Reconstruction of univariate functions from persistence diagrams

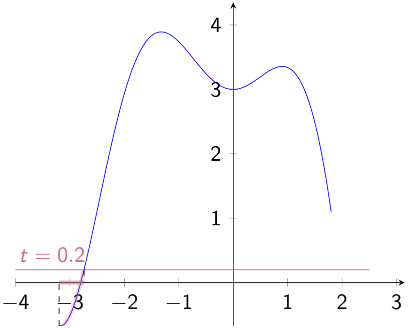
Aina Ferrà Marcús Dr. Carles Casacuberta Dr. Oriol Pujol

Universitat de Barcelona

June 1, 2022

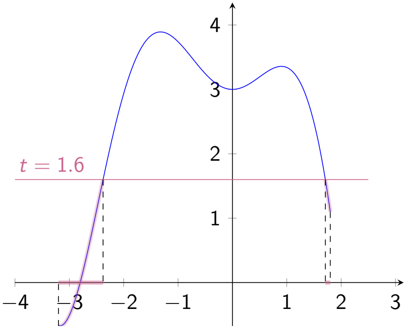
Motivation - Finding the shape

- Take some topological space.
- Select a parameter and perform a *filtration*.
- At each step, count its *topological features*.



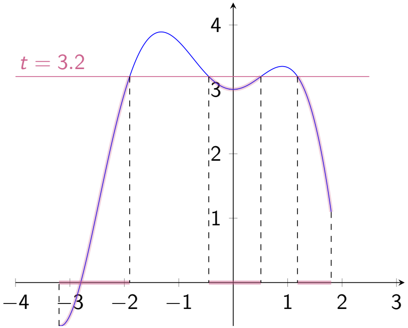
Motivation - Finding the shape

- Take some topological space.
- Select a parameter and perform a *filtration*.
- At each step, count its *topological features*.



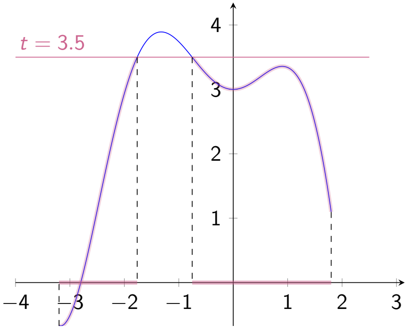
Motivation - Finding the shape

- Take some topological space.
- Select a parameter and perform a *filtration*.
- At each step, count its *topological features*.



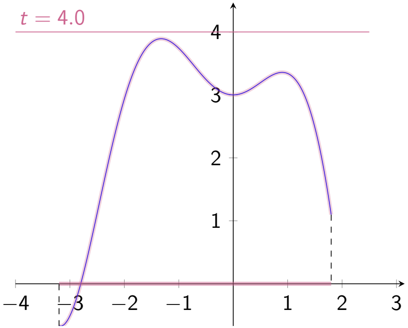
Motivation - Finding the shape

- Take some topological space.
- Select a parameter and perform a *filtration*.
- At each step, count its *topological features*.



Motivation - Finding the shape

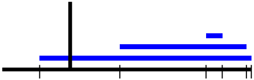
- Take some topological space.
- Select a parameter and perform a *filtration*.
- At each step, count its *topological features*.



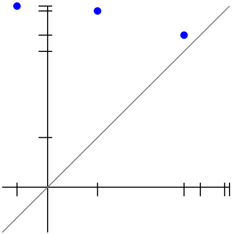
Motivation - Shape summary

There are many ways of summarize the information obtained through a filtration.

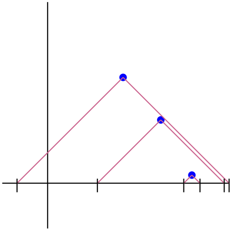
- **Vectorized / graphical summaries**



(a) Persistence barcodes



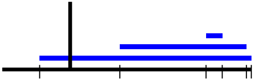
(b) Persistence diagrams



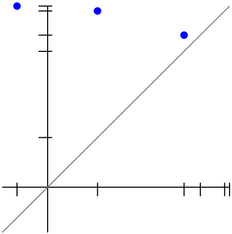
(c) Persistence landscapes

There are many ways of summarize the information obtained through a filtration.

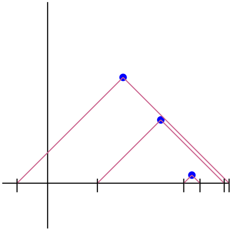
- **Vectorized / graphical summaries**



(a) Persistence barcodes



(b) Persistence diagrams



(c) Persistence landscapes

Given one vectorized representation, can we recover the original space?

Outline

- Mathematical formalization
  - Previous work
  - Current work
- A Machine Learning example
- General settings
- The piecewise linear case
  - Definitions
  - Theorem
  - The rolling ball algorithm
- The smooth case
  - Detection
  - Convergence
  - The five line algorithm

Mathematical formalization

Let M be a finite geometric simplicial complex in Euclidean space $\mathbb{R}^d$ with $d \geq 2$.

Definition

The **persistent homology transform** is a function

$$S^{d-1} \to \mathrm{Dgm}$$

that associates to each v the persistence diagram of $H_*(M_t(v); \mathbb{R})$, where $M_t(v) = \{x \in M \mid x \cdot v \leq t\}$,

- The PHT is continuous with respect to any Wasserstein distance on the set of persistence diagrams.
- Some theory has been developed to study its differenciability.

Previous work

- In 2013, it was proved that the PHT is injective for $d = 2$ and $d = 3$.
- In 2018, the result was extended over all dimensions.

Takeaway

Finite simplicial complexes embedded in $\mathbb{R}^d$ are uniquely determined by the collection of persistence diagrams of sublevel sets in **all** possible directions.

However, we need to produce efficient algorithms that use a **small number** of directions in order to reconstruct.

- In the case of graphs, three directions suffice.
- Bounds on the number of directions needed for reconstruction of compact definable sets in $\mathbb{R}^d$ can be found.

Current work

- Given $f : [a, b] \to \mathbb{R}$ continuous piecewise linear, with finitely many vertices, the graph

$$G(f) = \{(x, y) \in \mathbb{R}^2 \mid y = f(x)\}$$

  is as a finite geometric simplicial complex of dimension 1.
  We give an algorithm that recovers f in polynomial time requiring **three** admissible directions.
- We also present an algorithm that locates the local maxima and minima of a *smooth* function $f : [a, b] \to \mathbb{R}$, assuming that the second derivative does not vanish at critical points, using **five** admissible directions.

Current work

- Given $f\colon [a,b] \to \mathbb{R}$ continuous piecewise linear, with finitely many vertices, the graph

$$G(f) = \{(x,y) \in \mathbb{R}^2 \mid y = f(x)\}$$

  is as a finite geometric simplicial complex of dimension 1.
  We give an algorithm that recovers f in polynomial time requiring **three** admissible directions.
- We also present an algorithm that locates the local maxima and minima of a *smooth* function $f\colon [a,b] \to \mathbb{R}$, assuming that the second derivative does not vanish at critical points, using **five** admissible directions.

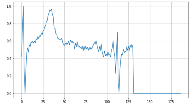
But why this?

Outline

Persistence Homology and Machine Learning

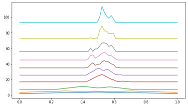
- Persistence Landscapes were introduced in 2015 and built a bridge between topological features and statistics. Such relation gave rise to **Topological Data Analysis**.
- The interest towards finding new and reliable Machine Learning algorithms has grown considerably, specially in the field of **importance attribution**.

Since Topological Data Analysis is able to capture the **shape of the data**, we propose a pipeline to **extract important information of input data functions**.



Input

→



Persistence Landscapes

→

Neural Network

- We start with a problem.

Question

Given graph of a function that represents a heartbeat, how can we know which kind of arrhythmia it has?

- We obtain a dataset of functions that represent heartbeats with different arrhythmia.
- We preprocess each of the examples and we obtain its persistence landscape representation.
- We train a neural network so it can make a classification. At the same time, the neural network will decide which levels of landscapes are important.
- We reconstruct an approximation of the function using only the most important landscapes, **it is the shape that the network regards as important.**

Topological Data Analysis works

We performed several experiments.

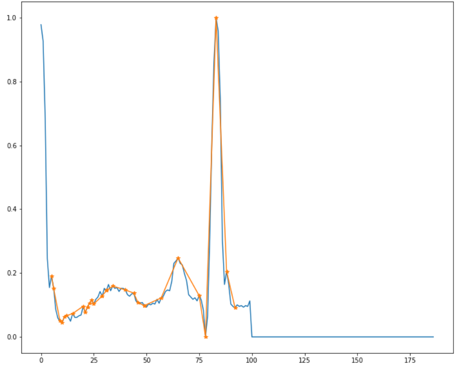
- First, we trained the neural network with 10 different levels of landscapes.
- We made the neural network choose which were important.
- We then retrain another neural network using only the most significant levels of landscapes and another using the least significant levels.
- We make sure that our technique works.

	Accuracy Training	Accuracy Test
Original	0.982	0.984
All landscapes	0.957	0.946
Most significant	0.958	0.946
Least significant	0.833	0.832

The accuracy is the percentage of properly classified samples.

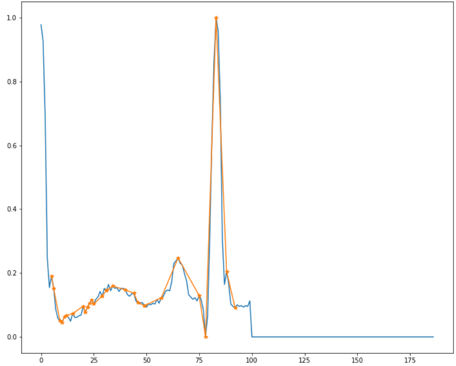
Machine Learning and reconstruction

Now it is time to see what the neural network is regarding as important, time to **reconstruct**.



Machine Learning and reconstruction

Now it is time to see what the neural network is regarding as important, time to **reconstruct**.



Ok but, what else?

Outline

Definitions

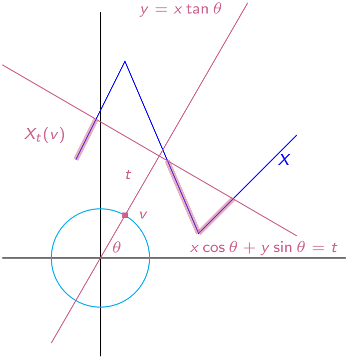
Let S^1 be the unit circle, view each $v \in S^1$ as a complex number $e^{i\theta}$.

- The *sublevel set* of $G(f)$ in the direction $v = e^{i\theta}$ at height t is

$$G(f)_t(v) = \{(x, f(x)) \in \mathbb{R}^2 \mid x\cos\theta + f(x)\sin\theta \le t\}.$$

- The *directional persistence module* of f at height t in the direction v is

$$M_t(f, v) = H_0(G(f)_t(v); \mathbb{R})$$



Definitions

Critical lines

A critical line at a height t for a direction v is a line in $\mathbb{R}^2$ orthogonal to v such that the filtered space $G(f)(v)$ changes its number of connected components at height t.

- Lines passing through the boundary points may not be critical lines.
- **Piecewise linear case**: each critical line in any direction contains at least one critical point of f.
- **Smooth case**: critical lines are tangent to the graph of f (hopefully, *close* to a critical point).

Definitions

Proposition

For a direction v, a line orthogonal to v at a height t is critical for a function f if and only if either $t = \beta$ for some point (β, δ) in the persistence diagram for zero-homology H_0 of sublevel sets of f in the direction v, or $t = \delta$ when δ is finite.

- We will always work with persistence diagrams.
- With a simple computation, we can get all the critical lines.

Outline

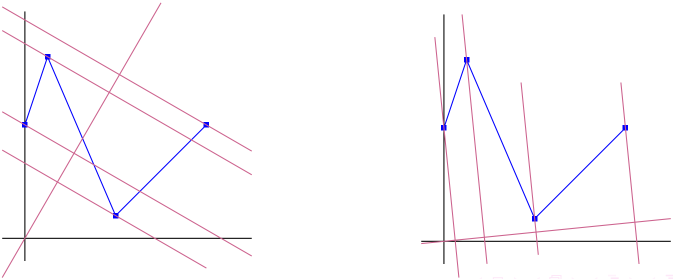
Piecewise linear case - definition

Let f be a piecewise linear continuous function with n critical points.

- The number of critical lines in each direction is **less or equal than** n.
- A line passing through a critical point **can fail to be critical** and a critical line can pass through two or more critical points.

Admissible direction

A direction is *admissible* if for each critical point of f there is a neighbourhood where the graph fully lies at one side of a line orthogonal to the direction going through the critical point.



Piecewise linear case - Theorem

Theorem

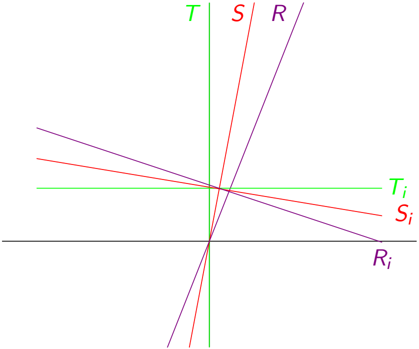
Let f be a continuous piecewise linear function, and let v_0, v_1, v_2 be admissible directions.
Let $\mathcal{P}$ be the set of **triple intersection points** determined by the three directional persistence diagrams. Suppose that no critical line orthogonal to v_0, v_1 or v_2 contains two or more points from $\mathcal{P}$.

Then $\mathcal{P}$ **is equal to the set of critical points of** f, excluding the boundary points if these are local maxima.

- By joining each pair of consecutive triple intersection points we obtain a piecewise linear approximation with the same critical points as f.
- The subset of S^1 of those directions for which there are critical lines passing through two or more triple points has measure zero.

The rolling ball algorithm

- Fix three directions, normally $\theta_0 = 90^\circ$, $\theta_1 = 85^\circ$ and $\theta_2 = 80^\circ$.
- Look for **triple intersections** within the set of critical lines determined by the three directional persistence diagrams.



A naive algorithm would take $O(n^3)$ where n is the number of critical points of the original function.

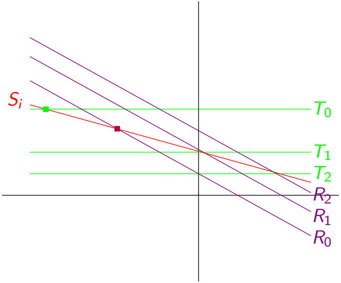
The rolling ball algorithm makes at most $2n \log n + 2n^2$ operations, hence it takes $O(n^2)$!

The rolling ball algorithm

- For a line S_i, consider the intersections

$$p_t = S_i \cap T_0, \qquad p_r = S_i \cap R_0.$$

- If $p_t = p_r$ we have found a triple intersection and we move to the next S_{i+1}.
- Otherwise, compare the x-values of the intersections and discard the line with a lower x value.



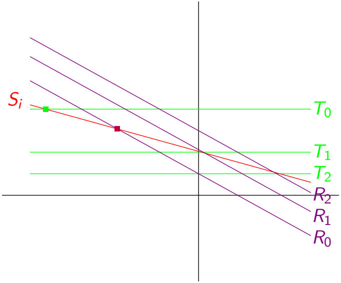
(a) $x(p_t) < x(p_r)$

The rolling ball algorithm

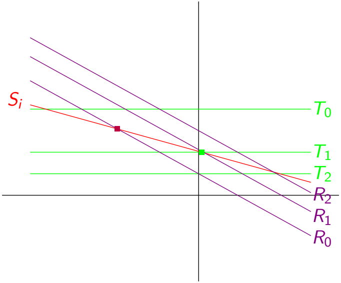
- For a line S_i, consider the intersections

$$p_t = S_i \cap T_0, \qquad p_r = S_i \cap R_0.$$

- If $p_t = p_r$ we have found a triple intersection and we move to the next S_{i+1}.
- Otherwise, compare the x-values of the intersections and discard the line with a lower x value.

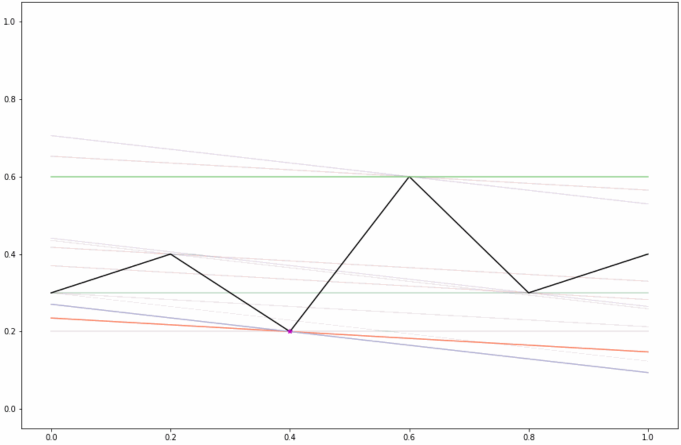


(a) $x(p_t) < x(p_r)$



(b) $x(p_r) < x(p_t)$

The rolling ball algorithm



Outline

The smooth case

- In the smooth case, we cannot have critical lines of different directions be tangent to the same critical point, so the problem becomes a bit more involved.
- The closest thing to a triple intersection that we can do with three different lines is a **triangle**. So now we look for suitable triangles.
- But once we have a triangle, we have to produce a good approximation of the actual critical point.

We will treat detection and convergence separately.

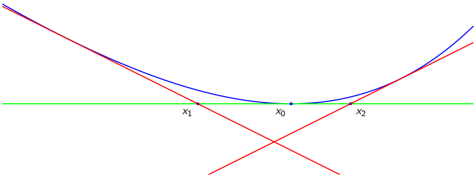
- Now, directions are not completely independent from each other.
- The vertical direction **will yield the y-coordinates** of the critical points.
- Each time we choose a direction, we will also look at its **symmetric direction with respect to the vertical line**.

The smooth case - detection

Detection

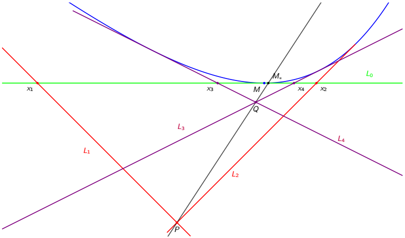
Let (x_0, y_0) be a critical point of $f : [a, b] \to \mathbb{R}$. For a direction $v = e^{i\theta}$ and a positive real number $\tau > 0$, we say that (x_0, y_0) is τ-*detected* by the direction v if there are critical lines of slopes $\pm 1/\tan\theta$ intersecting $y = y_0$ at points (x_1, y_0) and (x_2, y_0) with

- x_0 is between x_1 and x_2.
- x_1 and x_2 are at distance less than τ.
- There is no other critical point with x-coordinate between x_1 and x_2.



The smooth case - convergence

- Let (x_0, y_0) be a local *minimum*, detected by $v_1 = e^{i\theta_1}$.
- If $m_1 = 1/\tan\theta_1$, consider lines L_1 and L_2 with slopes $-m_1$ and m_1.
- Let L_0 the horizontal critical line $y = y_0$.
- Choose another direction $v_2 = e^{i\theta_2}$ with $0 < \theta_1 < \theta_2 < \pi/2$.
- If $m_2 = 1/\tan\theta_2$, consider lines L_3 and L_4 with slopes $-m_2$ and m_2
- Let (x_i, y_0) denote the intersection point of L_i with L_0 for $i = 1, 2, 3, 4$.



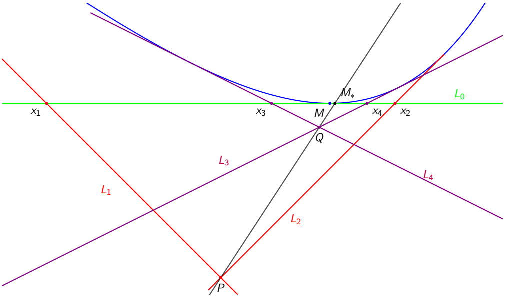
Then x_* is taken as an approximation of x_0, where

$$x_* = \frac{m_2(x_1 + x_2)(x_3 - x_4) - m_1(x_1 - x_2)(x_3 + x_4)}{2m_2(x_3 - x_4) - 2m_1(x_1 - x_2)}$$

The smooth case - convergence

Theorem

Let $f : [a,b] \to \mathbb{R}$ be a smooth function, and let (x_0, y_0) be a critical point of f with $a < x_0 < b$ and such that $f''(x_0) \neq 0$. Then x_0 can be approximated with any arbitrary degree of precision by means of critical lines.



The five line algorithm

The algorithm works in the following way

- **Detect**. We start with a predefined value of τ and a direction $v = e^{i\theta_1}$ that we assume admissible.
  - All triangles with base less than τ formed by critical lines are tentatively assumed to contain a critical point
- **Clean and converge.** We try to eliminate false positives and accurately locate true positives.
  - The working hypothesis is that critical lines with slope close to 0 will only pass inside a triangle from the first step if that triangle truly contains a critical point.
  - So, we pick another direction $v = e^{i\theta_2}$ that will produce a slope of critical lines close to 0. We test if a pair of such critical lines lies inside a triangle of the first step. We disregard candidate triangles where betweenness fails.

The five line algorithm

The choice of parameters depends on the signal functions to which the algorithm is to be applied, and they are **unknown to the algorithm**.

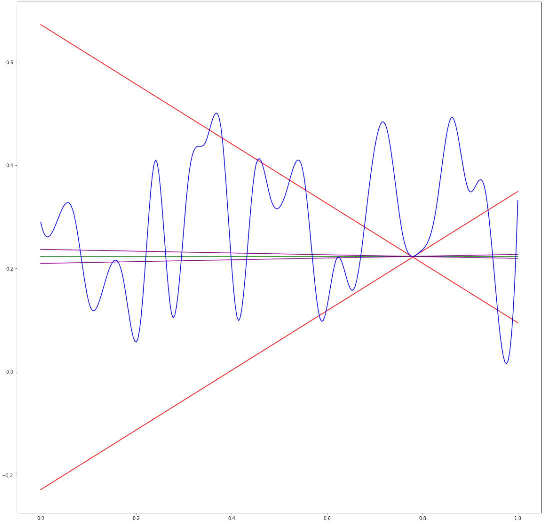
- The choice of τ is crucial.
  - Bigger values of τ ensure that no critical point is missed, but the chances of false positives increase.
  - Smaller values of τ may miss some critical points but also reduce the number of candidate triangles, thus diminishing the possibility of a false positive.

The five line algorithm

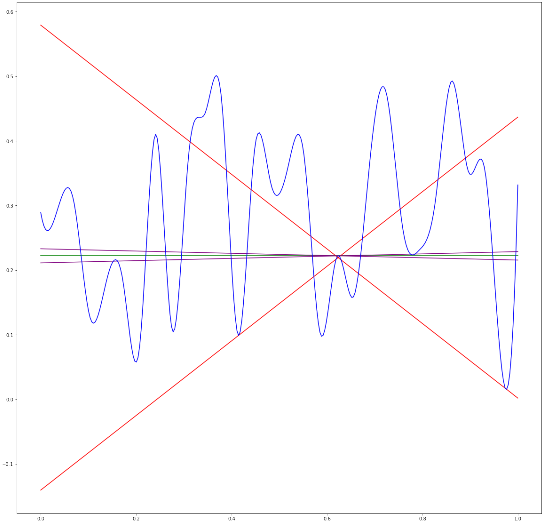
The choice of parameters depends on the signal functions to which the algorithm is to be applied, and they are **unknown to the algorithm**.

- The algorithm can fail due to the existence of lines that are tangent at a critical point and very close to being tangent at another critical point of the graph, called *quasi-multiple* tangent lines.
  - Triangles formed with quasi-multiple tangent lines usually appear next to triangles that truly contain a critical point.
  - The same pair of critical lines with small slope appears between both triangles, resulting in closeby duplicate images of the same critical point.
  - The algorithm eliminates the duplicates.

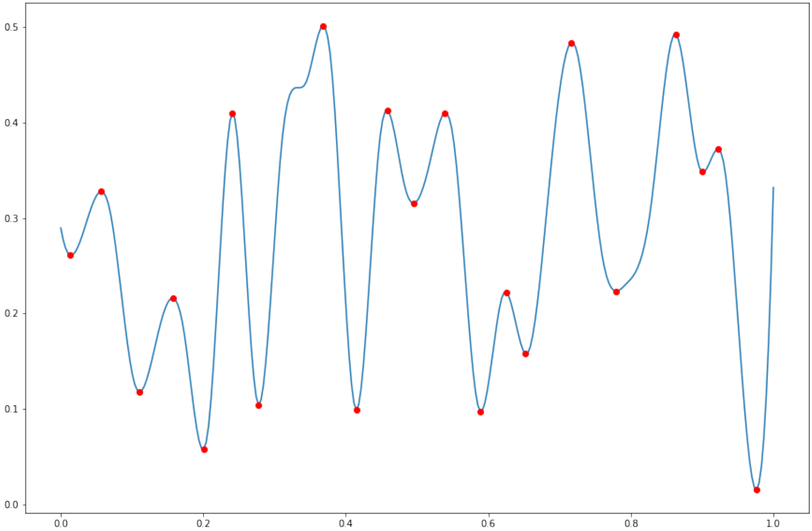
The five line algorithm



The five line algorithm



The five line algorithm



Thank you for your attention!