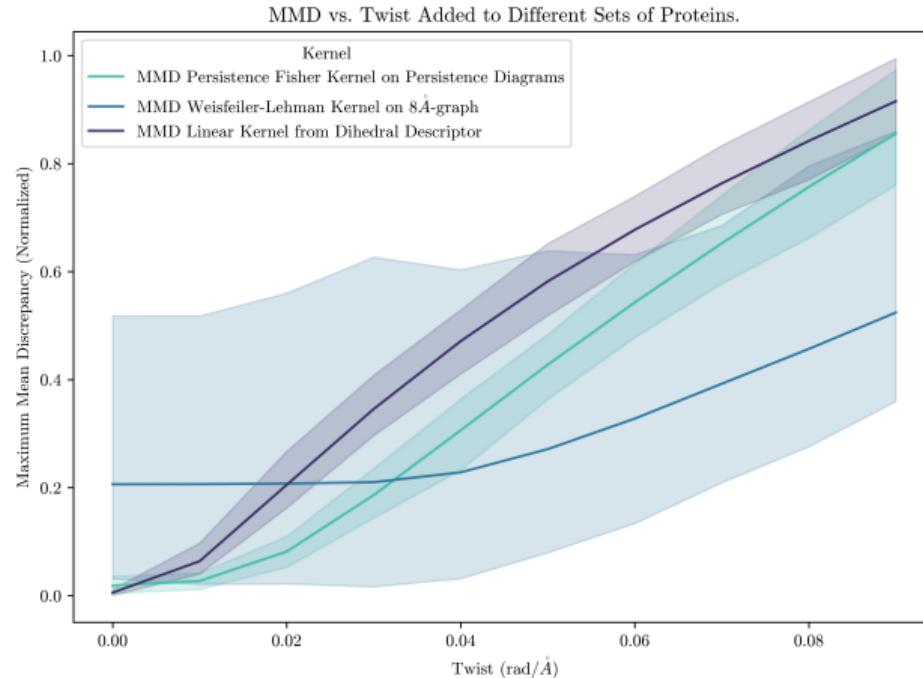
Čech filtrations

Čech filtrations captures connected components, cycles and holes by varying ε .

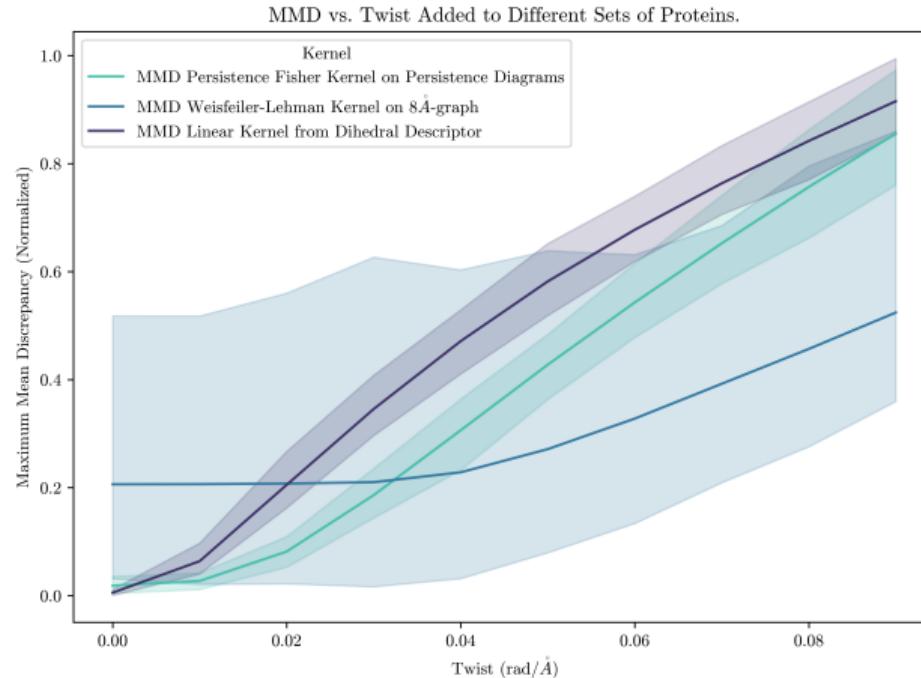
Experiment 2 – Twist



Experiment 2 – Twist

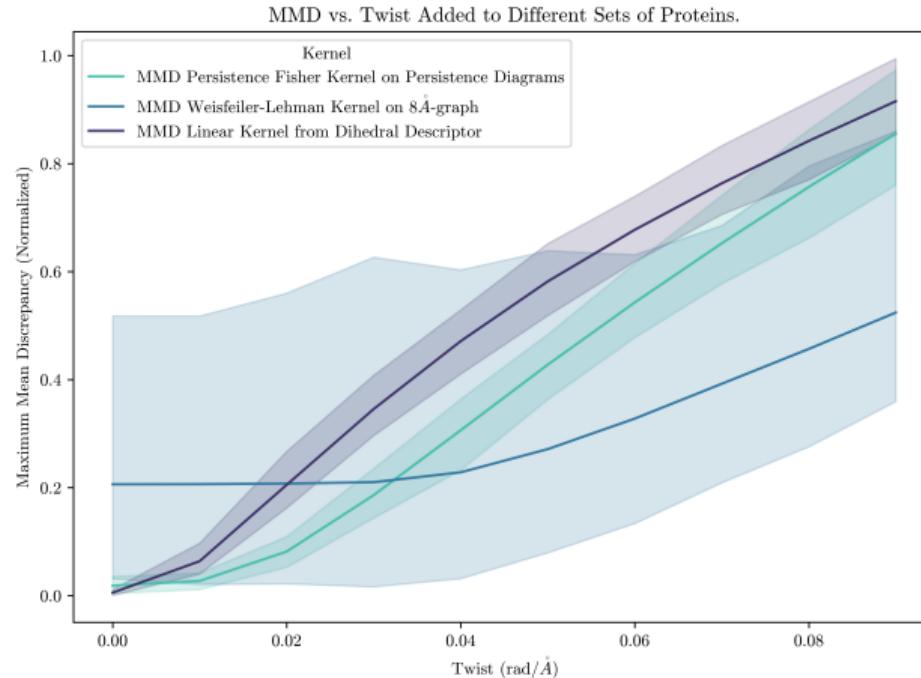


Experiment 2 – Twist



1 source of variance:

Experiment 2 – Twist

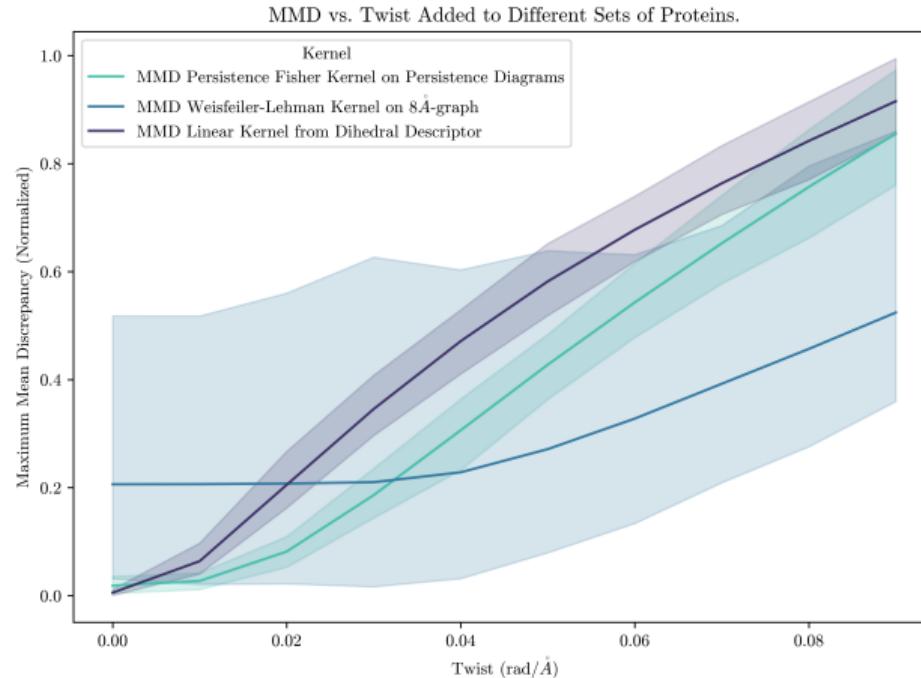


1 source of variance:

- Data

Conclusions

Experiment 2 – Twist



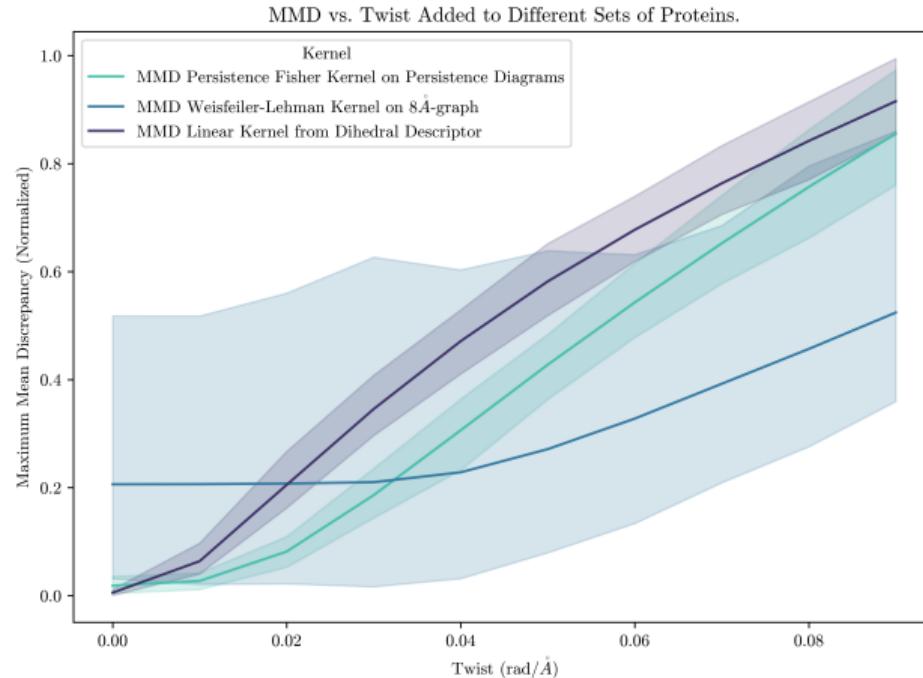
1 source of variance:

- Data

Conclusions

1. TDA behaves very well,
computationally complex

Experiment 2 – Twist



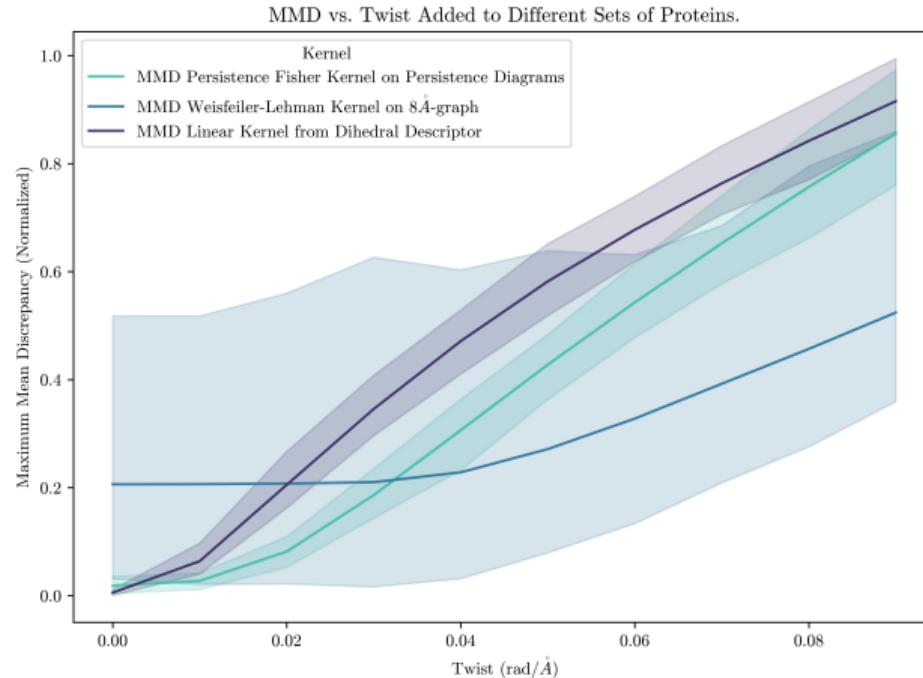
1 source of variance:

- Data

Conclusions

1. TDA behaves very well, computationally complex
2. Dihedral descriptor behave well, fast to compute

Experiment 2 – Twist



1 source of variance:

- Data

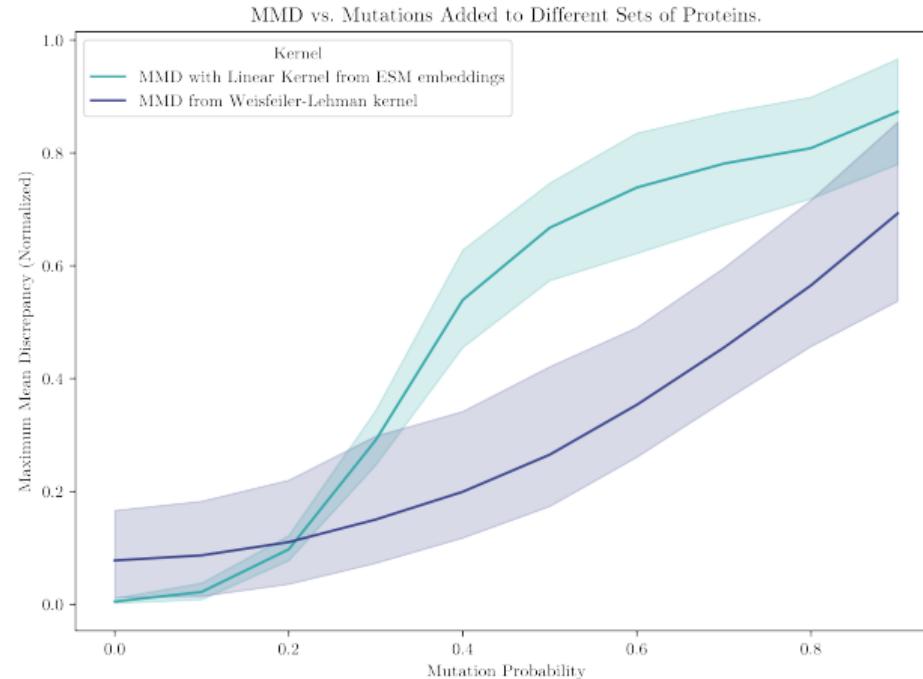
Conclusions

1. TDA behaves very well, computationally complex
2. Dihedral descriptor behave well, fast to compute
3. Weisfeiler-Lehman kernel does not capture global shape changes

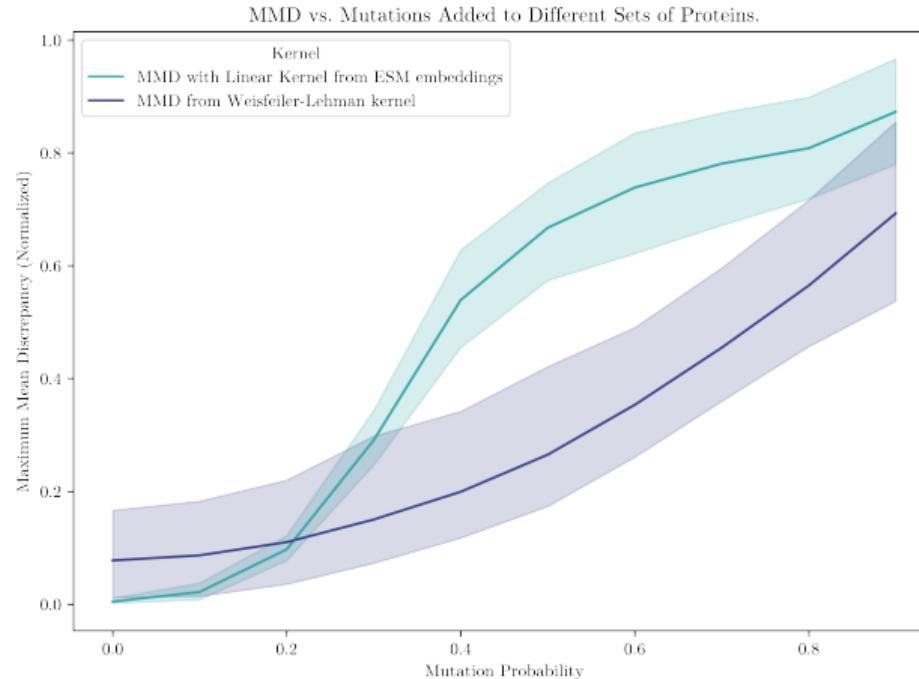
Experiment 3 – Mutate



Experiment 3 – Mutate

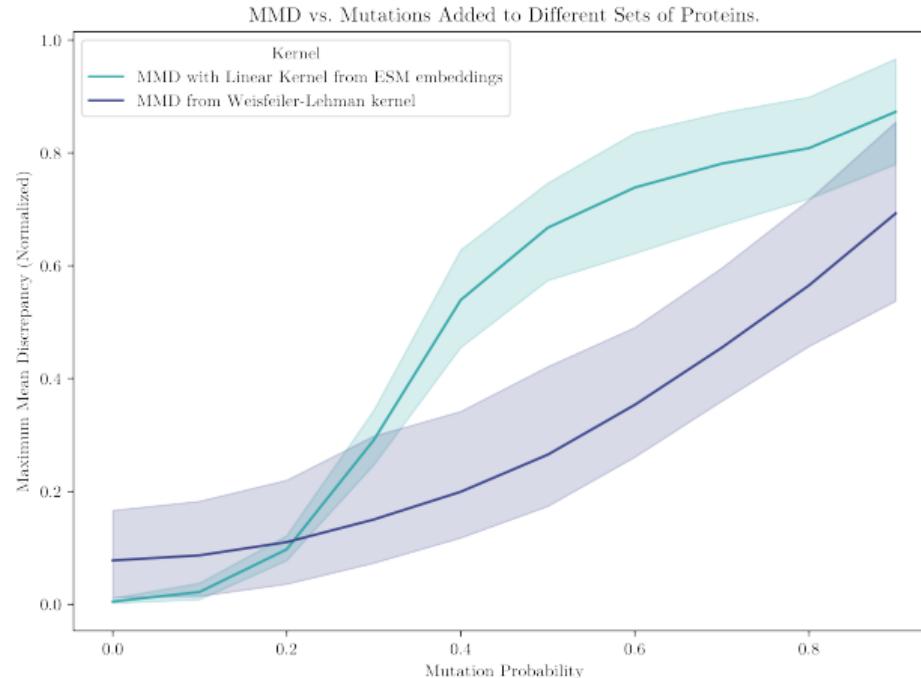


Experiment 3 – Mutate



2 sources of variance:

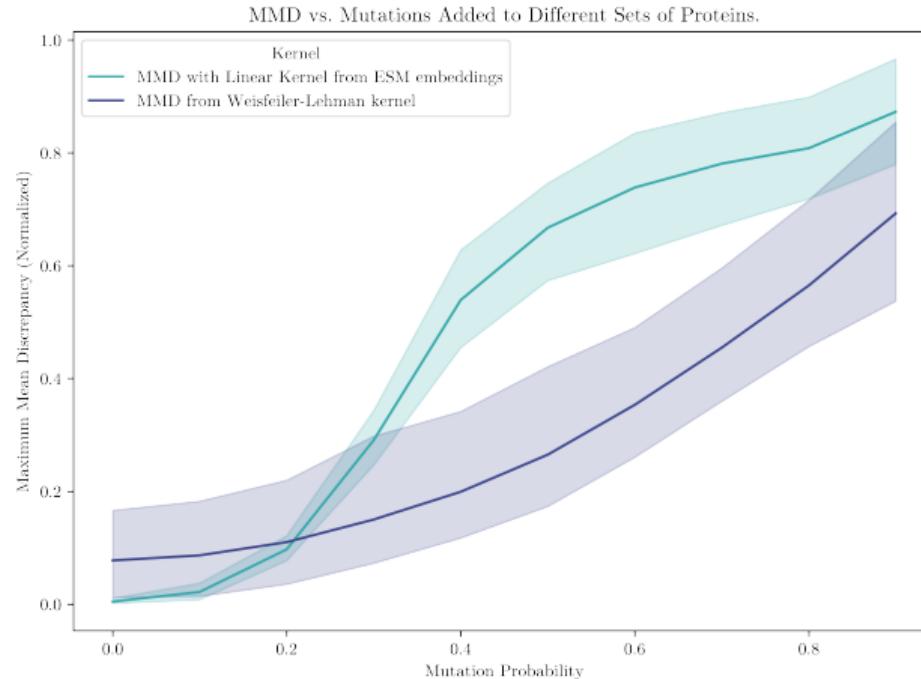
Experiment 3 – Mutate



2 sources of variance:

- Data

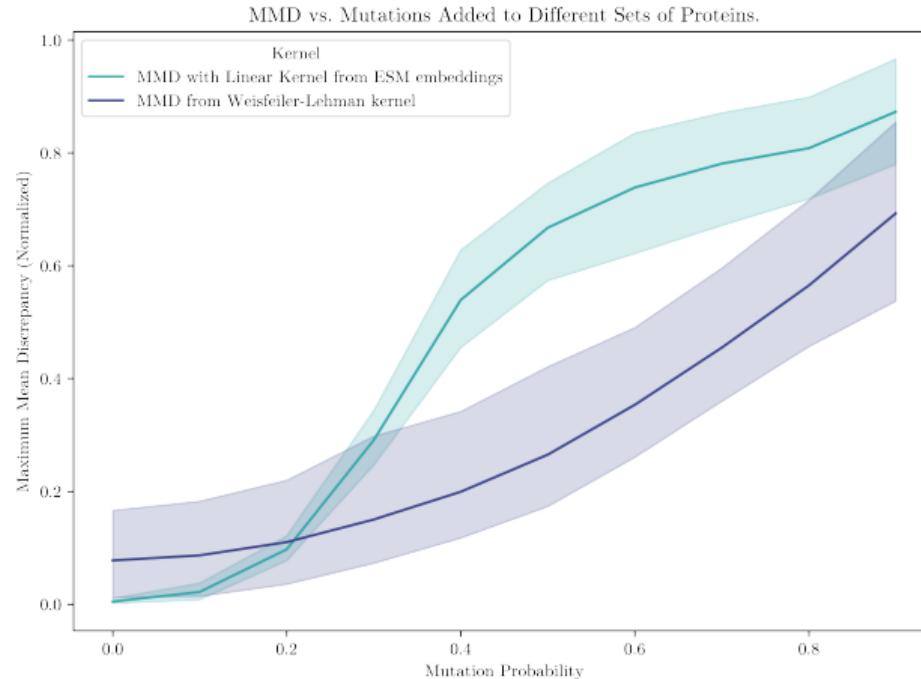
Experiment 3 – Mutate



2 sources of variance:

- Data
- Mutation seed

Experiment 3 – Mutate

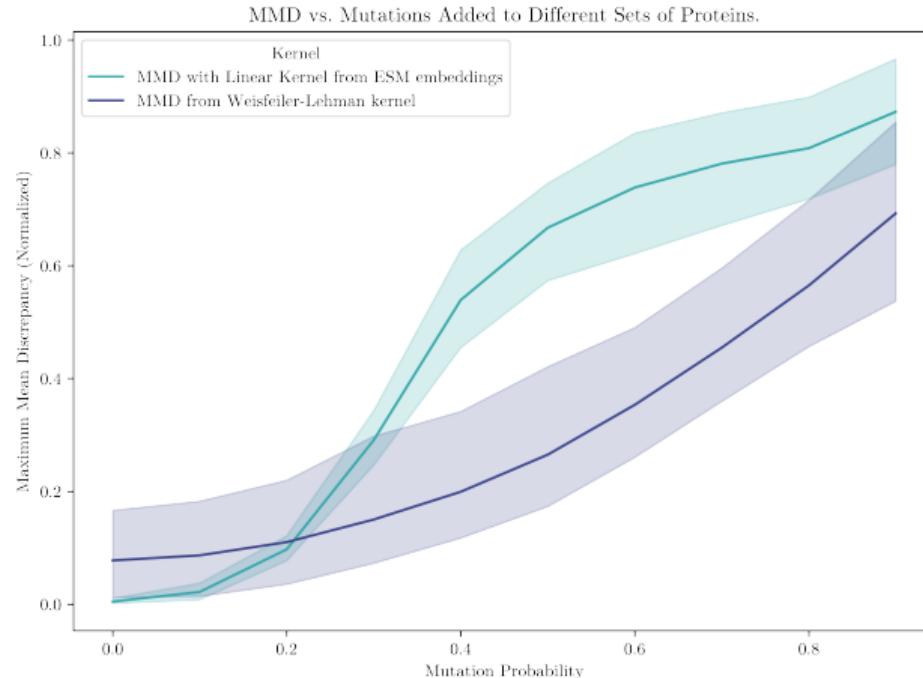


2 sources of variance:

- Data
- Mutation seed

Conclusions

Experiment 3 – Mutate



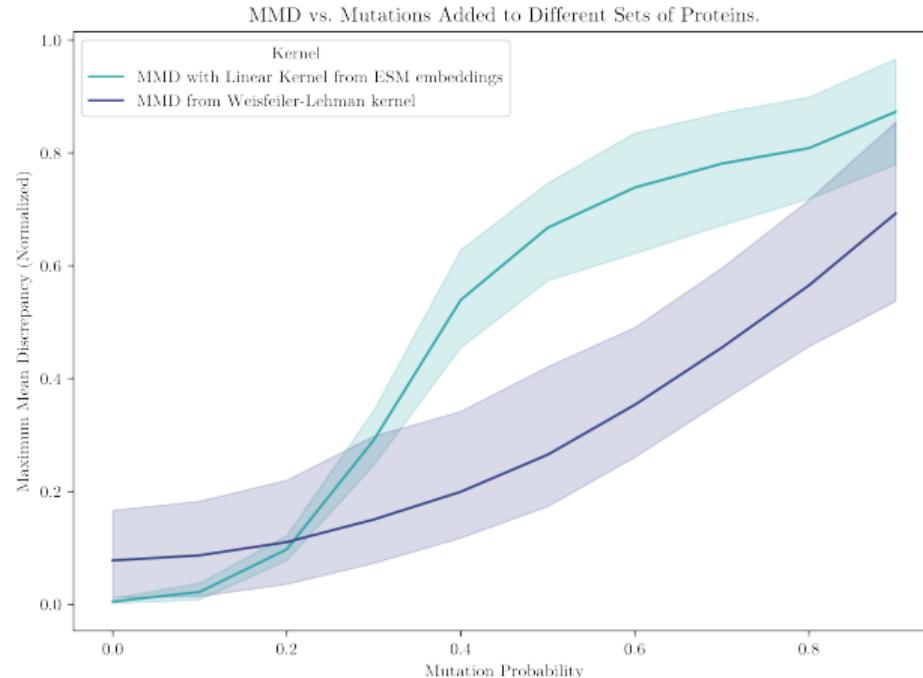
2 sources of variance:

- Data
- Mutation seed

Conclusions

1. Weisfeiler-Lehman kernel captures changes but noisy

Experiment 3 – Mutate



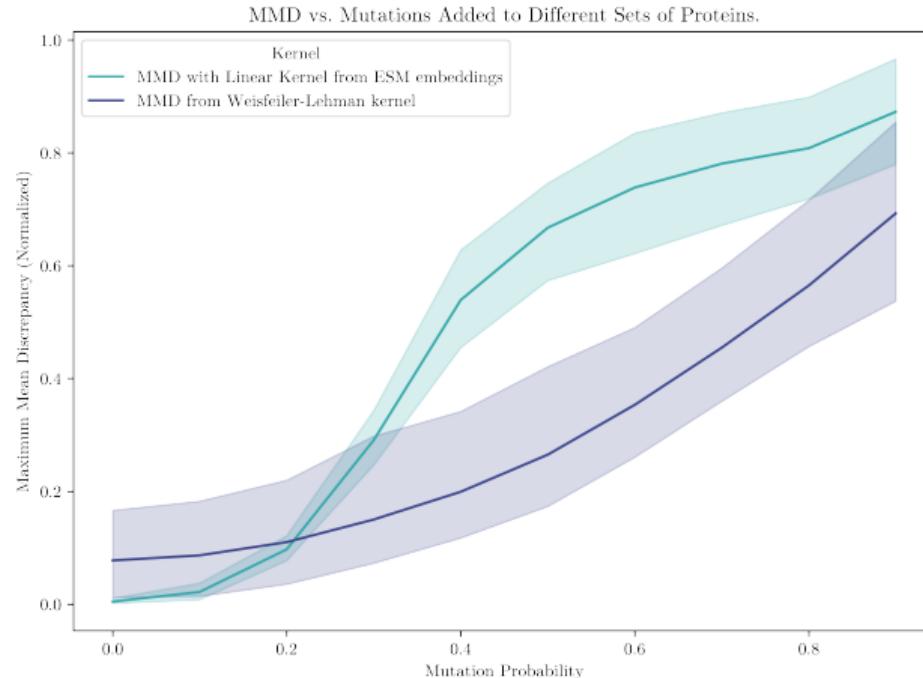
2 sources of variance:

- Data
- Mutation seed

Conclusions

1. Weisfeiler-Lehman kernel captures changes but noisy
2. ESM captures changes.

Experiment 3 – Mutate



2 sources of variance:

- Data
- Mutation seed

Conclusions

1. Weisfeiler-Lehman kernel captures changes but noisy
2. ESM captures changes.
3. Further study with lower mutation probabilities.

Questions - 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	ε graphs	k -NN graphs	Point Clouds	Sequence
Files	PDB Files			

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} .

API

```
base_feature_pipeline = pipeline.Pipeline(  
    [  
        ("coordinates", Coordinates(granularity="CA", n_jobs=12)),  
        (  
            "add gaussian noise",  
            GaussianNoise(  
                random_state=42, noise_mean=0, noise_variance=10, n_jobs=12,  
            ),  
        ),  
        ("contact map", ContactMap(metric="euclidean", n_jobs=12)),  
        ("epsilon graph", EpsilonGraph(epsilon=epsilon, n_jobs=12)),  
    ],  
)  
  
proteins_perturbed = base_feature_pipeline.fit_transform(paths_to_pdb_files)
```

```
base_feature_pipeline = pipeline.Pipeline(
[
    ("coordinates", Coordinates(granularity="CA", n_jobs=12),),
    (
        "add gaussian noise",
        GaussianNoise(
            random_seed=42, noise_mean=0, noise_variance=10, n_jobs=12,
        ),
    ),
    ("contact map", ContactMap(metric="euclidean", n_jobs=12),),
    ("epsilon graph", EpsilonGraph(epsilon=epsilon, n_jobs=12),),
]
)

proteins_perturbed = base_feature_pipeline.fit_transform(paths_to_pdb_files)

graphs = load_graphs(proteins, graph_type="eps_graph")
graphs_perturbed = load_graphs(proteins_perturbed, graph_type="eps_graph")

mmd = MaximumMeanDiscrepancy(
    biased=True,
    squared=True,
    kernel=WeisfeilerLehmanKernel(
        n_jobs=12, n_iter=5, normalize=True, biased=True,
    ),
).compute(graphs, graphs_perturbed)
```

MMD with 8- \AA -MMD with Weisfeiler-Lehman kernel

MMD of random samples from Human and E. Coli Proteome.

