Alma Mater Studiorum Università di Bologna
Archivio istituzionale della ricerca

Symmetric functions for fast image retrieval with persistent homology

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

This version is available at: https://hdl.handle.net/11585/646964 since: 2018-10-17

Published:

Terms of use:

(Article begins on next page)

Symmetric functions for fast image retrieval with persistent homology

Alessia Angeli[*], Massimo Ferri[†]and Ivan Tomba[‡]

May 10, 2018

Abstract

Persistence diagrams, combining geometry and topology for an effective shape description used in pattern recognition, have already proven to be an effective tool for shape representation with respect to a certain filtering function. Comparing the persistence diagram of a query with those of a database allows automatic classification or retrieval, but unfortunately, the standard method for comparing persistence diagrams, the bottleneck distance, has a high computational cost. A possible algebraic solution to this problem is to switch to comparisons between the complex polynomials whose roots are the cornerpoints of the persistence diagrams. This strategy allows to reduce the computational cost in a significant way, thereby making persistent homology based applications suitable for large scale databases. The definition of new distances in the polynomial framework poses some interesting problems, both of theoretical and practical nature. In this paper, these questions have been addressed by considering possible transformations of the half-plane where the persistence diagrams lie onto the complex plane, and by considering a certain re-normalisation the symmetric functions associated to the polynomial roots of the resulting transformed polynomial. The encouraging numerical results, obtained in a dermatology application test, suggest that the proposed method may even improve the achievements obtained by the standard methods using persistence diagrams and the bottleneck distance.

1 Introduction

Persistent Homology is a recent branch of Computational Topology, which combines geometry and topology for an effective shape description used in Pattern Recognition.

It has already proven to be an effective tool for shape representation in various applications, in particular it is a fairly popular tool for comparing, retrieving and classifying objects having a natural origin, especially medical images. Examples of such applications are liver lesion image classification (see [1]), brain data analysis ([2] and [3]) and dermatology, where persistent homology has been tested with encouraging results [4, 5, 6, 7].

---

[*]Dept. of Mathematics, University of Bologna, Piazza di Porta San Donato 5, Bologna, Italy

[†]Dept. of Mathematics, University of Bologna, Piazza di Porta San Donato 5, Bologna, Italy

[‡]R&D Dept., CA-MI S.r.l., Via Ugo La Malfa 13, Pilastro di Langhirano, Parma, Italy

1

The standard tool for shape comparison is the natural pseudo-distance [8], which provides a dissimilarity measure between two topological spaces under a certain filtering function, which represents the aspect of the spaces to be compared. Due to the abstract nature of its definition the natural pseudo-distance is not directly computable. Luckily, the natural pseudo-distance can be approximated by comparing persistence diagrams (also known as bar codes), which condense the essence of the pair constituted by the topological space and the filtering function in finite sets of points in the plane [9]: the bottleneck distance (or matching distance) between persistence diagrams yields an optimal lower bound to the natural pseudo-distance [8, 10]. However, the bottleneck distance suffers from combinatorial explosion [11], so it may become impractical to scan a large database to produce a retrieval or an automatic classification. Approximations such as image size reductions, smart organization of the database according to the metric, progressive application of different classifiers come to help, but do not solve the problem completely. In fact, reducing images too much may result in a significant loss of information; smart organization of the database lightens the problem of a real-time retrieval as it allows to avoid some distance computations between couples of objects, but has the drawback of requiring an a-priori computation of all relative distances in the database: this results in $O(N_{db}^2)$ bottleneck distance computations (where N_{db} is the number of images stored in the database), which for persistence diagrams with thousands of cornerpoints may become too demanding from a computational point of view.

A possible solution, proposed for the first time in [12], is to represent a persistence diagram as the set of complex roots of a polynomial; then comparison can be performed on coefficients, or on a proper subset of the coefficients [13]. From the computational point of view, this has a double advantage: first, it allows to store only a fixed number of coefficients of the polynomials instead of the persistence diagrams, sparing memory; second, it accelerates the computation of the distances, as the distance between two coefficient vectors is cheaper than the bottleneck distance. This method is intriguing, but requires the solution of an important issue: in real situations there are a lot of points near the diagonal $\Delta = \{(u, v) \in \mathbb{R}^2 : u = v\}$, due to noise, which are less meaningful in shape representation, but with a heavy impact on polynomial coefficients. Indeed, the metric of the half-plane $\Delta^+ = \{(u, v) \in \mathbb{R}^2 : u < v\}$, where persistence diagrams lie, is not to be considered Euclidean, as the diagonal acts like a unique point/object and all points close to the diagonal are close to each other in the sense introduced by the bottleneck distance. Thus, it makes sense, before passing from the persistence diagrams to the polynomial coefficients, to apply a geometric transformation which maps the entire diagonal to the origin of the complex plane and all cornerpoints close to the diagonal close to the origin. Such transformations are of interest on their own (i.e. independently on the choice to consider polynomial coefficients), as they map the persistence diagrams onto a more standard metric space, such as the complex plane with the Euclidean distance. Some transformations have been proposed in [13]: here a new transformation is proposed and compared to them in the numerical experiments.

Another practical problem, not discussed in [13], arises when dealing with persistence diagrams with many cornerpoints: some coefficients may become very large, causing instabilities in the distance computations and more generally in the metric of the database. A solution of this problem

is proposed as well.

At last it should be remembered, from polynomial theory, that a small distance between polynomial roots implies a small distance of coefficients, but the converse is false. Thus, it is possible that two very different persistence diagrams have very similar polynomial coefficients: this is the price to be paid for attempting a computational simplification. A possible solution is to use polynomial comparison as a preprocessing phase in shape retrieval, i.e. as a very fast way of getting rid of definitely far objects, so that the bottleneck distance can be computed only for a small set of candidates. However, as [13] and this paper show, the use of polynomials appears to work well in the experiments, suggesting that in practice the unfortunate case in which dissimilar persistence diagrams give rise to similar coefficient vectors is probably quite rare.

The paper is organised as follows: Section 2 provides the theoretical background required for understanding the paper; Section 3 defines a new distance between transformed persistence diagrams, in comparison to the bottleneck distance; in Section 4 the new algorithm is stated and its computational complexity is discussed; Section 5 presents the numerical simulations conducted in the dermatology application and Section 6 is dedicated to further possible developments and to the conclusions.

2 Preliminaries

In persistence, each shape is viewed as a pair (X, f): a topological space X, and a continuous function $f : X \to \mathbb{R}$, called *filtering function* (or *measuring function*), to define a family of subspaces $X_\ell = \{x \in X : f(x) \le \ell\}$, $\ell \in \mathbb{R}$, nested by inclusion, i.e. a filtration of X.

Given $u, v \in \mathbb{R}$, $u < v$, we consider the continuous inclusion $i^{u,v} : X_u \to X_v$. This inclusion induces a homeomorphism of homology groups $i_*^{u,v} : H_k(X_u) \to H_k(X_v)$ for every $k \in \mathbb{Z}$. The image of this application are the k-homology classes that are born before or at level u and are still alive at the level v and it is called the *k-th persistent homology group of (X, f) at (u, v)*. If this group is finitely generated, we will denote by $\beta_k^{u,v}(X, f)$ its rank.

Persistent homology groups of (X, f) are often described using *persistence diagrams*, i.e. multisets of points whose abscissa and ordinate are, respectively, the level at which k-homology classes are created and the level at which they are annihilated though the filtration (*proper cornerpoints*). If a homology class does not die along the filtration, the ordinate of the corresponding point is set to $+\infty$ (*cornerpoint at infinity*).

Let $\Delta^+ = \left\{(u, v) \in \mathbb{R}^2 : u < v\right\}$, $\Delta = \left\{(u, v) \in \mathbb{R}^2 : u = v\right\}$, $\bar{\Delta}^+ = \Delta^+ \cup \Delta$.

Definition 2.1. Proper cornerpoint.
For every point $(u, v) \in \Delta^+$, let us define the number $\mu_k(u, v)$ as the minimum over all the positive real numbers ϵ, with $u + \epsilon < v - \epsilon$, of

$$\beta_k^{u+\epsilon, v-\epsilon}(X, f) - \beta_k^{u-\epsilon, v-\epsilon}(X, f) - \beta_k^{u+\epsilon, v+\epsilon}(X, f) + \beta_k^{u-\epsilon, v+\epsilon}(X, f)$$

The finite number $\mu_k(u, v)$ is called multiplicity of $(u, v) \in \Delta^+$ for $\beta_k(X, f)$ *and the points $(u, v) \in \Delta^+$ such that $\mu_k(u, v) > 0$ are called* proper cornerpoints *for $\beta_k(X, f)$.*

Definition 2.2. Cornerpoint at infinity.
For every vertical line r, with equation $x = w$, let us identify r with the

pair $(w, +\infty)$, and define the number $\mu_k(r)$ as the minimum, over all the positive real numbers ϵ with $w + \epsilon < \frac{1}{\epsilon}$, of

$$\beta_k^{w+\epsilon, \frac{1}{\epsilon}}(X, f) - \beta_k^{w-\epsilon, \frac{1}{\epsilon}}(X, f)$$

The finite number $\mu_k(r)$ is called multiplicity of r for $\beta_k(X, f)$ and the lines r such that $\mu_k(r) > 0$ are called cornerlines or cornerpoints at infinity for $\beta_k(X, f)$.

A cornerline is identified as a cornerpoint with infinite ordinate so it escapes our processing. Therefore, in this work, we take the cornerline to a particular cornerpoint with finite coordinates.

Given a cornerline with abscissa w, we substitute it with the point

$$(w, \max \{v \, : \, (u, v) \text{ is a proper cornerpoint}\})$$

In this way we represent the cornerlines with particular proper cornerpoints so in this paper, from now on, we will simply talk about cornerpoints.

Definition 2.3. Persistence diagram.
The k-th persistence diagram $\mathcal{D}_k(X, f)$ is the set of cornerpoints, each counted with its multiplicity, union the set of points of Δ, counted with infinite multiplicity.

Persistence diagrams are usually compared computing the so called *bottleneck distance* (or *matching distance*) because of the robustness of these descriptors with respect to it: roughly, small changes in a given filtering function produces just a small change in the associated persistence diagram. The bottleneck distance between two persistence diagrams measures the cost of finding a correspondence between their points. In particular, the cost of taking a point P to a point P' is measured as the minimum between the cost of moving one point onto the other and the cost of moving both points onto the diagonal and the matching of a cornerpoint P with a point of Δ can be interpreted as the destruction of the cornerpoint P.

Definition 2.4. Bottleneck distance *or* matching distance.
Let $\mathcal{D}_k$ and $\mathcal{D}'_k$ be two persistence diagrams with a finite number of cornerpoints, the bottleneck distance $d_B(\mathcal{D}_k, \mathcal{D}'_k)$ *is defined as*

$$d_B(\mathcal{D}_k, \mathcal{D}'_k) = \min_\sigma \max_{P \in \mathcal{D}_k} \hat{d}(P, \sigma(P))$$

where σ varies among all the bijections between $\mathcal{D}_k$ and $\mathcal{D}'_k$ and

$$\hat{d}((u, v), (u', v')) = \min \left\{ \max \{|u - u'|, |v - v'|\}, \max \left\{ \frac{v - u}{2}, \frac{v' - u'}{2} \right\} \right\}$$

given $(u, v) \in \mathcal{D}_k$ and $(u', v') \in \mathcal{D}'_k$.

All persistence diagrams considered in this work, related to the problem about recognition of melanocytic lesions, are 0-persistence diagrams, so in this paper, from now on, we shall drop the 0- prefix.

Detailed information on persistence can be found in [14, 15].

3 Symmetric functions of warped diagrams

Evaluating the bottleneck distance between persistence diagrams can be computationally very expensive, making its usage not practicable on large databases. In this work we propose a new procedure taken from the procedure explained in [13] and based on the preliminary idea introduced in [12]. The problem of comparing directly two persistence diagrams through the bottleneck distance is translated into the problem of comparing complex vectors associated with each persistence diagram through an appropriate metric between vectors. The components of these complex vectors are the values of elementary symmetric functions of transformed cornerpoints of persistence diagrams.

3.1 Warping the plane

Given a persistence diagram, firstly we define a transformation taking the cornerpoints to the set of complex numbers. Secondly, we consider the elementary symmetric functions of transformed cornerpoints to construct complex vectors.

Initially we analysed the transformation previously published in [13]:

$$T : \bar{\Delta}^+ \to \mathbb{C}, \quad T(u,v) = \frac{v-u}{2}(\cos(\alpha) - \sin(\alpha) + i(\cos(\alpha) + \sin(\alpha)))$$

where $\alpha = \sqrt{u^2 + v^2}$.

Successively we defined a new transformation in the following way:

$$R : \bar{\Delta}^+ \to \mathbb{C}, \quad R(u,v) = \frac{v-u}{\sqrt{2}}(\cos(\theta) + i\sin(\theta))$$

where $\theta = \pi(u + v)$.

Both T and R are continuous maps.

We define the multiplicity of a complex number in the range of T or R to be the sum of the multiplicities of the points belonging to its preimage.

These transformations warp the "diagonal" Δ to the origin $(0,0)$ and take the cornerpoints with distance $\bar{d}$ from Δ to points with distance $\bar{d}$ from the origin. In this way the cornerpoints near Δ, principally due to noise, are all moved close to the origin. In fact the goal of these transformations is to try to reduce the impact of these cornerpoints near Δ in the sums of products which will build the elementary symmetric functions of transformed cornerpoints we are going to compare.

However, these transformations have different effects and the main difference is the following: the transformation T rotates the ray on which a given cornerpoint lies, proportionally to the distance from the origin $(0,0)$ of the cornerpoint itself; the transformation R takes segments orthogonal to Δ to rays coming out of the origin $(0,0)$.

Remark 3.1. *Regarding the transformations T and R, it is important to observe that they are not injective on Δ^+, but injectivity is restored if they are restricted to the triangle $\mathcal{T} = \{(u,v) \in \mathbb{R}^2 : 0 \leq u < v \leq 1\}$. The restriction can be achieved by normalising all filtering functions in the range $[0,1]$. We also considered several variations of these to try to better distribute the noise. However, the use of these variants did not lead to significant or global improvements in the obtained results, so we decided to keep their simplest versions.*

Example 3.1. *When dealing, e.g., with melanocytic lesions, we report an example of how the transformations just described take the cornerpoints to complex numbers. The graphs of Fig. 1 are obtained from image $IMD435$*

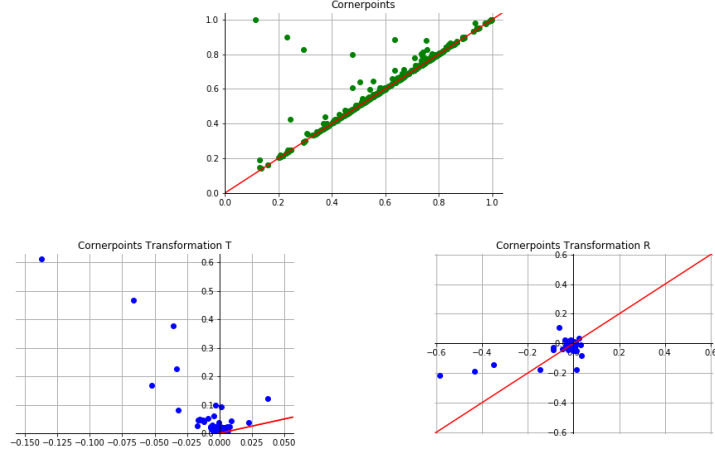*of the database PH^2, with the filtering function Bound and transformations T and R.*

Figure 1: *Database PH^2, image $IMD435$, filtering function Bound.*

In [13] only proper cornerpoints are considered, while in this work we consider also cornerlines, because they carry information which sometimes turns essential in the present experimentation.

Example 3.2. *When dealing, e.g., with melanocytic lesions, with shade of brown as a filtering function, given two images both having a fairly homogeneous hue but one light and one dark colored, these could have very similar persistence diagrams except for the abscissas of cornerlines.*

3.2 Comparing warped diagrams through symmetric functions

Let $\mathcal{D}$ be a persistence diagram and $c_1 = (u_1, v_1), \dots, c_h = (u_h, v_h)$, where $c_1, \dots, c_h \in \mathbb{R}^2$, its cornerpoints with multiplicity $r_1, \dots, r_h$ respectively. Let now the complex numbers $z_1, \dots, z_h$ be obtained from $c_1, \dots, c_h$ by one of the transformations T, R. We associate to $\mathcal{D}$ the complex vector $a_{\mathcal{D}}$ whose components are the values of elementary symmetric functions of transformed cornerpoints. So we have:

$$a_{\mathcal{D}}(j) = \sum_{1 \le i_1 < i_2 < \dots < i_j \le n} z_{i_1} z_{i_2} \dots z_{i_j}, \qquad j = 1, 2, \dots, n \qquad (1)$$

where n is the number of cornerpoints of $\mathcal{D}$ counted with multiplicity.

Now let $\mathcal{D}$ and $\mathcal{D}'$ be a persistence diagrams with n and m cornerpoints counted with multiplicity, respectively. If $n \ne m$ ($m < n$ say) the complex vectors $a_{\mathcal{D}}$ and $a_{\mathcal{D}'}$ corresponding respectively to $\mathcal{D}$ and $\mathcal{D}'$ have different lengths. We can manage this problem by also computing $a_{\mathcal{D}'}(j)$ with $j = m+1, \dots, n-1, n$ adding the complex number zero with multiplicity $n - m$ to the set of complex numbers obtained from the cornerpoints of $\mathcal{D}'$. In so doing, we can build two vectors of complex numbers $a_{\mathcal{D}}$ and $a_{\mathcal{D}'}$ of the same length and then we are ready to compute a distance between them.

As for the distance to be used among complex vectors $a_{\mathcal{D}}$ and $a_{\mathcal{D}'}$, we can observe that the preliminary tests in [13] suggested that the first components of these vectors were more meaningful. Therefore we consider only the first k components of these vectors, $k \in \{1, ..., n\}$. The chosen metric is the following:

$$d(a_{\mathcal{D}}, a_{\mathcal{D}'}) = \sum_{j=1}^{k} |a_{\mathcal{D}}(j) - a_{\mathcal{D}'}(j)| \tag{2}$$

Applying the algorithm previously described to the persistence diagrams $\mathcal{D}$ and $\mathcal{D}'$ it happens that the difference between the orders of magnitude of $a_{\mathcal{D}}(j)$ and $a_{\mathcal{D}'}(j)$, $j \in \{1, ..., n\}$, can be relevant (more precisely, we are talking about the order of magnitude of the absolute value of the real and imaginary parts): up to six! If this occurs only for some $j \in \{1, ..., n\}$, then the distance will essentially depend only on the components with these indices, stymieing the contribution of the others.

We actually compare only the first k components, but we want the distance to take all of them into account. Moreover, if this happens for a rather big index j, this contrasts with our assumption of greater importance of the first components.

The main cause of this phenomenon seems to be the dependence on the number of cornerpoints, which can vary quite a lot from image to image. The summation yielding the j-th component will have $\binom{n}{j}$ addends but many of these addends could be null if this persistence diagram has a number of cornerpoints much smaller than n. Instead, if all the addends of this summation have values close to 1, then $a_{\mathcal{D}}(j)$ will be close to $\binom{n}{j}$ with great growth at the approach of j to $\frac{n}{2}$.

Example 3.3. *With the database PH^2 and the filtering function LI2, there are persistence diagrams with 47, 77 or 156 cornerpoints but there are also persistence diagrams with a higher number of cornerpoints like 1444, 1247 or 1815.*

We then apply a further transformation to $a_{\mathcal{D}}(j)$ by extracting the j-th root and dividing by the number of cornerpoints of the persistence diagram considered to try to mitigate the dependence of the values of elementary symmetric function of transformed cornerpoints on that number.

More precisely, let $a_{\mathcal{D}}$ and $a_{\mathcal{D}'}$ be two complex vectors with the same length n and associated respectively to persistence diagrams $\mathcal{D}$ and $\mathcal{D}'$, the defined transformation is the following:

$$a_{\mathcal{D}}(j) \quad \longmapsto \quad \tilde{a}_{\mathcal{D}}(j) = \frac{|a_{\mathcal{D}}(j)|^{\frac{1}{j}}}{N} e^{i arg(a_{\mathcal{D}}(j))}, \quad j \in \{1, ..., k\}$$

$$a_{\mathcal{D}'}(j) \quad \longmapsto \quad \tilde{a}_{\mathcal{D}'}(j) = \frac{|a_{\mathcal{D}'}(j)|^{\frac{1}{j}}}{M} e^{i arg(a_{\mathcal{D}'}(j))}, \quad j \in \{1, ..., k\} \tag{3}$$

where N and M are respectively the number of cornerpoints of the considered persistence diagrams counted with multiplicity and k is the number of components of $\tilde{a}_{\mathcal{D}}$ and $\tilde{a}_{\mathcal{D}'}$ selected for the vector comparison.

Example 3.4. *In our experimentation, by applying (3) after one of our two transformations to the persistence diagrams, we obtain some values of complex vectors, related to the same component, with order of magnitude between 10^{-5} and 10^{-3}, with prevalence of 10^{-3}, or between 10^{-2} and 10^{-3}.*

4 Algorithms

The algorithms below resume the principal steps of our scheme for computing the coefficients associated to the persistence diagrams extrapolated from a database of objects db. In $Algorithm$ 1 the transformation F_1 (line 2) corresponds to one of the transformations T or R, while in $Algorithm$ 2 the transformation F_2 (line 4) corresponds to the transformation (3) defined above.

$Algorithm$ 1
Input: List A of cornerpoints of a persistence diagram $\mathcal{D}$,
$n = \max\limits_{\{A : A \in db\}} |A|$
Output: List B of complex numbers associated to $\mathcal{D}$

1:    **for each** $(u, v) \in A$
2:      **replace** (u, v) by $F_1(u, v)$
3:    **end for**
4:    **if** $|B| < n$
5:      **append** $n - |B|$ zeros to B
6:    **end if**

$Algorithm$ 2
Input: n, $B = list(z_1, ..., z_n)$ associated to $\mathcal{D}$, $k \in \{1, ..., n\}$
Output: Complex vector $\tilde{a}_\mathcal{D}^k = (\tilde{a}_\mathcal{D}(1), ..., \tilde{a}_\mathcal{D}(k))$ associated to $\mathcal{D}$

1:    **set** $\tilde{a}_\mathcal{D}^k = list()$
2:    **for** $j \in \{1, \ldots, k\}$
3:      **compute** $\tilde{a}_\mathcal{D} = \sum\limits_{1 \le i_1 < i_2 < ... < i_j \le n} z_{i_1} z_{i_2} ... z_{i_j}$
4:      **replace** $\tilde{a}_\mathcal{D}$ by $F_2(\tilde{a}_\mathcal{D})$
5:      **append** $\tilde{a}_\mathcal{D}$ to $\tilde{a}_\mathcal{D}^k$
6:    **end for**

After applying $Algorithm$ 1 and $Algorithm$ 2 to each persistence diagram belonging to the considered database, we can compute distances between the complex vectors obtained using the distance d previously defined in (2) and create the matrix of distances M_{dist} between the objects of the database by considering

$$V = \left\{ \tilde{a}_\mathcal{D}^k : \tilde{a}_\mathcal{D}^k \text{ complex vector associated with } \mathcal{D} \text{ for each } \mathcal{D} \in db \right\}$$

and defining $M_{dist}(i, j) = d(\tilde{a}_i^k, \tilde{a}_j^k)$, for $i \in \{1, ..., |V|\}$ and $j \in \{1, ..., |V|\}$.

Now let $N_{db} = |V|$ and $\mathcal{C}_{F_1}$ and $\mathcal{C}_{F_2}$ be the computational complexities of F_1 and F_2 respectively. Considering all persistence diagrams belonging to the database, the computational complexities of $Algorithm$ 1 and $Algorithm$ 2 are $\mathcal{C}_1 = O(n \cdot \mathcal{C}_{F_1} \cdot N_{db})$ and $\mathcal{C}_2 = O(k \cdot (k + n) \cdot \mathcal{C}_{F_2} \cdot N_{db})$ respectively.
The cost of $Algorithm$ 2 depends on how the formulas to compute the values of elementary symmetric functions of transformed cornerpoints (1) are implemented. Following [13], we implemented these formulas directly by computing the values of the first k elementary symmetric functions of transformed cornerpoints by means of Viète's formulas. Additionally, it is straightforward to verify that the cost of the transformations F_1 and F_2

is $O(1)$, so we can consider $\mathcal{C}_1 = O(n \cdot N_{db})$ and $\mathcal{C}_2 = O(k \cdot (k+n) \cdot N_{db})$ respectively. As regards, instead, the calculation of distances, the computational complexities related to it is $\mathcal{C}_d = O(k \cdot N_{db}^2)$. Therefore the whole computational cost of computing the matrix M_{dist} turns out to be $\mathcal{C} = O(k \cdot (k+n) \cdot N_{db} + k \cdot N_{db}^2)$, where k is the number of components of the complex vectors used to compare them and n is the maximum number of cornerpoints of persistence diagrams considered. In general, if k is chosen negligible compared to the value of n, the cost of computing M_{dist} becomes $O(n \cdot N_{db} + N_{db}^2)$.

Therefore, with an appropriate choice of the number k of computed values of elementary symmetric functions of transformed cornerpoints, our algorithm to compare persistence diagrams results to be cheaper than using the bottleneck distance. In fact, considering two persistence diagrams $\mathcal{D}$ and $\mathcal{D}'$ with N and M, respectively, the number of their cornerpoints, the cost of computing the bottleneck distance $d_B(\mathcal{D}, \mathcal{D}')$ is $\mathcal{C}_B = O(\max(N, M)^{\frac{3}{2}} \cdot log(\max(N, M)))$ [16]. Thus, in the case of the Bottleneck distance, the cost of computing M_{dist} is $O(n^{\frac{3}{2}} \cdot \log(n) \cdot N_{db}^2)$.

Moreover, consider the case a new object is to be compared with those of the database. This situation may occur both in the case the user wants to retrieve the first nearest neighbors of the new object, or in the case the new object needs to be classified on the base of its relative distances with respect to the objects of the database. This requires an almost instant computation of N_{db} distances to provide the user with a real-time answer, thus in these applications the algorithm based on the polynomial coefficients may be more suitable for this reason as well.

5 Experimental results

In this work we used the dataset PH^2 [17], a public database available online since the EMBC 2013 conference. The PH^2 database contains a total number of 200 melanocytic lesions, including 80 common nevi, 80 atypical nevi, and 40 melanomas. The dermoscopic images were obtained under the same conditions using a magnification of $20\times$. They are 8-bit RGB color images with a resolution of 768×560 pixels. In particular we tested the algorithm previously described considering the images of the database PH^2 and 19 features, 11 of which filtering functions obtained using 1-dimensional *persistent homology* theory, and 8 of them related to the *ABCDE* analysis.

The features considered are summarised in Table 1.

persistence features	ABCDE features
Light Intensity (LI)	Colour Histogram ($Histo$)
Blue (B)	Form Factor (FF)
Green (G)	Haralick's Circularity (HC)
Red (R)	Asymmetry ($Asym$)
Excess Blue ($ExcB$)	Ellipticity (Elt)
Excess Green ($ExcG$)	Eccentricity (Ecc)
Excess Red ($ExcR$)	Diameter ($Diam$)
Light Intensity 2 ($LI2$)	Colour Entropy ($Entr$)
Boundary Light Intensity (BLI)	
Boundary ($Bound$)	
Boundary 2 ($Bound2$)	

Table 1: Features.

The procedure used to compute the persistence diagrams from a dermoscopic image and the ABCDE features are the same as described in [7]. Of the persistent homology related features, the filtering functions B, G, R, $ExcB$, $ExcG$, $ExcR$ have already been described in [7]. Here, we recall that the persistence related features, i.e. the 1-dimensional persistence diagrams, are obtained from a segmented image as follows:

- computation of the global graph, whose nodes are all pixels of the segmentation mask, with the 6-connectivity notion;
- definition of the filtering functions on the nodes of the graphs;
- computation of the 1-dimensional PBN functions through the algorithm described in [7] and the references therein.

We also recall that, for computational reasons, prior to these operations the image and the corresponding segmentation mask needed to be rescaled into smaller size images by taking averages onto squared blocks of pixels. Since in [7] the use blocks of 4×4 pixels, provided good classification results for images of a similar size to those of the PH^2 dataset, here the same choice has been made. The global graph has also been used to compute the light intensity features, which are obtained from the R, G, B filtering functions as follows:

- $LI = (R + G + B)/3$;
- $LI2 = 255 - LI$.

The $LI2$ filtering function represents the inverse filtration of the LI filtering function. In a similar way, $Bound2$ is obtained by taking the inverse filtration of the distance from the barycenter on the boundary graph, described in [7]. Since the boundary graph has much fewer nodes than the boundary graph, no subdivision into blocks was needed. The BLI feature is obtained by considering the light intensity on the subgraph of the global graph composed by the pixels whose distance with respect to 4-connectivity is lower or equal to a certain constant C. This is obtained by taking one fifth of the of the maximal distance of the global graph pixels from the boundary graph. This choice allows to consider smaller blocks with respect to the global graph and to get a more detailed analysis of the texture of the skin lesion near the boundary: in this case, 2×2 block sizes have been used. Thus, for each method considered, 19 distance matrices have been computed, each containing the relative distances between the images of the database (of course, the main diagonal of these matrices is null):

- 11 matrices for the 1-dimensional filtering functions described above, using 1-dimensional matching distances between the corresponding PBNs;
- 8 matrices for the ABCDE features, using the same distances as in [7].

Three different methods have been considered for the computation of the 11 persistent homology distance matrices:

M1: Bottleneck distance with persistence diagrams;

M2: Algorithm 1 with transformation T and *Algorithm* 2 with $k = 10$;

M3: Algorithm 1 with transformation R and *Algorithm* 2 with $k = 10$.

Re-ordering the j-th column of the distance matrices, and considering only the first nearest neighbors, it is possible to compute the retrieval of

the j-th image of the database. From the retrieval, using the histopathological diagnosis of the images provided by the PH^2 database, it is possible to compute an automatical diagnosis and to compare it with the effective diagnosis of the query, therewith obtaining specificity and sensibility results of each method, with respect to a specific filtering function, on the entire database.

By considering linear combinations with non-negative coefficients of the distance matrices, it is possible to obtain global distance matrices, which in turn provide global retrievals and global sensitivity and specificity results.

All results will be reported in percentage terms of:

- *accuracy*: number of images classified correctly on the total images of the database;

- *sensitivity*: number of melanomas classified correctly on the total number of melanomas;

- *specificity*: number of nevi classified correctly on the total number of nevi.

5.1 Test 1

Initially we report in Table 2 the results related of each individual filtering function about test carried out considering database PH^2 and the three methods **M1**, **M2** and **M3**.

| | k=10 | | | | | | | | |
| | *bottleneck* | | | *transformation T* | | | *transformation R* | | |
	acc.	sens.	spec.	acc.	sens.	spec.	acc.	sens.	spec.
LI	73.5	62.5	76.3	77.5	80.0	76.9	72.5	60.0	75.6
B	75.5	52.5	81.3	76.5	70.0	78.1	72.0	52.5	76.9
G	74.5	60.0	78.1	73.5	72.5	73.8	71.0	70.0	71.3
R	79.5	75.0	80.6	74.5	77.5	73.8	72.5	65.0	74.4
ExcB	91.5	87.5	92.5	91.0	95.0	90.0	91.5	92.5	91.3
ExcG	79.5	90.0	76.9	81.5	87.5	80.0	80.0	87.5	78.1
ExcR	84.0	87.5	83.1	89.5	92.5	88.8	88.0	92.5	86.9
LI2	82.5	72.5	85.0	80.5	70.0	83.1	84.5	60.0	90.6
BLI	79.5	65.0	83.1	82.0	75.0	83.8	79.0	67.5	81.9
Bound	84.5	80.0	85.6	77.5	87.5	75.0	76.5	85.0	74.4
Buond2	82.0	67.5	85.6	77.5	85.0	75.6	74.0	87.5	70.6

Table 2: *Filtering function results, database PH^2.*

Remark 5.1. *In Table 2 we reported only the singular results about **persistence features** because the new algorithm does not concern the procedure by which the results relative to the individual **ABCDE features** are computed.*

Observing the results shown in Table 2 we can affirm that in this case the results obtained applying the new algorithm with $k = 10$, considering both transformation T and transformation R, are completely comparable to those obtained by using the bottleneck distance. Furthermore, among the methods used, there does not seem to be one that globally gives significantly better results than the others.

5.2 Test 2

In the previous tests we examined only the results related of each individual filtering function while now we are going to consider the set of all

the features reported in Table 1: **persistence features** (filtering functions) and **ABCDE features**. In particular we report in Table 3 the global results obtained considering database PH^2 and applying the new algorithm varying or the value of k such that $k \in \{5, 10, 20, 50\}$ or the transformation to take cornerpoints to complex number among T and R. Then we compare these results to each other and to those obtained by applying the bottleneck distance.

The global results reported below are obtained by weighted averages of the distances relative to the different filtering functions and features. The values attributed to these weights may be due to theoretical or experimental reasons and may also be changed during construction using a relevance feedback. In particular in this case the weights to be assigned to the different features are computed with a percentage optimization on all the 19 features considered.

		acc.	sens.	spec.
bottleneck		96.0	95.0	96.3
transformation T	k=5	96.0	90.0	97.5
	k=10	95.5	92.5	96.3
	k=20	97.5	95.0	98.1
	k=50	97.0	92.5	98.1
transformation R	k=5	97.5	95.0	98.1
	k=10	97.5	97.5	97.5
	k=20	97.0	92.5	98.1
	k=50	97.0	95.0	97.5

Table 3: *Global results, database PH^2.*

Observing the results shown in Table 3 we can affirm that in this case also the global results obtained applying the new algorithm with $k \in \{5, 10, 20, 50\}$, both considering transformation T and transformation R, are completely comparable to those obtained by using the bottleneck distance. Furthermore, there does not seem to be a significant difference in the results obtained when k changes in $\{5, 10, 20, 50\}$ therefore it is advisable to consider a value of k small to avoid raising the execution time of the algorithm too much.

Unfortunately, we could not test the algorithms of the considered methods on the same platform, so the computational times are not directly comparable. Nevertheless, it is significant that the computational time required for computing a distance matrix with method **M1** was approximately 10 times higher than that required by the methods **M2** and **M3**.

6 Further developments and conclusions

The strategy of using symmetric functions of warped persistence diagrams is an interesting way to solve the problem of the high computational cost of the bottleneck distance, keeping the persistent homology framework intact and allowing scalability on large databases.

The warping plane transformations are of interest on their own, as they map the noisy cornerpoints near the diagonal to a neighbourhood of the origin, simplifying the geometry of the plane where cornerpoints lie: for example, we believe that defining an appropriate matching distance

between the transformed diagrams could lead to interesting results as well, and we propose to investigate this in future researches.

The symmetric functions, unfortunately, appear to be highly dependent on the number of cornerpoints, thus the implementation of this method requires to filter the noise in some way. The solution proposed in Section 3, based on taking the j-th square root of the j-th symmetric function, works well in the experiments, but requires further investigation as it only seems to attenuate the problem.

Overall, the proposed method may serve to fill the gap between simulation and real practice on large scale databases for persistent homology based applications, but requires to be validated by further experiments and possibly improved by theoretical adjustments.

References

[1] Adcock Aaron, Rubin Daniel, Carlsson Gunnar. Classification of hepatic lesions using the matching metric. *Computer Vision and Image Understanding.* 2014;121:36–42.

[2] Chung Moo K., Bubenik Peter, Kim Peter T.. Persistence diagrams of cortical surface data. In: :386–397Springer; 2009.

[3] Bendich Paul, Marron J. S., Miller Ezra, Pieloch Alex, Skwerer Sean. Persistent homology analysis of brain artery trees. *Annals of Applied Statistics.* 2016;10(1):198–218.

[4] d'Amico Michele, Ferri Massimo, Stanganelli Ignazio. Qualitative asymmetry measure for melanoma detection. In: :1155–1158; 2004.

[5] Ferri Massimo, Stanganelli Ignazio. Size Functions for the Morphological Analysis of Melanocytic Lesions. *Journal of Biomedical Imaging.* 2010;2010:5:1–5:5.

[6] Stanganelli Ignazio, Brucale Aldo, Calori Luigi, et al. Computer-aided diagnosis of melanocytic lesions. *Anticancer Research.* 2005;25(6):4577–4582.

[7] Ferri Massimo, Tomba Ivan, Visotti Andrea, Stanganelli Ignazio. A feasibility study for a persistent homology based k-Nearest Neighbor search algorithm in melanoma detection. *J. Math. Imaging Vis..* 2017;57(3):324–339.

[8] Donatini Pietro, Frosini Patrizio. Natural pseudodistances between closed manifolds. *Forum Mathematicum.* 2004;16(5):695–715.

[9] Frosini Patrizio, Landi Claudia. Size functions and formal series. *Applicable Algebra in Engineering, Communications and Computing.* 2001;12(4):327–349.

[10] d'Amico Michele, Frosini Patrizio, Landi Claudia. Natural pseudo-distance and optimal matching between reduced size functions. *Acta Appl. Math..* 2010;109:527–554.

[11] d'Amico Michele, Frosini Patrizio, Landi Claudia. Using matching distance in Size Theory: A survey. *Int. J. Imag. Syst. Tech..* 2006;16(5):154–161.

[12] Ferri Massimo, Landi Claudia. Representing size functions by complex polynomials. *Proc. Math. Met. in Pattern Recognition.* 1999;9:16–19.

[13] Di Fabio Barbara, Ferri Massimo. Comparing persistence diagrams through complex vectors. In: Murino V., Puppo E., eds. *Image Analysis and Processing - ICIAP 2015 Part I*, LNCS 9279, Springer 2015 (pp. 294–305).

[14] Edelsbrunner Herbert, Harer John. Persistent homology - a survey. In: Contemp. Math., vol. 453: Providence, RI: Amer. Math. Soc. 2008 (pp. 257–282).

[15] Edelsbrunner Herbert, Harer John. *Computational Topology: An Introduction*. American Mathematical Society; 2009.

[16] Kerber Michael, Morozov Dmitriy, Nigmetov Arnur. Geometry helps in Comparing Persistence Diagrams. In: ; 2016. Algorithm Engineering and Experiments (ALENEX).

[17] Mendonca Teresa, Ferreira Pedro M., Marques Jorge S., Marcal André R. S., Rozeira Jorge. PH2-A dermoscopic image database for research and benchmarking. In: Soc. Proc. IEEE Eng. Med. Biol., ed. *35th Annu. Int. Conf. (EMBC)*, 2013 (pp. 5432–5440).