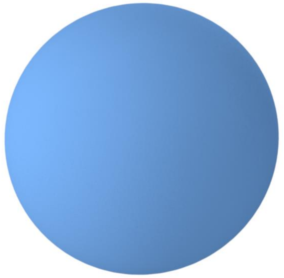
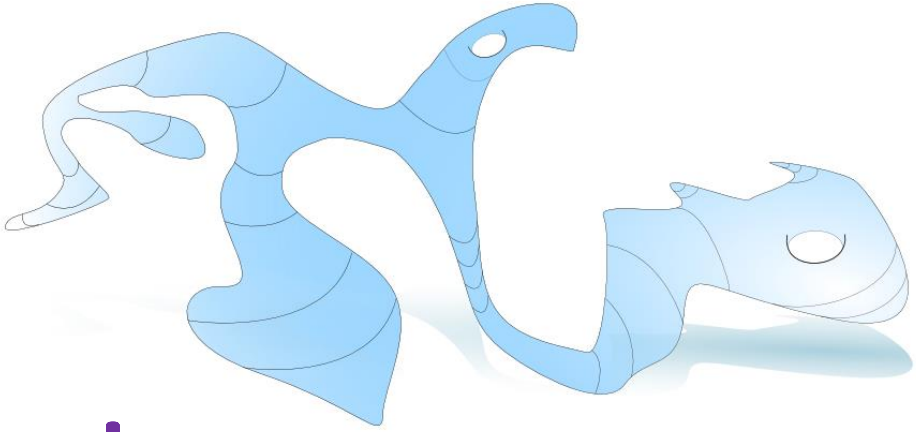
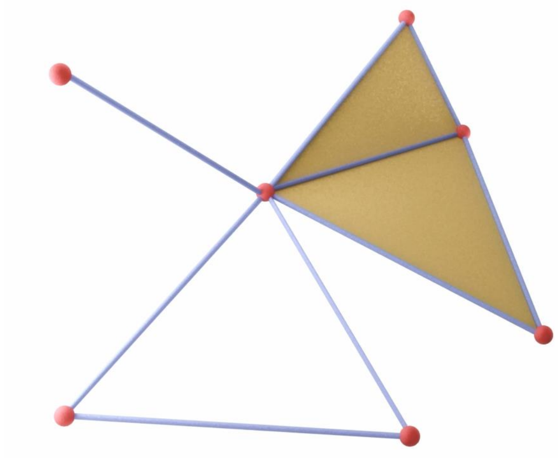




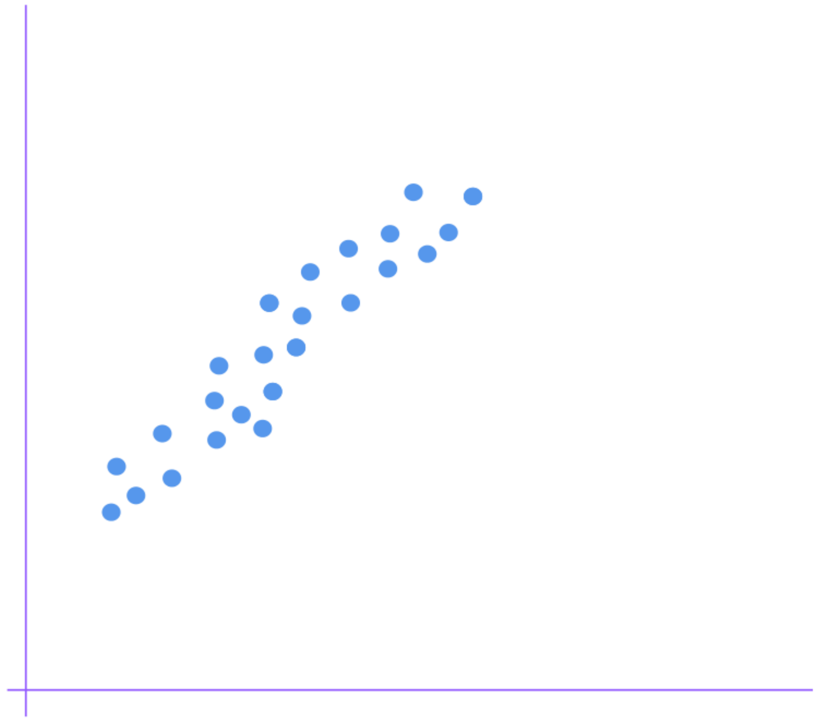
An Introduction to Topological Data Analysis



Mustafa Hajij

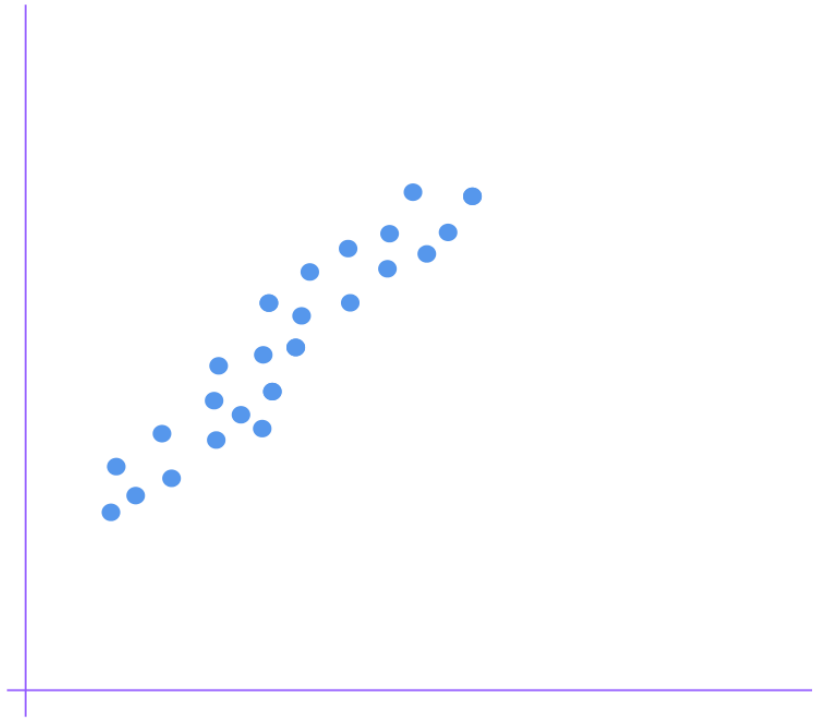
Motivation

The classical problem of fitting data set of point in R n using linear regression



Motivation

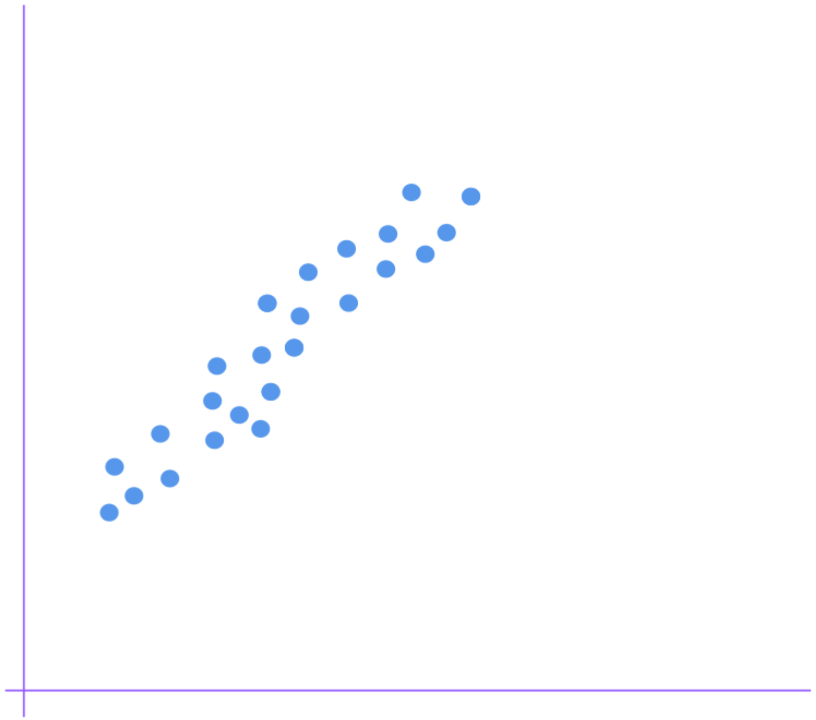
The classical problem of fitting data set of point in R n using linear regression



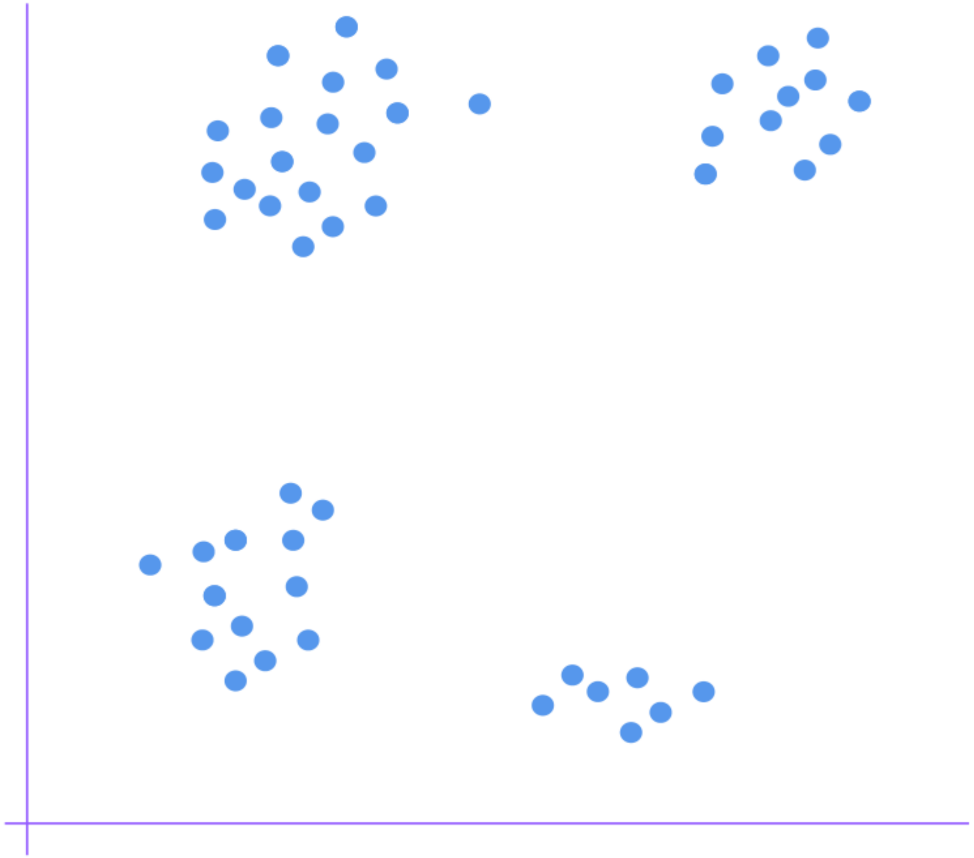
The linear shape of the data is a fundamental assumption underlying the linear regression method

Motivation

The classical problem of fitting data set of point in R n using linear regression

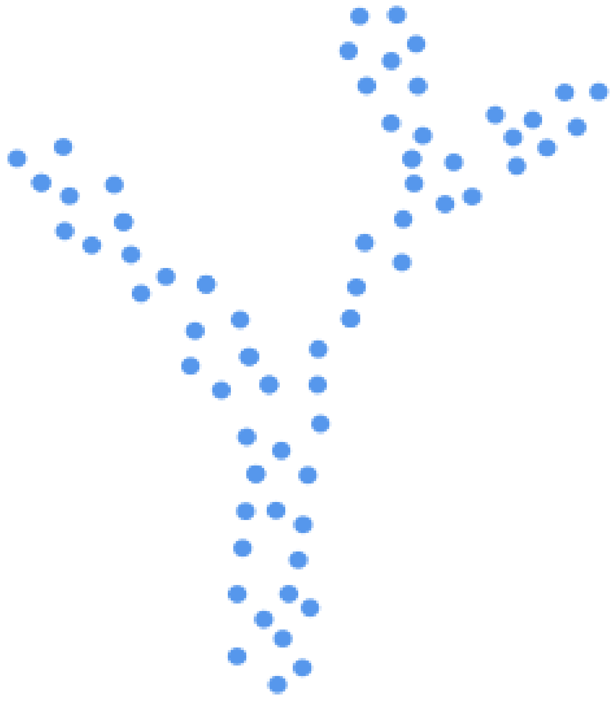


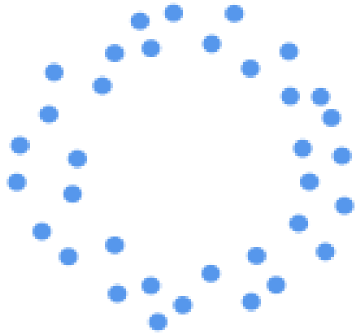
The linear shape of the data is a fundamental assumption underlying the linear regression method



Clustering algorithms assume that the data is clustered in a certain way.

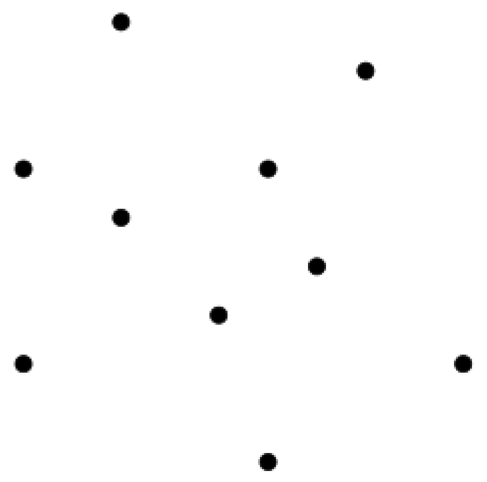
Motivation





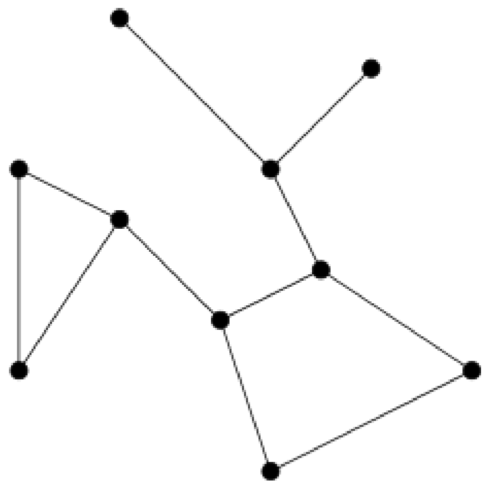
Understanding the shape of the data is a fundamental assumption underlying the analytical method

Concept



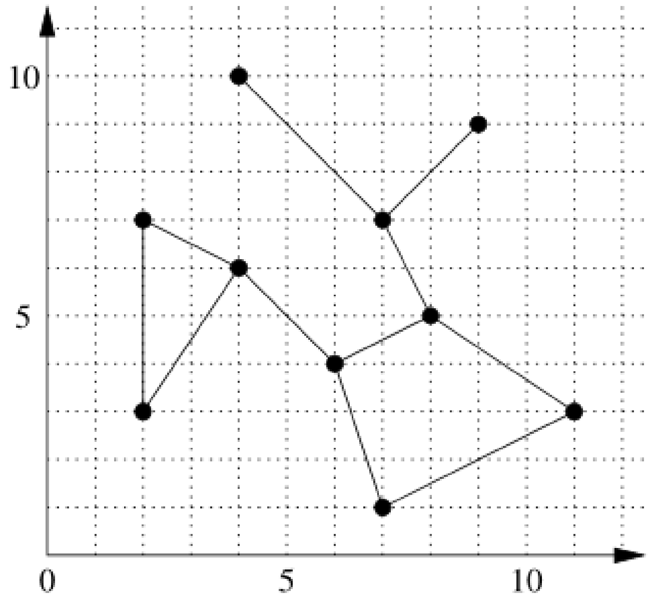
Space

Concept



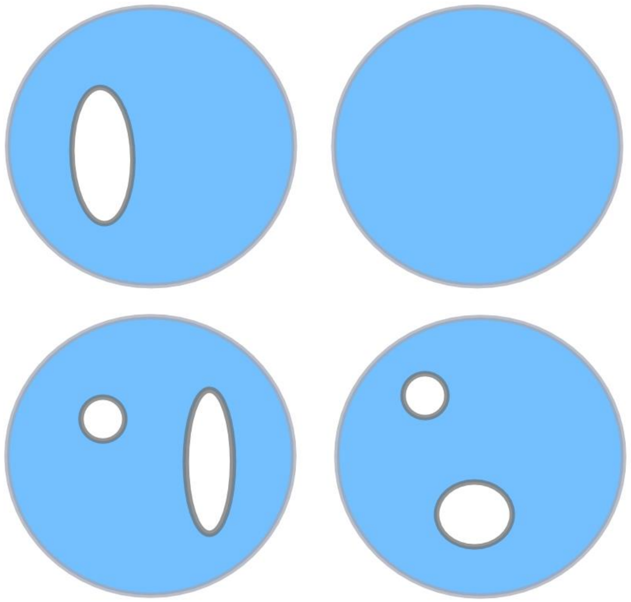
Topological space

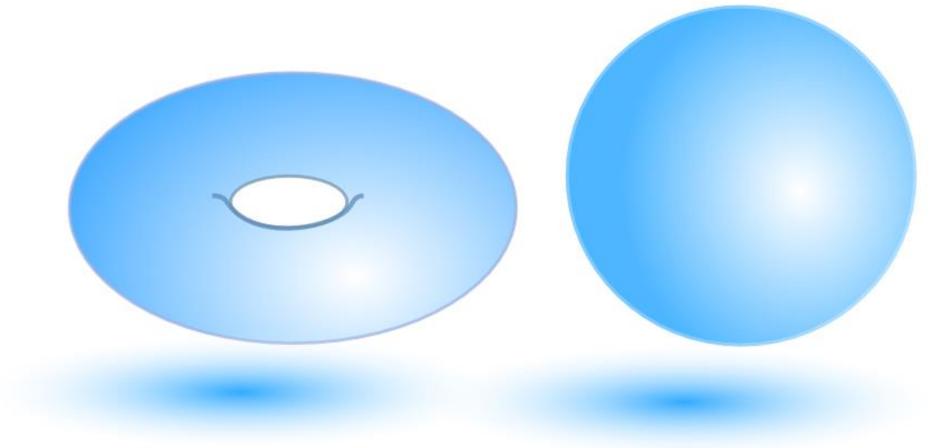
Concept



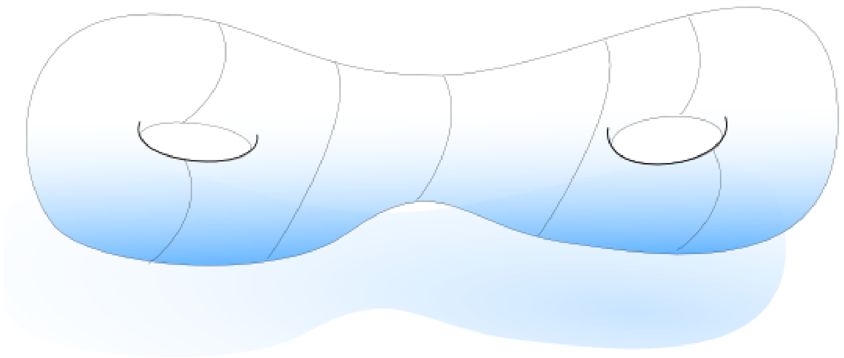
Metric Space

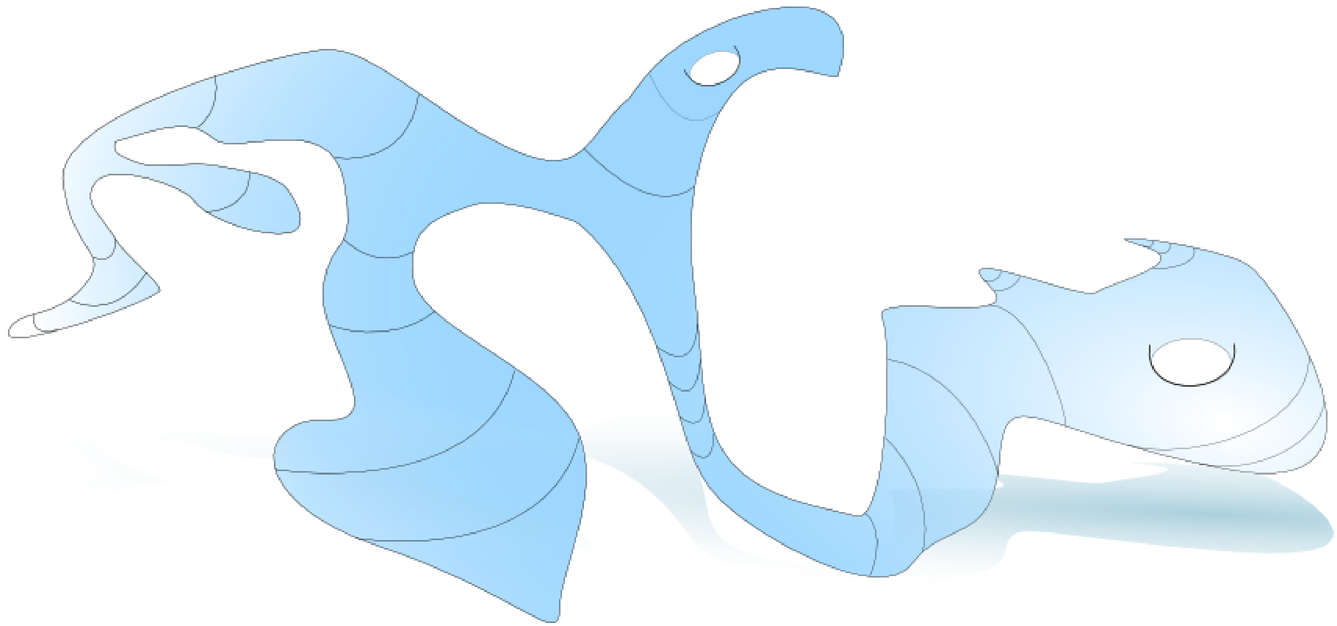
Concept





Concept





Motivation

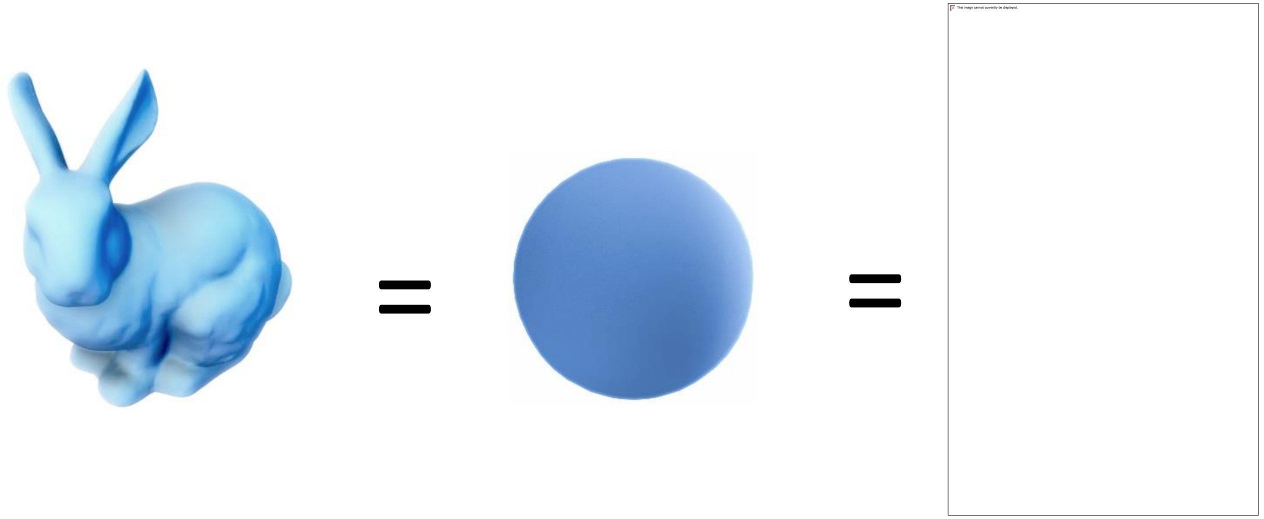
Roughly speaking, topology of an object studies the way this object is connected.

Topology studies the properties of shapes that do not change under continuous deformations.

Motivation

Roughly speaking, topology of an object studies the way this object is connected.

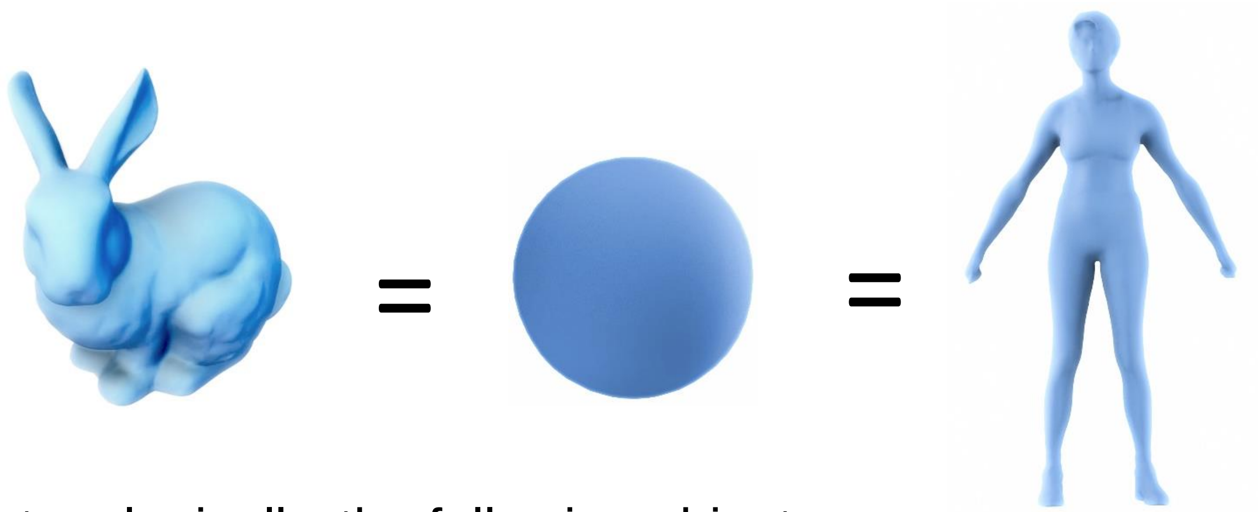
Topology studies the properties of shapes that do not change under continuous deformations.



Motivation

Roughly speaking, topology of an object studies the way this object is connected.

Topology studies the properties of shapes that do not change under continuous deformations.



So topologically, the following objects are equivalent because we can deform each one of them contentiously without tearing into the other.

Motivation

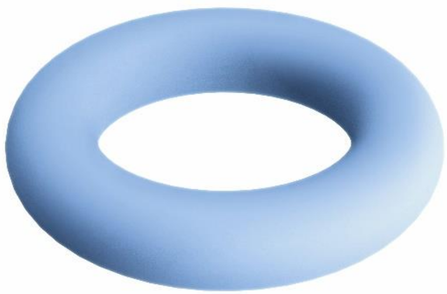
Roughly speaking, topology of an object studies the way this object is connected.

Topology studies the properties of shapes that do not change under continuous deformations.

=

=

=/=



So topologically, the following objects are equivalent because we can deform each one of them contentiously without tearing into the other.

However, the sphere cannot be continuously deformed into the torus. Hence the sphere and the torus are topologically district.

Motivation

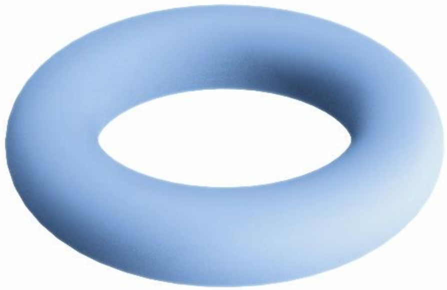
Roughly speaking, topology of an object studies the way this object is connected.

Topology studies the properties of shapes that do not change under continuous deformations.

=

=

=/=



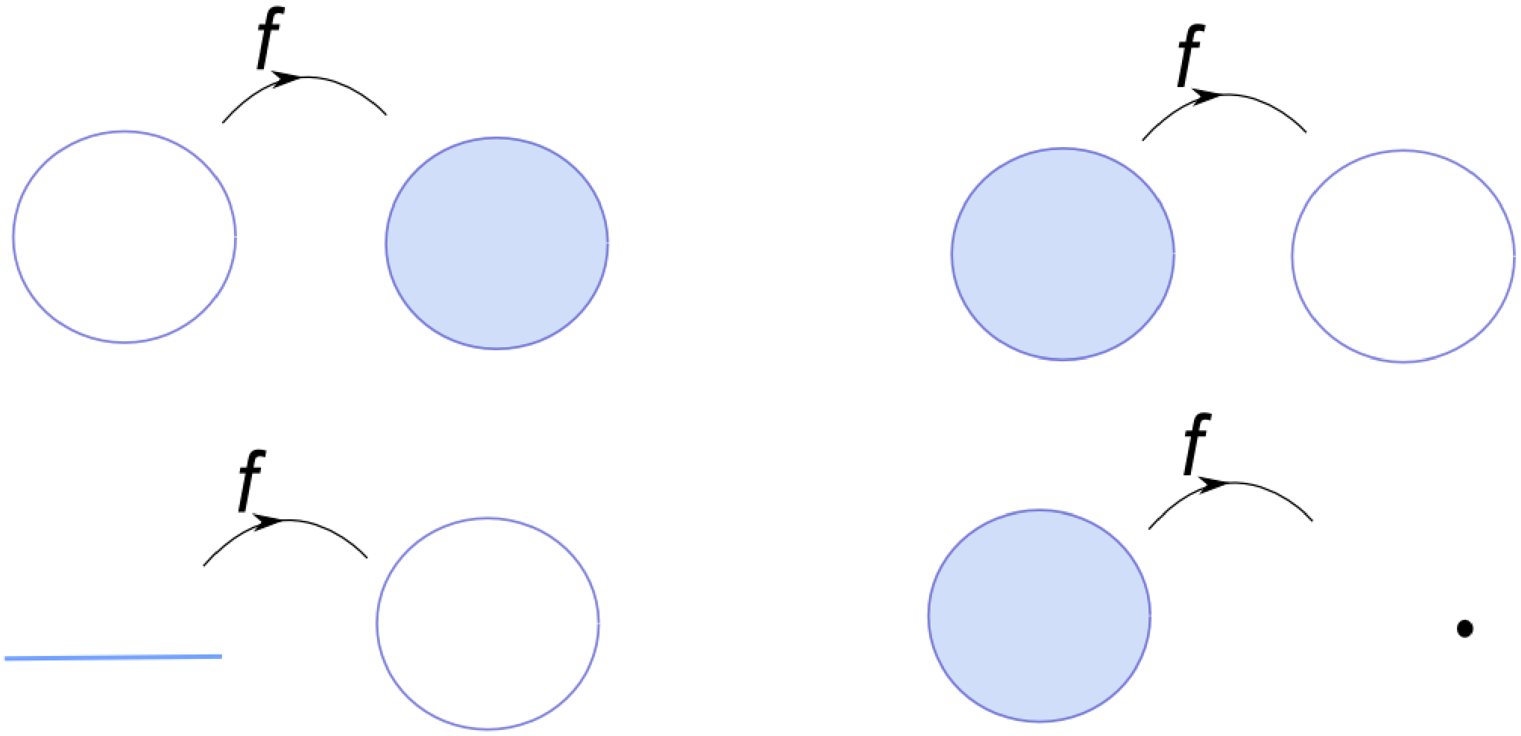
So topologically, the following objects are equivalent because we can deform each one of them contentiously without tearing into the other.

However, the sphere cannot be continuously deformed into the torus. Hence the sphere and the torus are topologically district.

Topology makes these notions precise.

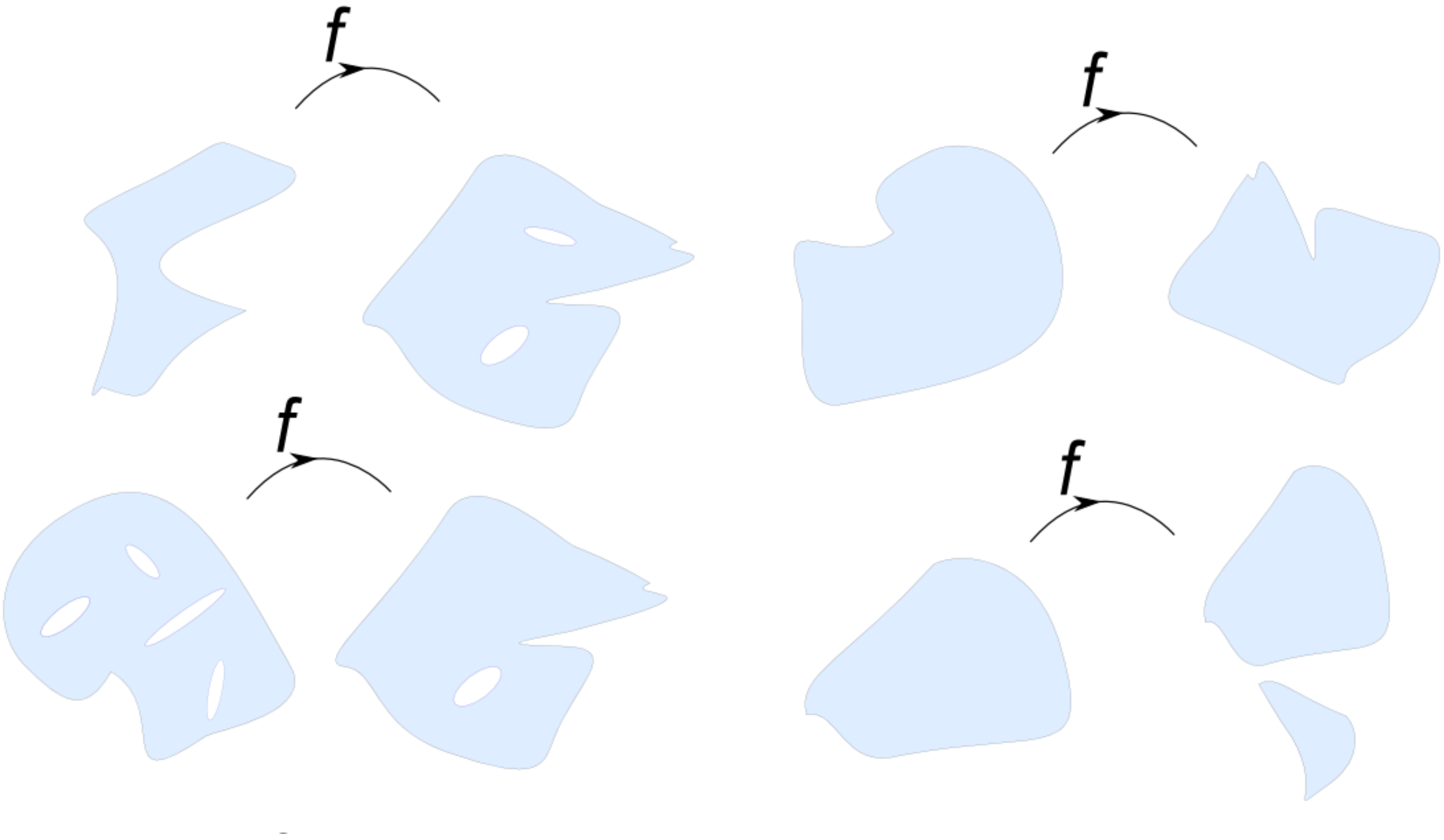
Continuous functions

Which of the following functions is continuous ?

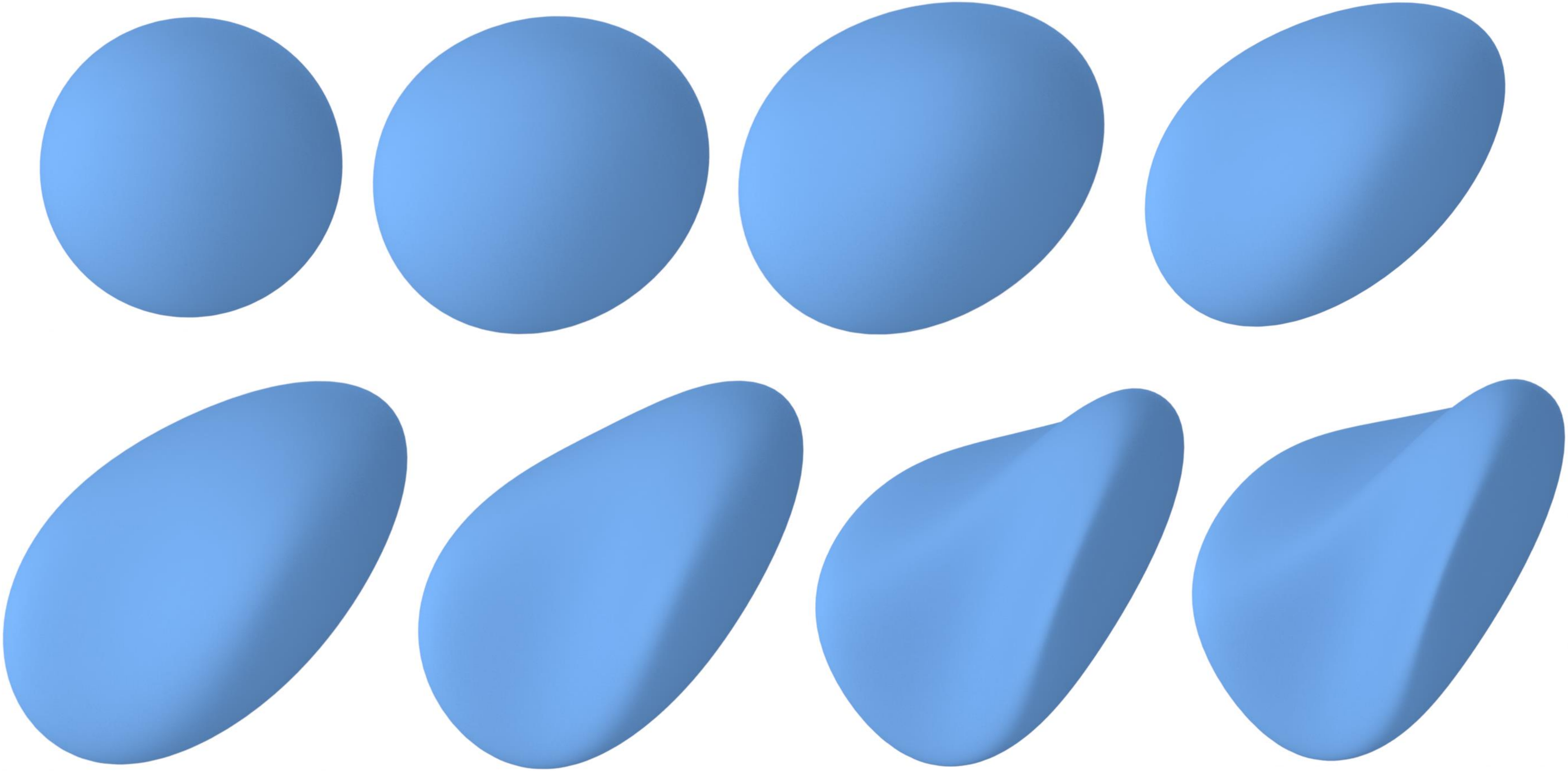


Continuous functions

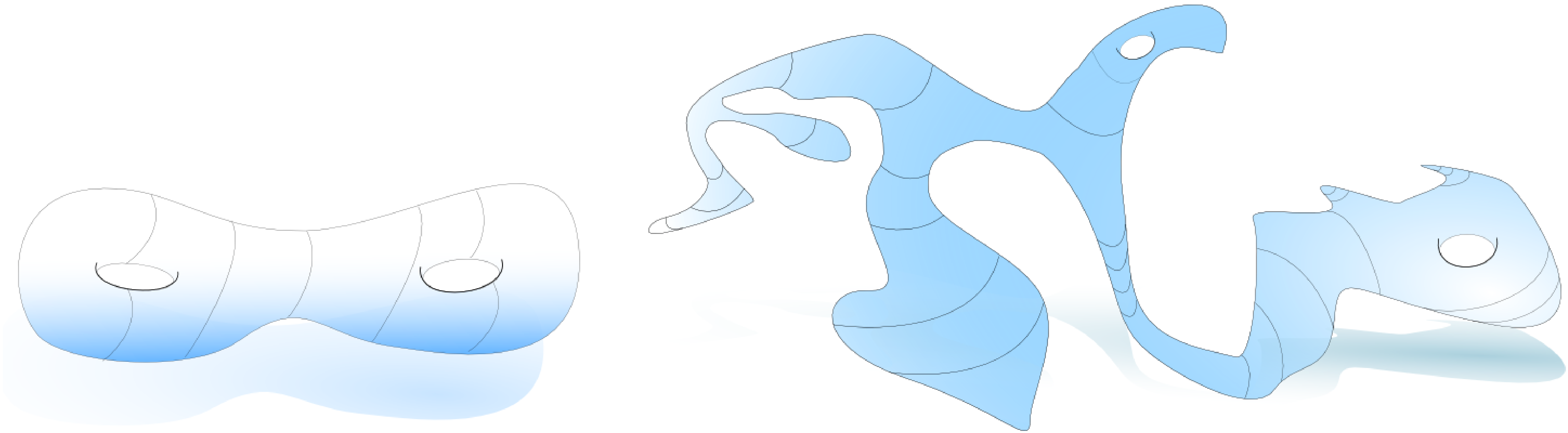
Which of the following functions is continuous ?



Homeomorphism

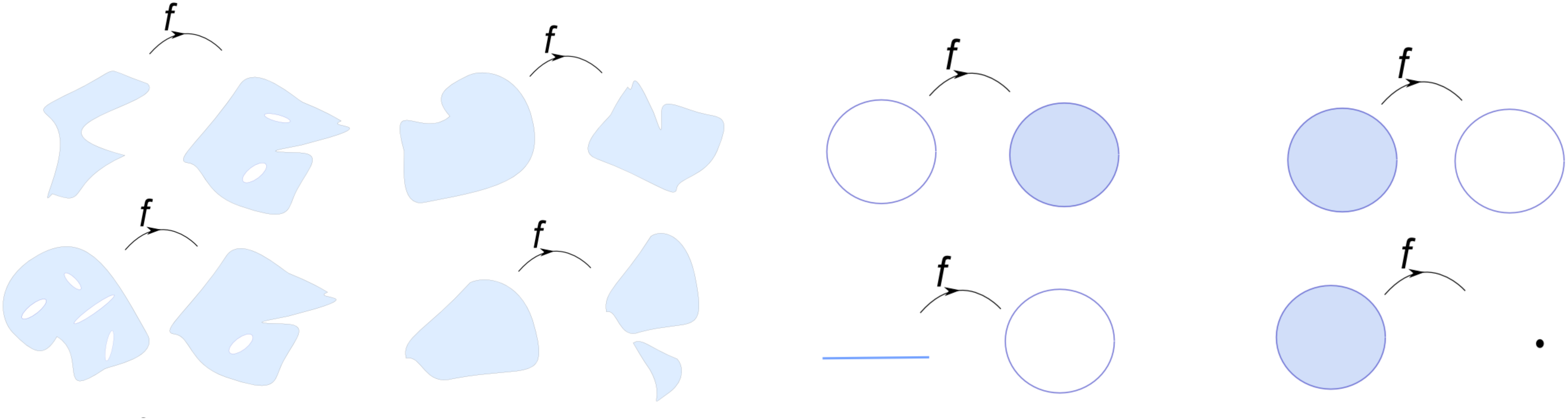


Homeomorphism

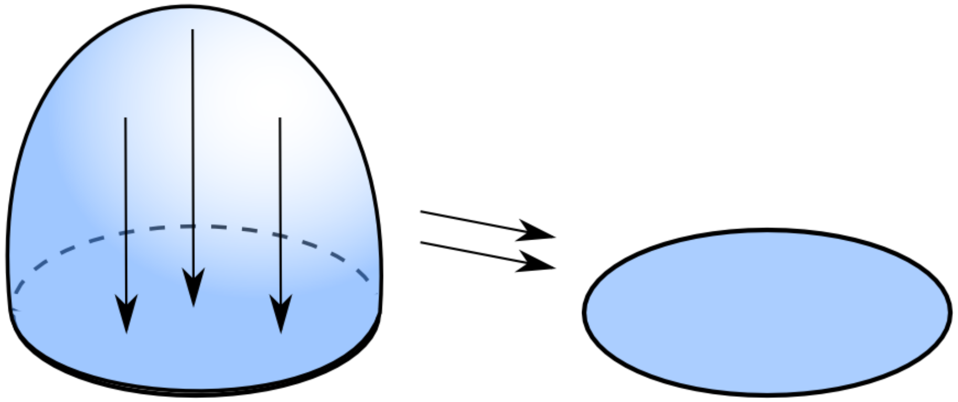


Homeomorphism

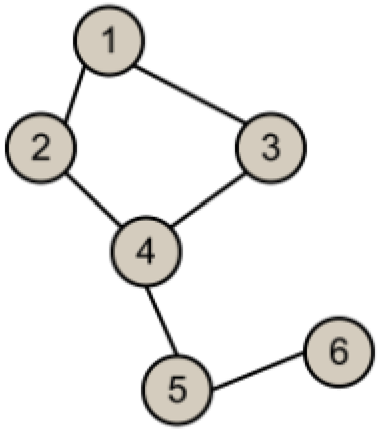
Which of the following functions is homeomorphism? If not, which one of the three conditions is violated?



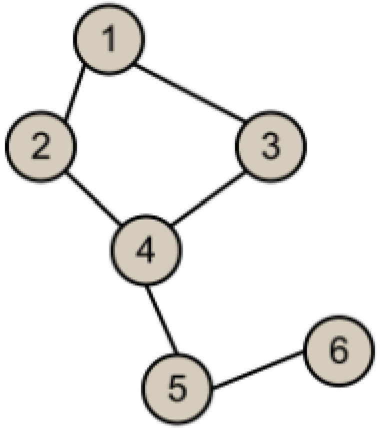
Homeomorphism-examples

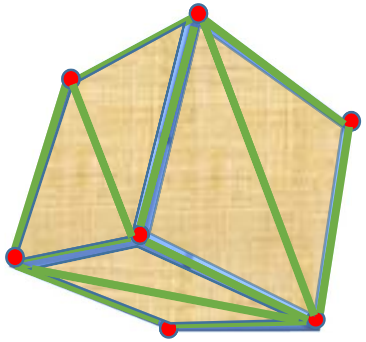


How can we explain a topological space to a computer ?

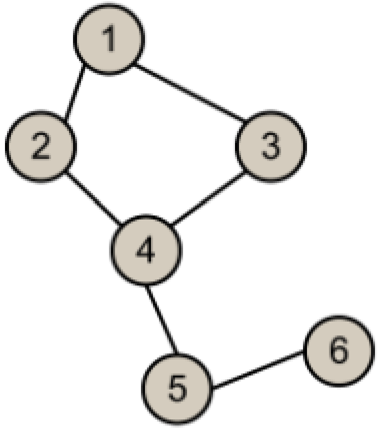


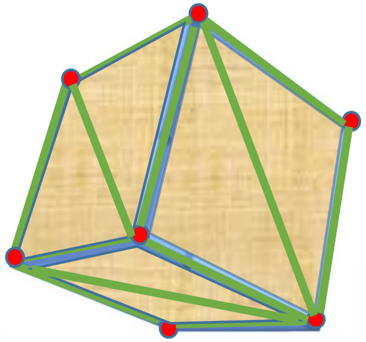
How can we explain a topological space to a computer ?

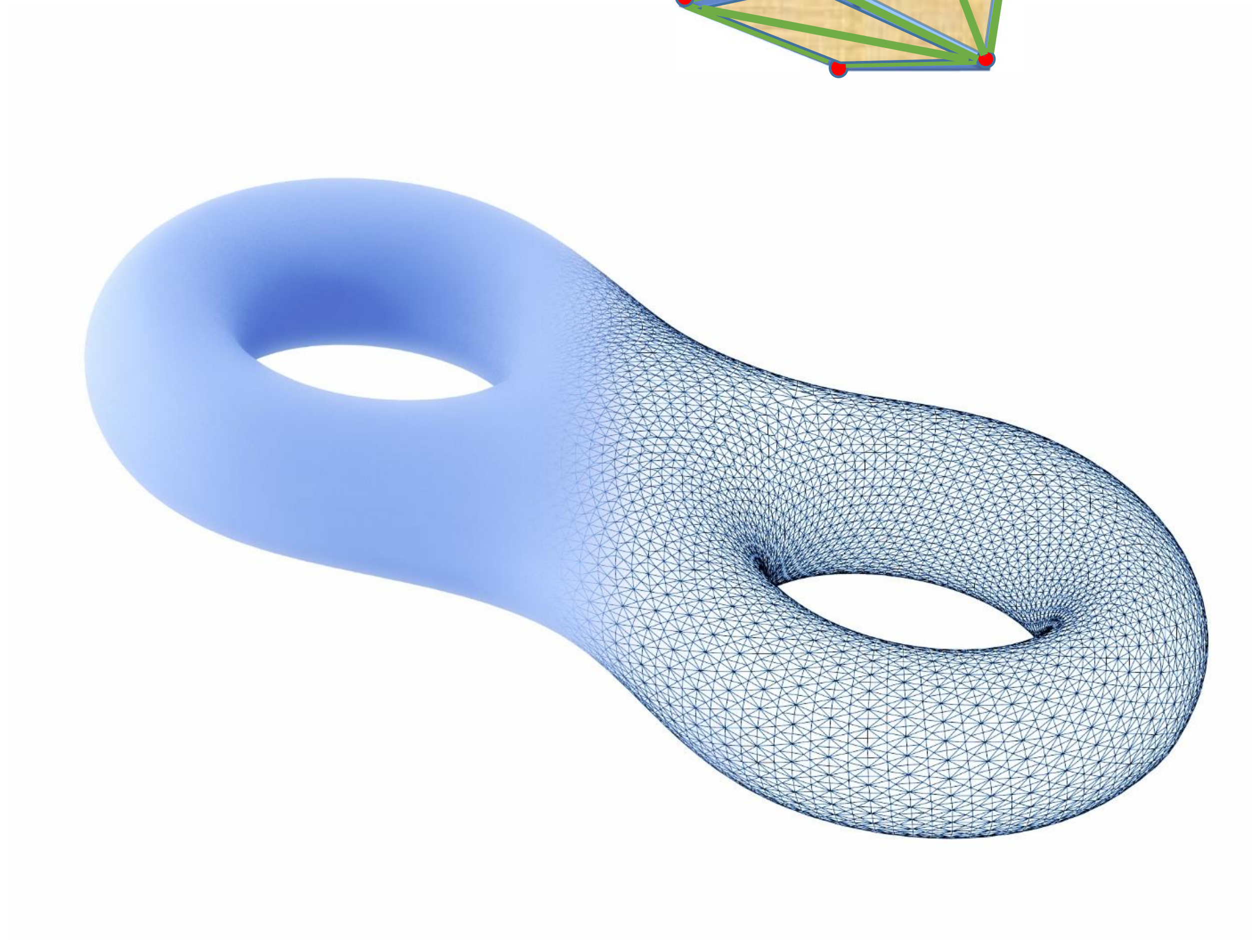




How can we explain a topological space to a computer ?

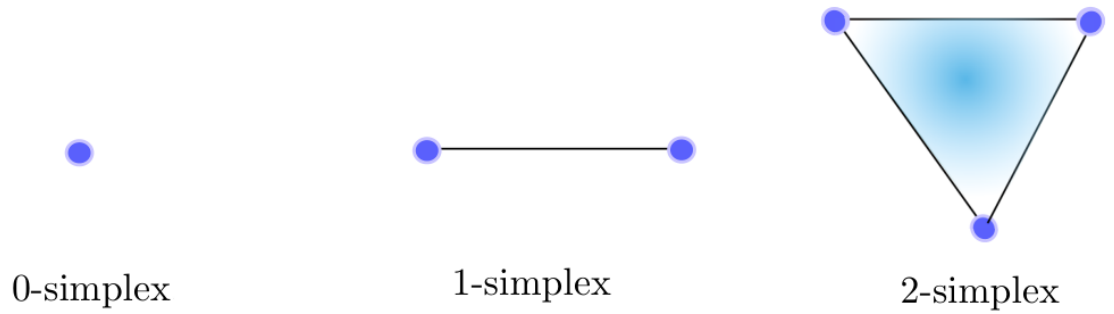






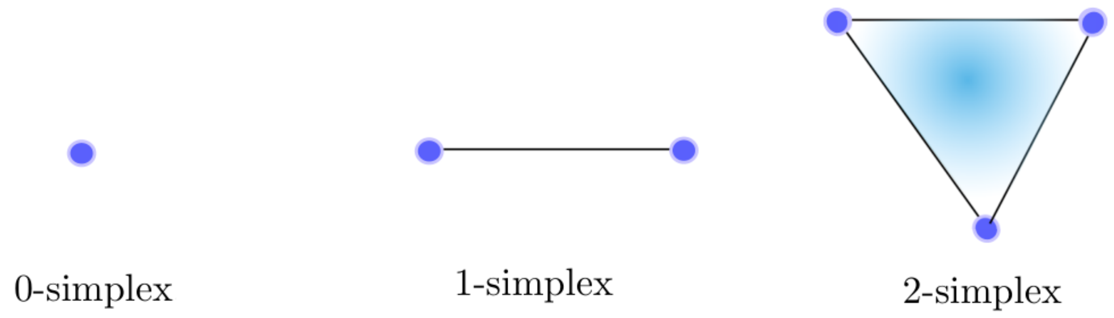
How can we explain a topological space to a computer ?

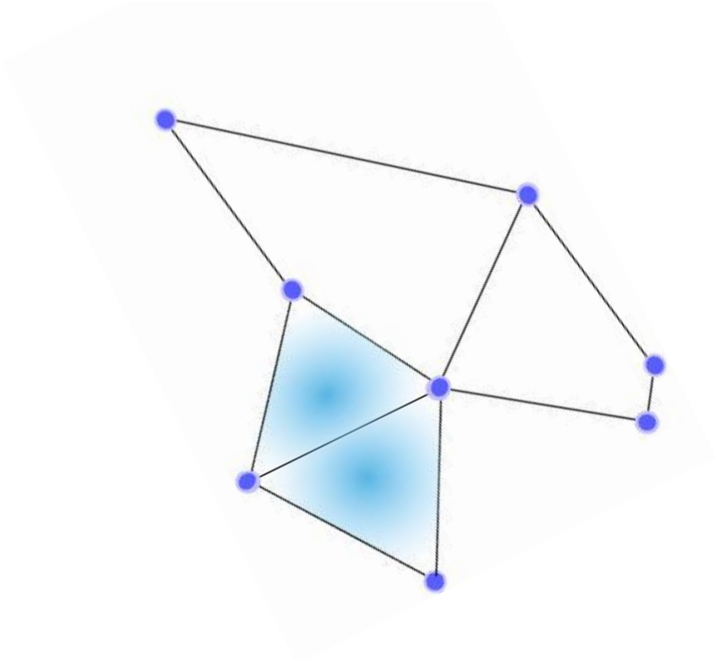
Key Idea : we use simple building blocks (called simplices)



How can we explain a topological space to a computer ?

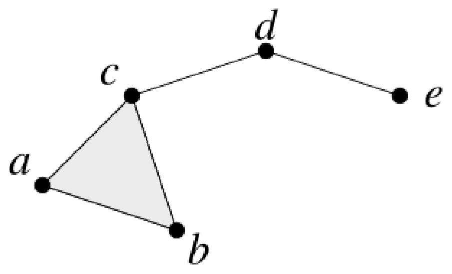
Key Idea : we use simple building blocks (called simplices)



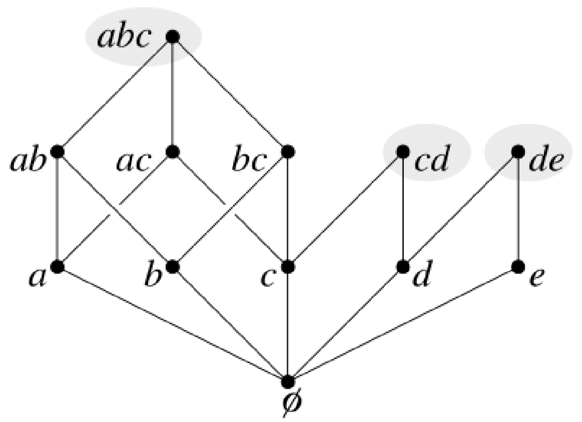


to build more complicated shape.

How can we explain a topological space to a computer ?

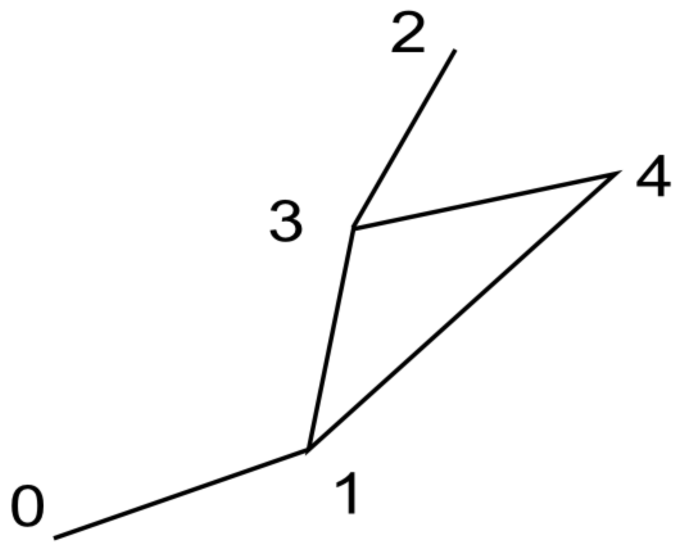


(a) A small complex



(b) Poset of the small complex, with principal simplices marked

Graphs representation: list of edges



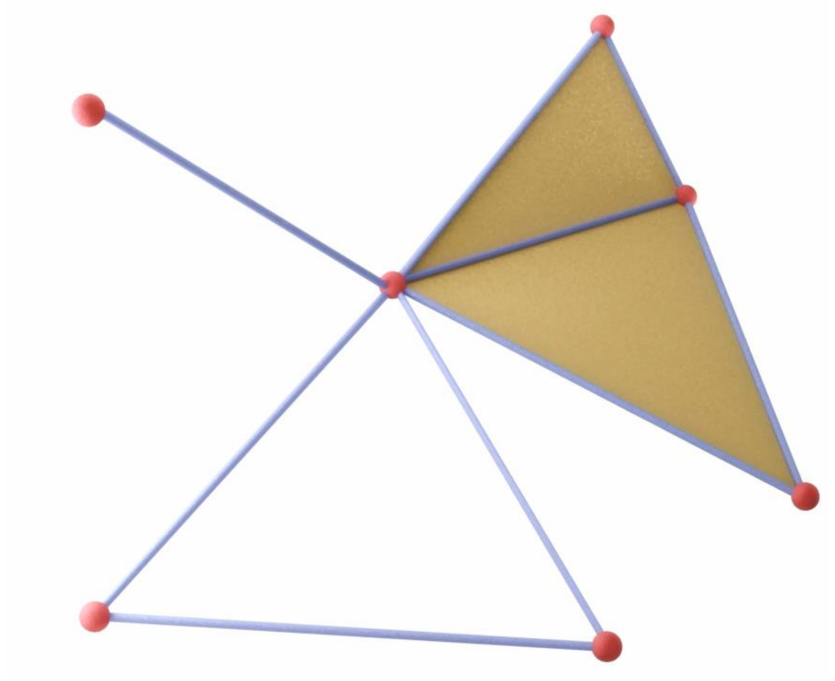
[[0,1], [1,3],[1,4] [3,4], [3,2]]

- *The vertices can be recovered from the edges.*
- *The order of the vertices is important only if the graph is directed.*

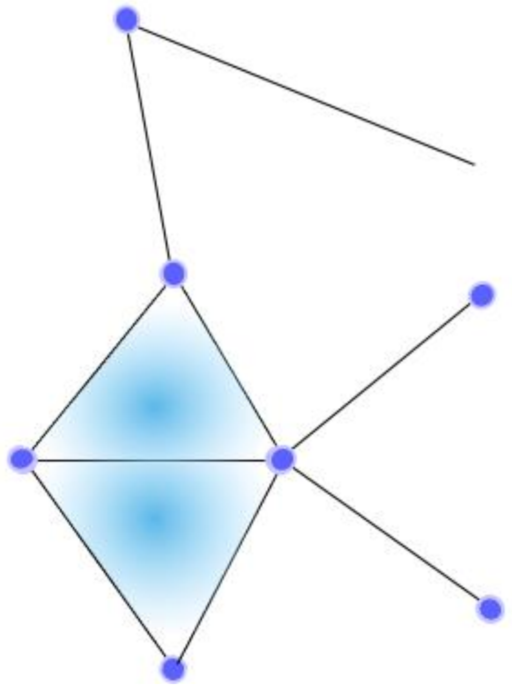
Simplicial Complex-precise definition

Definition A simplical complex is a finite collection of simplices Σ that satisfies the following conditions:

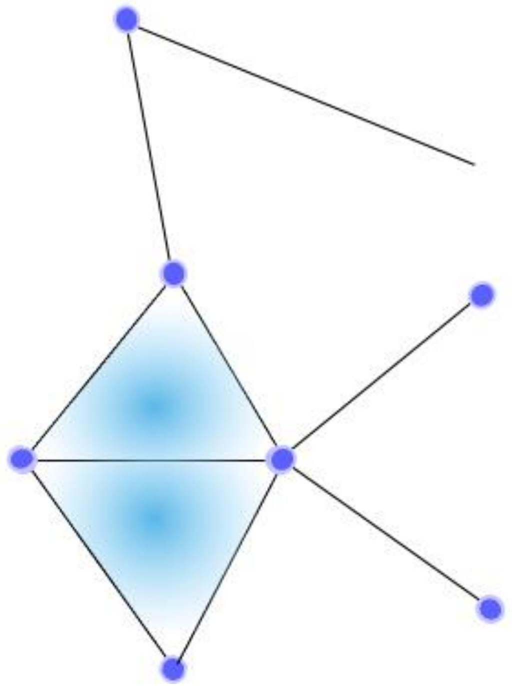
(1) If σ is in Σ then all the facets of σ are also in Σ.
(2) If $\sigma_1, \sigma_2 \in \Sigma$, $\sigma_1 \cap \sigma_2 \neq \phi$, then $\sigma_1 \cap \sigma_2$ is in Σ.

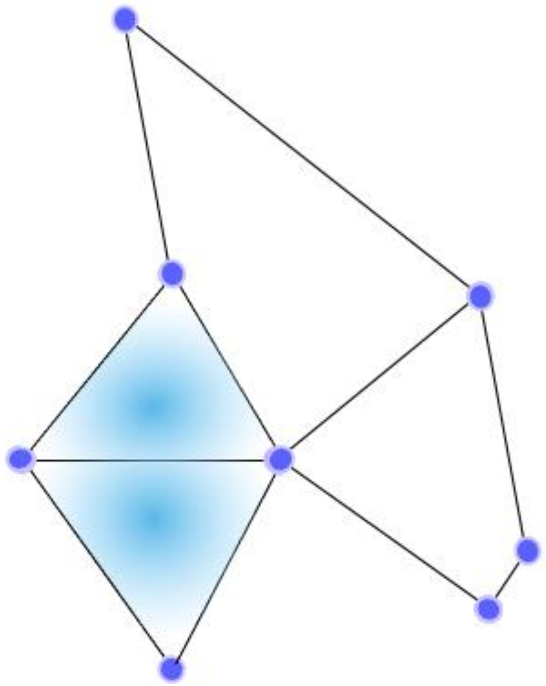


Examples and non-examples

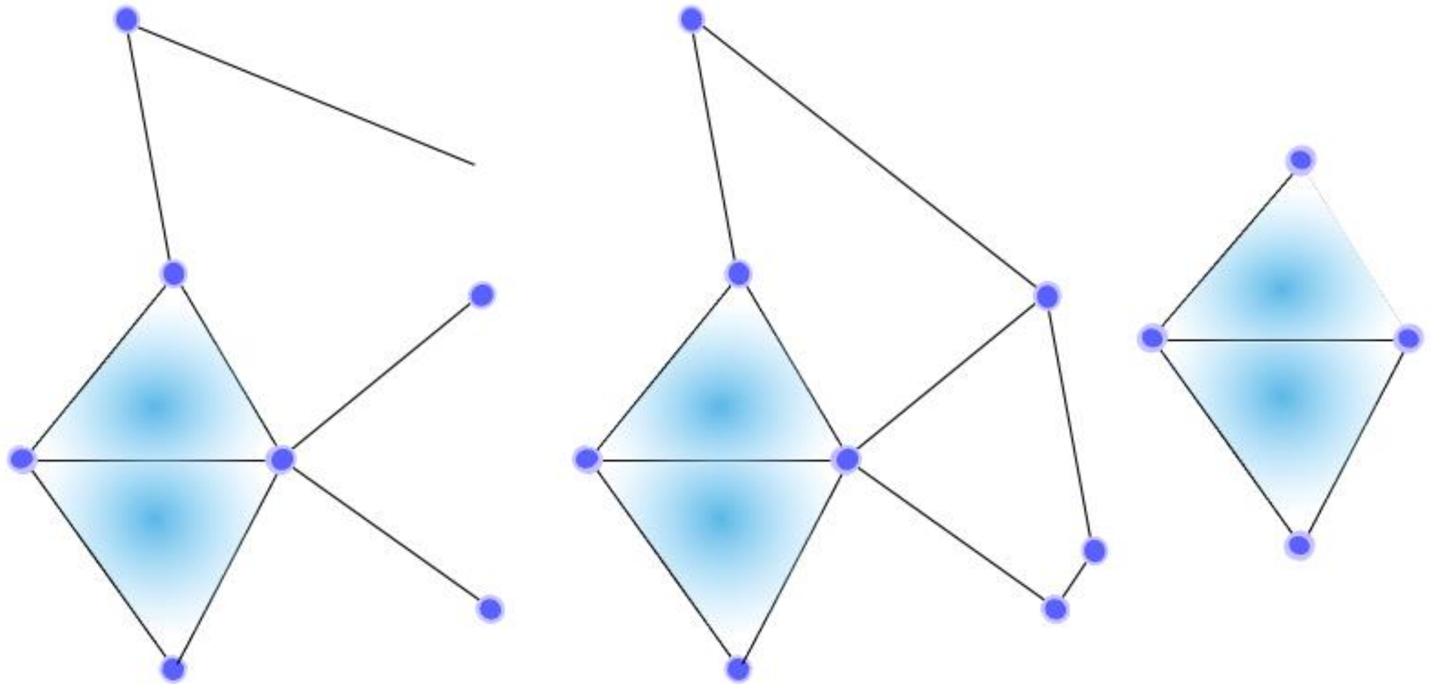


Examples and non-examples

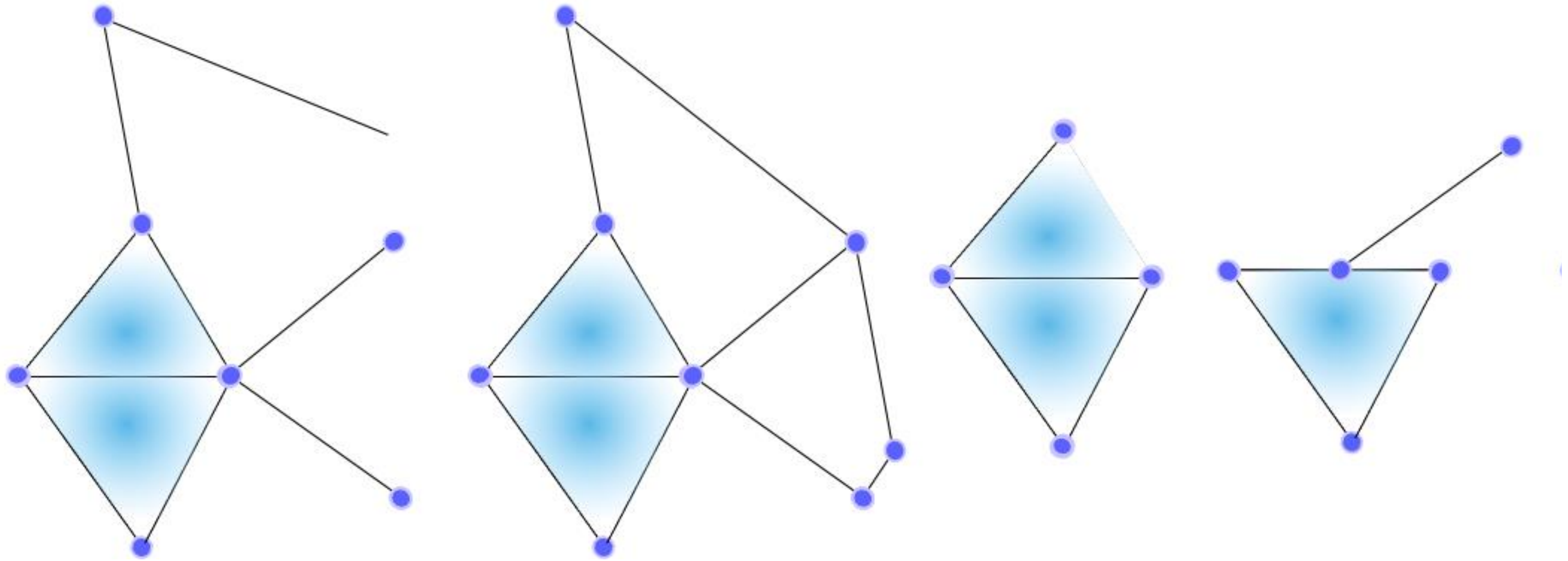




Examples and non-examples



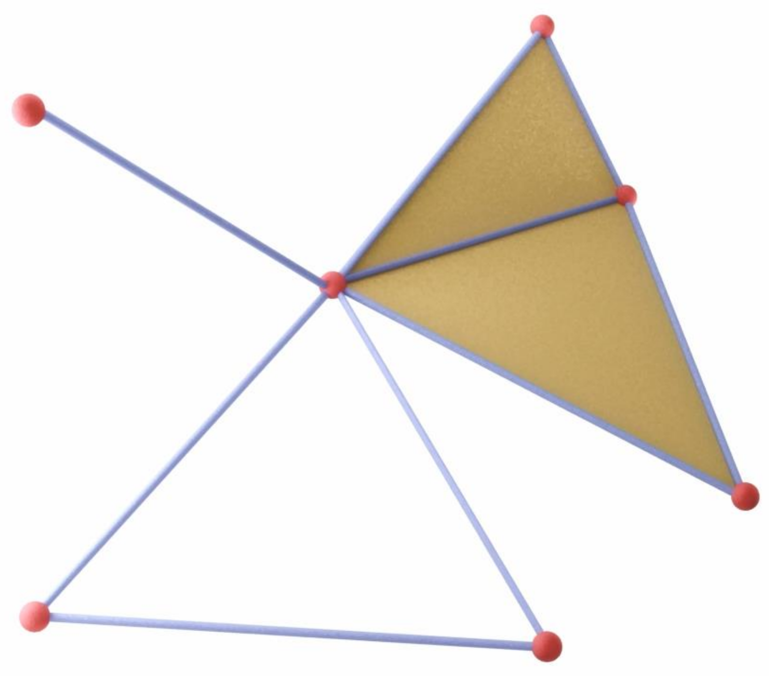
Examples and non-examples



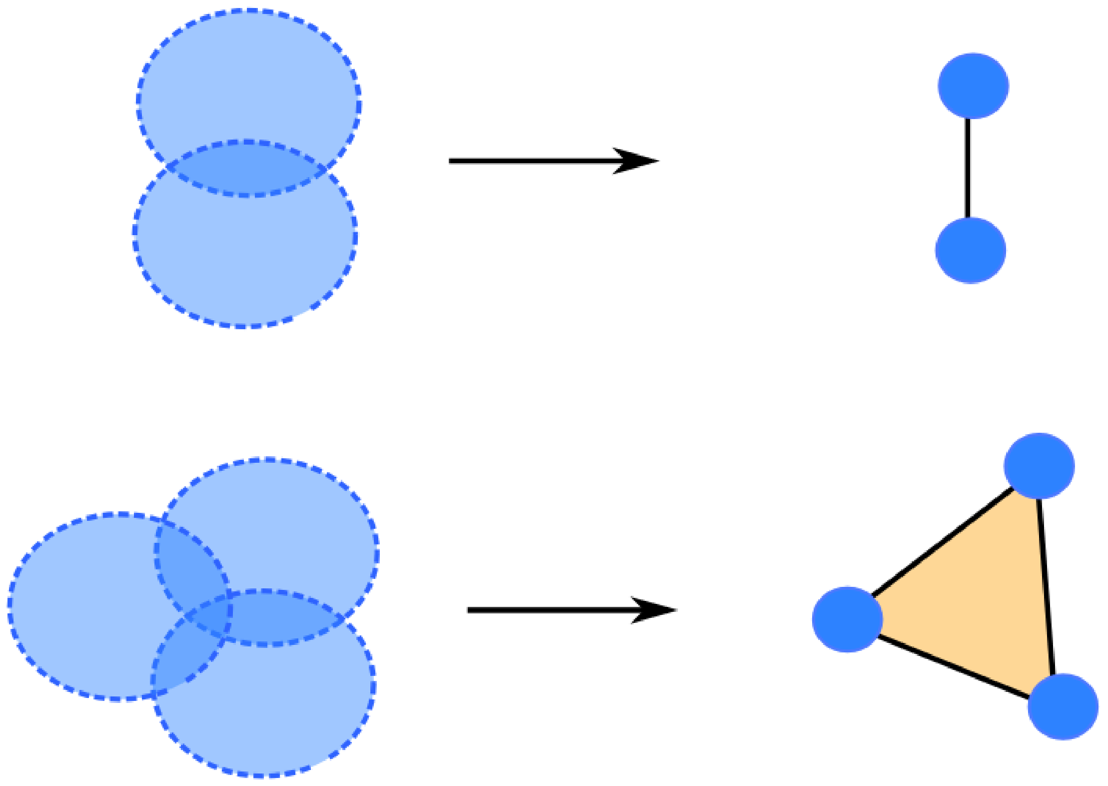
Simplicial Complex

Definition A simplical complex is a finite collection of simplices Σ that satisfies the following conditions:

(1) If σ is in Σ then all the facets of σ are also in Σ.
(2) If $\sigma_1, \sigma_2 \in \Sigma$, $\sigma_1 \cap \sigma_2 \neq \phi$, then $\sigma_1 \cap \sigma_2$ is in Σ.

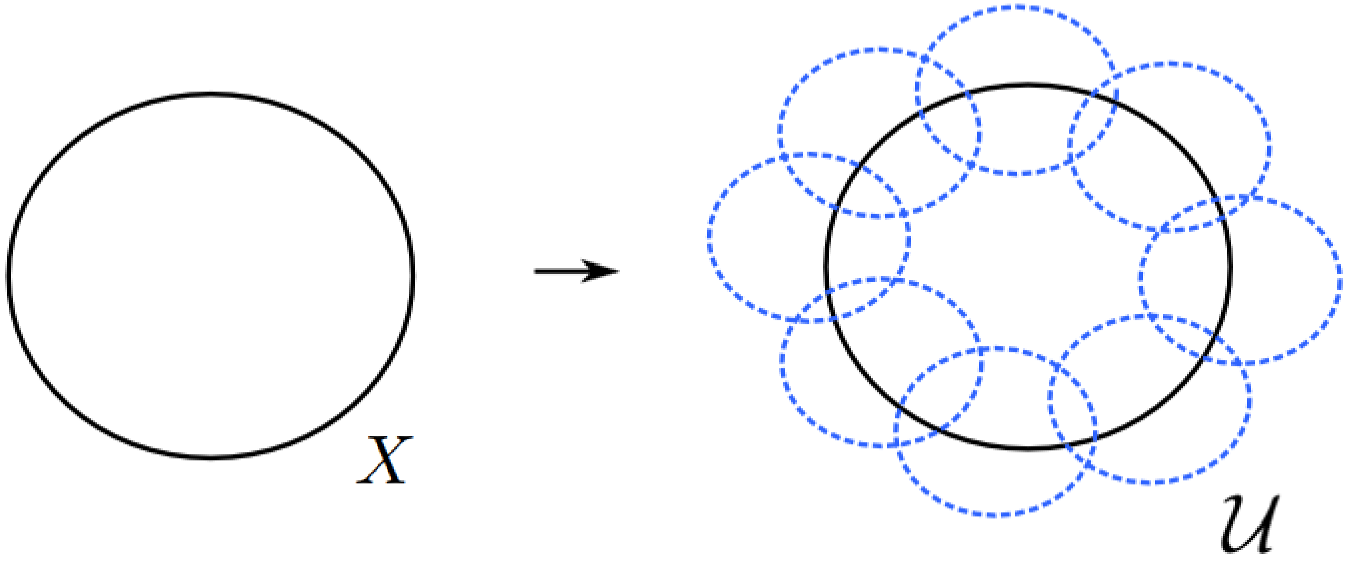


Approximation of the shape



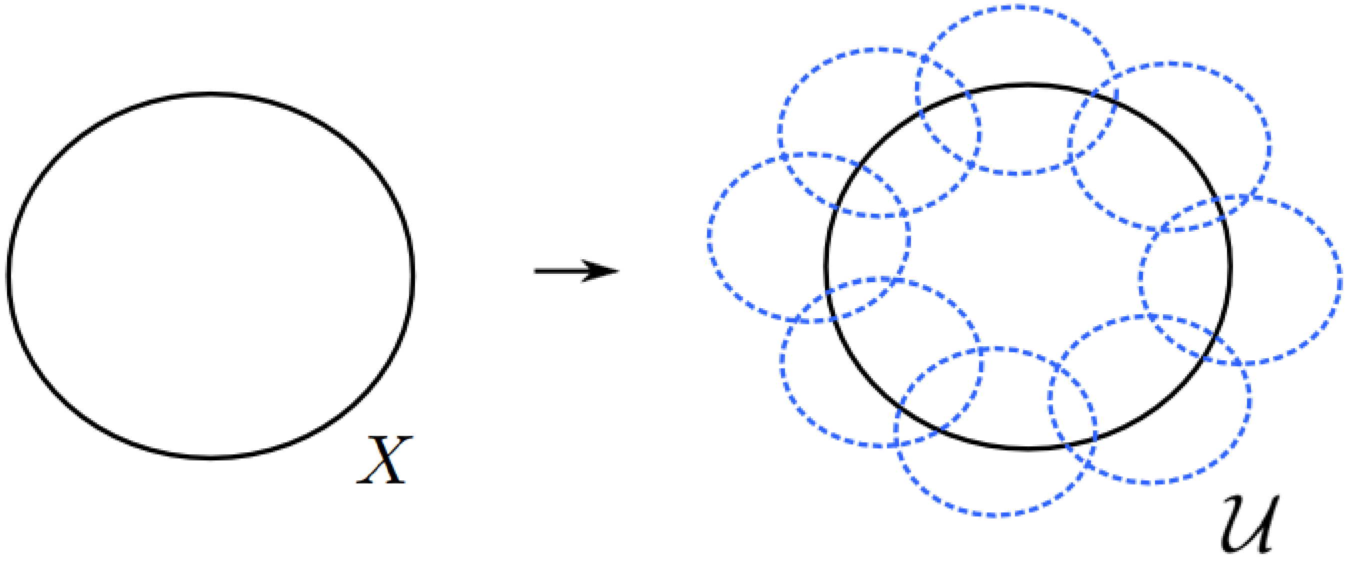
Cover of a space

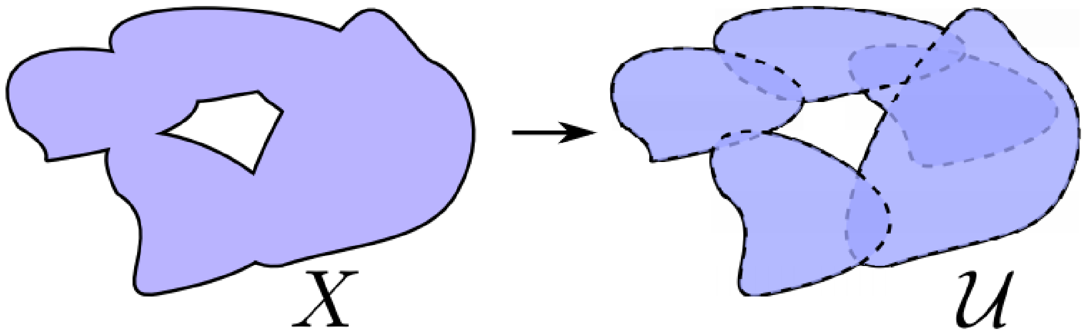
A cover of a space X is a collection of sets U whose union is the entire space



Cover of a space

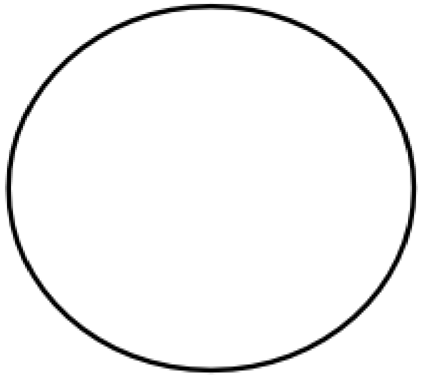
A cover of a space X is a collection of sets U whose union is the entire space





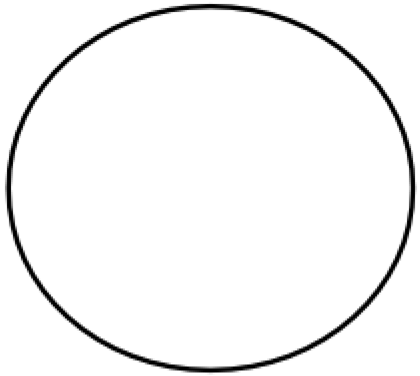
Approximation of the shape

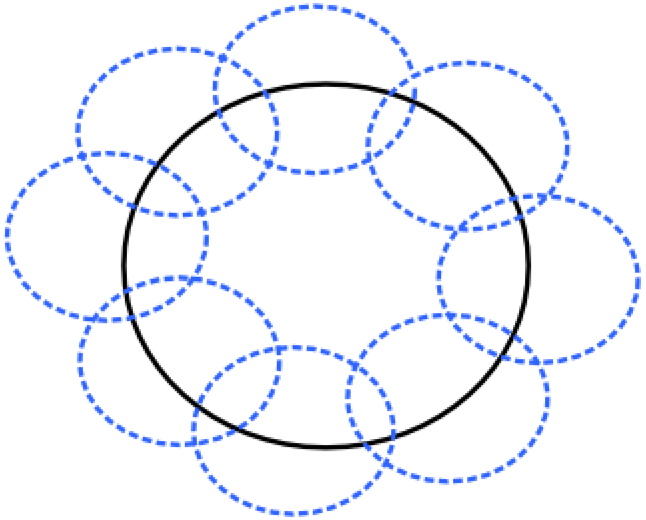
Covers can be used to obtain an approximation for the underlying data



Approximation of the shape

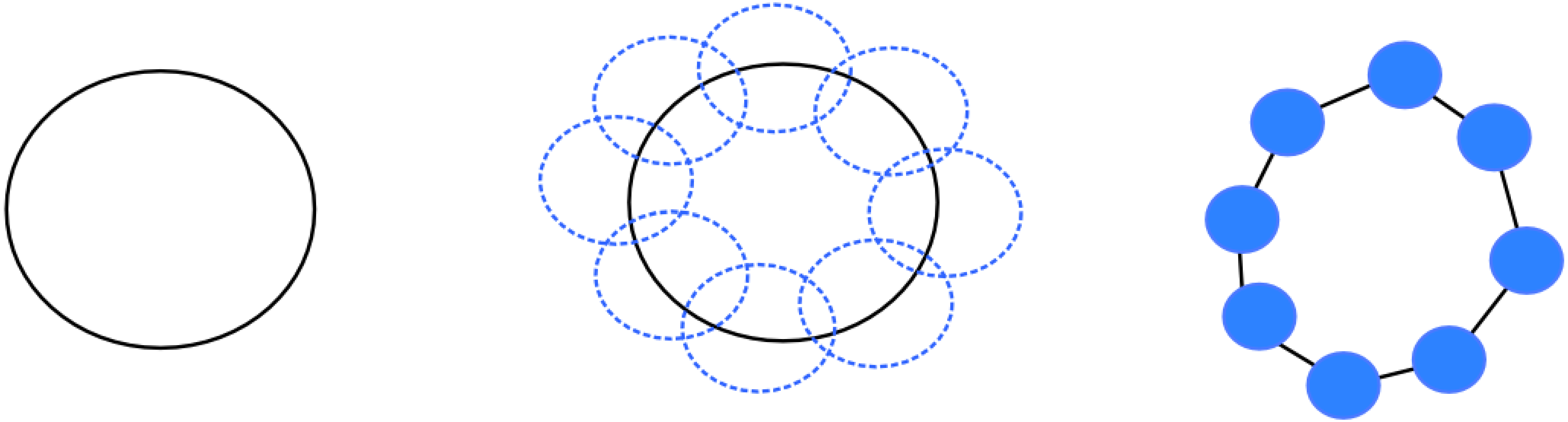
Covers can be used to obtain an approximation for the underlying data





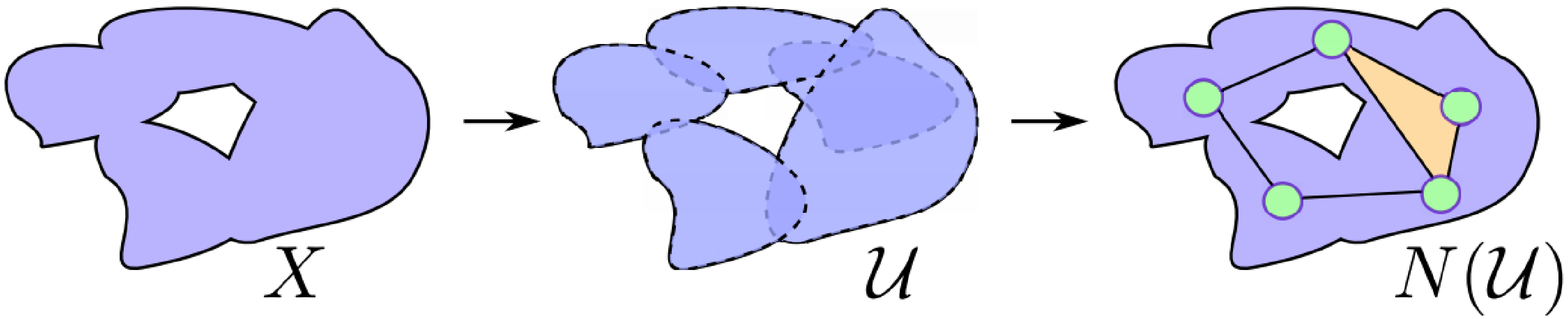
Approximation of the shape

Covers can be used to obtain an approximation for the underlying data



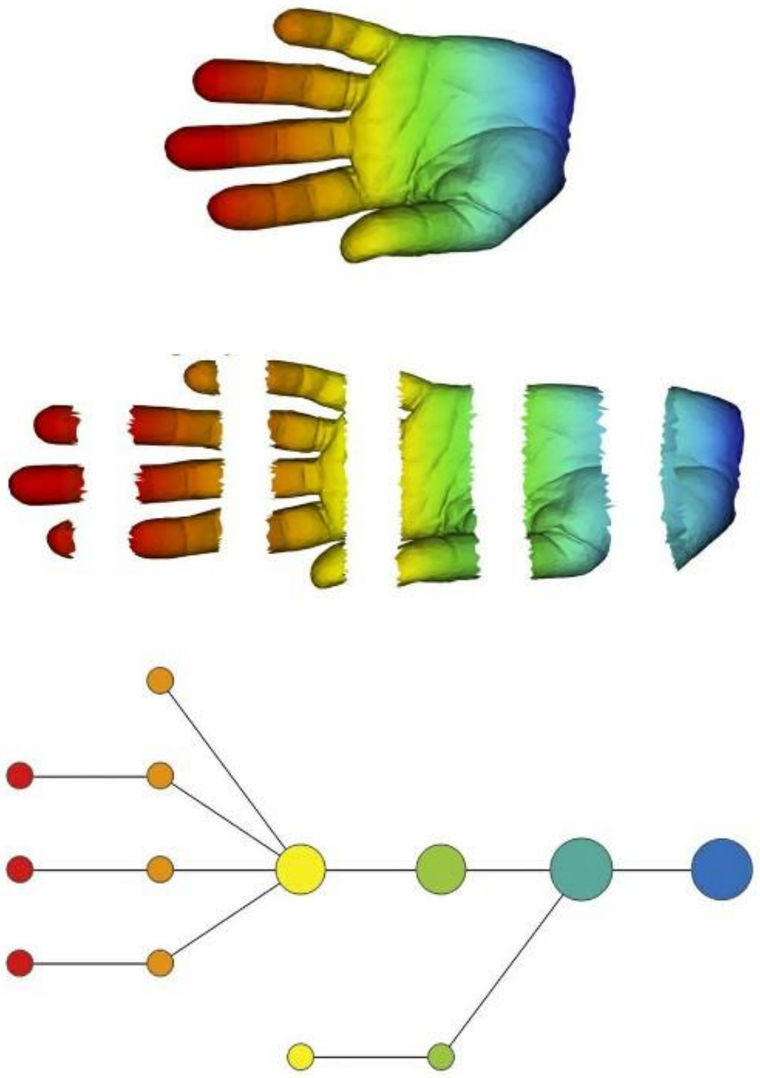
Nerve of a space

Approximation of the shape



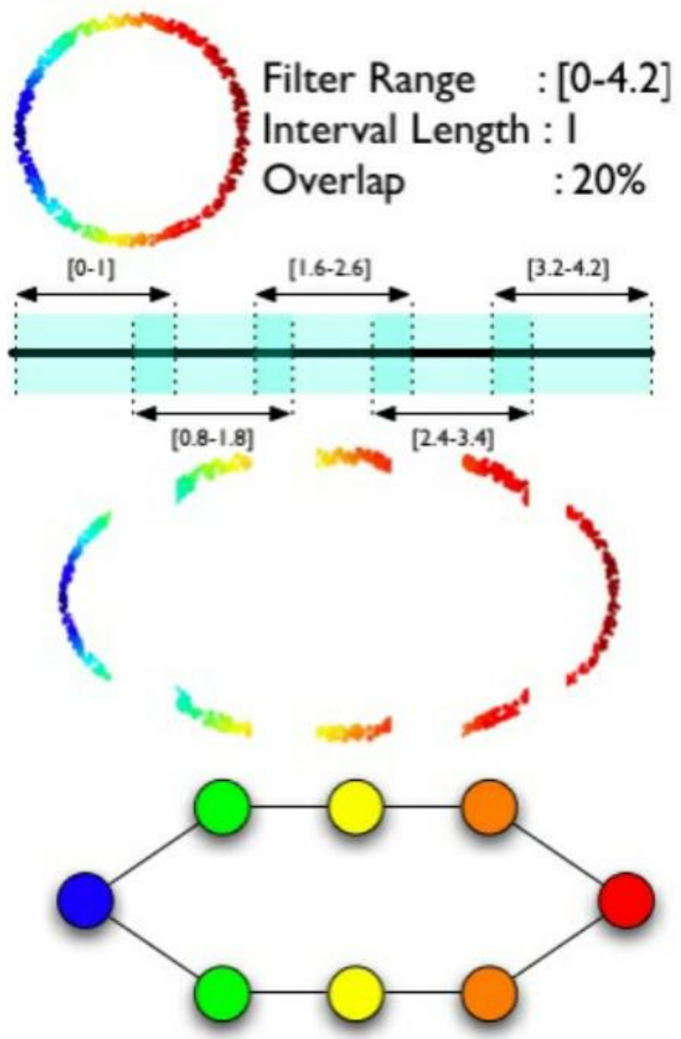
Key idea: every set is replaced by a node
every intersection is replaced by an edge
if we have intersection between three sets we replace them with a face and so on.

Mapper Construction



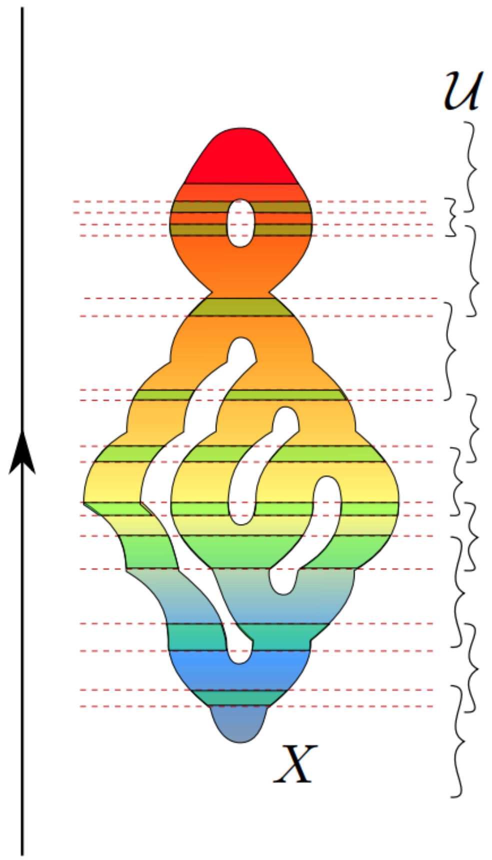
source

Mapper Construction

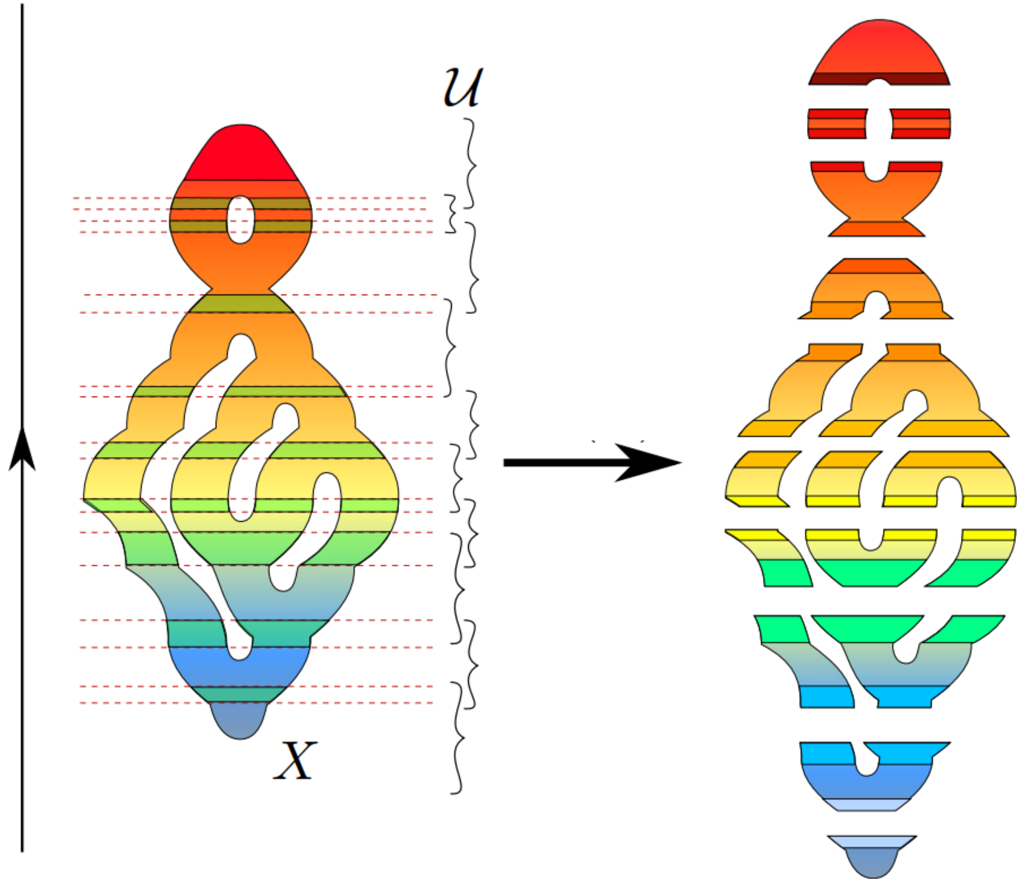


source

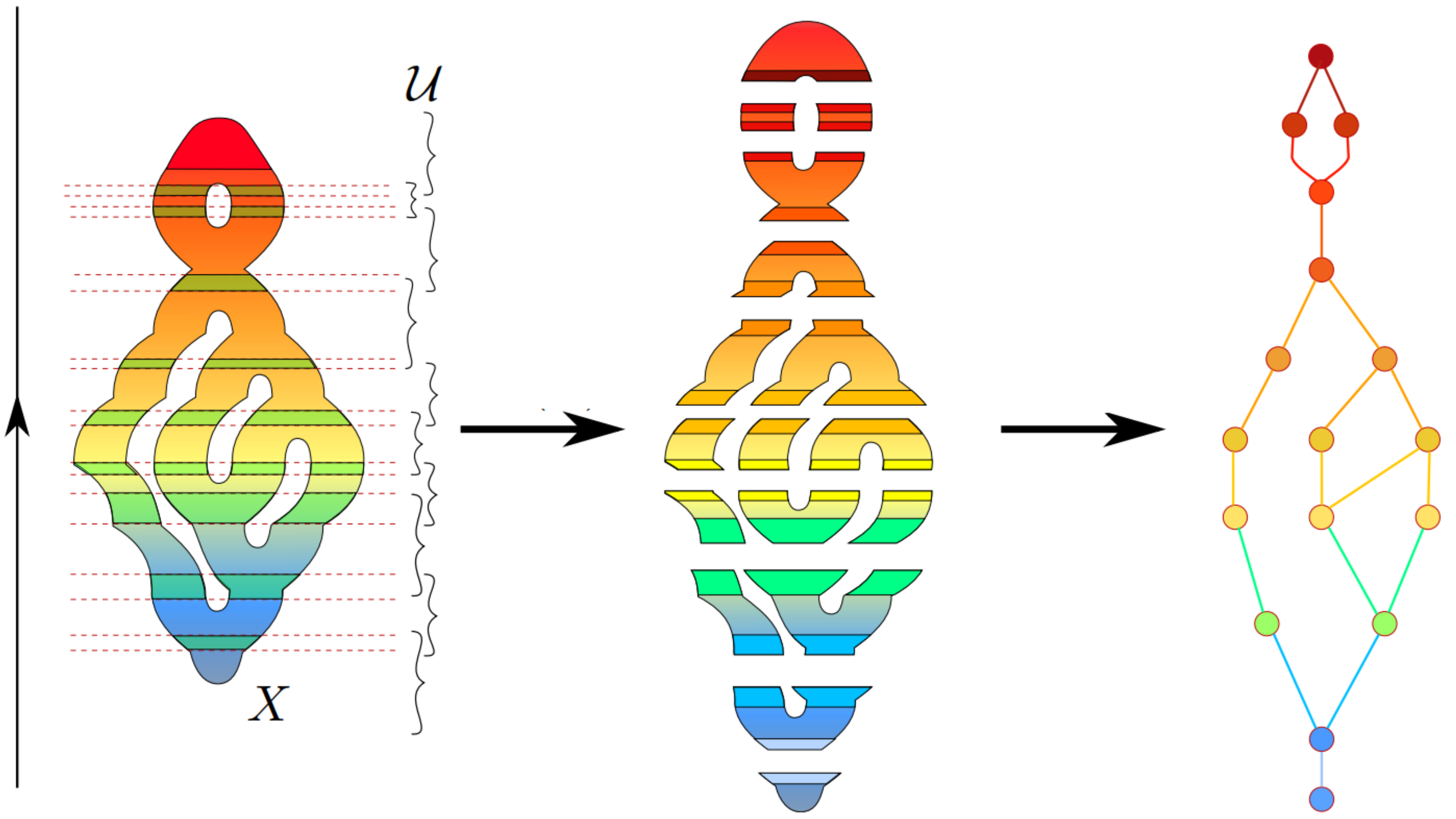
Mapper Construction



Mapper Construction

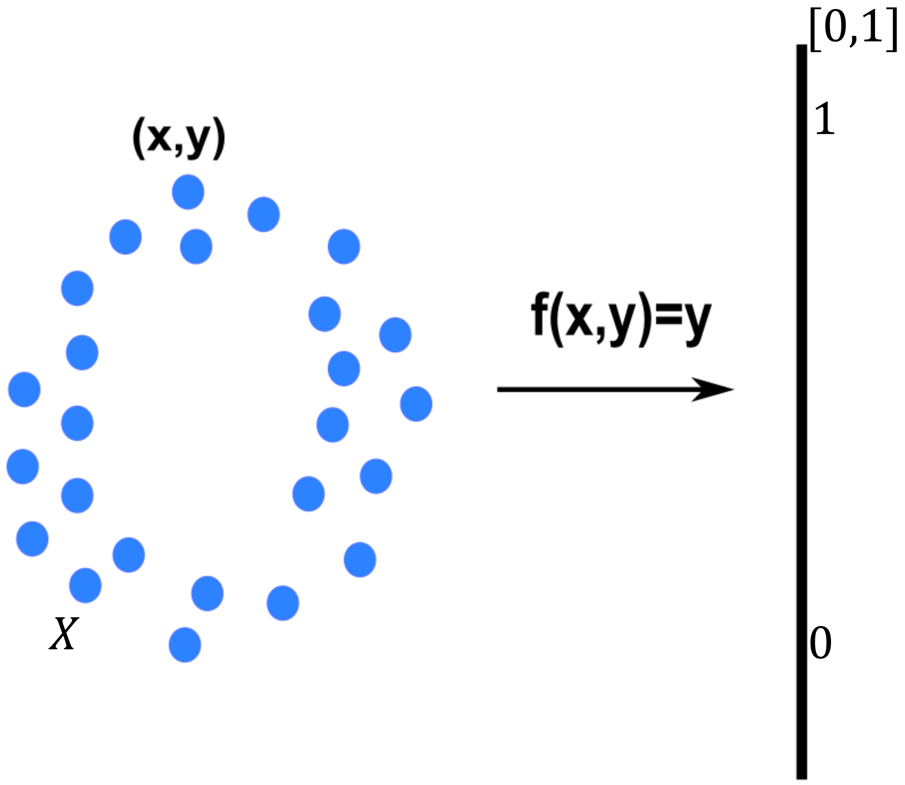


Mapper Construction



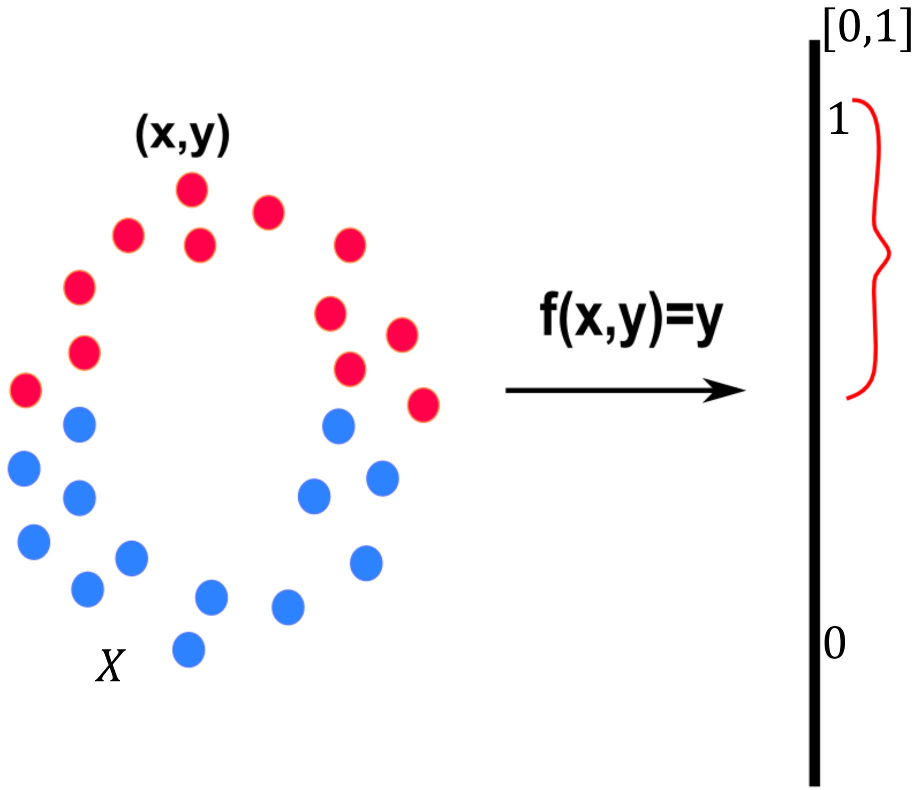
Mapper Example

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.



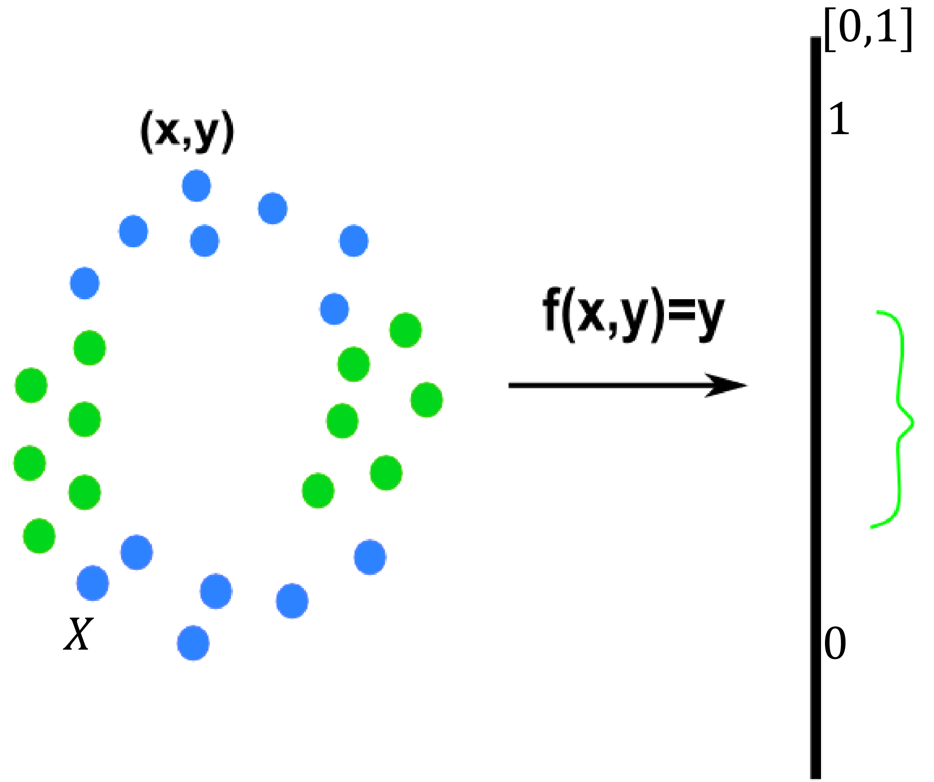
Mapper Example

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.



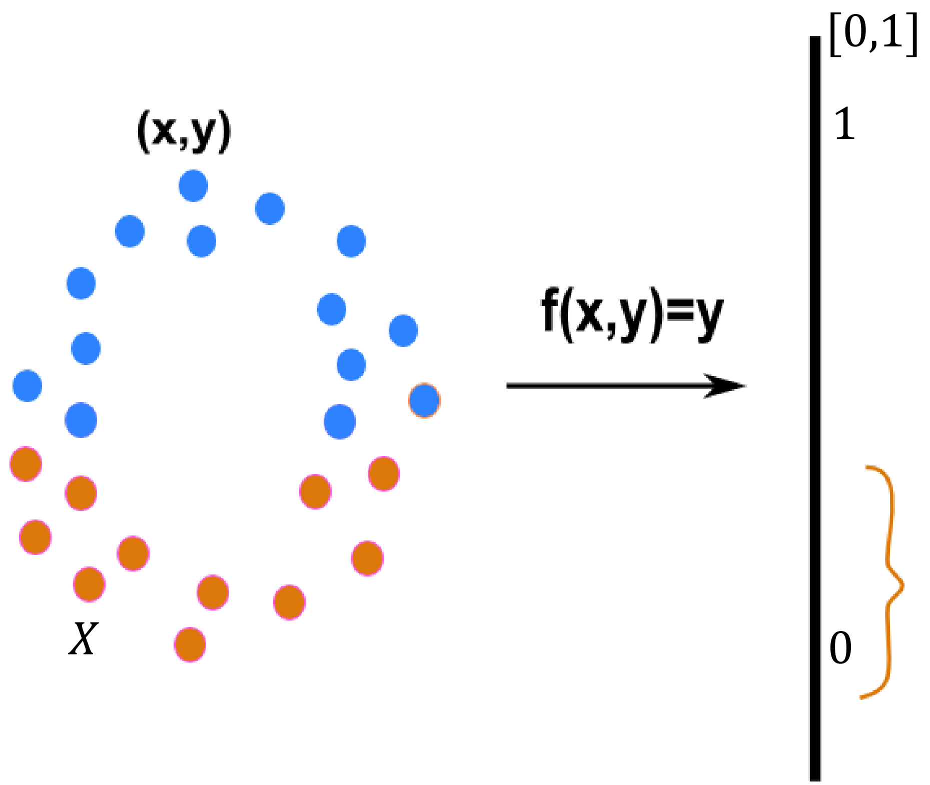
Mapper Example

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.



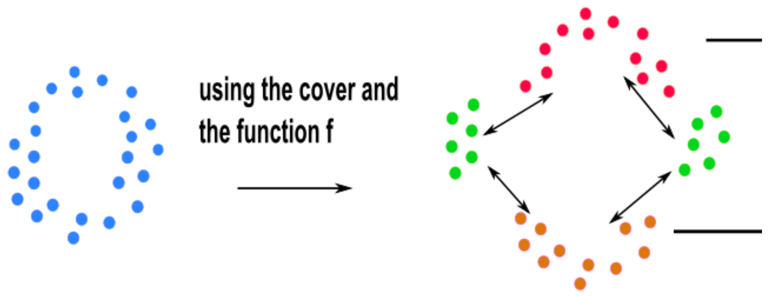
Mapper Example

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

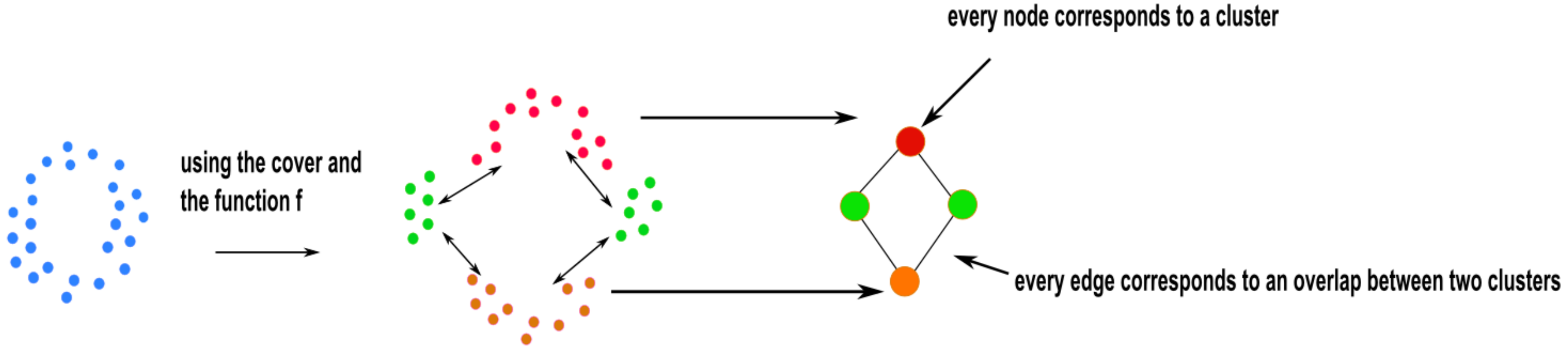


Mapper Example

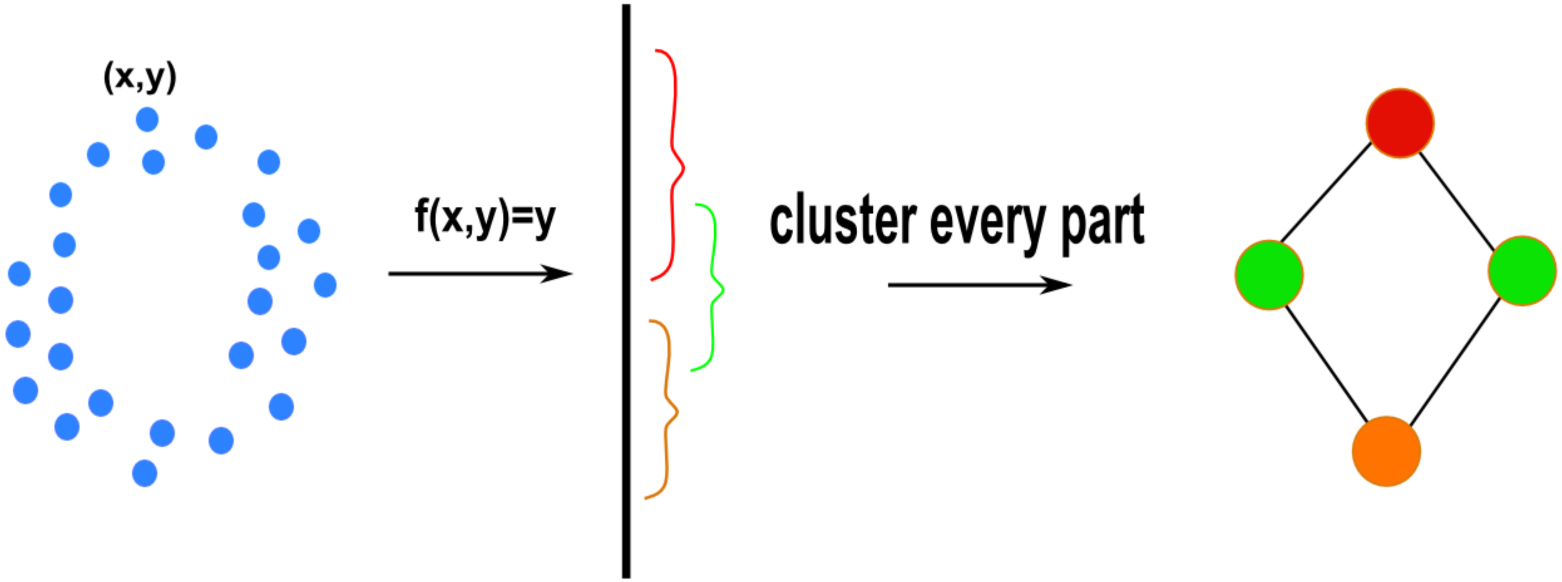
Mapper Example



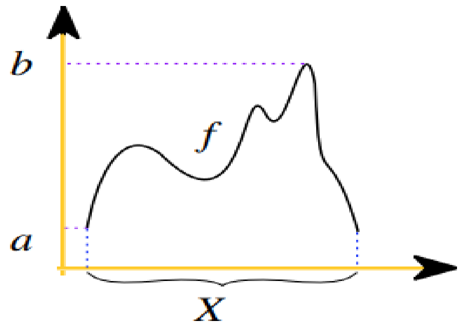
Mapper Example



Mapper Example



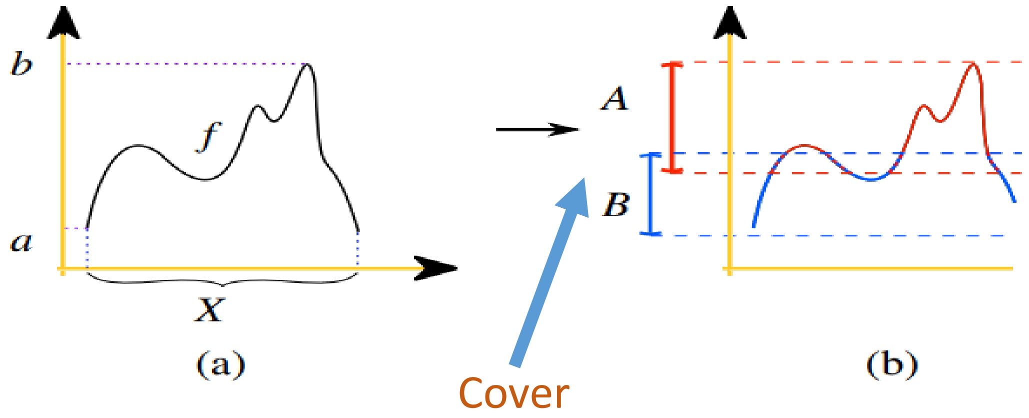
The construction of Mapper on a 1d function



(a)

(a) A scalar function f : X −→ [a,b].

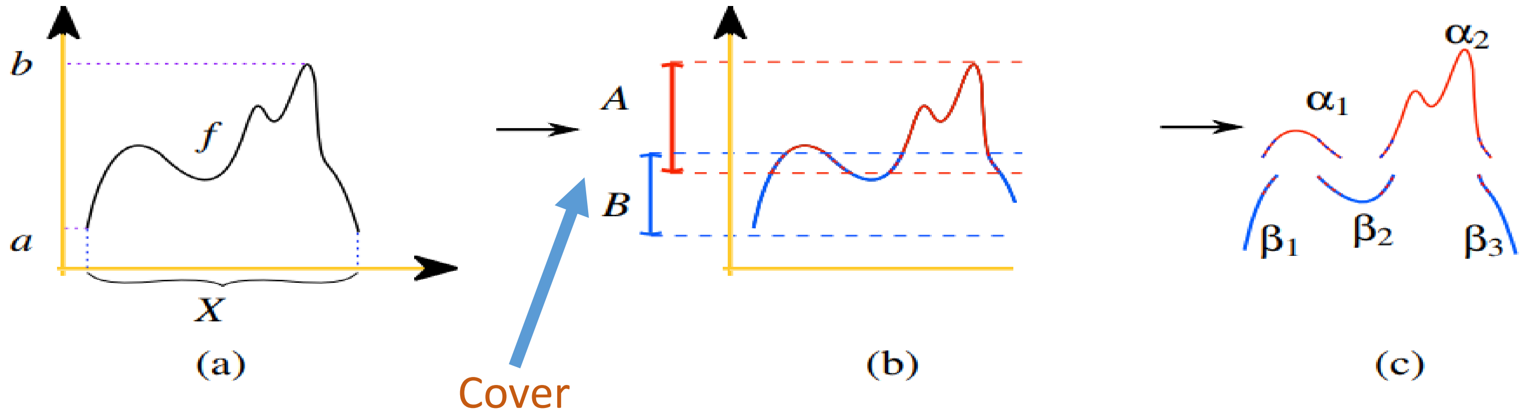
The construction of Mapper on a 1d function



(a) A scalar function f : X –→ [a,b].

(b) The range [a,b] is covered by the two intervals A,B.

The construction of Mapper on a 1d function

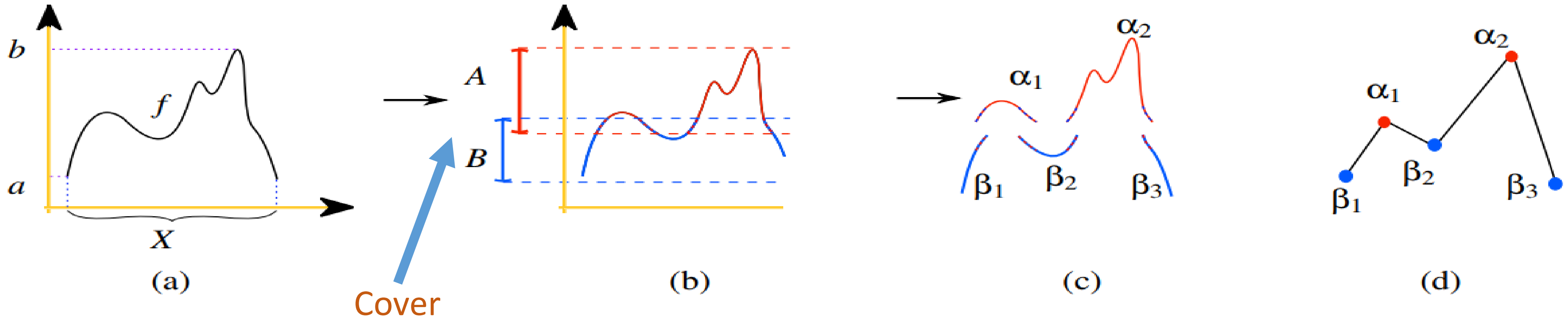


(a) A scalar function f : X –→ [a,b].

(b) The range [a,b] is covered by the two intervals A,B.

(c) This gives a decomposition of the domain the domain X. The inverse image of A consists of two connected components α1 and α2, and the inverse image of B consists of three connected components β1, β3 and β3.

The construction of Mapper on a 1d function



(a) A scalar function f : X –→ [a,b].

(b) The range [a,b] is covered by the two intervals A,B.

(c) This gives a decomposition of the domain the domain X. The inverse image of A consists of two connected components α1 and α2, and the inverse image of B consists of three connected components β1, β3 and β3.

(d) The connected components are represented by the nodes in the Mapper construction.

an edge is inserted whenever two connected components overlap.

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a,b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \ldots, U_n\}$

2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \; in \; U$

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i\ in\ U$
5-Run a clustering algorithm on on V_1 and store those clusters.

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \; in \; U$
5-Run a clustering algorithm on on V_1 and store those clusters.
6- Run the same clustering algorithm on every V_i

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \ldots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \; in \; U$
5-Run a clustering algorithm on on V_1 and store those clusters.
6- Run the same clustering algorithm on every V_i
7-Create an empty graph G.

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \ldots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \ in \ U$
5-Run a clustering algorithm on on V_1 and store those clusters.
6- Run the same clustering algorithm on every V_i
7-Create an empty graph G.
8- For every cluster we obtain from $\{V_i | 1 \leq i \leq n\}$ create a node for the graph G

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \rightarrow [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \, in \, U$
5-Run a clustering algorithm on on V_1 and store those clusters.
6- Run the same clustering algorithm on every V_i
7-Create an empty graph G.
8- For every cluster we obtain from $\{V_i | 1 \leq i \leq n\}$ create a node for the graph G
9- Check overlap between the clusters (nested for loop on all clusters) : whenever there is an overlap insert an edge between the corresponding nodes.

The Mapper Algorithm

Suppose that we are given a data set X and a scalar function $f: X \to [a, b]$ defined on every point in X.

1- Define a cover for the interval [a,b]. Say that this cover is $U = \{U_1, \dots, U_n\}$
2- Consider all the points x in X with f(x) in U_1. Put those points in container, say V_1
3-Consider all the points x in X with f(x) in U_2. Put those points in container, say V_2
4-Do that for every interval $U_i \; in \; U$
5-Run a clustering algorithm on on V_1 and store those clusters.
6- Run the same clustering algorithm on every V_i
7-Create an empty graph G.
8- For every cluster we obtain from $\{V_i | 1 \leq i \leq n\}$ create a node for the graph G
9- Check overlap between the clusters (nested for loop on all clusters) : whenever there is an overlap insert an edge between the corresponding nodes.
10-return the graph G