The intrinsic dimension of biological data landscapes

A Thesis submitted for the degree of
PhilosophiæDoctor.

Candidate:
Elena Facco

Supervisor:
Alessandro Laio

September, 14^{th} 2017

Ph.D. curriculum in Molecular and Statistical Biophysics

Table of Contents

Chapter 1

Outline

Over the past 20 years, data availability has exponentially increased in various fields. The term Big Data has been coined to refer to datasets that, due to their size and the number of features, pose important challenges in terms of acquisition, storage, and management (1). Generally, there is a high level of redundancy in datasets: redundancy reduction and data compression are key tools to make data more meaningful for computer analysis and user interpretation, provided that the "value", or relevant information, of the original data is preserved. Thus, when dealing with large datasets in high-dimensional spaces it is advantageous to devise some inner structure in the data. In this perspective a fortunate thing is that, most of the times, data generation is governed by a number of parameters much lower than the bare number of data features, or coordinates. The minimum number of parameters needed to accurately describe the important characteristics of a system is called the *Intrinsic Dimension* (ID) (2) of the dataset.

Due to the relevance of this number, many techniques have been developed to estimate it; the main practical application of these methods is effective projection of the data in a lower dimensional space ((3)). From this point of view, the ID is an *auxiliary* value, that allows to select the number of independent directions along which to project the data. In this Thesis we develop a new method to estimate the intrinsic dimension that relies on minimal points neighborhoods only, thus overcoming some common problems related to density variations and curvatures in the datasets at study (see Chapter 2). Moreover, we pair for the first time the use of ID estimation to that of probability density estimation (see Chapter 4), claiming that an appropriate estimation of the probability has to take into account the dimension of the manifold containing the dataset rather than that of the embedding space.

But, besides being used as an auxiliary information and preliminary step for data analysis techniques, we demonstrate that the ID can be informative by itself, un-

raveling the structure of complex datasets and allowing for new interpretations of natural phenomena.

In Chapter 2 we introduce the concept of Intrinsic Dimension (ID), that can be intuitively defined as the minimum number of parameters needed to describe the salient characteristics of a system (2). Estimating the ID of a dataset poses some important difficulties. First, high-dimensional spaces behave in an extremely counterintuitive way due to the so called "curse of dimensionality": in high dimensions the smallest sampled distance between points increases, and nearly all of the space is spread "far away" from every point (see (4)). Second, when dealing with a dataset a fundamental question is about the *scale* we are interested in. In real datasets a good choice for the scale is fundamental, as some dimensions have a leading importance from the point of view of data interpretation; some of them identify directions in which the features of the dataset change remarkably, while others are characterized by small variations that can be irrelevant for the analysis and can be labeled as "noise". For example, for a sample of configurations explored during a molecular dynamics run at finite temperature of a complex biomolecule, a practically meaningful estimate of the ID should provide the number of "soft" directions, associated to important conformational changes. A third difficulty is ubiquitous in real world datasets: complex datasets can lie on twisted manifolds and are usually characterized by a varying density. Such characteristics do not affect in principle the local topology of the manifold and therefore an intrinsic dimension estimator should be able to provide an ID as independent as possible of these features. Finally, to be practically suitable in the case of large datasets, a method should allow for a fast implementation. Taking into account all these issues, in Chapter 2 we develop a new ID estimator, able to provide an ID measure by employing only the distances from the first TWO Nearest Neighbors of every point in the dataset, hence the name TWO-NN. Thanks to the use of a minimal neighborhood information, TWO-NN limits the influence of curvatures and density variations; moreover, if paired with block analysis, it allows for a multi scale study of the dataset that results particularly useful to capture the meaningful length scale on which the ID should be defined. We apply TWO-NN to the study of the dimension of image datasets, as well as to the analysis of a set of configurations generated in a finite temperature molecular dynamics trajectory of a biomolecule in water solution. The ID in the latter case was computed making use of two intrinsically different notions of distance, namely the Euclidean distance between the coordinates associated to each sample by Time-lagged Independent Component

Analysis(TICA)(5) and the Root Mean Square Deviation (RMSD) between all the atoms in the trinucleotide. Noticeably, as shown in Figure 2.11, the scaling features of the ID d versus the sample size N with the two metrics are comparable. The estimated ID values are ~ 9 for both the metrics.

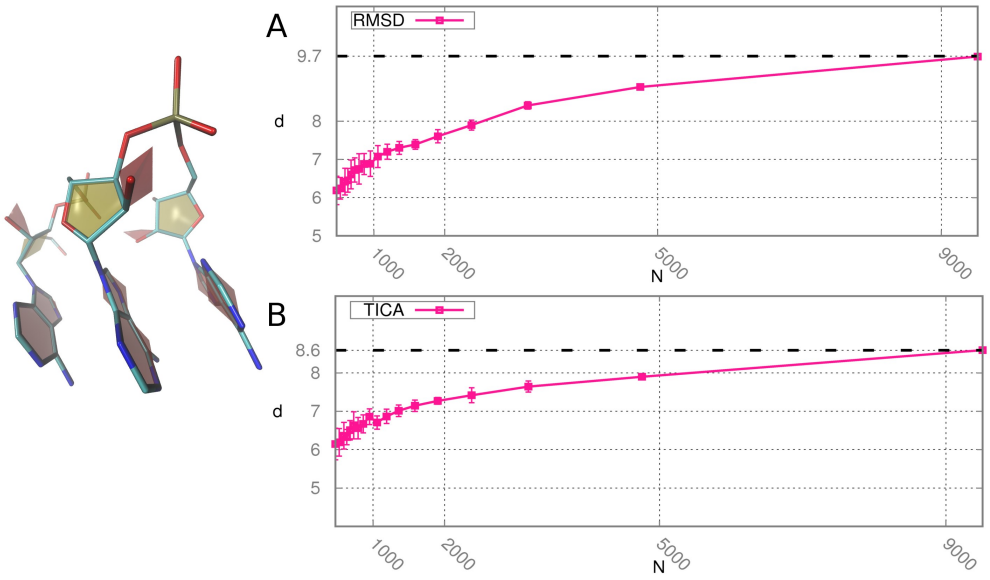

Figure 1.1: *Scaling of the estimated ID with respect to the number of configurations for the dynamics of trinucleotide AAA in the case of RMSD distances (panel A) and TICA distances (panel B). On the left a configuration of the system.*

In Chapter 3 we study the intrinsic dimension of samples of protein sequences belonging to the same families. A key point here is to properly define a *distance* between data points, namely the protein sequences. In fact, in this Chapter we show that different metrics that can be defined in this context do not lead to the same measure of intrinsic dimension; indeed, the distances between protein sequences *shape* the space and its topological properties. We point out the features that a distance between protein sequences should have to be suitable for our purposes: (i) being a proper metrics (we show that many dissimilarity measures used in bioinformatics do not satisfy the triangular inequality), (ii) depending only on the two sequences it is computed on (thus we exclude distances deriving from multiple sequence alignments) and (iii) being computationally fast to compute. Taking into consideration these issues, we develop a notion of distance that we call Modified Hamming Distance, in symbols d_{MH}, that is a fast version of the well established Edit distance ((6)). This part of the work was realized in collaboration with Elena Tea Russo. By using d_{MH} we are able to compute the dimensions of several Pfam protein families,

discovering that their IDs range approximately from 6 to 12. These values do not depend on the average length of the sequences, but are weakly correlated to the number of architectures present in the family. We speculate that the low ID of the manifold containing the sequences could possibly be interpreted in terms of allowance for mutations: the evolutionary pressure results in a lack of variations at specific positions and in correlated variations across different positions, both restricting the number of degrees of freedom of the sequences. Further research will be performed in this direction.

In Chapter 2, finally, we study the reliability of artificial generative models for protein sequences from the point of view of the intrinsic dimension. This analysis was performed in collaboration with Andrea Pagnani, from Politecnico di Torino. The ID is a complex function of the data, but a meaningful one, and we suggest that it can be employed to assess the goodness of artificial models. In particular, if the ID of artificial sequences is higher than in the natural ones, it can be argued that more constraints are to be imposed on the model; thus ID computation can help not only to establish the invalidity of a model, but to suggest a direction to fix it.

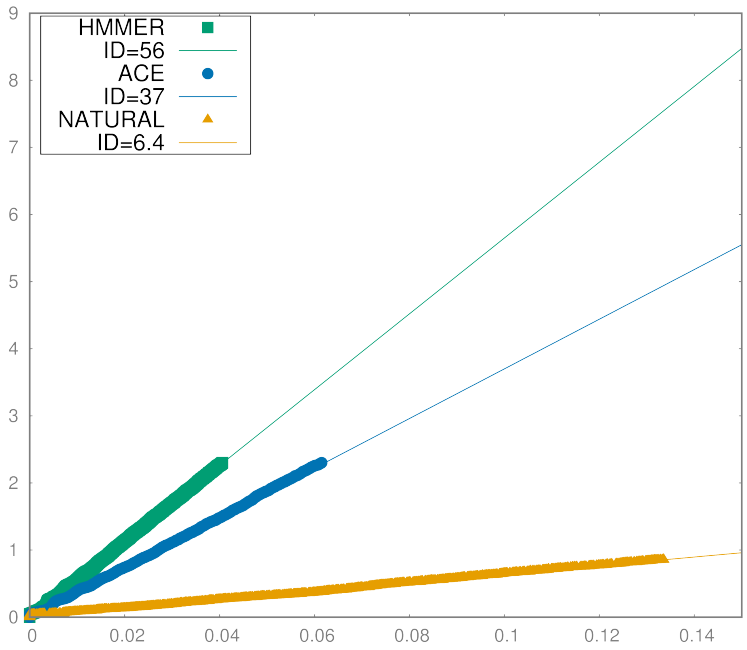

Figure 1.2: *S set and fitting lines for sequences generated with HMMER (7), ACE (8) and for natural ones in the case of Pfam family PF00076.*

In Figure 3.21 we show the ID of sequences generated with HMMER (7), a method

that takes into account only site-specific statistics, and ACE (8), a method that describes accurately the joint probability of pair of mutations in different sites, and compare it to the ID of natural sequences for a specific Pfam family. The ID of artificially generated sequences is significantly higher than the ID of natural ones, suggesting that none of the methods is optimal to design artficial protein sequences; to this task, probably, pairwise couplings and local fields are not sufficient, and a more complex model has to be considered.

In Chapter 3 we demonstrate how the ID can be used in an informative way, to investigate the properties of protein families datasets; in Chapter 4 the ID is employed in an auxiliary way, to describe the dimension of the minimal manifold containing the dataset in order to be able to perform density estimation, and clustering, on this space. This research was performed in collaboration with Alex Rodriguez and Maria d'Errico.
The specific application we work on is also related to protein sequences. In particular, we introduce a protocol to assign sequences to families in an automatic fashion. The idea is that families correspond to density peaks in the distribution of protein sequences in the space described by d_{MH} distances. To this purpose, we need three fundamental tools. The first one is an intrinsic dimension estimator providing the ID of the space in which the density has to be computed: we use the TWO-NN method developed in Chapter 2 together with the procedure to determine the ID of protein families developed in Chapter 3. The second one is an estimator for the density of the distribution of sequences: in this Chapter we outline a novel parameter-free density estimator able to work even in the case of high intrinsic dimensions and to additionally provide an error on the density measure. Finally, the third one is a way to automatically group sequences according to their density, namely a clustering algorithm: in this Chapter we describe a new clustering algorithm that extends Density Peaks clustering (DP)(9), and is able to deal with a hierarchical organization of data.
The collection of techniques illustrated so far possesses all the features to analyze complex landscapes characterized by relatively high intrinsic dimensions and significant density inhomogeneities, such as Pfam protein clans. We apply the procedure to the Pfam clan PUA: not only the method is able to successfully partition domains into families, but it can detect a further level of complexity, namely that of domain architectures, as illustrated in Figure 4.4.
This result is striking if one takes into account that Pfam provides the sequences

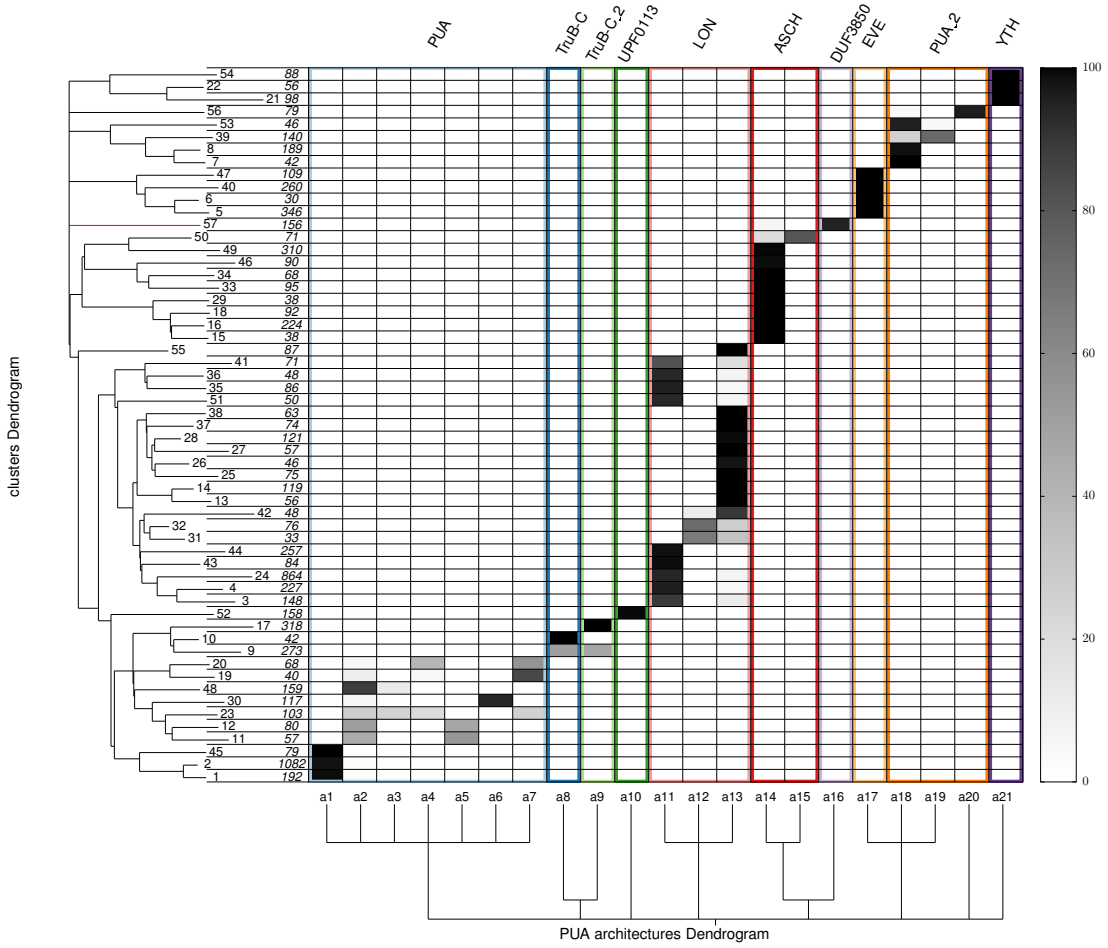

Figure 1.3: *Cluster analysis of the clan PUA from the Pfam database. We represent the Purity Matrix for clusters and architectures with a population greater than 40. Color boxes correspond to families. The purity of the clusters with respect to architectures is associated to a grey palette: the darker the cell, the higher the purity. On the left of the Purity Matrix we show the dendrogram of the clusters. In correspondence of each leaf we indicate the cluster label and the population of the cluster. The Dendrogram at the bottom is a schematic visualization of the hierarchical relation existing between architectures according to Pfam: architectures connected at a higher level (e.g. a1, a2, a3) belong to the same family, while those connected at a lower level (e.g. a12, a13) belong to families that are closely related according to the Pfam definition.*

of the *single domains* only, so that architectures composed by many domains are 'deduced' by the method from the sequence of just one of them. The clustering procedure thus proves to be able to detect the complexity of a structure by the interaction of the rest of the sequence with the single domain at disposal.

As we illustrate in Chapters 3 and 4, the scope of applicability of TWO-NN is wide; however, it can be employed to meaningfully estimate the ID only if an important assumption is fulfilled: the ID must be one and well defined along the dataset. In many cases though the data can be reasonably conceived as being sampled from a *finite number of manifolds, each with its own well-defined dimension*, rather than from a single one, as assumed in the TWO-NN model. Thanks to its simplicity, the TWO-NN model proves ductile and allows an extension that enables to cope with these cases.

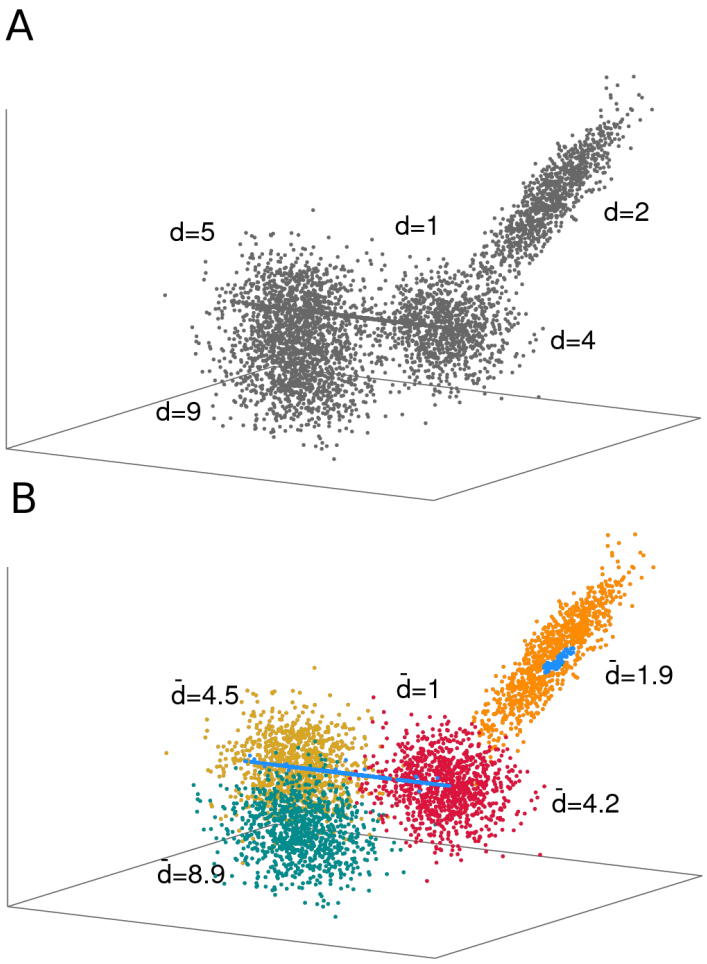

Figure 1.4: *A. A dataset consisting of* 5 *Gaussians with unitary variance in dimensions* $1, 2, 4, 5, 9$. *B. Final assignation of points to the respective manifolds together with the estimated dimensions, denoted by* $\bar{d}$.

In Chapter 5 we introduce a Bayesian technique that allows inferring the dimensions

of the manifolds, their size, and which points are sampled from which manifold. We choose to work within a Bayesian estimation framework because it allows to incorporate any prior expectation one may have about the quantities of interest, for instance the number, size, and dimension of the manifolds, the size of their intersection, and so on. We show that such prior information is essential to obtain a reliable estimation in the most challenging datasets. We test the method on artificial datasets, as for instance the dataset in Figure 5.7, consisting of 5 Gaussians in different IDs. The method is able to assign points to the respective manifold with a high degree of accuracy.

This research was performed in collaboration with Michele Allegra and Antonietta Mira from USI, Universita' degli Studi dell'Insubria.

As an application to real datasets, we consider the dynamics of the villin headpiece (PDB entry: 2F4K) (10). In this case, TWO-NN model fails at properly describing the data, suggesting that more than one manifolds with different IDs coexist in the dataset. This feature could be related to the transition of the system between the folded and unfolded state. The method reveals indeed two manifolds M_1 and M_2 with dimensions $d_1 \sim 23$, $d_2 \sim 10$ respectively, and we show that configurations belonging to the low dimensional manifold are almost for sure unfolded. We further apply the approach to analyze a series of volumetric images of a human brain obtained through functional magnetic resonance imaging (fMRI). Also in this case, the method reveals two manifolds M_1 and M_2 with dimensions $d_1 \sim 12$, $d_2 \sim 37$; the signals giving rise to coherent patterns mostly belong to the lower dimensional manifold, meaning that they exhibit a considerably reduced variability than the remainder of the signals that have a noisy and incoherent behavior. These are remarkable results, since the intrinsic dimension is a purely topological observable, unaware of any chemical or physiological detail.

Chapter 2

Estimating the intrinsic dimension by a minimal neighborhood information

2.1 Introduction

When dealing with high-dimensional data sets, it can be advantageous to devise some inner structure in the data. Indeed, data generation is governed by a certain number of parameters, lower than the bare number of data features, or coordinates. The minimum number of parameters needed to accurately describe the important characteristics of a system is called the *Intrinsic Dimension* (ID) (2) of the dataset. An intuitive definition of intrinsic dimension is provided by Bishop in (11): a set in D dimensions has an ID equal to d if the data lies within a d-dimensional subspace of $\mathbb{R}^D$ entirely, without information loss.
Before going into detail about ID definitions and estimators, it is important to point out that the properties of high dimensional spaces can be extremely counterintuitive. As the intrinsic dimension of a dataset gets higher, the smallest sampled distance between points increases, and nearly all the space is spread "far away" from every point: this phenomenon is known as the "curse of dimensionality". To illustrate it one can consider the case of a hypersphere with radius r inscribed in a hypercube with edge $2r$; in dimension d the volume of such a sphere, say $V_1(d)$, is $\frac{2r^d\pi^{\frac{d}{2}}}{d\Gamma(\frac{d}{2})}$ (here Γ is the gamma function) while the volume of the cube $V_2(d)$ is $(2r)^d$, so that the fraction $\frac{V_1(d)}{V_2(d)}$ goes to 0 as d increases to infinity. The high-dimensional hypercube consists almost entirely of its "corners".

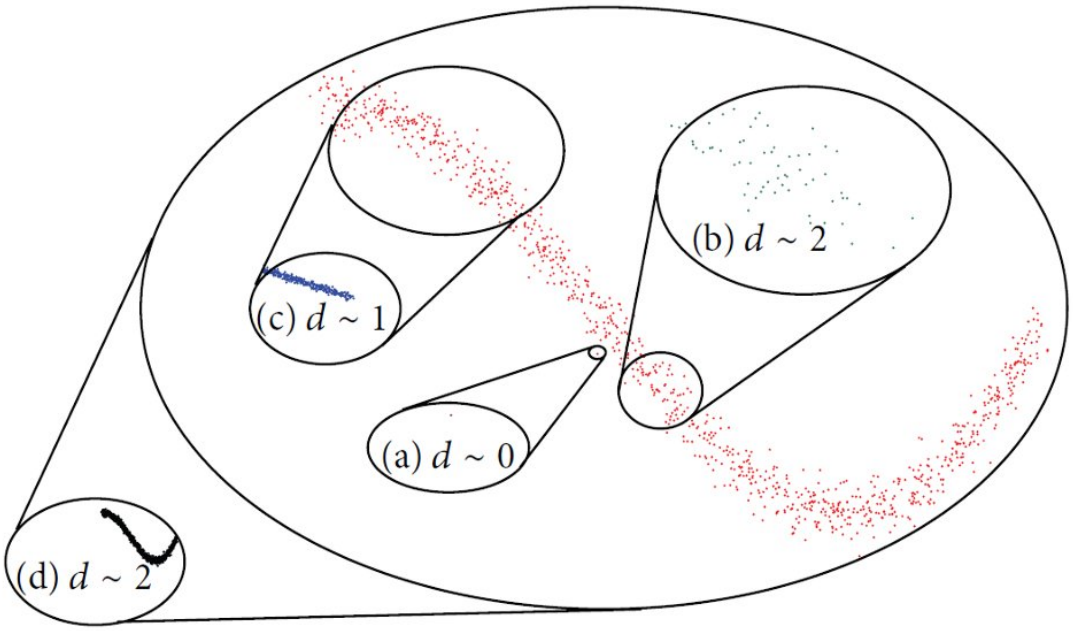

Figure 2.1: *If the scale of interest is minimal (a) the dataset has intrinsic dimension* 0*. If we increase the scale, we encounter an intrinsic dimension of* 2 *(b),* 1 *(c), and again* 2 *for the entire dataset (d) [from ref (12)].*

When trying to estimate the intrinsic dimension of a dataset, first of all we should ask ourselves which is the scale we are interested in, as shown in Figure 2.1 (12). Here a curved line perturbed by noise is represented; at a wide scale the two-dimensional character of the dataset emerges (d); if we restrict the scale (c) the one-dimensional feature is predominant, as locally a one-dimensional curve can be mapped into $\mathbb{R}$. At a smaller scale (b) the noise is not negligible any more, and two independent directions are necessary to describe the dataset. The extreme case occurs when the scale is small enough to contain a point only: in this case the dimension is 0 (a).

In real datasets a good choice for the scale is fundamental, as some dimensions have a leading importance from the point of view of data interpretation; some of them identify directions in which the features of the dataset change remarkably, while others are characterized by small variations that can be irrelevant for the analysis and can be labeled as "noise". Consider for example a sample of configurations explored during a molecular dynamics run at finite temperature of a complex biomolecule. In the absence of constraints on the bond length, the intrinsic dimension of the

hypersurface explored in this dynamics is $3N$, where N is the number of atoms. A well-defined estimator should in principle provide this number when infinitely many configurations are sampled. However, this asymptotic estimate is clearly irrelevant for practical purposes. Indeed, most of the $3N$ possible directions are highly restrained, due for example to steric clashes between neighboring atoms, and those in which the system can move by a significant amount are normally much fewer. A practically meaningful estimate of the ID should provide the number of these *soft* directions.

The problem of the scale is amplified in the case of data lying on curved subspaces: taking again as an example Figure 2.1, if the dataset is embedded in a higher dimensional space through a nonlinear map, points that are far according to a geodesic distance on the manifold can be close from the point of view of Euclidean distance. This may result in an overestimation of the ID if the dimension of interest corresponds to the one of the space tangent to the manifold (case (b)).

2.2 Intrinsic dimension estimators

In recent years, a variety of techniques have been proposed to compute the intrinsic dimension of data (12), (13); in this section we review the most effective state-of-the-art ID estimators, that can be roughly grouped into four main categories; *projective methods* search for a subspace to project the data by minimizing a projection error. *Fractal methods* count the number of observations in a neighborhood of radius r and estimate its growth as r increases. *Graph based methods* take advantage of the well established theory of distances on graphs. Finally, *Nearest-Neighbors methods* usually assume that close points are uniformly drawn and describe data neighborhood distributions parametrized by the intrinsic dimension. In the following we describe more in detail the different typologies together with the most effective algorithms, highlighting their advantages and drawbacks.

Projective methods

Projective estimators can be divided into two main groups: variants of the Multidimensional Scaling (MDS) (14),(15), and techniques based on Principal Component Analysis (PCA) (3). The original purpose of MDS is to provide a visual representa-

tion of the pattern of proximities (similarities or distances) among a set of objects, but later extensions of the procedure to higher dimensions than two or three allow to employ it as an ID estimator. Given a dissimilarity matrix D_{ij} between points, the aim of MDS is to find the best projection on a lower dimensional space such that close points are mapped into close points; this is achieved by assigning to each projection a measure of goodness called *stress*. A possible stress function is given by

$$\mathcal{E} = \Big[\sum_{i<j} \frac{D_{ij} - \Delta_{ij}}{D_{ij}}\Big]\Big[\sum_{i<j} D_{ij}\Big]^{-1}, \tag{2.1}$$

where Δ_{ij} is the dissimilarity of the points once projected on the lower dimensional space. At this point the minimum stress for projections at different dimensionalities is computed, and the MDS guess for the ID is the dimension giving the lowest stress. Examples of MDS algorithms are MDSCAL (16), Sammon's mapping (17), ISOMAP (18), that employs geodesic distances between points to overcome the effects of curvature in the manifold containing the dataset, and Local Linear Embedding, in short LLE (19).

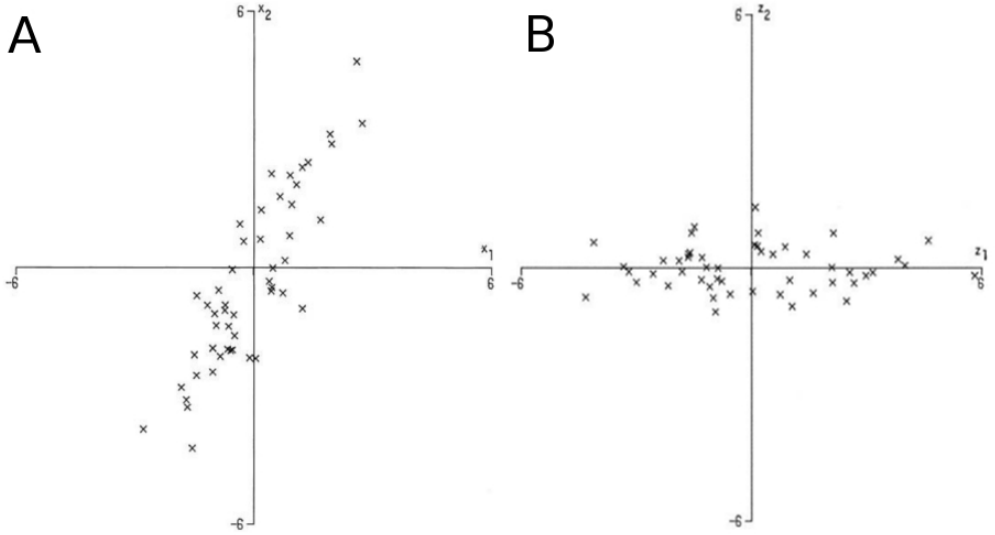

Figure 2.2: *Panel A: 50 observations in the variables x_1, x_2; Panel B: plot of the 50 observations with respect to their PCs z_1, z_2 [from ref. (3)].*

PCA (3) is one of the most well-known projective ID estimators; it is based on the computation of the N eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$ of the covariance matrix of the sample. Upon the choice of a threshold θ, the ID is determined by dividing each eigenvalue by the largest one λ_1 and chosing the integer d such that $\lambda_d \geq \theta$ and

$\lambda_{d+1} \leq \theta$. Thus, the ID estimate coincides with the number of the "most relevant" eigenvectors of the covariance matrix, also called principal components (PCs). PCA in its basic formulation has two critical drawbacks: it fails to estimate correctly the ID in the case of curved manifolds, and it relies on the choice of the parameter θ. In order to cope with the latter problem, Tipping and Bishop provide PCA with a probabilistic model, reformulating the method as the maximum likelihood solution of a latent variables model (PCCA,(20)). Extensions of PCCA are Bayesian PCA (BPCA, (21)), Simple Exponential family PCA (SePCA, (22)), and Sparse PCA (SPCA, (23)); all these methods are linear, thus not adequate in case of data points lying on curved manifolds; a variant of PCA that is able to deal with nonlinear manifolds is Kernel PCA (24). An alternative method is Multiscale SVD (MLSVD, (25)), that applies Singular Value Decomposition, basically a variant of PCA, in a local fashion so as to limit the effects of curvature.

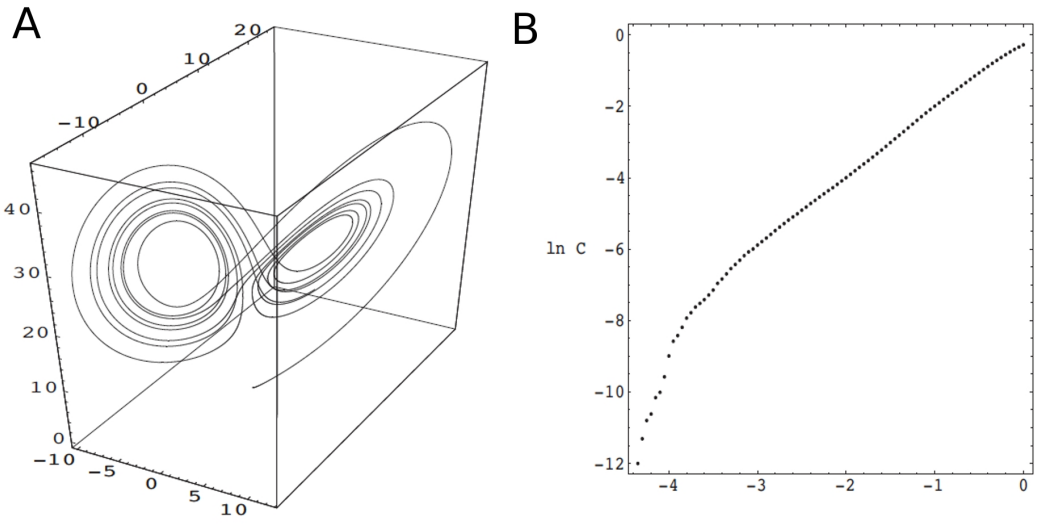

Figure 2.3: *Panel A: the attractor of the Lorents system; Panel B: the log-log plot of a dataset generated by a Lorents-like system (13).*

Fractal methods

The basic concept of fractal dimension is that the volume of a ball in d dimensions with radius r scales as r^d. So, it is possible to relate the number of observations in a neighborhood of radius r to the intrinsic dimension d by estimating the rate of growth of this number as r increases. One of the most cited estimators exploiting this idea is presented in (26); the method was originally developed to measure the

fractal dimension of strange attractors and is based on the concept of correlation dimension (dim_{Corr}). Given a sample $\mathbf{X}_N$, let

$$C_N(r) \doteq \frac{2}{N(N-1)} \sum_{i=1, i<j}^{N} \Theta\Big(r - \parallel \mathbf{x}_i - \mathbf{x}_j \parallel \Big), \tag{2.2}$$

where $\parallel \cdot \parallel$ is the Euclidean norm and $\Theta(\cdot)$ is the Heaviside step function. Basically, $C_N(r)$ counts the average number of distances lower than r between data points. Since in the limit $N \to \infty$ and $r \to 0$ it is expected that $C_N(r) \sim r^d$, where d is the intrinsic dimension of the dataset, the correlation dimension is defined as:

$$dim_{Corr} = \lim_{r \to 0} \lim_{N \to \infty} \frac{logC_N(r)}{log(r)}. \tag{2.3}$$

A practical way to compute the ID is then computing $C_N(r_i)$ for different values of r_i and applying least squares to fit a line trough the points $(log(r_i); log(C_N(r_i)))$: the slope of the line is the estimated ID.
A fundamental limitation of fractal methods is that to get accurate estimations the size N of the dataset with intrinsic dimension d has to satisfy the following inequality (27):

$$d < 2 \log_{10} N. \tag{2.4}$$

This means that the number of data points needed to estimate the ID of a d-dimensional dataset is at least $10^{\frac{d}{2}}$, therefore fractal-based algorithms are not feasible in case of high intrinsic dimension. To overcome this difficulty Camastra and Vinciarelli (28) propose to measure the underestimation induced by low sampling on datasets of known ID and use it to correct the negative bias. Another approach is adopted in (4), that pairs the study of fractal properties with the use of geodesic distances in order to handle nonlinear manifolds.

Graph-Based methods

Theories describing graphs can be exploited in many ways to compute the ID of datasets; Given a sample $\mathbf{X}_N = \{p_i\}_{i=1,\dots,N}$ a graph built on $\mathbf{X}_N$ is a set of points called vertices together with a set of edges connecting the vertices, formally $G(\mathbf{X}_N) = \Big(\{p_i\}_{i=1,\dots,N}, \{e_{i,j}\}_{i,j=1,\dots,N}\Big)$. To provide edges with proper weights a similarity measure has to be employed, usually the Euclidean distance. A k-Nearest

Neighbors Graph, in short $kNNG_N(\mathbf{X}_N)$, is a directed graph, meaning that the edges have a direction, connecting every point with its k nearest neighbors. A Minimum Spanning Tree ($MST(\mathbf{X}_N)$) is the spanning tree (meaning a tree that connects all the vertices) that minimizes the sum of the weights of the edges.
Costa-Hero's algorithm (29), (30) assumes that the dataset $\mathbf{X}_N = \{p_i\}_{i=1,\dots,N} \subset \mathbb{R}^{\mathbb{D}}$ lies on a smooth compact d-dimensional manifold and it is based on building the Euclidean neighborhood graph $\mathcal{G}$ over $\mathbf{X}_N$, and from it the $MST(\mathbf{X}_N)$. Let $\mathcal{L}_\gamma$ be the so called *Geodetic Minimum Spanning Tree Length (GMSTL)*, defined as $\mathcal{L}_\gamma(\mathbf{X}_N) \doteq \min_{T \in ST} \sum_{e \in T} w(e)^\gamma$, where ST is the set of spanning trees built on the graph $\mathcal{G}$, $w(e)$ indicates the weight of edge e and $\gamma \in (0, D)$ is the *edge exponent*. Costa and Hero derive an equation connecting the $GMSTL$ to the intrinsic dimension d of the dataset. If we define $\mathcal{L}_N \doteq \log \mathcal{L}_\gamma(\mathbf{X}_N)$, the following equation holds:

$$\mathcal{L}_N = a \log N + b + \epsilon_N, \tag{2.5}$$

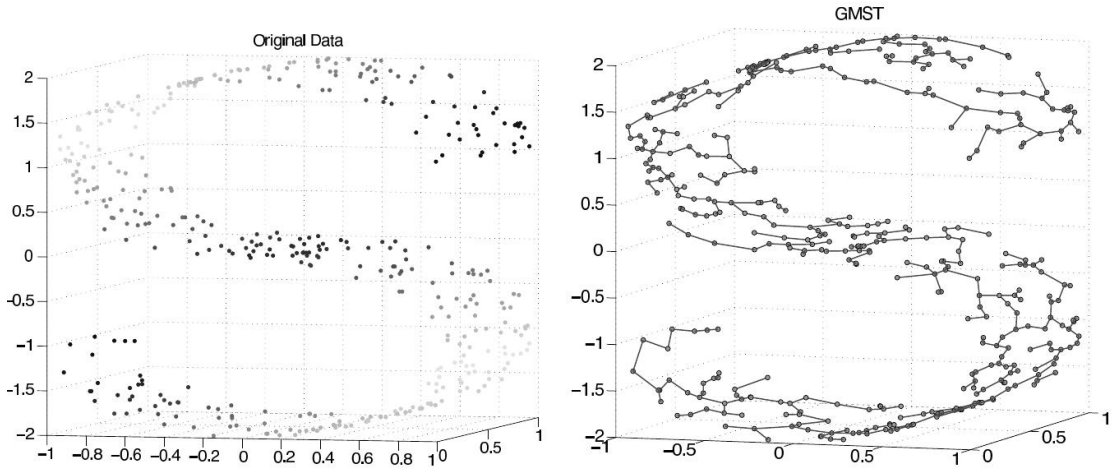

Figure 2.4: *Geodesic Minimum Spanning Tree built on* 400 *points from an S-shaped manifold (29).*

where $a = \frac{d-\gamma}{\gamma}$, and ϵ_N is an error residual that goes to 0 as $N \to \infty$ with probability 1. At this point one can collect datasets $\mathbf{X}_{N_i}$ each with a different cardinality N_i, obtained from $\mathbf{X}_N$ by bootstrap, and compute for each $\mathbf{X}_{N_i}$ the corresponding $\mathcal{L}_{N_i}$. The following step is to provide estimates of a and b, that we call $\hat{a}$ and $\hat{b}$ respectively, by regressing $\mathcal{L}_{N_i}$ vs N_i by a Linear Least Squares method. Finally, by

fixing exponent γ to one the estimate for the intrinsic dimension is given by

$$\hat{d} = \frac{1}{1 - \hat{a}}. \tag{2.6}$$

In (31) an extension of the above methods is introduced to deal with multi-dimensional datasets, consisting of a union of disjoint manifolds with possibly different IDs. The authors propose a heuristic to automatically determine the local neighborhoods with similar geometric structure and compute the ID on these restricted neighborhoods.

Nearest Neighbors-Based methods

Under the assumption that close points are uniformly drawn from d-dimensional hyperspheres, it is possible to describe data neighborhoods as functions of the intrinsic dimension. One of the first methods providing a mathematical motivation to the use of nearest neighbor distances is (32); if $\rho(\mathbf{x})$ is a density distribution defined on $\mathbb{R}^d$, the following approximation holds:

$$\frac{k}{N} \simeq \rho(\mathbf{x})\omega_d r^d, \tag{2.7}$$

where k is the number of nearest neighbors to $\mathbf{x}$ within the hypersphere $\mathcal{B}(\mathbf{x}, r)$ with radius r and centered on $\mathbf{x}$, while ω_d is the volume of the d-dimensional unit sphere in $\mathbb{R}^d$. Intuitively this tells that the proportion of points in the sampling falling into the ball $\mathcal{B}(\mathbf{x}, r)$ is $p(\mathbf{x})$ times the volume of the ball. If the density is constant, by generalizing equation 2.7 to average values of r_k obtained on the full dataset the authors propose the following ID estimator:

$$\hat{d} = \frac{\bar{r}_k}{k(\bar{r}_{k+1} - \bar{r}_k)}, \tag{2.8}$$

where $\bar{r}_k = (\frac{1}{N}) \sum_{i=1}^{N} r_i^{(k)}$ is the average of the distances of each sample point i to its kth nearest neighbor. A drawback of this technique lies in the arbitrariness of the choice of k, though in (33) a variant was proposed to overcome this difficulty.
Another estimator belonging to this category is Maximum Likelihood Estimation or MLE in short (34); let $x_1, ..., x_N \in \mathbb{R}^n$ be i.i.d. observations representing an embedding of a lower-dimensional sample, i.e. $x_i = g(y_i), \quad y_1, ..., y_N \in \mathbb{R}^d$, where g is smooth enough to guarantee that close points are mapped into close points. In this framework d represents the intrinsic dimension of the dataset. Suppose that vectors $\{y_i\}_{i=1,...,N}$ are sampled from an unknown smooth density function f; once

selected a point $\bar{x}$, if we assume that f is constant on a ball $\mathcal{B}(\bar{\mathbf{x}}, r)$ we can view the sample as an inhomogeneous Poisson process counting the number of observations within a distance r from $\bar{x}$:

$$N(r, \bar{x}) = \sum_{i=1}^{N} \mathbb{1}\{x_i \in \mathcal{B}(\bar{\mathbf{x}}, r)\}. \tag{2.9}$$

The distribution of f around $\bar{x}$ is usually not constant, but imposing $f \equiv \bar{f}$ around $\bar{x}$ allows treating the observations as a homogeneous Poisson process $N(r)$ with rate

$$\lambda(r) = d\bar{f}\omega_d r^{d-1}. \tag{2.10}$$

This follows from the fact that in Poisson processes $E[N(r)] = \lambda r$ together with equation 2.7. Following the observations in (35), if $\theta \doteq \log(\bar{f})$ it is possible to write the log-likelihood of $N(r)$ as

$$\mathcal{L}(d, \theta) = \int_0^R \log(\lambda(r)) dN(r) - \int_0^R \lambda(r) dr. \tag{2.11}$$

At this point the estimate for d given R is obtained through classical maximization technique:

$$\hat{d}_R(\bar{x}) = \left[\frac{1}{N(R, \bar{x})} \sum_{j=1}^{N(R,\bar{x})} \log\left(\frac{R}{r_j(\bar{x})}\right) \right]^{-1}, \tag{2.12}$$

where $r_j(\bar{x})$ is the Euclidean distance between $\bar{x}$ ant its j-th neighbor. It can be convenient to fix the number of neighbors instead of the radius R, to obtain the estimate

$$\hat{d}_k(\bar{x}) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log\left(\frac{r_k(\bar{x})}{r_j(\bar{x})}\right) \right]^{-1}. \tag{2.13}$$

At this point ID estimates from every point in the dataset can be collected into a single measure for instance by averaging over the full sample. A further step is to alleviate the dependence on k by averaging over different values $k_1, ..., k_2$. A variant of the method is proposed in (36), where the arithmetic mean over the full sample is substituted by the harmonic average.

An important drawback of the MLE and other nearest neighbors techniques is the sensitivity to curvature effects; an error due to inhomogeneities in the density is unavoidable if the model lies on the assumption of local uniformity, but it is amplified by the use of large neighborhoods. In (37) the authors introduce a maximum

likelihood technique based on the distribution of the distances to the first neighbor only. The method is still affected by a negative bias due to the assumption unlimited availability of data; thus, the authors propose a novel estimator (37) that tries to correct the bias through a comparison of the measures obtained on the data with those collected on a controlled environment (uniform hyperspheres).
An improvement of the aforementioned techniques is described in (38). Here the authors reduce the underestimation effect by combining information about nearest-neighbor distances and mutual angles. This method is commonly referred to as DANCo (12) and it is considered the best state-of-the-art method; since we mainly compare our results with those obtained by DANCo, a detailed description of the approach is provided in Appendix C.

2.3 Methods

The most important drawbacks one can find in most of the estimators described above are fundamentally three: difficulties at facing varying densities and curvatures, significant computational burden, and a lack of a procedure to detect errors in the measure. While the first two issues have been addressed in different ways, the third one has been substantially neglected. Though, it is crucial to be able to accredit an estimation with an index of reliability; for instance, many models work on the assumption of local uniformity of the data distribution, where uniformity refers to the density and curvature but also to the dimensionality of the space; if this hypothesis is violated, for instance if more manifolds with different dimensions coexist in the same dataset, extracting a single measure for the ID is meaningless.
In this chapter we derive an estimator for the intrinsic dimension that, besides being computationally feasible in the case of large datasets and relatively insensitive to density variations and curvatures, is able to 'ring' an alarm bell when the model fails to describe the data (and thus the measure is unreliable). In order to speed up the estimation and to limit the effects of inhomogeneities in the dataset we develop a model involving only the first two nearest neighbors for each point; the model is based on the computation of the probability distribution of volumes shells in a uniform Poisson process, and the premises are similar to those described in (37), but instead of performing a maximum likelihood estimation we introduce a fitting procedure that allows detecting at first sight whether the model is correct or not. Moreover, we focus on the problem of noise detection, a fundamental point in the case of real datasets; we propose a technique based on block analysis that is capable

of detecting the number of relevant directions of the dataset.

Let i be a point in the dataset, and consider the list of its first k nearest neighbors; let r_1, r_2, ..., r_k be a sorted list of their distances from i. Thus, r_1 is the distance between i and its nearest neighbor, r_2 is the distance with its second nearest neighbor and so on; in this definition we conventionally set $r_0 = 0$.
The volume of the spherical shell enclosed between two successive neighbors $l-1$ and l is given by

$$\Delta v_l = \omega_d \left(r_l^d - r_{l-1}^d \right), \tag{2.14}$$

where d is the dimensionality of the space in which the points are embedded and ω_d is the volume of the d-sphere with unitary radius. It can be proved (see Appendix A for a derivation) that if the density is constant around point i all the Δv_l are independently drawn from an exponential distribution with rate equal to the density ρ :

$$P(\Delta v_l \in [v, v + dv]) = \rho e^{-\rho v} dv. \tag{2.15}$$

This is a special case of equation 2.10 for the radius equal to the distance to the second neighbor, and is also related to the method described in (37). Consider two shells Δv_1 and Δv_2, and let R be the quantity $\frac{\Delta v_i}{\Delta v_j}$; the previous considerations allow us, in the case of constant density, to compute exactly the probability distribution (pdf) of R:

$$\begin{aligned} &P(R \in [\bar{R}, \bar{R} + d\bar{R}]) \\ &\quad = \int_0^\infty dv_i \int_0^\infty dv_j \rho^2 e^{-\rho(v_i + v_j)} \mathbb{1}_{\{\frac{v_j}{v_i} \in [\bar{R}, \bar{R} + d\bar{R}]\}} \\ &\quad = d\bar{R} \frac{1}{(1+\bar{R})^2}, \end{aligned}$$

where $\mathbb{1}$ represents the indicator function. Dividing by $d\bar{R}$ we obtain the pdf for R :

$$g(R) = \frac{1}{(1+R)^2}. \tag{2.16}$$

The pdf does not depend explicitly on the dimensionality d, which appears only in the definition of R. In order to work with a cdf depending explicitly on d we define

quantity $\mu \doteq \frac{r_2}{r_1} \in [1, +\infty)$. R and μ are related by equality

$$R = \mu^d - 1. \tag{2.17}$$

This equation allows finding an explicit formula for the distribution of μ :

$$f(\mu) = d\mu^{-d-1}\mathbb{1}_{[1,+\infty)}(\mu), \tag{2.18}$$

while the cumulative distribution (cdf) is obtained by integration:

$$F(\mu) = (1 - \mu^{-d})\mathbb{1}_{[1,+\infty)}(\mu). \tag{2.19}$$

Functions f and F are independent of the local density, but depend explicitly on the intrinsic dimension d.

2.3.1 A Two Nearest Neighbors estimator for intrinsic dimension

The derivation presented above leads to a simple observation: the value of the intrinsic dimension d can be estimated through the following equation

$$-\frac{log(1 - F(\mu))}{log(\mu)} = d. \tag{2.20}$$

Remarkably the density ρ does not appear in this equation, since the cdf F is independent of ρ. This is an innovation with respect to, for instance, (26) where the dimension estimation is susceptible to density variations. If we consider the set $S \subset \mathbb{R}^2$, $S \doteq \{(log(\mu), -log(1 - F(\mu)))\}$, equation 2.20 claims that in theory S is contained in a straight line $l \doteq \{(x, y) \quad | \quad y = d * x\}$ passing through the origin and having slope equal to d. In practice $F(\mu)$ is estimated empirically from a finite number of points; as a consequence, the left term in equation 2.20 will be different for different data points, and the set S will only lie around l. This line of reasoning naturally suggests an algorithm to estimate the intrinsic dimension of a dataset:

1. Compute the pairwise distances for each point in the dataset $i = 1, ..., N$.
2. For each point i find the two shortest distances r_1 and r_2.
3. For each point i compute $\mu_i = \frac{r_2}{r_1}$.

4. Compute the empirical cumulate $F^{emp}(\mu)$ by sorting the values of μ in an ascending order through a permutation σ, then define $F^{emp}(\mu_{\sigma(i)}) \doteq \frac{i}{N}$.

5. Fit the points of the plane given by coordinates $S = \{(log(\mu_i), -log(1 - F^{emp}(\mu_i)))|i = 1, ..., N\}$ with a straight line passing through the origin.

Even if the results above are derived in the case of a uniform distribution of points in equations (2.18) and (2.20) there is no dependence on the density ρ; as a consequence from the point of view of the algorithm we can a posteriori relax our hypothesis: we require the dataset to be only *locally* uniform in density, where locally means in the range of the second neighbor. From a theoretical point of view, this condition is satisfied in the limit of N going to infinity. By performing numerical experiments on datasets in which the density is non-uniform we show empirically that even for a finite number of points the estimation is reasonably insensitive to density variations. The requirement of local uniformity only in the range of the second neighbor is an advantage with respect to competing approaches where local uniformity is required at larger distances.

2.4 Benchmark

In Figure 2.5 we plot $-log(1 - F^{emp}(\mu_i))$ as a function of $log(\mu_i)$ for three exemplar datasets containing 2500 points: a dataset drawn from a uniform distribution on a hypercube in dimension $d = 14$, analyzed with periodic boundary conditions (pbc), a dataset drawn from a uniform distribution on a Swiss Roll embedded in a three-dimensional space, and a Cauchy dataset in $d = 20$. By "Cauchy dataset" we refer to a dataset where the norms of points are distributed according to the pdf $f(x) = \frac{1}{1+x^2}$. The hypercube with pbc is the pdf that best resembles a uniform distribution on a linear space; nevertheless it has to be noticed that the pbcs introduce correlations in the distances whenever the typical distance of the second neighbor is comparable with the box size. In the same figure, we draw the straight line passing through the origin and fitting the points $\{(log(\mu_i), -log(1-F^{emp}(\mu_i)))|i = 1, ..., N\}$. The slope of this line is denoted in the following by $\hat{d}$. According to the TWO-NN estimator, the value of the ID for the uniform hypercube is $\hat{d} = 14.09$, a measure that is consistent with the ground truth values. For the Swiss Roll the ID estimated by TWO-NN is 2.01 . This value corresponds to the dimension of a hyperplane tangent to the Swiss Roll: in fact, by employing only the first two neighbors of each point, the TWO-NN

estimator is sensible to the local dimension even if the points are relatively few and are embedded in a curved hypersurface.

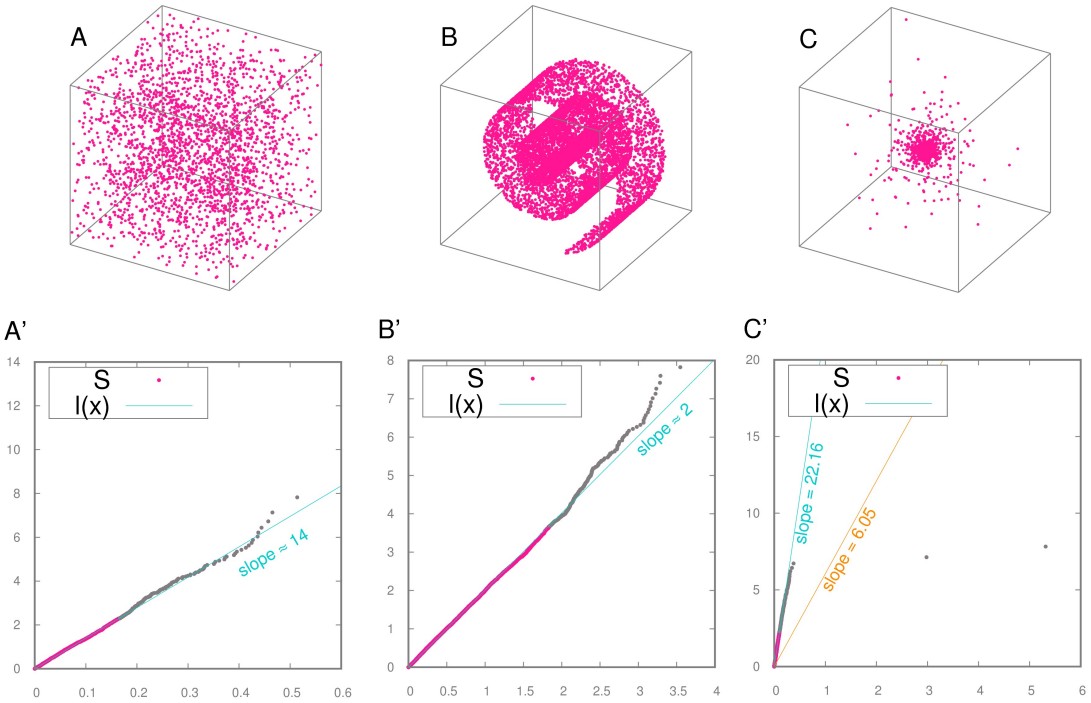

Figure 2.5: *The fitting function $l(x)$ in three exemplar datasets of* 2500 *points. In the first column we display the dataset while in the second one we represent dataset S (red dots) together with the discarded points (gray dots) and the fitting function $l(x)$. Panel A, A': cube in dimension* 14 *(in panel A only the first* 3 *coordinates are represented) analyzed with pbc. Panel B, B': a Swiss Roll. Panel C, C': a Cauchy dataset in dimension* 20 *(only the first* 3 *coordinates are represented).*

For the Cauchy dataset, we obtain $\hat{d} = 6.05$, a value sizeably different form the correct one. Indeed, the slope of the fitting line is strongly affected by a few points characterized by a high value of μ_i. In distributions characterized by heavy tails there is a significant probability of having $r_2 >> r_1$ and a large value of the ratio $\frac{r_2}{r_1}$. This makes the fit unstable. In order to cope with these situations and make the procedure more robust, we discard the 10% of the points characterized by highest values of μ from the fitting. The slopes of the lines obtained in this manner are 13.91, 2.01 and 22.16 for the hypercube, the Swiss Roll and the Cauchy dataset respectively. Remarkably, the value of the slope is practically unchanged for the hypercube and the Swiss Roll, while it is quite different for the Cauchy dataset; in this case by discarding the last points the measure is closer to the ground truth, the fit is more stable and the overall procedure more reliable. Therefore, from now on we discuss

results obtained by fitting the line only on the first 90% of the points. In SI we discuss more in detail the effects of discarding different fractions of data points, and show that the estimate for the dimension is robust with respect to this threshold. We then tested the asymptotic convergence of $\hat{d}$ when the number of points goes to infinity.

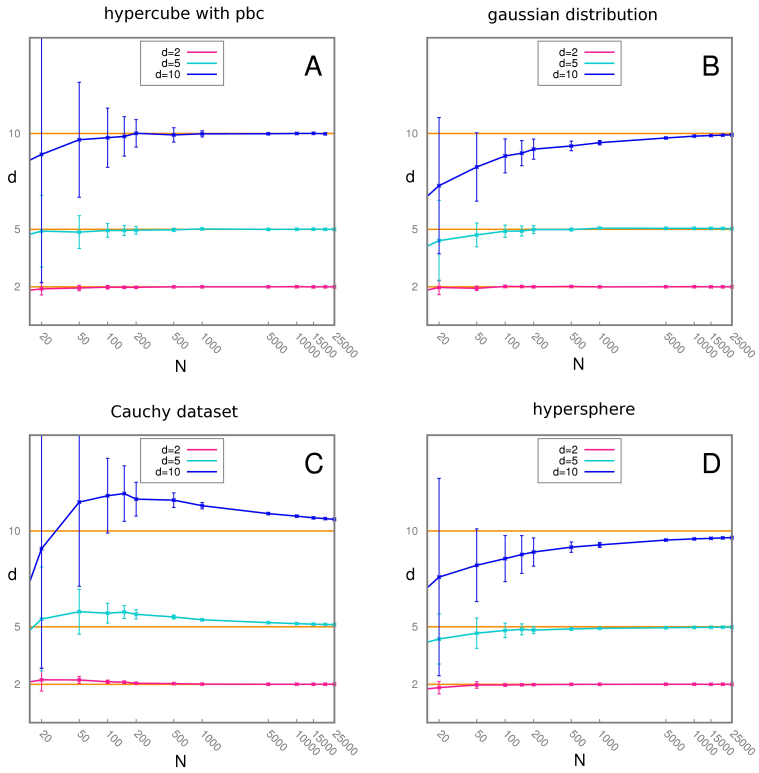

Figure 2.6: *Scaling of the estimated ID with respect to the number of points; for each distribution and for a number of points going from* 20 *to* 25000 *we harvest* 200 *instances of the dataset and average the resulting estimates for the ID. The test is carried out in dimension* 2, 5 *and* 10. *Panel A: Hypercube with pbc. Panel B: Gaussian distribution. Panel C: Cauchy dataset. Panel D: uniform distribution on a hypersphere.*

As the number of points drawn from a probability distribution grows the distances to the second neighbor get smaller and the effects of curvature and density variations become negligible. As a consequence, the hypothesis of local uniformity in the range of the second neighbor is more strongly justified and the distribution of μ approximates better and better the pdf f; moreover, as the number of points goes to infinity the empirical cumulate F^{emp} converges to the correct one F almost surely. Hence, we expect the estimates obtained by TWO-NN to approach the correct value.

In Figure 2.6 we analyze the asymptotic behavior of the measure obtained on a uniform hypercube with periodic boundary conditions, a Gaussian distribution, a Cauchy dataset, and a uniform distribution on a hypersphere. The Gaussian and the Cauchy datasets are interesting test cases as they display a variation in density, while the hypersphere is a case of a uniform distribution on a curved space. In all the cases the estimated dimension appears to converge to the real one. The convergence is faster at lower dimensions: such behavior is expected since if we fix the number of points and the size of the domain the average distance to the second neighbor is shorter in the case of low dimensions, and the hypothesis of local uniformity is closer to being satisfied. The Cauchy dataset is characterized by high variance in the case of a few points, due to the presence of outliers in the S set even when the 10% of points with higher μ is discarded. We performed additional tests by comparing the estimates of TWO-NN with those obtained with DANCo (38), one of the best state-of-the-art methods according to (12). In the next section we provide a detailed description of the results.

Additional benchmark: a comparison between TWO-NN and DANCo

We compare our results with those obtained with DANCo (38) since, according to the analysis in (12), it seems to outperform the other estimators (a public version of DANCo algorithm is available at https://it.mathworks.com/matlabcentral/fileexchange/40112-intrinsic-dimensionality-estimation-techniques/content/idEstimation/DANCoFit.m.). In order to test DANCo in the case of uniform hypercubes with periodic boundary conditions we modified the computation of distances in the code. First of all we analyzed the estimates of DANCo and TWO-NN on datasets with 2500 points and dimension ranging from 1 to 20. The selected datasets are hypercubes without periodic boundary conditions, hypercubes with periodic boundary conditions, Cauchy dataset and Gaussians. We embed the datasets in higher dimensional spaces through the identity map since the algorithms select as an upper bound the dimension of the embedding space (trivially corresponding to the number of coordinates), and in some cases it can correspond to the true ID. In the case of hypercubes without pbc (panel A) TWO-NN produces an underestimation (about 1.5 in dimension 10 and 4 in dimension 20), due to the sharp drop in density at the border. This systematic error becomes smaller and smaller when the number of points is increased. A similar but lighter effect is visible in the case of Gaussian distributions (panel D): here the

density changes rapidly but in a smoother fashion. We notice an underestimation of around 0.1 in dimension 10 and 3 in dimension 20. In panel B we see that considering periodic boundary conditions (and thus reproducing a most uniform environment) allows TWO-NN to estimate the ID almost correctly, with an underestimation of the order of 1 in dimension 20. In the case of Cauchy dataset (panel C) TWO-NN slightly overestimates the intrinsic dimension.

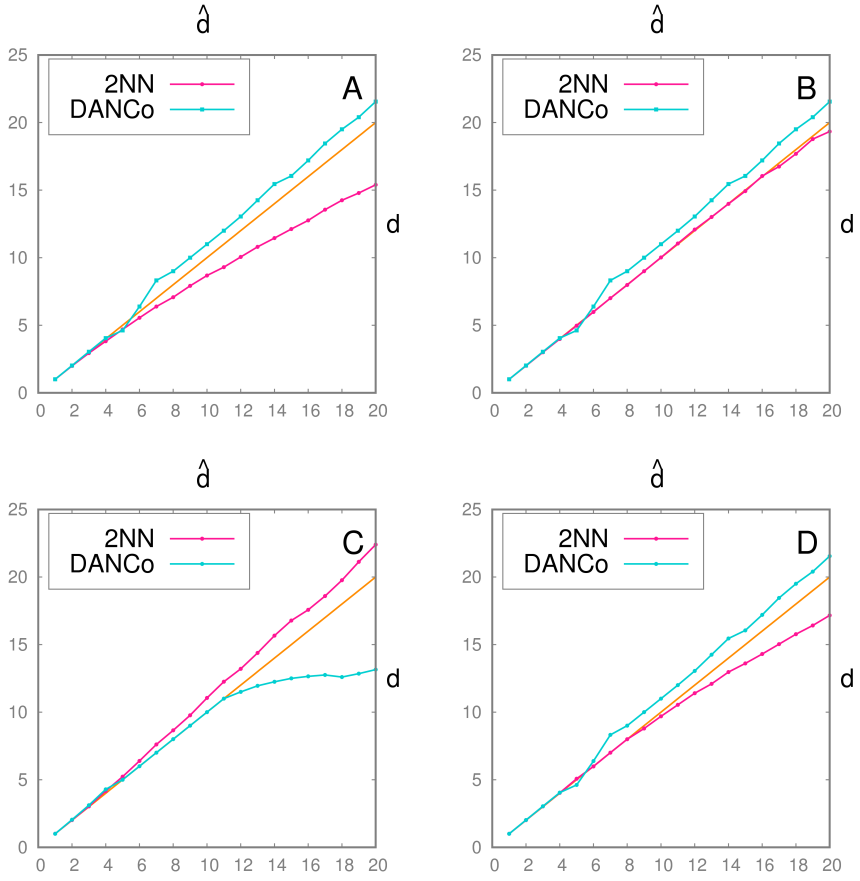

Figure 2.7: *ID estimates for DANCo and TWO-NN on selected datasets of 2500 points. For each dimension we take as ID estimate the average over 20 instances of the dataset. On the x-axis and y-axis we represent the true dimension of the dataset d and the estimated dimension $\hat{d}$ respectively. Panel A: Hypercubes embedded in a space of dimension $d+5$ through the identity map; the test is carried out with no periodic boundary conditions. Panel B: Hypercubes embedded in a space of dimension $d+5$ through the identity map; the test this time is carried out with periodic boundary conditions. Panel C: Cauchy datasets embedded in a space of dimension $d+3$. Panel D: Gaussian distributions embedded in a space of dimension $d+5$.*

As for DANCo, we notice that it slightly overestimates the dimension for the Hypercubes and for the Gaussian, while it strongly underestimates the value of the

ID in the case of Cauchy dataset (the estimate for a Cauchy dataset in dimension 20 is around 13).

Table 2.1: *The 21 synthetic datasets proposed in (12)*

Dataset	Description	N	d	D
M_1	10-dimensional hypersphere linearly embedded	2500	10	11
M_2	Affine space	2500	3	5
M_3	Concentrated figure, mistakable with a 3 dimensional one	2500	4	6
M_4	Nonlinear manifold	2500	4	8
M_5	2-dimensional helix	2500	2	3
M_6	Nonlinear manifold	2500	6	36
M_7	Swiss-Roll	2500	2	3
M_9	Affine space	2500	20	20
M_{10a}	10-dimensional hypercube	2500	10	11
M_{10b}	17-dimensional hypercube	2500	17	18
M_{10c}	24-dimensional hypercube	2500	24	15
M_{10d}	70-dimensional hypercube	2500	70	71
M_{11}	Möebius band 10-times twisted	2500	2	3
M_{12}	Isotropic Multivariate Gaussian	2500	20	20
M_{13}	1-dimensional helix curve	2500	1	3
M_{N1}	Manifold non-linearly embedded in $\mathbb{R}^{72}$	2500	18	72
M_{N2}	Manifold non-linearly embedded in $\mathbb{R}^{96}$	2500	24	96
M_{beta}	Manifold non-linearly embedded in $\mathbb{R}^{40}$	2500	10	40
M_{P3}	Manifold non-linearly embedded in $\mathbb{R}^{12}$	2500	3	12
M_{P6}	Manifold non-linearly embedded in $\mathbb{R}^{21}$	2500	6	21
M_{P9}	Manifold non-linearly embedded in $\mathbb{R}^{30}$	2500	9	30

Table 2.2: *The 7 additional datasets*

Dataset	Description	N	d	D
$C10$	10-dimensional cauchy dataset linearly embedded in $\mathbb{R}^{15}$	2500	10	15
$C15$	15-dimensional cauchy dataset linearly embedded in $\mathbb{R}^{20}$	2500	15	20
$C30$	30-dimensional cauchy dataset linearly embedded in $\mathbb{R}^{35}$	2500	30	35
$M8$	12-dimensional manifold embedded in $\mathbb{R}^{72}$	2500	12	72
$HC10$	10-dimensional hypercube linearly embedded in $\mathbb{R}^{15}$	2500	10	15
$HC17$	17-dimensional hypercube linearly embedded in $\mathbb{R}^{22}$	2500	17	22
$HC24$	24-dimensional hypercube linearly embedded in $\mathbb{R}^{29}$	2500	24	29

We believe that the origin of this significant systematic error lies in the fact that DANCo estimates the ID by comparing the theoretical functions obtained in the

dataset with those retrieved on uniform spheres: this strategy works well in the case of sharp boundaries but is less suitable in the presence of heavy tails.
We further tested TWO-NN on the synthetic benchmarks proposed in (12), listed in Table2.1 together with their relevant features N, d, D; they include datasets characterized by high dimensionality, sharp edges, or described by a complex non-linear embedding.

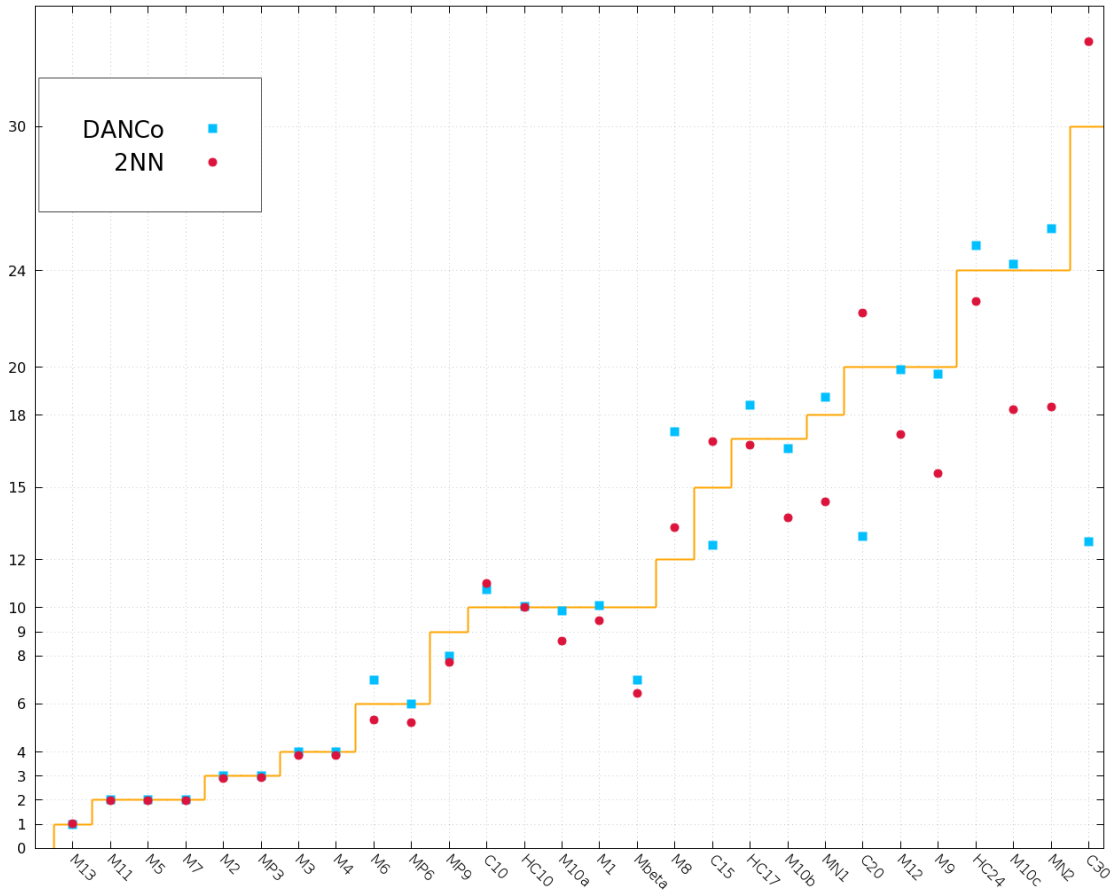

Figure 2.8: *ID estimates for DANCo and TWO-NN on 20 selected datasets of 2500 points described in Table2.1 plus 7 additional datasets described in Table2.2 . For each dimension we take as ID estimate the average over 20 instances of the dataset. On the x-axis and y-axis we represent the true dimension of the dataset d and the estimated dimension $\hat{d}$ respectively.*

Datasets from M_1 to M_{13} are generated from the publicly available tool (http://www.mL.uni-saarland .de/code/IntDim/IntDim.htm) proposed by Hein and Audibert in (39); only dataset M_8 is missing from the analysis in (12), since according to the authors it is particularly challenging for its high curvature and induces pronounced overestimates in many relevant ID estimators (see (12)). Datasets M_{P3}, M_{P6}, M_{P9} (see (40))

are interesting because the underlying manifold is characterized by a non constant curvature.
Finally, datasets M_{N1}, M_{N2}, M_{beta} are proposed by the authors of (12) themselves. For a full description of the datasets and tools to generate them refer to (12).
Since the large majority of the datasets proposed in (12) are characterized by boundaries where the density drop is very sharp, or even discontinuous, we added to the proposed benchmarks some synthetic datasets we list and describe in Table2.2; the 7 new benchmarks display a smooth behavior at the boundaries. $C10$, $C15$, $C30$ are Cauchy datasets and $HC10$, $HC17$, $HC24$ are uniform hypercubes embedded through an identity map in a higher dimensional space; on the latters we test the method applying periodic boundary conditions (pbc) in order to simulate as much as possible a uniform environment. As suggested in (12) we generated 20 instances of each dataset and averaged the achieved results; The result of the tests is summarized in Figure 2.8. We omit to display the measure for dataset M_{10d} since its ID is 70, and estimating the dimension of such datasets is beyond the intentions of TWO-NN (indeed, as we expect we undergo a strong underestimation of 41 in this case). Note that in the case of sharp boundaries, as for instance in the hypercube without pbc $M10b$, DANCo performs better, while in the case of smooth boundaries as in $HC17$ TWO-NN is more accurate. In the case of outliers as in the Cauchy datasets $C20$ and $C30$ DANCo severely underestimates the true dimension.

2.5 Estimating a scale-dependent intrinsic dimension

An important feature of the TWO-NN estimator is its locality: it provides an estimate of the ID by considering only the first and second neighbor of each point. This makes it suitable for analyzing how the ID varies with the scale, and distinguishing in this way the number of "soft" directions. As a basic example, consider a sample of points harvested form a uniform distribution on a plane perturbed by a Gaussian noise with variance σ in a large number of orthogonal directions. This example mimics what is observed in samples extracted from a finite temperature molecular dynamics run, in which most of the possible directions are strongly disfavored by steric constraints. In the example, if the scale of interest is much larger than σ, say 10σ the relevant ID is 2.
We notice that what makes the notion of ID well-defined in this example is the

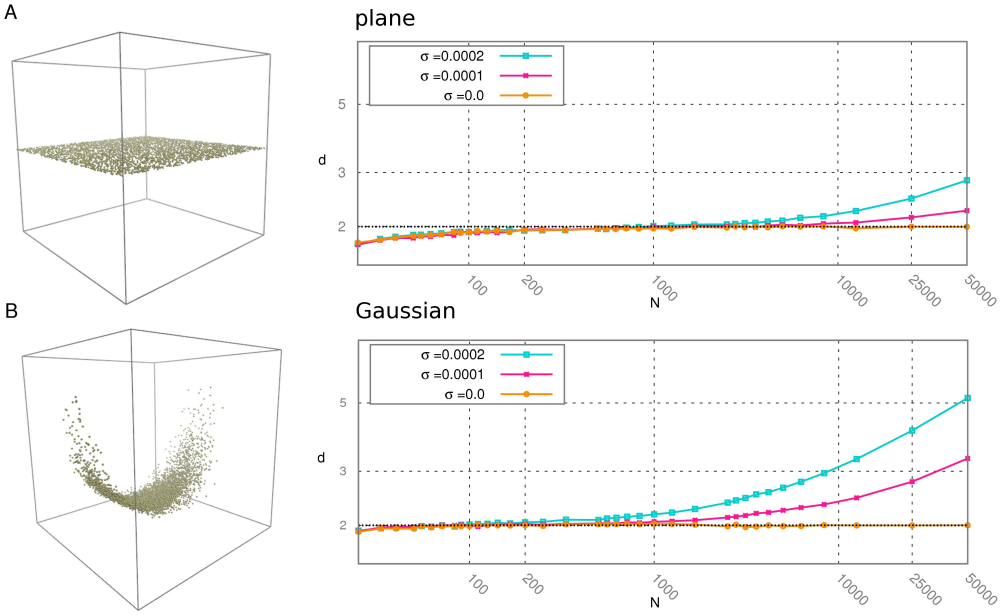

Figure 2.9: *Estimated dimension d vs the number of points N in logarithmic scale; for each value of N the dataset is partitioned in a number of independent sets containing exactly N points, d is computed on each subdataset and a measure $d(N)$ is obtained as an average of these values. In Panel A we study the case of a uniform plane of* 50000 *points in dimension* 2 *perturbed by a Gaussian noise with variance σ along* 20 *independent directions; σ takes the three values* 0.0, 0.0001 *and* 0.0002. *In Panel B we analyze a dataset composed of a two-dimensional Gaussian of* 50000 *points wrapped around a Swiss Roll and perturbed by a Gaussian noise with variance σ along* 20 *independent directions. Again σ takes the three values* 0.0, 0.0001 *and* 0.0002.

stability of the measure with respect to changes in the scale of interest: the ID would be 2 also on an even larger scale, say 100σ.

in Nearest Neighbors-Based estimators the reference scale is the size of the neighborhood involved in the estimation; this depends on the density of points in the sample and does not necessarily coincide with the scale of interest. Going back to the example of the plane with noise, the more data points are used for the estimate, the smaller the average distance of the second neighbor will become, and the larger the ID. These observations suggest that in order to check the relevance of our measure we can study the stability of the estimation with respect to changes in the neighborhood size like in a standard block analysis. In the case of TWO-NN it is possible to modify the neighborhood size by reducing the number of points in the dataset: the smaller N, the larger the average distance to the second neighbor. In practice, similarly to the approach adopted in (41), the analysis of the scaling of the dimension vs the

number of points can be carried out by extracting sub samples of the dataset and monitoring the variation of the estimate $\hat{d}$ with respect to the number of points N. The relevant ID of the dataset can be obtained by finding a range of N for which $\hat{d}(N)$ is constant, and thus a plateau in the graph of $\hat{d}(N)$. The value of d at the plateau is the number of "soft", or relevant, directions in the dataset.
In Figure 2.9 we analyze the dimension. In Panel A we study the case of a uniform plane in dimension 2 perturbed by a high-dimensional Gaussian noise with variance σ. We see that for $\sigma = 0.0001$ and $\sigma = 0.0002$ $d(N)$ displays a plateau around $N = 1000$, and the value of the dimension at the plateau is 2, equal to the number of soft directions in the dataset. As the number of points grows also noisy dimensions are sampled, and the value of the estimated ID increases. For critically low values of N the estimated ID decreases to one, as expected (two points are always contained in a line). In Panel B we analyze a more challenging dataset composed of a two-dimensional Gaussian wrapped around a Swiss Roll and perturbed by a high-dimensional Gaussian noise with variance σ. Also in this case we find a plateau, around 100 for $\sigma = 0.0002$ and around 500 for $\sigma = 0.0001$, at which the estimated dimension is 2. It is important to notice that even if the dataset analyzed in Panel B is far more complex than the simple plane in Panel A the behavior of the dimension vs the number of points is essentially the same in the two cases.

Analysis of image datasets

Estimating a scale-dependent intrinsic dimension is highly useful in the case of real datasets. In Figure 2.10 we compute the intrinsic dimension of two complex datasets: the Isomap face database and the handwritten "2"s from the MNIST database (18). The first dataset consists of 598 vectors with 4096 components, representing the brightness values of 64 pixel by 64 pixel images of a face with different lighting directions and poses. The second one is composed by 1032 vectors with 784 components representing handwritten "2"s. Despite the relatively low number of points the block analysis is able to robustly detect the instrinsic dimension of the two datasets. In the case of Isomap faces we see a plateau for a number of points greater than roughly 400 and the measure of the ID in the range of the plateau is 3.5, slightly higher but consistent with 3, the value considered to be correct. In the case of MNIST dataset the plateau is located in a range between 300 and 500 points, and the measure of the ID corresponding to the plateau is 13.4, consistently with previous estimations that state the ID to be between 12 and 14 (39), (31). Here, the

measure we would obtain with the whole dataset would overestimate the ID due to the presence of noise, similarly to what observed in the artificial datasets in Figure 2.9.

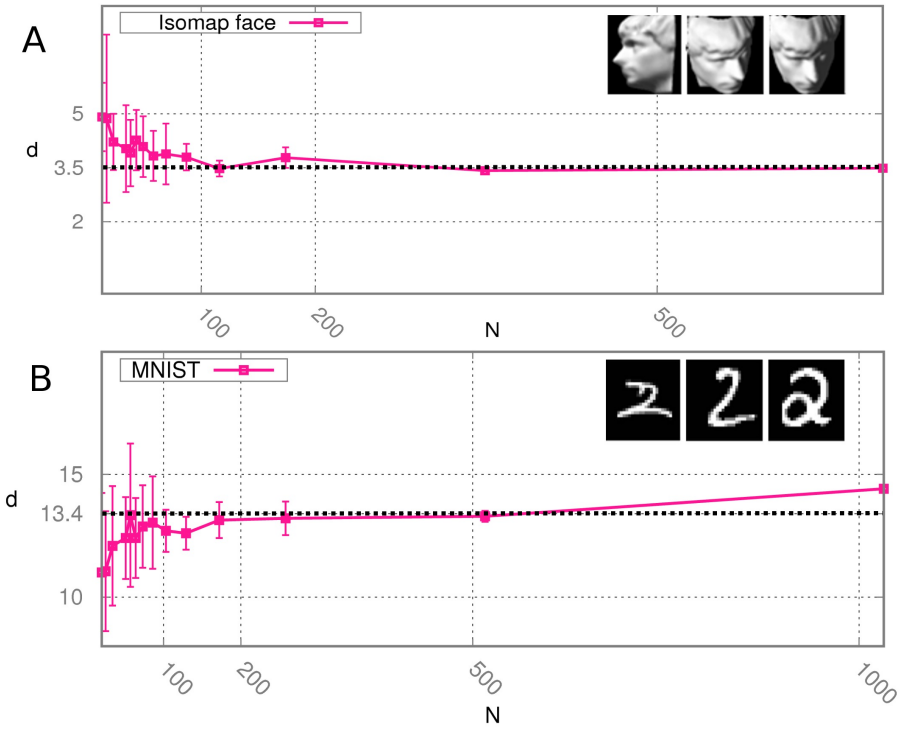

Figure 2.10: *Scaling of the estimated ID with respect to the number of points for ISOMAP face (panel A) and MNIST database (panel B).*

Estimating the ID of trinucleotide AAA dynamics

We finally estimate the ID of the configurational space explored during a molecular dynamics trajectory of the RNA trinucleotide AAA (42). The dynamics was performed using GROMACS 4.6.7 (43) at a temperature $T = 300$ K. RNA molecules were solvated in explicit water. From the original trajectory of 57 ms, we keep a configuration every 6 ns, obtaining a total number of 9512 structures. The simulation was originally carried out to provide insight into the main relaxation modes of short RNA. Computing the ID can provide a guideline for performing dimensionality reduction thus retaining in the description a meaningful number of variables. We perform the analysis of the ID making use of two notions of distance; the first one is the Euclidean distance between the coordinates associated to each sample by Time-lagged Independent Component Analysis (TICA)(5). The second one is the

Root Mean Square Deviation (RMSD) between all the atoms in the trinucleotide. These two distances are intrinsically different from each other, but strikingly the measure of the ID obtained in the two cases is comparable as shown in Figure 2.11, with values of approximately 9.5 and 8.5 for the two metrics (estimated by using all the 9512 configurations). In the range of N we considered the estimate of d slowly grows, with a trend similar to the one observed in Figure 2.6 on artificial data sets. It is possible in principle to further refine the procedure by fitting the these curves and finding the asymptotic value of d. Noticeably, the scaling features of d vs N with the two metrics are comparable and the ID values on the full datasets differ for only for one unit in dimension nine.

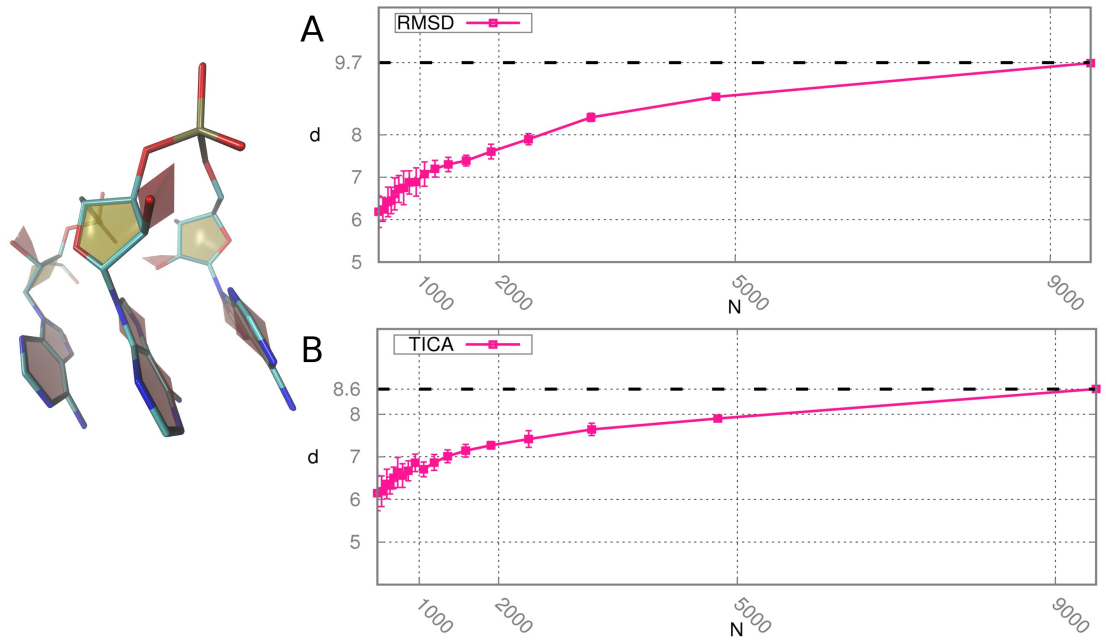

Figure 2.11: *Scaling of the estimated ID with respect to the number of configurations for the dynamics of trinucleotide AAA in the case of RMSD distances (panel A) and TICA distances (panel B). On the left a possible configuration is represented.*

2.6 Discussion

In this chapter we address the problem of finding the minimal number of variables needed to describe the relevant features of a dataset; this number is known as intrinsic dimension (ID). We develop TWO-NN, an ID estimator that employs only the distances to the first two nearest neighbors of every point. The approach

belongs to the class of Nearest-Neighbors based estimators, and it is based on the same premises as in (37) but it adopts a different approach to the estimation that allows recognizing at a first sight cases where the model fails to describe the data. Considering a minimal neighborhood size has some important advantages: first of all it allows to lower the effects of density inhomogeneities and curvature in the estimation process; moreover, it grants a measure that does not mix the features of the dataset at different scales. In the case of locally uniform distributions of points TWO-NN relies on a robust theoretical framework while in the general case, namely in the presence of curvatures and density variations, TWO-NN is numerically consistent. In addition, it is able to provide reliable estimates even in the case of a low number of points.
A primary issue in the case of real datasets is discriminating the number of relevant dimensions. To this purpose we discuss a new method based on the use of TWO-NN to compute the ID on sub samples randomly extracted from the dataset, and analyze the behavior of the estimated dimension with respect to the number of points. A plateau in such graph is indicative for a region in which the ID is well-defined and not influenced by noise. The minimal neighborhood character of TWO-NN is a major advantage in this operation, since it allows exploring in a clean way the different length scales of the sub samples. We show that even in the case of a complex dataset displaying both curvature and density variations and perturbed by high dimensional Gaussian noise we are able to successfully detect the number of relevant directions. We demonstrate that these features allow to estimate the ID even in real world datasets including sets of images and a set of configurations along a finite temperature molecular dynamics trajectory of a biomolecule in water solution. Finally, we remark that using only two nearest neighbors grants a further advantage in terms of time complexity: by employing dedicated algorithms it is possible to find the first few neighbors of each point in an almost linearithmic time (44).

In Chapter 3 we apply TWO-NN to the study of the intrinsic dimension of protein families; the analysis is challenging since it requires a careful definition of the distances between sequences and the datasets are affected by low sampling and correlations. In Chapter 5 instead we see how it is possible to exploit the simplicity of the TWO-NN model to perform an analysis of datasets with components of different dimension in a Bayesian framework.

Chapter 3

Computing the intrinsic dimension of Pfam protein families

3.1 Introduction

Proteins are among the most abundant organic molecules in living systems and comprise a much wider collection of structures and functions than other classes of macromolecules. A single cell can contain thousands of proteins, each of them a with a unique and specific function. Some are structured, involved in keeping cells shape or movement. Others work as signals, drifting between cells, and others are metabolic enzymes, putting together or breaking the biomolecules needed by the cell. Despite the fact that their structures and functions are most diverse all proteins are made up of chains of twenty different amino acids. The rich variety of three-dimensional shapes they are able to display is due to the fact that, even if constrained by a linear backbone, a chain of amino acids possesses a vast rotational and conformational freedom. In addition, even if all amino acids share a common backbone that allows them to interlink, their side chains can be extremely different from each other; this variability ensures them precise physico-chemical properties and is responsible for their lower or higher propensity to make specific chemical bonds and to interact with water.

Thus given a sequence of amino acids some three-dimensional conformations are favored and other disfavored depending on the capability to perform stabilizing chemical interactions, namely hydrogen bonds and salt bridges, between the side chains and on the extent to which the hydrophobic residues are hidden from water. In the 1960s the Nobel prize Anfinsen *et al.* (45) demonstrated that in physiological conditions the most stable conformation of a protein depends only on its sequence.

This conformation is referred to as *native conformation*, and corresponds to the structure characterized by the minimal free energy. Determining the native conformation of a protein is fundamental to investigate its functions; among the experimental techniques devoted to this task the most successful are X-ray chrystallography (46) and Nuclear Magnetic Resonance (NMR) spectroscopy (47), but they are expensive and time consuming. Another way to face the problem is to detect a significant similarity between a new sequence and one about which something is already known, and then transfer some information about structure and function from one another. Learning how to predict which amino acids sequences share a common native conformation can expand our knowledge exponentially and very cheaply, since sequencing is much faster than resolving a structure by direct experimentation. A hint of the importance of developing such procedures is displayed in Figure 3.1; if we compare the size of the two main databases respectively for protein structures, Protein Data Bank or PDB in short (48), and for protein sequences UniProt (49) we see that up to now Uniprot contains almost seven hundred times the number of entries of PDB.

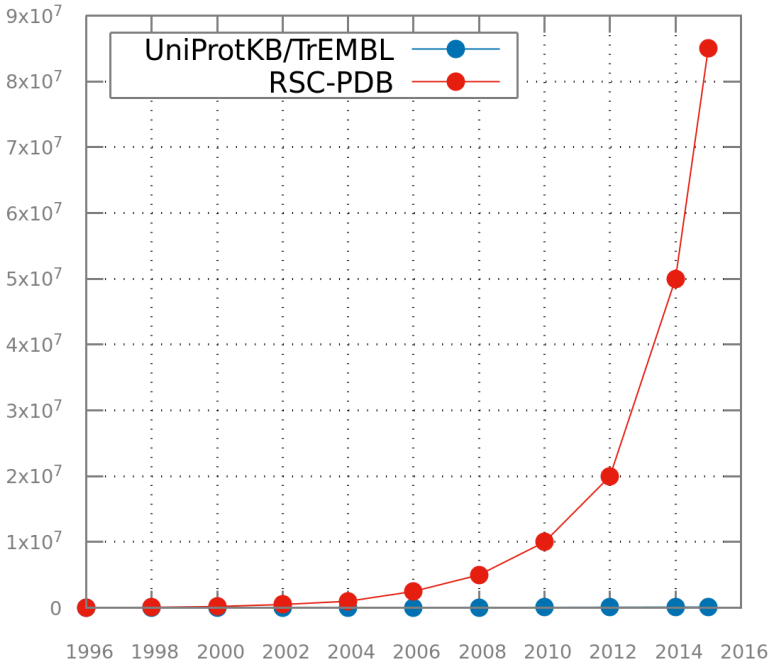

Figure 3.1: *Number of resolved sequences in UniProt compared to the number of resolved structures in the Protein Data Bank per year.*

In nature new sequences are adapted from preexisting ones rather than invented *ex*

novo. In this process structures seem to be conserved much more than sequences, probably due to the fact that structure is entangled with function. If the sequence mutates while maintaining the same structure the function could be preserved, while mutations that are disruptive for the structure usually inhibit the capability of the protein to perform its task. The vast majority of the sequences sharing the same structure are related by evolution; such sequences are said to be *homologous*. Groups of proteins that share the same structure and function are known as *protein families* (50).

The fundamental question is then how to detect common features in proteins that share the same structure. Sequence similarity is a mark of structural similarity, yet almost identical structures can differ significantly in their sequences. Since forty years the pursuit of techniques able to recognize homologs are at the heart of bioinformatics research. The task is far from trivial, as the evolutionary connection between proteins is remote in time, and only the present generation is known. The family tree where present generation proteins represent the leaves is known as *phylogenetic tree*.

If the only information available about proteins is the sequence of amino acids, deciding that they are similar is equivalent to deciding that two text strings are similar. A set of methods for biological sequence analysis is thus rooted in computer science, where a vast literature about string comparison is available. Crucial in this framework is the concept of *alignment*. Evolving sequences accumulate insertions, deletions, substitutions, so before their similarity can be evaluated it is necessary to align them in such a way that most significant parts are superimposed; the tools are scoring schemes and alignment algorithms. Scoring schemes associate to each alignment a score according to its probability to derive from a common ancestor; it can be as simple as '+1' for a match, '-1' for mismatch, but much more complex solutions are adopted to reflect the biological nature of the problem. To this purpose, one has to assign different probabilities of observing certain substitutions between amino acids rather than others; in fact even if all the amino acids differ from each other in their physico-chemical properties, some of them are more compatible than others and mutations can affect the stability and functionality of a protein to very different extents.

Many algorithms have been proposed to perform sequence alignment based on specific scoring schemes. The simplest approach is to compare sequences two by two, choosing the most convenient starting and ending points for an alignment and deciding the possible insertion of gaps to maximize the score; this process is called *Pairwise Sequence Alignment*. More complex is to analyze many sequences at once, that is to

say building a *Multiple Sequence Alignment*: in this case one has to take into account site specific properties.

3.1.1 Pairwise Sequence Alignment and Multiple Sequence Alignment

Most bioinformatics analysis is based on some kind of sequence alignment; aligning sequences consists in recognizing regions of similarity suggestive of evolutionary relationships and proposing a rearrangement of the sequences in which some of the positions are paired. Generally gaps insertions are allowed, as motivated from an evolutionary point of view, but provided that a penalty is paid in the score. A simple case of an alignment between two strings SJRYEVYY and JRYEKLY is

SJRYE-AYY
-JRYEKBY-

Here some residues are conserved (for instance J in the second position, R in the third), some are substituted by chemically similar residues (V with L in the seventh position) and some are deleted or inserted (K in the sixth position, Y in the ninth). To perform an alignment one needs a scoring scheme to associate each alignment with a score; this reflects the ratio between its probability to come from a common ancestor and that of being the work of pure chance. In addition, an algorithm is necessary to explore the possible alignments and, based on their scores, to find the best one.

Scoring Schemes

A scoring scheme is a function associating a score to each pair of aligned residues and a penalty for the insertion of gaps in the alignment. *Scoring matrices* for amino acids are 20×20 matrices providing, for each pair of states (i, j), a score $s_{i,j}$ that is higher or lower according to the probability that i and j are aligned because of a common ancestor rather than by chance. Scoring matrices can be roughly divided into two main classes: PAM-like matrices (51) and BLOSUM-like matrices (52). The former see the evolutionary process as a Markov chain of independent Point Accepted Mutations (from which the acronym PAM), and focus on the description of transition probabilities between the states (or equivalently of instantaneous substitution rates

for amino acids); the latter on the contrary do not make assumptions about the underlying evolutionary process behind substitutions and learn in a direct fashion, from a set of multiple sequence alignments, the probabilities of observing a certain amino acids interchange.
Besides analyzing substitutions between residues most of the aligning algorithms permit the insertion of gaps, that can be interpreted from an evolutionary point of view as insertions or deletions (*indels* in short). If indels are treated as gaps, they are associated to a penalty score; such penalty grows linearly with the number of consecutive gaps or distinguishes between the opening of a gap and its extension. In this way indels lose their evolutionary meaning, as they are barely considered as mismatches. More precise approaches are typical of algorithms based on Hidden Markov Models (53), (54), where gap-penalties are not uniform along the alignment.

Algorithms for sequence alignment

Algorithm for sequence alignment are divided into *pairwise sequence alignments* and *multiple sequence alignments* according to the number of sequences they treat at once; they are further partitioned into *global*, when they required all amino acids in the sequence to be included in the alignment, or *local*, when only parts of the sequences are aligned. Note that local alignment is preferable when it is suspected that two protein sequences share a common domain, and it is usually the most sensitive way to detect similarity in case of highly diverged sequences; in fact in these sequences only a part of the sequence has been under a selection strong enough to preserve a discernible similarity.

Pairwise sequence alignment

Most of the tools for performing pairwise alignment are based on *dynamic programming* (55), a method for solving a complex problem by breaking it down into a collection of simpler sub problems. Solving each of those sub problems just once, and storing their solutions, allows saving computation time at the expense of storage space. Once given a scoring scheme, such methods are guaranteed to find one of the optimal alignments.
The most renowned algorithms in this field are the *Needleman-Wunsh* (56) for global alignments and the *Smith-Waterman* (57) for local ones. Both of them are based on the compilation of a matrix in which columns correspond to letters of the first sequence and rows to letters of the second sequence.

Let $F(i,j)$ be a matrix indexed by positions i and j in the first and second sequence respectively; the value $F(i,j)$ corresponds to the score of the best alignment between the initial segment $x_{1,\dots,i}$ of the first sequence and the initial segment $y_{1,\dots,j}$ of the second one. $F(i,j)$ is filled recursively by initializing $F(0,0) = 0$ and then proceeding from top left to bottom right; the score of each cell in the matrix can be progressively computed from the knowledge of the scores in the three top-left neighboring cells (see Figure 3.2).

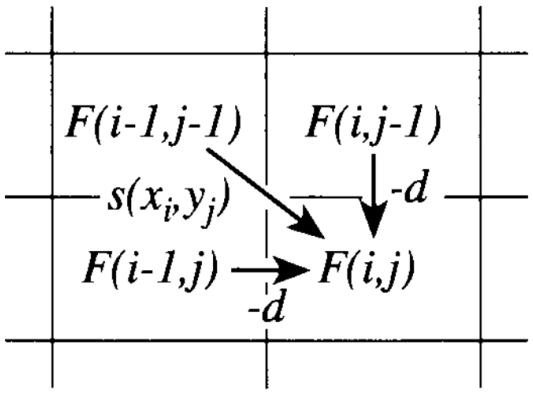

Figure 3.2: *The value $F(i,j)$ can result from one of the three top-left neighboring cells; if it comes from the diagonal, it corresponds to an amino-acids alignment and adds the score $s(x_i, y_i)$; otherwise it corresponds to a gap insertion and adds up the penalty $-d$; Figure reference: (58)*

In this way the score for a global alignment up to position (i,j) is given by the formula:

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + s(x_i, y_j) & \text{top-left diagonal;} \\ F(i-1,j) - d & \text{top;} \\ F(i,j-1) - d & \text{left,} \end{cases} \tag{3.1}$$

where $s(x_i, y_j)$ is the scoring matrix and d the penalty for the insertion of a gap. Equation 3.1 is applied repeatedly in order to fill matrix $F(i,j)$ until all the amino-acids are aligned (see Figure 3.3); it is essential in this process to keep track of where each cell's score was computed from; if n is the length of the first sequence and m that of the second one, the value in the final cell of the matrix $F(n,m)$ is by definition the best score for an alignment of $x_{1,\dots,n}$ to $y_{1,\dots,m}$;

At this point the alignment itself is retrieved by following the path of choices that led to the best score in the reverse order, a procedure called *traceback*.

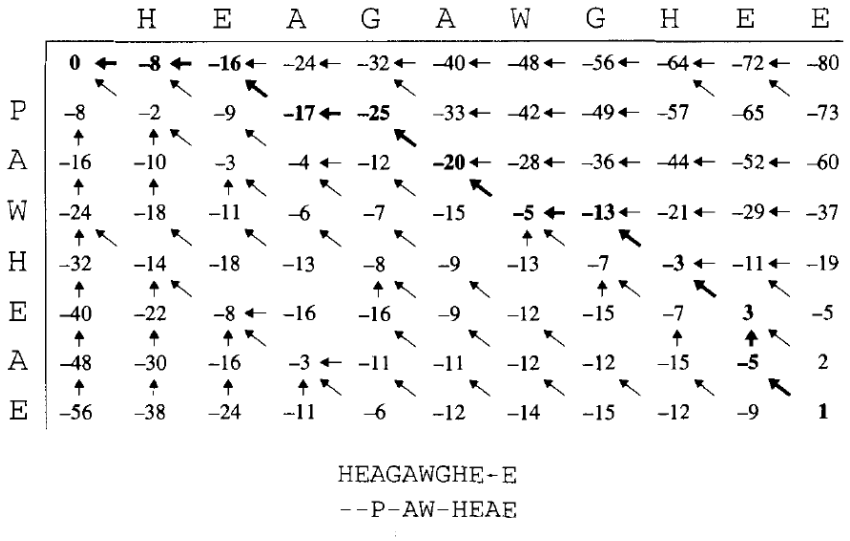

Figure 3.3: *The score matrix $F(i, j)$ together with the directed paths; the traceback giving the optimal alignment is highlighted in bold. Figure reference: (58)*

A slight variation of this procedure is used to compute the best *local alignment* by the Smith-Waterman algorithm; the differences are first that an extra possibility is added for $F(i, j)$ to take the value 0 if all the top left neighbors display negative values: this corresponds to starting a new alignment. Second, alignments can end wherever across the matrix, so instead of starting the traceback from $F(n, m)$ one has to look for the highest score in the matrix and reconstruct the path backwards from there. The traceback ends when meeting a cell with value 0.
Deterministic algorithms grant optimal solution, but are time consuming; for this reason *heuristic techniques* have become very popular, since they are much quicker. The most famous algorithm in the field of heuristic pairwise alignment is BLAST (59). Since we make extensive use of BLAST in the next sections, we provide a more detailed description of the method in the following.

BLAST. The BLAST package provides programs for finding high scoring local alignments between a query sequence and a target database (it could be either proteins or DNA). BLAST relies on the idea that true match alignments are very likely to contain short stretches of identities, or very high scoring matches. So a good starting point when comparing two sequences is to look for short stretches and use them as 'seeds' to extend in search of a longer good alignment. If seeds are short enough, it is possible to preprocess the query sequence in order to make a table of all the possible seeds with their starting points, and making a list of all possible

'neighborhood words' (i.e. words that are similar to the seed) of a fixed length (3 for proteins) that match somewhere the query sequence with a score higher than a threshold. The next step is scanning the database, whenever a word of the list is found a 'hit extension' is started to broaden the possible match as an ungapped alignment in both directions. The extension stops when the maximum score is reached. In this way, a segment pair is defined locally maximal (in short, an MSP) if its score cannot be improved either by extending or by shortening both segments.

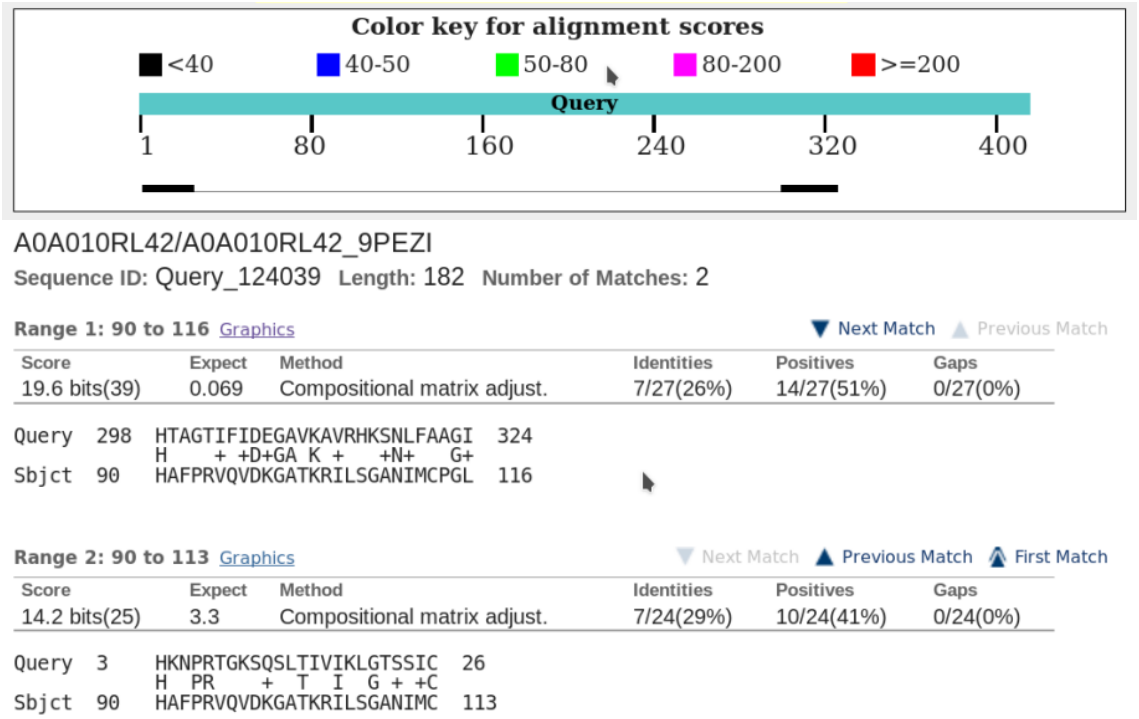

Figure 3.4: *BLAST representation of the possible MSPs between two FASTA sequences from the Pfam family PUA, A0A010RL42 and A0A010RL42_9PEZI. Two meaningful alignments are detected: the first one has E-value* 0.069*, the second has E-value* 3.3*.*

Together with MSP scores BLAST is able to produce statistical information about the significance of alignments; namely, Karlin-Altschul statistics (60) provides a theory for computing the probability that a local alignment of a given score will be found between two random sequences of the same lengths as the query and database sequences. This probability is often expressed as an expectation value, abbreviated to E-value.

E-values are related to the number of BLAST hits you expect to see by chance with the observed score or higher: the lower the E value, the more significant the score is. They are essentially inverse powers of the binary logarithm of the score. For instance, if we have a BLAST alignment with E-value around 1, then it could be easily due to chance. Instead, if the E-value of the alignment is $e-10$, you expect to see that alignment by chance $e-10$ times; in this case the alignment is very unlike to be random and it could be related to a biologically meaningful relationship between sequences. E-value thresholds provide also a way to limit the BLAST output, and thus speed up the alignment: in fact, one can provide an E-value limit above which the alignment is considered not informative, and thus the computation is dropped before being completed.
Note that, due to its heuristic nature, BLAST is not symmetric, meaning that it can in principle provide different alignments between a pair of sequences (A, B) and the other way round (A, B). Moreover, as shown in Figure 3.4, BLAST can provide more MSPs options between two sequences.

Multiple sequence alignment

MSA are computationally difficult to handle, and most of their formulations lead to NP-complete optimizations problems (61). Dynamic programming is in principle extensible to many sequences, but it is extremely slow already for small numbers and is then rarely used for more than three or four sequences. An alternative to dynamic programming is to use approximate methods which generally produce a MSA by first aligning the most similar sequences and successively adding less related sequences and are called progressive methods (an example is ClustalW (62)). Their results depend on the selection of the most related sequences and thus may suffer from inaccuracies in the initial pairwise alignments. To overcome this problem, some algorithms repeatedly realign the initial group of sequences as well as adding new ones to the growing MSA; such methods are defined iterative, and an example is given by Muscle (63). Another class of aligning algorithms realize ad hoc scoring schemes for each family by learning the parameters from the sequences of an initial (seed) multiple sequence alignment (heuristic methods such as T-Coffee (64) and Clustal Omega (65).

The information contained in a MSA can be used to detect new members of a protein family by the use of position-specific scoring matrices as in PSI-BLAST (59)

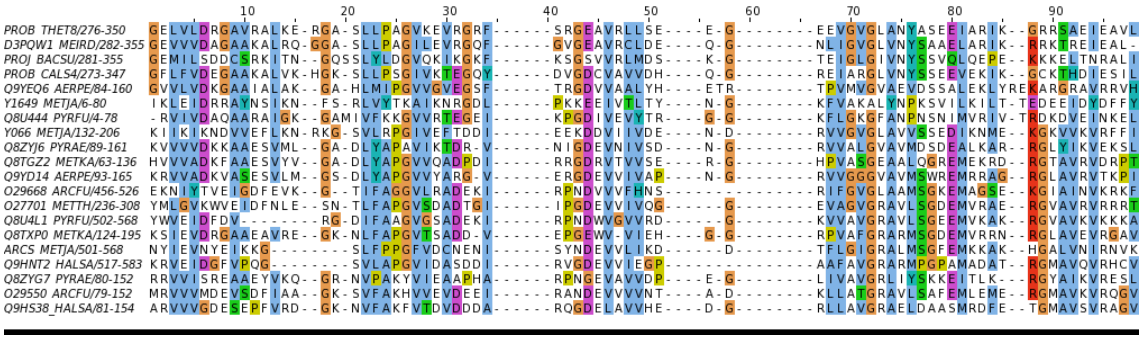

Figure 3.5: *Part of the MSA of the family PF01472 (PUA) in Pfam (50) viewed by the Java software Jalview (66). One can easily observe that, while some positions strongly vary, others are almost perfectly conserved.*

or hidden Markov models as in HMMer (7). This last tool is the one used to classify protein domains into families in the renowned database Pfam (50). See figure 3.5 for an example of MSA obtained from the Pfam website. In the next section we describe more in detail the Pfam database.

3.1.2 Protein families and the Pfam database

Protein domains are conserved parts of protein sequences and tertiary structures with the capability to evolve, perform a function, and exist independently of the rest of the protein chain. Domains normally build compact three-dimensional structures and can be in many cases independently stable and folded (see Figure 3.6). Proteins may consist of several structural domains, and on the other hand the same domain can show up in a number of different proteins: they represent the building blocks of molecular evolution, and can be arranged in diverse ways to create proteins with different functions.

The number of available sequences is becoming increasingly large, and it is fundamental to develop protocols for their organization. The key challenge is to simultaneously satisfy two conflicting requirements: completeness on the one hand, and quality on the other. Despite the large numbers at stake, new sequences often show significant similarity to proteins with known function; thus for classifying new sequences and manage those already known, it is advantageous to organize them into families and use multiple alignment based approaches. The first step to this aim is to define the clusters, i.e. a list of members for each family, while the second one is to build a

Figure 3.6: *Crystal structure of a PUA domain (APE0525) from the Aeropyrum pernix K1 (tartrate complex) (50)*

multiple sequence alignment on them; due to the high number of sequences to align, a partially automated approach is necessary, but if a fully automated cluster partition is done based on sequence similarity the sensitivity of the search is compromised.
In (50) a combination of manual and automatic approaches is adopted. The key ingredient is to make use of two alignments: a high quality *seed* alignment, and a *full* alignment, based on the alignment of the members to a Hidden Markov Model (HMM) derived from the seed.
Hidden Markov Models are probabilistic models able to exploit site-specific information. They consist of a linear chain of match, delete and insert states; the transition probabilities between them can be estimated from a multiple alignment. A given sequence could in principle be generated following different paths along the states: some of them have higher probability, or score, others lower. A protein sequence can be 'aligned to an HMM', namely it is possible to find its most probable path through the states, by using dynamic programming.

Seed and Full alignments together with the corresponding HMM are what make up the database Pfam-A. The purpose is to construct a seed alignment for each family from a non redundant representative set of full-length domain sequences; it is important that such representatives truly belong to the family, and the quality of each alignment is manually checked. From each seed alignment an HMM is built, and then compared with all the sequences in the protein databases Swissprot and TrEMBL (68). In this way sequences are assigned to families; a further grouping is given by protein *clans*, meaning sets of related families that share a single evolutionary origin,

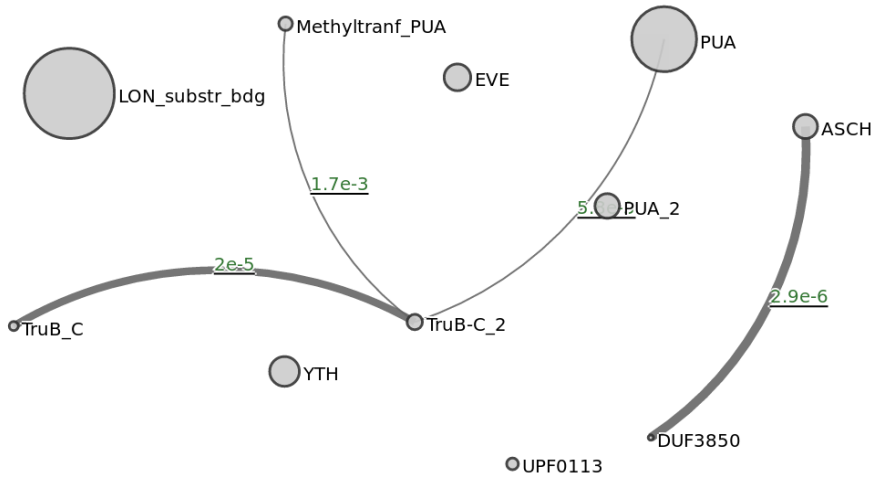

Figure 3.7: *Clan relationships of PUA clan (CL0178). Relationships between families in a clan are determined using HHsearch (67). Families are deemed to be closely related if their E-value is less than* 10^{-3} *and these relationships are shown with a solid line. Less closely related family pairs, with an E-value of between* 10^{-3} *and* 10^{-1}*, are shown with a dashed line. (50)*

as confirmed by structural, functional, sequence and HMM comparisons.

3.2 Methods

In this section we describe in detail the procedure we developed to compute the intrinsic dimension of protein families. In particular, we address the following issues:

- Data preprocessing
- Definition of a distance between sequences
- ID computation

3.2.1 Data preprocessing

The datasets we analyze are obtained by downloading the FASTA sequences of full families from the Pfam website (50). Sequences datasets are subject to under sampling and correlations; in fact, the set of sequences appearing in databases naturally inherits a phylogenetic correlation structure from the species to which they belong to, although it is well known that evolutionary mechanisms such as

horizontal gene transfer makes phylogenetic structure at organism level much different to that at the single sequence level. Moreover, not all species or taxa, have been equally sequenced. For instance, for obvious medical reasons, typically known human pathogens tend to be sequenced much thoroughly than other organisms. Taken together all these biases make the space of protein sequences available unevenly distributed. A common way to mitigate these effects is to impose a cutoff on the sequence similarity: in this way sequences that are similar over a certain threshold are considered correlated and only a representative of the group is kept. To this purpose a powerful tool is CD-HIT (69), (70), a widely used program for clustering biological sequences to reduce sequence redundancy.

CD-HIT. CD-HIT is based on short word filtering (71), a fast algorithm that is able to compare two sequences by breaking them into short manageable pieces. The idea behind short word filtering is that the minimum number of identical short substrings, called 'words', such as dipeptides, tripeptides and so on, shared by two proteins can be expressed as a function of their sequence similarity. In this way it is possible to effectively estimate that the similarity of two sequences is below a certain threshold by simple word counting and without an actual sequence alignment. Clustering is then performed by a greedy incremental algorithm. Sequences are first sorted in order of decreasing length, then the longest sequence becomes the representative of the first cluster; each remaining sequence is compared with the representative of the first cluster, and if their similarity is above a given threshold, it is grouped into that cluster. Otherwise, a new cluster is defined with that sequence as representative. This procedure is carried out in a recursive way through all the sequences in the dataset. For each sequence comparison, short word filtering is applied to the sequences to confirm whether the similarity is below the clustering threshold. If this cannot be established, an actual sequence alignment is performed.

In our case we want to cluster sequences with a relatively high sequence identity in order to clean the dataset from correlated entries without losing important features (an excessively harsh dataset reduction would accentuate the problem of under sampling). Therefore, we run CD-HIT on our FASTA sequences datasets with a c value of 0.8, meaning that the threshold on the sequence identity is 80%. We adopted the suggested word size, that for thresholds $0.7 \sim 1.0$ is 5 as large-scale statistical analysis confirmed that at high sequence identity protein sequences still have statistically significant number of common pentapeptides. Even if the threshold

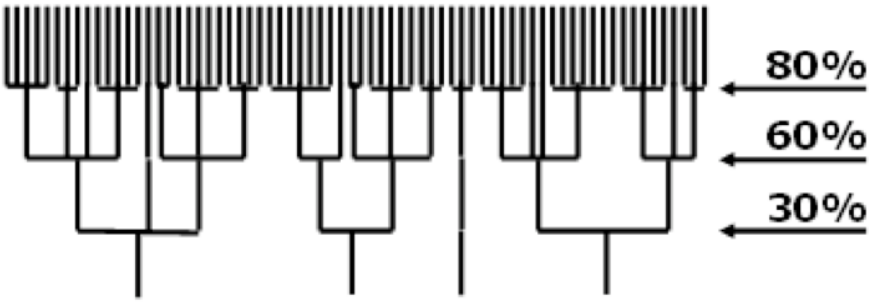

Figure 3.8: *Schematic representation of cd-hit thresholds.*

is relatively high, filtering the dataset through CD-HIT leads to a significant size reduction: for instance, pfam family DnaJ originally counts 40776 sequences, while after CD-HIT clustering the dataset contains 18030 sequences.

3.2.2 Definition of a Modified Hamming distance between sequences

Once clustered the correlated entries by CD-HIT, we have to define a notion of dissimilarity between protein sequences. Several methods are available in the literature to estimate pairwise sequence distances (72), and it is necessary to make a careful choice; our definition of distance in fact has to fulfill some fundamental requirements in order to describe a set of relationships between sequences where operations as ID estimation, and later density estimation and clustering are well defined.

First of all we want our definition of dissimilarity Δ to resemble as much as possible a metric, meaning that if s_1, s_2 and s_3 are three sequences it has to satisfy the following requirements:

1. $\Delta(s_1, s_2) \geq 0$ non-negativity or separation axiom
2. $\Delta(s_1, s_2) = 0 \Leftrightarrow s_1 = s_2$ identity of indiscernibles
3. $\Delta(s_1, s_2) = \Delta(s_2, s_1)$ symmetry
4. $\Delta(s_1, s_2) + \Delta(s_2, s_3) \geq \Delta(s_1, s_3)$ triangular inequality (TI).

The Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different, or the minimum number

of substitutions required to change one string into the other, and it is an example of metric. While the first three points in the metric definition are trivial prerequisites (and define the so called *definite dissimilarity*) the fourth one is crucial and not always met (nor checked) in the definition of dissimilarity between sequences. If the triplet $\{s_1, s_2, s_3\}$ violates the triangular inequality it is possible to have two objects s_1 s_2 that are very similar to a third object s_3 yet very dissimilar to each other. If a clustering is performed on such dataset, the definition of clusters may be not robust and sensitive to the order in the assignation. In (73) the impact of using non-metric dissimilarities on the quality of clustering is investigated in the case of medoid-based clustering algorithms (often employed in the framework of protein sequences analysis).
Another requisite we prescribe is that the notion of dissimilarity between two sequences depends only on the sequences themselves. For instance, a measure of dissimilarity between sequences that fulfills all the metric requirements is the Hamming distance between the sequences aligned in a multiple sequence alignment; it is clear though that since the MSA is obtained on the full dataset the distance between two entries depends in principle on all the other sequences in the dataset. In this situation, a simple operation as adding new entries to the dataset could change the overall distribution of distances. For this reason our definition of dissimilarity will rely only on pairwise sequence alignments.
When dealing with large datasets speed is often an issue, so that the third requirement we make to our distance definition is to be characterized by a fast implementation. Correct alignment algorithms are expensive, with a complexity of the order of the product of the sequence lengths; this motivates the use of heuristic approaches that sacrifice some sensitivity, in the sense that they could miss the best scoring alignment, yet allow a large scale search. We perform pairwise alignment between sequences with BLAST (see section 3.1.1), one of the best-known heuristic algorithms.

Modified Hamming Distance. At this point we are ready to define our notion of distance between protein sequences. So far we described how to filter out correlated entries by means of CD-HIT. The next step is to perform an all against all pairwise alignment of the remaining sequences by BLAST. Since to our purposes we only need to compute distances between close neighbors (TWO-NN makes use of the first two distances only, see Chapter 2)) we set an $E - value$ threshold of 10. for the query, so as to limit the search space to sequences that are relatively close. Then, if for sequences S_1 and S_2 two (or more) relevant MSPs are found, the one is retained with

lowest E-value; in the same fashion, if the alignment between s_1 and s_2 is different from the one obtained between s_2 and s_1, we choose the best one according to the E-value.
Once retained the best MSPs for each couple of sequences, we can extract the percent identity of the aligned parts. This is a very simple measure of similarity, corresponding to the Hamming distance between the aligned hits. The alignment though has not the same meaning for two sequences with approximately the same length and for sequences with very different lengths, nor if the not aligned part is long or short compared to the aligned one. Imagine that we aligned two sequences s_1 and s_2 with lengths m_1 and m_2 respectively, and that the aligned part has length L and percent identity P. If $m_1 \sim m_2$ and P is high, then we know that the two sequences are very similar; but for the same value of P if $m_2 \sim 10 \times m_1$, s_1 and s_2 should be considered far, since only a small part of m_2 was aligned. The point is that in the concept of percent identity the parameter L corresponding to the alignment length is not explicit. A percent identity $P = 90$ can occur because of a difference of 10 aminoacids over a length $L = 100$ as well as in the case of 100 different aminoacids over a length $L = 1000$; in general longer alignments are preferable, so that we want couples of sequences characterized by large values of L compared to the lengths of the sequences to be considered closer. This observation suggests adding a negative bias for not aligned parts.
Moreover, by construction we have to deal with a sparse matrix, as to speed up the process we set an E-value threshold over which BLAST drops the computation of the alignment. This means that for a large number of sequences (that we consider "far enough" from each other) the pairwise alignment is not performed, and we have to set a default value for the distance; This value should be higher than all the distances actually deriving from a proper alignment. In order to be consistent with this requirement, our notion of distance is forced to be bounded from above: in fact, if we could have sequences s_1 and s_2 at an arbitrarily large distance, it would be impossible to define a proper default value for "far" sequences.
Therefore we define the Modified Hamming distance as:

$$d_{MH}(s_1, s_2) = \begin{cases} \frac{m - L \times \frac{P}{100}}{m} & \text{if E-value}(s_1, s_2) < 10. \\ 10 & \text{otherwise} \end{cases} \tag{3.2}$$

Here by E-value(s_1, s_2) we indicate the E-value of the alignment between s_1 and s_2,

while $m = \max\{m_1, m_2\}$ is the maximum between the two sequences lengths. A schematic representation of the Modified Hamming distance is given in Figure 3.9.

Figure 3.9: *Schematic representation of an alignment of sequences with different lengths. In the Modified Hamming distance matches are assigned score* 0*, while mismatches are assigned score* +1*; not aligned parts are considered mismatches; the score obtained in this way is then normalized by the largest of the two sequences lengths.*

Note that by definition the distances lower than 10., that is to say those actually deriving from an alignment, are bounded above by 1; in fact in the extreme (only hypothetical) case of an alignment where only mismatches appear $P = 0$ and the numerator and the denominator in 3.2.2 are equal.

Up to now we improperly referred to the Modified Hamming as a distance, but there is no guarantee that it fulfills all the requirements to be a metric. So far we can only consider it a similarity measure, and ask ourselves to what extent it could be considered a metric.
Non negativity holds since $L \times \frac{P}{100} \leq m$. The identity of indiscernibles is satisfied, since if $d_{MH}(s_1, s_2) = 0$, then the sequences have the same length as the alignment $m_1 = m_2 = L$ and the percent identity is $P = 100$, so that $s_1 = s_2$. The opposite implication is also easily verified. Symmetry is fulfilled by construction. The tricky point is to verify that the triangular inequality holds; in fact, though the Hamming distance between sequences of the same length is a metric, the operations behind the Modified Hamming are as complicated as a pairwise sequence alignment and a rescaling whose amount varies according to each pair of sequences: there is no a priori reason to believe that the triangular inequality is fulfilled. The only way to investigate this property is by empirically counting the percent of violations. We analyzed all significant triplets of sequences in a number of different pfamA families to monitor the percent of violations. We excluded from the analysis triplets including the maximum default distance.
The results obtained confirm that Modified Hamming is susceptible to TI violations but in such a small amount that it can be considered a metric. For instance on

pfamA family DnaJ the number of violations is 645 over a number of proper triplets of 686820043, thus only $\sim 9 \times 10^{-5}\%$ of the entries are involved. A possible strategy is to remove sequences that violate the TI, but we believe that the percent is low enough to consider the Modified Hamming a metric.

3.2.3 Other notions of distance between sequences

Defining a 'good' distance between points is a crucial step to compute the intrinsic dimension of the space. Note that in general under different metrics the ID can change, and if the dataset is not representable in terms of coordinates (as in the case of protein sequences) the space itself is fully described by a set of pairwise distances. In a formal context two metrics d and $\tilde{d}$ are said to be *equivalent* if and only if there exists a finite positive constant C such that

$$\frac{1}{C}d(x,y) \leq \tilde{d}(x,y) \leq Cd(x,y). \tag{3.3}$$

It is a known fact that fractal dimension is unchanged when the metric is altered to an equivalent metric (74). In the case of proteins, dissimilarity measures cannot be described by a clear functional form, and in many cases they do not satisfy the triangular inequality: in this scenario defining a formal equivalence between metrics is not feasible. Still, when comparing two notions of metrics it is possible to look at their correlation plot to infer their equivalence; in fact given a dataset of points $\{x_i\}_{i=1,\dots,N}$ and two metrics d and $\tilde{d}$ defined on it, if the correlation plot is good there are two constants A and B such that $A < \tan \frac{d(x_i,y_i)}{\tilde{d}(x_i,y_i)} < B$: this leads to the inequality

$$\arctan A \times \tilde{d}(x_i,y_i) < d(x_i,y_i) < \arctan B \times \tilde{d}(x_i,y_i), \tag{3.4}$$

saying that in case of a qualitatively good enough correlation the two metrics are equivalent and as a consequence the fractal dimension is conserved. Thus, even if we have at our disposal only the finite set of distances defined on couples of sequences, we expect that in case of a good correlation the ID will not vary; this means that the intrinsic dimension is not only an attribute of a notion of distance, but rather of a class of distances associated to each other in terms of correlation.

In this section we describe the most common and significant definitions of dissimi-

larity between sequences, investigate the extent to which they can be considered a metric, and describe their relationships with the Modified Hamming distance d_{MH}. We focus on the following four dissimilarity measures (72), (75):

- p distance (or uncorrected distance)
- Kimura distance
- Jukes Cantor distance
- Edit/Levinstein distance
- BLOSUM distance

p-distance. Given two sequences s_1 and s_2 the most basic notion of distances between them is obtained by aligning the full sequences in an MSA and computing the Hamming distance on the strings; this corresponds to counting the number of mismatches n_m: the higher is the number, the larger is the distance. If a set of sequences is characterized by a uniform length, as for instance in a multiple sequence alignment, then the Hamming distance is a true metric. A MSA is characterized by columns of gaps that should be eliminated in order to take into account only aminoacids substitutions; besides that, the number n_m has different meanings in the case of very short sequences and in the case of long ones, so that it can be convenient to rely on a proportion of matches rather than on absolute numbers. The uncorrected distance or p-distance is thus given by:

$$d_p(s_1, s_2) = \frac{n_m}{L}, \tag{3.5}$$

where L is the number of amino acids actually compared (meaning the length of the alignment minus the number of columns showing only gaps). The p-distance was initially defined to build phylogenetic trees and is usually computed on multiple sequence alignments: thus given two sequences s_1 and s_2 their distance depends not only on their amino acids sequences but also on all the other entries in the alignment. The p-distance does not fulfill the triangular inequality, even if the number of violations is small. Indeed, we computed the p-distance between the aligned FASTA sequences of pfam family PUA by means of the software EMBOSS distmat (76) and retrieved a 1.5% of TI violations over the whole number of triangles. The p-distance shows a very bad correlation with d_{MH} due to the fact that the former is computed on a MSA while the latter makes use of pairwise alignments.

Kimura distance. A variation of the p-distance is the Kimura distance (77), originally developed for nucleotides and later extended to amino acids. This distance is based the knowledge of the dynamic process underlying the sequence modification, and accounts for the possibility of multiple substitutions at the same site. The Kimura distance between two sequences s_1 and s_2 is given by:

$$d_K(s_1, s_2) = -\ln(1 - d_p - 0.2 \times d_p^2). \tag{3.6}$$

Note that the Kimura is a function of the uncorrected distance, so that the correlation between Kimura and d_{MH} is again bad. We checked the number of TI violations on the same dataset as before, and the result is a 9% of violations, a number that is rather significant.

Jukes Cantor distance. Another variant of the p-distance is the Jukes Cantor, defined in (78).

$$d_{JC}(s_1, s_2) = -\frac{19}{20}\ln(1 - d_p - \frac{20}{19} \times d_p). \tag{3.7}$$

We verified that also in this case the number of violations of the triangular inequality is high; as before we employed the software EMBOSS distmat of PUA family and obtained a 58% of violations.

Edit/Levinstein distance. In information theory the Levenshtein (or Edit) distance is a string metric for measuring the difference between two sequences (see (79)). Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. Edit distance in its basic formulation is a true metric, and the TI can be formally proved. A crucial drawback of such metric is that its computation is extremely slow; even if it is usually carried out by means of dynamic programming techniques, the time complexity of such algorithms is $O(L_1 L_2)$, where L_1 and L_2 is the lengths of s_1 and s_2 respectively; if one aims at calculating pairwise distances between N sequences the time complexity is $O(N^2 L^2)$ where L is the average sequence length. The Edit distance seems the most appropriate for our purposes: it allows comparing two sequences at a time avoiding multiple sequence alignments, it is appealing from a biological point of view and thus one of the most exploited in the literature. Finally, and most importantly, it is a true distance. It has though the drawback of being too slow to be a realistic tool for computing similarities in large datasets.

We verified that the Edit distance is not well correlated with d_{MH} due to the fact that in the Modified Hamming we introduce a normalization factor corresponding to the length of the alignment; but if instead of using plain Edit we normalize it by the average length of the sequences, we note that the correlation is remarkably good at low distances (see Figure 3.10). Observe that since TWO-NN takes into account only the first two distances of every point the range of short distances is exactly the one we are interested in.

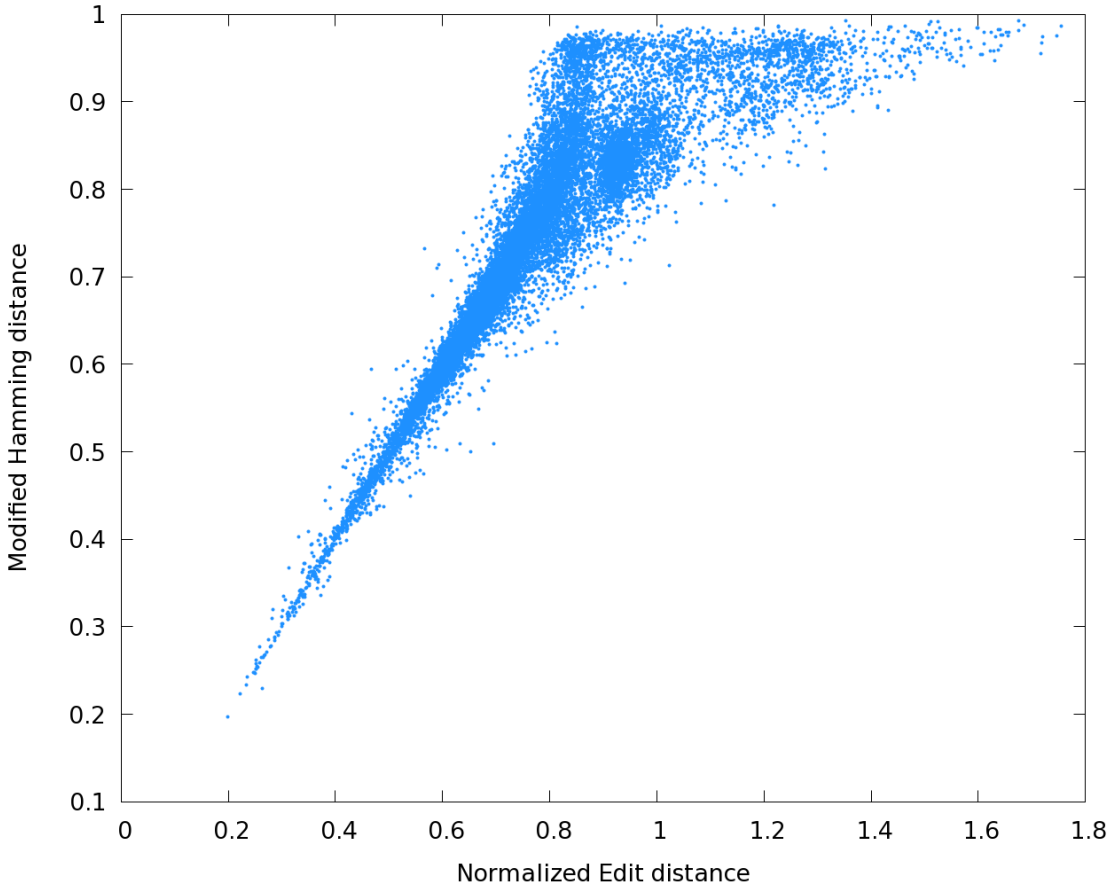

Figure 3.10: *Correlation plot between normalized Edit distances and d_{MH} distances for Pfam family PUA.*

We empirically verified that the normalized Edit is a metric (the number of violations to the TI encountered in case of normalized Edit distance was null). The correlation between d_{MH} and the normalized Edit allows considering the two distances equivalent for our purposes, and in particular we expect them to describe the same space; in fact, we verified that they to provide the same ID measure. The observations above

lead to the conclusion that, to our purposes, the Modified Hamming distance is to all extents a fast version of the normalized Edit.

BLOSUM distance. The Modified Hamming distance is a discrete distance, meaning that due to its definition values arbitrarily near to zero cannot be reached, and there is a positive minimum step between possible distance values. If one measures the distances between a set of sequences with different length the normalization factor introduces a 'fragmentation' of the distances, noticeably reducing the step and approximating a continuous notion of dissimilarity. If instead the distances are computed on a set of sequences sharing the same sequence length L, for instance sequences aligned in a MSA, or artificially generated according to a HMM, the minimum step is $\frac{1}{L}$: this means that for each sequence it is likely to have many neighbors at the same distance, breaking the hypothesis of the TWO-NN model as well as any claim of continuity. The root of the problem is the fact that the Hamming distance weights $+1$ matches and -1 mismatches, so that a natural idea to explore more distances values is to assign weights based on a score matrix. We can define a BLOSUM distance, in short d_{BL}, following exactly the same steps as in the case of the Modified Hamming distance up to the pairwise alignment; at this point, instead of extracting from the BLAST output the percent similarity of the alignment, and thus the Hamming distance of the aligned part, we can focus on the bitscore, that is based on the score matrix BLOSUM62. Given two sequences s_1 and s_2 we define the BLOSUM distance as:

$$d_{BL}(s_1, s_2) = \begin{cases} \frac{M-S}{M} & \text{if E-value}(s_1, s_2) < 10. \\ 10 & \text{otherwise} \end{cases} \tag{3.8}$$

Here M is the maximum bitscore between S_1 against itself and S_2 against itself, and S is the best bitscore for the alignment. Note that definition 3.2.3 strongly reminds of definition 3.2.2, and it can be thought as the same object in which matches and mismatches are counted as many times as prescribed by the BLOSUM62 matrix. The BLOSUM distance is correlated with d_{MH}, as shown in figure 3.11, meaning that they describe the same space and predict the same dimension when the distances are approximately continuous; instead, in case of extremely discretized distances d_{BL} allows computing the intrinsic dimension, while d_{MH} is not suitable for the task due to the high number of coincident distances.
To conclude this section, the normalized Edit distance, d_{MH} and d_{BL} are distances

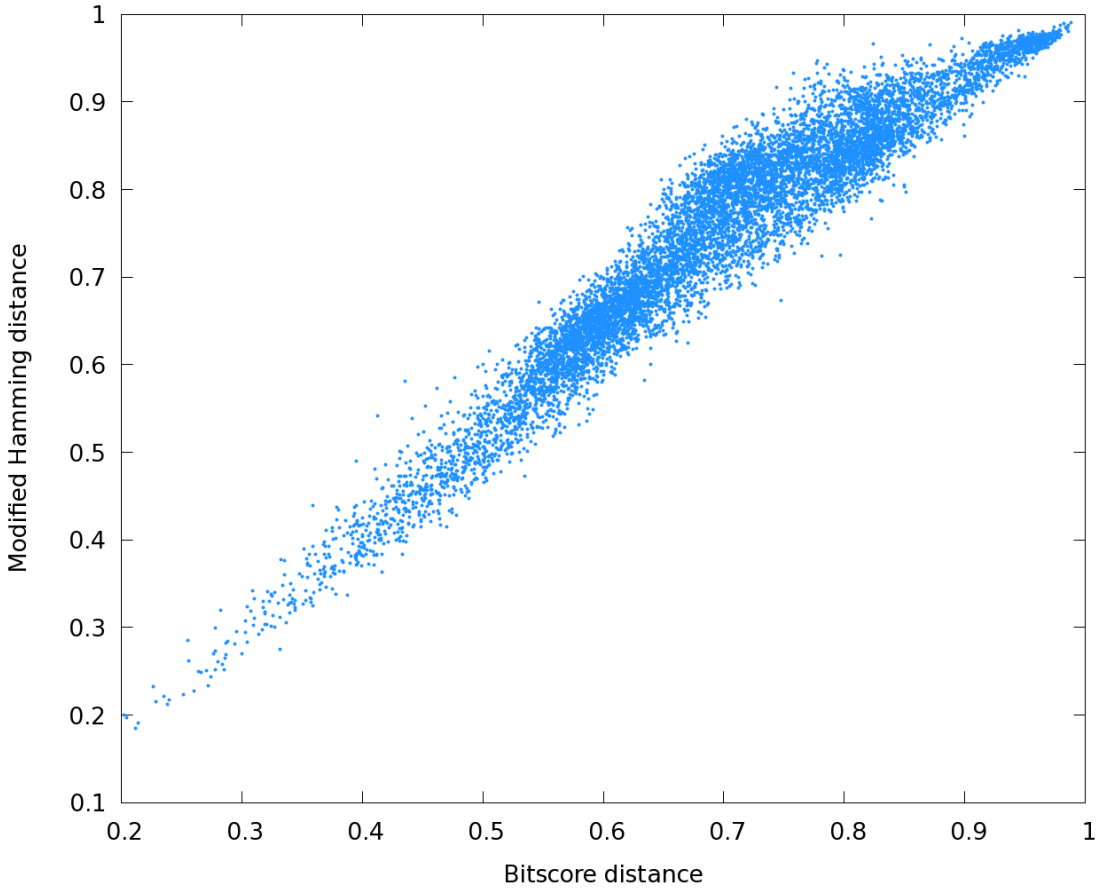

Figure 3.11: *Correlation plot between d_{BL} and d_{MH} distances for Pfam family PUA.*

related to each other, and they provide roughly the same information about the space, but while computing the Edit distance is extremely slow, d_{MH} and d_{BL} are fast; finally, d_{BL} is a finer version of d_{MH}.

3.2.4 ID computation

.

In the previous section we described a measure of similarity, the Modified Hamming distance, that can be considered to a good extent a metric. In this section we want to set up a procedure to estimate the intrinsic dimension of protein families where distances between sequences are defined according to d_{MH}. Even if d_{MH} is to a good approximation a metric, and thus a basic requirement of TWO-NN is fulfilled, it has an upper bound $u = 1$ that may induce artificial inhomogeneities in the space. Recall that TWO-NN method (Chapter 2) is rooted on the computation,

for each point $\mathbf{x}$ in the dataset, of its first and second nearest neighbors distances r_1 and r_2; the ratios $\mu(\mathbf{x}) \doteq \frac{r_2(\mathbf{x})}{r_1(\mathbf{x}))}$ are collected to provide a measure of ID by a fitting procedure; if the hypothesis of the method are fulfilled, the set S given by $S = \{(log(\mu_i), -log(1 - F^{emp}(\mu_i))) \quad | \quad i = 1, ..., N\}$ is well fitted by a straight line whose slope corresponds to the intrinsic dimension of the dataset.
If we blindly apply TWO-NN to a dataset of d_{MH} distances obtained on Pfam family DnaJ we obtain the S set in figure 3.12.

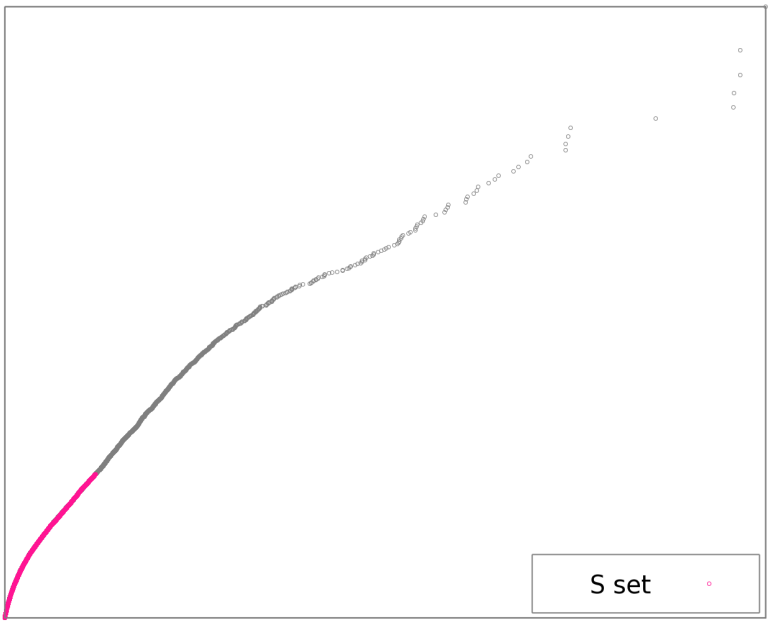

Figure 3.12: *S set for the Pfam family DnaJ*

If the hypothesis of local uniformity was fulfilled together with that of a constant dimensionality along the dataset, then S set should be a line; clearly this does not happen. This effect could be caused by the presence of manifolds with different dimensionalities in the dataset, or it could either be a consequence of the upper bound interference. To discriminate between the two possibilities, first we observe that the upper bound has an influence on the measure only if the neighborhoods of points defined by their second neighbors are affected by such threshold. So if we could restrict our measure to neighborhoods whose radius r is relatively smaller than u, we should be able to wash away the effects of the upper bound. We proceed as follows.

Denote by A the set of values for μ obtained on the whole dataset of d_{MH} distances on family DnaJ. For different values of $\bar{r}$ we retain sets of μ values $A_{\bar{r}} \subset A$ such that

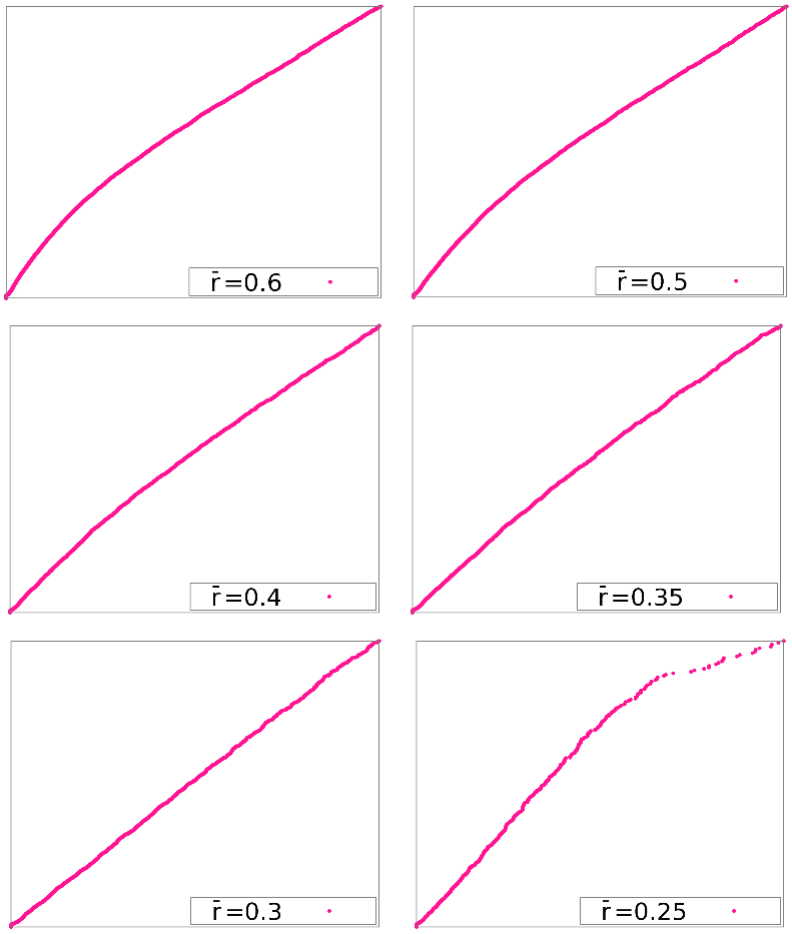

Figure 3.13: *$S_{\bar{r}}$ for different values of $\bar{r}$.*

a value $\mu = \frac{r_2}{r_1}$ belongs to $A_{\bar{r}}$ if and only if $r_2 \leq \bar{r}$. In symbols:

$$A_{\bar{r}} \doteq \{ \quad \mu \in A \quad \| \quad r_2 < \bar{r} \quad \}. \tag{3.9}$$

For each set $A_{\bar{r}}$ we compute the ID by means of TWO-NN and monitor the related S set $S_{\bar{r}}$. In Figure 3.13 we plot $S_{\bar{r}}$ for six different values of $\bar{r} = 0.25, 0.3, 0.35, 0.4, 0.5, 0.6$. It is clear that as $\bar{r}$ decreases towards the value $\bar{r} = 0.3$ the pronounced curvature that is visible in the original S set as well as in the reduced set $S_{0.6}$ starts to diminish, and $S_{0.3}$ can be very well fitted by a straight line. For lower values of $\bar{r}$ other effects become visible as we begin to see the effects of a new boundary, this time represented by $\bar{r}$. We see that our hypotheses are fulfilled within a small range of $\bar{r}$ in which r_2 is far enough from the boundary, yet $\bar{r}$ is large enough not to influence the μ distribution.

To find out the optimal range for $\bar{r}$ in a quantitative way we can, for each value of $\bar{r} = 0.25, 0.3, 0.35, 0.4, 0.5, 0.6$, compute the Root-Mean-Square Deviation (RMSD) of the fit to a straight line. The best choice for $\bar{r}$ is the one minimizing the RMSD.

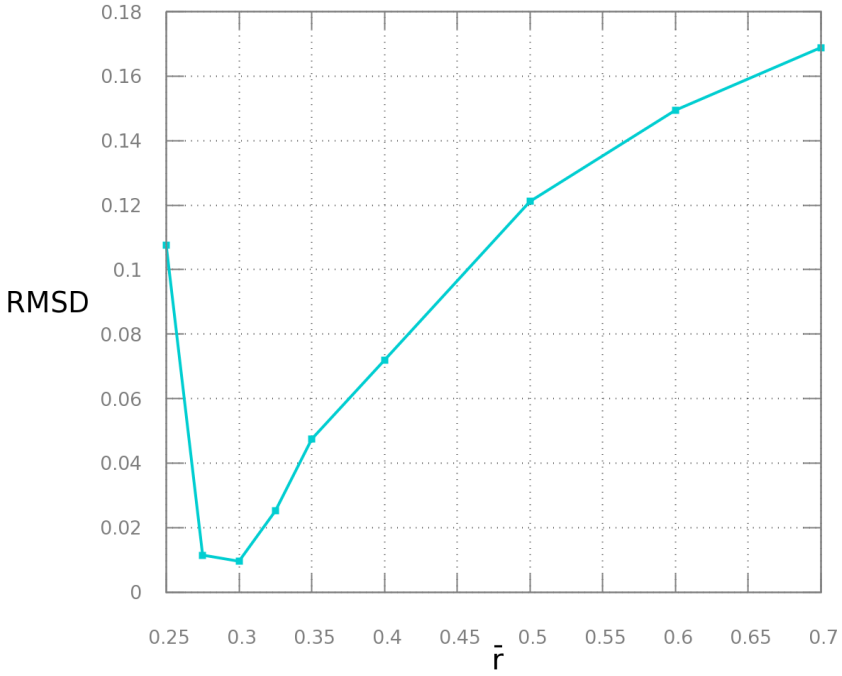

Figure 3.14: *RMSD vs $\bar{r}$ for Pfam family DnaJ.*

In Figure 3.14 we plot the RMSD vs $\bar{r}$; the minimum is well defined and located at a value of 0.3. In the same fashion, we can plot the intrinsic dimension obtained by restricting the fit to sets $A_{\bar{r}}$, as shown in Figure 3.15: the ID corresponding to $\bar{r} = 0.3$ is ~ 9.
The discussion above provides a procedure to compute the intrinsic dimension of protein families in a general framework. The steps are the following:

- Cluster sequences by sequence similarity through CD-HIT in order to obtain a reduced dataset where only cluster representatives are present: in this way the effects of correlation are reduced

- Use BLAST to perform pairwise alignments and compute the Modified Hamming distance between sequences. The result is a sparse matrix where very high distances are not measured, but set to a default value. This choice speeds up the computation of distances but it introduces a boundary effect.

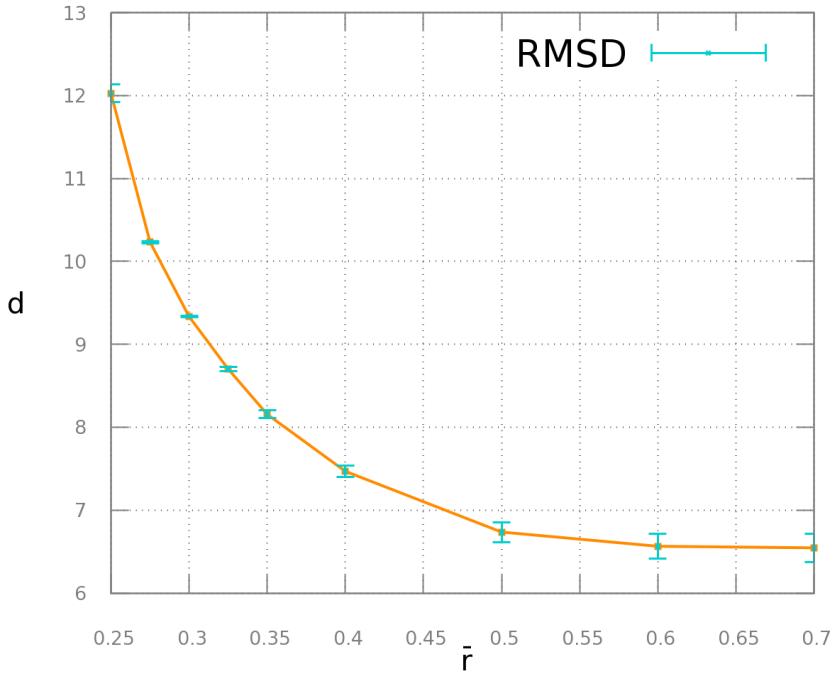

Figure 3.15: *ID vs $\bar{r}$ for the Pfam family DnaJ.*

- Exclude the boundary effects by computing the ID on a reduced dataset whose second nearest neighbors are "far enough" from the boundary, that is to say they are below a threshold $\bar{r}$. To find the optimal upper bound $\bar{r}$ try different values and choose the value corresponding to a set $S_{\bar{r}}$ that minimizes the Root-Mean-Square Deviation, meaning that it is best fitted by a straight line.

In the next section we apply the methods discussed above to the analysis of the ID of several Pfam families, with the aim of shedding some light on the factors that determine a high rather than a low intrinsic dimension in protein families.

3.3 Results

We analyzed several Pfam families belonging to different Pfam clans in order to explore a wide range of cases. The procedure explained in the previous section was first applied on the families of Pfam release 31.0 enumerated in Table 3.1:
The clans appearing in Table 3.1 are very different from each other: clan CL0239 for instance includes the insulin like hormones, while clan CL0192 contains various transmembrane receptors and related proteins. In figure 3.16 we summarize the

Family	Clan
PF08666	CL0489
PF06814	CL0192
PF16177	CL0378
PF07786	CL0316
PF01472	CL0178
PF02984	CL0065
PF00049	CL0239
PF00185	CL0399

Table 3.1: *Pfam familes in Pfam notation together with the respective clans or superfamilies.*

results obtained on these families.

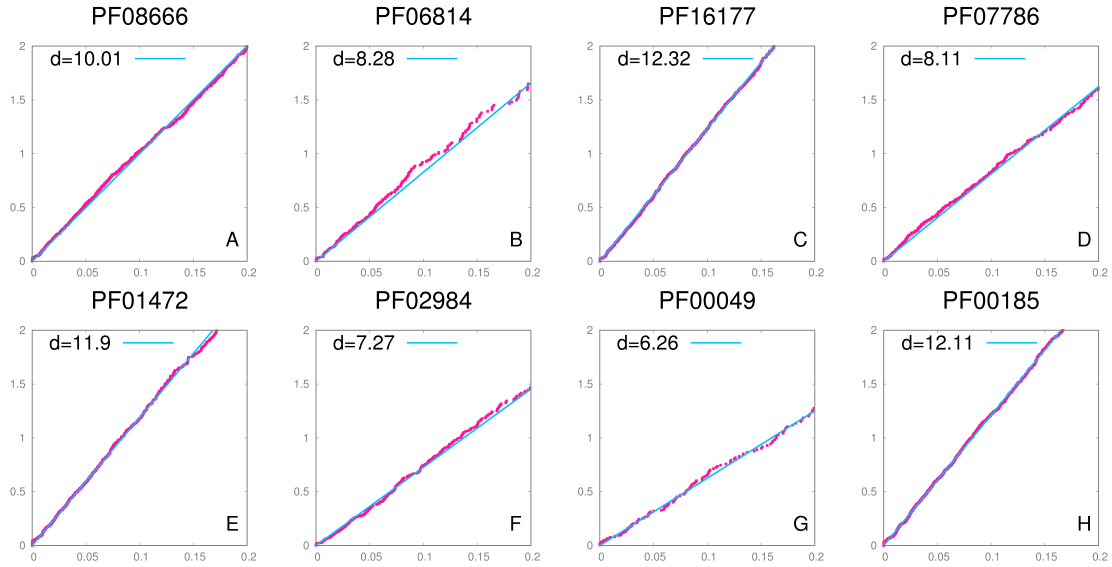

Figure 3.16: *S set and estimated ID for eight Pfam families;*

First of all we note that in all the cases the reduced S-sets are very well fitted by a straight line, meaning that the procedure introduced in this chapter leads to a well defined intrinsic dimension. The slopes of these straight lines, corresponding to the dimensions of the families, span from a value of around 6 to around 12; these values are quite similar and low relatively to the dimension of the embedding space: in fact, representing all the sequences of a family in a vector space would require in principle L coordinates, where L is the maximum sequence length in the family; L is normally of the order of at least 300 in all the cases listed in table 3.1. If we look at the sequence similarity within a family, we find entries sharing only 20% of the

aminoacids so that the number of mutations observed in a family is enormous.
The results obtained on these families are compatible with the analysis in (4); here the authors measure the dimension of a family of sequences in a multiple sequence alignment, using a metric based on the Hamming distance. The estimate obtained for the ID is 5. Even if, as we already pointed out, distances based on MSAs do not correlate with the Modified Hamming distance, and thus describe different spaces, the measure obtained in (4) is not far from the values computed by our procedure: the common feature between our estimates and that obtained by Granata *et al.* is that they are low.

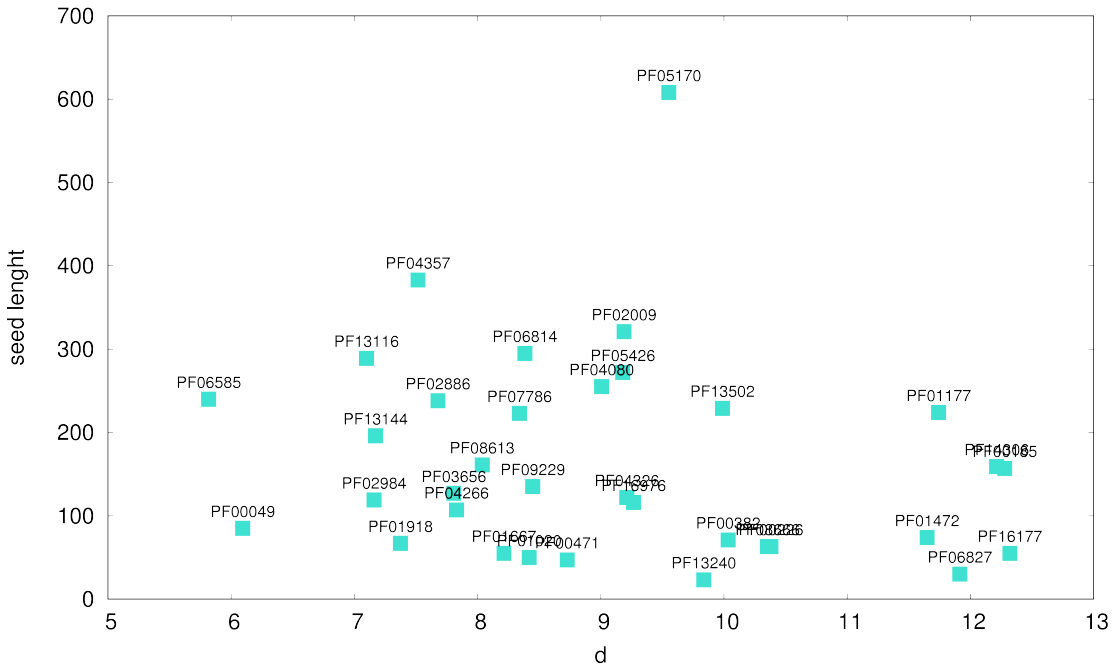

Figure 3.17: *S set and estimated ID for eight Pfam families;*

As observed by the authors, the low ID of the manifold contaning the sequences can be interpreted in terms of allowance for mutations: the evolutionary pressure results in a lack of variations at specific positions and in correlated variations across different positions, both restricting the number of degrees of freedom of the sequences.
So far we have interpreted the ID of protein sequences as a number of degrees of freedom: this number is in principle different in different families, and this is what we observe in our measures. The natural question arising at this point is why we observe this exact range of sizes, what is the meaning of an intrinsic dimension of 8

instead of 6 or 12.

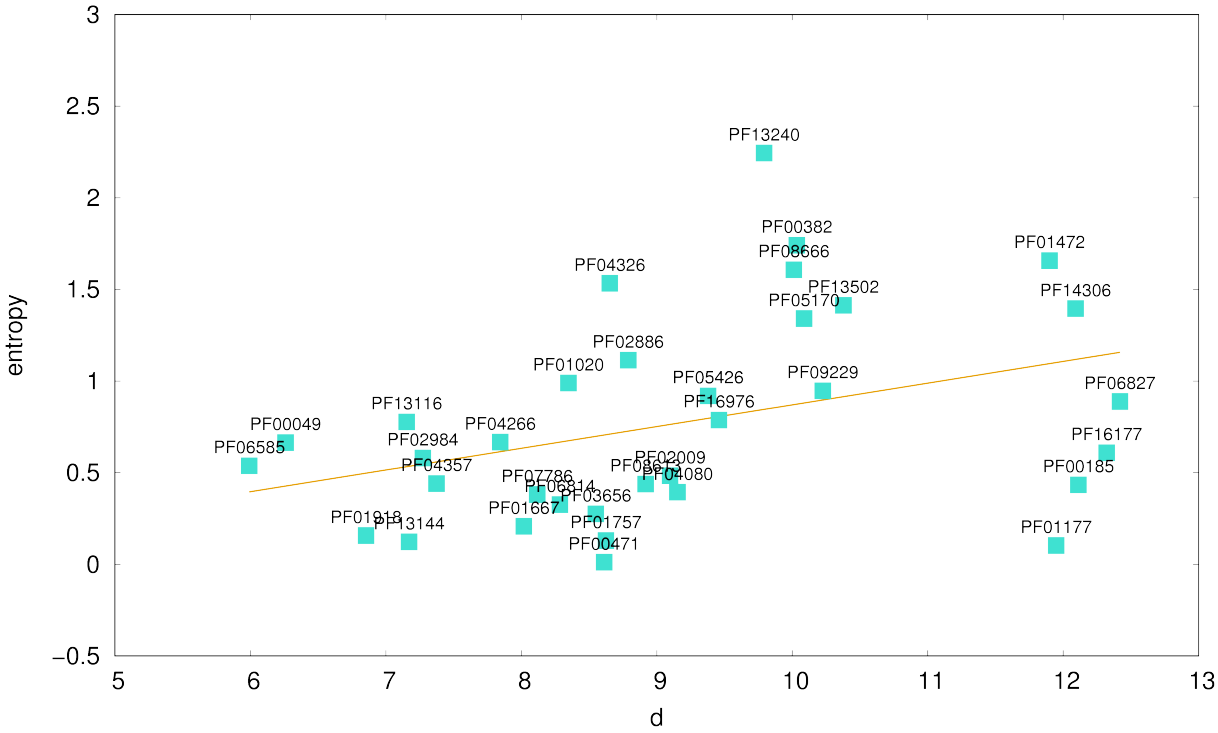

Figure 3.18: *S set and estimated ID for eight Pfam families;*

We first tested the dependence of the ID on the length of the alignment seed of the family. We recall that in Pfam the seed is the MSA of selected sequences believed to capture the salient features of the family, and defines the family itself; the length of the HMM built from this seed is calles seed length. A possible guess is that the longer is the seed MSA length, the larger the D: in fact, longer seeds could allow for larger variability across the family. We compared the seed length with the ID of 36 Pfam families; the result is shown in Figure 3.17: no correlation is visible, and the Spearman coefficient has an R value of 0.17213 with a two-tailed P value is 0.31546. By normal standards, the association between the two variables cannot be considered statistically significant. Thus, the length of the seed is not a relevant parameter to determine the value of the intrinsic dimension.

Another possibility is that the ID, namely the number of independent directions in which the sequences can vary, can be related to domain architectures. Some families show a great variability of architectures, with more of them strongly populated,

while other families are well represented by a single architecture. It is plausible that families with a more complex architectures landscape have a greater variability and thus a larger number of degrees of freedom. We then tested the correlation between the intrinsic dimension and the entropy of the distribution of the domain architectures across the family. To characterize this distribution we computed the entropy

$$S = -\sum_a \frac{N_a}{N} log(\frac{N_a}{N}).$$

Here N is the total number of sequences in the family and N_a is the number of sequences in the family with a given architecture a. The outcome is displayed in Figure 3.18. This time there is evidence of a weak correlation between the two sets of values, with a Spearman R value of 0.4413.
This correlation, though interesting, is not enough to capture the reasons for the ID variability in protein families. We are currently investigating more deeply the topic.

3.3.1 Estimating the ID of artificially generated protein sequences

In this section we study the reliability of artificial generative models for protein sequences from the point of view of the intrinsic dimension; we have seen that the dimension of protein families is relatively low; this is likely to be due to the constraints imposed by the three dimensional structure, that narrow to a great extent the allowed mutations. A virtually perfect generative model should be able to reproduce such constraints, and as a consequence to preserve the low dimensionality of the space defined by the true sequences.

In the previous sections we highlighted the importance of organizing protein sequences into families of homologs in order to make predictions about their folded structure. Statistical models of protein sequence evolution aim at capturing the sequence variability of homologous sequences, unveil statistical constraints acting on such variability and relate this information to the conserved biological structure and function of the proteins in a family (see (80)). The starting point is a Multiple Sequence Alignment (see section 3.1.1) of an entire protein family. We can describe MSAs as rectangular matrices $A = \{a_i^\nu | i = 1, ..., L, \nu = 1, ..., M\}$, where M is the number of sequences, L the length of the alignment and a_i^ν is either one of the 20

amino acids or a gap "-" standing for insertions ar deletions (often represented as the 21^{st} aminoacid), as in Figure 3.5. A viable assumption for modeling the MSA is that it constitutes a sample of a Boltzmann distribution:

$$P(a_1, ..., a_L) = \frac{1}{\mathcal{Z}} \exp\{-\beta \mathcal{H}(a_1, ..., a_L)\}, \tag{3.10}$$

defining a probability for each full-length amino-acid sequence $\mathbf{a} = (a_1, ..., a_L)$ (note that points can be correlated, so the sample is generally not i.i.d.). Inverse statistical physics attempts to reconstruct the Boltzmann distribution P using the MSA A. To perform homology detection, namely to assign a sequence with unknown biological function to a known family of homologous proteins, the classical choice for $\mathcal{H}$ is the following:

$$\mathcal{H}(a_1, ..., a_L) = -\sum_{i=1}^{L} h_i(a_i). \tag{3.11}$$

This formalizes the idea of a local field capturing only site-specific patterns, and is a simplified version of profile HMMs. The assumption of independence in such profile models limits the information that can be extracted from a MSA, since aminoacids at different position do not evolve independently; single-site mutations in fact often perturb the physical properties of the surrounding residues in the folded structure with disruptive effects for the function.

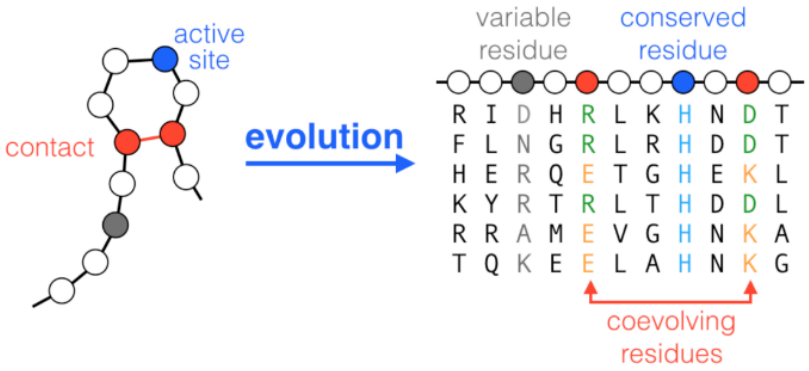

Figure 3.19: *The conservation of contacts leads to the coevolution of structurally neighboring residues and therefore to correlations between the columns of a MSA (here an artificial alignment is shown for illustration). Figure from reference(80).*

Thus, residues in contact *coevolve*, and correlated occurrences of certain aminoacids

in certain sites are visible in the covariation of columns in the MSA (see Figure 3.19); such covariation cannot be used directly to infer coupling, that is to say amino-acids contact in the folded structure, as it can be the result of a much more complicated pattern of contacts than the simple pairwise one. The aim of Direct-Coupling Analysis, or DCA (81), (82), is to explain correlations through a network of direct coevolutionary couplings, expressed in the model as:

$$\mathcal{H}(a_1, ..., a_L) = -\sum_{1 \leq i < j \leq L} J_{ij}(a_i, a_j) - \sum_{i=1}^{L} h_i(a_i). \tag{3.12}$$

Here both local fields h_i and pairwise couplings J_{ij} have to be inferred from the MSA, in such a way that the first and second moment of the Boltzmann distribution correspond to the empirical frequencies; this means that there are the following constraints:

$$f_i(a) = \langle \delta_{a_i,a} \rangle_P \tag{3.13}$$

$$f_{i,j}(a, b) = \langle \delta_{a_i,a} \delta_{a_j,b} \rangle_P, \tag{3.14}$$

where δ is the Kronecker Delta. In practice calculating such averages is complicated, and DCA methods differ in the way they treat such computations.

It turns out that the strongest pairwise couplings provide accurate prediction of contacts between residues, enabling protein-structure prediction. It is then natural to attempt using 3.3.1 to design artificial protein sequences by Monte Carlo sampling. To this purpose, the model needs to be *generative*, in the sense that besides reproducing the statistics of single and pair-columns in the MSA it should provide sequences that are by no means distinguishable from the natural ones. The challenge is thus to mirror by artificial sequences higher order characteristics of natural sequences, which are not explicitly fitted by equation 3.3.1. One of such characteristics, tested in (80), is the set of Hamming distances of sequences to the consensus sequence $(a_1^*, ..., a_L^*)$, defined by the most frequent amino acids $a_i^* = \text{argmax}_a f_i(a_i)$ in the MSA. In Figure 3.20 we show the histograms of Hamming distances from the consensus for natural sequences and artificial ones in the case of Pfam family PF00076; here the DCA method employed to generate arificial sequences is Adaptive Cluster Expansion (ACE) (8), (83), that accurately reproduces the sampled frequencies and correlation

at the cost of a high computational demand.

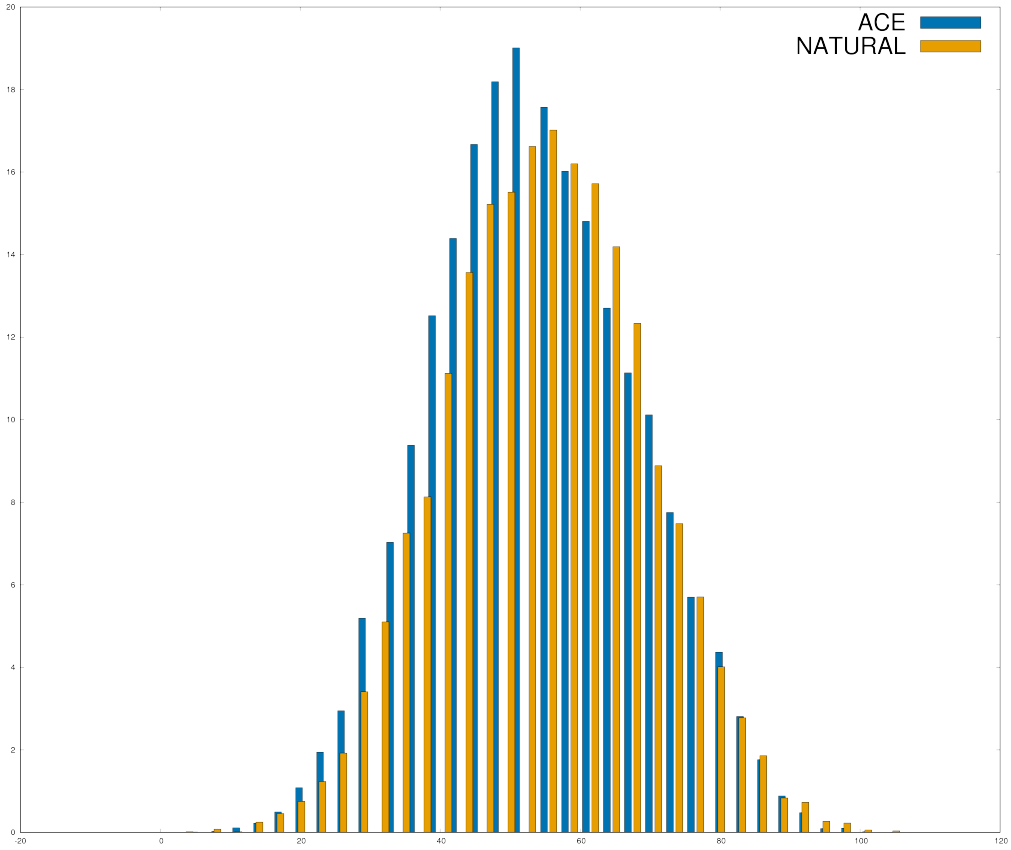

Figure 3.20: *Histograms of Hamming distances of natural and model-generated sequences from Pfam family PF00076.*

The two histograms in Figure 3.20 show that from the point of view of the Hamming distance natural and artificial sequences are indistinguishable.
We then computed the intrinsic dimension of the two datasets; we know that natural sequences lie on implicit low-dimensional manifolds, where low-dimensional means in the range of 6 to 12 (see section 3.3). The ID is a complex function of the data, but a meaningful one, and we suggest that it can be employed in general to assess the goodness of artificial models. In particular, if the ID of artificial sequences is higher than in the natural ones, it can be argued that more constraints are to be imposed on the model; thus ID computation can help not only to establish the invalidity of a model, but to suggest a direction to fix it.
In Figure 3.21 we analyze the ID of sequences generated with HMMER (7) and ACE, and compare it to the ID of natural ones, again in the case of Pfam family PF00076. HMMER employs HMM models, and thus is able to capture only site-specific patterns,

according to equation 3.11. The intrinsic dimension of sequences generated with this model is the highest, with a value of ~ 56, since by construction the constraints related to covariation are mot taken into account. Sequences generated by means of ACE have a lower ID, as a consequence of the couplings, but the dimension is ~ 37, still high with respect to the natural ones; in fact, according to the other Pfam families we analyzed, natural sequences from family PF00076 lie on a manifold with ID ~ 6.

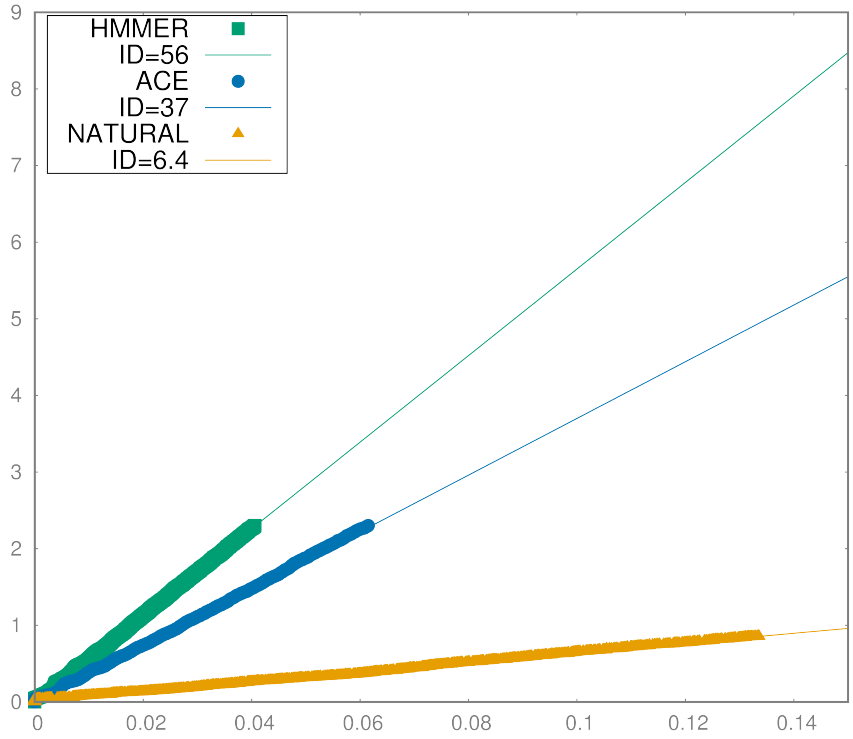

Figure 3.21: *S set and fitting lines for sequences generated with HMMER (7), ACE (8) and for natural ones in the case of Pfam family PF00076.*

These observations suggest that even if Direct Coupling Analysis is able to provide accurate predictions of contacts between residues, and thus to give new insight into protein folding, it is not optimal to design artificial protein sequences; to this task, probably, pairwise couplings and local fields are not sufficient, and a more complex model has to be considered.

Chapter 4

An automatic procedure to estimate the density topography of Pfam protein clans

4.1 Introduction

As pointed out in Chapter 3 , grouping protein sequences into families according to their tertiary structure, and thus function, is of crucial importance to their analysis. Pfam database (50) provides an accurate assignation of sequences to families, and a further grouping of families into clans or superfamilies, in a hierarchical organization. Unfortunately, the construction of such databases is extremely time-consuming as it relies on manual curation (see Chapter 3). An automatic mechanism to partition protein sequences into families would noticeably speed up the procedure.

In this Chapter we describe a way to assign sequences to families in an automatic fashion. The idea is that families correspond to density peaks in the distribution of protein domains in the space described by d_{MH} distances defined in section 3.2. To this purpose, we need three fundamental tools. The first one is an intrinsic dimension estimator providing the ID of the space in which the density has to be computed: we already have at our disposal TWO-NN (see Chapter 2) together with a procedure to determine the ID of protein families (see Chapter 3). The second one is an estimator for the density of the distribution of sequences: in section 4.2 we describe a new density estimator. The third one is a way to automatically group sequences according to their density, namely a clustering algorithm; a new clustering technique is described in section 4.3. Finally, in section 4.4 we illustrate the results

obtained on the Pfam clan PUA: not only the method is able to successfully partition domains into families, but it can detect a further level of complexity, namely that of domain architectures. This result is striking if one takes into account that Pfam provides the sequences of the *single domains* only, so that architectures composed by many domains are 'deduced' by the method from the sequence of just one of them. The clustering procedure thus proves to be able to detect the complexity of a structure by the interaction of the unknow parts with the single domain at disposal.

4.2 A Pointwise Adaptive k-Nearest Neighbors density estimator

In this section we describe a novel approach to compute the density distribution of datasets. The available approaches for estimating the density (84; 85; 86; 87) can be broadly divided in parametric and non-parametric. In parametric methods it is assumed that the underlying density function has a specific functional form, for example a mixture of Gaussians (88); on the contrary in non-parametric methods no strong assumption is made on the functional form of the density function (85; 89; 90). An important non-parametric method is the Kernel estimator (91; 92), where the density is estimated by a sum of kernel functions centered at each point. Another popular non-parametric estimator is the k -Nearest Neighbor estimator (k -NN) (93), where the local density at a point is measured as the ratio of the number of nearest neighbors k to the volume they occupy. A common trait of non-parametric methods is the dependence from a cutoff (usually called smoothing parameter) which determines the length-scale of influence of the single point (86; 89; 94). This cutoff is represented by the width of the kernel function in kernel estimators and by the value of k in k-NN. It is well-known that using a global smoothing parameter in highly inhomogeneous data sets induces systematic errors (see (95; 85) and references therein). In order to address this problem some authors propose to use a smoothing parameter that varies at different data points (96; 89). A point-dependent smoothing parameter can be determined, for example, by optimizing a global measure of discrepancy of the density estimation from the true density (85; 97); however, this global measure is generally a non-linear function of all the data points (98) and its optimization can be computationally demanding for large data sets. A different approach for determining the smoothing parameter is the point adaptive estimation, first proposed by Lepskii (99) and further developed by Spokoiny, Polzehl and other

authors (100; 101; 102; 103; 104; 105). A related approach is introduced in Gach (103) where the density estimator is chosen among a family of Haar wavelets with variable resolution levels. This method is based on the hypothesis that the density is locally constant, but it only takes into consideration univariate density functions. Alternatively, the problem of determining the appropriate size of the neighborhood for computing the density can be addressed by searching the largest region in which the density is isotropic, namely it varies in a similar manner in different directions (106; 107).

Our approach is based on the ideas introduced in these works, and takes advantage of the knowledge of the intrinsic dimension (see Chapter 2) of the dataset at study; in fact, in several real cases the probability distribution is extremely anisotropic, as the data lie on a manifold whose dimension is significatively lower than that of the embedding space; estimating the density without taking into account the real dimension of the space spanned by data points could dramatically affect the accuracy, as density and space dimensionality are tightly linked to each other.
In the next section we introduce a procedure based on finding for each point the size of the neighborhood in which the density is approximately constant. This test is automatically performed only along the "soft" directions, meaning the directions in which the data vary significantly, whose number is determined by the intrinsic dimension.

4.2.1 A Likelihood approach to estimate the density

Let $\{X_1, \ldots, X_N\}$ be a set of N independent and identically distributed random vectors with values in $\mathbb{R}^D$. We assume that the X_i-s lie on a manifold of dimension $d \leq D$, constant in the dataset. We aim to estimate the local density around each point X_i, defined with respect to the Lebesgue measure (108) on the hyperplane of dimension d locally tangent to the manifold in that point (here d indicates the ID of the dataset). Note that we can avoid to project explicitly the points on the manifold, since in a sufficiently small neighborhood of each point the distance on the manifold is well approximated by the Euclidean distance in $\mathbb{R}^D$. We denote by $\{r_{i,l}\}_{l \leq k}$ the sequence of the ordered distances from i of the first k nearest neighbors, so that $r_{i,1}$ is the distance between i and its nearest neighbor, $r_{i,2}$ is the distance with its second nearest neighbor and so on. The volume on the tangent hyperplane

of the hyperspherical shell enclosed between neighbors $l-1$ and l is given by

$$v_{i,l} = \omega_d \left(r_{i,l}^d - r_{i,l-1}^d \right), \tag{4.1}$$

where the proportionality constant ω_d is the volume of the d-sphere with unitary radius. We conventionally take $r_{i,0} = 0$. It can be proved that if the density is constant all the $v_{i,l}$ are independently drawn from an exponential distribution with rate equal to the density ρ (see Appendix A for a derivation). Therefore, the log-likelihood function of the parameter ρ given the observation of the k nearest neighbor distances from point i is:

$$\mathcal{L}(\rho | \{v_{i,l}\}_{l \leq k}) \doteq \mathcal{L}_{i,k}(\rho) = k \cdot \log \rho - \rho \sum_{l=1}^{k} v_{i,l} = k \cdot \log \rho - \rho V_{i,k}, \tag{4.2}$$

where $V_{i,k} = \sum_{l=1}^{k} v_{i,l}$ is the volume of the hypersphere with center at i containing k data points.
By maximizing $\mathcal{L}$ with respect to ρ we find $\rho = k/V_{i,k}$, that is to say the standard k-nearest neighbor estimator (k-NN) of the local density (85). The estimated error on ρ in this case is given by the asymptotic standard deviation of the parameter estimate: $\varepsilon_\rho = \frac{\rho}{\sqrt{k}} = \frac{\sqrt{k}}{V_{i,k}}$.

Finding the optimal number of neighbors

The error of the standard k-NN density estimator by definition gets smaller as k increases. However, when k increases, the density in the neighborhood within a distance $r_{i,k}$ from the data point i can become non constant, inducig systematic errors in the k-NN estimate. To cope with this problem, in our approach we choose as k the largest possible value for which the condition of constant density holds within a given level of confidence. In order to find the optimal k for each point i we compare, for increasing k, the maximum likelihood of two models:

- Model 1 (M1) in which the densities of point i and of its $k+1$-th nearest neighbor are different. We denote by j the index of the $k+1$-th nearest neighbor of point i. The likelihood of M1 is obtained by maximizing $\mathcal{L}_{i,k}(\rho) + \mathcal{L}_{j,k}(\rho')$ with respect to two independent parameters ρ and ρ'. This gives

$$\mathcal{L}_{M1} = \max_{\rho,\rho'} \mathcal{L}_{i,k}(\rho) + \mathcal{L}_{j,k}(\rho') = k \cdot \log \frac{k^2}{V_{i,k} V_{j,k}} - 2 \cdot k.$$

- Model 2 (M2), in which the density at point i is assumed to be equal to the density at its $k+1$-th nearest neighbor of index j. The likelihood of M2 is obtained by maximizing $\mathcal{L}_{i,k}(\rho) + \mathcal{L}_{j,k}(\rho)$ with respect to a single parameter ρ. This gives

$$\mathcal{L}_{M2} = \max_{\rho} \mathcal{L}_{i,k}(\rho) + \mathcal{L}_{j,k}(\rho) = 2 \cdot k \cdot \log \frac{2 \cdot k}{V_{i,k} + V_{j,k}} - 2 \cdot k.$$

The two models are compared by a Likelihood Ratio test (109) which requires to estimate the difference

$$D_k = -2 \cdot (\mathcal{L}_{M2} - \mathcal{L}_{M1}) = -2 \cdot [\log(V_{i,k}) + \log(V_{j,k}) - 2 \cdot \log(V_{i,k} + V_{j,k}) + \log(4)]. \quad (4.3)$$

At this point we consider the two models distinguishable with a high statistical confidence if D_k is larger than a threshold value D_{thr}. Since the numbers of parameters of the two models differ by one, D_k has an approximate χ^2 distribution with one degree of freedom (110). We take $D_{thr} = 23.928$ which corresponds to a p-value of 10^{-7}.

For each point i we choose the optimal value of k, hereafter denoted by $\hat{k}_i$, according to the condition

$$\hat{k}_i : \quad \left(D_k < D_{thr} \; \forall \; k \leq \hat{k}_i\right) \wedge \left(D_{\hat{k}_i+1} \geq D_{thr}\right).$$

This implies that for point i models M1 and M2 are consistent within a p-value of 10^{-7} for all the choices of k smaller or equal to $\hat{k}_i$.

To illustrate this procedure, we consider a sample of 2000 points extracted from a uniform distribution (Fig. 4.1A) and the same sample with 2000 additional points extracted from a Gaussian distribution (Fig. 4.1B). In Fig. 4.1 C and D we show the value of D_k as a function of k, estimated respectively for a point in the uniform data set and for a point near a region of high curvature in the Gaussian data set (highlighted in orange in panels A and B). In the case of the Gaussian distribution, as a consequence of the non-uniform density there exists a value of k where D_k is greater than the threshold value, implying that density at the k-th neighbor of i is significantly different from the density at i.

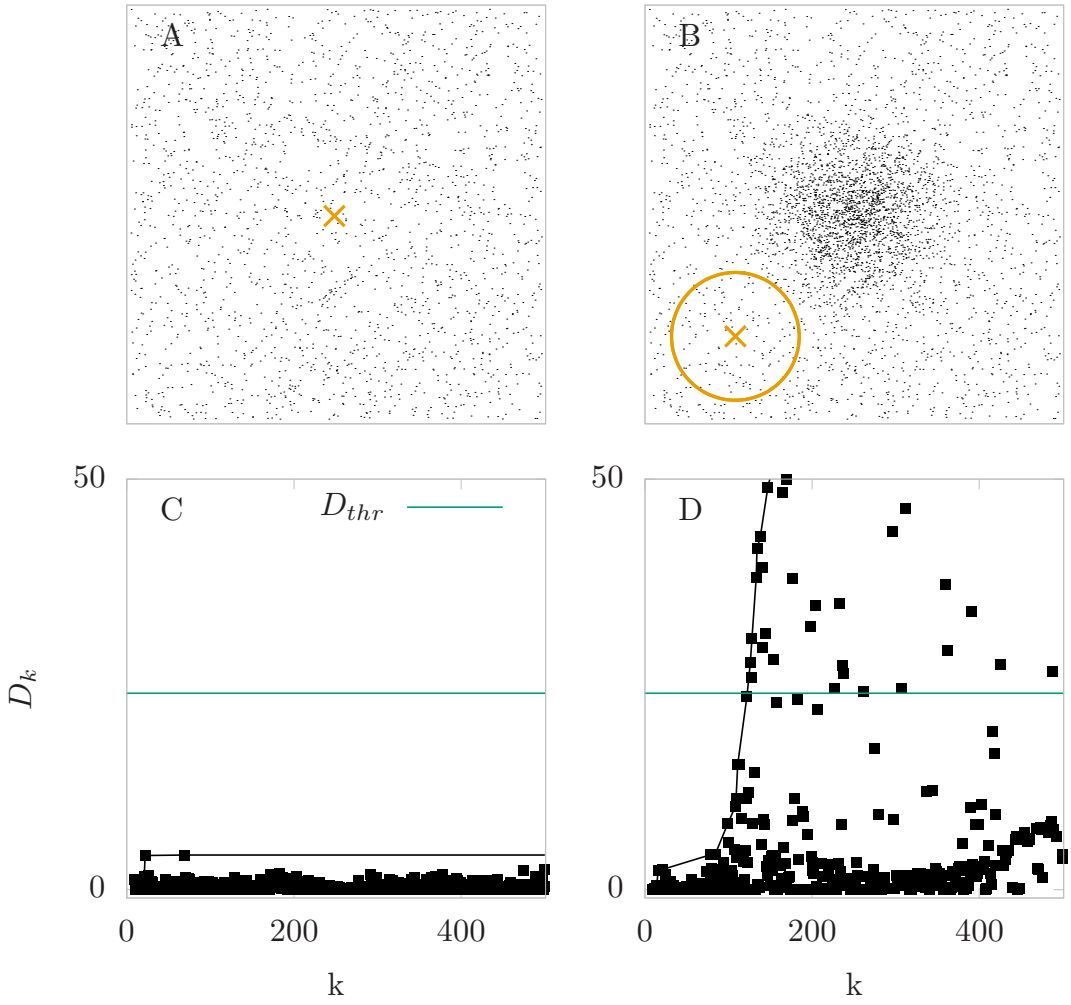

Figure 4.1: *A schematic representation of the density estimation for a two-dimensional example. Panels A and B: a sample of 2000 points extracted from a uniform distribution and the same sample with 2000 additional points extracted from a Gaussian distribution. Panels C and D: D given in Eq. 4.3 as a function of k estimated for the two points highlighted in orange in panels A and B. The green line corresponds to the threshold above which the probability that the distributions around the neighborhoods of points i and k can be described with a single parameter is smaller than* 10^{-7}.

Computing the density

Once the optimal k for point i has been found, it is possible to compute the density distribution at point i as $\rho_i = k/V_{i,k}$. However, we verified that this estimator is affected by small systematic errors, which become important if the data points are

embedded in a high dimensional manifold. Indeed, in the procedure described in the previous section, at the exit value $\hat{k}$ the log-Likelihoods of the two models M1 and M2 are distinguishable with a very high statistical confidence (p-value=10^{-7}). It is therefore likely that the density close to the $\hat{k}$-th neighbor is already substantially different from the density close to data point i. In order to correct this trend, we use a likelihood model depending explicitly on the density, and with an extra variational parameter, hereafter denoted by a, aimed at describing the linear trend in the density as one moves further and further from the central data point. This leads to the following log-Likelihood:

$$\mathcal{L}\left(\rho, a \left| \{v_{i,l}\}_{l\leq\hat{k}_i}\right.\right) = \sum_{l=1}^{\hat{k}_i} \log\left(\rho \cdot e^{a\cdot l} \cdot e^{-\rho\cdot e^{a\cdot l} v_{i,l}}\right) = \\ = \hat{k}_i \log\rho + a\frac{\hat{k}_i\left(\hat{k}_i+1\right)}{2} - \sum_{l=1}^{\hat{k}_i} v_{i,l}\cdot\rho\cdot e^{a\cdot l}. \quad (4.4)$$

Upon setting $a = 0$, this likelihood reduces to the one defined in equation 4.2. We maximize $\mathcal{L}$ in Eq. 4.4 with respect to ρ and a by the Newton-Raphson approach. The asymptotic standard deviation of the estimate is obtained by computing the corresponding element of the inverse of the Fisher information matrix (see for instance (111)):

$$\varepsilon_i = \sqrt{\frac{4\cdot\hat{k}_i + 2}{\left(\hat{k}_i - 1\right)\cdot\hat{k}_i}}. \quad (4.5)$$

The Pointwise-Adaptive k-Nearest Neighbor density estimator, in short PAk, is provided by the ρ_i obtained by maximizing Eq. 4.4. The uncertainty of this estimate is given by Eq. 4.5.

4.2.2 Validation of the PAk estimator

The difficulties that hinder the estimate of the density in real systems are many: the existence of minima with very different density values, the anisotropy of the shape of these minima, a possibly large dimension of the manifold in which the data lay and a large curvature of this manifold. To benchmark the performance of PAk in such cases, we considered the non-isotropic density surface shown in the left panel of Fig. 4.2.

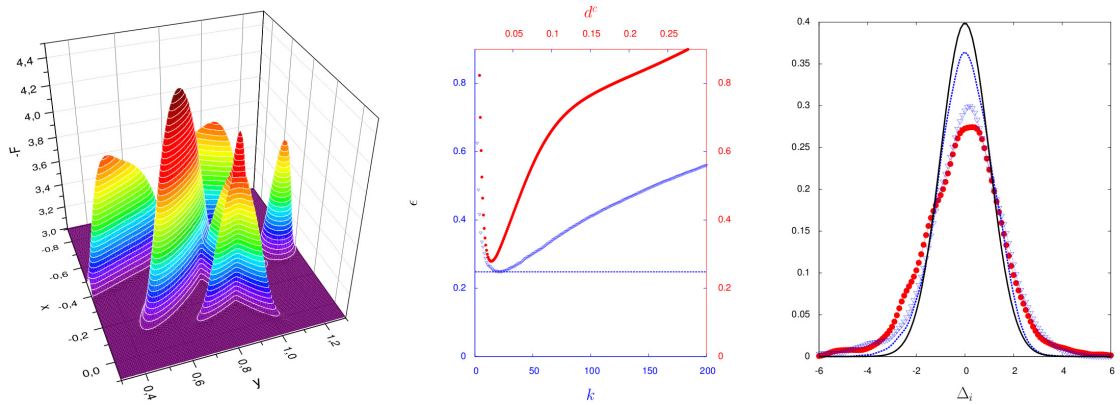

Figure 4.2: *The performance of the density estimator on an anisotropic data set. (Left): The probability distribution from which the data points are drawn, in Logarithmic scale. (Center): The average L1 error ϵ for different values of d^c (top x-axis), ranging from* 0.001 *to* 0.3*, and different values of k (bottom x-axis), ranging from 1 to 200. We compare three density estimators: Fixed Gaussian kernel (red dots), standard k-NN (blue triangles) and PAk (blue dashed line). (Right): The distribution of the pull for a realization of the dataset*

In this landscape the density varies with significantly different rates along different directions. We compare the results with those obtained using other well-established density estimators: (i) the standard k-NN method; and (ii) the Gaussian kernel method, in which the density is estimated as a sum of Gaussians of fixed width d^c centered on each data point of the sample. To compare the accuracy of the three methods we measure the deviation of the density estimate from the corresponding true value by the average L1-error in logarithmic scale $\epsilon = \frac{1}{N}\sum_{i=1}^{N} |-\log \rho_i^{true} + \log \rho_i|$, where ρ_i^{true} is the density value at point i and ρ_i is the estimated density. In addition to the density, the approach introduced in this work provides an estimate of the uncertainty ε_i of the density ρ_i (Eq. 4.5). In order to assess the performance of this estimate, we computed the distribution of the pull (112), again in logarithmic scale, defined for point i as

$$\Delta_i = \frac{(-\log \rho_i^{\text{true}} + \log \rho_i)}{\varepsilon_i}, \tag{4.6}$$

where ρ_i^{true} is the ground truth density value. A shift from zero of the average Δ implies that the density estimate is biased. Moreover, if the estimated error predicts correctly the difference between the true and the estimated density, the standard deviation of the Δ should be close to one, and the probability distribution of Δ should be Gaussian.

The performance of the three methods is shown in the center panel. PAk achieves an accuracy similar to the non-adaptive estimators with their optimal parameters and the pull distribution obtained strongly resembles a Gaussian with zero mean and variance equal to one (Right panel). On the same data set, the Gaussian kernel estimator and k-NN estimator with optimal parameters ($d^c_{opt} = 0.022$ and $k_{opt} = 20$) underestimate the error.

The approach we described in this section has the features we need to analyze complex landscapes characterized by relatively high dimensions and significant density inhomogeneities, such as Pfam protein clans; in fact, in Chapter 3 we learned that protein families have IDs ranging roughly from 7 to 12, and that they present inhomogeneities due to correlations and undersampling in certain regions. Moreover, Pfam clans consist of many families, and families can be further partitioned according to their domain architectures, in a hierarchical structure. A feature of the PAk estimator that proves essential to describe such hierarchies is the possibility to estimate the error on the measure. In the next section we are going to describe a clustering algorithm that is able to exploit the information about the error to arrange clusters in a hierarchical way and thus uncover complex density topographies.

4.3 Density peaks and saddles: uncovering complex clusters topographies

Clustering methods are generally divided into five groups: hierarchical clustering, partitioning clustering, density-based clustering, grid-based clustering and model-based clustering.
In density-based algorithms, clusters are compact sets of data characterized by a density higher than the surrounding regions. This allows detecting non-spherical clusters without specifying their number a priori. The approaches based on this idea include DBSCAN (113), OPTICS (114), Mean-shift clustering (115) and Density Peaks clustering (DP) (9). Each method has its own strengths and weaknesses, mainly concerning the robustness of results with respect to the choice of user-defined parameters. In particular, the DP algorithm requires selecting by hand a region in the so called Decision Graph. The ability to detect fine local structures can be of fundamental importance; if we describe a dataset as a realization of an underlying probability density function, in real cases such density can be high dimensional, it can lie on twisted manifolds as well as present a complex non-uniform profile with

multiple level hierarchies.
In the previous section we described a new Point-Adaptive k-NN density estimator (PAk), that is fully automatic and non-parametric, as it is independent from the external choice of the neighbors k; moreover, it provides an estimate of the uncertainty of the density. It appears natural to combine such estimator with the Density-Peaks algorithm, in order to describe fine local structures; in order to recognize how these structures are related to each other, and organize them in a hierarchy, it will prove fundamental to accurately estimate the error.

In this section we introduce an improved version of the Density-Peaks algorithm combined with PAk density estimator. The joint approach is fully unsupervised and parameter-free and, as we pointed out before, it allows to analyze the density and thus the clusters in the reduced space with a dimensionality given by the ID estimate described in Chapter 2.

4.3.1 Automatic detection of density peaks

As in the standard Density Peaks clustering, we assume that density peaks are characterized by two properties: they are surrounded by neighbors with lower local density, and they are at a relatively large distance from any points with a higher local density. However, PAK provides additional information besides the density that we can exploit for our analysis: an estimate of the error and a neighborhood size around each point in which the density can be considered approximately constant.
To find the density peaks we rank the points not only according to their density, which can be affected by non-uniform errors, but also taking into account their estimated errors. Thus, we rank point i according to quantity g_i defined as

$$g_i = \rho_i - \varepsilon_{\rho_i}. \tag{4.7}$$

At this point, we define as cluster centers the local maxima of g_i; in this way, points with relatively large error are disfavored with respect to points with a smaller error, and they are less likely to be selected as cluster centers.
This definition is a generalization of the one exploited in (9), since the local maxima of g_i coincide with the local maxima of ρ_i if the error is uniform. Following reference (9) we then compute $\delta_i = \min_{j:g_j>g_i} r_{ij}$, namely the distance to the nearest point with higher g. In order to find automatically the cluster centers we exploit the estimate of the

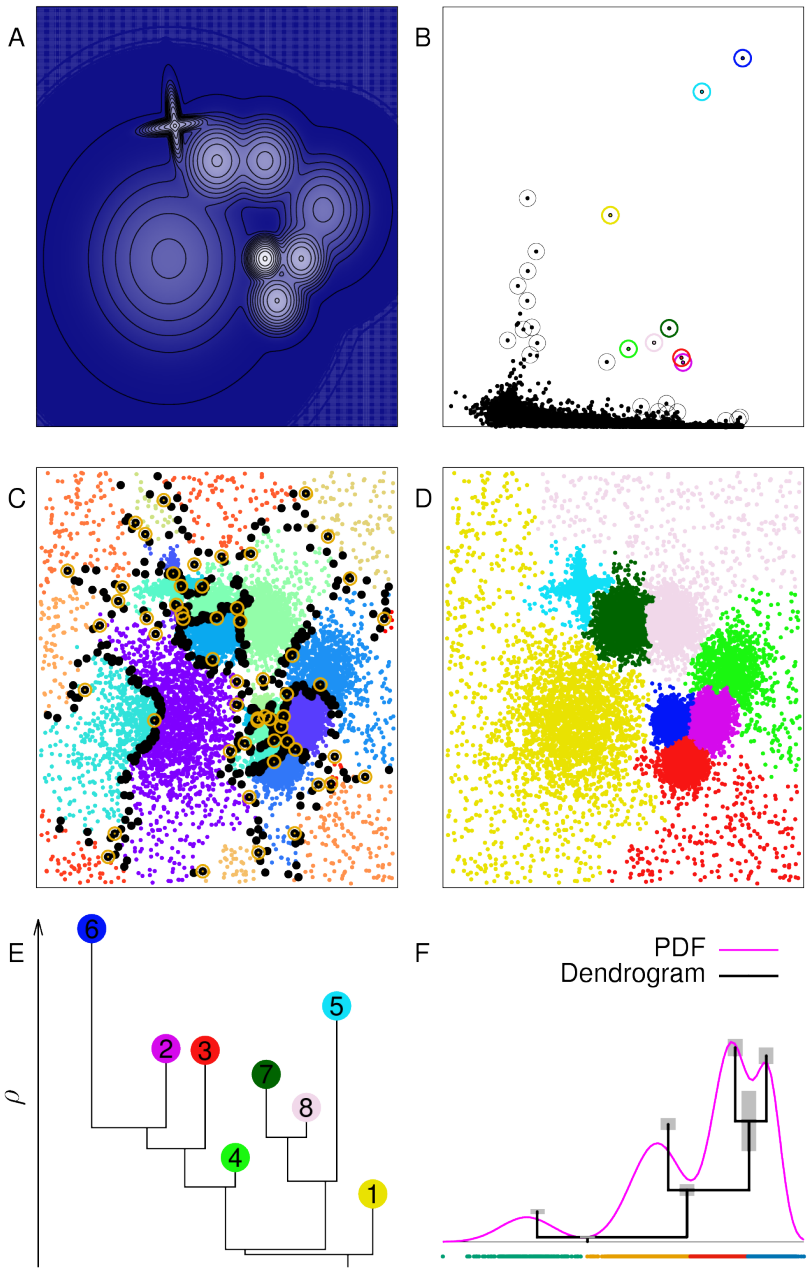

Figure 4.3: *Figure caption*

size of the neighborhood in which the density can be considered constant provided by PAk estimator. Therefore, we consider as putative centers all the points for which

$$\delta_i > r_{\hat{k}_i} \tag{4.8}$$

where $\hat{k}_i$ is the optimal number of nearest neighbors defined in the previous section. Thus, a data point is a center only if all its $\hat{k}$ nearest neighbors, which contribute to determine the value of its density, have a value of g lower than g_i.

In Panel 4.3B we show the Decision Graph for a sample of 20000 points extracted from the probability density distribution shown in Panel A. The points surrounded by a circle are those that are automatically chosen as putative centers according with the above mentioned criterion.

As a next step one assigns all the points that are not centers, in order of decreasing g, to the same cluster as the nearest point with higher g. In Panel 4.3-C we show the result of this preliminary assignation. Different clusters are represented with different colors.
As illustrated in the example in the figure, most of the clusters found by these automatic procedure are genuine density peaks, but some are not. Indeed, the criterion in equation 4.8 can be satisfied even by chance, as all the quantities entering in this equation are affected by statistical errors. In the next section we describe a protocol that allows discriminating meaningful density peaks from statistical fluctuations upon the choice of a confidence level.

4.3.2 Assessing peaks significance

The first step to establish the significance of a cluster is to identify the data points that are at the border between two clusters. A point i, belonging to cluster c, is assumed to be at the border between cluster c and c' if its closest point j belonging to c' is within a distance $r_{\hat{k}_i}$ and if i is the closest point to j among those belonging to c. At this point we can find the saddle point between each pair of clusters c and c', defined as the point with the highest value of g among those at the border between c and c'; the value of the density at this point and its error are denoted by $\rho_{cc'}$ and $\varepsilon_{\rho_{cc'}}$. The border points between the clusters are shown in black in Fig. 4.3C, while the estimated saddle points between the clusters are circled in yellow.

Based on the value of the saddle densities $\rho_{cc'}$ and their errors we introduce a criterion for distinguishing genuine density peaks from statistical fluctuations of the density due to finite sampling. Qualitatively, if all the points in a cluster have density values compatible, within their errors, with the border density, then the cluster can be considered as the result of a statistical fluctuation and merged with another cluster.

More formally, cluster c is merged with cluster c' if

$$(\rho_c - \rho_{cc'}) < Z \cdot \left(\varepsilon_{\rho_c} + \varepsilon_{\rho_{cc'}}\right), \tag{4.9}$$

where ρ_c is the density of the center of cluster c. The constant Z entering in this equation fixes the level of statistical confidence at which one decides to consider a cluster meaningful. It is the (only) free parameter of our approach. In the following we will describe a criterion for choosing reasonable values of Z is real applications. Condition 4.9 is checked for all the clusters c and c', in order of decreasing $\rho_{cc'}$. This procedure allows to prune the set of clusters from those corresponding to density maxima that are not statistically robust, thus recovering the topography of the underlying density function. In Fig. 4.3D it can be seen that the cluster assignation after merging (with $Z = 1.5$) resembles almost perfectly the peaks shown in panel A. Indeed, these results correspond with those obtained in the standard Density Peaks method choosing as centers the colored circles in panel B.

4.3.3 Hierarchy of peaks

The information about the saddles is crucial in two ways: it allows assessing the significance of the density peaks, and the height of the saddles can be employed to determine a hierarchical structure; for instance if two maxima are linked by a saddle at a high density value, they can be considered a single maximum in a low-resolution analysis.

Following a strategy similar to reference (116), we propose a visualization of the topography of the density distribution. We build a dendrogram using the Single Linkage algorithm (117), that can be applied using as a distance between cluster c and c' a strictly decreasing function of the border density $\rho_{cc'}$. We design the heights of the branches as proportional to the density of the peak that determines the cluster; in this way the visualization summarizes the density topography underlying the data set as well as the hierarchical relation between the maxima. Figure 4.3E provides the resulting dendrogram of the above mentioned dataset. Note that the dendrogram truly reflects the relationship between the different peaks. Figure 4.3F provides a further insight into how the dendrogram should be interpreted: At the bottom of the graph, we can see a single realization of the one dimensional probability density function drawn in purple solid line; the final cluster assignation is highlighted by the different colors of the points while the dendrogram reproduces the actual shape of the density function.

4.4 Clustering Pfam clan PUA: a hierarchical partition into families and architectures

With the toolbox discussed above, we are able to analyze the complex case of a Pfam clan. As we discussed in Chapter 3, Pfam clans group several protein families that are believed to share a common evolutionary history; up to four independent factors are employed to help assess whether families are related: related structure, related function, significant sequence matching to HMMs from different families and profile-profile comparisons (118). Profile-profile comparison consists in aligning and scoring two profile HMMs; the result of the comparison is expressed through an *E-value* giving an estimate of how significant the matches are (119). An example of graphical representation of the relationships between families in PUA clan is given in Figure 3.7: only the significant relationships (E-value below 0.001) are represented by a solid line with thickness proportional to the E-value, appearing as a label. After all edges have been added in this way, any detached family is connected by adding a dashed line between it and the node in the clan with the highest scoring profile, provided that its E-value is between 0.001 and 10.
After identifying a set of related families, Pfam curators try to merge them in a single comprehensive model that detects all the proteins detected by the individual models of the families; if this cannot be achieved a clan is built, ensuring the maximum coverage with the minimum number of models.

We analyze the case of Pfam clan PUA in release 28.0, consisting of the RNA binding PUA domain and ASCH domain. PUA is a large clan enclosing tens of thousands sequences, divided into ten families named PUA, TruB-C, TruB-C_2, UPF0113, LON, ASCH, DUF3850, EVE, PUA_2, YTH. Every family comprises sequences with several different domain architectures, where the domain characterizing the family can be found alone or combined with other domains: in this way it is possible to define two levels of organization within the clan: the one into families, and the one into architectures.
We analyze PUA clan by the clustering algorithm described in the previous section; in this case, the accuracy of the clusters description as well as the capability of detecting multiple levels of organization can be tested at once, by comparing the results with the ground truth classification.

The accuracy of clustering with respect to the architectures can be measured by

introducing the concept of cluster purity. Intuitively, the purity of a cluster C with respect to an architecture A is defined as the number of sequences in C belonging to A divided by the total population of C. In Fig. 4.4 we represent the correspondence between clusters ordered according to the dendrogram (y-axis) and architectures (x-axis). Only architectures and clusters with a population greater than 40 are displayed. In this representation the purity of clusters with respect to architectures is associated to a grey palette: the darker the cell, the higher the purity. Fig. 4.4 shows that clusters are substantially pure with respect to architectures (most of the clusters are over 90% pure). The quality of the results was also assessed by computing the Jaccard index (120) of the clustering partition with respect to the Pfam classification. To compute the indices a family (or architecture) label is assigned to each cluster according to a majority rule. We find a Jaccard index of 0.98 for the classification in families, and of 0.79 for the classification in architectures. This reveals a high degree of consistency between the clustering partition and the Pfam classification.
Note that the dendrogram provides further information about the complex topography of the data set, showing, for instance, that clusters belonging to the same architecture are closely related to each other. It essentially reflects the similarity between families in the clan as well as their division into architectures. The only important exception is that families TruB-C_2 and TruB-C are merged in a single cluster. These two families are characterized by a low similarity in the sequences within the same family, thus the error in the estimated densities is so large that th e faint saddle point that separates the two families is classified by our algorithm as a statistical fluctuation. Note that in Pfam classification these two families are related by a profile-profile score higher than all the others.

The analysis we carried out shows that not only the clustering algorithm we developed is able to reconstruct families with a precision near to 100%, but it can detect a further level of complexity, namely that of domain architectures. This result, as we anticipated, is striking, as it proves that the method is sensitive to the differences in the sequences of a domain when it appears alone, or coupled with other domains. In other words, by analyzing a single domain, the algorithm is able to tell which is the architecture of the sequence from which it was extracted.
This procedure can be employed to guide the classification of protein sequences as well as to validate the existing one. For instance, from the dendrogram we can notice that PUA family is close to families TruB-C and TruB-C_2; in release 28. 0 the latters are close to each other, but no significant similarity was underlined between them and PUA, while in the current release (31.0) the profile-profile score between

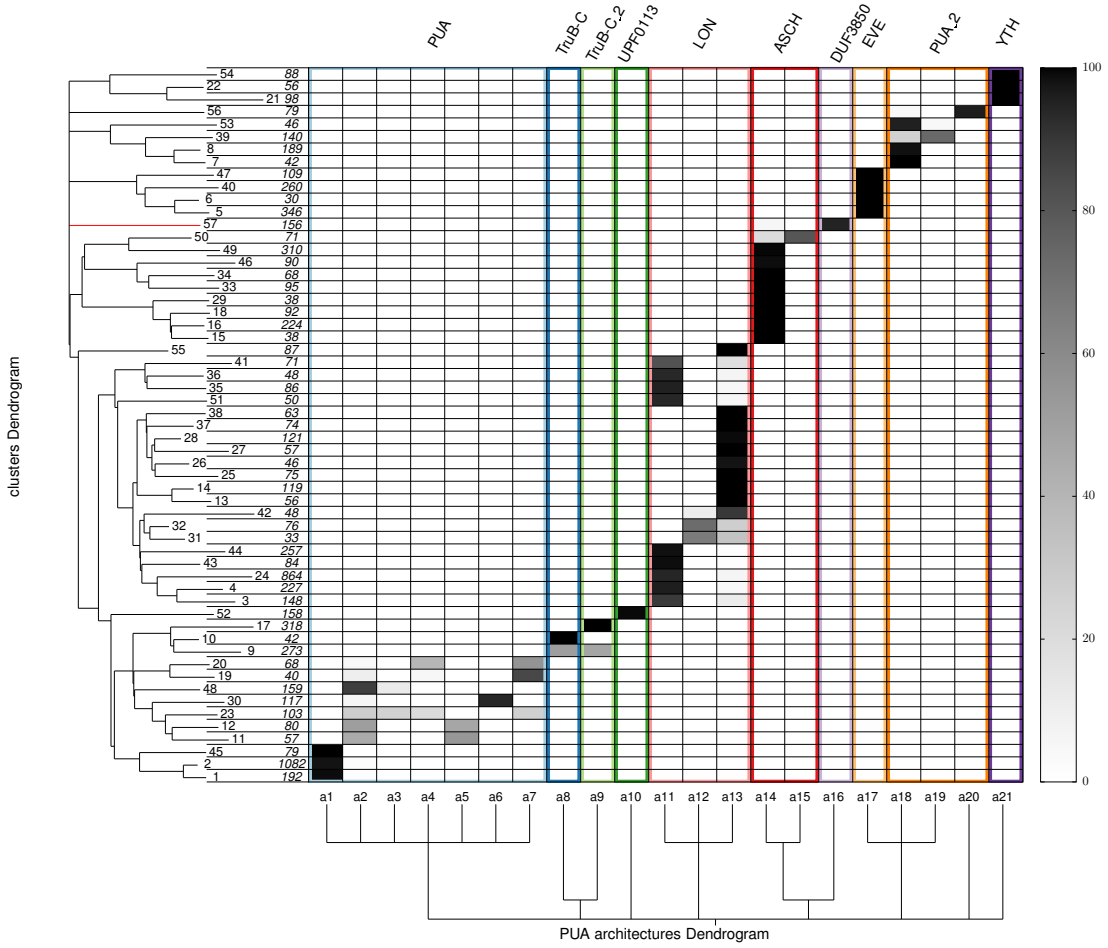

Figure 4.4: *Cluster analysis of the clan PUA from the Pfam database. We represent the Purity Matrix for clusters and architectures with a population greater than 40. Color boxes correspond to families. The purity of the clusters with respect to architectures is associated to a grey palette: the darker the cell, the higher the purity. On the left of the Purity Matrix we show the dendrogram of the clusters. In correspondence of each leaf we indicate the cluster label and the population of the cluster. The Dendrogram at the bottom is a schematic visualization of the hierarchical relation existing between architectures according to Pfam: architectures connected at a higher level (e.g. a1, a2, a3) belong to the same family, while those connected at a lower level (e.g. a12, a13) belong to families that are closely related according to the Pfam definition.*

PUA and TruB-C and between PUA and TruB-C_2 is annotated, meaning that the three families are now considered similar to each other.

In this analysis it is crucial to work in the reduced space whose dimension is the ID

of the dataset, as well as to develop a notion of distance that reflects the relevant features of similarity between sequences. Other fundamental ingredients are the capability to detect fine structures through an accurate estimation of the density and that of reconstructing statistically meaningful hierarchies by means of an error on the density. Finally, all the steps are designed in order to be applicable in case of large datasets.

Up to now we analyzed only clans partitions into families. The study of datasets with multiple clans will be object of further research. In this case a third level of complexity is added, as the first partition is the one into superfamilies.

Chapter 5

ID estimation of datasets with multiple components: a Bayesian point of view

5.1 Introduction

The TWO-NN estimator is built on the assumption that points are sampled from a manifold with a well-defined dimensionality, uniform along the dataset. Under this assumption, together with the assumption of local uniformity in the density, a *minimal neighborhood information* - the ratios of the distances of the second and first neighbor of each point, that we called ν - is sufficient to infer the intrinsic dimension d. The distribution of ν is independent of the density, while it depends parametrically on d: such dependence can be cast in terms of a simple linear relation from which d can be immediately inferred by linear regression. As we have shown in Chapter 3, the scope of applicability of TWO-NN is wide: successful dimension estimation can be achieved not ony in toy examples, but also in many real datasets. However, one frequently encounters situations where the assumption of uniformity in the intrinsic dimension is violated, leading to failure of the TWO-NN model. Such violations manifest by a rupture of the linear relation predicted by TWO-NN; in this case, in the plot of the $\log(\nu)$ against $\log(1 - F(\nu))$, namely in the set we called S, points do not lay on a line, but on a curve (usually well-approximated by a piecewise linear function). In the most extreme cases, the structure of the data is so complex that a global notion of dimension appears to be ill-defined, and a pointwise, continuous denfinition of dimension is more appropriate. However, in many cases the data can be reasonably conceived as being sampled from a *finite number of manifolds, each*

with its own well-defined dimension. This is, for instance, a subject addressed by manifold learning (121). If the manifolds are sufficiently separated - such that the neighborhood of each point mostly contains points sampled from the same manifold - then a minimum neighborhood approach to dimension estimation is still meaningful. Thanks to its simplicity, the TWO-NN model proves ductile and allows an extension that enables to cope with these cases. The key observation is that even in presence of more than one manifold, the ν values still follow a predictable distribution that is independent of the density, and related to the local dimensions. Inferring the dimensions of the manifolds, their size, and which points are sampled from which manifold can be recast as an inference problem, that can be addressed through well-established techniques.

Two immediate options are maximum likelihood estimation and Bayesian estimation. We choose to work within a Bayesian estimation framework because it allows incorporating any prior expectation one may have about the quantities of interest, for instance the number, size, and dimension of the manifolds, the size of their intersection, and so on. Such prior information can be critical for a reliable estimation in the most challenging datasets. If instead no prior expectation is available, Bayesian estimation allows using non-informative priors that make the estimation procedure essentially equivalent to a maximum-likelihood one.

In this Chapter we explore an extension of the TWO-NN approach to multi-dimensional datasets. By working in a Bayesian framework, and adding the further hypothesis that close points are likely to have the same local dimension, we are able to separate components with different IDs within the same dataset, inferring their dimensions and the assignation of each point to its manifold.

5.2 Tools

Basic notions of Bayesian statistics

Bayesian statistics is an approach to statistical problems that interprets probability as a measure of confidence that one may possess about the occurence of a particular event; it provides mathematical tools to rationally update subjective beliefs after new evidence has been collected. The basis of Bayesian statistics is the well known

Bayes' theorem:

$$P(\theta|E) = \frac{P(E|\theta)P(\theta)}{P(E)}, \tag{5.1}$$

where θ is a set of unknown parameters to be estimated and E is the evidence. $P(\theta)$ is called the prior, and summarizes our belief about the parameters before considering the evidence E. $P(\theta|E)$ is the posterior, or our belief about θ once the evidence E has been taken into account. Instead, $P(E|\theta)$ is called the likelihood, and it represents the probability of seeing the evidence E as generated by a model with parameters θ; finally $P(E)$ is the probability of the evidence.
The key difference with respect to the frequentist approach is that here the parameter θ is not fixed, but it is subject to statistical investigation; it is possible to enclose our knowledge or predictions about θ in the prior, that is represented by a probability distribution: in this way we can not only express a guess for the prior, but also quantify our uncertainty about the true value. For instance, if we know nothing about the true value, we could use a uniform distribution on a suitable finite domain; if instead we have a guess, we can pick a bell-shaped prior distribution. Then, after observing the data, the prior is updated to the posterior, which is used for inference; in this way, step by step, the probabilities are updated.
There are fundamentally two difficulties to face when dealing with Bayesian analysis. One of them is computational, as Bayesian calculations almost invariably require integration over uncertain parameters; for instance, at the denominator in 5.2 the probability of the evidence $P(E)$ has to be estimated through integration over all the possible values for the parameter θ. Many times there is no analytical solution at disposal, and numerical integration is necessary, for instance by Markov chain Monte Carlo methods (see the next section).
Another difficulty that is intrinsic to the method lies in the specification of the prior probability distributions. To overcame the problems due to integration, it is common practice to make use of conjugate priors, meaning priors that belong to the same family of distributions as the posterior after the updating. In this way, no integration is needed, as the posterior has the same shape as the prior and only the parameters have to be updated. For example, if the likelihood $P(E|\theta)$ is in the form of n Bernoulli trials with parameter $q \in [0, 1]$ it is distributed according to the

Binomial distribution:

$$P(x) = \binom{n}{x} q^x (1-q)^{n-x}. \tag{5.2}$$

As a function of q, this has the form $f(q) \propto q^a(1-q)^b$ for some constants a and b. Therefore, upon normalization it has the same shape of a Beta function. We can choose as a prior exactly a Beta function with parameters α and β:

$$P(q) = Beta(\alpha, \beta) = \frac{q^{\alpha-1}(1-q)^{\beta-1}}{B(\alpha, \beta)}. \tag{5.3}$$

By tuning these two parameters one can center the distribution around any value between 0 and 1. In this way the posterior behaves as the product of two Beta distributions, and is itself proportional to a Beta distribution with updated parameters.

Gibbs sampling

Gibbs sampling is a Markov chain Monte Carlo (MCMC) algorithm which allows sampling observations from a multivariate distribution when direct sampling is difficult, for instance in case of unknown or complicated joint distribution. The Gibbs sampling algorithm generates an instance from the distribution of each variable in turn, conditional on the current values of all the other variables. In this way it is possible to sample the variables one at a time, making use for instance of the Metropolis-Hastings algorithm. It is particularly useful to sample the posteriors in the framework of Bayesian approaches, since in this case the posterior is already in the form of a collection of conditional distributions.
Suppose we want to generate N samples $\{\mathbf{X}^{(i)}\}_{i=1,...,N} \doteq \{(x_1^{(i)}, ..., x_n^{(i)})\}_{i=1,...,N}$ in $\mathbb{R}^n$ from a joint distribution $f(x_1, ..., x_n)$. The procedure is the following:

- Choose an initial value $\mathbf{X}^{(i)}$.
- To obtain $\mathbf{X}^{(i+1)}$ sample x_1^{i+1} from $f(x_1^{i+1}|x_2^i, ..., x_n^i)$,
- then sample x_2^{i+1} from $f(x_2^{i+1}|x_1^{i+1}, x_3^i, ..., x_n^i)$,
- Repeat the procedure N times, one for each x_j.

In the next sections we rephrase the approach described in Chapter 2 in a Bayesian framework and extend it to the multi-dimensional case. First we set up the technique

in the case of a single well defined dimension (section 5.3), and then extend it to the multidimensional case 5.4. The hypotheses underlining the TWO-NN model in Chapter 2 prove not sufficient to properly separate manifolds with different dimensions, thus in section 5.5 we introduce a new, but natural hypothesis concerning the homogeneity of the assignment of neighboring points. This last assumption allows for a clean separation of the manifolds.

5.3 One-dimensional framework

Let points $\{x_i\}_{i=1,2,...,N}$ be sampled from a Poisson process that is homogeneous in the scale of the second neighbor of each point and defined on a manifold with unknown intrinsic dimension d. As we demonstrated in Chapter 2, if r_{i1} and r_{i2} are the distances of the first and second neighbor of i respectively, then set $\{\mu_i \doteq \frac{r_{i1}}{r_{i2}}\}_{i=1,2,...,N}$ follows a Pareto distribution on $[1, +\infty)$:

$$P(\mu_i|d) = d\mu_i^{-(d+1)}. \tag{5.4}$$

If we suppose that the μ_i are independent (this can be obtained by retaining only a sub sample of the points after computing the $\{\mu_i\}_{i=1,2,...,N}$) we can write the likelihood of vector $\boldsymbol{\mu} \doteq (\mu_1, \mu_2, ..., \mu_N)$ as

$$P(\boldsymbol{\mu}|d) = d^N \prod_{i=1}^{N} \mu_i^{-(d+1)} = d^N e^{-(d+1)V}, \tag{5.5}$$

where $V \doteq \sum_{i=1}^{N} log(\mu_i)$. To choose a prior on the dimension d, we note that the conjugate distribution of a Pareto distribution is a Gamma distribution. If we assume a generic Gamma prior on d,

$$d \sim Gamma(a, b), \qquad P_{prior}(d) = \frac{d^{a-1}b^a}{\Gamma(a)} e^{-bd}, \tag{5.6}$$

then the posterior distribution of d

$$P_{post}(d) = P(d|\boldsymbol{\mu}) \propto P(\boldsymbol{\mu}|d) P_{prior}(d) \propto d^{a-1+N} e^{-d(b+V)} \tag{5.7}$$

is again a Gamma distribution:

$$P_{post}(d) = \Gamma(a + N, b + V). \tag{5.8}$$

Thus, the parameter d can be estimated as the posterior mean $\frac{a+N}{b+V}$. This procedure

is an alternative approach to the one described in Chapter 2: the local neighborhood model is the same but the information obtained on all the points in the dataset is exploited in a Bayesian framework. If we assume to deal with a dataset where the ID is homogeneous the fitting technique is preferable; in fact, a curvature in the supposed straight line l employed in Chapter 2 is an important alarm bell indicating the presence regions with different IDs. Once it has been detected though, the fitting procedure is not able to separate components with different IDs nor to compute their intrinsic dimensions separately; a Bayesian approach instead allows for a clean description of a multi-dimensional model with an arbitrarily high number of components. It is important to underline that for this characterization a light model as the two-neighbors one is fundamental to describe the posterior in a tractable way. In the following we present a multi-dimensional extension of the one-dimensional Bayesian approach described in this section.

5.4 Multi-dimensional framework

Let the points $\{x_i\}_{i=1,2,...,N}$ be sampled from a space that is a union of K manifolds with dimensions $d_1, d_2, ..., d_K$ respectively. We can write a mixture model for the data $\{\mu_i\}_{i=1,2,...,N}$:

$$P(\mu_i|d_1, d_2, ..., d_K) \doteq \sum_{k=1}^{K} p_k d_k \mu_i^{-(d_k+1)}, \tag{5.9}$$

where the weights $\{p_k\}_{k=1,2,...,K}$ are the a priori probabilities for a point to belong to manifold $k \in \{1, 2, ..., K\}$. The probabilities $\{p_k\}_{k=1,2,...,K}$ are to be estimated from the data, as the dimensions $\{d_k\}_{k=1,2,...,K}$. If we define $\boldsymbol{p} \doteq (p_1, p_2, ..., p_K)$, $\boldsymbol{d} \doteq (d_1, d_2, ..., d_K)$, we can write the likelihood for $\boldsymbol{\mu}$ as

$$P(\boldsymbol{\mu}|\boldsymbol{d}, \boldsymbol{p}) = \prod_{i=1}^{N} \sum_{k=1}^{K} p_k P(\mu_i|d_k) = \prod_{i=1}^{N} \sum_{k=1}^{K} p_k d_k \mu_i^{-(d_k+1)}. \tag{5.10}$$

Since this product is hard to compute, we follow reference (122) and introduce latent variables $\{Z_i\}_{i=1,2,...,N}$, where $Z_i \in \{1, 2, ..., K\}$ represents the assignation of point i to one of the K manifolds. Note that the latent variables are not simply a technical device: they provide fundamental information about how to separate the points into the proper components. If we define $\boldsymbol{Z} \doteq (z_1, z_2, ..., z_N)$ we can rewrite the likelihood

as

$$\begin{aligned} P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{p}) =& \frac{P(\boldsymbol{\mu},\boldsymbol{d},\boldsymbol{p})}{P(\boldsymbol{d},\boldsymbol{p})} \\ =& \sum_{\boldsymbol{Z}} \frac{P(\boldsymbol{\mu},\boldsymbol{d},\boldsymbol{p},\boldsymbol{Z})}{P(\boldsymbol{d},\boldsymbol{p})} \\ =& \sum_{\boldsymbol{Z}} \frac{P_{prior}(\boldsymbol{p})P_{prior}(\boldsymbol{Z}|\boldsymbol{p})P(\boldsymbol{d}|\boldsymbol{Z},\boldsymbol{p})P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z},\boldsymbol{p})}{P(\boldsymbol{d},\boldsymbol{p})} \end{aligned} \tag{5.11}$$

Now we assume that $P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z},\boldsymbol{p}) = P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z})$ and $P(\boldsymbol{d}|\boldsymbol{Z},\boldsymbol{p}) = P_{prior}(\boldsymbol{d})$; the first equality is motivated by that fact that once conditioning on $\boldsymbol{Z}$ the data $\boldsymbol{\mu}$ are independent of $\boldsymbol{p}$, while the second one derives from the choice of a prior on $\boldsymbol{d}$ that is independent of $\boldsymbol{Z}$ and $\boldsymbol{p}$. At this point we can rewrite 5.11 as

$$\begin{aligned} P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{p}) =& \sum_{\boldsymbol{Z}} \frac{P_{prior}(\boldsymbol{p})P_{prior}(\boldsymbol{Z}|\boldsymbol{p})P_{prior}(\boldsymbol{d})P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z})}{P(\boldsymbol{d},\boldsymbol{p})} \\ =& \sum_{\boldsymbol{Z}} P_{prior}(\boldsymbol{Z}|\boldsymbol{p})P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z}), \end{aligned} \tag{5.12}$$

where the last equality derives from an assumption of independence of $\boldsymbol{d}$ and $\boldsymbol{p}$ (the cardinality of the components is independent of their intrinsic dimension). Now if we define quantities $N_k(\boldsymbol{Z}) \doteq \sum_{i=1}^{N} \mathbb{1}_{Z_i=k}$ and $V_k(\boldsymbol{Z}) \doteq \sum_{i=1}^{N} log(\mu_i)\mathbb{1}_{Z_i=k}$, the following equalities hold:

$$P(\boldsymbol{\mu}|\boldsymbol{d},\boldsymbol{Z}) = \prod_{i=1}^{N} d_{Z_i}{\mu_i}^{-(d_{Z_i}+1)} = \prod_{k=1}^{K} d_k^{N_k(\boldsymbol{Z})} e^{-(d_k+1)V_k(\boldsymbol{Z})}, \tag{5.13}$$

$$P_{prior}(\boldsymbol{Z}|\boldsymbol{p}) = \prod_{i} p_{Z_i} \propto \prod_{k=1}^{K} p_k^{N_k(\boldsymbol{Z})}. \tag{5.14}$$

In this way, upon conditioning on the latent variables $\{Z_i\}_{i=1,2,...,N}$ we can rewrite equation 5.10 as the product of K likelihoods of type 5.5, corresponding each to a single manifold. We can proceed in a similar fashion as in the one-dimensional framework, assuming again independent Gamma priors on $d_1, d_2, ..., d_K$:

$$d_k \sim Gamma(a_k, b_k), \qquad P_{prior}(\boldsymbol{d}) = \prod_{k=1}^{K} \frac{d_k^{a_k-1}{b_k}^{a_k}}{\Gamma(a_k)} e^{-b_k d_k}. \tag{5.15}$$

In order to choose a prior for $\boldsymbol{p}$ we note that $P_{prior}(\boldsymbol{Z}|\boldsymbol{p})$ is proportional to a multino-

mial distribution. It is natural then to choose as a prior for $\boldsymbol{p}$ the conjugate of the multinomial distribution, that is to say the Dirichlet distribution:

$$\boldsymbol{p} \sim Dir(\boldsymbol{c}), \qquad P_{prior}(\boldsymbol{p}) = \prod_{k=1}^{K} \frac{p_k^{c_k-1}}{B(\boldsymbol{c})}, \tag{5.16}$$

where $B(\boldsymbol{c}) \doteq \left[\prod_{k=1}^{K}\Gamma(c_k)\right]\left[\Gamma(\sum_{k=1}^{K} c_k)\right]^{-1}$. With this choice for the priors, the posterior is described by the following formula:

$$P_{post}(\boldsymbol{Z},\boldsymbol{d},\boldsymbol{p}) \equiv P(\boldsymbol{Z},\boldsymbol{d},\boldsymbol{p}|\boldsymbol{\mu}) \propto P(\boldsymbol{\mu}|\boldsymbol{Z},\boldsymbol{d},\boldsymbol{p})P_{prior}(\boldsymbol{d})P_{prior}(\boldsymbol{Z}|\boldsymbol{p})P_{prior}(\boldsymbol{p}). \tag{5.17}$$

This can be rewritten as:

$$P_{post}(\boldsymbol{Z},\boldsymbol{d},\boldsymbol{p}) \propto \prod_{k=1}^{K} d_k^{N_k(\boldsymbol{Z})+a_k-1} e^{-d_k(V_k(\boldsymbol{Z})+b_k)} p_k^{N_k(\boldsymbol{Z})+c_k-1}. \tag{5.18}$$

In order to perform a Gibbs sampling of the distribution we need to isolate one variable at a time and compute its probability conditioned to all the other variables. First of all, if we define $\hat{\boldsymbol{d}}_{\boldsymbol{k}} \doteq \boldsymbol{d} \setminus \{d_k\}$, we have:

$$P_{post}(d_k|\boldsymbol{Z},\hat{\boldsymbol{d}}_{\boldsymbol{k}},\boldsymbol{p}) \propto d_k^{N_k(\boldsymbol{Z})+a_k-1} e^{-d_k(V_k(\boldsymbol{Z})+b_k)}, \tag{5.19}$$

that corresponds to a Gamma function with shifted parameters $a_k \longmapsto N_k(\boldsymbol{Z}) + a_k$, $b_k \longmapsto V_k(\boldsymbol{Z}) + b_k$. Now we have to compute the posterior probabilities of p_k given all the other variables. Let $\hat{\boldsymbol{p}}_{\boldsymbol{k}} \doteq \boldsymbol{p} \setminus \{p_k, p_K\}$ for $k = 1, 2, ..., K-1$ (there are only $K-1$ independent p since $p_K = 1 - \sum_{k=1}^{K-1} p_k$). To compute the conditioned probability $P_{post}(p_k|\hat{\boldsymbol{p}}_{\boldsymbol{k}},\boldsymbol{Z},\boldsymbol{d})$ we first note that

$$P_{post}(\boldsymbol{p}|\boldsymbol{Z},\boldsymbol{d}) \propto \prod_{k=1}^{K} p_k^{N_k(\boldsymbol{Z})+c_k-1}; \tag{5.20}$$

so that $P_{post}(\boldsymbol{p}|\boldsymbol{Z},\boldsymbol{d})$ is a Dirichlet distribution:

$$\begin{aligned} P_{post}(\boldsymbol{p}|\boldsymbol{Z},\boldsymbol{d}) =& Dir(N_1(\boldsymbol{Z}) + c_1 - 1, ..., N_K(\boldsymbol{Z}) + c_K - 1) \\ =& \frac{\prod_{k=1}^{K} p_k^{N_k(\boldsymbol{Z})+c_k-1}}{B(N_1(\boldsymbol{Z}) + c_1 - 1, ..., N_K(\boldsymbol{Z}) + c_K - 1)}, \end{aligned} \tag{5.21}$$

where $B(\alpha_1, ..., \alpha_K) = \frac{\prod_{k=1}^{K}\Gamma(\alpha_i)}{\Gamma(\sum_{k=1}^{K}\alpha_i)}$.

Then we can compute

$$P(p_1|p_2,...,p_K) = \frac{P(p_1,p_2,...,p_K)}{P(p_2,...,p_K)} = \frac{P(p_1,p_2,...,p_{K-1})}{P(p_2,...,p_{K-1})}, \tag{5.22}$$

The denominator can be obtained by integration (we rename the exponents $N_k(\boldsymbol{Z}) + c_k - 1 \doteq c_k - 1$):

$$\begin{aligned} P(p_2,...,p_{K-1}) =& \int dp_1 p_1^{c_1-1} \cdots p_{K-1}^{c_{K-1}-1}(1-\sum_{k=1}^{K-1} p_k)^{c_K-1} \\ =& C \int dp_1 p_1^{c_1-1}(1-p_1-p')^{c_K-1}, \end{aligned} \tag{5.23}$$

where $C \doteq \frac{p_2^{c_2-1}\cdots p_{K-1}^{c_{K-1}-1}}{B(c_1,...,c_k)}$ and $p' \doteq \sum_{k=2}^{K-1} p_k$. Now if we define $A' \doteq 1-p'$ we can rewrite 5.23 as:

$$\begin{aligned} P(p_2,...,p_{K-1}) =& C \int dp_1 \Big(\frac{p_1}{A'}\Big)^{c_1-1}\Big(1-\frac{p_1}{A'}\Big)^{c_K-1} A'^{c_1+c_K-2} \\ =& C \int d\tilde{p} \quad \tilde{p}^{c_1-1}\Big(1-\tilde{p}\Big)^{c_K-1} A'^{c_1+c_K-1} \\ =& C A'^{c_1+c_K-1} B(c_1,c_K). \end{aligned} \tag{5.24}$$

Now we can compute the posterior probability in 5.25:

$$\begin{aligned} P(p_1|p_2,...,p_K) =& \frac{P(p_1,p_2,...,p_{K-1})}{P(p_2,...,p_{K-1})} \\ =& \frac{p_1^{c_1-1}\Big(1-p_1-p'\Big)^{c_K-1}}{\Big(1-p'\Big)^{c_1+c_K-1}} \frac{B(c_2,...,c_1+c_K)}{B(c_1,...,c_K)} \\ =& \Big(\frac{p_1}{1-p'}\Big)^{c_1-1}\Big(1-\frac{p_1}{1-p'}\Big)^{c_K-1} \frac{\Gamma(c_1+c_k)}{\Gamma(c_1)\Gamma(c_k)}. \end{aligned} \tag{5.25}$$

Equation 5.25 shows that $P(p_1|p_2,...,p_K)$ is a Beta function of a rescaled parameter $\frac{p_1}{1-p'}$; in this way we can sample $\frac{p_1}{1-p'}$ from a Beta distribution and multiply the sample by $(1-p')$.
Finally, we define $\boldsymbol{Z'} \doteq \boldsymbol{Z} \setminus \{Z_i\}$ and compute the posterior of $\{Z_i\}_{i=1,2,...,N}$:

$$P_{post}(Z_i|\boldsymbol{Z'},\boldsymbol{d},\boldsymbol{p}) \propto d_{Z_i}\mu_i^{-(d_{Z_i}+1)} p_{Z_i}. \tag{5.26}$$

Note that while in the one-dimensional case $K = 1$ there is no need of applying the Gibbs algorithm, since the estimation depends only on parameters $\boldsymbol{\mu}$, in the multi-dimensional case the distribution of dimensions $\{d_k\}_{k=1,\dots,K}$ depends on the hidden variables $\boldsymbol{Z}$ (see 5.19); in this case the Gibbs procedure allows sampling the distribution of $\boldsymbol{d}$ and obtain the estimates for the dimensions $\{d_k\}_{k=1,\dots,K}$ from the sample averages.
Though appealing, this construction is affected by a fundamental problem: it is not able to estimate correctly the latent variables $\boldsymbol{Z}$, and thus the true dimensions $\boldsymbol{d}$ (see section 5.6). This drawback is due to the fact that Pareto distributions with different parameters d_k overlap to a great extent (see Figure 5.1).

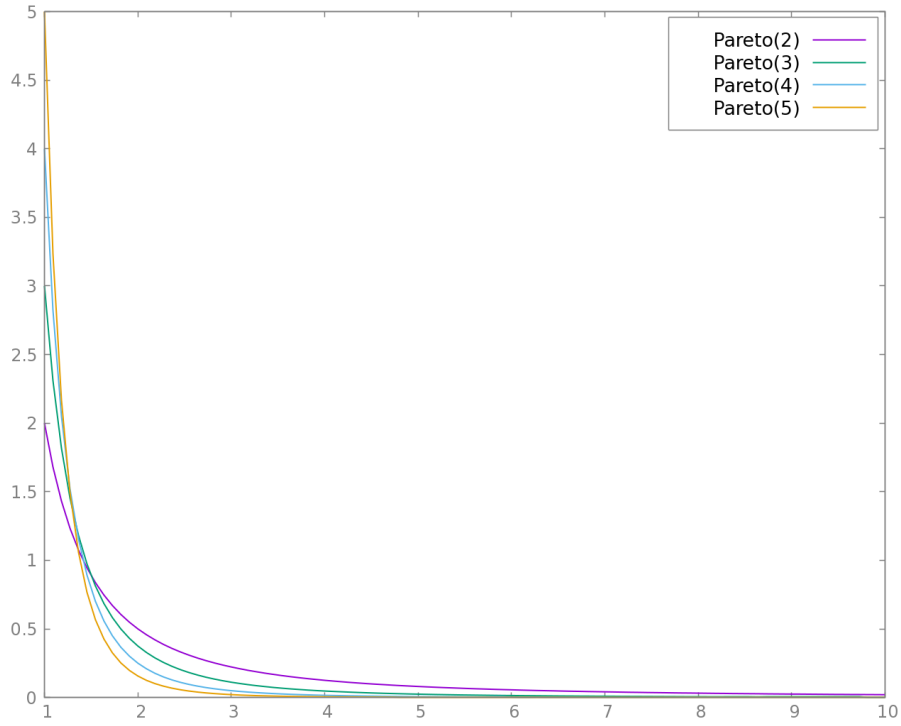

Figure 5.1: *Pareto distributions with parameters* $2, 3, 4, 5$.

This means that without additional information the method is not able to assign a certain value of μ_i to a specific component (even when the components display very different IDs) because it is compatible with all of them.
Up to this point the only data we exploited are the μ_i, a minimal neighborhood information following from the *only* assumption of TWO-NN: that the sampling of the manifolds is sufficiently large that variations in the density distribution are negligible on the scale of the first two neighbors. Yet another reasonable assumption can be introduced without loss of generality; in fact, when dealing with multiple

components we implicitly presume that manifolds are separated in space, at least to some extent. In other terms, *the neighborhood of a point in general should be more likely to contain points sampled from the same manifold than points sampled from a different manifold*. If this hypothesis is violated the dataset is affected by a lack of structure that makes it hard (or impossible) to even define intrinsic dimension.
In the next section we complement the original model with this additional assumption.

5.5 A neighborhood matrix extension of TWO-NN for multi-dimensional datasets

Here we to embed the notion of homogeneity between close neighbors into the original multi-dimensional framework described in the previous section. To this extent, consider the *neighbor matrix* $\mathcal{N}_{ij}^{(q)}$ defined as follows:

$$\mathcal{N}_{ij}^{(q)} = \begin{cases} 1 & \text{if } j \neq i \text{ is among the first } q \text{ neighbors of } i \\ 0 & \text{otherwise} \end{cases} \tag{5.27}$$

Points i and j are said q-neighbors if $\mathcal{N}_{ij}^{(q)} = 1$. Then, the above assumption can be rephrased by saying that *points sampled from the same manifold are more likely to be neighbors than points sampled from different manifolds* (see Figure 5.2). In general, we cannot impose such a condition in the form of a rigid constraint: we cannot strictly require that $\mathcal{N}_{ij}^{(q)} = 1$ if and only if i and j are sampled from the same manifold. In fact in real cases, due for instance to the presence of noise, regions with different IDs can be not completely separated. The existence of points interspersed between such regions, when coupled with a strict constraint of coherence in the neighborhood, may lead to the assignation of all the entries to the same component.

Having said this, we can formalize our intuition as follows. Given a point i, the number of its neighbors is $\sum_j \mathcal{N}_{ij}^{(q)} = q$; the number of possible ways in which we could choose q neighbors among $N-1$ points (i has to be excluded) is $\frac{(N-1)!}{q!(N-1-q)!}$. Define $\boldsymbol{\mathcal{N}}_i^{(q)}$ as the $i-th$ row of the neighborhood matrix, formally $\boldsymbol{\mathcal{N}}_i^{(q)} \equiv \{\mathcal{N}_{ij}^{(q)}, j = 1, \ldots, N\}$. If it were equally likely that the neighbors are points from the same manifold or from different manifolds, then all configurations, given $\mathbf{Z}$ and thus specifying the partition of points into the components, would be equally likely (in this case $\mathbf{Z}$ would

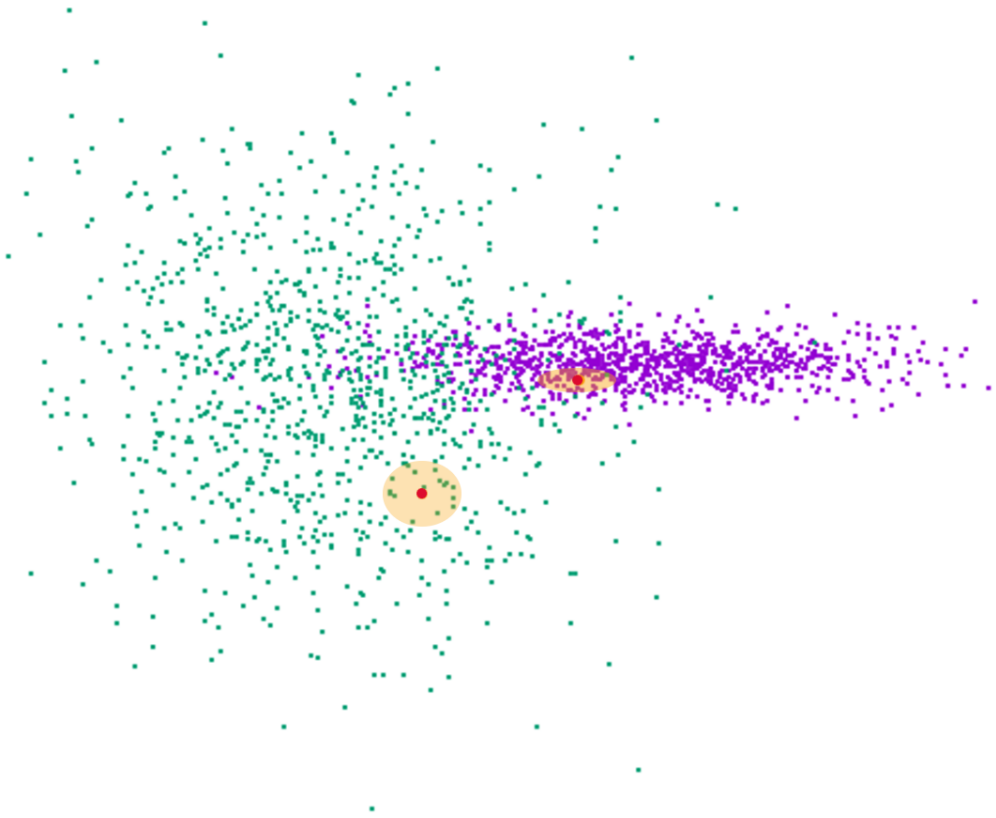

Figure 5.2: *Representation of two intersected Gaussians of dimensions* 2 *and* 3*. Our approach is based on the assumption that points sampled from the same manifold are more likely to be neighbors than points sampled from different manifolds.*

be uninformative) and we would have

$$\mathcal{L}(\boldsymbol{\mathcal{N}}_i^{(q)}|\mathbf{Z}) = \frac{q!(N-1-q)!}{(N-1)!}.$$

We instead assume that points sampled from the same manifold are more likely to be neighbors than points sampled from different manifolds, and as a consequence computing the likelihood requires some more effort. Let ζ be the likelihood that points from the same manifold are neighbors, and $1-\zeta$ the likelihood that points from different manifolds are neighbors. Then we can compute the likelihood as

$$\mathcal{L}(\boldsymbol{\mathcal{N}}_i^{(q)}|\mathbf{Z}\zeta) = \frac{\zeta^{n_i^{in}(\mathbf{Z})}(1-\zeta)^{q-n_i^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_i})}, \tag{5.28}$$

where

$$n_i^{in}(\mathbf{Z}) = \sum_j \mathcal{N}_{ij}^{(q)} \mathbb{I}_{Z_j = Z_i} \tag{5.29}$$

is the number of neighbors of i sampled from the same manifold, and

$$n_i^{out} = \sum_j \mathcal{N}_{ij}^{(q)} \mathbb{I}_{Z_j \neq Z_i} = q - n_i^{in}(\mathbf{Z}) \tag{5.30}$$

is the number of neighbors of i sampled from a different manifold. Function $\mathcal{Z}$ is the normalization factor, and it is by no means negligible; in fact it depends directly on parameter ζ:

$$\mathcal{Z}(\zeta, N_{Z_i}) = \sum_{\{\boldsymbol{\mathcal{N}}_i^{(q)}\}} \zeta^{n_i^{in}(\mathbf{Z})} (1-\zeta)^{q - n_i^{in}(\mathbf{Z})}. \tag{5.31}$$

Function $\mathcal{Z}$ can be expressed in a compact way as

$$\mathcal{Z}(\zeta, N_{Z_i}) = (1-\zeta)^q \binom{N - N_{Z_i}}{q} {}_2F_1(-q, 1 - N_{Z_i}, N - N_{Z_i} - q, \frac{\zeta}{1-\zeta}), \tag{5.32}$$

where ${}_2F_1(a, b, c, x)$ is the hypergeometric function (see appendix B). By considering all points i, we obtain the global likelihood

$$\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)} | \mathbf{Z}\zeta) = \prod_i \mathcal{L}(\boldsymbol{\mathcal{N}}_i^{(q)} | \mathbf{Z}\zeta) = \prod_k \frac{\zeta^{n_k^{in}} (1-\zeta)^{qN_k - n_k^{in}}}{\mathcal{Z}(\zeta, N_k)^{N_k}} \tag{5.33}$$

where

$$n_k^{in} = \sum_{ij} \mathcal{N}_{ij}^{(q)} \mathbb{I}_{Z_i = k} \mathbb{I}_{Z_j = k} \tag{5.34}$$

is the total number of "internal" neighbors of points from manifold k and

$$n_k^{out} = \sum_{ij} \mathcal{N}_{ij}^{(q)} \mathbb{I}_{Z_i = k} (1 - \mathbb{I}_{Z_i = k}) = qN_k - n_k^{in} \tag{5.35}$$

is the total number of “external” neighbors of points from k. Note that since $\mathcal{Z}$ depends on i only through the hidden variables $\mathbf{Z}$ we are able to split the product into K components.

Now we have a new parameter ζ, that can be embedded in the Bayesian framework upon specifying a suitable prior. Note that such prior should be biased towards

$\zeta > 1/2$ to accomplish our working assumption. If $\zeta = 1/2$ no homogeneity hypothesis is presumed, and the model is the same as in the previous section.
We choose a Beta prior on ζ: $P_0(\zeta) = \frac{\zeta^\alpha (1-\zeta)^\beta}{B(\alpha,\beta)}$. At this point we can compute the posterior probability of the whole model as in 5.19, but taking into account the new parameter ζ:

$$\begin{aligned} P_{post}(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta) \equiv & P(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta|\boldsymbol{\mu},\boldsymbol{\mathcal{N}}^{(q)}) \\ \propto & P(\boldsymbol{\mu},\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z},\mathbf{d},\mathbf{p},\zeta)P(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta) \\ = & P(\boldsymbol{\mu}|\boldsymbol{\mathcal{N}}^{(q)},\mathbf{Z},\mathbf{d},\mathbf{p},\zeta)P(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z},\mathbf{d},\mathbf{p},\zeta)P(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta) \\ = & P(\boldsymbol{\mu}|\boldsymbol{\mathcal{N}}^{(q)},\mathbf{Z},\mathbf{d},\mathbf{p},\zeta)P(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z},\mathbf{d},\mathbf{p},\zeta)P(\mathbf{Z}|\mathbf{d},\mathbf{p},\zeta)P(\mathbf{d},\mathbf{p},\zeta) \\ = & P(\boldsymbol{\mu}|\mathbf{Z},\mathbf{d})P(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z},\zeta)P(\mathbf{Z}|\mathbf{p})P(\mathbf{d})P(\mathbf{p})P(\zeta), \end{aligned} \tag{5.36}$$

where we have exploited the same independence observations as in Section 5.4. As a consequence

$$\begin{aligned} P_{post}(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta) \equiv & P(\mathbf{Z},\mathbf{d},\mathbf{p},\zeta|\boldsymbol{\mu},\boldsymbol{\mathcal{N}}^{(q)}) \\ \propto & \prod_{i=1}^{N} \mathcal{L}(\boldsymbol{\mu_i}|\mathbf{Z}_i,\mathbf{d})\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z},\zeta)P(\mathbf{Z}|\mathbf{p})P_{prior}(\mathbf{d})P_{prior}(\mathbf{p})P_{prior}(\zeta). \end{aligned} \tag{5.37}$$

In this case we cannot split the product into K components as we did in 5.18, since the term $\mathcal{N}_i^{(q)}$ depends explicitly on point i.

The conditional posteriors $P_{post}(d_k|\dots)$ and $P_{post}(p_k|\dots)$ are the same as in the original model, and the corresponding Gibbs steps are unmodified. The conditional posterior on $\mathbf{Z}$ is instead modified, together with the corresponding Gibbs step. In the next section we derive the expression for $P_{post}(Z_i = k|\mathbf{Z}',\mathbf{d},\mathbf{p})$

Finally, one has to add a Gibbs step on ζ, based on the corresponding conditional posterior

$$P_{post}(\zeta|\dots) \propto \prod_k \frac{\zeta^{n_k^{in}}(1-\zeta)^{qN_k - n_k^{in}}}{\mathcal{Z}(\zeta,N_k)^{N_k}} \frac{\zeta^\alpha(1-\zeta)^\beta}{B(\alpha,\beta)} = \tag{5.38}$$

$$= \frac{\zeta^{\alpha+n^{in}}(1-\zeta)^{\beta+n^{in}_{out}}}{B(\alpha,\beta)} \prod_k \mathcal{Z}(\zeta, N_k)^{-N_k}$$

where

$$n^{in} = \sum_k n^{in}_k$$

is the total number of "internal" neighbors and $n^{out} = q - n^{in}$ the total number of external neighbors. Note that the beta prior on ζ is not conjugate to the likelihood $\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z}\zeta)$, so the posterior is not a Beta. The term $\prod_k \mathcal{Z}(\zeta, N_k)^{-N_k}$ counterbalances the strong update of ζ towards increasing values when $n^{in} > n^{out}$ (decreasing values when $n^{in} < n^{out}$).

Computing the conditional probability of $\mathbf{Z}$ is trickier; if as usual we define $\mathbf{Z}'$=$\mathbf{Z} - \{Z_i\}$), the object at study is

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) = \frac{P_{post}(Z_i = k, \mathbf{Z}', \mathbf{d}, \mathbf{p})}{P_{post}(\mathbf{Z}', \mathbf{d}, \mathbf{p})}. \tag{5.39}$$

The details of the computation are presented in Appendix C. The result is that for large $N_k \gg 1$ we have $\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}) + 1) \sim \mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}))$ so that it approximately holds

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) \approx \frac{p_k d_k (\mu_i)^{d_k+1}}{\mathcal{Z}(\zeta, N_k)} \left(\frac{\zeta}{1-\zeta}\right)^{n^{in}_i + m^{in}_i}.$$

Once derived the expression of all the conditional posteriors, it is immediate to implement the Gibbs sampling as outlined in section 5.2 and obtain posterior estimates of the parameters of interest, $\mathbf{d}$, $\mathbf{p}$, and $\mathbf{Z}$.

The effectiveness of the method relies upon a sensible choice of two parameters: the number of neighbors preferentially sampled from the same manifold, denoted by q and the statistical bias towards preferential sampling of neighboring points from the same manifold, denoted by ζ. q is a free parameter, and as such it is best fixed by empirically verifying which values ensure a good performance of the estimation. If the manifolds are not perfectly separated, in which case there are neighborhoods containing points sampled from two different manifolds, it is convenient to choose a small q. It is tempting to just restrict to $q = 2$, since this is the most congruous choice with respect to the TWO-NN approach; nevertheless, the idea behind the "minimal neighborhood" choice in TWO-NN is to limit inhomogeneity and curvature

effects, while in this case we are interested in the range where data are sampled from the same manifold, that in general is wider. The optimal value of q can thus depend on the degree of separation of the manifolds: if the separation is sharp, a reasonable set up may employ values of q greater than 2 ; if the manifolds instead have a large intersection, $q = 2$ is the most effective choice. The degree of separation can be roughly measured a posteriori by the fraction of "external" neighbors $\frac{N_{out}}{q \times N}$ (see section 5.5).
Contrary to q, ζ is in principle not a free parameter, but it can be estimated from the data, and its average posterior value should be related to the actual degree of separation between the manifolds. However, for simplicity in the following we will treat ζ as an additional free parameter. This choice is the limiting case of using a prior strongly peaked around a value $\zeta = \zeta_0$; such prior can be realized, for instance, using a Beta distribution with parameters α, β such that $\frac{\alpha}{\alpha+\beta} = \zeta_0$, in the limit $\alpha, \beta \to \infty$. Thus, ζ will not be sampled, but will be kept fixed. We will thus consider values of ζ in the range $(0.5, 1)$. In the limit $\zeta \to 0.5$, we just reproduce the model in section 5.4. As observed in section 5.4, without imposing a constraint about the homogeneity of the assignation of neighbors the dataset is affected by a lack of structure that hinders the correct separation of manifolds; this occurs because of the shape of Pareto distributions, that by definition overlap on large regions (see Figure 5.1). As ζ is increased, one is imposing an increasingly stricter constraint on the homogeneity of neighborhoods. In the limit $\zeta \to 1$, all neighborhoods are required to be completely homogeneous. Unless the manifolds are perfectly separated (i.e., whenever there is a non vanishing intersection in a neighborhood), the only way to satisfy this constraint is to forcibly merge all points in a single manifold.
In the next section we investigate the performance of the method on artificial datasets.

5.6 Benchmark

To test the method we start from the simple case of two manifolds with different IDs. We opt for multivariate Gaussians as they display a continuous distribution with varying density; the performance of TWO-NN on single Gaussians was investigated in detail in Chapter 2. We analyze sets consisting of two multivariate Gaussians with unitary variance each, drawn in different dimensions; while the lower dimension is fixed and equivalent to 4, the higher one ranges from 5 to 9. In order to challenge the method with overlapping regions, the centers of the Gaussians are at a distance of 0.5, corresponding to half the variance. The most difficult case is the one with

two Gaussians with dimensions 4 and 5 respectively; in this case the problem of the overlap is paired with the one of distinguishing two very similar distributions.

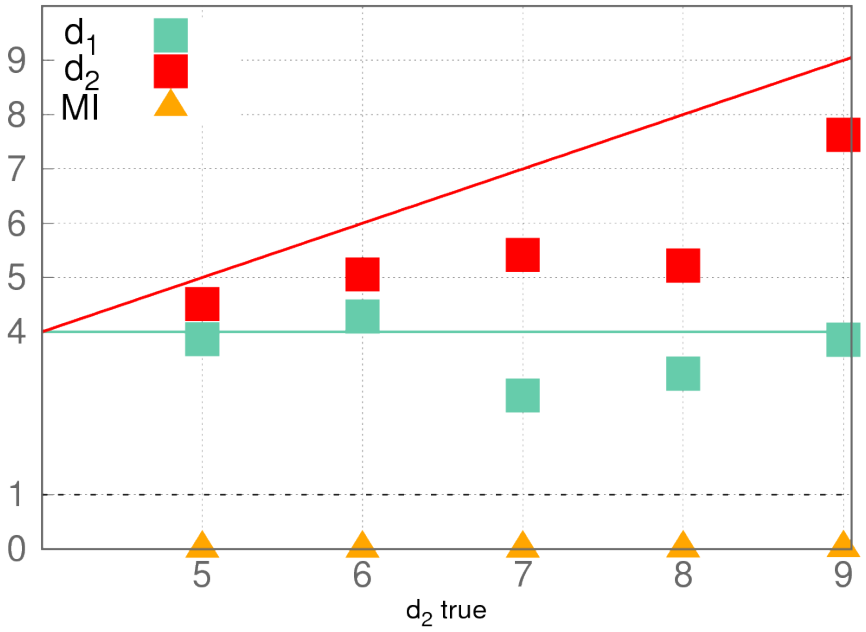

Figure 5.3: *Estimated dimensions of datasets composed by two Gaussians with different dimensions in the case $\zeta = 0.5$. The dimension of one of the Gaussians is fixed to 4 while the other dimension, d_2 true, ranges from 5 to 9. The mutual information MI between the estimated and the correct assignation of points is displayed in yellow triangles.*

In Figure 5.3 we illustrate the results obtained in the case of fixed $\zeta = 0.5$, equivalent to exploit the model in section 5.4; the estimate for the two dimensions is represented together with the mutual information MI between the estimated attribution of points and the true one. As we expect, without a constraint on the assignation of neighbors, the method is not able to separate correctly the points and thus to estimate the dimensions of the two manifolds, even in the case of quite different IDs. Thus, imposing $\zeta = 0.5$ leads to a poor estimation of the dimensions at stake.

In order to look for the optimal working point of the method in the (q, ζ_0) parameter space, we perform tests with several values of $q \in \{2, 3, \ldots, 20\}$, $\zeta_0 \in [0.5, 1)$. We focus on the most challenging case, the one of two Gaussians in dimensions 4 and 5. The crucial figure of merit to assess the performance of the method is the mutual information (MI) between the estimated assignation $\mathbf{Z}$ and the true assignation $\mathbf{Z}$,

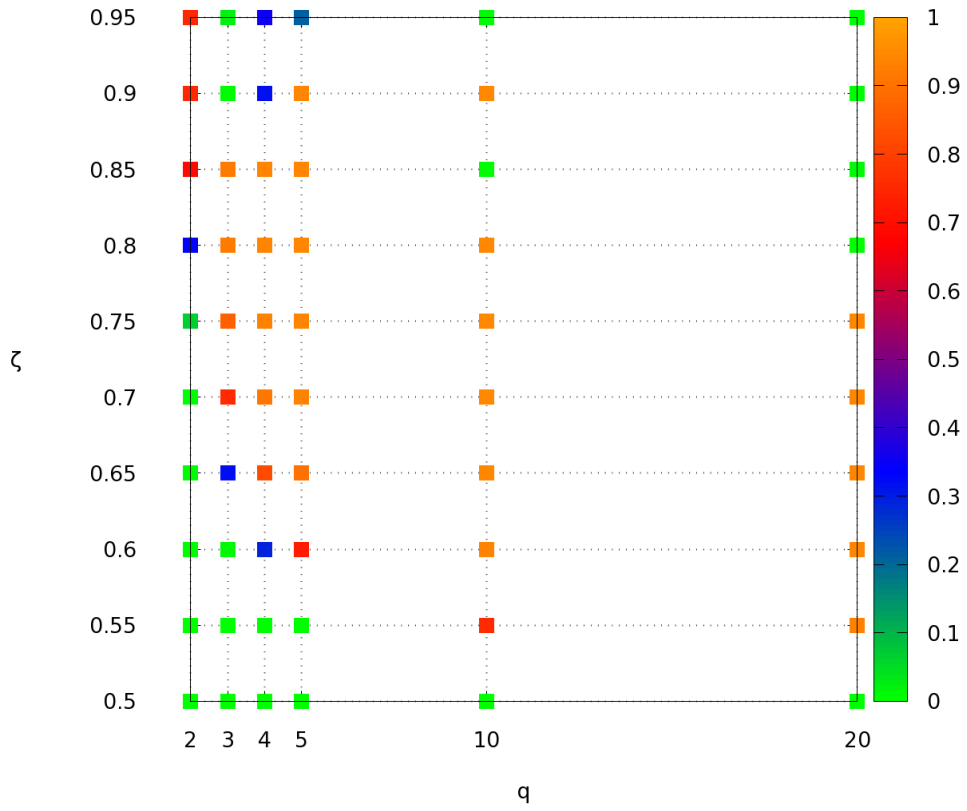

Figure 5.4: *The Mutual Information (MI) between the estimated assignation* $\mathbf{Z}$ *and the true assignation* $\mathbf{Z}$ *as a function of* (q, ζ_0).

which measures the quality of the assignation of points to the manifolds. Indeed, once the manifolds are correctly separated, the problem is reduced to a dimension estimation within the single manifolds (as described in Chapter 2). In Figure 5.4 we show the MI as a function of (q, ζ_0) for a Gibbs sampling with 10^5 iterations.

For most values of q, the MI first increases, then decreases with ζ. This can be expected on the basis of the following considerations. When ζ is close to 0.5, as we discussed above, the method cannot discriminate different manifolds. When ζ is increased, the posterior distribution starts to prefer configurations that approximately satisfy the neighborhood homogeneity constraint. For sufficiently high ζ, the posterior distribution is sharply peaked at the configuration that optimally satisfies this constraint; correspondingly, if the sampling is able to explore the parameter space exhaustively, it will eventually find this region and remain trapped there. Hence, the MI achieves average values close to 1. However, for ζ close to 1 the posterior distribution is very likely to have also pronounced local maxima and depending on the initial configuration the sampling may remain "stuck" in one of them. Hence,

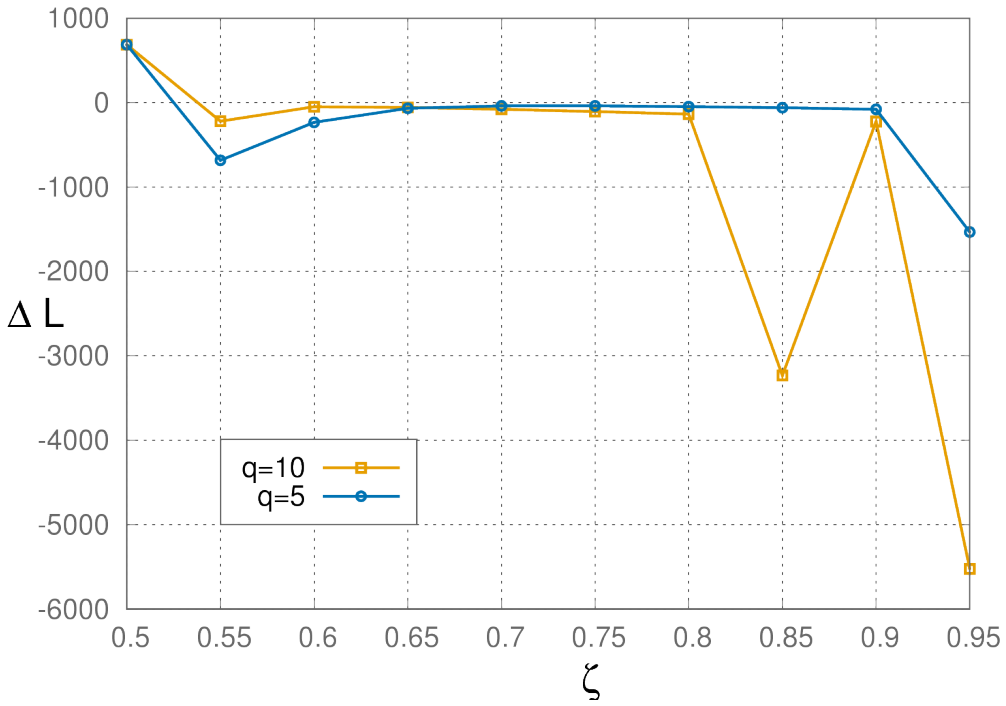

Figure 5.5: *$\|\Delta L\| \doteq \mathcal{L}_{est} - \mathcal{L}_{opt}$ as a function of ζ for $q = 5$ and $q = 10$.*

one can have a drop in the MI. This phenomenon can occur for values of ζ lower than those that will force a merging of the manifolds. In order to validate this interpretation we consider the quantity $\Delta L \doteq \mathcal{L}_{est} - \mathcal{L}_{opt}$, where $\mathcal{L}_{est}$ is the average likelihood of the sampled configurations and $\mathcal{L}_{opt}$ is the likelihood obtained by fixing all parameters to their ground truth values. Small values of $\|\Delta L\|$ imply that the sampling is confined to a region close to the optimal configuration. In Figure 5.5 we plot ΔL for $q = 5$ and $q = 10$ as a function of ζ. In the case $q = 5$ we see that $\|\Delta L\|$ is low in the range $\zeta \in [0.65, 0.9]$, while it drops at $\zeta = 0.95$. Instead, for $q = 10$ $\|\Delta L\|$ is low in the range $\zeta \in [0.65, 0.9]$ except for $\zeta = 0.85$, where it shows a drop corresponding to the drop in the MI in Figure 5.4. We verified that this drop is caused by a sampling failure (the sampling is trapped in a local maximum); indeed if we initialize the system at a configuration close to the optimal one we achieve values of MI and ΔL consistent with those found for $\zeta = 0.8$ and $\zeta = 0.9$. In general these sampling issues can be worsened when q is increased since the local maxima become more and more pronounced. This problem can be alleviated by means of well established sampling techniques such as, for instance, simulated annealing and replica exchange methods. For simplicity in the following we restrict to a region of the parameter space where the results appear the most stable. Based on Figure 5.4

we identify an optimal working point of the method for $q \in \{3, 4, 5\}$ and $\zeta \in [0.7, 0.85]$.

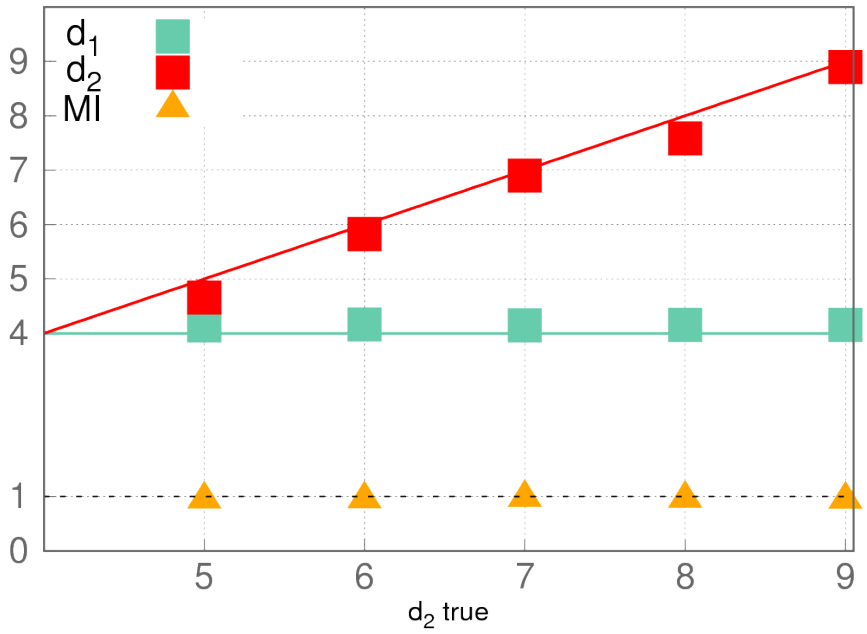

Figure 5.6: *Estimated dimensions of datasets composed by two Gaussians with different dimensions in the case $\zeta = 0.8, q = 3$. The dimension of one of the Gaussians is fixed to 4 while the other dimension, d_2 true, ranges from 5 to 9. The mutual information between the estimated and the correct assignation of points is displayed in yellow triangles.*

In Figure 5.6 we perform the same tests as in Figure 5.3 but with $q = 3$ and $\zeta = 0.8$. Now the MI is almost 1 in all the cases and correspondingly the estimation of d_1 and d_2 is precise. Qualitatively analogous results are achieved for similar datasets obtained from different distributions (for instance uniform distributions on hypercubes).

Finally we test our method on a challenging dataset consisting of 5 Gaussians with unitary variance in dimensions $1, 2, 4, 5, 9$ respectively, shown in Figure 5.7 A.
Some of the Gaussians have similar IDs, as in the case of dimensions 1 and 2, or 4 and 5; moreover they can be extremely close to each other, for instance the means of those in dimensions 4 and 5 are only half a variance far from each other, and they are crossed by the Gaussian in dimension 1. To analyze such dataset we choose the parameters $\zeta = 0.85$ and $q = 3$, as for such choice, at least in the case of two Gaussians in dimensions 4 and 5 respectively, the results appear to be stable (see Figure 5.4). In Figure 5.7 B we illustrate the final assignation of points to the

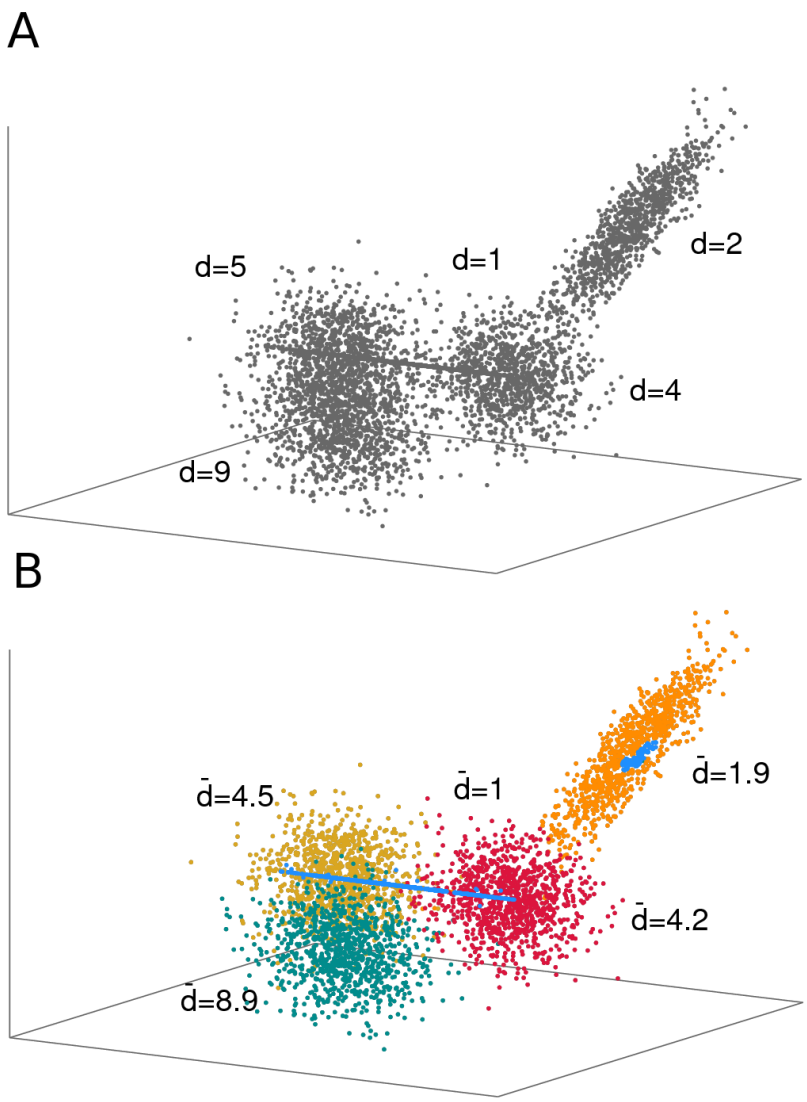

Figure 5.7: *A. A dataset consisting of* 5 *Gaussians with unitary variance in dimensions* $1, 2, 4, 5, 9$. *B. Final assignation of points to the respective manifolds together with the estimated dimensions, denoted by* $\bar{d}$.

respective manifolds together with the estimated dimensions, denoted by $\bar{d}$, upon setting the number of manifolds to 5. The separation of the manifolds is generally good, except for an inaccurate assignation of some of the points of the manifold in dimension 1 to the one with dimension 2 and vice versa; this is possibly due to the sampling being stuck in a local maximum close to the optimal solution, as we discussed above (see Figure 5.5).

In all these test cases we performed the analysis by fixing the number of manifolds K at its ground truth value. A further step would be to let the method estimate K without relying on a priori information. This can be achieved in two ways. The simplest one is to rely on a Bayesian model selection criterion that performs a test on the posteriors with different values of K. The other option is to work in a fully Bayesian setting fixing a prior on K and then estimating it by means of a Reversible jump Markov Chain Monte Carlo simulation (see (123)). This issue will be the object of future investigation.

5.7 Application to real datasets

5.7.1 ID of protein folding trajectories

As a first application of our method, it is natural to address ID estimation for a dynamical system. In fact, the very problem of ID estimation was originally formulated in this field of research, where it first arose from the need to characterize the attractor states of chaotic systems(26). More in general, whether a system is chaotic or not, the physical problem of characterizing its behavior often translates into the geometric problem of describing the regions of phase space it visits. In most cases, motion is effectively restricted to specific regions of phase space. An essential geometric description of such regions necessarily involves an evaluation of their ID. If the system has a single stable attractor, and one focuses on the long-time behavior of the system, then the trajectories will be asymptotically confined to a region of phase space characterized by a single well-defined ID (124). However, in the presence of multiple unstable attractors, or if one does not neglect the short-time behavior of the system, an appropriate description of the visited phase space may require the use of multiple IDs, corresponding to different attractors as well as to portions of phase space transiently visited when the system is outside a basin of attraction. Therefore, it is interesting to verify whether the approach developed in this chapter can be used to capture such situations, providing a more detailed geometric description of the system's dynamics.

Here, we consider the dynamics of the villin headpiece (PDB entry: 2F4K). Villin is an actine-binding protein containing multiple domains capped by a small "headpiece" at the C-terminus consisting of an independently folding three-helix bundle stabilized by hydrophobic interactions. The headpiece domain is commonly studied by molecular

dynamics simulations due to its small size and fast folding kinetics.
Our analysis is based on the longest available molecular dynamics trajectory of the system from ref. (10). The trajectory has a length of 125 μs, corresponding to $N \sim 32000$ configurations. During the simulation time, the protein performs approximately 10 transitions between the folded and the unfolded state. For each configuration of the trajectory, we extract the value of the 32 Ψ backbone dihedral angles. As a result, we have a set of $N = 32000$ vectors embedded in a 32-dimensional space. The distance between each pair of configurations is computed by the Euclidean metric with periodic boundary conditions on the vectors of the dihedral angles. If we simply apply TWO-NN (see Chapter 2) to such dataset, we obtain the S set shown in Figure 5.8.

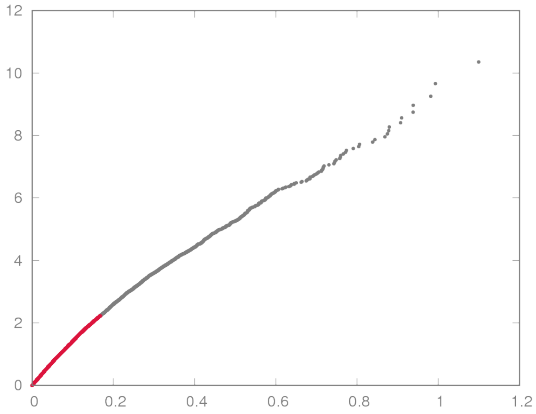

Figure 5.8: *S set (see Chapter 2) for the molecular dynamics trajectory of the villin headpiece.*

Since the S set displays a curvature, we suspect that more than one manifolds with different IDs coexist in the dataset. This feature could be related to the oscillation of the system between the folded and unfolded state. We expect to find different dimensionalities in the folded and unfolded state, since these two states are characterized by different chemical and physical features: the folded state is compact and dry in its core, while the unfolded state is swollen, with most of the residues interacting with a large number of water molecules. Moreover, and possibly more importantly, in the unfolded state the protein is much more mobile: the residue-residue contacts, when formed, survive on average for a much shorter time than in the folded state.
We apply to this system the multi-dimensional analysis introduced in this Chapter with $K = 2$, $q = 2$, and $\zeta = 0.7$. ζ is chosen in the optimal range found on artificial datasets, while q is set to a low value since this empirically allowed for optimal

manifold discrimination, probably due to the low degree of separation between the manifolds ($\frac{N_{out}}{q \times N} \sim 0.2$).

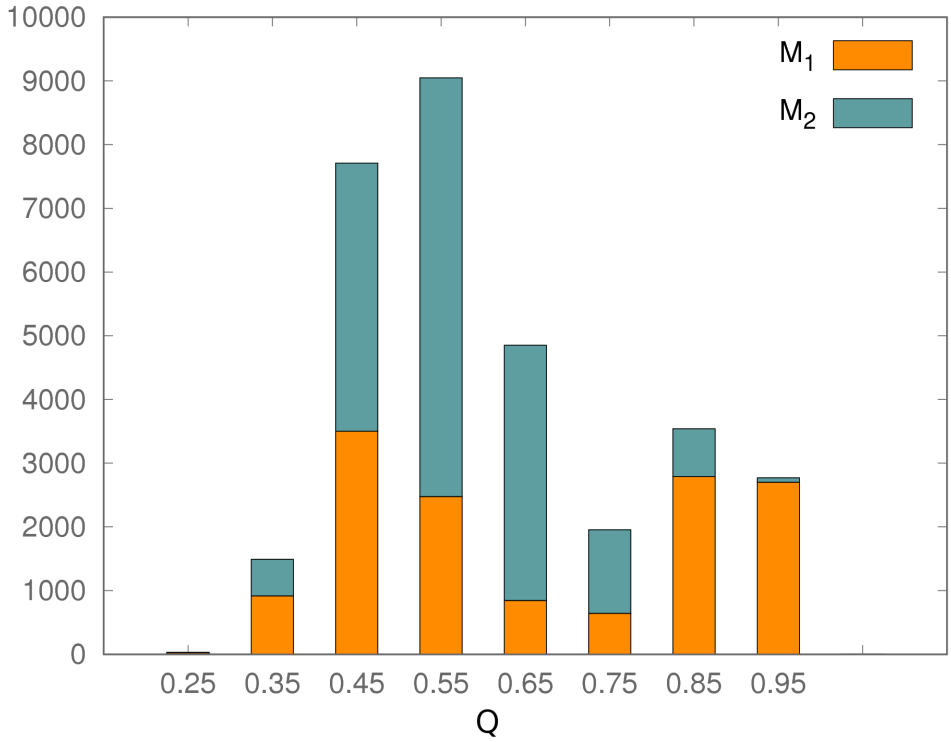

Figure 5.9: *Histogram of the fraction of native contacts Q, showing the number of configurations assigned to manifold M_1 and M_2 respectively.*

We find two manifolds M_1 and M_2 with parameters $d_1 \sim 23$, $d_2 \sim 10$, $p_1 = 0.46$, $p_2 = 0.54$ respectively. Therefore, the dataset can be divided into two manifolds with approximately the same size but significantly different dimensions. To test whether this partition is related to the separation between the folded and the unfolded state we relate the predicted partition into manifold $\mathbf{Z}$ to the fraction of native contacts Q, which can be straightforwardly estimated on each configuration of the system. Q is close to one only if the configuration is folded, while it approaches zero when the protein is unfolded. In Figure 5.9 we show a histogram of Q, where we explicitly highlight the fraction of points assigned to manifold M_1 (higher dimension) and M_2 (lower dimension).
The vast majority of the folded configurations $Q > 0.8$) are assigned to the high-dimensional manifold. The unfolded configurations ($Q < 0.7$) are most of the times assigned to the low-dimensional manifold, and approximately 25% of the times to

the high dimensional manifold.
This implies that a configuration belonging to the low dimensional manifold is almost for sure unfolded. This is a remarkable result, since the intrinsic dimension is a purely topological observable, unaware of any chemical detail. We also remark that the dimension of the ordered (folded) state is *higher* than the dimension of the disordered (unfolded) state. This result seems counterintuitive, since the folding state is quasi-rigid, namely it is characterized by the frustration of all the degrees of freedom that would trigger its unfolding. However, we have seen in Chapter 2 how the intrinsic dimension is sensitive to the length scale. Therefore, we speculate that the large value of the ID that we find in the folded state might be related to the absence of a clear separation between "soft" and "hard" directions: all the directions are approximately equally hard or, more precisely, no clear gap is present between a set of soft directions and the other directions. Instead, in the unfolded state the system is allowed to move much more, but only in a few independent directions. Our analysis suggests that the number of these directions is of the order of 10.

5.7.2 ID of fMRI signals

As our next example, we employ our tool to analyze a series of volumetric images of a human brain obtained through functional magnetic resonance imaging (fMRI). In each image, voxel (volumetric pixel, see Figure 5.10) left) intensities roughly reflect the oxygen fraction (ratio of oxygenated/deoxygenated hemoglobin) in the blood, which is a proxy of metabolic activity.

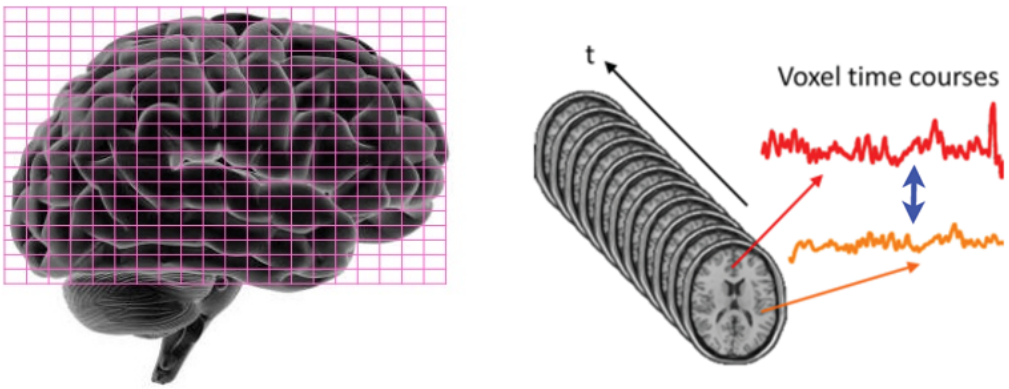

Figure 5.10: *Cartoon of the voxels (left) and of the extraction procedure of voxel time series (right).*

By considering the same voxels in different images one can extract an intensity time-series for each voxel (see Figure 5.10 right). This time series represents a variation

in time of the metabolic activity of the brain area captured by the hosting voxel. The typical number of voxels is $10^4 - 10^5$ and a typical scanning run yields $10^2 - 10^3$ images. To the best of our knowledge, a detailed study of the intrinsic dimensionality of fMRI time-series has not been carried out yet. Such study is nontrivial since one has to analyze a dataset of $10^4 - 10^5$ points in an embedding space of $10^2 - 10^3$ dimensions, but it is worthwhile for several reasons. First, a preliminary noise- or dimensionality-reduction step is often performed on the data before further analysis. The most common approaches, band-pass filtering or Principal Component Analysis, involve linear projections (125). In both cases significant signal loss can occur if the number of components to be retained is selected with a bad criterion. It is therefore interesting to provide an precise estimation of the number of components that ought to be kept. Secondly, while most fMRI studies focus on the local signals associated with specific brain areas to look how they relate to a task under execution or among each other, it may be also worth investigating global features of the signal. Apart from their general interest for the understanding of physiological processes (126), the global features of the signal may prove useful biomarkers. For example, it is known that gross characteristics of the power spectrum of the signal can be predictive of clinical status, or age (127).

By finding groups of voxels with different dimensionality, our method may uncover rich structure in the data, complementary to that found by standard analysis methods (linear response analysis, independent component analysis or network connectivity analysis).

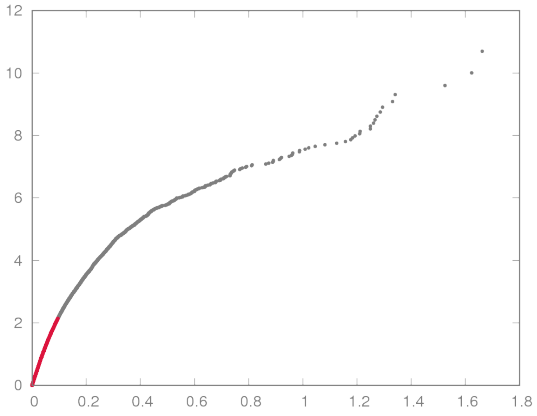

Figure 5.11: *S set (see Chapter 2) for the Brix dataset.*

In what follows, we apply our dimension estimation to two single-subject, single

run fMRI recordings taken from different experiments. The first dataset ("Brix") contains $D = 180$ images collected while the subject was performing the Brixton inductive reasoning task (128). We compute a distance between pairs of signals by using the dissimilarity metric devised in (129) (essentially, a Euclidean distance between signals).
By applying the TWO-NN model (see Chapter 2) to the Brix dataset, we obtain the S set shown in Figure 5.11. Also in this case, the strong departure from a linear behavior hints at the presence of more than one manifold with different dimensions. We now apply the multi-dimensional analysis of section 5.5 with $K = 2$, $q = 2$, and $\zeta = 0.7$. We find two manifolds M_1 and M_2 with parameters $d_1 \sim 12$, $d_2 \sim 37$, $p_1 \sim 0.4$, $p_2 \sim 0.6$ respectively. Thus, the data appear to be split between two manifolds of very different dimensions.
We further analyze this result by relating the Z of each voxel with a quantity F that is a measure of its relevance for the task. F can be obtained through a completely unrelated analysis (129), as follows:

1. from the whole series of 207 images, we select short series of 12 consecutive images with a "running-window" approach: the first window comprises images 1-12, the second images 2-13, and so forth

2. within each window t, we apply the analysis described in (129) to the voxel time-series restricted to that window. This analysis identifies a set of clusters (group of similar time-series) in the given window. Only a small subset A_t of voxels is assigned to clusters, while the others are discarded as not significant.

3. by collecting the results of all time windows, we compute the clustering frequency F for each voxel, counting the fraction of time windows t in which they belong to A_t. Essentially, F quantifies how frequently each voxel is involved in short-term coherent activation patterns along the whole run.

As extensively discussed in (129), voxels with high clustering frequency ($F \geq 0.5$) are likely to be "relevant" brain areas involved in the cognitive task at hand. In Figure 5.12 we show a histogram of F, where we expilitly show the fraction of points assigned to manifold M_1 (lower dimension) and M_2 (higher dimension). We find that the "relevant" voxels ($F > 0.5$) mostly belong to the manifold with low dimensionality, as is apparent from the inset of Figure 5.12.
This result finds a natural and appealing interpretation: that the subset of signals (those of the "relevant voxels") giving rise to coherent patterns exhibits a considerably

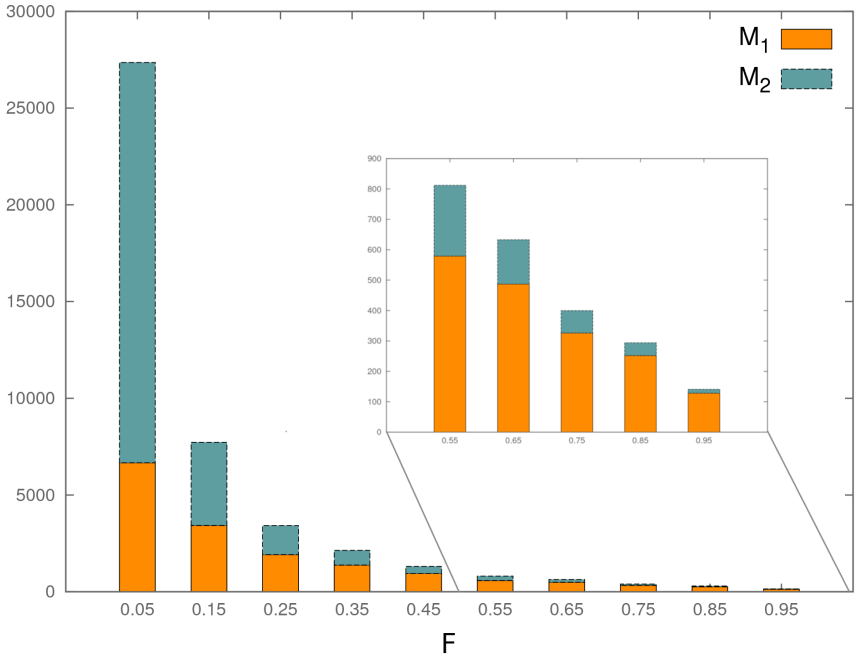

Figure 5.12: *Histogram of the clustering frequency F, showing the number of voxels assigned to manifold M_1 and M_2 respectively.*

reduced variability than the remainder of the signals that have a noisy and incoherent behavior. Again, this feature has such a strong impact on the global topology of the data that it can be revealed by our ID analysis, without need of much more refined analyses such as clustering or independent component analysis.

Appendix

A. Distribution of shells volumes for a homogeneous Poisson process

Let Φ be a homogeneous Poisson process in $\mathbb{R}^2$ with intensity λ (see (131) for more information about Poisson processes); in particular Φ satisfies the following properties:

i) for any disjoint Borel sets A_1 and A_2 the random variables $N(A_1)$ and $N(A_2)$ describing the number of points falling in A_1 and A_2 respectively are independent,

ii) the number of points $N(A)$ falling in a Borel set A is distributed as a Poisson variable with parameter $\lambda\mu(A)$, where $\mu(A)$ is the measure of A:

$$P(A \text{ contains exactly } n \text{ points}) \doteq P(n, A) = \frac{(\lambda\mu(A))^n}{n!} e^{-\lambda\mu(A)}$$

The intensity λ corresponds to the average density of points: $E[P(n, A)] = \lambda\mu(A)$. Moreover, the second property implies that in an infinitesimally small area dA there are no multiple points. From the definition of a Poisson process it also follows that the probability of having no points in a Borel set A (void probability) is given by:

$$P(0, A) = e^{-\lambda\mu(A)}. \tag{5.40}$$

Given a point o in Φ, let $d_1, d_2, ..., d_n$ be the ordered distances from o of the first n neighbours. If we define Δv_1 as the volume of the ball B_{o,d_1}, Δv_2 as the volume of the annulus C_{r_1,r_2}, and so on we see that the distances $d_1, d_2, ..., d_n$ identify n disjoint volumes $\Delta v_1, \Delta v_2, ..., \Delta v_n$ that can be seen as the volumes 'occupied' by the neighbours. We want to find an expression for the joint probability distribution

$g(\Delta v_1, \Delta v_2, ..., \Delta v_n)$. To this purpose, we start from a slightly easier problem and look for the joint probability distribution of the distances $f(d_1, d_2, ..., d_n)$.

The probability of the first distance d_1 to fall in an infinitesimally small annulus C_{r_1,r_1+dr_1} is given by the probability of having no points in the ball B_{o,r_1} and having at least one point in the annulus C_{r_1,r_1+dr_1}:

$$\begin{aligned} P(d_1 \in C_{r_1,r_1+dr_1}) =& P(N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) \\ =& P(N(B_{o,r_1}) = 0) P(N(C_{r_1,r_1+dr_1}) \geq 1) \\ =& P(N(B_{o,r_1}) = 0)(1 - P(N(C_{r_1,r_1+dr_1}) = 0) \\ =& e^{-\lambda r_1^2 \pi}(1 - e^{-\lambda \pi r_1 dr_1}). \end{aligned}$$

Here the second equality is due to independence property, while the last one comes from the formula for the void distribution. Since dr_1 is very small we conclude that

$$P(d_1 \in C_{r_1,r_1+dr_1}) \sim e^{-\lambda r_1^2 \pi} 2\pi\lambda r_1 dr_1. \tag{5.41}$$

The second step is to define the probability that the second nearest neighbour is found at a distance r_2 from o given that the first one is found at a distance r_1.

$$\begin{aligned} P(r_2 \mid r_1) \doteq& P(\text{the second nearest neighbour is at a distance } r_2 \text{ given that the first is at a distance } r_1) \\ =& P(\text{the second nearest neighbour is at a distance } r_2 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) \\ =& P(N(C_{r_1,r_2}) = 0, N(C_{r_2,r_2+dr_2}) \geq 1 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) \\ =& P(N(C_{r_1,r_2}) = 0 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) \cdot \\ & \cdot P(N(C_{r_2,r_2+dr_2}) \geq 1 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1). \end{aligned}$$

We can compute separately the two terms in the product using equation 5.40; the first term is straightforward:

$$P(N(C_{r_1,r_2}) = 0 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) = e^{-\lambda\pi(r_2^2 - r_1^2)} \ ,$$

while we can write the second term as
$1 - P(N(C_{r_2,r_2+dr_2}) = 0 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}))$,

so that

$$P(N(C_{r_2,r_2+dr_2}) \geq 1 \mid N(B_{o,r_1}) = 0, N(C_{r_1,r_1+dr_1}) \geq 1) = 1 - e^{-\lambda\pi r_2 dr_2} \sim 2\lambda\pi r_2 dr_2$$

Finally we obtain a formula for $P(r_2 \mid r_1)$:

$$P(r_2 \mid r_1) \sim e^{-\lambda\pi(r_2^2 - r_1^2)} 2\lambda\pi r_2 dr_2.$$

Now we can compute the joint probability $P(r_1, r_2)$:

$$P(r_1, r_2) = P(r_2 \mid r_1)P(r_1) \sim e^{-\lambda\pi r_2^2}(2\lambda\pi)^2 r_1 r_2 dr_1 dr_2.$$

This result can be generalized to the third neighbour:

$$P(r_1, r_2, r_3) = P(r_3 \mid r_1, r_2)P(r_2 \mid r_1)P(r_1) \sim e^{-\lambda\pi r_3^2}(2\lambda\pi)^3 r_1 r_2 r_3 dr_1 dr_2 dr_3,$$

and so on to the nth neighbor:

$$P(r_1, r_2, ..., r_n) \sim e^{-\lambda\pi r_n^2}(2\lambda\pi)^n r_1 r_2 \cdots r_n dr_1 dr_2 \cdots dr_n,$$

so that the expression for the joint probability distribution of the distances is given by:

$$f(r_1, ..., r_n) = e^{-\lambda\pi r_n^2}(2\lambda\pi)^n r_1 r_2 \cdots r_n.$$

Now, we are interested in the distribution of volumes. The change of variables $\alpha : (r_1, r_2, ..., r_n) \longmapsto (\Delta v_1, \Delta v_2, ..., \Delta v_n)$ defined as

$$(r_1, r_2, ..., r_n) \longmapsto (\pi r_1^2, \pi(r_2^2 - r_1^2), ..., \pi(r_n^2 - r_{n-1}^2))$$

is an omeomorphism on $\mathbb{R}^2_{>0}$; let β be the inverse. If we denote by $\mid D\beta \mid$ and $\mid D\alpha \mid$ the jacobians of β and α respectively, we obtain

$$\begin{aligned} g(\Delta v_1, \Delta v_2, ..., \Delta v_n) =& f(\beta(\Delta v_1, \Delta v_2, ..., \Delta v_n)) \mid D\beta \mid_{\mid_{\Delta v_1, \Delta v_2, ..., \Delta v_n}} \\ =& f(\beta(\Delta v_1, \Delta v_2, ..., \Delta v_n)) \mid D\alpha \mid^{-1}_{\mid_{\beta(\Delta v_1, \Delta v_2, ..., \Delta v_n)}} . \end{aligned}$$

Now we can easily compute the jacobian of α as

$$\mid D\alpha \mid_{\mid r_1, r_2, ..., r_n} = \pi^n 2^n r_1 \cdots r_n = (2\pi)^n (\beta(\Delta v_1), \beta(\Delta v_2), ..., \beta(\Delta v_n)).$$

Finally, the expression for g is given by:

$$g(\Delta v_1, \Delta v_2, ..., \Delta v_n) = \lambda^n e^{-\lambda(\Delta v_1 + \Delta v_2 + ... + \Delta v_n)},$$

so that the joint distribution of volumes is exponential with parameter equal to the average density of points.

This argument can be easily generalized to $\mathbb{R}^N$.

B. Explicit formula for $\mathcal{Z}(\zeta, N_{Z_i})$

Our basic assumption that *points sampled from the same manifold are more likely to be neighbors than points sampled from different manifolds* can be formalized as follows: we assume that

$$\mathcal{L}(\mathcal{N}_{ij}^{(q)} = 1 | Z_i = Z_j) \propto \zeta, \qquad \mathcal{L}(\mathcal{N}_{ij}^{(q)} = 1 | Z_i = Z_j) \propto 1 - \zeta \tag{5.42}$$

Given a point i, the number of its neighbors is $\sum_j \mathcal{N}_{ij}^{(q)} = q$. For a given configuration $\boldsymbol{\mathcal{N}}_i^{(q)} \equiv \{\mathcal{N}_{ij}^{(q)}, j = 1, \ldots, N\}$, $n_i^{in} \leq q$ neighbors are from the same manifold as i, and $q - n_i^{in}$ are from a different manifold. According to (5.42), we then have

$$\mathcal{L}(\boldsymbol{\mathcal{N}}_i^{(q)} | \mathbf{Z}) \propto \zeta^{n_i^{in}} (1 - \zeta)^{q - n_i^{in}}$$

(the problem is analogous to the problem where we have to select q balls from a box containing N balls, of which N_b are blue and N_r red. We pick a ball at random from the box: if it is blue, we keep it with probability ζ , otherwise we put it back in the box; if it is red, we keep it with probability $1 - \zeta$. We continue the process until we have collected q balls. The probability of a choice with n_b blue and $q - n_b$ red balls is then proportional to $\zeta^{n_b}(1 - \zeta)^{q - n_b}$). Function

$$\mathcal{Z}(\zeta, N_{Z_i}) = \sum_{\{\boldsymbol{\mathcal{N}}_i^{(q)}\}} \zeta^{n_i^{in}} (1 - \zeta)^{q - n_i^{in}}$$

can be evaluated as follows. The number of possible configurations of $\boldsymbol{\mathcal{N}}_i^{(q)} \equiv \{\mathcal{N}_{ij}^{(q)}, j = 1, \ldots, N\}$ is given by the combinations $\binom{N-1}{q}$. However, for a given n_i^{in}, the number of possible configurations is

$$\binom{N_{Z_i} - 1}{n_i^{in}} \binom{N - N_{Z_i}}{q - n_i^{in}}$$

which is the number of ways one can choose n_i^{in} points among the $N_{Z_i} - 1$ points from manifold Z_i (excluding point i) times the number of ways one can choose $q - n_i^{in}$ points among the $N - N_{Z_i}$ points from other manifolds (one can easily verify that $\sum_{n_i^{in}=0}^{q} \binom{N_{Z_i}-1}{n_i^{in}} \binom{N-N_{Z_i}}{q-n_i^{in}} = \binom{N-1}{q}$). Then function $\mathcal{Z}$ is given by

$$\mathcal{Z}(\zeta, N_{Z_i}) = \sum_{n_i^{in}=0}^{q} \binom{N_{Z_i} - 1}{n_i^{in}} \binom{N - N_{Z_i}}{q - n_i^{in}} \zeta^{n_i^{in}} (1 - \zeta)^{q - n_i^{in}} = \tag{5.43}$$

$$= (1-\zeta)^q \sum_{n_i^{in}=0}^{q} \binom{N_{Z_i}-1}{n_i^{in}} \binom{N-N_{Z_i}}{q-n_i^{in}} \left(\frac{\zeta}{1-\zeta}\right)^{n_i^{in}}.$$

The formula above can be expressed in a more compact form by using a hypergeometric function. By using the formula (Abramowitz and Stegun, 15.4.1)

$${}_2F_1(-m,b,c,z) = \sum_{n=0}^{m}(-)^n \binom{m}{n} \frac{(b)_n}{(c)_n} z^n.$$

We have

$${}_2F_1(-m,-b,c-m,z) = \sum_{n=0}^{m}(-)^n \binom{m}{n} \frac{(-b)_n}{(c-m)_n} z^n =$$

$$= \sum_{n=0}^{m}(-)^n \binom{m}{n} \frac{(-)^n b(b-1)\dots(b-n+1)}{(c-m)(c-m+1)\dots(c-m+n-1)} z^n =$$

$$= \sum_{n=0}^{m} \binom{m}{n}\binom{b}{n} n! \frac{(c-m+n-1)...(c+1)c}{(c-m)(c-m+1)\dots c} z^n =$$

$$= \frac{m!}{(c-m)(c-m+1)\dots c} \sum_{n=0}^{m} \frac{1}{(m-n)!} \binom{b}{n} (c-m+n-1)...(c+1)c\, z^n =$$

$$= \binom{c}{m}^{-1} \sum_{n=0}^{m} \binom{b}{n}\binom{c}{n-m} z^n$$

hence

$$\sum_{n=0}^{m} \binom{b}{n}\binom{c}{n-m} z^n = \binom{c}{m}\ {}_2F_1(-m,-b,c-m,z)$$

Therefore, we can write (5.43) as

$$\mathcal{Z}(\zeta, N_{Z_i}) = (1-\zeta)^q \binom{N-N_{Z_i}}{q} {}_2F_1(-q, 1-N_{Z_i}, N-N_{Z_i}-q, \frac{\zeta}{1-\zeta}).$$

In Fig. 5.13 we plot $\mathcal{Z}(\zeta, N_{Z_i})$ as a function of ζ and N_{Z_i} for $q = 5$, $N = 2000$. For $\zeta > 1/2$, $\mathcal{Z}(\zeta, N_{Z_i})$ is a monotonically increasing function of N_{Z_i}, while the opposite holds for $\zeta < 1/2$.

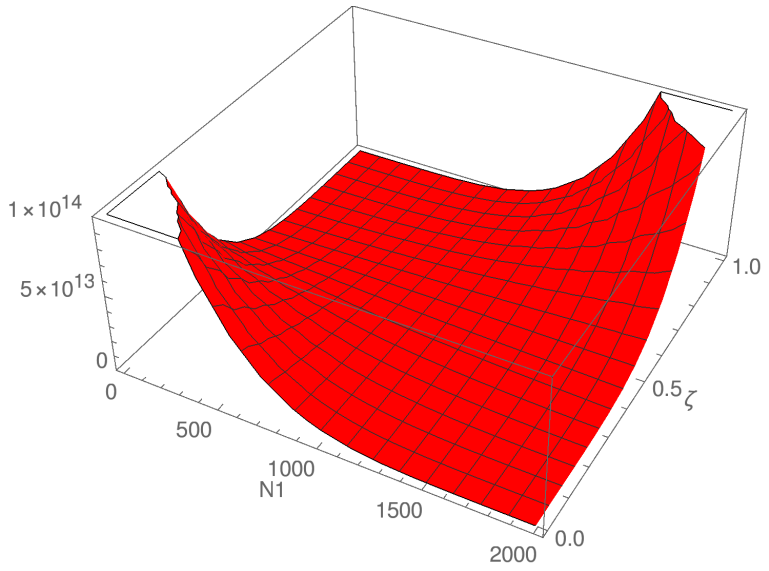

Figure 5.13: *$\mathcal{Z}(\zeta, N_{Z_i})$ as a function of ζ and N_{Z_i} for $q = 5$, $N = 2000$*

C. Computing the conditional probability of Z

We have (here as usual $\mathbf{Z}'$=$\mathbf{Z} - \{Z_i\}$)

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) = \frac{P_{post}(Z_i = k, \mathbf{Z}', \mathbf{d}, \mathbf{p})}{P_{post}(\mathbf{Z}', \mathbf{d}, \mathbf{p})}. \tag{5.44}$$

We can write this term extensively by means of 5.37:

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) = \frac{\prod_{j=1}^{N} \mathcal{L}(\mu_j|\mathbf{d}, Z_j)\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z}, \zeta)\ P_{prior}(\mathbf{d})P_{prior}(\mathbf{Z}|\mathbf{p})P_{prior}(\mathbf{p})P_{prior}(\zeta)}{\sum_{Z_i}\prod_{i=1}^{N} \mathcal{L}(\mu_j|\mathbf{d}, Z_j)\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z}, \zeta)\ P_{prior}(\mathbf{d})P_{prior}(\mathbf{Z}|\mathbf{p})P_{prior}(\mathbf{p})P_{prior}(\zeta)}.$$

We can simplify this equation by dropping the terms that do not depend on Z_i:

$$P_{post}(Z_i = k|\mathbf{dp}, \mathbf{Z}') = \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z}, \zeta)}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\mathcal{L}(\boldsymbol{\mathcal{N}}^{(q)}|\mathbf{Z}, \zeta)}.$$

Now, by using Eq. (5.28) we get

$$P_{post}(Z_i = k|\mathbf{dpZ}') = \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\prod_j \frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{q-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\prod_j \frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{q-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}} =$$

$$= \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\prod_j \frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\prod_j \frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}$$

By isolating the i-th term in the product $\prod_j$, we get

$$\frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\frac{\zeta^{n_i^{in}(\mathbf{Z})(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_i}(\mathbf{Z}))}\prod_{j\neq i}\frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\frac{\zeta^{n_i^{in}(\mathbf{Z})(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_i}(\mathbf{Z}))}\prod_{j\neq i}\frac{\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}.$$

If we separate the product of denominators we obtain:

$$\frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})}\prod_{j\neq i}\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}\prod_j \frac{1}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})}\prod_{j\neq i}\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{-n_j^{in}(\mathbf{Z})}\prod_j \frac{1}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}$$

Now, by equation (5.29) we get

$$\prod_{j\neq i}\zeta^{n_j^{in}(\mathbf{Z})}(1-\zeta)^{q-n_j^{in}(\mathbf{Z})} = \prod_{j\neq i}\zeta^{\sum_k \mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}(1-\zeta)^{-\sum_k \mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}} =$$

$$= \prod_{j\neq i}\zeta^{\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}+\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}(1-\zeta)^{-\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}-\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}} =$$

$$= \prod_{j\neq i}\zeta^{\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}}(1-\zeta)^{-\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}}\prod_{j\neq i}\zeta^{\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}(1-\zeta)^{-\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}$$

Note that $\prod_{j\neq i}\zeta^{\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}(1-\zeta)^{-\sum_{k\neq i}\mathcal{N}_{jk}^{(q)}\mathbb{I}_{Z_k=Z_j}}$ does not depend on Z_i, so it will cancel with a corresponding term in the denominator of Eq. (5.44). Moreover,

$$\prod_{j\neq i}\zeta^{\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}}(1-\zeta)^{-\mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}} = \zeta^{\sum_j \mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}}(1-\zeta)^{-\sum_j \mathcal{N}_{ji}^{(q)}\mathbb{I}_{Z_i=Z_j}} =$$

$$= \zeta^{m_i^{in}(\mathbf{Z})}(1-\zeta)^{-m_i^{in}(\mathbf{Z})},$$

where m_i^{in} is the number of points sampled from the same manifold that have i as

neighbor:

$$m_i^{in} = \sum_j \mathcal{N}_{ji}^{(q)} \mathbb{I}_{Z_j = Z_i}. \tag{5.45}$$

In this way we obtain:

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) = \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})-m_i^{in}(\mathbf{Z})}\prod_j \frac{1}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})-m_i^{in}(\mathbf{Z})}\prod_j \frac{1}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))}}.$$

Now, we have

$$\prod_j \frac{1}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))} = \prod_j \frac{\mathbb{I}_{Z_i=Z_j}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}))} \prod_j \frac{\mathbb{I}_{Z_i\neq Z_j}}{\mathcal{Z}(\zeta, N_{Z_j}(\mathbf{Z}'))} =$$

$$= \prod_j \frac{\mathbb{I}_{Z_i=Z_j}}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}')+1)} \prod_j \frac{\mathbb{I}_{Z_i\neq Z_j}}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}'))}$$

where $N'_{Z_j}(\mathbf{Z}') = \sum_{j\neq i} \mathbb{I}_{Z_i=Z_j}$.

The term $\prod_j \frac{\mathbb{I}_{Z_i\neq Z_j}}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}'))}$ can be very small, however, we can multiply both the numerator and denominator of Eq. (5.44) by

$$\prod_{j\neq i} \mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}'))$$

since this quantity is independent of Z_i. We get

$$\prod_j \frac{\mathbb{I}_{Z_i=Z_j}}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}')+1)} \prod_j \frac{\mathbb{I}_{Z_i\neq Z_j}}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}'))} \prod_{j\neq i} \mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}')) =$$

$$= \frac{1}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}')+1)} \prod_{j\neq i} \frac{\mathbb{I}_{Z_i=Z_j}\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}'))}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}')+1)} =$$

$$= \frac{1}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}')+1)} \left(\frac{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}'))}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}')+1)}\right)^{N'_{Z_i}(\mathbf{Z}')}$$

Thus, we finally have

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) =$$

$$= \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})-m_i^{in}(\mathbf{Z})}\frac{1}{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z})+1)}\left(\frac{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z}'))}{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z}')+1)}\right)^{N'_{Z_i}(\mathbf{Z}')}}{\sum_{Z_i} p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}, Z_i)\zeta^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})}(1-\zeta)^{-n_i^{in}(\mathbf{Z})-m_i^{in}(\mathbf{Z})}\frac{1}{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z})+1)}\left(\frac{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z}'))}{\mathcal{Z}(\zeta,N'_{Z_i}(\mathbf{Z}')+1)}\right)^{N'_{Z_i}(\mathbf{Z}')}},$$

i.e.

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) \propto \left(\frac{\zeta}{1-\zeta}\right)^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})} \frac{p_{Z_i}\mathcal{L}(\mu_i|\mathbf{d}\ Z_i)}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z})+1)}\left(\frac{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}'))}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z}')+1)}\right)^{N'_{Z_i}(\mathbf{Z}')} =$$

$$= \frac{p_{Z_i} d_{Z_i}(\mu_i)^{d_{Z_i}+1}}{\mathcal{Z}(\zeta, N'_{Z_i}(\mathbf{Z})+1)}\left(\frac{\zeta}{1-\zeta}\right)^{n_i^{in}(\mathbf{Z})+m_i^{in}(\mathbf{Z})}\left(\frac{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}))}{\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z})+1)}\right)^{N'_{Z_j}(\mathbf{Z})-1}.$$

For large $N_k \gg 1$ we have $\mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z})+1) \sim \mathcal{Z}(\zeta, N'_{Z_j}(\mathbf{Z}))$ so that it approximately holds

$$P_{post}(Z_i = k|\mathbf{Z}', \mathbf{d}, \mathbf{p}) \approx \frac{p_k d_k(\mu_i)^{d_k+1}}{\mathcal{Z}(\zeta, N_k)}\left(\frac{\zeta}{1-\zeta}\right)^{n_i^{in}+m_i^{in}}.$$

D. DANCo: an ID estimator exploiting nearest neighbors distances and angles

Consider a manifold $\mathcal{M} \equiv \mathbb{R}^d$ embedded in a higher dimensional space $\mathbb{R}^D$ through a locally isometric smooth map $\Psi : \mathbb{R}^d \longrightarrow \mathbb{R}^D$. Suppose that points are sampled from $\mathcal{M}$ according to a smooth function $f : \mathcal{M} \longrightarrow \mathbb{R}^+$.
Model each neighborhood of a point in the dataset as a set of points uniformly sampled from a d-dimensional hypersphere, where d is the dimension of the manifold; such hypersphere has radius equal to the neighborhood size defined by its k-th nearest neighbor and is centred on the given point. This model is exact when $N \to \infty$, so that the radius of the k-points ball goes to zero. First of all we need to define the probabilistic density function relative to the distances of points uniformly sampled from the hypersphere from its center. Let $\mathcal{B}_d(0_d, 1) \subset \mathbb{R}^d$ be the unit hypersphere centered in 0_d, and let $\{z_i\}_{i=1}^k$ be k points uniformly drawn from it; The aim is to find the pdf $g(r; d; k)$ of $min_{i\in\{1,\dots,k\}}\|z_i\| = r$.
Let $p(r)$ be the pdf of the event $\|z_i\| = r$ $(r \in [0, 1])$, where $\|\cdot\|$ is the L_2 norm operator; let $F(r)$ be the probability of the event $\|z_i\| < r$.
The volume of a d dimensional hypersphere with radius r is:

$$V_r = r^d \frac{\pi^{(\frac{d}{2})}}{\Gamma(\frac{d}{2}+1)} = r^d V_1.$$

So $F(r) = \frac{V_r}{V_1} = r^d$, and to obtain p we just have to derive F with respect to r obtaining $p(r) = dr^{d-1}$.
now $g(r; d; k)$ is proportional to the probability of drawing one point at distance r multiplied by that of drawing $k - 1$ points at distance greater that r; after normalization we obtain:

$$g(r; d; k) = kdr^{d-1}(1 - r^d)^{k-1}.$$

Now consider a sample $x_N \doteq \{x_i\}_{i=1}^N = \{\psi(z_i)\}_{i=1}^N \subset \mathbb{R}^D$, where points z_i are i.i.d.drawn from our manifold $\mathcal{M} \equiv \mathbb{R}^d$ according to f.
For every point x_i let $\hat{x}_{k+1}(x_i) \doteq \{\hat{x}_j\}_{j=1}^{k+1} \subset x_N$ be the set of $k+1$ nearest neighbors of x_i ordered according to the distance from x_i, and let $\hat{x}(x_i) \doteq \hat{x}_{k+1}(x_i)$ be the furthest.
Define $\rho(x_i)$ as the distance between x_i and $\hat{x}(x_i)$ normalized by the distance between x_i and x_1:

$$\rho(x_i) \doteq \frac{\|x_i - x_1\|}{\|x_i - \hat{x}\|}.$$

Suppose that the quantities $\rho(x_i)$ are samples drawn from the pdf $g(r;d;k)$, where k is a parameter and d is to be estimated. To this extent a natural approach is to maximize the log-likelihood function

$$\begin{aligned} ll(d) &= \sum_{x_i \in x_N} \log\, g(r;d;k) \\ &= N\log\, k + N\log\, d + (d-1)\sum_{x_i \in x_N} \log\, \rho(x_i) + (k-1)\sum_{x_i \in x_N} \log\, (1-\rho(x_i)^d). \end{aligned}$$

A way to estimate d is then finding

$$\hat{d} = \underset{d \in \{1,\dots,D\}}{\operatorname{argmax}}\, ll(d),$$

where D is an upper bound for the dimension of the sample. To obtain a more reliable estimate of the i.d. one can minimize the Kullback-Leibler divergence between the pdf of the distances of the neighbors points of the dataset and those calculated on synthetic data of known dimensionality.

Once k is fixed $g(r;k;d)$ is a finite family of D pdfs for all the parameters values $1 \leq d \leq D$, so we can compare each of the D theoretical pdfs obtained on synthetic uniform datasets with the distribution obtained from the sample and take the d that minimizes the Kullback-Leibler divergence. The point is then defining a closed form of the KL Divergence.
For each $1 \leq d \leq D$ we compute the estimated dimension $d1_d$ of d-dimensional uniform hyperspheres. For each d the KL divergence between the synthetic distribution with parameter $d1_d$ and the sample one with parameter $d2$ is given by:

$$\begin{aligned} \overline{KL_d} &\doteq \mathcal{KL}(g(\cdot;k;d1_d), g(\cdot;k;d2)) \\ &= \int_0^1 g(r;k;d1_d)\, \log\, \frac{g(r;k;d1_d)}{g(r;k;d2)} \mathrm{d}r \\ &= \mathcal{H}_k \frac{d2}{d1_d} - 1 - \mathcal{H}_{k-1} - \log\, \frac{d2}{d1_d} - (k-1)\sum_{i=0}^{k} (-1)^i \binom{k}{i} \Psi\left(1 + \frac{i \times d1_d}{d2}\right), \end{aligned}$$

where $\mathcal{KL}(\cdot,\cdot)$ is the Kullback-Leibler divergence operator, $\mathcal{H}_k$ is the k-th harmonic number $\left(\mathcal{H}_k \doteq \sum_{i=0}^{k} \frac{1}{i}\right)$, and $\Psi(\cdot)$ is the digamma function.

To obtain a more reliable estimation it is useful to exploit information about mutual angles. Following the procedure employed with distances we can explicit

the Kullback-Leibler Divergence between the distribution on the data and the one obtained on uniform hyperspheres. First of all it is necessary to model the distribution of angles.
Working on pairwise angles separately allows to use the Von Mises distribution: if $\theta \in [-\pi, \pi]$ is the angle between two uniformly sampled vectors, the VM distribution of θ is given by:

$$q(\theta; \nu, \tau) = \frac{e^{\tau \cos(\theta - \nu)}}{2\pi I_0(\tau)} \chi_{[-\pi,\pi]}(\theta),$$

where I_0 is the modified Bessel function of the first kind with order 0, and τ is a concentration parameter that gets high values in the case of a high concentration of the distribution around the mean direction. The VM distribution is also known as the circular normal distribution, as it is a close approximation to the wrapped normal distribution (the circular analogue of the normal distribution). Let $\{\theta_1, ..., \theta_N\}$ be a sample drawn from a VM distribution with parameters ν and τ. Then the ML of ν equals the sample mean direction:

$$\hat{\nu} = \text{atan}_2 \left(\sum_{i=1}^N \sin \theta_i, \sum_{i=1}^N \cos \theta_i \right).$$

The ML of τ is obtained in the following way: let

$$\eta \doteq \sqrt{\left(\frac{1}{N} \sum_{i=1}^N \sin \theta_i \right)^2 + \left(\frac{1}{N} \sum_{i=1}^N \cos \theta_i \right)^2}.$$

Then

$$\hat{\tau} = \begin{cases} 2\eta + \eta^3 + \frac{5\eta^5}{6} & \eta < 0.53 \\ -0.4 + 1.39\eta + \frac{0.43}{1-\eta} & 0.53 \leq \eta < 0.85 \\ \frac{1}{\eta^3 - 4\eta^2 + 3\eta} & \eta \geq 0.85 \end{cases}.$$

Once we have ML estimates for ν and τ, we can define a closed form of the Kullback-Leibler divergence between two VM pdfs of parameters $\nu 1_d$, $\tau 1_d$ and $\nu 2$, $\tau 2$ in the following way:

$$\begin{aligned} \overline{KL}_{\nu,\tau} &\doteq \mathcal{KL}(q(\cdot; \nu 1_d, \tau 1_d), q(\cdot; \nu 2, \tau 2)) \\ &= \int_{-\pi}^{\pi} q(\cdot; \nu 1_d, \tau 1_d) \log \frac{q(\cdot; \nu 1_d, \tau 1_d)}{q(\cdot; \nu 2, \tau 2)} \mathrm{d}r \\ &= \log \frac{I_0(\tau 2)}{I_0(\tau 1_d)} + \frac{I_1(\tau 1_d) - I_1(-\tau 1_d)}{2I_0(\tau 1_d)} (\tau 1_d - \tau 2 \cos(\nu 2 - \nu 1_d)). \end{aligned}$$

Notice that if parameters $\nu 1_d$, $\tau 1_d$ are obtained from a uniform unit hypersphere sample in dimension d, they will implicitly depend on d, even if such dependence is not evident from the definition.
We want to compare the joint pdf $h(r, \theta)$ of the nearest neighbor distances r and pairwise angles θ related to the real dataset with the D pdfs computed on samples drawn from hyperspheres of increasing dimensionality $h_d(r, \theta)$, $d = 1, ..., D$; the estimate for d is given by:

$$\hat{d} = \arg\min_{d \in \{1,...,D\}} \int_{-\pi}^{\pi} \int_{0}^{1} h_d(r, \theta) \log \left(\frac{h_d(r,\theta)}{h(r,\theta)} \right) \mathrm{d}r\mathrm{d}\theta.$$

It can be proved that the norm distribution $g(r; k; d)$ and the angle distribution $q(\theta; \nu; \tau)$ are independent when the data are uniformly drawn from a spherical distribution, so that the joint pdf factorizes in the product of the two marginals:

$$h_d(r, \theta) = g(r; k; d)q(\theta; \nu; \tau);$$

we are then able to split the Kullback-Leibler Divergence $\overline{KL}_{d,\nu,\tau}$ between $h_d(r, \theta)$ and $h(r, \theta)$ into the sum of $\overline{KL}_d$ and $\overline{KL}_{\nu,\tau}$:

$$\overline{KL}_{d,\nu,\tau} = \overline{KL}_d + \overline{KL}_{\nu,\tau}.$$

At this point

$$\hat{d} = \underset{d \in \{1,...,D\}}{\operatorname{argmax}} \overline{KL}_{d,\nu,\tau}.$$

.

Acknowledgements

I want to thank my supervisor, Alessandro Laio, and my colleagues and friends Michele Allegra, Alex Rodriguez and Maria d'Errico for being always present and patient, and especially positive. I also want to acknowledge the people who actively contributed in the work presented here: Daniele Granata for the development of the method in chapter 2, Andrea Pagnani, Elena Tea Russo and Marco Punta for the project presented in Chapter 3, Antonietta Mira for the project presented in 5. I thank also Francesca Rizzato and Giovanni Pinamonti for the useful advice and discussions, together with all the group.
My most grateful thanks go to my parents, for being always there, supportive and encouraging, to my brother, and to Carlo.

Bibliography

[1] M. Chen, S. Mao, and Y. Liu, "Big data: a survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[2] K. Fukunaga, *Introduction to statistical pattern recognition*. Academic press, 2013.

[3] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[4] D. Granata and V. Carnevale, "Accurate estimation of the intrinsic dimension using graph distances: Unraveling the geometric complexity of datasets," *Scientific Reports*, vol. 6, 2016.

[5] L. Molgedey and H. G. Schuster, "Separation of a mixture of independent signals using time delayed correlations," *Physical review letters*, vol. 72, no. 23, p. 3634, 1994.

[6] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 5, pp. 522–532, 1998.

[7] R. D. Finn, J. Clements, and S. R. Eddy, "Hmmer web server: interactive sequence similarity searching," *Nucleic acids research*, vol. 39, no. suppl_2, pp. W29–W37, 2011.

[8] S. Cocco and R. Monasson, "Adaptive cluster expansion for inferring boltzmann machines with noisy data," *Physical review letters*, vol. 106, no. 9, p. 090601, 2011.

[9] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, pp. 1492–1496, June 2014.

[10] K. Lindorff-Larsen, S. Piana, R. O. Dror, and D. E. Shaw, "How fast-folding proteins fold," *Science*, vol. 334, no. 6055, pp. 517–520, 2011.

[11] C. M. Bishop, *Neural networks for pattern recognition*. Oxford university press, 1995.

[12] P. Campadelli, E. Casiraghi, C. Ceruti, and A. Rozza, "Intrinsic dimension estimation: Relevant techniques and a benchmark framework," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[13] F. Camastra and A. Staiano, "Intrinsic dimension estimation: Advances and open problems," *Information Sciences*, vol. 328, pp. 26–41, 2016.

[14] J. B. Kruskal and M. Wish, *Multidimensional scaling*, vol. 11. Sage, 1978.

[15] T. F. Cox and M. A. Cox, *Multidimensional scaling*. CRC press, 2000.

[16] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[17] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on computers*, vol. 100, no. 5, pp. 401–409, 1969.

[18] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[19] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[20] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[21] C. M. Bishop, "Bayesian pea," in *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pp. 382–388, 1999.

[22] J. Li and D. Tao, "Simple exponential family pca," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 453–460, 2010.

[23] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of computational and graphical statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[24] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International Conference on Artificial Neural Networks*, pp. 583–588, Springer, 1997.

[25] A. V. Little, M. Maggioni, and L. Rosasco, "Multiscale geometric methods for data sets i: Multiscale svd, noise and curvature," *Applied and Computational Harmonic Analysis*, 2016.

[26] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors," in *The Theory of Chaotic Attractors*, pp. 170–189, Springer, 2004.

[27] J.-P. Eckmann and D. Ruelle, "Fundamental limitations for estimating dimensions and lyapunov exponents in dynamical systems," *Physica D: Nonlinear Phenomena*, vol. 56, no. 2-3, pp. 185–187, 1992.

[28] F. Camastra and A. Vinciarelli, "Estimating the intrinsic dimension of data with a fractal-based method," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 10, pp. 1404–1407, 2002.

[29] J. A. Costa and A. O. Hero, "Geodesic entropic graphs for dimension and entropy estimation in manifold learning," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2210–2221, 2004.

[30] J. A. Costa and A. O. Hero, "Learning intrinsic dimension and intrinsic entropy of high-dimensional datasets," in *Signal Processing Conference, 2004 12th European*, pp. 369–372, IEEE, 2004.

[31] J. A. Costa, A. Girotra, and A. Hero, "Estimating local intrinsic dimension with k-nearest neighbor graphs," in *Statistical Signal Processing, 2005 IEEE/SP 13th Workshop on*, pp. 417–422, IEEE, 2005.

[32] K. W. Pettis, T. A. Bailey, A. K. Jain, and R. C. Dubes, "An intrinsic dimensionality estimator from near-neighbor information," *IEEE Transactions on pattern analysis and machine intelligence*, no. 1, pp. 25–37, 1979.

[33] P. J. Verveer and R. P. W. Duin, "An evaluation of intrinsic dimensionality estimators," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 17, no. 1, pp. 81–86, 1995.

[34] E. Levina and P. J. Bickel, "Maximum likelihood estimation of intrinsic dimension," in *Advances in neural information processing systems*, pp. 777–784, 2004.

[35] D. L. Snyder and M. I. Miller, *Random point processes in time and space*. Springer Science & Business Media, 2012.

[36] M. D. Gupta and T. S. Huang, "Regularized maximum likelihood for intrinsic dimension estimation," *arXiv preprint arXiv:1203.3483*, 2012.

[37] A. Rozza, G. Lombardi, C. Ceruti, E. Casiraghi, and P. Campadelli, "Novel high intrinsic dimensionality estimators," *Machine learning*, vol. 89, no. 1-2, pp. 37–65, 2012.

[38] C. Ceruti, S. Bassis, A. Rozza, G. Lombardi, E. Casiraghi, and P. Campadelli, "Danco: An intrinsic dimensionality estimator exploiting angle and norm concentration," *Pattern recognition*, vol. 47, no. 8, pp. 2569–2581, 2014.

[39] M. Hein and J.-Y. Audibert, "Intrinsic dimensionality estimation of submanifolds in r d," in *Proceedings of the 22nd international conference on Machine learning*, pp. 289–296, ACM, 2005.

[40] M. Brito, A. Quiroz, and J. E. Yukich, "Intrinsic dimension identification via graph-theoretic methods," *Journal of Multivariate Analysis*, vol. 116, pp. 263–277, 2013.

[41] R. Badii and A. Politi, "Hausdorff dimension and uniformity factor of strange attractors," *Physical review letters*, vol. 52, no. 19, p. 1661, 1984.

[42] G. Pinamonti, J. Zhao, D. E. Condon, F. Paul, F. Noè, D. H. Turner, and G. Bussi, "Predicting the kinetics of rna oligonucleotides using markov state models," *Journal of Chemical Theory and Computation*, vol. 13, no. 2, pp. 926–934, 2017. PMID: 28001394.

[43] S. Pronk, S. Páll, R. Schulz, P. Larsson, P. Bjelkmar, R. Apostolov, M. R. Shirts, J. C. Smith, P. M. Kasson, D. van der Spoel, B. Hess, and E. Lindahl, "Gromacs 4.5: a high-throughput and highly parallel open source molecular simulation toolkit," *Bioinformatics*, vol. 29, no. 7, p. 845, 2013.

[44] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, 2014.

[45] C. B. Anfinsen, E. Haber, M. Sela, and F. White, "The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain," *Proceedings of the National Academy of Sciences*, vol. 47, no. 9, pp. 1309–1314, 1961.

[46] J. C. Kendrew, G. Bodo, H. M. Dintzis, R. Parrish, H. Wyckoff, and D. C. Phillips, "A three-dimensional model of the myoglobin molecule obtained by x-ray analysis," *Nature*, vol. 181, no. 4610, pp. 662–666, 1958.

[47] K. Wüthrich, "The way to nmr structures of proteins," *Nature Structural & Molecular Biology*, vol. 8, no. 11, pp. 923–925, 2001.

[48] H. M. Berman, W. K. Olson, D. L. Beveridge, J. Westbrook, A. Gelbin, T. Demeny, S.-H. Hsieh, A. Srinivasan, and B. Schneider, "The nucleic acid database. a comprehensive relational database of three-dimensional structures of nucleic acids," *Biophysical journal*, vol. 63, no. 3, pp. 751–759, 1992.

[49] U. Consortium *et al.*, "Uniprot: a hub for protein information," *Nucleic acids research*, p. gku989, 2014.

[50] R. D. Finn, P. Coggill, R. Y. Eberhardt, S. R. Eddy, J. Mistry, A. L. Mitchell, S. C. Potter, M. Punta, M. Qureshi, A. Sangrador-Vegas, *et al.*, "The pfam protein families database: towards a more sustainable future," *Nucleic acids research*, vol. 44, no. D1, pp. D279–D285, 2016.

[51] M. O. Dayhoff, "Atlas of protein sequence and structure," 1965.

[52] S. Henikoff and J. G. Henikoff, "Automated assembly of protein blocks for database searching," *Nucleic acids research*, vol. 19, no. 23, pp. 6565–6572, 1991.

[53] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[54] S. R. Eddy, "Profile hidden markov models.," *Bioinformatics (Oxford, England)*, vol. 14, no. 9, pp. 755–763, 1998.

[55] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena scientific Belmont, MA, 1995.

[56] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[57] W. R. Pearson, "Searching protein sequence libraries: comparison of the sensitivity and selectivity of the smith-waterman and fasta algorithms," *Genomics*, vol. 11, no. 3, pp. 635–650, 1991.

[58] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[59] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of molecular biology*, vol. 215, no. 3, pp. 403–410, 1990.

[60] S. Karlin and S. F. Altschul, "Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes," *Proceedings of the National Academy of Sciences*, vol. 87, no. 6, pp. 2264–2268, 1990.

[61] L. Wang and T. Jiang, "On the complexity of multiple sequence alignment," *Journal of computational biology*, vol. 1, no. 4, pp. 337–348, 1994.

[62] J. D. Thompson, T. Gibson, D. G. Higgins, *et al.*, "Multiple sequence alignment using clustalw and clustalx," *Current protocols in bioinformatics*, pp. 2–3, 2002.

[63] R. C. Edgar, "Muscle: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[64] P. Di Tommaso, S. Moretti, I. Xenarios, M. Orobitg, A. Montanyola, J.-M. Chang, J.-F. Taly, and C. Notredame, "T-coffee: a web server for the multiple sequence alignment of protein and rna sequences using structural information and homology extension," *Nucleic acids research*, vol. 39, no. suppl_2, pp. W13–W17, 2011.

[65] F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Söding, *et al.*, "Fast, scalable generation of high-

quality protein multiple sequence alignments using clustal omega," *Molecular systems biology*, vol. 7, no. 1, p. 539, 2011.

[66] A. M. Waterhouse, J. B. Procter, D. M. Martin, M. Clamp, and G. J. Barton, "Jalview version 2—a multiple sequence alignment editor and analysis workbench," *Bioinformatics*, vol. 25, no. 9, pp. 1189–1191, 2009.

[67] J. Söding, "Protein homology detection by hmm–hmm comparison," *Bioinformatics*, vol. 21, no. 7, pp. 951–960, 2004.

[68] B. Boeckmann, A. Bairoch, R. Apweiler, M.-C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'donovan, I. Phan, *et al.*, "The swiss-prot protein knowledgebase and its supplement trembl in 2003," *Nucleic acids research*, vol. 31, no. 1, pp. 365–370, 2003.

[69] W. Li and A. Godzik, "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, p. 1658, 2006.

[70] L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li, "Cd-hit: accelerated for clustering the next-generation sequencing data," *Bioinformatics*, vol. 28, no. 23, p. 3150, 2012.

[71] L. Holm and C. Sander, "Removing near-neighbour redundancy from large protein sequence collections.," *Bioinformatics (Oxford, England)*, vol. 14, no. 5, pp. 423–429, 1998.

[72] V. Hollich, L. Milchert, L. Arvestad, and E. L. L. Sonnhammer, "Assessment of protein distance measures and tree-building methods for phylogenetic tree reconstruction," *Molecular Biology and Evolution*, vol. 22, no. 11, pp. 2257–2264, 2005.

[73] S. Baraty, D. Simovici, and C. Zara, "The impact of triangular inequality violations on medoid-based clustering," *Foundations of Intelligent Systems*, pp. 280–289, 2011.

[74] M. F. Barnsley, *Superfractals*. Cambridge University Press, 2006.

[75] M. Nei and J. Zhang, "Evolutionary distance: estimation," *eLS*, 2005.

[76] P. Rice, I. Longden, and A. Bleasby, "Emboss: the european molecular biology open software suite," 2000.

[77] M. Kimura, *The neutral theory of molecular evolution*. Cambridge University Press, 1983.

[78] N. Takezaki, A. Rzhetsky, and M. Nei, "Phylogenetic test of the molecular clock and linearized trees.," *Molecular biology and evolution*, vol. 12, no. 5, pp. 823–833, 1995.

[79] S. Mantaci, A. Restivo, and M. Sciortino, "Distance measures for biological sequences: Some recent approaches," *International Journal of Approximate Reasoning*, vol. 47, no. 1, pp. 109–124, 2008.

[80] S. Cocco, C. Feinauer, M. Figliuzzi, R. Monasson, and M. Weigt, "Inverse statistical physics of protein sequences: A key issues review," *arXiv preprint arXiv:1703.01222*, 2017.

[81] F. Morcos, A. Pagnani, B. Lunt, A. Bertolino, D. S. Marks, C. Sander, R. Zecchina, J. N. Onuchic, T. Hwa, and M. Weigt, "Direct-coupling analysis of residue coevolution captures native contacts across many protein families," *Proceedings of the National Academy of Sciences*, vol. 108, no. 49, pp. E1293–E1301, 2011.

[82] M. Weigt, R. A. White, H. Szurmant, J. A. Hoch, and T. Hwa, "Identification of direct residue contacts in protein–protein interaction by message passing," *Proceedings of the National Academy of Sciences*, vol. 106, no. 1, pp. 67–72, 2009.

[83] J. P. Barton, E. De Leonardis, A. Coucke, and S. Cocco, "Ace: adaptive cluster expansion for maximum entropy graphical model inference," *Bioinformatics*, vol. 32, no. 20, pp. 3089–3097, 2016.

[84] S. J. Sheather, "Density Estimation," *Statistical Science*, vol. 19, pp. 588–597, 11 2004.

[85] B. W. Silverman, *Density estimation for statistics and data analysis*. Chapman and Hall, New York, 1986.

[86] A. J. Izenman, "Review Papers: Recent Developments in Nonparametric Density Estimation," *Journal of The American Statistical Association*, vol. 86, pp. 205–224, 1991.

[87] E. Fix and J. L. Hodges Jr, "Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties," *International Statistical Review / Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989.

[88] D. B. R. A. P. Dempster, N. M. Laird, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[89] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[90] M. P. Wand and M. C. Jones, *Kernel smoothing*. Crc Press, 1994.

[91] É. A. Nadaraya, "On Non-Parametric Estimates of Density Functions and Regression Curves," *Th. Prob. Appl.*, vol. 10, pp. 186–190, ???? 1965.

[92] G. S. Watson and M. R. Leadbetter, "On the Estimation of the Probability Density, I," *Ann. Math. Statist.*, vol. 34, pp. 480–491, 06 1963.

[93] Y. Mack and M. Rosenblatt, "Multivariate k-nearest neighbor density estimates," *Journal of Multivariate Analysis*, vol. 9, no. 1, pp. 1–15, 1979.

[94] J. Orava, "K-nearest neighbour kernel density estimation, the choice of optimal k," *Tatra Mountains Mathematical Publications*, vol. 50, no. 1, pp. 39–50, 2011.

[95] N.-B. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth selection for kernel density estimation: a review of fully automatic selectors," *AStA Advances in Statistical Analysis*, vol. 97, no. 4, pp. 403–433, 2013.

[96] M. C. Jones, J. S. Marron, and S. J. Sheather, "A brief survey of bandwidth selection for density estimation," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 401–407, 1996.

[97] J. S. Simonoff, *Smoothing methods in statistics*. Springer Science & Business Media, 2012.

[98] M. Jones, J. S. Marron, and S. Sheather, "Progress in data-based bandwidth selection for kernel density estimation," *Computational Statistics*, vol. 11, no. 3, pp. 337–381, 1996.

[99] O. V. Lepskii, "A problem of adaptive estimation in gaussian white noise," *Theory of Probability & Its Applications*, vol. 35, no. 2, pp. 459–470, 1990.

[100] O. V. Lepski, E. Mammen, and V. G. Spokoiny, "Optimal spatial adaptation to inhomogeneous smoothness: an approach based on kernel estimates with variable bandwidth selectors," *Ann. Statist.*, vol. 25, pp. 929–947, 06 1997.

[101] J. Polzehl and V. Spokoiny, "Image denoising: pointwise adaptive approach," *Ann. Statist.*, vol. 31, pp. 30–57, 02 2003.

[102] J. Polzehl and V. Spokoiny, "Propagation-separation approach for local likelihood estimation," *Probability Theory and Related Fields*, vol. 135, no. 3, pp. 335–362, 2006.

[103] F. Gach, R. Nickl, and V. Spokoiny, "Spatially adaptive density estimation by localised haar projections," *Ann. Inst. H. Poincaré Probab. Statist.*, vol. 49, pp. 900–914, 08 2013.

[104] G. Rebelles, "Pointwise adaptive estimation of a multivariate density under independence hypothesis," *Bernoulli*, vol. 21, no. 4, pp. 1984–2023, 2015.

[105] S. M. A. Becker and P. Mathé, "A different perspective on the propagation-separation approach," *Electron. J. Statist.*, vol. 7, pp. 2702–2736, 2013.

[106] D. H. D. Martinez, F. Mémoli, and W. Mio, "Multiscale covariance fields, local scales, and shape transforms," in *Geometric Science of Information*, pp. 794–801, Springer, 2013.

[107] D. H. D. Martínez, C. H. Lee, P. T. Kim, and W. Mio, "Probing the geometry of data with diffusion fréchet functions," *arXiv preprint arXiv:1605.04955*, 2016.

[108] J. S. Rosenthal, *A first look at rigorous probability theory*. World Scientific Publishing Co Inc, 2006.

[109] J. Neyman and E. S. Pearson, "On the problem of the most efficient tests of statistical hypotheses," *Philosophical Transactions of the Royal Society of*

London. Series A, Containing Papers of a Mathematical or Physical Character, vol. 231, pp. 289–337, 1933.

[110] S. S. Wilks, "The large-sample distribution of the likelihood ratio for testing composite hypotheses," *The Annals of Mathematical Statistics*, vol. 9, no. 1, pp. 60–62, 1938.

[111] W. H. Greene, "Econometric analysis 4th edition," *International edition, New Jersey: Prentice Hall*, 2000.

[112] L. Demortier and L. Lyons, "Everything you always wanted to know about pulls," *CDF note*, vol. 43, 2002.

[113] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Second International Conference on Knowledge Discovery and Data Mining* (E. Simoudis, J. Han, and U. M. Fayyad, eds.), vol. 96, pp. 226–231, AAAI Press, 1996.

[114] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *SIGMOD Rec.*, vol. 28, pp. 49–60, June 1999.

[115] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[116] D. Wales, *Energy landscapes: Applications to clusters, biomolecules and glasses*. Cambridge University Press, 2003.

[117] J. A. Hartigan, "Consistency of single linkage for high-density clusters," *Journal of the American Statistical Association*, vol. 76, no. 374, pp. 388–394, 1981.

[118] R. D. Finn, J. Mistry, B. Schuster-Böckler, S. Griffiths-Jones, V. Hollich, T. Lassmann, S. Moxon, M. Marshall, A. Khanna, R. Durbin, *et al.*, "Pfam: clans, web tools and services," *Nucleic acids research*, vol. 34, no. suppl_1, pp. D247–D251, 2006.

[119] M. Madera, "Profile comparer: a program for scoring and aligning profile hidden markov models," *Bioinformatics*, vol. 24, no. 22, pp. 2630–2631, 2008.

[120] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *Journal of intelligent information systems*, vol. 17, no. 2, pp. 107–145, 2001.

[121] T. Lin and H. Zha, "Riemannian manifold learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 796–809, 2008.

[122] S. Richardson and P. J. Green, "On bayesian analysis of mixtures with an unknown number of components (with discussion)," *Journal of the Royal Statistical Society: series B (statistical methodology)*, vol. 59, no. 4, pp. 731–792, 1997.

[123] J.-M. Marin, K. Mengersen, and C. P. Robert, "Bayesian modelling and inference on mixtures of distributions," *Handbook of statistics*, vol. 25, pp. 459–507, 2005.

[124] E. Ott, *Chaos in dynamical systems*. Cambridge university press, 2002.

[125] R. A. Poldrack, J. A. Mumford, and T. E. Nichols, *Handbook of functional MRI data analysis*. Cambridge University Press, 2011.

[126] S. M. Pincus and A. L. Goldberger, "Physiological time-series analysis: what does regularity quantify?," *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 266, no. 4, pp. H1643–H1656, 1994.

[127] M. D'Esposito, L. Y. Deouell, and A. Gazzaley, "Alterations in the bold fmri signal with ageing and disease: a challenge for neuroimaging," *Nature reviews. Neuroscience*, vol. 4, no. 11, p. 863, 2003.

[128] C. Crescentini, S. Seyed-Allaei, N. De Pisapia, J. Jovicich, D. Amati, and T. Shallice, "Mechanisms of rule acquisition and rule following in inductive reasoning," *Journal of Neuroscience*, vol. 31, no. 21, pp. 7763–7774, 2011.

[129] M. Allegra, S. Seyed-Allaei, F. Pizzagalli, F. Baftizadeh, M. Maieron, C. Reverberi, A. Laio, and D. Amati, "fmri single trial discovery of spatio-temporal brain activity patterns," *Human brain mapping*, vol. 38, no. 3, pp. 1421–1437, 2017.

[130] G. A. Tribello, M. Ceriotti, and M. Parrinello, "Using sketch-map coordinates to analyze and bias molecular dynamics simulations," *Proceedings of the National Academy of Sciences*, vol. 109, no. 14, pp. 5196–5201, 2012.

[131] D. Moltchanov, "Distance distributions in random networks," *Ad Hoc Networks*, vol. 10, no. 6, pp. 1146–1166, 2012.