ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Designing Performance Measures to Evaluate Protein Generative Models

Master Thesis

Philip Jean Hartout

July 11, 2022

Advisors: Tim Kucera, Leslie O'Bray, Prof. Dr. Karsten Michael Borgwardt

Department of Biosystems Science and Engineering, ETH Zürich

Acknowledgments

First, I would like to thank my supervisors Tim Kucera and Leslie O'Bray for their insightful and enriching discussions throughout the research and writing process of this thesis. Their support and collegiality was both very helpful and enjoyable. Second, I want to thank Prof. Dr. Karsten Borgwardt for guiding the search for the topic that lead to this thesis and providing an excellent environment to conduct the research presented in this document.

Third, I want to also thank Bas Straathof for proof-reading parts of this thesis and providing helpful feedback. Lastly, I want to thank my partner Emily, my sister Flore, my parents, and my family for supporting me throughout my studies and without whom the following could not have been achieved.

Abstract

Generative models applied to proteins are poised to revolutionize the *in silico* design of novel proteins satisfying various functional and topological constraints. However, such models are notoriously hard to evaluate. The Maximum Mean Discrepancy (MMD), a statistic used in a kernel two-sample test, has emerged as a highly versatile evaluation metric used to evaluate generative graph models. This versatility is due to the fact that any kernel, and, therefore, any appropriate underlying data representation, can be used. This makes it relevant for proteins, because they can be represented as graphs, point clouds, and sequences of amino acids. In this thesis, we aim to evaluate the applicability of different representations, descriptor functions and kernel combinations for use in Maximum Mean Discrepancy (MMD) to design relevant metrics to evaluate generative models operating in the protein domain. Through a set of graph-based and point cloud-based perturbation experiments, we first evaluate various configurations of graph descriptors traditionally used to evaluate generative graph models. Second, we expand the use of MMD to encompass previously unused configurations, such as (i) graph kernels, (ii) novel protein descriptor functions tailored to evaluate structural properties of proteins such has the dihedral angles histogram formed by each pair of amino acid, and the histogram of pairwise distances between amino acids, and (iii) kernels operating on topological descriptors of proteins. Using meta-metrics that accurately capture the *desiderata* of suitable metrics (expressivity, robustness and efficiency), we find that there are multiple configurations of MMD that accurately gauge the quality of proteins.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Generative modelling is a highly active branch of machine learning aiming to capture the distribution of a certain feature set, be it images [Saharia et al., 2022, Ramesh et al., 2022], text [Kojima et al., 2022], or graphs [Guo and Zhao, 2020]. Modelling this distribution presents a lot of advantages, but it is particularly consequential in biology [Lopez et al., 2020, Strokach and Kim, 2022]. In protein science, the application of generative models can solve the frequently occurring problem of generating novel proteins to perform a specific industrial, experimental, or therapeutic function [Jendrusch et al., 2021, Madani et al., 2020, 2021]. This process has so far been restricted to experimental techniques such as directed evolution, which is costly both in time and resources [Wang et al., 2021].

Generating novel samples has generally been most successful by using differentiable architectures such as autoregressive models, the most successful being transformers [Vaswani et al., 2017], and Generative Adversarial Networks (GANs) [Goodfellow et al., 2014]. While most of the efforts in this field have been focusing on generating novel amino acid sequences analogous to natural sequences fulfilling desired properties [Riesselman et al., 2018, Biswas et al., 2021, Weinstein and Marks, 2021, Repecka et al., 2021, Kucera et al., 2022], there have been several attempts at taking structural aspects of proteins into account, since structural features ultimately determine a protein's function [Anand and Huang, 2018, Ingraham et al., 2019, Maddhuri et al., 2021].

However, the design and improvement of generative models applied to proteins is prohibited by the lack of suitable evaluation metrics. It is notoriously hard to find expressive, robust and efficient performance measures that accurately gauge the quality of generated samples *in-silico* [Theis et al., 2016, Betzalel et al., 2022]. Efforts have been made to solve this challenge in the image domain by using the fixed-length representations obtained from a neural network such as the Inception v3 network, and subsequently calculating a Wasserstein distance between the set of generated samples and test data using such representations [Heusel et al., 2017]. Recently, the community has investigated this challenge specifically for a common representation of proteins: graphs [Thompson et al., 2022, O'Bray et al., 2022]. In this domain, the standard measure used is a highly versatile statistic for a kernel two-sample test called Maximum Mean Discrepancy (MMD) introduced by Gretton et al. [2012]. MMD computes the distance between two sets of data by computing a distance measure in a Reproducing Kernel Hilbert Space (RKHS), so its versatily stems from the fact that any valid kernel and appropriate underlying data representation can be used. However, this framework has yet to be applied to proteins, with practitioners mostly relying on either sequence model evaluation metrics such as perplexity scores [Belinkov and Glass, 2019, Ingraham et al., 2019, Hesslow et al., 2022], physics-based tools to validate 3D structures such as RosettaRemodel [Huang et al., 2011, Anand and Huang, 2018, Ingraham et al., 2019], which has sometimes limited applicability [Leman et al., 2020], or experimental validation [Strokach et al., 2020]. While experimental validation is the ultimate metric, better metrics for the fast *in-silico* assessment of generative proteins is required, and the flexibility of MMD could be leveraged to fill this gap.

O'Bray et al. [2022] recently performed an in-depth evaluation of MMD, and their results unveiled a number of pitfalls related to MMD. Depending on the kernel and its associated parameters, different MMD configurations ranked the quality of samples generated by different models differently. In addition, when progressively perturbing a set of synthetic graphs, certain MMD configurations with respect to another set of unperturbed graphs issued from the same distribution was not always found to monotonically increase with increasing amounts of perturbations, casting doubt on the expressivity of MMD in certain configurations and data.

In this thesis, we set out to quantify the quality of different MMD configurations on protein data sets and perform a meta-evaluation of MMD-based metrics. We investigate the values of various MMD configurations by applying perturbations to one set of proteins. These perturbations were actively designed to be relevant for protein design use cases and include sequence perturbations, graph perturbations and geometric perturbations. In this setting, we investigate frequently used MMD configurations typically used to evaluate generative models in the graph domain. In addition, we expand this set of configurations by devising novel combinations of protein representations, descriptors, and kernels, all relevant for different aspects of protein design, and integrate them to the well-established MMD evaluation framework.

This thesis is organized as follows: first, Chapter 2 introduces fundamental concepts that we are going to build upon, and discusses the surrounding literature. Chapter 3 details the methodology of the experiments that we carry out in this thesis, as well as describes all the configurations of MMD that we will explore (i.e. all combinations of representations, descriptor functions, kernels, and kernel parameters). Chapter 4 presents and contextualizes the findings of those experiments. We summarize key findings, make recommendations for practitioners and highlight some limitations and directions for future work in Chapter 5 before concluding in Chapter 6.

Chapter 2

Background and Related Work

This chapter introduces fundamental concepts built upon in this thesis, which lies at the interface between structural biology and machine learning. We start by defining some relevant biological properties of proteins, as well as several representations leveraged in later chapters. We then introduce generative models, applied to graphs and other domains. Crucially, we discuss in detail the evaluation problem when it comes to generative models, as well as some of the unique challenges arising in the graph domain. Furthermore, we discuss the current landscape of methods used to evaluate graph generative models, such as the Maximum Mean Discrepancy (MMD). Finally, we introduce kernels that can be used within the MMD computational framework.

2.1 Proteins

Proteins are large biomolecules that are formed from a sequence of amino acids, performing their functions as determined by their three-dimensional structure, which in turn is determined by their amino acid sequence. They support a vast array of functions in living organisms, such as catalyzing metabolic reactions, DNA replication, providing structural support to cells, transporting molecules and sensing stimuli.

Each protein is made up of one or more chains of amino acids and contains a backbone and different side chains. The atoms in the backbone include an α-carbon, another carbon (the β-carbon), and a nitrogen atom. An overview of the peptide backbone is shown in Figure 2.1. α-carbons also form the anchor point of the *side chain of* each amino acid, which endows each amino acid with various chemical properties related to acidity, polarity, electronic charges, etc. These side chains together with their collective geometries enable protein to interact with substrates, other proteins, or form structural backbones for higher-order biomolecules [O'Connor and Adams, 2010].

α-carbons also play a role in the geometry of the protein backbone. Interestingly, a plane is formed by two alpha carbons, the carboxyl group, and the hydrogen atom attached to the nitrogen atom (see Figure 2.1), making the peptide bond between the nitrogen and carbon atom resistant to twisting. That means that the rotations of these planes enabling the 3D folding of a protein are governed by the angle of the bonds linking the nitrogen atom to the α-carbon and the other carbon atom to the α-carbon, named φ and ψ. These angles' values are frequently used to validate proteins [Gore et al., 2017], or characterize the secondary structure of proteins [Wood and Hirst, 2005].

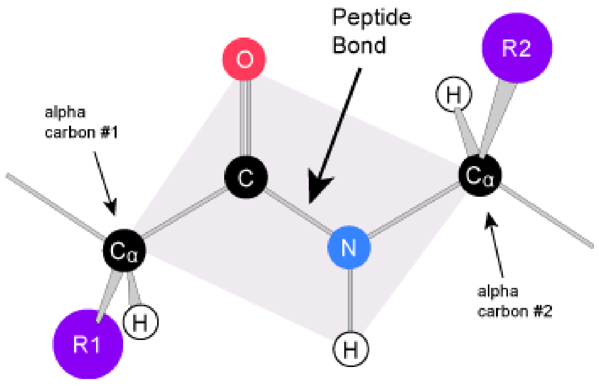


Figure 2.1: Schematic of the backbone of a protein. Two α-carbons are shown as well as a β-carbon in the middle. R1 and R2 represent the side chains of the amino acid. Image by Marc T. Facciotti.

To visualize such angles, a Ramachandran plot can be constructed for any protein [Ramachandran et al., 1963], where the x-axis represents the value of the φ-angles and the y-axis represents the ψ-angles. Such a plot can reveal secondary structural features such as β-sheets, α-helices, etc. An example of a Ramachandran plot together with a 3D model of a protein can be found in Figure 2.2.

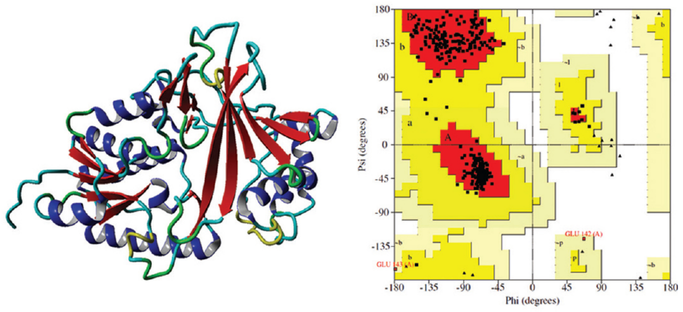


Figure 2.2: 3D structure of uridine diphosphogalactofuranose-galactopyranose mutase with a corresponding Ramachandran plot. The α-helices can be found on the middle left part of the Ramachandran plot, the β sheets on the upper right quadrant, and the left handed α-helices can be found in the middle upper right part of the plot. This figure is adapted from Nayak et al. [2018].

2.2 Graphs

Proteins are often abstracted using graphs [Anand and Huang, 2018, Ingraham et al., 2019]. A graph G is a pair of vertices V and edges E such that $G = (V, E)$, $|V| = n$ and $|E| = m$. Two vertices i and j are adjacent if there is an edge between them, i.e. $e_{ij} \in E$. The relationship between nodes can be represented as an $n \times n$ adjacency matrix A, where:

$$A_{ij} = \begin{cases} 1, & \text{if } e_{ij} \in E \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

In the case of a *weighted* graph, 1 is then substituted by a weight w in Equation 2.1.

The neighborhood of a node v is the set of nodes with an edge directly to v, i.e. $N(v) = \{u \in V | e_{uv} \in E\}$. A graph is undirected if the edges do not contain directional information, i.e. $A_{ij} = A_{ji}$. A directed graph would result in directionality being encoded in edges, where A_{ij} would not contain any information about A_{ji}. Nodes and edges in each graph can contain one or more labels. In this thesis, we will mostly deal with labeled undirected graphs, where each node will be labeled according to the amino acid type that it belongs to.

There are multiple ways of constructing graphs from proteins. First, one can extract a *contact map* of a protein by computing the (Euclidean) distance between any two points belonging to each amino acid. The α-carbon is often used for this purpose [Anand and Huang, 2018, Ingraham et al., 2019]. This is a fully connected graph with weighted edges representing the distance between each pair of nodes. From there, it is possible to either extract a k-nearest neighbor (k-NN) graph, where $k \in \mathbb{N} > 0$ defines the number of nodes directly connected to any given node; or an ε-graph, where each node within a given distance $\varepsilon \in \mathbb{R}^+ \setminus \{0\}$ of another node is connected. Both are graphs where each node is labeled with the residue name to which the α-carbon belongs and the edges are unlabeled.

2.3 Topological Data Analysis

Although graphs are a powerful representation of proteins, the latter can also be represented as *point clouds*. One powerful field of study of topological properties of point clouds (among other structured data) is *topological data analysis*.

Topology has witnessed relentless theoretical progress since Henri Poincaré first addressed topological ideas as a distinct branch of mathematics in his 1895 publication of *Analysis Situs* [Poincaré, 1895]. Only recently – with the advent of modern computing – has the field of computational topology and Topological Data Analysis (TDA) gained momentum to investigate (high-dimensional) data in physics, biology, and beyond [Dey et al., 1999, Ghrist, 2008, Amézquita et al., 2020]. For material providing an extensive and formal introduction to topology and persistent homology, please refer to Freedman and Chen [2009], Edelsbrunner and Harer [2010], and Ghrist [2008].

A powerful computational technique to analyze topological properties of point clouds is *persistent homology*, which first requires us to define simplicial homology. Simplicial homology refers to a way of assigning connectivity information to topological objects, such as point clouds, which are represented by simplicial complexes. A simplicial complex K is a set of simplices that correspond to vertices in dimension 0, edges in dimension 1, and triangles in dimension 2. The subsets of a simplex $\sigma \in K$ are referred to as its faces, and each face $\tau \in K$. Moreover, any non-empty intersection of two simplices also needs to be part of the simplicial complex, i.e. $\sigma \cap \sigma' \neq \emptyset$ for $\sigma, \sigma' \in K$ implies $\sigma \cap \sigma' \in K$, meaning that K is closed under calculating the faces of a simplex.

Persistent homology extends simplicial homology by employing filtrations to imbue K with scale information. This process captures rich topological information related to K in a principled way. The filtration process is generally defined by a function $f : K \to \mathbb{R}$ satisfying some finite number of values m and $f^0 \leq f^1 \leq \cdots \leq f^{m-1} \leq f^m$. This allows us to sort K using f, for instance by extending f linearly to higher-dimensional simplices via $f(\sigma) := \max_{v \in \sigma} f(v)$, leading to a nested sequence of simplicial complexes like so:

$$\emptyset = K^{(0)} \subseteq K^{(1)} \subseteq \cdots \subseteq K^{(m-1)} \subseteq K^{(m)}, \tag{2.2}$$

where $K^{(i)} := \{\sigma \in K \mid f(\sigma) \leq f^{(i)}\}$. This relationship enables tracking the appearance or birth (i.e. a connected component arising) and the dissapearance or death (i.e. two connected components merging into one) of topological features across scales as one transitions from $K^{(i)}$ to $K^{(i+1)}$. The birth and death of topological features for different values of f are usually summarized in a *persistence diagram*, which is a multiset of tuples, each of which contains the values at which each feature is born or dies[1].

A common construction for obtaining such features is the Vietoris-Rips complex [Vietoris, 1927]. It requires a distance threshold ε and a metric $\dot{(\cdot,\cdot)}$ (usually, the Euclidean distance, as we will use in this thesis). The Vietoris-Rips complex at scale ε of an input protein point cloud is defined as $\mathcal{V}_\varepsilon(X) := \{\sigma \subseteq X | \dot{(x(i), x(j))} \leq \varepsilon\}, \forall x(i), x(j) \in \sigma$, i.e. $\mathcal{V}_\varepsilon$ contains all

[1]The name persistence diagram is derived from the observation that points far from the diagonal line in the diagram are deemed persistent, because they span a high range of values of the filtration function.

subsets of the input space whose pairwise distances are less than or equal to ε. $\mathcal{V}_\varepsilon$ is conceptually very similar to the ε-graphs discussed in section 2.2, except that ε here ranges over the entire space of possible distance values, and $\mathcal{V}_\varepsilon$ also tracks topological features over all three dimensions, instead of only connected nodes.

Note that the multiplicity of the persistence diagram corresponds to the number of homology dimensions under study. In this thesis, given proteins are represented as three-dimensional point clouds, we choose to track topological features across three homology dimensions: 0, 1 and 2. Effectively, this tracks connected components in dimension 0, circular holes in dimension 1, and two dimensional voids or cavities in dimension 2 as the filtration function is applied. For a more thorough introduction to homology and homology groups, please refer to Edelsbrunner and Harer [2010].

2.4 Generative Models

While discriminative machine learning techniques aim to learn some dependent variable $\mathcal{Y}$ from a set of (independent) features $\mathcal{X}$, generative machine learning models generate synthetic samples $\mathcal{X}'$ following the distribution of $\mathcal{X}$. Computing such probabilistic distributions through maximum likelihood estimation and related methods is intractable in many cases [Rayner and MacGillivray, 2002, Drovandi and Pettitt, 2011] and real-world scenarios [Yıldırım et al., 2015]; as such, new learning paradigms were established to enable the modeling of complex, real-world distributions through gradient-based methods [Bond-Taylor et al., 2021].

The earliest generative models were based on Hidden Markov Models (HMMs), where one estimates the hidden parameters of the distributions emitting the observed samples [Baum and Sell, 1968, Baum et al., 1970]. However, this process assumes a Markov process, whereby earlier elements of a sequence of observes do not influence the current state being estimated, which is particularly prohibitive in real world contexts. As such, more powerful models were required.

One seminal method transforming the field of generative modeling was that of generative adversarial learning, which was pioneered by Goodfellow et al. [2014], where a (deep) generator is pitted against a (deep) discriminator. The former's goal is to generate samples identical to the training distribution, while the latter is to classify whether the sample originated from the generator or the training distribution. Simultaneously developed methods by Kingma and Welling [2013] introduced a similar framework rooted in probability theory and introduced Variational Auto-Encoders (VAEs), where instead of a discriminator, the second network leverages the representation of the generator to perform approximate inference. In both cases, the two

Figure 2.3: Sample images generated by StyleGAN-XL, the state-of-the-art GAN by Sauer et al. [2022] at the time of writing.

networks (i.e. the generator and the discriminator/inference network) are jointly trained using backpropagation to minimize some appropriate loss function. Autoregressive models such as the transformer architecture introduced by Vaswani et al. [2017] leverages masking to perform next-token predictions, and has also seen success in domains such as graphs since [You et al., 2018]. A recent review of the existing landscape of generative modeling methods has been provided by Bond-Taylor et al. [2021].

Differentiable generative modeling techniques have been particularly successful in the image domain [Sauer et al., 2022, Ramesh et al., 2022, Saharia et al., 2022], a testament to the fact that modern Generative Adversarial Networks (GANs) have been able to tackle multiple practical challenges such as mode collapse[2] and convergence failure[3] to produce realistic images, such as the sample seen in Figure 2.3. More pertinent to this thesis is the application of generative models to graphs. The application domain has been reviewed by Zhou et al. [2020]. In short, graph generative networks are capable of operating on the highly versatile and extensive graph domain. It has been shown that they can produce small molecules as well as generate social networks, and knowledge graphs, among many other real-world tasks. Generative networks can be grouped into two categories: those that generate nodes in each graph sequentially, such as GraphRNN by You et al. [2018], and those that generate graphs from some latent distribution directly using GANs, such as MolGAN by De Cao and Kipf [2018], or using VAEs [Grover et al., 2019].

Operating in the graph domain incurs some unique challenges. From a modeling standpoint, dealing with graphs means dealing with a much larger and variable output space. In the general case, at least n^2 values need to be specified. Additionally, the number of edges and nodes varies from sam-

[2]We define mode collapse as the situation when a particular type of generated output (i.e. intra-mode outputs) lacks variety (see also Section 5.3.3)

[3]This refers to when a model cannot reach an even remotely optimal set of parameters to reduce a particular loss function.

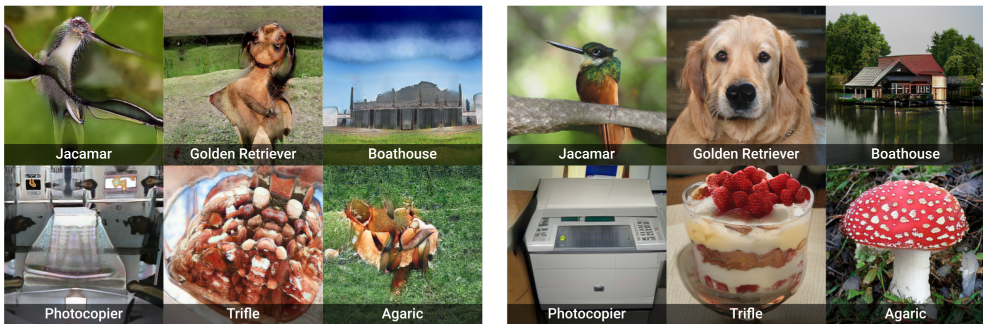


Figure 2.4: Class-conditional samples generated by StyleGAN3 (left) and StyleGAN-XL (right) trained on the same dataset at the same resolution. This figure is adapted from Sauer et al. [2022]. The pathologies seen here are diverse, but we can see that it seems that the model on the left does not seem to be powerful enough to situate various animal body parts with respect to one another; a similar phenomenon can be seen in objects where symmetries and higher order structure seems to be difficult to model.

ple to sample, which also needs to be accounted for in the model structure. Furthermore, by building a generative model generating graphs of up to n nodes, $n!$ possible and equivalent adjacency matrices can be generated. Such a high representation complexity is challenging to model, difficult and expensive for objective functions to optimize, and difficult to evaluate. The last modeling-related issue when dealing with graphs is that the presence of one edge is not independent of another, i.e. real-world graphs often exhibit patterns of local connectedness which need to be accounted for in the model.

2.5 The Evaluation Problem

Perhaps the most significant problem plaguing all generative models is the evaluation problem, which consists in evaluating the quality of generated samples from a model with respect to a test set. While sidestepping the problem is possible in the image domain by manually inspecting generated samples, a practice that might reveal interesting modelling pathologies (see Figure 2.4), this cannot be done at scale, nor can it be done for generative models operating in the graph domain, where human perception cannot easily evaluate the quality of a set of generated graphs. The community has therefore devised a set of measures to attempt to rank models more adequately.

Before going through existing metrics, it is useful to state broad goals, or *desiderata* of metrics concerning generative modelling. As highlighted by O'Bray et al. [2022], (pseudo)-metrics must be endowed with the following properties:

1. **Expressivity**: Given two sets of samples $\mathcal{X}_1$ and $\mathcal{X}_2$, a suitable mea-

sure d should have $\mathrm{d}(\mathcal{X}_1, \mathcal{X}_2)$ increasing monotonically as $\mathcal{X}_1$ and $\mathcal{X}_2$ become more and more dissimilar.

2. **Robustness**: $\mathrm{d}(\mathcal{X}_1, \mathcal{X}_2)$ should be robust to small perturbations in either sets.
3. **Efficiency**: $\mathrm{d}(\mathcal{X}_1, \mathcal{X}_2)$ should be fast to calculate should scale well with size and number of graphs.

For images, an interesting metric (and the current standard for that domain) is the Fréchet Inception Score, as introduced by Heusel et al. [2017]. Overall, the goal of this metric is to calculate some distance between the real-world images and the synthetic images using the activations of a neural network normally used for classification tasks. Concretely, this is achieved by calculating the squared Wasserstein metric between the generated and real representations computed from a convolutional network (commonly, the Inception v3 architecture from Szegedy et al. [2016] is used) as two multidimensional Gaussian distributions with parameters $\mathcal{N}(\mu, \Sigma)$ and $\mathcal{N}(\mu_{rw}, \Sigma_{rw})$, respectively. The general formulation of the p^{th} Wasserstein distance between two distributions u and v is given by

$$W_p(u, v) := \left(\inf_{\gamma \in \Gamma(u,v)} \int_{M \times M} d(x, y)^p \, \mathrm{d}\gamma(x, y) \right)^{1/p}, \tag{2.3}$$

where (M, d) is a metric space, $\Gamma(u, v)$ denotes the collection of all measures on $M \times M$ with marginals u and v on the first and second factors, respectively. Intuitively, $W_p(u, v)$ can be interpreted as a generalization of the Minkowski distance between probability distributions instead of fixed-length vectors, the latter being given by:

$$d\left(X, Y\right) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}. \tag{2.4}$$

In the case of the Fréchet Inception Score, the squared Wasserstein distance between the Inception v3-derived representations of the images can be reformulated as follows:

$$\mathrm{FID} = ||\mu - \mu_{rw}||_2^2 + \mathrm{tr}(\Sigma + \Sigma_{rw} - 2(\Sigma^{1/2}\Sigma_{rw}\Sigma^{1/2})^{1/2}). \tag{2.5}$$

For the graph domain, such a measure is infeasible, due to the lack of a common consensus on embedding Xu [2021]. Interesting strides have been made in some domains, such as in the drug discovery field, where the penultimate layer of the ChemNet neural network can be used as input to the FID as shown by Preuer et al. [2018].

However, an interesting approach recently explored by Thompson et al. [2022] leverages the observation, made in part by Xu et al. [2018a], Morris et al. [2019], and Kipf and Welling [2016], that certain Graph Neural Networks (GNNs) have the ability to extract meaningful representations without any training. Through a set of two perturbation experiments, similar to the work done by Xu et al. [2018b] and O'Bray et al. [2022], Thompson et al. [2022] show that using randomly initialized Graph Isormorphism Networks (GINs), first introduced by Xu et al. [2018a], provides a strong, domain-agnostic metric to evaluate generative GNNs. GINs – like the majority of GNNs – consist of (i) L propagation layers that perform some form of message passing between the nodes aiming to convey information within a region of a graph, computing rich representations of each node's neighbourhoods in the process, and (ii) some readout layer, aiming to compute some embedding and subsequent output. For GINs, the message passing layers computing each (hidden) node embedding v at layer l (denoted $\mathbf{h}_v^{(l)}$) is assigned the following value:

$$\mathbf{h}_v^{(l)} := \mathrm{MLP}^{(l)}\left(\mathbf{h}_v^{(l-1)} + f^{(l)}\left(\left\{\mathbf{h}_v^{(l-1)} : u \in N(v)\right\}\right)\right), \tag{2.6}$$

$\forall v \in V$ where V is as defined in section 2.2, $\forall\ l > 0, \mathbf{h}_v^{(l)} \in \mathbb{R}^d$, $\mathrm{MLP}^{(l)}$ is a multilayer perceptron, and $f^{(l)}$ is some aggregating function, such as the mean, max or sum. The second part, i.e. the graph readout layer with skip connections, aggregates features from all nodes at each layer $l \in [1, L]$, concatenating them into one (L^d) dimensional vector x_i as follows:

$$\mathbf{x}_i = \mathrm{CONCAT}\left(g\left(\left\{\mathbf{h}_v^{(l)} \mid v \in V\right\}\right) \mid l \in [1, L]\right) \tag{2.7}$$

where g can be chosen from the same set of functions as $f^{(l)}$.

While these developments are encouraging, practitioners designing generative GNNs such as Liao et al. [2019], Niu et al. [2020], and You et al. [2018] have generally gravitated towards the MMD measure to evaluate the quality of the graph, most likely due to the fact that it provides a solid statistical framework that can yield solid statistically significant evidence that two sample distributions are issued from the same distribution.

2.6 Maximum Mean Discrepancy

A significant part of this thesis is centered around investigating the MMD statistic, so we define and examine existing MMD research here. Introduced by Borgwardt et al. [2006] and further exposited by Gretton et al. [2012], it leverages the expressive power and versatility of *kernel functions* to evaluate a distance function between two sample distributions. Moreover, Gretton et al. [2012] describe how this distance function can be treated as a statistic, from which various tests can be derived to evaluate whether or not two distributions are equivalent. MMD is therefore an ideal platform to leverage when trying to assess generative models.

Let us now reformulate the problem described above more formally following the notation from Gretton et al. [2012]. Let x and y be random variables defined on a topological space $\mathcal{X}$ with respective Borel probability measures p and q. Given observations $X = \{x_1, \ldots, x_n\} \subseteq \mathcal{X}$ and $Y = \{y_1, \ldots, y_m\} \subseteq \mathcal{X}$ i.i.d. sampled from p and q, respectively, can we decide whether $p \neq q$?

Gretton et al. [2012] observe in Lemma 1 that:

$$p = q \iff \mathbf{E}_x(f(x)) = \mathbf{E}_y(f(y)) \; \forall f \in C(\mathcal{X}) \tag{2.8}$$

where $C(\mathcal{X})$ is the space of bounded continuous functions on $\mathcal{X}$. Critically, Gretton et al. [2012] observe that even though $C(\mathcal{X})$ is able to identify $p = q$ uniquely, such a function class is not practical to work with. They therefore define a more general class of statistic $\mathcal{F}$ to measure the disparity between q and p to be $f : \mathcal{X} \to \mathbb{R}, f \in \mathcal{F}$. From there, they defined the MMD as:

$$\mathrm{MMD}[\mathcal{F}, p, q] := \sup_{f \in \mathcal{F}} (\mathbf{E}_x(f(x)) - \mathbf{E}_y(f(y))). \tag{2.9}$$

Given n samples from X and m samples from Y, the biased empirical estimate of the MMD is given by:

$$\mathrm{MMD}_b[\mathcal{F}, X, Y] := \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^{m} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} f(y_i) \right) \tag{2.10}$$

Gretton et al. [2012] go on to prove in Section 2.2 that *kernel functions* be used as one of the possible function classes $\mathcal{F}$. Let a kernel function $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ satisfying the following properties:

- $\mathrm{k}(x_i, x_j) = \mathrm{k}(x_j, x_i) \; \forall \; x_i, x_j \in \mathcal{X}$
- $\sum_{i,j} c_i, c_j \mathrm{k}(x_i, x_j) \geq 0 \; \forall \; x_i, x_j \in \mathcal{X}, \forall \; c_i, c_j \in \mathbb{R}$.

At last, we then arrive at the biased empirical estimate of MMD using kernel functions, which is given by:

$$\mathrm{MMD}^2(X, Y) := \frac{1}{m^2} \sum_{i,j=1}^{m} \mathrm{k}(x_i, x_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} \mathrm{k}(y_i, y_j) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathrm{k}(x_i, y_j) \tag{2.11}$$

In accordance with Lemma 6 of Gretton et al. [2012], the diagonal elements of the first two kernel matrices in Equation 2.11 can be removed to obtain an unbiased estimate of MMD, which we will use throughout this thesis, and can be formulated as follows:

$$\mathrm{MMD}^2(X,Y) := \frac{1}{m(m-1)} \sum_{i,j=1}^{m} \mathrm{k}(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i,j=1}^{n} \mathrm{k}(y_i, y_j) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathrm{k}(x_i, y_j) \tag{2.12}$$

While one of the advantages of kernels is that they are able to operate *directly* on a variety of structured data (see Section 2.7 for further details), it can be convenient to extract intermediate representations of the data using *descriptor functions* $g : \mathcal{X} \to \mathbb{R}^d$ prior to being used in a kernel. Descriptor functions, which we will investigate in more detail below, can be designed in such a way that they can distill relevant properties of the structured data under study, potentially saving on computation costs. Similarly, the kernel choice and parameters also heavily impacts how each dataset is being analyzed. Importantly, one consequence of the free choice of descriptor functions and kernel functions is that MMD does not inherently have a scale [O'Bray et al., 2022]

In the graph domain, it is incumbent upon the practitioner to (i) choose an appropriate (optional) graph descriptor and (ii) kernel with (iii) appropriate kernel hyperparameters. This process, along with its pitfalls and current practices are discussed in more detail by O'Bray et al. [2022], but we want to give an overview of possible, common, and sensible choices for descriptor, kernel, and hyperparameter here.

A common practice in the literature is to first extract some fixed-length graph representation using a range of commonly used descriptors such as:

- **The degree histogram.** Given a graph $G = (E, V)$ as defined in Section 2.2, we can calculate $\deg(v), \forall v \in V$, where position i of the resulting histogram is the number of vertices with degree i. With a given maximum degree d, we obtain a mapping $f : G \mapsto \mathbb{R}^d$. We will normalize the entries of the histogram to obtain a density histogram (i.e. all the entries add up to one), which then becomes a size-invariant descriptor.
- **The clustering coefficient histogram.** The clustering coefficient of a vertex v is defined as the fraction of edges within its neighborhood divided by all possible edges between neighbors, i.e.

$$C(v) := \frac{2\left|\left\{(v_i, v_j) \in E \mid v_i \in N(v) \vee v_j \in N(v)\right\}\right|}{\deg(v)(\deg(v)-1)}. \tag{2.13}$$

$C(v) \in [0,1]$ measures the extent to which each vertex v forms a clique [Watts and Strogatz, 1998]. The collection of coefficients can be captured for each graph in a histogram, which is also normalized.

- **The Laplacian spectrum histogram.** The normalized graph Laplacian is given by $\mathcal{L} := I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ where A is the adjacency matrix (see Section 2.2), I the identity matrix and D the degree matrix, where $D_{ii} = \deg(v_i)$ and $D_{ij} = 0, i \neq j$. Since A is symmetric (all graphs in this thesis are undirected), and that $\mathcal{L}$ is real-valued, it is also diagnonalizable, with eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \forall \lambda \in [0,2]$ – see Chung [1997], 1997, Chapter 1, Lemma 1.7 for a proof of the boundedness. This lends itself again to some bounded, normalized histogram representation. As discussed by O'Bray et al. [2022], it is unknown if graphs can be fully determined by their spectrum, and we know that for certain classes of graphs this is not the case [Schwenk, 1973].

It is worth nothing that none of those three descriptors take node labels into account. This is why we are going to investigate alternative kernels leveraging node labels (see Section 2.7), as well as use fixed-length vectors derived from powerful transformer-based protein language models, specifically from the Evolutionary Scale Modeling (ESM) family [Rives et al., 2021]. This family of models allows us to obtain an embedding vector $h \in \mathbb{R}^d$ for each residue, which we can then average to obtain one protein-level embedding by taking the average across residues. Because transformers don't constitute an essential part of this thesis, we redirect the reader to the original publication describing it by Vaswani et al. [2017], as well as the excellent explainer by Alammar [2018] for a discussion on the foundations of the topic.

2.7 Kernels

Kernels are a class of functions computing the similarity between structured data in any Reproducing Kernel Hilbert Space (RKHS). Once graph representations are computed, it is possible to compute a kernel between any two such vectorized representations using kernels. In this thesis, we will use:

- **The linear kernel**. Let $\mathbf{x}, \mathbf{y} \subseteq \mathcal{X} \in \mathbb{R}^d$, where d denotes the dimensionality of the graph descriptor (e.g. the number of bins in the clustering histogram). Then, the linear kernel is defined as:

$$\mathrm{k}(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y} + c \tag{2.14}$$

  with $c \in \mathbb{R}$.

- **The Gaussian kernel**, which is given by:

$$\mathrm{k}(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}\right) \tag{2.15}$$

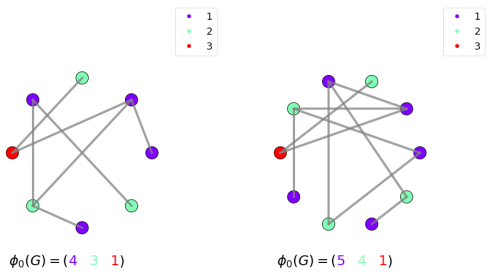
We will neglect certain kernels used in the literature, either because they are not positive semi-definite, such as the total variation kernel (see O'Bray et al. [2022], Appendix A1 for a proof), or because they capture little more information compared to existing accepted alternatives, or because they are inefficient to compute and therefore not recommended to evaluate generative models, e.g. the Earth mover's distance-based kernel [O'Bray et al., 2022].

We will, however, leverage other classes of kernels that are applicable to models evaluating protein generative model performance but were previously unused in the literature. The first are graph kernels (reviewed by Borgwardt et al. [2020]). In particular, we will examine an efficient and expressive kernel used for biological data: the Weisfeiler-Lehman kernel. The second class of kernels leveraged here operate directly on the persistence diagrams obtained using the filtration procedures described in section 2.3. Specifically, we will use the persistence Fisher kernel (PFK) introduced by Le and Yamada [2018] and the multi-scale kernel (MSK) introduced by [Reininghaus et al., 2015]. We define both in the next paragraphs.

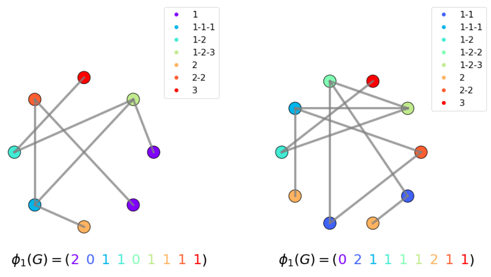
The Weisfeiler-Lehman algorithm Originally designed as a graph isomorphism test by Weisfeiler and Lehman [1968], the eponymous algorithm provides a powerful and computationally efficient way of capturing local node neighborhood information to quantify the degree of similarity between any two graphs, which works for both labeled and unlabeled graphs. The procedure can be described as follows:

1. If the nodes of the graph do not already have node label assigned to them, assign them one, e.g. using their degree.
2. For each node, fetch the node label of each neighbouring node and sort the labels including the node's own label in ascending order. Concatenate the resulting node labels into a string.
3. For each node, compress the string representation of the node label using a hash function, i.e. only returning the same hash if the inputs are the same.
4. For each node, assign the result of the hash function to the label of the node.

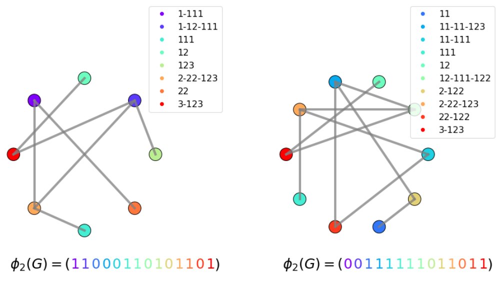
This process can be repeated multiple times to integrate information for farther neighbourhoods. In this thesis, the amino acid type will be used as



(a) First iteration of the Weisfeiler-Lehman algorithm.


1
1-1-1
1-2
1-2-3
2
2-2
3
$\phi_1(G) = (2\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1)$
1-1
1-1-1
1-2
1-2-2
1-2-3
2
2-2
3
$\phi_1(G) = (0\ 2\ 1\ 1\ 1\ 1\ 2\ 1\ 1)$

(b) Second iteration of the Weisfeiler-Lehman algorithm.



(c) Third iteration of the Weisfeiler-Lehman algorithm.

Figure 2.5: Three iterations of the Weisfeiler-Lehman node relabelling algorithm. The detailed explanation of each step of the algorithm are provided in the main text. The kernel between the two graphs can be computed by computing $k_{WL}(G, G') = \phi_2(G) \cdot \phi_2(G')$. Imagery adapted from Mengin [2019].

a node label. An example of a Weisfeiler-Lehman procedure can be seen in Figure 2.5. A kernel can be computed between two graphs by computing the dot product of two resulting hash histograms obtained at the end of the Weisfeiler-Lehman algorithm [Shervashidze et al., 2011].

Persistence Fisher kernel Kernels between persistence diagrams also offer powerful similarity measures between the global shape of proteins using the persistence diagrams produced by the Vietoris-Rips filtration, thereby going beyond just looking at the neighborhoods of each node in the case of the Weisfeiler-Lehman procedure described above.

We first look at the Persistence Fisher kernel, introduced by Le and Yamada [2018]. Given two diagrams Dg_i and Dg_j, the Persistence Fisher kernel k_{PF} is given by:

$$k_{\text{PF}}(\text{Dg}_i, \text{Dg}_j) := \exp\left(-t d_{\texttt{FIM}}(\text{Dg}_i, \text{Dg}_j)\right) \tag{2.16}$$

where:

$$d_{\texttt{FIM}}(\text{Dg}_i, \text{Dg}_j) := d_{\mathcal{P}}\left(\rho_{(\text{Dg}_i \cup \text{Dg}_{j\Delta})}, \rho_{(\text{Dg}_j \cup \text{Dg}_{j\Delta})}\right), \tag{2.17}$$

and

$$d_{\mathcal{P}}(\rho_i, \rho_j) = \arccos\left(\int \sqrt{\rho_i(x)\rho_j(x)}\mathrm{d}x\right). \tag{2.18}$$

is defined as the Fisher Information Metric, with ρ_i and ρ_j being two persistence diagrams which can be represented as points in a probability simplex $\mathbb{P} := \{\rho | \int \rho(x)\underset{\cdot}{x} = 1, \rho(x) \geq 0\}$. We can consider persistence diagrams as points by setting:

$$\rho_{\mathrm{Dg}} := \left[\frac{1}{Z}\sum_{u\in\mathrm{Dg}} \mathcal{N}(x; u, \sigma I)\right]_{x\in\Theta} \tag{2.19}$$

where $Z = \int_{\Theta} \sum_{u\in\mathrm{Dg}} \mathcal{N}(x; u, \sigma I)\mathrm{d}x$, $\mathcal{N}$ is a Gaussian distribution, I the identity matrix, and $\sigma > 0$ is the smoothing parameter is chosen by the practitioner. Note that if the set Θ is set to the Euclidean space, as will be the case in this thesis, each persistence diagram then turns to a probability distribution, which is what allows us to compute the Fisher information metric [Anirudh et al., 2016, Adams et al., 2017].

Multi-scale kernel .

In this thesis, we will also use the MSK introduced by Reininghaus et al. [2015]. Using the same notation as above, they define the MSK k_{MS} as:

$$k_{\mathrm{MS}}(\mathrm{Dg}_i, \mathrm{Dg}_j) = \frac{1}{8\pi\sigma} \sum_{\substack{p\in\mathrm{Dg}_i \\ q\in\mathrm{Dg}_j}} e^{-\frac{\|p-q\|^2}{8\sigma}} - e^{-\frac{\|p-\bar{q}\|^2}{8\sigma}} \tag{2.20}$$

where σ is specified by the user and $\bar{q}$ is a point on the persistence diagram mirrored at the diagonal, i.e. if $q = (b, d)$, then $\bar{q} = (d, b)$ where b is the birth of the topological feature d is its corresponding death.

2.8 Summary

In this section, we introduced the fundamental characteristics of proteins by first discussing how amino acids form a backbone and each of them forms two dihedral angles with adjacent amino acids to make up the three-dimensional structure of the protein. We then showed how one can represent proteins using graphs, either by using k-nn graphs or ε-graphs. We explored an alternative representation strategy using topological data analysis and discussed how it allows one to capture the global structural features of the protein.

We then moved on to introduce generative models, and discussed recent advances in such models in the image domain, where such models were first developed. We then proceeded to discuss generative models in the graph domain, along with the unique computational challenges that it incurs from a modeling standpoint. Importantly, we outlined the evaluation problem arising when evaluating generative networks, specifically in the graph domain, and highlighted the desiderata for good metrics: expressivity, robustness, and efficiency. We examined currently accepted practices and introduced MMD, the main method used in this thesis. We finally introduced the collection of kernels that we are going to leverage when using the MMD.

Chapter 3

Methods

The primary methodology employed in this thesis to assess the quality of metrics used to evaluate generative protein models is heavily inspired by O'Bray et al. [2022], and consists in evaluating how well a particular combination of representation, optional descriptor function, kernel and parameters applied to all three aforementioned elements[1] correlates with the amount of perturbation applied to one set of proteins. This can be broken down in the following steps:

1. Take two i.i.d. samples from a database of proteins.
2. Progressively perturb to one of the samples.
3. Measure the MMD between the unperturbed and perturbed sample.
4. Once the varying degrees of perturbations have been applied and accompanying MMDs catalogued, compute the correlation coefficient between the MMD and the amount of perturbation.

A particular MMD configuration is considered superior to another if the resulting correlation coefficients of the former are higher than that of the latter.

In this chapter, we start by motivating the datasets that will be used to simulate a generative protein model. We then describe and motivate the experimental setups – i.e. perturbations – that we will employ to test the various configurations of MMD. Finally, we enumerate and justify which configurations of MMD are tested, including all the combinations of protein representations, descriptor functions, and kernels. Finally, we describe and motivate the experimental setups – i.e. perturbations – that we will employ to test the various metric configurations.

[1]We also subsequently refer to the parametrization and choice of all these options as an MMD *configuration*.

3.1 Datasets

In this thesis, except otherwise stated, all results will be derived from 10 random samples from the *homo sapiens* monomeric proteome downloaded from the EBI AlphaFold2 database [Varadi et al., 2022, Tunyasuvunakool et al., 2021], a repository comprising predicted 3D structures of protein sequences obtained from AlphaFold2, the current state-of-the-art method to predict protein structure from sequences [Jumper et al., 2021]. Multiple reasons justify this choice. First, despite being a proxy for experimentally validated proteins, AlphaFold2 is known to provide predicted 3D structures for naturally occurring proteins with the same accuracy as experimentally acquired 3D structures [Jumper et al., 2021], ensuring that the conclusions we reach in this thesis will be broadly applicable to experimentally acquired protein structures. Second, this allows us to establish the ground truth of a range of MMD configurations, hence also enabling the gauging the quality of Third, there are practical advantages related to this database because it contains consistently formatted `pdb` files exclusively cataloguing heavy atoms directly contributing to the 3D structure of a single monomer, which simplifies downstream processing.

3.2 Perturbations

While O'Bray et al. [2022] focused on *graph perturbations* specifically, we wanted to augment and refine the set of perturbations applied to the perturbed sample of proteins to be more pertinent to proteins. Three categories of perturbations can be distinguished:

Graph Perturbations These perturbations mostly overlap with those defined by O'Bray et al. [2022], as they include (i) adding edges to a graph (ii) removing edges from a graph, and (iii) rewiring, i.e. swapping, edges within a graph.

Point Cloud Perturbations These perturbations aim to add changes to the underlying coordinates of each atom in the protein. Such perturbations include injecting Gaussian noise (Equation 3.1), (ii) twisting, (iii) shearing, and (iv) tapering. Importantly, where appropriate, we extract the graphs *after* applying the point cloud perturbation, to ensure the changes in the point cloud can be reflected in the graph structure. We proceed to detail each of the equations governing the perturbations below. An illustration of each of those perturbations can be found in Figure 3.1

In the notation that follows, x, y z represent the unperturbed coordinates, and x', y', and z' represent the perturbed coordinates. The *Gaussian noise* added to a coordinate system is given by:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x + \text{Noise} \\ y + \text{Noise} \\ z + \text{Noise} \end{bmatrix} \tag{3.1}$$

where Noise $\sim \mathcal{N}(0, \sigma)$ and σ is set by the user. *Twisting* is achieved by adding the following transformation to the coordinate system:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \cdot \cos(\alpha \cdot z) - y \cdot \sin(\alpha \cdot z) \\ x \cdot \sin(\alpha \cdot z) + y \cdot \sin(\alpha \cdot z) \\ z \end{bmatrix} \tag{3.2}$$

where $\alpha \in \mathbb{R}$ is in rad $\cdot$ Å^{-1} is set by the user. *Shearing* the coordinate system is achieved by applying

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a \cdot z + x \\ b \cdot z + y \\ z \end{bmatrix} \tag{3.3}$$

to the coordinate system, where $a, b \in \mathbb{R}$ are in Å and set by the user. In this thesis, we set $a = b$. Similarly, *tapering* is achieved by applying

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} (0.5 \cdot a^2 \cdot z + b \cdot z + 1) \cdot x \\ (0.5 \cdot a^2 \cdot z + b \cdot z + 1) \cdot y \\ z \end{bmatrix} \tag{3.4}$$

where $a, b \in \mathbb{R}$ are in Å and set by the user. Similarly to Equation 3.3, we set $a = b$ in this thesis.

Mutation These simply consist in (i) selecting the positions that will be mutated by sampling from a Bernoulli distribution with parameter p and (ii) for selected positions, swap the amino acid by any of the 20 possible naturally-occurring amino acids. While graph perturbation probabilities (e.g. of adding an edge) range from 0 to 1, here we mostly concentrate on lower regimes of mutation, i.e. between 0 and 0.1, so see how sensitive various MMD configurations are to a few point mutations, which covers most of the real-world protein engineering use cases [Poluri and Gulati, 2017].

For each perturbation, a range of degrees of perturbation is defined and 20 different evenly-spaced degrees of perturbation are examined and repeated 10 times with 10 pairs of i.i.d. proteins to estimate the sensitivity of the particular MMD configuration to the perturbation. The ranges used for each parameter used in this thesis are shown in Table 3.1

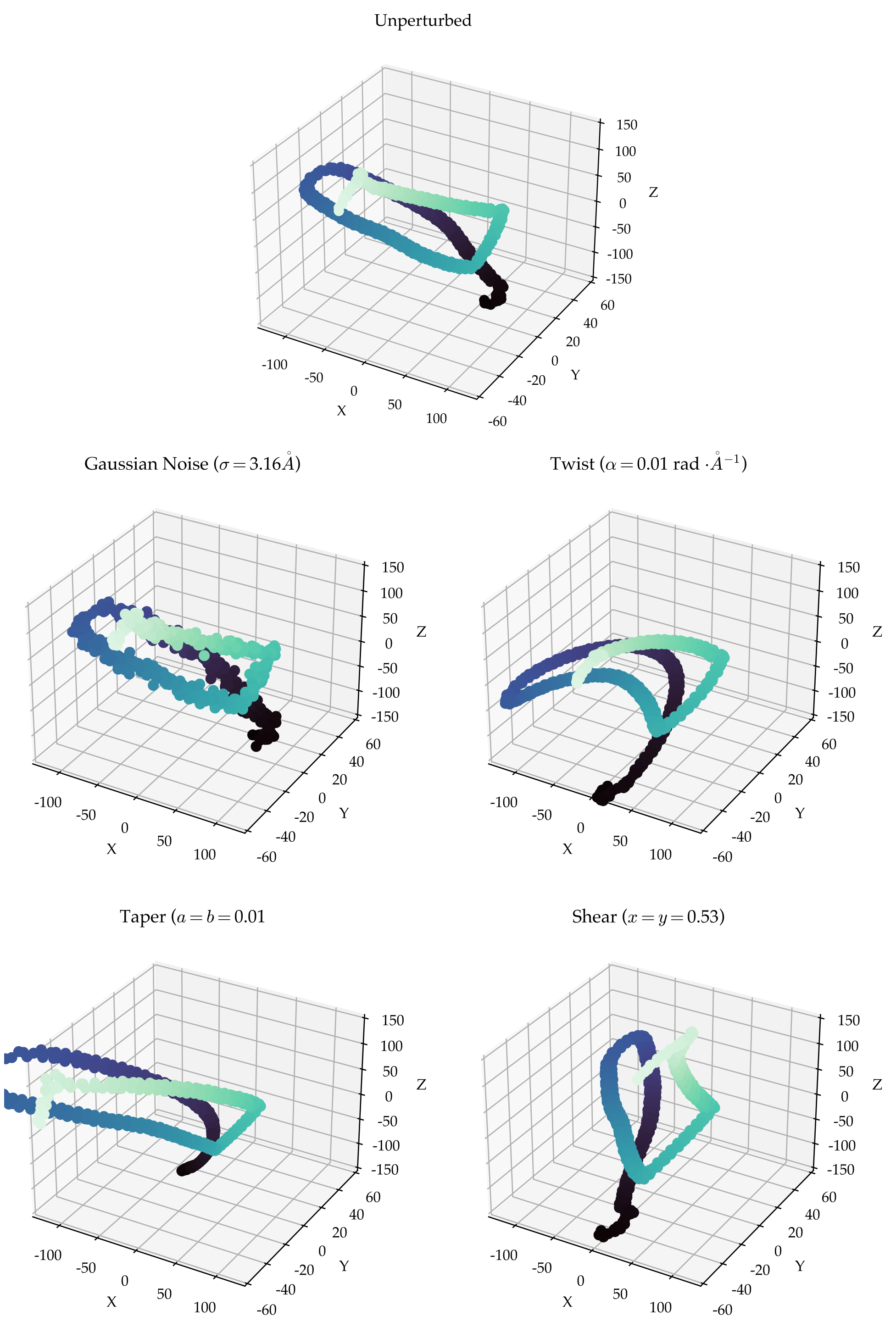


Figure 3.1: Illustration of all point cloud perturbations applied to a protein. The amount of each perturbation shown here corresponds to 10% of the maximum value of the maximum value of each perturbation as shown in Table 3.1. The protein displayed here is the cilia- and flagella-associated protein 53 (UniProt code Q96M91). Each α carbon is colored according to its position on the chain. On the unperturbed protein, the lighter colors are located closer to the viewer.

Perturbation	**Range**
Twist	[0, 0.1] (rad/Å)
Shear	[0, 5] (Å)
Taper	[0, 0.1] (Å, Å)
Gaussian Noise	[0, 30] (Å)
Mutation	[0, 0.1]
Graph Perturbations (remove, add, rewire edges)	[0, 1]

Table 3.1: Perturbation ranges used in this thesis. Each interval was split into 20 evenly-spaced degrees of perturbation.

3.3 MMD Configurations

We introduced MMD in Section 2.6 (Equation 2.11 and 2.12), and noted that an important aspect of using MMD in practice consists in choosing the right descriptor function and kernel, because it heavily impacts which aspects of the data are distilled for analysis in MMD and also heavily impacts how the data is processed by the kernel functions inside of MMD. Here, we list and motivate all graph extraction techniques, descriptor functions employed, and kernels adopted in this experiment.

First, throughout our experiments, we will use the unbiased squared MMD estimate, which removes self-comparison terms in the kernel matrices (Gretton et al. [2012], Lemma 6), which can result in negative values. We will normalize the resulting MMD over the whole range of perturbation for each curve as well to compare behaviors of different MMD configurations not operating on the same scale. As highlighted in Section 2.6, MMD does not have an inherent sense of scale anyway, so this normalization step does not impact our analysis negatively.

$$\mathrm{MMD}_{\mathrm{Normalized}} = \frac{\mathrm{MMD} - \min(\mathrm{MMD}_{\mathrm{Experiment}})}{\max(\mathrm{MMD}_{\mathrm{Experiment}}) - \min(\mathrm{MMD}_{\mathrm{Experiment}})} \tag{3.5}$$

Where $\mathrm{MMD}_{\mathrm{Experiment}}$ is the collection of MMD values for a particular experiment, i.e. a particular MMD configuration tracked through the whole range of a particular perturbation.

3.3.1 Representations

We will use several different representations of proteins in this thesis, which can be grouped into three categories. The first is *coordinates*. These are parsed from each `.pdb` file. The second is *graphs*. They include k-NN and

Graph type	Values
k-NN graphs	$k \in \{2, 6, 8\}$
ε-graphs	$\varepsilon \in \{8, 16, 32\}$

Table 3.2: Ranges of parameters used to extract graphs from point clouds in this thesis.

ε-graphs, introduced in Section 2.2. A summary table of the different types of graphs extracted from proteins in this thesis can be found in Table 3.2. The third is the simple protein *sequence*. Each protein's sequence parsed from each `.pdb` file as is. Since we are only dealing with monomers, no additional processing is required.

3.3.2 Descriptor Functions

As discussed in Section 2.6, some kernels require some alternative vectorized representation to work. We will use the following protein descriptor functions here.

Graph descriptors They are defined in Section 2.6 and include the degree distribution histogram, the clustering coefficient histogram and the Laplacian spectrum histogram. These are all fixed-length vectors. The maximum value of the degree histogram was determined based on the longest sequence length of the proteins in the dataset (2699), because the node degree of any given graph will be at most equal to the size of the largest graph in the dataset. This aspect is particularly relevant when considering the perturbation adding edges to a protein graph.

Coordinates descriptors In order to capture the information of the 3D structure of the protein beyond local neighborhood information, a topological descriptor of each protein in the form of a persistence diagram is extracted using a Vietoris-Rips filtration introduced in detail in Section 2.3. To speed up computation (and since we do not take amino acid type into account for this analysis), we sampled every other point to dramatically reduce the running time and memory footprint without significantly affecting the shape of the protein.

Protein-specific descriptors In this thesis we introduce two novel protein-specific descriptor functions resulting in fixed-length vector representations for subsequent use in kernels accepting inputs in $\mathbb{R}^d$. The first consists of a histogram of the pairwise distance of each α-carbon[2]; the second consists

[2]In principle, an all-atom histogram could also be computed; we just anticipate that the added value of such an all-atom histogram would be minimal in comparison to the significantly increase computation cost and memory footprint of the resulting histogram.

Descriptor Name	Number of Bins	Range of Bins
Degree Histogram	2699	[0, 2699]
Laplacian Spectrum Histogram	100	[0, 2]
Distance Histogram	1000	[0, 1000] (Å)
Dihedral Angles Histogram	100	$[-\pi, +\pi]$ (rad)

Table 3.3: Descriptor function bin numbers and ranges of descriptor functions used in this thesis. The ranges are given by [minimum value, maximum value] and (unit) when applicable.

in concatenating the two histograms of the two dihedral angles ϕ and ψ formed by each amino acid discussed in Section 2.1. The inspiration behind those two descriptors comes from elements of the validation pipeline of novel Protein Data Bank structures [Read et al., 2011, Gore et al., 2012, 2017], where atoms too close together are flagged and unusual dihedral angles are reported to penalize the overall validity of the protein 3D model to be validated. Those unusual dihedral angles are also called "Ramachandran outliers", after the scientist who discovered a way to display the ϕ and ψ in a 2-D histogram, and recovered features in this histogram related to the secondary structure of the protein from such plots. [Ramachandran et al., 1963]. Overall, both the biological relevance and the established scientific success of distance histograms and dihedral angles histogram in both the validation and analysis of proteins lend credence to those newly established descriptors.

Sequence descriptors In Section 2.6, we discussed the Evolutionary Scale Modelling family to construct descriptors of a fixed length using a learned embedding. We use the 6-layer variant trained on the UniRef50 snapshot from March 2018 which contains approximately 43 million parameters, because it is able to process the full range of sequence lengths that we have in our dataset (the longest sequence fragment is 2699 amino acids long).

Table 3.3 summarizes the parameters used to set up the descriptor functions in this thesis.

3.3.3 Kernels

Once a suitable representation and descriptor function is selected, one requires a kernel to evaluate the MMD in the corresponding RKHS. We detail which kernels we are going to evaluate and why in this section. Kernels used in this thesis can be grouped into three separate categories.

The first category has been discussed in the literature extensively since it was used to evaluate generative graph neural network models, i.e. the *fixed-length vector kernels* like the Gaussian (RBF) kernel and the linear kernel, both of which are defined in Section 2.7[3]. Since the only condition for each vector to be valid inputs for those kernels is that they are in $\mathbb{R}^d$, the protein-specific vector representations outlined in Section 3.3.2 are valid inputs.

As alluded to in Section 2.7, we introduce two new classes of kernels for use in MMD which so far have not been used to evaluate generative models due to the unique aspects of proteins that need to be captured. This brings us to the second category of kernels used in this thesis: *graph kernels*. Specifically, we are going to use the Weisfeiler-Lehman kernel discussed in Section 2.7 because it captures *local* patterns in the neighbourhood of each node. In Appendix A.2, we detail how we achieved an 80% improvement in the runtime of the Weisfeiler-Lehman kernel by leveraging the sparsity of the graphs used in this thesis.

To estimate *global* changes in the shape of the protein, we will also use kernels accepting persistence diagrams as input, specifically, we will adopt the PFK [Le and Yamada, 2018] and the MSK [Reininghaus et al., 2015] defined in Section 2.7.

3.4 Experimental Setup

3.4.1 Measuring the Quality of MMD Configurations

To objectively evaluate the various representation, descriptor, and kernel combinations used in MMD, we will use two correlation coefficients, namely the Spearman's correlation coefficient and Pearson's correlation coefficient, each given by:

$$\rho_s(X,Y) = \frac{\operatorname{cov}(\mathrm{RANK}(X), \mathrm{RANK}(Y))}{\sigma_{\mathrm{RANK}(X)}\sigma_{\mathrm{RANK}(Y)}}, \tag{3.6}$$

and

$$\rho_p(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y}, \tag{3.7}$$

respectively. Here, X is the vector containing the set of values used to perturb one of the protein sets and Y the vector of MMD values between the unperturbed and perturbed set for each perturbation level. In this setting, a high Spearman correlation coefficient (Equation 3.6) is crucial to satisfy the first criterium of a performance metric: expressivity (see Section 2.5). This will guarantee that the metric increases monotonically with the amount of perturbation. As O'Bray et al. [2022] have shown, some configurations of MMD on synthetic datasets revealed that an increasing MMD value with increasing perturbation is not guaranteed. While Thompson et al. [2022]

[3]We will follow the speed-up trick outlined in Appendix A.5 by O'Bray et al. [2022] to reduce the computation time of each different bandwidth parameter.

highlighted that linearity is not a requirement, the Pearson correlation coefficient (Equation 3.7) will allow us to further refine the selection of the metrics behaving most predictably, and distill the most relevant configurations for a given perturbation range.

Another important metric we can employ to quantify the quality of a particular MMD configuration is the standard deviation of the various MMD values across runs over the whole range of pertubations. As indicated in Section 2.5, one of the desiderata for a generative model metric is robustness. Since all experiments have been run with the same number of i.i.d. samples every time, using the standard deviation allows us to estimate how much a particular MMD configuration is sensitive to the underlying data used. An MMD configuration with a high standard deviation under a particular perturbation regime would be indicative of low robustness, and would therefore be a less reliable estimate of the quality of the samples compared to an MMD configuration with a high standard deviation. We abbreviate this standard deviation measure as σ_{MMD} to avoid confusion with the σ parameter of the RBF kernel presented in Section 3.3.3. In Chapter 4, wherever we conduct statistical tests, and our highest significance threshold is $5 \cdot 10^{-2}$.

3.4.2 Software Library Design

Due to the complexity and modularity of the methodology explored above, it is required to have a scalable library to execute all the evaluation experiments at scale and efficiently. To accomplish this task, we developed a custom Python library leveraging parallel processing of data to the greatest extent possible. We were inspired by the standards of `scikit-learn` and implemented multiple modules following the same design patterns to ensure that we could build `Pipeline` objects with the necessary steps required to compute an MMD. All the code used for this thesis can be found here: `https://github.com/pjhartout/msc_thesis`. Access can be granted upon request.

3.5 Summary

In this chapter, we detailed the methodological setup employed in this thesis. We first described the datasets we used, as well as the type of perturbations applied to them. We then discussed the representations of the proteins used in this study, as well as described the descriptor functions used for those representations. Crucially, we motivated our choice for the collection of kernels used for estimating the MMD. Finally, we outlined the methodology leveraged to assess the quality of various MMD configurations objectively.

Chapter 4

Results

In this chapter, we will introduce the results of the experiments subjecting protein representations to various relevant perturbation regimes highlighted in chapter 3. We first discuss the results and implications of frequently used MMD configurations, namely by showing how it behaves on protein graph descriptors. We then move on to show the results of the sensitivity of the MMD values depending on the underlying graph representation used to extract the various graphs. Finally, we explore more exotic configurations of MMD that we hypothesize might be more suitable for proteins. We conclude this chapter with a short section on the runtimes of the various elements of the computational pipelines shown in this chapter.

4.1 Overall MMD Behavior

4.1.1 General observations on the correlation coefficients

Surprisingly, we found that the behavior of MMD was not as inconsistent for the types of graphs extracted from proteins as was found on synthetic graphs by O'Bray et al. [2022]. Figure 4.1 show trajectories and correlations of MMD values with different perturbation types using ε-graphs with ε set to 8 Å. Both the Spearman and Pearson correlation coefficients averaged across runs are high. There is, however, an exception: the correlation between the MMD obtained from the degree histogram and the addition of edges is comparatively low with $\rho_P = 0.25$ and $\rho_S = -0.44$ versus that obtained from the Laplacian spectrum histogram (with $\rho_P = 0.95$, $\rho_S = 0.98$), and the clustering histogram (with $\rho_P = 0.95$, $\rho_S = 0.97$). Curiously, the next lowest (Pearson) correlation coefficient from Figure 4.1 is also associated to the degree distribution histogram under the rewiring perturbation regime ($\rho_p = 0.82$ $\rho_S = 0.98$).

4.1.2 General observations on the standard deviations

In parallel, Table 4.1 also supports the fact that the standard deviation of the normalized MMD of the degree distribution descriptor is the highest, suggesting that this is not a very robust descriptor. Since modelling graph connectivity is one of the primary challenges of generative graph models [Li et al., 2018], based on these results we do not recommend using a degree histogram as a descriptor function for MMD. The remaining descriptors do show high correlations ($\rho_P \geq 0.89$, $\rho_S \geq 0.97$) and reasonably low standard deviations (> 0.64), which make them good candidates.

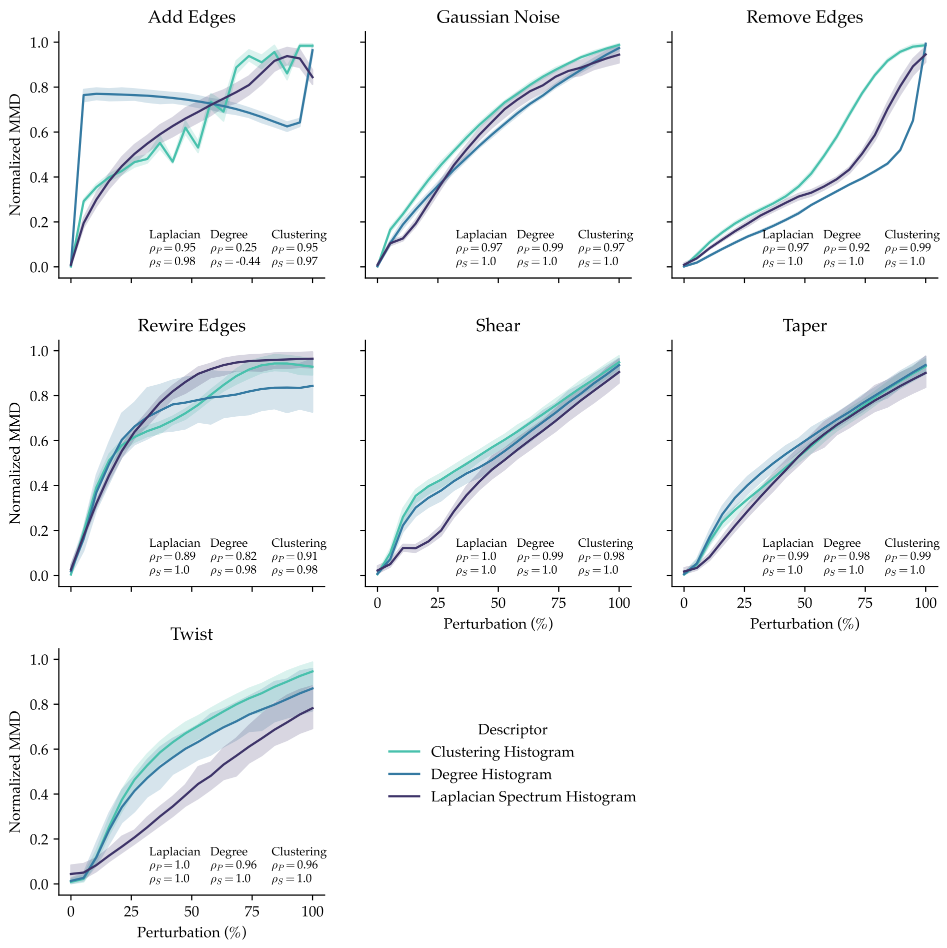


Figure 4.1: MMD vs. perturbation (in % of the maximum values shown in Table 3.1) for various graph descriptors of the 8Å-graphs under different perturbations regimes. The kernel used to obtain these graphs is the RBF kernel with bandwidth 0.01. ρ_S: average Spearman correlation coefficient across runs. ρ_P: average Pearson correlation coefficient across runs. Except for the degree histogram behaviour when edges are added, we see that most MMD configurations behave well, i.e. there is a high correlation betweeen the MMD values and the perturbation.

Perturbation Type	**Descriptor**	σ_{MMD}
Add Edges	Clustering Histogram	0.023
	Degree Histogram	0.024943
	Laplacian Spectrum Histogram	0.039
Gaussian Noise	Clustering Histogram	0.011
	Degree Histogram	0.012793
	Laplacian Spectrum Histogram	0.027
Remove Edges	Clustering Histogram	0.009
	Degree Histogram	0.004400
	Laplacian Spectrum Histogram	0.018
Rewire Edges	Clustering Histogram	0.029
	Degree Histogram	**0.104**
	Laplacian Spectrum Histogram	0.032
Shear	Clustering Histogram	0.031
	Degree Histogram	0.041
	Laplacian Spectrum Histogram	0.0410
Taper	Clustering Histogram	0.026
	Degree Histogram	0.035
	Laplacian Spectrum Histogram	0.043
Twist	Clustering Histogram	0.053
	Degree Histogram	0.081
	Laplacian Spectrum Histogram	0.064

Table 4.1: Average standard deviation of the various MMD configurations shown in Figure 4.1 under the same perturbation types. The highest standard deviation is observed for the degree histogram descriptor under rewiring perturbations.

4.1.3 Influence of the choice of kernel

We next investigate the overall influence of the kernel on the correlations between perturbations and MMD shown in Figure 4.2. We can see that for $\sigma \lessapprox 0.1$ (σ being the hyperparameter of the Gaussian kernel, see Section 2.7) and for the linear kernel, MMD values behave as desired, i.e. $\rho_P, \rho_S \geq 0.8$ and, if one excludes the Laplacian spectrum histogram at $\sigma = 0.1$, we have $\rho_P, \rho_S > 0.95$. However, correlation coefficients drop sharply when increasing $\sigma > 0.1$, most likely due to oversmoothing, which is a phenomenon arising when the bandwidth of the kernel is large enough to obscure any structure in the data [Hwang et al., 1994]. This can have unpredictable consequences on resulting MMD values: in the case of the degree histogram or the clustering histogram, this results in an overly sensitive kernel sharply increasing in value at the slightest perturbation, while the clustering histogram remains oblivious to large amounts of perturbation. This can be explained by the relative scale of each of the embeddings and their pairwise distances, which we catalogue in Appendix A.3.

4.2 Influence of the Graph Representation on MMD

4.2.1 Comparing Graph Construction Technique

To compare which graph construction technique was overall most adviseable, we computed the correlation coefficients of the various available combinations of graph type, graph extraction parameter, and description function with an RBF kernel with $\sigma = 0.01$, which was shown to behave reasonably stably across descriptors (Figure 4.2). We then compared the distributions of the two correlation coefficients and computed a Mann-Whitney $\mathcal{U}$ test [Fay and Proschan, 2010]. Figure 4.3 shows the distributions of both ρ_S and ρ_P for the k-NN and ε-graphs. Both the test for the Pearson correlation coefficient and the Spearman correlation coefficient were significant ($p = 8.35 \cdot 10^{-4}$ and $p = 1.765 \cdot 10^{-2}$, respectively) indicating that one distribution is stochastically greater than the other. In both cases, the distribution of the correlation coefficients from the ε-graphs is higher, as we can see in the legend of Figure 4.3. This result is intuitive, because ε-graphs are likely more sensitive to the underlying topology of the protein. We therefore proceed with ε-graphs in the subsequent results discussed below.

4.2.2 Lower Values of ε Are More Stable

In the previous paragraph, we established that ε-graphs were more appropriate to compute the MMD. Now, we investigate which specific threshold(s) ε are most appropriate using the same meta-metrics as before. Figure 4.4 shows the normalized MMD values with the varying degree of different

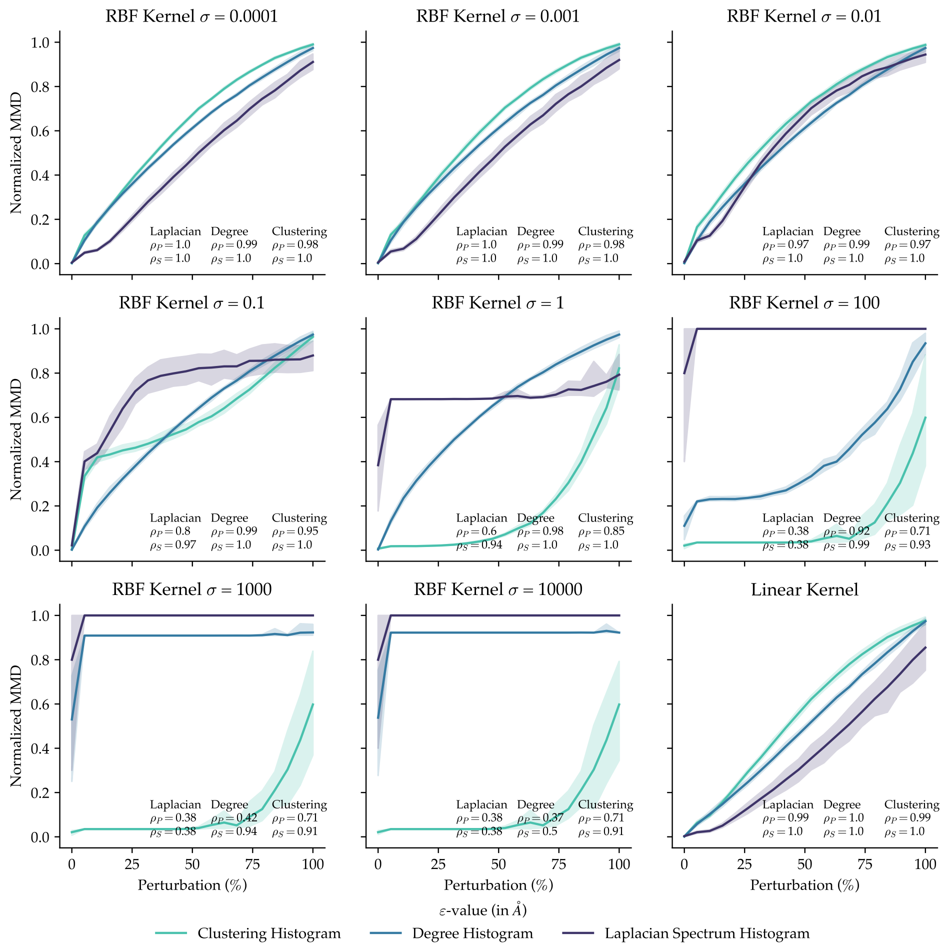


Figure 4.2: MMD vs. Gaussian noise perturbation (in % of the maximum values shown in Table 3.1) for various graph descriptors of the 8Å-graphs. ρ_S: average Spearman correlation coefficient across runs. ρ_P: average Pearson correlation coefficient across runs. The kernel here is shown on top of each subplots. We can see that reasonable behaviour of the RBF kernel can be seen when $\sigma < 1$. The linear kernel also behaves well.

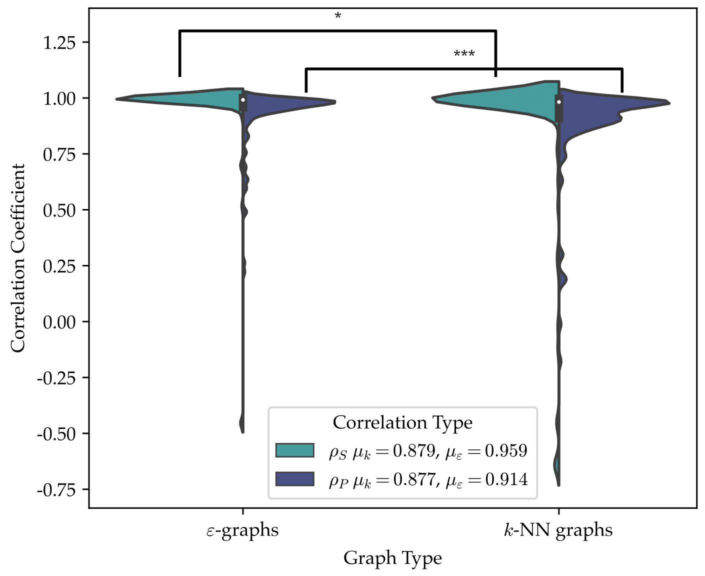


Figure 4.3: Violin plot of the distributions of the correlation coefficients of various MMD configurations derived from the two different graph construction methods. Each distribution contains various combinations of descriptor functions, perturbation types, and various values of the parameter used to extract the graphs (i.e. k in the case of k-NN graphs and ε in the case of ε-graphs). The distributions are then compared using a Mann-Whitney U test, yielding significant p-values for both ρ_S and ρ_P ($p = 8.35 \cdot 10^{-4}$ and $p = 1.765 \cdot 10^{-2}$, respectively), indicating that the correlation coefficients obtained from k-NN graphs are statisitcally significantly worse than those obtained from ε-graphs.

types of perturbations with various graph descriptors. While most configurations exhibit high correlations: excluding seven outliers of the 72 coefficients calculated, we have $\rho_P > 0.97$ and $\rho_S > 0.98$. In Figure 4.4, we can also see that lower coefficients tend to be reached when using a high threshold ε. For instance, in the case of the twisting perturbation and using the clustering histogram as a graph descriptor, we see $\rho_P = 0.73$ and $\rho_S = 0.71$ for 32 Å-graphs vs $\rho_P = 0.96$ and $\rho_S = 1.0$ for both 16 and 8 Å graphs. While this is the most extreme example, we also see a similar pattern when using a different descriptor, such as the Laplacian spectrum histogram, where we have $\rho_P = \rho_S = 0.93$ for 32Å graphs vs $\rho_P = 0.99 \rho_S = 1.0$ for 16Å graphs and $\rho_P = 1.0 \rho_S = 1.0$ for 8-Å graphs. While there are some exceptions to this pattern, the differences are not nearly as substantial (the highest difference where the correlation coefficient for the 32-Å graph is higher than the other two ε thresholds is 0.03, see lower mid pane of Figure 4.4).

The finding that increasing ε decreases the overall quality of MMD is further supported by the changes in standard deviation of the different runs averaged across the applied perturbation range, which is summarized in

Table 4.2. In this table, we can see that higher values of ε almost consistently incur a higher standard deviation, i.e. we almost always have $\sigma_{\text{MMD,32Å}} > \sigma_{\text{MMD,16Å}} > \sigma_{\text{MMD,8Å}}$. The degree histogram descriptor under shearing and tapering perturbations seems to be the two exceptions out of 12 cases. Interestingly, the standard deviations were highest when subjecting proteins to the twisting perturbation, most likely due to the fact that the spheres used to construct the graphs most likely increasingly overlap when some degree of twist is applied. In short, the conclusion of the last two paragraphs is that the sparser the graph representation by lowering ε, the more stable the resulting MMD.

4.2.3 Lower ε Values for Graph Contruction Are More Sensitive to Lower Perturbation Regimes

While we noted that choosing a lower ε value to extract the graph would likely improve the stability of resulting MMDs, there is also another consideration when choosing a graph representation. As shown in Figure 4.4, when 20% of the maximum amount of perturbation applied[1], in seven out of 12 cases, the normalized MMD of 8Å graphs is higher than that of the 16- or 32 Å graphs. In addition, we find that, generally, in lower perturbation regimes, higher percentages of the normalized MMD are reached with graphs constructed with a lower ε. Figure 4.5 illustrates this phenomenon. By grouping each configuration of MMD under different perturbation types at the 20% mark of the maximum perturbation applied, we see that the 8Å- and 16Å graphs are significantly higher than that of the 32Å graphs (Mann-Whitney $\mathcal{U}$ test: $p = 1.08 \cdot 10^{-4}$ and $p = 5.56 \cdot 10^{-3}$, respectively). In short, in addition to being overall more stable, lower ε values and sparser subsequent graphs also tend to be better at detecting smaller changes in protein topologies than larger ε values and denser graphs. This is useful for a practitioner in advanced stages of modeling, where small changes in the generated graph population need to be detected.

[1]See Table 3.1 for the respective maxima.

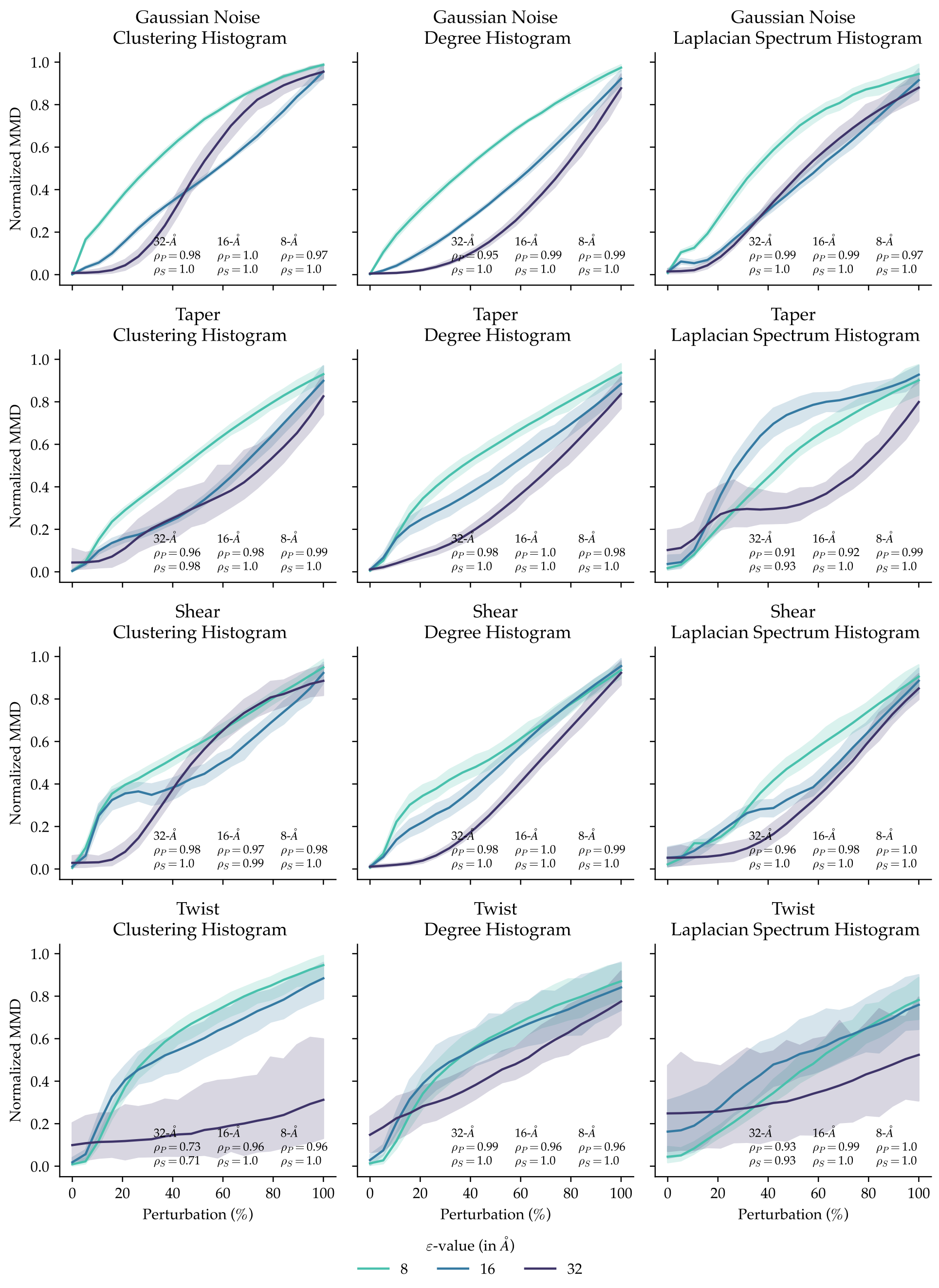


Figure 4.4: MMD vs. point cloud perturbation for various descriptors. In general, when graphs are extracted with a lower ε value, the MMD curve increases more rapidly. The only exception to this trend is the Laplacian spectrum histogram descriptor under the tapering perturbation.

Perturbation Type	**Descriptor Function**	ε	σ_{MMD}
Gaussian Noise	Clustering Histogram	8	0.011
		16	0.015
		32	0.043
	Degree Histogram	8	0.013
		16	0.017
		32	0.024
	Laplacian Spectrum Histogram	8	0.027
		16	0.030
		32	0.034
Shear	Clustering Histogram	8	0.031
		16	0.049
		32	0.055
	Degree Histogram	8	0.041
		16	0.038
		32	0.030
	Laplacian Spectrum Histogram	8	0.041
		16	0.048
		32	0.050
Taper	Clustering Histogram	8	0.026
		16	0.031
		32	0.088
	Degree Histogram	8	0.035
		16	0.049
		32	0.044
	Laplacian Spectrum Histogram	8	0.043
		16	0.053
		32	**0.089**
Twist	Clustering Histogram	8	0.053
		16	**0.082**
		32	**0.179**
	Degree Histogram	8	**0.081**
		16	**0.103**
		32	**0.104**
	Laplacian Spectrum Histogram	8	0.064
		16	**0.145**
		32	**0.229**

Table 4.2: Inter-run standard deviation values averaged across the whole pertubation range for all combinations of perturbation type, descriptor functions, and ε values. Values higher than 0.08 are in bold. Twisting perturbations show particularly high average standard deviations > 0.05, and higher ε values also shows the highest standard deviation values > 0.1 for the twisting perturbations. In almost all cases, we have $\sigma_{\text{MMD},32-\mathring{A}} > \sigma_{\text{MMD},16-\mathring{A}} > \sigma_{\text{MMD},8-\mathring{A}}$. The degree histogram descriptor under shearing and tapering perturbations seems to be the exceptions.

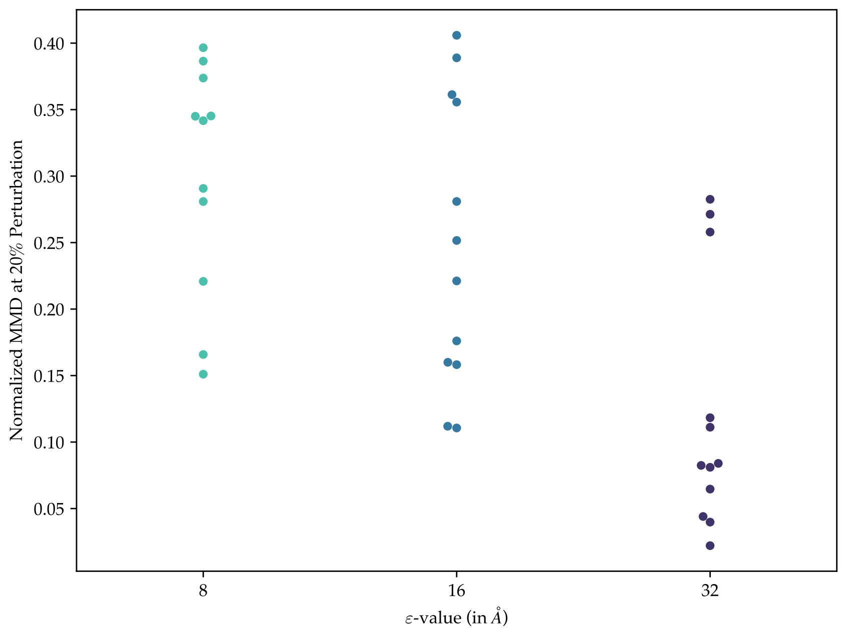


Figure 4.5: Normalized MMD value at 20% of the maximum perturbation amount (as shown in Table 3.1) for various MMD configurations at different ε values. The difference between the 8Å- and 32Å graphs is significant (Mann-Whitney $\mathcal{U}$-test $p = 1.08 \cdot 10^{-4}$). The difference between the 16Å- and 32Å graphs is also significant ($p = 5.56 \cdot 10^{-3}$). However, the difference between the 8Å- and 16Å graphs was not signicant ($p = 3.47 \cdot 10^{-1}, 5 \cdot 10^{-2}$ is our highest significance threshold.)

4.3 Graph Kernels

The results of the perturbation experiments using the Weisfeiler-Lehman kernel (see Sections 2.7 and 3.3.3) in MMD can be seen in Figure 4.6. Two important takeaways can be be derived from this figure.

4.3.1 Quality of MMD Using Weisfeiler-Lehman

The first conclusion is derived from the observation of the meta metrics. We can see that, correlations are relatively low for graph perturbations (Figure 4.6, top row). If we take the kernel using 5 iterations for instance, we get $\rho_P = 0.4$ and $\rho_S = 0.44$ when adding edges. Additionally, from the shape of the curve, we can also derive that the Weisfeiler-Lehman kernel is not sensitive to changes in the number of edges unless an extreme amount is added. Removing edges also results in a similar curve, but correlation coefficients are higher, with $\rho_P = 0.68$ and $\rho_S = 1.0$ (ρ_P is still considerably lower than other configurations of MMD examined thus far, see earlier Sections 4.1 and 4.2 for details). The next perturbation (rewiring edges), highlights the need for the standard deviation estimation of different samples (see Table 4.3), because although the correlation coefficients are reasonable ($\rho_P = 0.71$, $\rho_S = 0.99$), one could realistically not distinguish highly rewired protein graphs from another, because σ_{MMD} is extremely high compared to other configurations or other types of perturbations (for the rewiring, all $\sigma_{\text{MMD}} > 0.28$ while the other $\sigma_{\text{MMD}} < 0.15$). This reflects the high confidence internal we see in the upper right pane of Figure 4.6. For point cloud perturbations, correlation coefficients are in line with the coefficients we have seen before ($\rho_P > 0.94$ and we consistently have $\rho_S = 1$), there seems to be a crucial caveat in the curves that we see in Figure 4.6 which we discuss next.

4.3.2 Insensitivty in Low Perturbation Regimes

Figure 4.6 indeed reveals a systematic pathology when using the Weisfeiler-Lehman kernel in MMD. When applying a low amount of perturbation, there does not seem to be a proportional rise in the (normalized) MMD value. As an example, the normalized MMD only rises substantially from the observed value on the unperturbed set after adding 10% of Gaussian noise to the data (which corresponds to 3.16Å, see Figure 3.1 for an illustration of what 10% of Gaussian noise looks like for a given protein). This 'inverted elbow' shape of the curve seen in almost all perturbation types except for the rewiring regime, where the very high σ_{MMD} seems to obscure any meaningful pattern as discussed in the previous paragraph. This phenomenon is particularly pronounced in the other two graph perturbation scenarios, where 80%-95% of the perturbation needs to be added to observe any meaningful change in MMD. This has dramatic implications for

Perturbation Type	**Iterations**	σ_{MMD}
Add Edges	Iterations: 1	0.018
	Iterations: 5	0.007
	Iterations: 10	0.006
Gaussian Noise	Iterations: 1	0.016
	Iterations: 5	0.014
	Iterations: 10	0.013
Mutation	Iterations: 1	**0.147**
	Iterations: 5	0.072
	Iterations: 10	0.061
Remove Edges	Iterations: 1	0.004
	Iterations: 5	0.003
	Iterations: 10	0.004
Rewire Edges	Iterations: 1	**0.296**
	Iterations: 5	**0.290**
	Iterations: 10	**0.287**
Shear	Iterations: 1	0.079
	Iterations: 5	0.046
	Iterations: 10	0.043
Taper	Iterations: 1	0.037
	Iterations: 5	0.030
	Iterations: 10	0.029
Twist	Iterations: 1	**0.147**
	Iterations: 5	0.072
	Iterations: 10	0.061

Table 4.3: Standard deviations of the various Weisfeiler-Lehman configurations under different perturbation regimes. Values higher than 0.1 are in bold. Rewiring edges results in the highest standard deviations by far with $\sigma_{\text{MMD}} > 0.28$. Lower iterations of the algorithm also results in higher standard deviations.

practitioners, as this means that one cannot distinguish between a generated sample of graphs and a real one unless the generated sample exhibits highly pathological features, which is undesirable for evaluation purposes. While the reason for these findings are not entirely clear, it is possible that the diversity of the hashes obtained during the Weisfeiler-Lehman algorithm is so high that substantial changes need to occur prior to observing any shifts in the dot product of the resulting hash histograms. Further analyses on the diversity of hashes and their relative population needs to be done to validate this hypothesis. Another way to remedy this pathology would be to not use the amino acid as node labels. While this would entail loosing sensitivity to mutations, the diversity of hashes might be substantially reduced so as to ensure that the resulting dot products will be more sensitive to changes in hash populations.

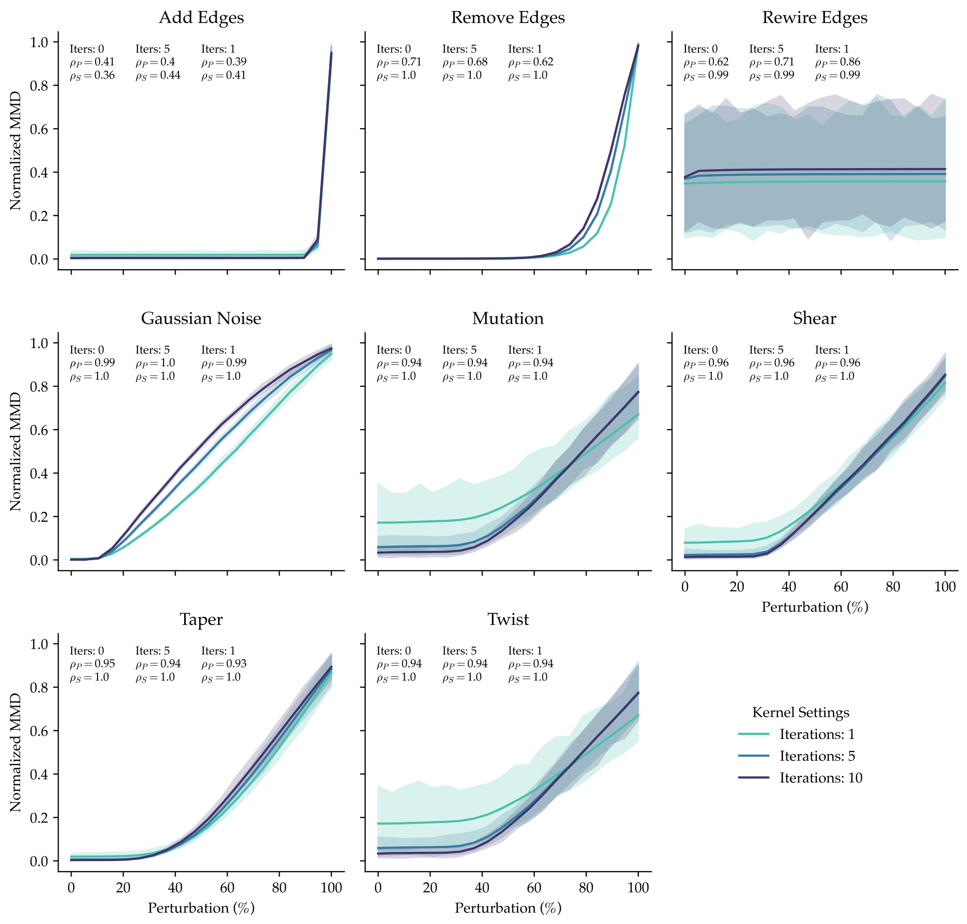


Figure 4.6: MMD vs. perturbations using the Weisfeiler-Lehman kernel using the 8Å-graphs as inputs. We see that high levels of perturbations are required to raise the normalized MMD values. We also see that MMDs computed with the Weisfeiler-Lehman kernel are insensitive to the rewiring of edges, and results in MMDs with a high inter-run variance.

4.4 Protein-Specific Descriptors Are Inexpensive, High-Quality Descriptor Functions

Figure 4.7 shows the normalized MMD as various types of perturbations are added to one set of proteins, where each row has a corresponding perturbation type. Each column also corresponds to a different kernel and kernel parameter configuration.

4.4.1 Dihedral Angles Histograms

We start our examination of the novel, protein-specific descriptors outlined in Section 3.3.2 by discussing the dihedral angles histograms of the φ and

ψ bond angles. We see that for this descriptor, the kernel choice heavily influences the correlation coefficients; indeed, if one chooses the linear kernel or the RBF kernel with a low bandwidth (10^{-5}) or the parameter-free linear kernel, we obtain high correlation coefficients ($\rho_P > 0.95$ and $\rho_S = 1$) if we exclude the Gaussian noise perturbation, which we discuss below. Indeed, correlation coefficients are low for this descriptor when subject to Gaussian noise ($0.41 > \rho_P > 0.39$ and $0.59 > \rho_S > 0.5$). While their associated standard deviation is not particularly high (see Table 4.4), this would indicate that they would not be useful for use in MMD.

However, when examining the profile of the perturbation curve (Figure 4.7, top row), we can see that with very low Gaussian noise levels (as low as 5%, which in this thesis corresponds to 1.5Å, see Table 3.1), the MMD already reaches its highest point. This phenomenon entails that the dihedral angles histogram is not necessarily an inappropriate descriptor, but a very sensitive one. This can be useful to practitioners; indeed, when predicting the 3D structure of proteins from the amino acid sequence for instance, Jumper et al. [2021] had to apply an additional refinement algorithm leveraging Amber force fields [Hornak et al., 2006] to ensure that the bond geometries were not violated (see [Jumper et al., 2021] p. 586 for a detailed discussion). Such a descriptor used in MMD can therefore provide statistical evidence to support the statement that (generated) bond geometries follow a natural distribution.

One conclusion additional conclusion from the analysis of this particular protein descriptor is that although both Pearson and Spearman correlation coefficients as well as σ_{MMD} provide powerful quantitative tools to gauge the quality of a metric, i.e. meta-metrics, one should still carefully consider the profile of individual perturbation experiments as well as reason through the expressive power of a particular descriptor or kernel to accurately assess the practicality of a particular MMD configurations.

4.4.2 α-Carbon Distance Histogram

The distance histogram also serves as a good protein descriptor, and does not exhibit some of the sensitivity-related pathologies that we discussed in Section. 4.4.1. If we set aside the twisting perturbation, which we discuss below, we get high correlation coefficients ($\rho_P > 0.97$ and $\rho_S > 0.99$ irrespective of the kernel or perturbation type chosen). Standard deviations are also reasonably low, since we consistently have $\sigma_{\text{MMD}} < 0.1$ not considering twisting.

The α-carbon distance histogram descriptor as used in MMD does not seem to behave well under twisting perturbations, with both low correlations and high standard deviations. While the RBF kernel configurations seem to have a high correlation ($\rho_P, \rho_S \geq 0.95$), using the linear kernel yields correlation

Perturbation Type	**Kernel**	**Descriptor**	σ_{MMD}
Gaussian noise	Linear Kernel	Dihedral Angles Histogram	0.062
		Distance Histogram	0.071
	RBF Kernel ($\sigma = 1$)	Dihedral Angles Histogram	0.048
		Distance Histogram	0.059
	RBF Kernel ($\sigma = 1 \cdot 10^{-5}$)	Dihedral Angles Histogram	0.042
		Distance Histogram	0.060
Shear	Linear Kernel	Dihedral Angles Histogram	0.037
		Distance Histogram	0.066
	RBF Kernel ($\sigma = 1$)	Dihedral Angles Histogram	0.035
		Distance Histogram	0.050
	RBF Kernel ($\sigma = 1 \cdot 10^{-5}$)	Dihedral Angles Histogram	0.028
		Distance Histogram	0.051
Taper	Linear Kernel	Dihedral Angles Histogram	0.059
		Distance Histogram	0.079
	RBF Kernel ($\sigma = 1$)	Dihedral Angles Histogram	0.040
		Distance Histogram	0.062
	RBF Kernel ($\sigma = 1 \cdot 10^{-5}$)	Dihedral Angles Histogram	0.049
		Distance Histogram	0.063
Twist	Linear Kernel	Dihedral Angles Histogram	0.060
		Distance Histogram	**0.250**
	RBF Kernel ($\sigma = 1$)	Dihedral Angles Histogram	0.039
		Distance Histogram	**0.228**
	RBF Kernel ($\sigma = 1 \cdot 10^{-5}$)	Dihedral Angles Histogram	0.050
		Distance Histogram	**0.230**

Table 4.4: Standard deviation of the various protein-specific descriptors devised in this thesis. $\sigma_{\text{MMD}} > 0.2$ are in bold. This table reflects two observations made from Figure 4.7: first, that overall standard deviation of the MMD values across the perturbation range is quite low. Second, the distance histogram seem to result in particularly high standard deviations when subject to twisting perturbations.

coefficients as low as $\rho_P = 0.35$ and $\rho_P = 0.37$. Additionally, the standard deviation for the MMD curves obtained using the α-carbon distance histogram descriptor are high across kernel choices ($\sigma_{\text{MMD}} > 0.22$), which does not make it a very robust descriptor.

While less sensitive than the dihedral angles descriptor, the α-carbon distance histogram descriptor still provides several advantages over the dihedral angles histogram. Although we have not tested this here, this descriptor can certainly be used to accurately gauge the size in 3D space of the proteins generated. This can be useful when crafting generative models that generate proteins conditioned on overall size in 3D space. Additionally, as reflected through the different perturbations shown in Figure 4.7, this descriptor is mostly able to detect changes in the interatomic space. This can also allow a practitioner to detect atom clashes and overall unrealistic distances in a protein model, much like the atom clash detection steps in the PDB validation pipeline [Read et al., 2011, Gore et al., 2012, 2017]. This could be further validated by progressively increasing the distance of each atom from the center of mass of the proteins.

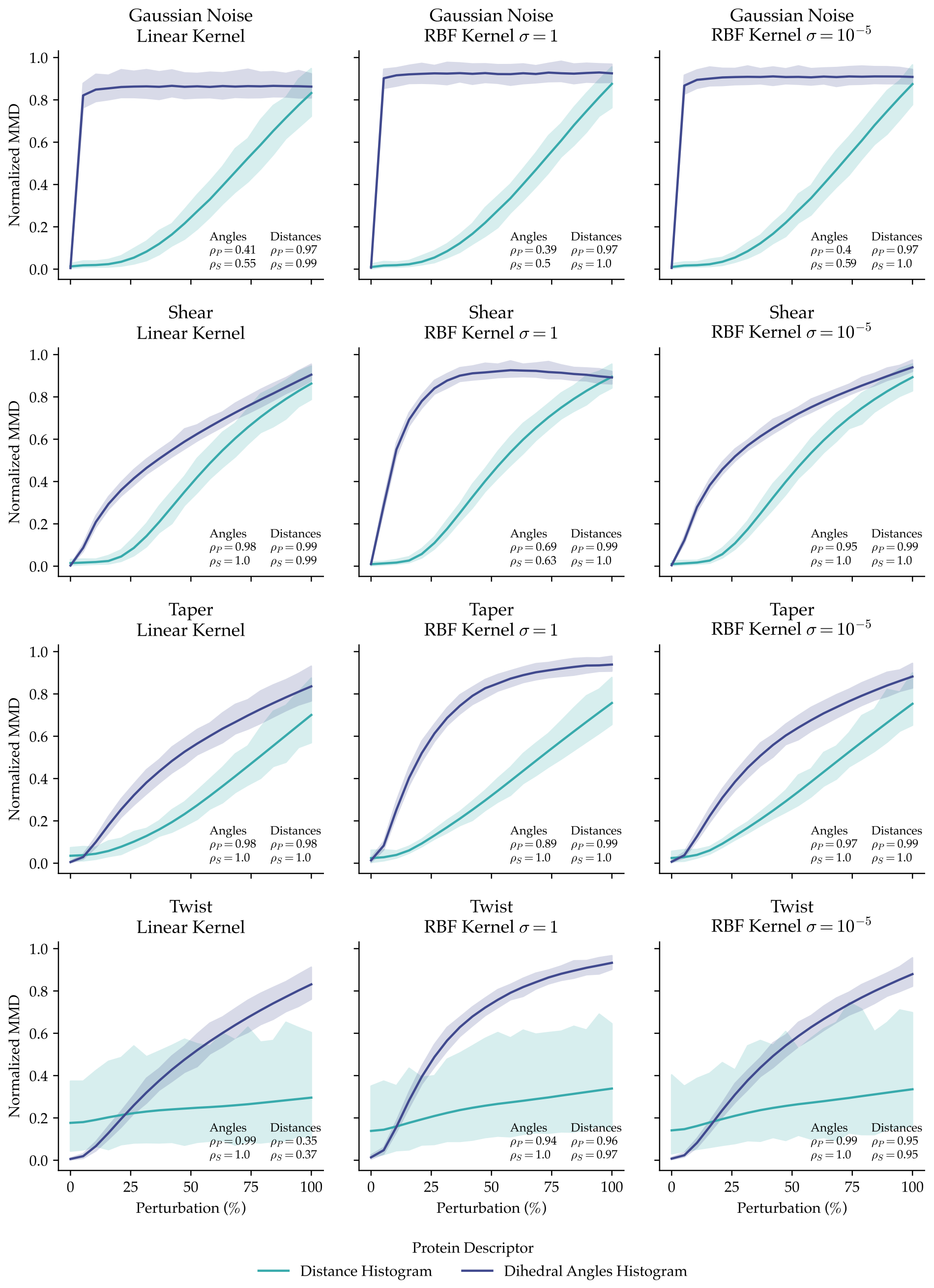


Figure 4.7: MMD vs. perturbations (in % of the maximum values shown in Table 3.1) using the two novel protein descriptors shown in Section 3.3.2. Each row has a corresponding perturbation type. Each column also corresponds to a different kernel and kernel parameter configuration. Those descriptors behave very well overall, with the dihedral angles descriptors being particularly sensitive to Gaussian noise. The distance histogram exhibits higher inter-run variance when subject to twisting pertubations.

Kernel	σ_{MMD}
Linear Kernel	0.091
RBF ($\sigma = 1e-05$)	0.089
RBF ($\sigma = 0.0001$)	0.092
RBF ($\sigma = 0.001$)	0.093
RBF ($\sigma = 0.01$)	0.093
RBF ($\sigma = 0.1$)	0.095
RBF ($\sigma = 1$)	**0.235**
RBF ($\sigma = 10.0$)	**0.447**

Table 4.5: σ_{MMD} values for the MMD using the ESM learned embedding. $\sigma_{\text{MMD}} > 0.1$ are in bold. These findings corroborate the correlation coefficients shown in Figure 4.8: using a linear kernel or an RBF kernel with$\sigma < 0.1$, we get $\sigma_{\text{MMD}} < 0.1$, we obtain higher quality (here, more robust) metrics, since σ_{MMD} is relatively low.

4.5 MMD from Learned Embeddings

As we noted in Section 2.7, the only constraint on some of the input vectors is that they need to be in $\mathbb{R}^d$. Additionally, as noted at the end of Section 4.3.2, there has been much progress in the development of learned embeddings. Finally, due to some of the shortcomings of the Weisfeiler-Lehman kernel highlighted in Section 4.3.2, we wanted to investigate the representative power of learned embeddings in MMD, the result of which are shown in Figure 4.8. In this figure, we see that choosing an RBF kernel with $\sigma \leq 0.01$ or a linear kernel results in excellent correlation coefficients with $\rho_P = 0.94$ and $\rho_S \geq 0.96$ across kernels with $\sigma \leq 0.01$. The standard deviation of the MMD across this range of kernels is also reasonable, with $\sigma_{\text{MMD}} < 0.1$ across the linear kernel and RBF kernel with $\sigma < 0.1$ (see Table 4.5 for details). Further work should focus on different kinds of perturbations applied to sequences and see if correlation coefficients stay consistent across kernel configurations. Results conducted by Kucera et al. [2022] suggest that this is not the case, highlighting the need for further work.

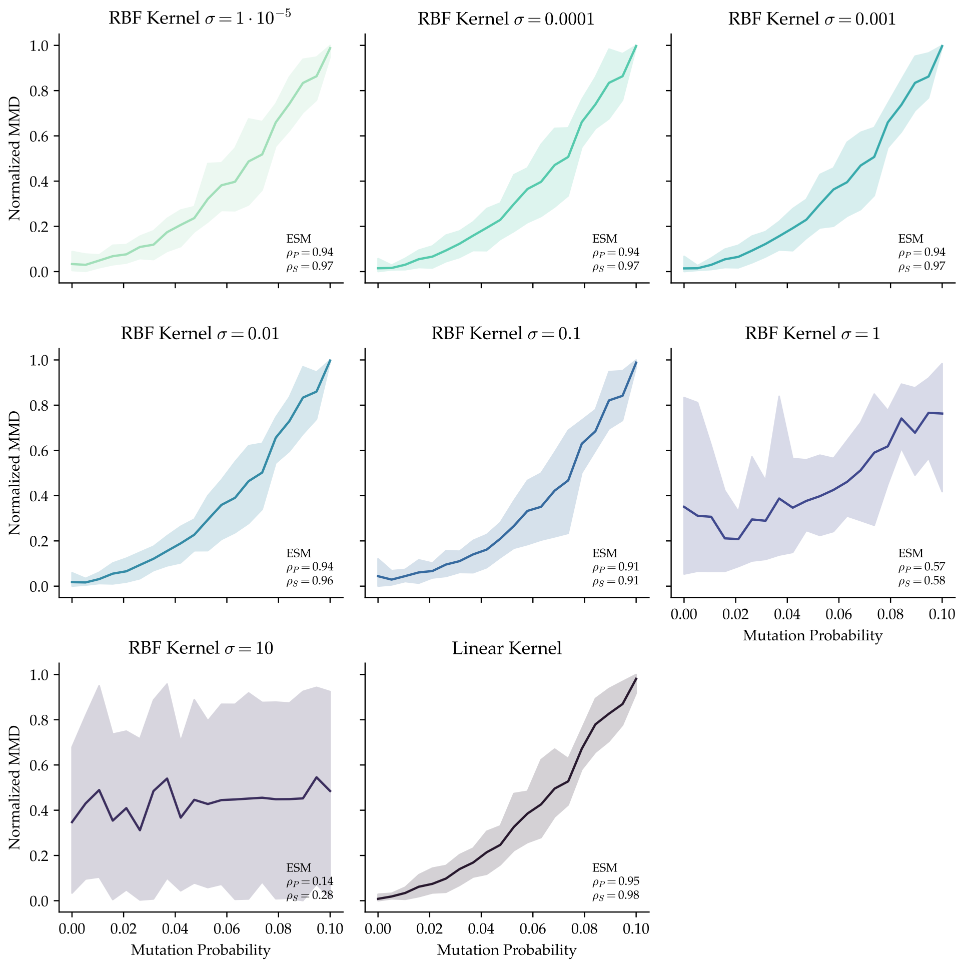


Figure 4.8: MMD vs. mutation probability using the ESM learned embeddings with various kernels. The correlation coefficients of the resulting MMDs with RBF kernels with $\sigma < 1$ and the linear kernel are high, therefore making them good configurations for sequences. This analysis should be complemented by complementary sequence-specific perturbations – see Section 5.2.3 for details.

Perturbation Type	**Kernel**	σ_{MMD}
gaussian_noise	Multi scale kernel	0.017
	Persistence Fisher Kernel	0.013
shear	Multi scale kernel	0.025
	Persistence Fisher Kernel	0.016
taper	Multi scale kernel	0.014
	Persistence Fisher Kernel	0.016
twist	Multi scale kernel	0.044
	Persistence Fisher Kernel	0.024

Table 4.6: σ_{MMD} values for the MMD using the persitence Fisher kernel (PSK) and the multi scale kernel (MSK).

4.6 Topological Descriptors and Kernels

Figure 4.9 shows the behavior of both the MSK and PFK. Overall, we can see that they are among the best performing metrics in this thesis, with high correlations and low inter-run variation. The PFK seems to perform the best based on the correlation coefficients, with $\rho_P > 0.95$ and $\rho_S = 1$ across perturbation types – while the the MSK shows $\rho_P > 0.93$ and $\rho_S > 0.99$. The PFK also has slightly lower σ_{MMD} values than the MSK with σ_{MMD}, with a maximum of 0.024 vs 0.044 for the MSK – see Table 4.6. Additionally, when looking at each perturbation type, we seem to have $\sigma_{\text{MMD, PSK}} < \sigma_{\text{MMD, MSK}}$ except for the tapering perturbation type, where there is a 0.02 difference $\sigma_{\text{MMD, PSK}} < \sigma_{\text{MMD, MSK}}$. These differences in kernel performance are minor, however, and both kernels perform consistently really well compared to the alternative configurations explored in this thesis. Another last finding related to the difference between the two kernels is that with identical bandwidths and perturbation level, the MSK seems to be consistently higher than the PFK. Therefore, if a more sensitive MMD configuration is required, the MSK might be preferred.

While the current TDA workflow cannot detect changes in node labels in the current setting, hence not being able to detect mutations, we will show in Section 5.3.2 how this could be alleviated. For computational reasons, we did not compute variations of different kernel parameters, although one could speed up the operation by precomputing the kernel matrices, and only subsequently multiplying the matrix by different constants (i.e. implement a modified version of the speed up trick explored in Appendix A.5 of O'Bray et al. [2022]).

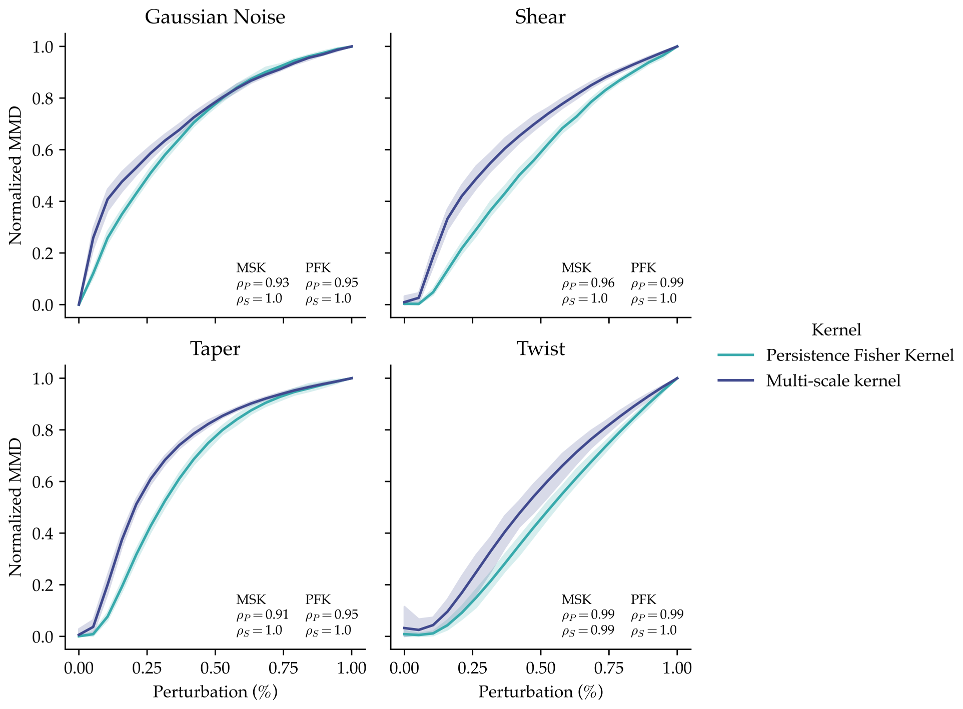


Figure 4.9: MMD vs. Perturbation (in % of the maximum values shown in Table 3.1) using the MSK and persistence Fisher kernel. For the PFK, both the kernel bandwidth parameter and the Fisher bandwidth parameter are set to 1. For the MSK (MSK), the bandwidth parameter is also set to 1. Overall, the correlation coefficients for those kernels are very high and show little inter-run variance, which is also supported by the σ_{MMD} values in Table 4.6.

4.7 Runtime

One of the desiderata of MMD is efficiency, i.e. a low computational complexity (see Section 2.5). As such, we report the various computation times for each of the important elements of the pipelines outlined in this thesis. The summary of all execution times can be found in Table 4.7.

First, we see that one computation stands out: computing the Vietoris-Rips filtration of a point cloud. Together with a large memory footprint, the Vietoris-Rips filtration is expensive to compute (3332s in our benchmarks), due to a runtime of $\mathcal{O}(n^{3(k+2)})$, where k is the number of dimensions (here, $k = 3$) and n the number of points [Adams et al., 2018]. Although some optimizations have been carried out by `Ripser` Bauer [2021], the software used to compute the Vietoris-Rips filtration, it remains very expensive to compute and does not enjoy the benefits of hardware acceleration through e.g. a GPU due to the difficulty of parallelizing such a filtration.

Second, computing the distance histogram and dihedral angles histogram are the fastest descriptors to compute (28s for the former and 4s for the latter), making them suitable to evaluate large collections of generated proteins. Other graph descriptors are also reasonably fast to compute (35-175s in our benchmarks, see Table 4.7), but they depend on the settings used to extract the graphs, and, generally, the runtime increases in the number of edges of the graph.

Third, the kernels used in this study all have reasonable runtimes, but we note the particular efficiency of the RBF and linear kernel ($<$ 7.7ms in our benchmarks) compared to the persistence Fisher, multi-scale and Weisfeiler-Lehman kernel. In addition, the PFK ran thee times slower (35s) than the MSK (11s). The implications of these runtimes will be further explored when making recommendations to the practioner in Section 5.2.

Operation	**Execution Time**
Descriptor Functions	
Vietoris-Rips Filtration	3332 s
ESM Embedding	163 s
Degree Distribution Histogram (32Å-graph)	35 s
Clustering Coefficient Histogram (32Å-graph)	175 s
Laplacian Spectrum Histogram (32Å-graph)	35 s
Distance Histogram	28 s
Dihedral Angles Histogram	4 s
Kernels	
Weisfeiler-Lehman Kernel (4 iterations)	32 s
Persistence Fisher Kernel	35 s
Multi-Scale Kernel	11 s
RBF Kernel	1.7 ms
Linear Kernel	7.7 ms

Table 4.7: Runtime and computational complexity of the various elements of the pipeline. These timings are obtained by executing the operation on 100 samples spread across 10 CPU cores from an Intel Xeon Gold 6254 CPU clocked at 3.10GHz.

4.8 Summary

In this chapter, we present the results of the behavior of MMD under various configurations and subject to different types of perturbations. We start by examining the graph descriptors used in the literature to evaluate generative graph models (4.1). For those descriptors, we notice that the correlation coefficients generally behave well, with the exception of the degree histogram, which shows lower correlations and in some cases higher standard deviations, making it the least expressive and robust of all descriptors for graph representations of proteins (Section 4.1.1 and 4.1.2). We also examined the influence of the choice of kernel, and observed that the RBF kernel with $\sigma < 0.1$ and the linear kernel showed high correlation coefficients, but that those coefficients drop sharply when increasing $\sigma > 0.1$, which likely due to the scale of the various descriptors, and for which increasing σ results in oversmoothing (Section 4.1.1). We then investigated the quality of the extracted graphs following different graph extraction techniques and parameters (Section 4.2). In Section 4.2.1, we established that ε-graphs were often better to use based on the correlation coefficient distribution. Next, in Section 4.2.2, we established that lower ε values yielded more stable MMD configurations. To wrap up our discussion of the combinations of descriptor and kernel traditionally used in the literature, we concluded in Section 4.2.3 that at lower perturbation regimes, higher values of the normalized MMD were reached by graphs constructed with a lower ε threshold.

We then moved on to our discussion of the Weisfeiler-Lehman kernel, which is a kernel operating directly on graphs, and noted it's low relative quality compared to MMD configured with graph descriptors, both in terms of correlation and standard deviation (Section 4.3.1). Further, we noted no increase in normalized MMD in low regimes of perturbation for MMDs configured with the Weisfeiler-Lehman kernel 4.3.2.

Next, we examined the protein-specific descriptor functions that we devised for use in MMD (Section 4.4). We found that both the dihedral angles (Section 4.4.1) and the distance histograms (4.4.2) were appropriate protein descriptors, although they investigate different aspects of the protein topology (bond angles and interatomic distances, respectively).

We briefly examined sequence embeddings in Section 4.5 and although the early mutation results seem promising, more work needs to be done to properly assess MMD configurations applied to sequences.

We discuss the last set of MMD configurations in Section 4.6, where we find that kernels operating on persistence diagrams tend to work exceptionally well to capture point cloud perturbations. We noted that the MSK was slight more sensitive (i.e. the resulting MMDs were consistently higher for this kernel at any given perturbation level) and but slightly less well performing than the PFK in terms of correlation and standard deviation, although those differences were minor.

Finally, we conclude this chapter with a summary of the runtime of the various elements of the computational pipleine to obtain MMD values (Section 4.7), and note that (i) the Vietoris-Rips filtration is excessively slow to compute, (ii) that the protein descriptors are particularly fast to compute, and (iii) comment on the various kernel runtimes.

Chapter 5

Discussion

In this chapter, we discuss the findings of this thesis presented in Chapter 4, which we will summarize in Section 5.1. We then formulate a set of recommendations in Section 5.2, and finally highlight some limitations and future directions of research in Section 5.3.

5.1 Key Findings

Overall, we found that MMDs show high correlation coefficients on graphs extracted from proteins (see Section 4.1). This finding is somewhat surprising considering recent findings indicating that MMD seems often me unstable on synthetic graphs such as Erdös-Rényi graphs, Watts-Strogatz graphs, and Barabási–Albert graphs under some combinations of perturbation types, kernels and descriptors [O'Bray et al., 2022]. While we found configurations of MMD with low correlations and high σ_{MMD} (see for instance Figure 4.1 upper left pane with the degree histogram and the Weisfeiler-Lehman kernel-based MMDs where protein graphs are subject to rewiring, see Figure 4.6), such MMD configurations were not prevalent. We hypothesize that this is due to the highly structured nature of the graphs used in this thesis (see Figure 5.1 for an example).

Furthermore, we found that the correlation coefficients and standard deviation of MMD configurations were greatly influenced by the graph representation extracted from the protein. Namely, for ε graphs (which overall seemed more stable than k-NN graphs, see Figure 4.3), the higher the ε value to extract the graph, the lower the correlation to the perturbation applied (Figure 4.4).

Next, we introduced in Section 3.3.2 two *novel* protein-specific descriptors that can be used in MMD. In Section 4.4, we show that those descriptors are both fast to compute (Table 4.7) and are able to detect point cloud perturbations with high fidelity (Figure 4.7, see also Section 4.4 for a more detailed discussion).

We found that the Weisfeiler-Lehman kernel, introduced in Section 3.3.3

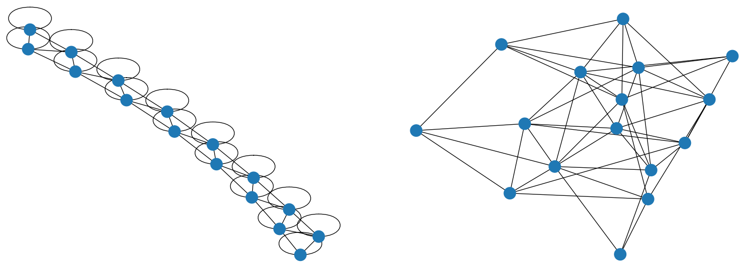


Figure 5.1: Example 8-Å-graph extracted from a protein versus Erdös-Rényi graph with the same number of nodes and edges. The protein entry used here to obtain the 8-Å graph is an uncharacterized protein (UniProtKB ID: A0A6Q8PFQ6). We chose this protein for illustrative purposes because it was the shortest protein found in the human proteome as predicted by the AlphaFold2 model. The self-loops shown in the protein graph are present in all graph extraction techniques used in this thesis.

applied to MMD was able to detect perturbations unable to be detected by traditional graph descriptors such as point mutations, which are relevant for the protein domain (see Section 4.3). However, the resulting MMD meta metrics obtained using the Weisfeiler-Lehman kernel were not as high as for other MMD configurations used in this thesis (see Section 4.3.1 for a detailed examination). Furthermore, we found that Weisfeiler-Lehman-based MMD configurations are not sensitive (i.e. the normalize MMD does not increase) when lower levels of perturbation are applied (see Figure 4.6). This can be alleviated by using an alternative descriptor such as the ESM protein embedding introduced in Section 2.5, which is more sensitive (i.e. requires less perturbation to reach higher normalized MMD values) to lower rates of point mutation (see Figure 4.8).

We analyzed the MMD obtained from kernels operating on persistence diagrams in Section 4.6, which show excellent meta metric performance since MMD values leveraging topological kernels both exhibit high correlations to point cloud perturbations and low σ_{MMD}. While unable to detect point mutations, we propose an alternative to be topological investigation of such point clouds in Section 5.3.2.

5.2 MMD in Practice: Recommendations

Leveraging those key findings, we now formulate a set of recommendations for a practitioner who needs to evaluate a generative protein model using one of the protein representations highlighted above.

5.2.1 Setting Up Appropriate Baselines

Overall, we advise the practitioner to carefully evaluate certain baselines when possible. The first baseline would be to evaluate the MMD between different i.i.d. samples of the reference distribution to establish the range of MMD to be expected in the best case, which we refer to as the *positive control*. We conducted such positive controls for various MMD configurations in Appendix A.4. From there, an accurate assessment of the quality of the model can be made. If the kernel and data representation allows it, it is also advisable to establish a *negative control*. This would show the worst possible performance between any two distributions. If this is not possible, as is the case in this thesis (e.g. there is no negative control for all MMD values perturbed using the Gaussian noise), it is useful to examine cases of extreme perturbations and compare the resulting MMD values to those. Such an assessment can provide a surrogate for the negative control and confers a sense of scale for the particular problem tackled. As with discriminative models, it is also useful to compare the MMD obtained from the generated model with other MMDs obtained from other models used in the literature to make comparative benchmarks of performance. In such a context, taking into account the values of σ_{MMD} to establish the robustness of a particular MMD configuration is important to establish the reliability of the resulting MMD values.

Moreover, since the MMD statistic is the basis for a kernel two-sample test [Gretton et al., 2012] (see also Section 2.6), provided the kernel is powerful enough and can be computed with reasonable computational resources, one can estimate a p-value from the statistic between the model's generated distribution and the empirical distribution.

5.2.2 Taking MMD Sensitivity into Consideration

Depending on the stage of modeling and the coarseness of the desired samples, one might choose different MMD configurations. As we have shown in Figure 4.4 and discussed in Section 4.2, choosing a lower threshold ε to extract ε-graphs from a protein point cloud results in a higher sensitivity to a host of perturbation. In practice, this means that the lower the ε, the better the MMD will be at discerning perturbed (i.e. generated) samples from reference samples. This might be desirable if one needs highly similar distributions of graphs. However, it can sometimes be desirable to relax this requirement, e.g. to explore a larger part of the design landscape or make a more approximate assessment of the generated samples for model selection purposes.

In addition, when choosing an MMD configuration, one should also investigate the magnitude of inter-run variance as a proxy for robustness, since the possible ranges of MMD values can be too wide to make any assessment as to the quality of the generated samples in the real-world. This can be achieved either by running perturbation experiments as we did in this thesis, or by sampling subsets of the generated and real data to estimate how much a particular MMD configuration varies from one subset to another.

5.2.3 Assessing Realistic Proteins

Some key findings of this study can be useful is in the context of assessing *realistic* proteins. Defining realistic proteins take on a myriad of aspects: i.e. are the generated protein *sequences* realistic? In this case, one might use a Weisfeiler-Lehman kernel or an ESM embedding to answer this. The literature, however, suggests that using embeddings does not yield the same optimal kernel parameters depending on the perturbation applied to the sequence. As such, it is recommended to use a spectrum kernel [Leslie et al., 2002] for sequences (see Kucera et al. [2022], Section 4.1). There are, however, other aspects of proteins to consider, the most important of which is their 3D shape. For this purpose, the protein-specific descriptors devised in this thesis and introduced in Section 3.3.2 whose results are shown in Section 4.4, Figure 4.7 can be used. This way, an unusual angle or abnormal interatomic distance distribution observed in the data will be reflected in the MMD value.

5.2.4 Choosing the Right Kernel and Kernel Parameters

In this thesis, we consistently observed that the linear kernel and RBF kernels with $\sigma < 0.01$ were often effective to detect perturbations and have shown increased correlations and lower standard deviations. Conversely, $\sigma > 1$ often resulted in insensitive and unstable MMD values. When investigating the mean distance distribution between the various embeddings of data points (Appendix A.3), we can see that this is because the order of magnitude of the data descriptors is consistently higher than this threshold, and choosing a threshold that is excessively high results in oversmoothing. We discuss the order of magnitude of the data and its impact on the choice of σ in Appendix A.3 as well.

5.3 Limitations and Future Directions

We summarized the main findings of this thesis (Section 5.1) and formulated a set of recommendations to the practitioner (Section 5.2). We finish this chapter by highlighting important limitations of this thesis.

5.3.1 Establishing Pseudo-Negative Controls

In Section 5.2.1, we highlighted the necessity of a *positive control*. While this is possible for certain kernels (e.g. for the aforementioned spectrum kernel, see Kucera et al. [2022]), many of the settings discussed in this thesis do not have such a *negative control*. While one could use absolute MMD values of perturbed sets of proteins as a reference for subpar performance, it is still highly recommended to find appropriate pathological cases for specific applications to estimate what a worst-case MMD value might take, i.e. establish a pseudo-negative control.

5.3.2 Limitations and Future Directions of TDA in MMD

One of the novel applications of TDA discussed in this thesis is its adoption in MMD by computing the kernel on two collections of point clouds, hence leveraging the shape of the proteins in the computation of MMD. Two challenges could potentially be prohibitive in adopting this approach. The first is computation time (see Table 4.7 and discussion in Section 4.7): the Vietoris-Rips filtration is expensive to compute. The second drawback is expressivity: the vanilla version of the Vietoris-Rips filtration is not sensitive to amino acid changes. Both issues could be tackled by dividing each point cloud into 20 different points cloud (1 for each type of amino acid) following a similar approach as was done for atoms, which has been shown to be powerful [Jiang et al., 2021]. The benefits of this approach are two-fold. On the one hand, it makes the Vietoris-Rips filtration sensitive to changes in the amino acids in addition to shape-related changes. On the other hand, it speeds up computation time, because each point cloud is more sparsely populated (see Section 4.7 for details), and because each amino acid-specific point cloud can be computed in parallel.

5.3.3 MMD and Mode Collapse

Mode collapse and mode dropping are the two distinctive and common failure modes of generative models [Salimans et al., 2016]. We define mode collapse as the situation when a particular type of generated output (i.e. intra-mode outputs) lacks variety. Mode drop refers to the situation when some modes of the data are not represented in the generated output. Both arise when implementing common generative model architectures such as GANs. Although some methods have been devised to tackle such issues for GANs [Arjovsky et al., 2017, Goodfellow et al., 2014], it remains a challenge that needs to be tackled by suitable evaluation measures. Since MMD takes the average of kernel matrices (see Equation 2.11), we anticipate its potential to detect mode collapse is limited, since MMD is invariant to changes that do not affect the mean of the distributions to be compared.

While others have simulated such pathologies in synthetic datasets by manually changing the composition of each mode and investigating evaluation metrics' behaviour to it [Thompson et al., 2022], applying such mode-related perturbations to protein datasets have yet to be tackled and are beyond the scope of this thesis. One method that could be used to investigate such situations would be to modify the composition of various protein families, within which proteins share structural similarities. To simulate mode collapse, one could impoverish the number of proteins within a given family. To simulate mode drop, one could remove those proteins entirely from the distribution. Defining protein families could be achieved by looking at evolutionary links between proteins using for instance CATH database [Orengo et al., 1997]. Alternatively, one could also use structural hierarchies to define protein families using the SCOP database [Murzin et al., 1995].

5.3.4 Kernel Composition

Kernel composition is a mechanism by which one can chain kernel functions to combine different representations of data points by either multiplying or adding kernel matrices. Although we did not investigate kernel composition here, because we wanted to assess the expressive power of individual kernels and representations, a practitioner could combine those to obtain an even richer descriptors combining the advantages of various MMD configurations in this thesis. As such, a practitioner might consider TDA-derived kernels together with protein-specific descriptors to capture global properties of proteins while ensuring that bond geometry is not violated between residues. Note that this approach would also have the effect of propagating any lack of robustness of one of the kernels in the chain to the composed kernel, which could potentially be nefarious. Overall, we recommend to choose which aspects of proteins are particularly important to capture based on the application, and select a range of descriptor functions and kernels that capture those aspects appropriately.

5.4 Summary

In Section 5.1, we show that the MMD on the real-world graphs used in this thesis is more stable than for synthetic graphs used in the literature. We highlight the sensitivity of MMD to the underlying parameters used to extract graph representations of proteins. We then discuss the novel, computationally efficient, and expressive novel protein descriptors. Furthermore, we summarize the findings related to the graph kernels, namely that they seem to not be the most appropriate kernels for MMD based on correlation and standard deviation meta-metrics. We close this section with a discussion on TDA-derived kernels in MMD, which seem to be excellent kernels despite being less inefficient.

In Section 5.2, we first advise the practitioner to establish a positive control to estimate what a best-case MMD value could be. Secondly, if conditions allow, one can also establish a negative control to estimate a worst-case MMD value. We also redirect the practitioner to our findings to use representations that are more or less susceptible to perturbations depending on the required fidelity of generated samples to the reference distribution. We then move on to discuss the various aspects that constitute the assessment of what makes a realistic protein. We conclude our recommendations with relevant kernel choices.

We outline the limitations of this thesis in Section 5.3 by starting off with discussing the often unfeasible crafting of negative controls. We then move on to discuss the limitations of a particular set of MMD configurations using TDA and how to potentially solve it. We then highlight the fact that MMD is insensitive to generative model pathologies that do not affect the mean of the distributions due to the averaging of the kernel matrices. We close our discussion by highlighting the composability of the kernels, which could greatly expand the building blocks highlighted in this thesis in future work.

Chapter 6

Conclusion

In this thesis, we performed a meta-evaluation of metrics based on MMD for protein generative models. We first examined known configurations of MMD, specifically by extracting graphs and using common descriptor functions such as the degree distribution, laplacian spectrum and clustering coefficient histogram to obtain a fixed-length vector, which we can then feed to appropriate kernels such as the linear kernel and the Gaussian kernel.

We then expanded the configurations of MMD to include a kernel that operates directly on graphs: the Weisfeiler-Lehman kernel, which we optimize to decrease runtime complexity on sparse graphs such as those used here. Furthermore, we used topological kernels such as the MSK and the persistence Fisher kernel to capture topological features of the protein. Finally, we designed two new protein-specific descriptors: the interatomic distance histogram and the dihedral angle histogram, which were inspired by the validation pipeline of Protein Data Bank.

Overall, we find that the majority of MMD configurations performed well with respect to the meta-metrics of expressivity and robustness, i.e. correlation coefficients and standard deviations. We hypothesize that the highly structured nature of the graphs used here explain why our MMD configurations behaved more predictably than those in O'Bray et al. [2022]'s perturbation experiments, where synthetic, more unstructured graphs were investigated.

Furthermore, we found that the procedure by which graphs were extracted from proteins had a sizeable influence on the resulting sensitivity of MMD to lower regimes of perturbations. For instance, a lower value of ε results in MMD values that were more sensitive to perturbations.

While the Weisfeiler-Lehman kernel performed relatively poorly on the specific graphs extracted in this thesis, we found that MMD configurations leveraging topological representations and the protein-specific descriptor functions behaved well under all appropriate perturbation regimes. However, the runtime of TDA-related workflows were substantially higher than the others, while the protein-specific descriptors we introduced here were among the fastest to compute.

The approach devised in this thesis has several limitations. First, a negative control cannot always be established, i.e. a worst-case MMD - practitioners then have to be satisfied with a highly perturbed set of proteins as a proxy for poor performance. One of the major drawbacks of MMD is that it is in general not sensitive to model pathologies that do not affect the average kernel similarity between samples, e.g. when mode collapse occurs.

To conclude, MMD forms the basis of a powerful computational platform to evaluate generative models. Configured with a powerful combination of representations, descriptor functions, and kernels, MMD could hopefully accelerate the design of even better generative models for protein design applications and beyond.

Appendix A

Appendix

A.1 Influence of Kernel Parameters on Sensitivity to Perturbations of k-NN Graphs

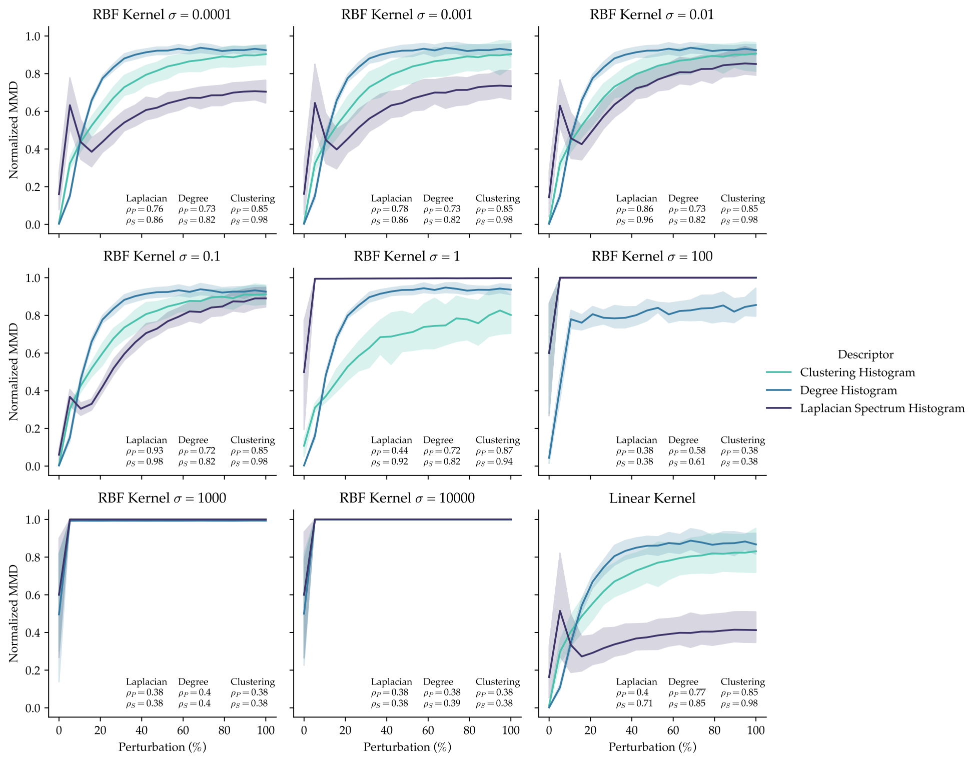


Figure A.1: MMD vs. Gaussian Noise Perturbation (in %) for various graph descriptors and kernel parameters of the 2-NN-graphs. The kernel here is shown on top of each subplots.

A.2 Weisfeiler-Lehman Runtime Improvements for Sparse Graphs

In this thesis, we devised a three-pronged method to speed up the runtime of the Weisfeiler-Lehman kernel by approximately 80% by leveraging the sparsity of the graphs used here. The three elements contributing to the speedup are the following:

1. Our implementation parallelizes the execution of the Weisfeiler-Lehman hash computations since each graph's hash can be computed independently prior to computing the kernel.
2. It also parallelizes the computation of similarity of graphs in RKHS by computing batches of the inner products independently.
3. When comparing graphs, lots of CPU cycles are spent processing positions/hashes that do not overlap between Weisfeiler-Lehman histograms. As such, we manually loop over the overlapping keys, outperforming NumPy dot product-based implementations on collections of sparse graphs.

We tested, covered, and open-sourced the implementation of this novel approach on GitHub, and is available at the following URL: `https://github.com/pjhartout/fastwlk`.

A.3 Distance Distribution of Descriptor Functions

To explain why certain Gaussian configurations tend to work better than others, it is useful to compute the pairwise distance of unperturbed proteins to get an idea of how far away in e.g. Euclidean space each embedding is from one another. Figure A.2 shows the pairwise Euclidean distance distribution within an unperturbed set of proteins. We can see that this varies quite a bit from descriptor to descriptor and that this distribution is also affected by the parameters used to extract the graphs in the case of graph descriptors. This probably explains why we see different behaviors for different kernels in Figure 4.2.

Although we did not do so in this thesis, one could leverage this information to set the σ parameter for the RBF kernel by centering it around the mean, and (logarithmically) scaling it up and down to find configurations with the highest correlations.

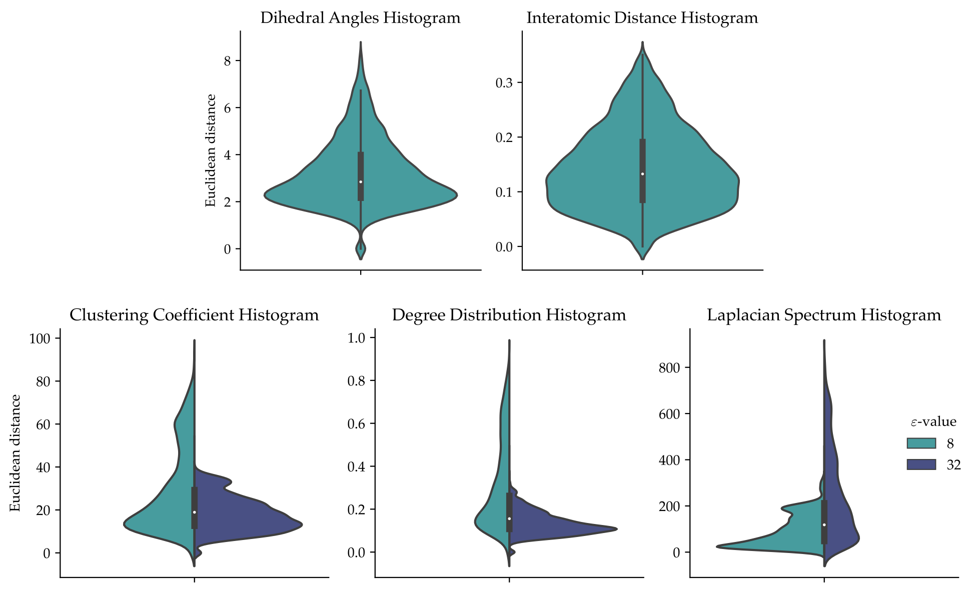


Figure A.2: Mean Euclidean distance between typical descriptor vectors used in this thesis. The top row shows the mean distance between each of the protein descriptors introduced in this thesis while the bottom row shows the distance between each graph descriptor for two ε-graph extraction settings.

A.4 MMD Baselines for Various Configurations

Baseline MMD distributions between two sets of 100 unperturbed proteins with various MMD configurations are shown here. This motivates why the practicioner should always make a positive control, as various MMD configurations can have vastly different ranges depending on the configurations.

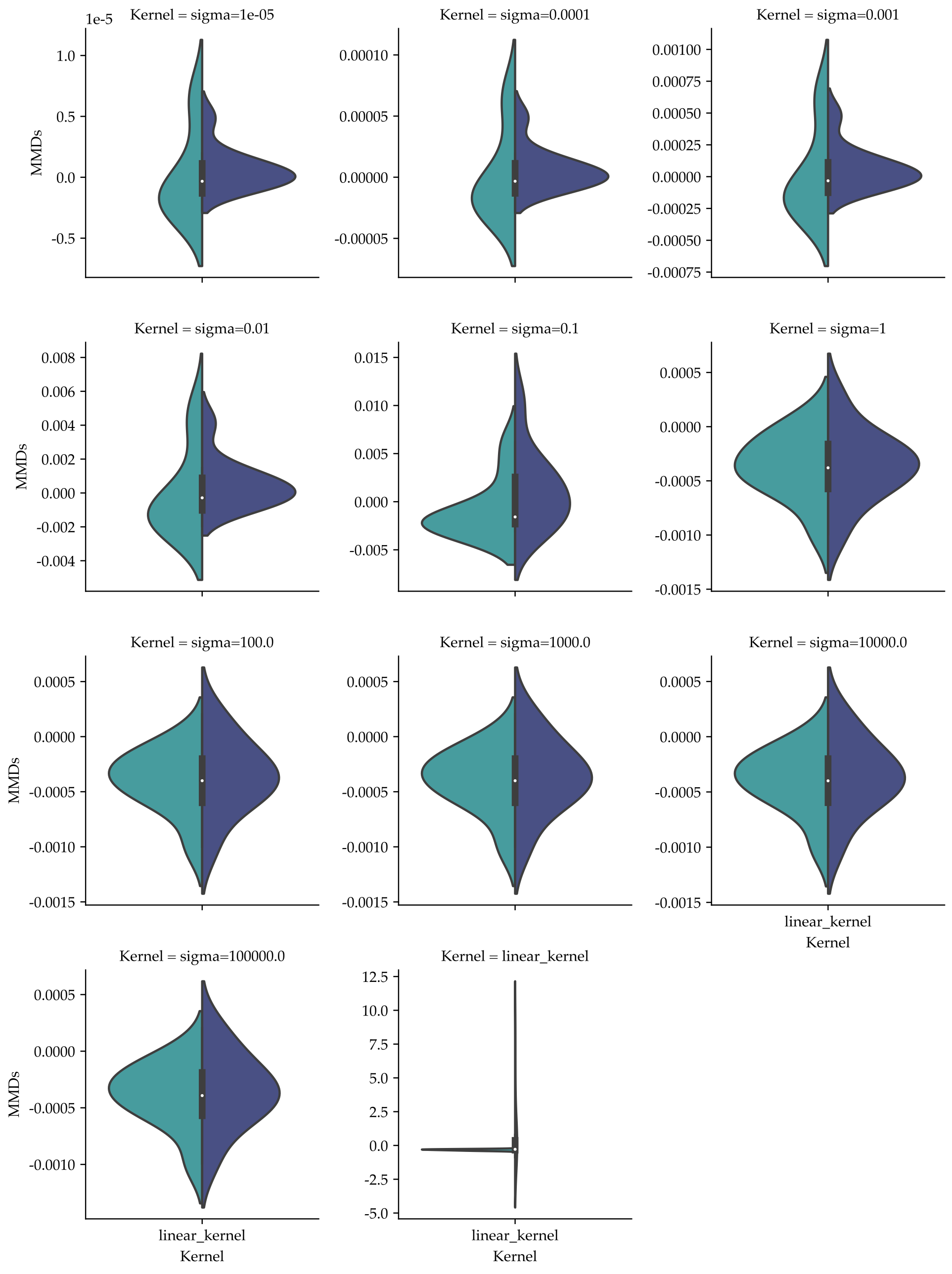


Figure A.3: Clustering coefficient MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$. Everywhere a plot contains "sigma" in the title, the RBF kernel with the indicated parameter was used.

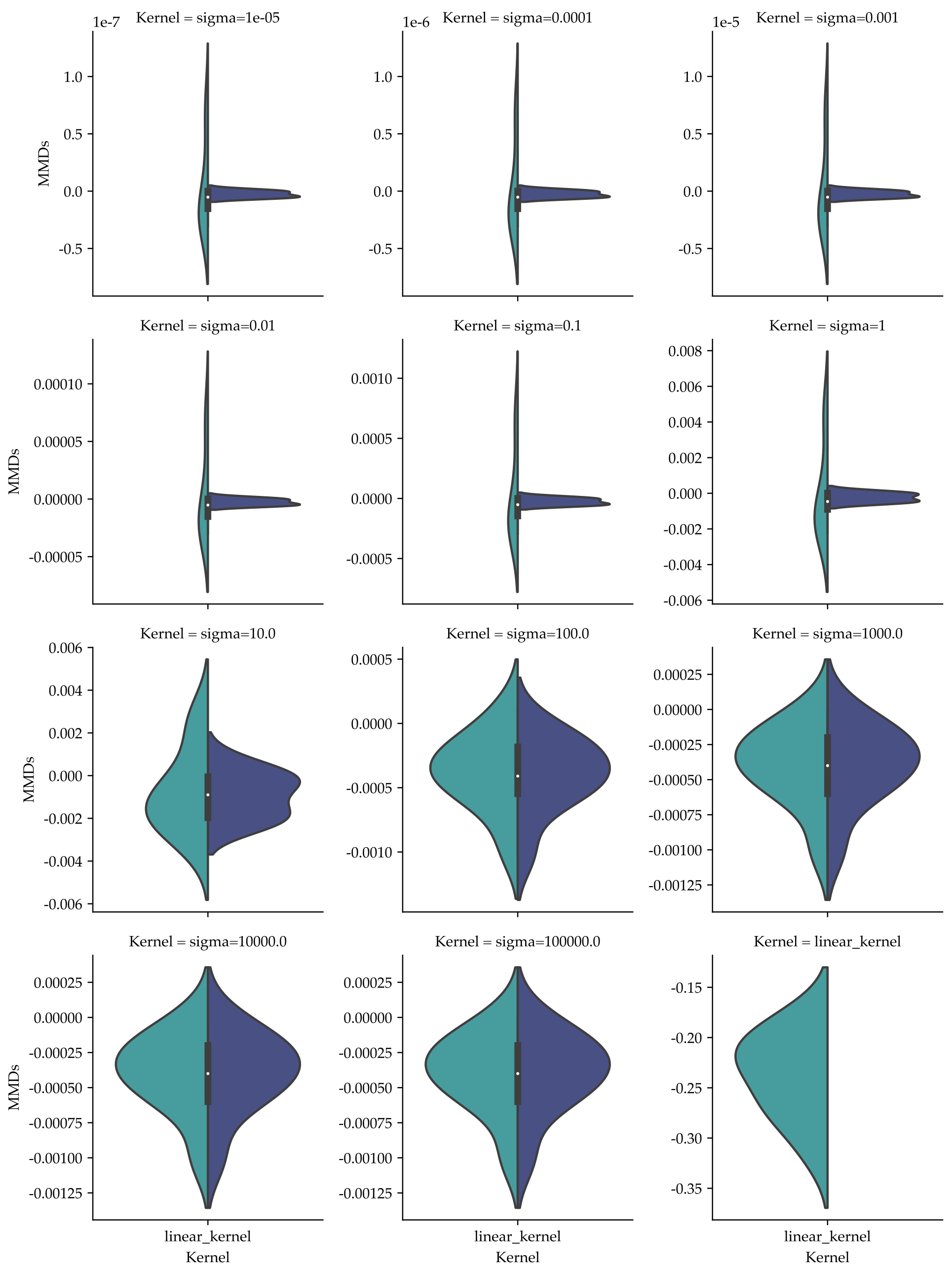


Figure A.4: Degree distribution histogram MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$. Everywhere a plot contains "sigma" in the title, the RBF kernel with the indicated parameter was used.

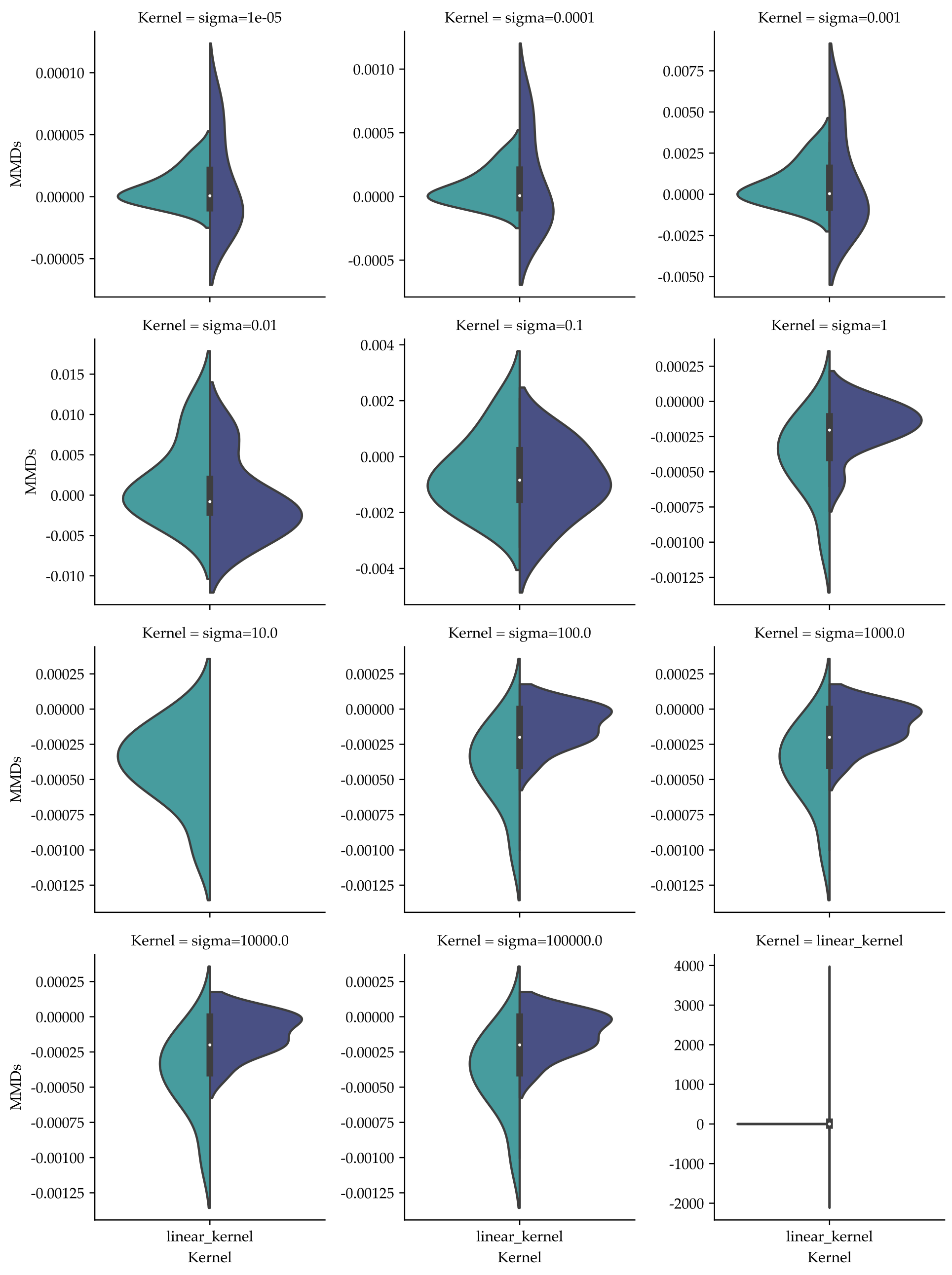


Figure A.5: Laplacian spectrum histogram MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$.Everywhere a plot contains "sigma" in the title, the RBF kernel with the indicated parameter was used.

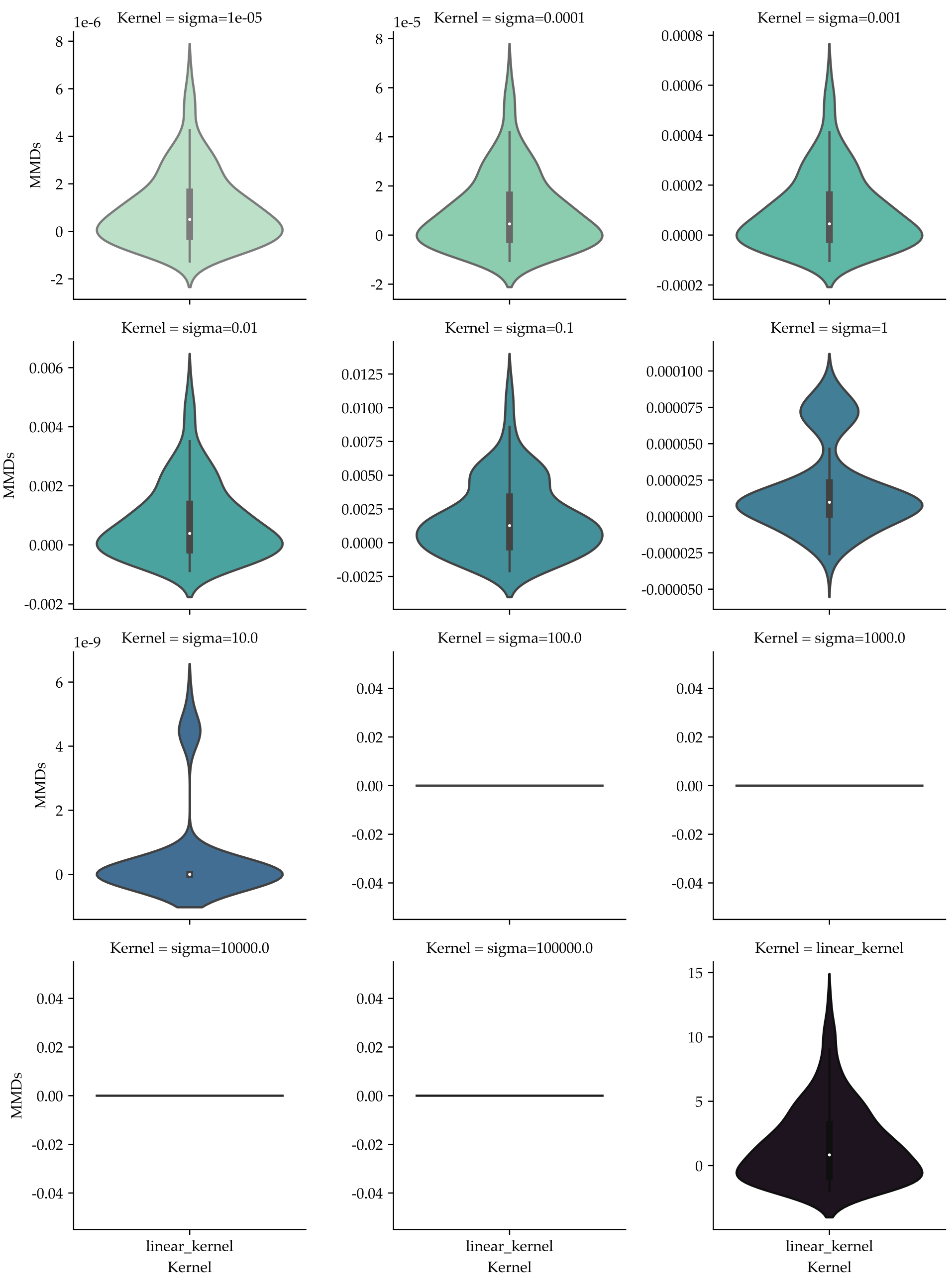


Figure A.6: ESM-based MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$. Everywhere a plot contains "sigma" in the title, the RBF kernel with the indicated parameter was used.

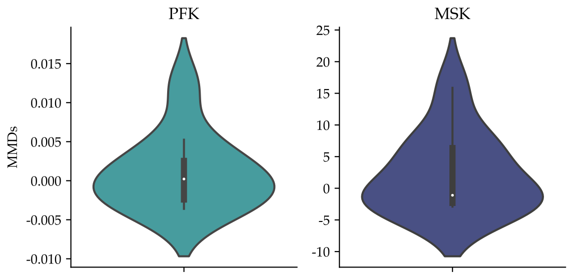


Figure A.7: TDA-based MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$. PFK: persistence Fisher kernel. MSK: multi-scale kernel.

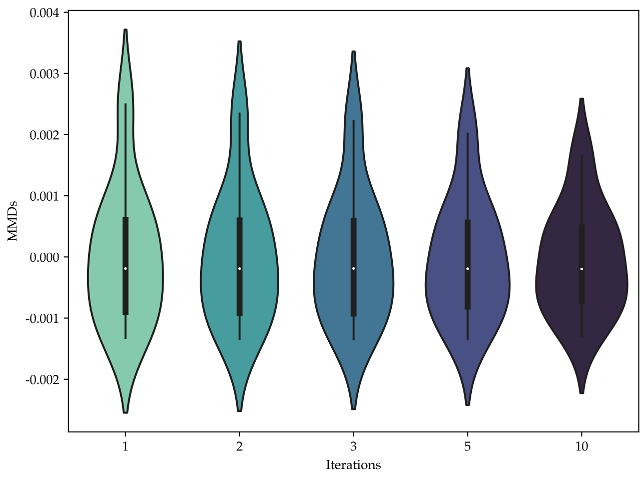


Figure A.8: TDA-based MMD baselines for two different ε values. The left side of each violin plot is obtained with $\varepsilon = 8$ and the right side with $\varepsilon = 32$. PFK: persistence Fisher kernel. MSK: multi-scale kernel. The different iterations correspond to the different iterations of the Weisfeiler-Lehman algorithm.

Bibliography

Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, and Lori Ziegelmeier. Persistence images: A stable vector representation of persistent homology. *The Journal of Machine Learning Research*, 18(1): 218–252, 2017.

Henry Adams, Ethan Coldren, and Sean Willmot. The Persistent Homology of Cyclic Graphs. *arXiv preprint arXiv:1812.03374*, 2018.

Jay Alammar. The Illustrated Transformer. https://jalammar.github.io/illustrated-transformer/, 2018.

Erik J Amézquita, Michelle Y Quigley, Tim Ophelders, Elizabeth Munch, and Daniel H Chitwood. The shape of things to come: Topological data analysis and biology, from molecules to organisms. *Developmental Dynamics*, 2020.

Namrata Anand and Possu Huang. Generative Modeling for Protein Structures. In *Proceedings of the 31st Conference on Advances in Neural Information Processing Systems*, 2018.

Rushil Anirudh, Vinay Venkataraman, Karthikeyan Natesan Ramamurthy, and Pavan Turaga. A Riemannian Framework for Statistical Analysis of Topological Persistence Diagrams. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 68–76, 2016.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. In *Proceedings of the International Conference on Machine Learning*, 2017.

Ulrich Bauer. Ripser: efficient computation of Vietoris-Rips persistence barcodes. *Journal of Applied and Computational Topology*, 2021. doi: 10.1007/s41468-021-00071-5.

Leonard E Baum and George Sell. Growth transformations for functions on manifolds. *Pacific Journal of Mathematics*, 27(2):211–227, 1968.

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1): 164–171, 1970.

Yonatan Belinkov and James Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019.

Eyal Betzalel, Coby Penso, Aviv Navon, and Ethan Fetaya. A Study on the Evaluation of Generative Models. *arXiv preprint arXiv:2206.10935*, 2022.

Surojit Biswas, Grigory Khimulya, Ethan C Alley, Kevin M Esvelt, and George M Church. Low-N protein engineering with data-efficient deep learning. *Nature Methods*, 18(4):389–396, 2021.

Sam Bond-Taylor, Adam Leach, Yang Long, and Chris George Willcocks. Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 2021.

Karsten Borgwardt, Elisabetta Ghisu, Felipe Llinares-López, Leslie O'Bray, and Bastian Alexander Rieck. Graph kernels: State-of-the-art and future challenges. *Foundations and Trends in Machine Learning*, 13(5-6):531–712, 2020.

Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by Kernel Maximum Mean Discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Nicola De Cao and Thomas Kipf. MolGAN: An implicit generative model for small molecular graphs. *Proceedings of the International Conference on Machine Learning Workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018.

Tamal K Dey, Herbert Edelsbrunner, and Sumanta Guha. Computational Topology. *Contemporary Mathematics*, 223:109–144, 1999.

Christopher C Drovandi and Anthony N Pettitt. Likelihood-free Bayesian estimation of multivariate quantile distributions. *Computational Statistics & Data Analysis*, 55(9):2541–2556, 2011.

Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

Michael P Fay and Michael A Proschan. Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. *Statistics surveys*, 4:1, 2010.

Daniel Freedman and Chao Chen. Algebraic Topology for Computer Vision. *Computer Vision*, pages 239–268, 2009.

Robert Ghrist. Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Proceedings of the 27th Conference on Advances in Neural Information Processing Systems*, 2014.

Swanand Gore, Sameer Velankar, and Gerard J Kleywegt. Implementing an X-ray validation pipeline for the Protein Data Bank. *Acta Crystallographica Section D: Biological Crystallography*, 68(4):478–483, 2012.

Swanand Gore, Eduardo Sanz García, Pieter MS Hendrickx, Aleksandras Gutmanas, John D Westbrook, Huanwang Yang, Zukang Feng, Kumaran Baskaran, John M Berrisford, Brian P Hudson, et al. Validation of structures in the Protein Data Bank. *Structure*, 25(12):1916–1927, 2017.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Aditya Grover, Aaron Zweig, and Stefano Ermon. Graphite: Iterative generative modeling of graphs. In *Proceedings of the International Conference on Machine Learning*, pages 2434–2444. PMLR, 2019.

Xiaojie Guo and Liang Zhao. A Systematic Survey on Deep Generative Models for Graph Generation. *arXiv preprint arXiv:2007.06686*, 2020.

Daniel Hesslow, Niccoló Zanichelli, Pascal Notin, Iacopo Poli, and Debora Marks. RITA: a Study on Scaling Up Generative Protein Sequence Models. *arXiv preprint arXiv:2205.05789*, 2022.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6629–6640, 2017.

Viktor Hornak, Robert Abel, Asim Okur, Bentley Strockbine, Adrian Roitberg, and Carlos Simmerling. Comparison of multiple Amber force fields and development of improved protein backbone parameters. *Proteins: Structure, Function, and Bioinformatics*, 65(3):712–725, 2006.

Po-Ssu Huang, Yih-En Andrew Ban, Florian Richter, Ingemar Andre, Robert Vernon, William R Schief, and David Baker. RosettaRemodel: a generalized framework for flexible backbone protein design. *PloS one*, 6(8):e24109, 2011.

Jenq-Neng Hwang, Shyh-Rong Lay, and Alan Lippman. Nonparametric multivariate density estimation: a comparative study. *IEEE Transactions on Signal Processing*, 42(10):2795–2810, 1994.

John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative Models for Graph-Based Protein Design. In *Proceedings of the 32nd Conference on Advances in Neural Information Processing Systems*, 2019.

Michael Jendrusch, Jan O Korbel, and S Kashif Sadiq. AlphaDesign: A de novo protein design framework based on AlphaFold. *bioRxiv*, 2021.

Yi Jiang, Dong Chen, Xin Chen, Li Tangyi, Wei Guo-Wei, and Feng Pan. Topological representations of crystalline compounds for the machine-learning prediction of materials properties. *NPJ Computational Materials*, 7 (1), 2021.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of the International Conference on Learning Representations*, 2013.

Thomas N Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the International Conference on Learning Representations*, 2016.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

Tim Kucera, Matteo Togninalli, and Laetitia Meng-Papaxanthos. Conditional generative modeling for de novo protein design with hierarchical functions. *Bioinformatics*, 38(13):3454–3461, 2022.

Tam Le and Makoto Yamada. Persistence Fisher Kernel: A Riemannian Manifold Kernel for Persistence Diagrams. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10028–10039, 2018.

Julia Koehler Leman, Brian D Weitzner, Steven M Lewis, Jared Adolf-Bryfogle, Nawsad Alam, Rebecca F Alford, Melanie Aprahamian, David Baker, Kyle A Barlow, Patrick Barth, et al. Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nature methods*, 17(7): 665–680, 2020.

C Leslie, E Eskin, and WS Noble. The Spectrum Kernel: A String Kernel for SVM Protein Classification. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 564–575, 2002.

Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. Learning Deep Generative Models of Graphs. *arXiv preprint arXiv:1803.03324*, 2018.

Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, William L Hamilton, David Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient Graph Generation with Graph Recurrent Attention Networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 4255–4265, 2019.

Romain Lopez, Adam Gayoso, and Nir Yosef. Enhancing scientific discoveries in molecular biology with deep generative models. *Molecular Systems Biology*, 16(9):e9198, 2020.

Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. ProGen: Language Modeling for Protein Generation. *arXiv preprint arXiv:2004.03497*, 2020.

Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Deep neural language modeling enables functional protein generation across families. *bioRxiv*, 2021.

Venkata Subramaniya Maddhuri, Raghavendra Sai, Genki Terashi, Aashish Jain, Yuki Kagaya, and Daisuke Kihara. Protein contact map refinement for improving structure prediction using generative adversarial networks. *Bioinformatics*, 37(19):3168–3174, 2021.

Elie Mengin. Weisfeiler-Lehman Graph Kernel for Binary Function Analysis. https://blog.quarkslab.com/weisfeiler-lehman-graph-kernel-for-binary-function-analysis.html, 2019.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 4602–4609, 2019.

Alexey G Murzin, Steven E Brenner, Tim Hubbard, and Cyrus Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *Journal of molecular biology*, 247(4):536–540, 1995.

Tapaswini Nayak, Lingaraja Jena, Pranita Waghmare, Bhaskar C Harinath, et al. Identification of potential inhibitors for mycobacterial uridine diphosphogalactofuranose-galactopyranose mutase enzyme: A novel drug target through in silico approach. *International Journal of Mycobacteriology*, 7(1):61, 2018.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation Invariant Graph Generation via Score-Based Generative Modeling. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 4474–4484. PMLR, 2020.

Leslie O'Bray, Max Horn, Bastian Rieck, and Karsten Borgwardt. Evaluation metrics for graph generative models: Problems, pitfalls, and practical solutions. In *Proceedings of the International Conference on Learning Representations*, 2022.

CM O'Connor and JU Adams. 2.4 The Functions of Proteins Are Determined by Their Three-Dimensional Structures, in The Essentials of Cell Biology. *Cambridge, MA: NPG Education*, 2010.

Christine A Orengo, Alex D Michie, Susan Jones, David T Jones, Mark B Swindells, and Janet M Thornton. CATH–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.

Henri Poincaré. *Analysis Situs*. Gauthier-Villars, 1895.

Krishna Mohan Poluri and Khushboo Gulati. Protein Engineering Techniques: Gateways to Synthetic Protein Universe. *SpringerBriefs in Forensic and Medical Bioinformatics*, 2017.

Kristina Preuer, Philipp Renz, Thomas Unterthiner, Sepp Hochreiter, and Günter Klambauer. Fréchet ChemNet Distance: A Metric for Generative Models for Molecules in Drug Discovery. *Journal of Chemical Information and Modeling*, 58(9):1736–1741, 2018.

G.N. Ramachandran, C. Ramakrishnan, and V. Sasisekharan. Stereochemistry of Polypeptide Chain Configurations. *Journal of Molecular Biology*, 7(1):95–99, 1963. ISSN 0022-2836. doi: https://doi.org/10.1016/S0022-2836(63)80023-6.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:1803.03324*, 2022.

Glen D Rayner and Helen L MacGillivray. Numerical maximum likelihood estimation for the g-and-k and generalized g-and-h distributions. *Statistics and Computing*, 12(1):57–75, 2002.

Randy J Read, Paul D Adams, W Bryan Arendall, Axel T Brunger, Paul Emsley, Robbie P Joosten, Gerard J Kleywegt, Eugene B Krissinel, Thomas Lütteke, Zbyszek Otwinowski, et al. A New Generation of Crystallographic Validation Tools for the Protein Data Bank. *Structure*, 10(19):1395–1412, 2011.

Jan Reininghaus, Stefan Huber, Ulrich Bauer, and Roland Kwitt. A Stable Multi-Scale Kernel for Topological Machine Learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pages 4741–4748. IEEE Computer Society, 2015.

Donatas Repecka, Vykintas Jauniskis, Laurynas Karpus, Elzbieta Rembeza, Irmantas Rokaitis, Jan Zrimec, Simona Poviloniene, Audrius Laurynenas, Sandra Viknander, Wissam Abuajwa, et al. Expanding functional protein sequence spaces using generative adversarial networks. *Nature Machine Intelligence*, 3(4):324–333, 2021.

Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature Methods*, 15(10):816–822, 2018.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *Proceedings of the 29th Conference on Advances in Neural Information Processing Systems*, 2016.

Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets. *arXiv preprint arXiv:2202.00273*, 2022.

Allen J Schwenk. Almost All Trees Are Cospectral. *New Directions in the Theory of Graphs*, pages 275–307, 1973.

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-Lehman Graph Kernels. *Journal of Machine Learning Research*, 12:2539–2561, 2011.

Alexey Strokach and Philip M Kim. Deep generative modeling for protein design. *Current Opinion in Structural Biology*, 72:226–236, 2022.

Alexey Strokach, David Becerra, Carles Corbi-Verge, Albert Perez-Riba, and Philip M Kim. Fast and Flexible Protein Design Using Deep Graph Neural Networks. *Cell Systems*, 11(4):402–411, 2020.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

L Theis, A van den Oord, and M Bethge. A note on the evaluation of generative models. In *Proceedings of the International Conference on Learning Representations*, pages 1–10, 2016.

Rylee Thompson, Boris Knyazev, Elahe Ghalebi, Jungtaek Kim, and Graham W Taylor. On Evaluation Metrics for Graph Generative Models. In *Proceedings of the International Conference on Learning Representations*, 2022.

Kathryn Tunyasuvunakool, Jonas Adler, Zachary Wu, Tim Green, Michal Zielinski, Augustin Žídek, Alex Bridgland, Andrew Cowie, Clemens Meyer, Agata Laydon, et al. Highly accurate protein structure prediction for the human proteome. *Nature*, 596(7873):590–596, 2021.

Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. AlphaFold Protein Structure Database: Massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 30th IEEE conference on computer vision and pattern recognition workshops*, 2017.

Leopold Vietoris. Über den höheren Zusammenhang kompakter Räume und eine Klasse von zusammenhangstreuen Abbildungen. *Mathematische Annalen*, 97(1):454–472, 1927.

Yajie Wang, Pu Xue, Mingfeng Cao, Tianhao Yu, Stephan T Lane, and Huimin Zhao. Directed evolution: Methodologies and applications. *Chemical Reviews*, 121(20):12384–12444, 2021.

Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

Eli N Weinstein and Debora Marks. A structured observation distribution for generative biological sequence prediction and forecasting. In *International Conference on Machine Learning*, 2021.

Boris Weisfeiler and Andrei Lehman. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series*, 2(9):12–16, 1968.

Matthew J Wood and Jonathan D Hirst. Protein Secondary Structure Prediction with Dihedral Angles. *PROTEINS: Structure, Function, and Bioinformatics*, 59(3):476–481, 2005.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? In *Proceedings of the International Conference on Learning Representations*, 2018a.

Mengjia Xu. Understanding Graph Embedding Methods and Their Applications. *SIAM Review*, 63(4):825–853, 2021.

Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018b.

Sinan Yıldırım, Sumeetpal S Singh, Thomas Dean, and Ajay Jasra. Parameter Estimation in Hidden Markov Models with Intractable Likelihoods Using Sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 24(3):846–865, 2015.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. In *Proceedings of the International Conference on Machine Learning*, 2018.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

DESIGNING MEANINGFUL MEASURES TO EVALUATE PROTEIN GENERATIVE MODELS

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Hartout	Philip Jean

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, Switzerland. 11/7/22	[illegible]

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.