Application of Persistent Homology in Signal and Image Denoising

Dissertation
zur Erlangung des mathematisch-naturwissenschaftlichen
Doktorgrades
Doctor rerum naturalium
der Georg-August-Universität Göttingen

im Promotionsstudiengang *Mathematical Sciences*
der Georg-August University School of Science (GAUSS)

vorgelegt von
Yi Zheng
aus Hanyuan, Ya'an, China

Göttingen 2015

Betreuungsausschuss

- Prof. Dr. Gerlind Plonka-Hoch, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen
- Prof. Dr. Max Wardetzky, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Mitglieder der Prüfungkommission

- Referentin:
  Prof. Dr. Gerlind Plonka-Hoch, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen
- Korreferent:
  Prof. Dr. Armin Iske, Fachbereich Mathematik, Universität Hamburg

Weitere Mitglieder der Prüfungkommission

- Prof. Dr. Stephan Huckemann, Institut für Mathematische Stochastik, Georg-August-Universität Göttingen
- Prof. Dr. Hartje Kriete, Mathematisches Institut, Georg-August-Universität Göttingen
- Prof. Dr. Gert Lube, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen
- Prof. Dr. Max Wardetzky, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Tag der mündlichen Prüfung: 12.06.2015

Acknowledgements

First of all, I would like to thank my supervisor Prof. Dr. Gerlind Plonka-Hoch for her insightful scientific ideas and exhaustless support during the preparation of this thesis and my PhD study period. Especially she introduced me to the very interesting research direction of persistent homology and its applications.

Furthermore, I express my gratitude to Prof. Dr. Max Wardetzky, who accompanied me as co-advisor, and to Prof. Dr. Armin Iske, who was so kind to undertake the task of being the co-referee of my thesis.

Besides, I would like to thank the complete group "Mathematical Signal and Image Processing" for a good working atmosphere. Working in the group has been always my pleasure. Especially, I would like to mention my former colleague Dr. Thomas Peter and Dr. Marius Wischerhoff for their helps in technical and German language related things.

Furthermore, I would like to express my gratitude to the financial support from the projects of the working group and the workshops of the "Research Training Group 1023" in which I was an associated member. The travel opportunities have been helpful for broadening horizons in every aspect.

Last but not least I want to thank my family and friends. They forgave me for all the times I was away from China and ignoring them. I am deeply grateful to have Jing Yang at my side always giving me support and appreciation.

Contents

1 Introduction

Motivated by recent developments in topological persistence for assessment of the importance of features in data sets, we study the ideas of persistence homology for one-dimensional digital signals and its application in signal and image denoising. The notions of persistence homology and persistence pairs were introduced in [24] for measuring the topological complexity of point sets in $\mathbb{R}^3$. Persistence pairs and corresponding persistence diagrams are well suited to quantify the topological significance of data structures and to develop a formalism for topological simplification [9, 14–16, 42]. In case of one-dimensional digital signals the idea of topological persistence boils down to the problem of pairing suitable local minima and maxima of the signal. Considering the persistence pairs and the corresponding persistences not only for the signal f but also for $-f$, we propose the new notion of *persistence distance* of f. Transferring from f to $-f$ switches the roles of the sets of local minima and local maxima of f. A comparison of the persistence pairs obtained for f and for $-f$ already provides us with an important categorization tool. Persistence pairs occurring for both, f and $-f$, are less significant than those occurring only once, for f or for $-f$.

We show that the persistence distance has a lot of favorable properties. Particularly, we show that the persistence distance is very closely related to the discrete total variation of f. This relation motivates us to employ the new notion of persistence distance for signal and image denoising.

Let u be a finite digital signal that is corrupted with white noise, i.e., we have given the data

$$f(x_j) = u(x_j) + n(x_j), \qquad j = 0, \dots, N$$

for a partition $a = x_0 < x_1 < \ldots < x_N = b$, where n has zero mean and (unknown) deviation σ. Using the celebrated discrete ROF-model, a reconstruction of u can be obtained as the minimizer of the functional

$$J(u) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{\ell=0}^{N-1} |u(x_{\ell+1}) - u(x_\ell)|,$$

where the second term denotes the discrete total variation of u. We propose in this thesis to replace this second term using the persistence distance of u that inherits the topological properties of the signal u. In contrast to the discrete total variation, the persistences $|u(x) - u(\tilde{x})|$ corresponding to persistence pairs $(x, \tilde{x})$ contain direct structural information on u. Small persistences $|u(x) - u(\tilde{x})|$ being related to pairs with small

distances $|x - \tilde{x}|$ correspond to oscillatory behavior like noise while large persistences describe important features of the signal. Therefore we will propose to apply suitably weighted persistences and show that the obtained new functional can be also regarded as a weighted ROF-functional of the form

$$J_w(u) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{\ell=0}^{N-1} w_\ell(u)\, |u(x_{\ell+1}) - u(x_\ell)|,$$

where the weights $w_\ell(u)$ depend on local chains of persistence pairs. In particular, the weights are taken in a way such that the denoised signal obtained by minimization of $J_w(u)$ preserves the essential peaks (discontinuities) of u well and yields good denoising performance at smooth subregions of u. By simply treating the rows and columns of an image as vectors, we can apply the one-dimensional persistence-weighted ROF scheme to image denoising. Furthermore, a two dimensional persistence-weighted ROF model can be established where the weights are determined according to the one dimensional persistence information.

Topological persistence and its application to extract important topological features from data has been extensively studied within the last years, see e.g. [9, 14–16, 24, 42] and references therein. In [15], it has been shown that persistence diagrams of real-valued functions are stable with regard to noise, i.e., for two functions f and g with corresponding persistence diagrams $D(f)$ and $D(g)$ one finds

$$d_B(D(f),\, D(g)) \leq \|f - g\|_\infty,$$

where d_B denotes the bottleneck distance and $\|\cdot\|_\infty$ the L_∞-norm. In [14], the p-norm of the persistence diagram and its changes under diffusion of f using a convolution with a Gaussian kernel with enlarging parameter is studied.

Though, for application of persistence in signal denoising we are only aware of the results in [5], where topological denoising methods have been proposed employing a persistence-based simplification and a so-called filling-based simplification of the signal. The latter method mimics the construction of cancelation of persistence pairs, where instead of the filling level the filling volume is increased, thereby taking into account both, the distance between points in one persistence pair and the distance of corresponding function values. This last approach is slightly related to the watershed transform, a frequently used tool in image segmentation, see e.g. [6, 36] and references therein. In a recent preprint [4], the problem of estimating the number of local maxima of a signal given by noisy measurements has been considered using persistence barcodes and Kolmogorov signatures.

Regarding weighted TV-minimization for signal and image denoising we refer to [1, 21, 26, 38, 39] etc. In the continuous setting, the adaptive TV denoising approaches usually consider the minimization functionals of the form

$$\frac{1}{2}\int_{\Omega}|u(x)-f(x)|^2dx+\int_{\Omega}\alpha(x)|\mathcal{D}u(x)|dx$$

over $u \in BV(\Omega)$, the space of functions of bounded variation, where $\int_{\Omega}|\mathcal{D}u(x)|dx$ denotes the total variation of u. The parameter $\alpha(x)$ is adaptively chosen depending on geometric properties of signal features [38, 39], and can be further improved using noise statistics and robust adjustment [21].

This thesis is organized as follows.

Chapter 2 gives a short introduction to persistent homology. Based on simplicial complexes, the necessary background concepts of homology are introduced. At the end of Chapter 2, we explain the connection of persistent homology for simplicial complexes and triangulation of topological spaces, and notice that one-dimensional piecewise linear functions can itself be viewed as simplicial complexes. Particularly, the procedure of computing persistence using lower-star filtration is closely related to the notion of persistence for Morse functions.

In Chapter 3, we recall the discrete ROF model, based on the celebrated Rudin-Osher-Fatemi approach [37] for signal denoising, and present iterative numerical procedures to minimize the discrete ROF functional. Further, we summarize the properties of the discrete total variation in the one-dimensional case.

In Chapter 4 we introduce our new concept of persistence distance based on persistence pairs and corresponding persistences of function values for one-dimensional signals on an interval. We show some favorable properties of the persistence distance and the close relation between persistence distance and discrete total variation for one-dimensional piecewise linear functions.

In Chapter 5, we apply the new notion of persistence distance to signal denoising. We propose a new weighted functional, where the regularization term is based on the persistence of the signal. We show that this functional can also be regarded as a weighted ROF functional. Furthermore, we present a numerical algorithm for the proposed weighted TV minimization based on persistence. At the end of this chapter, some denoising experiments are shown.

In Chapter 6, we apply the one-dimensional persistence distance to image denoising in two ways. One is based on a separable scheme. The second approach is based on a two-dimensional ROF model which can be understood as a weighted 2D ROF model. As before, image denosing experiments are given, where we compare our proposed method with the four-pixel scheme in [41], shearlet shrinkage [30], and with the BM3D denoising method [18].

The results obtained in Chapters 4 and 5 of this thesis are summarized in a preprint [35] that has been submitted for publication.

2 A short introduction to persistent homology

This chapter introduces necessary background knowledge on persistent homology which forms a concrete basement for our further discussion.

The first section shows how some topological properties can be characterized algebraically using the notion of of homology groups. Then, filtration is introduced for simplicial complexes and for real-valued functions. Based on filtration, we can define persistence of topological features. With the help of so-called persistence pairs we can describe how long a topological property, more specifically classes of a homology group (connectivity, holes etc.), sustains during a filtration. Depending on the application, the filtration process and the corresponding definition of persistence of certain topological features can be suitably defined. We will concentrate on lower-star filtration for one-dimensional piecewise linear functions. As we will see, the computation of the persistence of topological properties does not require the complete knowledge of the homology groups. Instead, we are interested in the change of the rank of the homology groups, the so-called Betti numbers. A powerful tool for computing the persistence using boundary matrices is also introduced.

Being interested in the application of persistent homology to special one- and two-dimensional functions, we concentrate in later chapters on computing of persistence pairs and corresponding persistence diagrams in a way that does not involve the theory on homology groups. However, we will refer to the close connection to these topological notions.

2.1 Topological Homology

2.1.1 Simplicial Complex

Simplicial complexes form the fundament of the homology theory. Intuitively, they can be constructed using simplices, i.e., points, line segments, triangles, tetrahedra etc., [32].

Definition 2.1:
A set $\{a_0, \ldots, a_n\}$ of points in $\mathbb{R}^N$ is called **geometrically independent** if for any (real) scalars t_i, the equations

$$\sum_{i=0}^{n} t_i = 0 \quad \text{and} \quad \sum_{i=0}^{n} t_i a_i = 0$$

imply that $t_0 = t_1 = \ldots = t_n = 0$.

Obviously, $\{a_0, \ldots, a_n\}$ are geometrically independent if and only if the vectors $a_1 - a_0, \ldots, a_n - a_0$ are linearly independent.

Definition 2.2:
Let $\{a_0, \ldots, a_n\}$ be a geometrically independent set in $\mathbb{R}^N$. We define the n**-simplex** σ spanned by $a_0, \ldots, a_n$ to be the set of all points x of $\mathbb{R}^N$ such that

$$x = \sum_{i=0}^{n} t_i a_i \quad \text{and} \quad \sum_{i=0}^{n} t_i = 1$$

and $t_i \geq 0$ for all i.

We denote the simplex as $\langle a_0, \ldots, a_n \rangle$. The points $a_0, \ldots, a_n$ that span σ are called the **vertices** of σ; the number n is called the **dimension** of σ. Any simplex spanned by a subset of $\{a_0, \ldots, a_n\}$ is called a **face** of σ.

Definition 2.3:
A **simplicial complex** (abbreviation: complex) K in $\mathbb{R}^N$ is a collection of simplices in $\mathbb{R}^N$ such that
(1) Every face of a simplex of K is in K.
(2) The intersection of any two simplices in K is a face of each of them.

2.1.2 Homology Group of a simplicial complex

Definition 2.4:
Let K be a simplicial complex. We denote the set generated by p-simplices of K over the binary field as $C_p = C_p(K)$. It consists of all p**-chains** defined as

$$c = \sum_j \gamma_j \sigma_j,$$

where γ_j are 0 or 1 and σ_j are p-simplices in K.

We can add two p-chains together componentwise. For example, if $c_0 = \sum_j \delta_j \sigma_j$ is a second p-chain, its addition with c is $c + c_0 = \sum_j (\gamma_j + \delta_j)\sigma_j$, where the coefficients are integers modulo 2. The p-chains in C_p with endowed addition over the binary field as above form a group which we call the **group of** p**-chains** denoted as $(C_p, +)$.

The set of p-chains in C_p can also be interpreted as follows: We consider all subsets of p-simplices in $C_p(K)$, and the addition of two sets is then equivalent with the symmetric difference of these sets.

Homology wants to distinguish different types of chains. For that purpose, we first need the definition of the boundary of a p-simplex which is the sum of all its $(p-1)$-faces. Then we can define the boundary of a p-chain by linearity according to the definition.

Definition 2.5:
Let $\sigma = \langle a_0, \ldots, a_n \rangle$ be a p-simplex. We define its boundary as

$$\partial_p \sigma = \sum_{i=0}^{n} (-1)^i \langle a_0, \cdots, \hat{a}_i, \cdots, a_n \rangle = \sum_{i=0}^{n} \langle a_0, \cdots, \hat{a}_i, \cdots, a_n \rangle,$$

where $\hat{a}_i$ means that a_i is dropped. The operator ∂_p is called **boundary operator**.

Observe that $(-1)^i = 1$ in the binary field, such that the boundary operator can be simplified as given above. We can define now the boundary of a p-chain by its linearity.

Definition 2.6:
Let $c = \sum_j \gamma_j \sigma_j$ be a p-chain. We define its boundary as the linear combination of boundaries of its p-simplices

$$\partial_p c = \sum_j \gamma_j \partial_p \sigma_j.$$

The boundary operator ∂_p maps a p-chain group C_p to a $(p-1)$-chain group C_{p-1}. Thus, the boundary operator connects different dimensional chain groups in the following **chain complex**

$$\cdots \xrightarrow{\partial_{p+2}} C_{p+1} \xrightarrow{\partial_{p+1}} C_p \xrightarrow{\partial_p} C_{p-1} \xrightarrow{\partial_{p-1}} \cdots.$$

Moreover, $\partial_p : C_p \longrightarrow C_{p-1}$ is a homomorphism since $\partial_p(c + c_0) = \partial_p(c) + \partial_p(c_0)$ holds.

Before we can define a homology group, we need to distinguish two special kinds of chains. A **p-cycle** is a p-chain with empty boundary, i.e., $\partial_p c = 0$. Since the operator $\partial_p c$ commutes with addition of p-chains, all p-cycles form a subgroup Z_p of C_p, denoted by $Z_p \leqslant C_p$. In other words, $Z_p = \ker \partial_p$.

A **p-boundary** c is a p-chain which is the boundary of a $(p+1)$-chain d, i.e., $c = \partial d$. Similarly, all p-boundaries form a subgroup B_p of the chain group C_p. Thus, we have $B_p \leqslant C_p$. More specifically, $B_p = \operatorname{Im} \partial_{p+1}$.

The fundamental property for the homology procedure is the following.

Lemma 2.7:
For all integers p and every $(p+1)$-chain d, we have $\partial_p \partial_{p+1} d = 0$.

This lemma tells us that a p-boundary is necessarily a p-cycle. Using group language and the notation above, a p-boundary group B_p is a subgroup of the p-cycle group Z_p, i.e. $B_p \leqslant Z_p$.

Thus, we have now that $B_p \leqslant Z_p \leqslant C_p$. A p-boundary is necessarily a p-cycle, but the reverse does not hold, i.e., a p-cycle is not necessarily a p-boundary. To distinguish those non-boundary p-cycles, the notion of homology group is a powerful tool as follows.

Definition 2.8:
Using the above notations, we define the p-homology group as $H_p := Z_p/B_p$.

Each element in H_p, which we call a class of H_p, is a collection of p-chains obtained by adding p-boundaries from B_p to a given p-cycle, $c + B_p$ with $c \in Z_p$. We use c as the **representative** of this class. Since we had considered the p-simplices over the binary field, the rank of the considered groups Z_p, B_p and H_p is the base 2 logarithm of their cardinality, i.e., we have $\operatorname{rank}(H_p) = \log_2(\operatorname{card} H_p)$. The rank of the p-homology group, $\operatorname{rank}(H_p)$, is called the p-**Betti number** of Z_p. We have the following relation,

$$\operatorname{rank}(H_p) = \operatorname{rank}(Z_p) - \operatorname{rank}(B_p).$$

Example 1:
Let K be given as $K = \{\langle a_1\rangle, \langle a_2\rangle, \langle a_3\rangle, \langle a_1, a_2\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle\} \in \mathbb{R}^2$. Then we obtain the following groups. For $p = 0$,

$$\begin{aligned}
C_0(K) &= \{\emptyset, \langle a_1\rangle, \langle a_2\rangle, \langle a_3\rangle, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle, \langle a_1\rangle + \langle a_2\rangle + \langle a_3\rangle\},\\
Z_0(K) &= C_0(K),\\
B_0(K) &= \{\emptyset, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle\},
\end{aligned}$$

and hence rank $H_0(K) = 1$. The 0-Betti number measures the number of unconnected parts of the complex K. For $p = 1$, we find

$$\begin{aligned}
C_1(K) = \{&\emptyset, \langle a_1, a_2\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle, \langle a_1, a_2\rangle + \langle a_2, a_3\rangle,\\
&\langle a_1, a_3\rangle + \langle a_2, a_3\rangle, \langle a_1, a_2\rangle + \langle a_2, a_3\rangle + \langle a_1, a_3\rangle\},\\
Z_1(K) = \{&\emptyset, \langle a_1, a_2\rangle + \langle a_2, a_3\rangle + \langle a_1, a_3\rangle\},\\
B_1(K) = \{&\emptyset.
\end{aligned}$$

Indeed, K possesses one non-boundary 1-cycle.

2.2 Persistent homology

The homology group describes properties of a simplicial complex statically, but it cannot quantify the topological changes in a certain kind of evolvement. In order to investigate the properties of homology groups dynamically the notion of filtration has been introduced in [24]. Filtration considers a sequence of subcomplexes of the simplicial complex K. By examining the corresponding homology groups at each stage, we can describe how long certain properties (classes) of the complex survive in the sequence of homology groups during the filtration. Usually, we do not need to know the exact structure of every homology group in the filtration. What we need to know is how the number of homology classes (the Betti number) changes during the filtration. Hence, a closer look to a p-simplex that creates a new class and to a $(p+1)$-simplex that destroys an existing class is sufficient to figure out the topological changes. The main procedure consists in recording, when a new homology class appears ("is born") at a certain stage, and when it becomes trivial or merges with another class. This information can be collected by a pairing procedure. An algorithm for pairing based on boundary matrices is introduced at end of this subsection.

2.2.1 Filtration of a simplicial complex and persistent homology

The filtration is based on ordering of subcomplexes K_i of a simplicial complex K such that

$$\emptyset = K_0 \subseteq K_1 \subseteq \ldots \subseteq K_n = K.$$

see [24].

Definition 2.9 (Filtration and filter)**:**
The sequence of subcomplexes $\{K_0, K_1, \cdots, K_n\}$ is called a **filtration** of the complex K, where $K_0 = \emptyset$. The corresponding sequence of sets $\{c_0, \cdots, c_{n-1}\}$ with the property that $K_{j+1} = K_j \cup c_j$ for $j = 0, \ldots, n-1$ is called a **filter**. If at each stage j, the set c_j consists of only one simplex σ_j, we call the filtration **complete**.

By definition, for a simplex $\sigma \in K_i$ it follows that $\sigma \in K_j$ for $j = i, \ldots, n$. Let now the **birth time** $\alpha(\sigma)$ of the simplex σ in the filtration be the smalls index i such that $\sigma \in K_\alpha$ iff $\alpha \geq \alpha(\sigma)$.

Example 2:
We can construct a filtration of the simplicial complex in Fig. 2.1 by using the following filter: $c_0 = \langle a_1 \rangle$, $c_1 = \langle a_2 \rangle$, $c_2 = \langle a_3 \rangle$, $c_3 = \langle a_2, a_3 \rangle$, $c_4 = \langle a_1, a_3 \rangle$, $c_5 = \langle a_4 \rangle$ and $c_6 = \langle a_2, a_4 \rangle$. Its corresponding filtration can be easily obtained as $K_i = \bigcup_{k=0}^{i-1} c_k$ for $i = 1, \cdots, 7$ and $K_0 = \emptyset$. In this filtration, the filter contains only simplices $c_i = \sigma_i$ at each stage, i.e., it is complete.

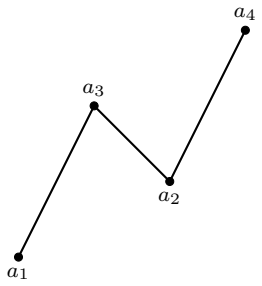


Fig. 2.1: 1D Filtration example.

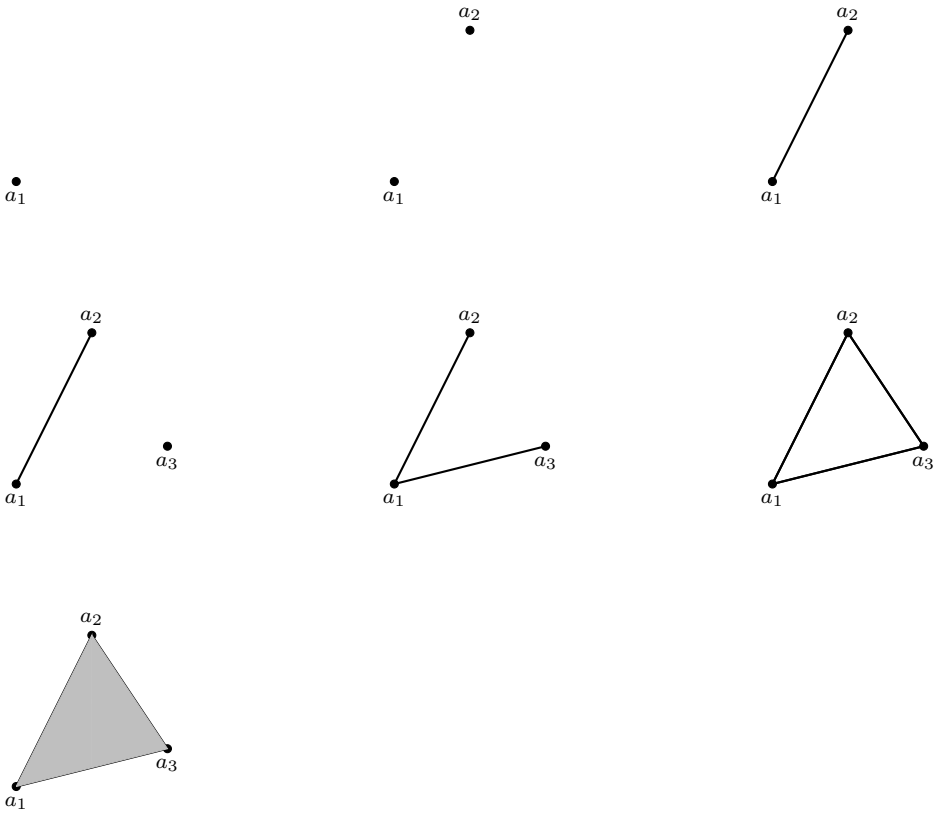


Fig. 2.2: 2D simplex Filtration example.

Example 3:
The simplicial complex K in Fig. 2.2 can be filtered by the sequence $c_0 = \sigma_0 = \langle a_1 \rangle$, $c_1 = \sigma_1 = \langle a_2 \rangle$, $c_2 = \sigma_2 = \langle a_1, a_2 \rangle$, $c_3 = \sigma_3 = \langle a_3 \rangle$, $c_4 = \sigma_4 = \langle a_1, a_3 \rangle$, $c_5 = \sigma_5 = \langle a_2, a_3 \rangle$ and $c_6 = \sigma_6 = \langle a_1, a_2, a_3 \rangle$. Its corresponding filtration is given by $K_i = \bigcup_{k=0}^{i-1} \sigma_k$ for $i = 1, \cdots, 6$ and $K_0 = \emptyset$. Let us further examine the boundary groups, cycle groups and the rank of the homology group at each stage K_j.

For $K_1 = \{\langle a_1 \rangle\}$ we have $C_0(K_1) = Z_0(K_1) = \{\emptyset, \langle a_1 \rangle\}$, $B_0(K_1) = \emptyset$ and hence $\text{rank}(H_0(K_1)) = 1$.

For $K_2 = \{\langle a_1 \rangle, \langle a_2 \rangle\}$ it follows that $C_0(K_2) = Z_0(K_2) = \{\emptyset, \langle a_1 \rangle, \langle a_2 \rangle, \langle a_1 \rangle + \langle a_2 \rangle\}$, $B_0(K_2) = \emptyset$ and thus $\text{rank}(H_0(K_2)) = 2$.

For $K_3 = \{\langle a_1 \rangle, \langle a_2 \rangle, \langle a_1, a_2 \rangle\}$ it follows that

$$\begin{aligned} C_0(K_3) = Z_0(K_3) &= \{\emptyset, \langle a_1 \rangle, \langle a_2 \rangle, \langle a_1 \rangle + \langle a_2 \rangle\}, \\ B_0(K_3) &= \{\emptyset, \langle a_1 \rangle + \langle a_2 \rangle = \partial \langle a_1, a_2 \rangle\}, \end{aligned}$$

and thus $\text{rank}(H_0(K_3)) = 1$. We can now consider also the twodimensional Betti number, since the simplex $\langle a_1, a_2 \rangle$ came into the filtration. We have $C_1(K_3) = \{\emptyset, \langle a_1, a_2 \rangle\}$, $Z_1(K_3) = \{\emptyset\}$ and $B_1(K_3) = \{\emptyset\}$. It follows $\text{rank}(H_1(K_3)) = 0$, i.e., there is no "hole" in K_3.

For $K_4 = \{\langle a_1 \rangle, \langle a_2 \rangle, \langle a_1, a_2 \rangle, \langle a_3 \rangle\}$ it follows that

$$\begin{aligned} C_0(K_4) = Z_0(K_4) &= \{\emptyset, \langle a_1 \rangle, \langle a_2 \rangle, \langle a_3 \rangle, \langle a_1 \rangle + \langle a_2 \rangle, \langle a_1 \rangle + \langle a_3 \rangle, \langle a_2 \rangle + \langle a_3 \rangle, \\ &\quad \langle a_1 \rangle + \langle a_2 \rangle + \langle a_3 \rangle\}, \\ B_0(K_4) &= \{\emptyset, \langle a_1 \rangle + \langle a_2 \rangle = \partial \langle a_1, a_2 \rangle\} \end{aligned}$$

and thus $\text{rank}(H_0(K_4)) = 2$. For dimension one, we have $C_1(K_4) = \{\emptyset, \langle a_1, a_2 \rangle\}$, $Z_1(K_4) = \{\emptyset\}$ and $B_1(K_4) = \{\emptyset\}$. It follows $\text{rank}(H_1(K_4)) = 0$, i.e., there is still no "hole" in K_4.

For $K_5 = \{\langle a_1 \rangle, \langle a_2 \rangle, \langle a_1, a_2 \rangle, \langle a_3 \rangle, \langle a_1, a_3 \rangle\}$ it follows that

$$\begin{aligned} C_0(K_5) = Z_0(K_5) &= \{\emptyset, \langle a_1 \rangle, \langle a_2 \rangle, \langle a_3 \rangle, \langle a_1 \rangle + \langle a_2 \rangle, \langle a_1 \rangle + \langle a_3 \rangle, \langle a_2 \rangle + \langle a_3 \rangle, \\ &\quad \langle a_1 \rangle + \langle a_2 \rangle + \langle a_3 \rangle\} \\ B_0(K_5) &= \{\emptyset, \langle a_1 \rangle + \langle a_2 \rangle = \partial \langle a_1, a_2 \rangle, \langle a_1 \rangle + \langle a_3 \rangle = \partial \langle a_1, a_3 \rangle, \\ &\quad \langle a_2 \rangle + \langle a_3 \rangle = \partial (\langle a_1, a_2 \rangle + \langle a_1, a_3 \rangle)\} \end{aligned}$$

and thus $\text{rank}(H_0(K_5)) = 1$. For dimension one, we have

$$\begin{aligned} C_1(K_5) &= \{\emptyset, \langle a_1, a_2 \rangle, \langle a_1, a_3 \rangle, \langle a_1, a_2 \rangle + \langle a_1, a_3 \rangle\}, \\ Z_1(K_5) &= \{\emptyset\}, \\ B_1(K_5) &= \{\emptyset\}. \end{aligned}$$

It follows $\text{rank}(H_1(K_5)) = 0$, i.e., there is still no "hole" in K_5.

For $K_6 = \{\langle a_1\rangle, \langle a_2\rangle, \langle a_1, a_2\rangle, \langle a_3\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle\}$ it follows that

$$\begin{aligned} C_0(K_6) &= Z_0(K_6) = \{\emptyset, \langle a_1\rangle, \langle a_2\rangle, \langle a_3\rangle, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle, \\ &\quad \langle a_1\rangle + \langle a_2\rangle + \langle a_3\rangle\}, \\ B_0(K_6) &= \{\emptyset, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle = \partial(\langle a_2, a_3\rangle)\} \end{aligned}$$

and thus $\text{rank}(H_0(K_6)) = 1$. For dimension one, we have

$$\begin{aligned} C_1(K_6) &= \{\emptyset, \langle a_1, a_2\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle, \langle a_1, a_2\rangle + \langle a_2, a_3\rangle, \\ &\quad \langle a_2, a_3\rangle + \langle a_1, a_3\rangle, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle + \langle a_2, a_3\rangle\}, \\ Z_1(K_6) &= \{\emptyset, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle + \langle a_2, a_3\rangle\} \end{aligned}$$

and $B_1(K_6) = \{\emptyset\}$. It follows $\text{rank}(H_1(K_6)) = 1$, i.e., we get one "hole" in K_6.

Finally, for $K_7 = \{\langle a_1\rangle, \langle a_2\rangle, \langle a_1, a_2\rangle, \langle a_3\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle, \langle a_1, a_2, a_3\rangle\}$ it follows that

$$\begin{aligned} C_0(K_7) = Z_0(K_7) &= \{\emptyset, \langle a_1\rangle, \langle a_2\rangle, \langle a_3\rangle, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle, \\ &\quad \langle a_1\rangle + \langle a_2\rangle + \langle a_3\rangle\}, \\ B_0(K_7) &= \{\emptyset, \langle a_1\rangle + \langle a_2\rangle, \langle a_1\rangle + \langle a_3\rangle, \langle a_2\rangle + \langle a_3\rangle = \partial(\langle a_2, a_3\rangle)\} \end{aligned}$$

and thus $\text{rank}(H_0(K_7)) = 1$. For dimension one, we have

$$\begin{aligned} C_1(K_7) &= \{\emptyset, \langle a_1, a_2\rangle, \langle a_1, a_3\rangle, \langle a_2, a_3\rangle, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle, \langle a_1, a_2\rangle + \langle a_2, a_3\rangle, \\ &\quad \langle a_2, a_3\rangle + \langle a_1, a_3\rangle, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle + \langle a_2, a_3\rangle\}, \\ Z_1(K_7) &= \{\emptyset, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle + \langle a_2, a_3\rangle\} \\ B_1(K_7) &= \{\emptyset, \langle a_1, a_2\rangle + \langle a_1, a_3\rangle + \langle a_2, a_3\rangle = \partial(\langle a_1, a_2, a_3\rangle)\}. \end{aligned}$$

It follows $\text{rank}(H_1(K_7)) = 0$ which indicates that the "hole" in K_6 disappears in K_7. This phenomenon will be elaborated in Example 5. Furthermore, also dimension two can be considered in K_7, and we find $C_2(K_7) = \{\emptyset, \langle a_1, a_2, a_3\rangle\}$, $Z_2(K_7) = B_2(K_7) = \{\emptyset\}$ and thus $\text{rank}(H_2(K_7)) = 0$.

For our purposes, we also need description tools of a "neighborhood" since a simplicial complex can be viewed as the triangulation of a topological space (see Subsection 2.2.5 for more details). **Star** and **link** are the analogy concepts for describing neighborhood in a simplicial complex.

Definition 2.10 (Star and Link [23, 25])**:**
For a set of vertices U of the simplicial complex K, we define its **star** w.r.t K as the set of simplices that have at least one vertex in U, and its **link** is the set of faces of simplices in star that do not also belong to star:

$$\text{St } U := \{\sigma \in K | \exists u \in U, u \in \sigma\},$$
$$\text{Lk } U := \{\tau \in K | \tau \subseteq \sigma \in \text{St } U, \tau \notin \text{St } U\}.$$

Remark:
If U consists in only one vertex $\langle u \rangle$, then we simply write St u and Lk u.

Example 4:
Considering Example 3 with $U = \{\langle a_2 \rangle\}$, we find

$$\text{St } a_2 = \{\langle a_2 \rangle, \langle a_1, a_2 \rangle, \langle a_2, a_3 \rangle, \langle a_1, a_2, a_3 \rangle\}$$
$$\text{Lk } a_2 = \{\langle a_1, a_3 \rangle, \langle a_1 \rangle, \langle a_3 \rangle\}.$$

If endowing the vertices u in K with real values $f(u)$ of a function f (see also triangulation in Subsection 2.2.5), we can sort the vertices and the neighborhood vertices in its star according to their endowed values. Let f be a function being defined and non-degenerate for all vertices u of a given complex K, i.e., we assume that the function values are different at all vertices, [25].

Definition 2.11 (Lower-star [23, 25])**:**
The **lower-star** of a vertex u is the set of simplices in St u for which u has the maximum function value over all vertices. The **lower-link** of u is the set of faces of simplices in the lower-star of u that do not also belong to the lower-star:

$$\text{St\_}u := \{\sigma \in \text{St } u | v \in \sigma \Rightarrow f(v) \leqslant f(u)\},$$

$$\text{Lk\_}u := \{\tau \in \text{Lk } u | v \in \tau \Rightarrow f(v) \leqslant f(u)\}.$$

Remark 1:
With the lower-star definition for a given simplicial complex with all vertices endowed with real values, we will be able to endow the filter with more information which forms the called lower star filtration. In Algorithm 4.3 we will drop the assumption that the function values for all vertices have to be non-degenerate. Instead, we will give a unique procedure, which function value has to be taken for computing persistence pairs.

Definition 2.12 (Lower-star filtration [25])**:**
For a simplicial complex K with endowed real function values $f(v_i)$ for each vertex, we consider the sequence of all vertices $\{v_0, \cdots, v_{n-1}\}$ being ordered according to their increasing function values. Then the sequence of subcomplexes $\{\text{St\_}v_0, \cdots, \text{St\_}v_{n-1}\}$ generates a filter that forms a filtration that is called **lower-star filtration** of f.

Remark 2:
Let us reconsider Example 2 as follows. The simplicial complex K in this example can also be interpreted as the graph of a piecewise linear function. Assuming that $a_j = (x_j, y_j)$, $j = 1, \ldots, 4$ and $y_j = f(x_j)$, we can also consider the vertices $x_j \in \mathbb{R}^1$ endowed with the function values $y_j = f(x_j)$. We order the vertices x_j according to the size of their function values and obtain $\{x_1, x_2, x_3, x_4\}$. The corresponding lower-star sets are

$$\begin{aligned}
&\mathrm{St}_x_1 = \{\langle x_1\rangle\},\\
&\mathrm{St}_x_2 = \{\langle x_2\rangle\},\\
&\mathrm{St}_x_3 = \{\langle x_1\rangle, \langle x_2\rangle, \langle x_3\rangle, \langle x_2, x_3\rangle, \langle x_1, x_3\rangle\},\\
&\mathrm{St}_x_4 = \{\langle x_2\rangle, \langle x_4\rangle, \langle x_2, x_4\rangle\}.
\end{aligned}$$

The corresponding lower-star filtration is obtained with $K_0 = \emptyset$, and $K_j = K_{j-1} \cup \mathrm{St}\_x_j$. We obtain a complete lower-star filtration by first sorting the simplices in each set according to their dimension and then concatenating them (disregarding multiple appearance) to obtain a full filter sequence $\{\langle x_1\rangle, \langle x_2\rangle, \langle x_3\rangle, \langle x_2, x_3\rangle, \langle x_1, x_3\rangle, \langle x_4\rangle, \langle x_2, x_4\rangle\}$. The corresponding filtration coincides with the filtration considered in Example 2.

2.2.2 Pairing of simplices and persistence

In persistent homology, we are not interested in the exact structure of every homology group corresponding to a filtration but we want to know, how the Betti numbers of the homology groups change within each stage of a given filtration. Let us assume that we have a complete filtration of K, i.e., each subcomplex K_{i+1} in the filtration is obtained from K_i by adding exactly one simplex σ_i. The p-Betti number of a p-homology group increases by 1 when a new p-homology class is created, i.e., when a p-simplex σ_i with a certain property (called positive) is added in the filtration. The p-Betti number of the p-homology group decreases by 1 when a $(p+1)$-simplex of certain property (called negative) is added in the filtration that "destroys" a p-homology class. Those properties are investigated by examining how the new simplex is connected with preceding subcomplex. For a given p-homology class that is created by a positive p-simplex at a certain stage i of the filtration and is destroyed by a $(p+1)$-simplex at a later stage j of the filtration, the corresponding p-simplex and the $(p+1)$-simplex can be "paired". Their birth time difference in the filtration is called their persistence.

Definition 2.13 (Positive and negative simplex)**:**
In a complete filtration $\{K_j, j = 1, \cdots, n\}$, with $K_{j+1} = K_j \cup \sigma_j$, a p-simplex σ_i is called **positive**, iff it forms, together with some other p-simplices in K_i, a new p-cycle c in K_{i+1} with respect to K_i. Then the p-Betti number increases by one, and we call c being created by σ_i. A $(p+1)$-simplex σ_j is called **negative**, iff it destroys an existing p-cycle c in K_j by turning this cycle from a non-boundary p-cycle into a boundary. Then the p-Betti number decreases by one, i.e., $\partial\sigma_j = c$. In this case, we say that c is destroyed by σ_j.

Remark:
The definition here is given by checking the relationship of a newly added simplex with the existing subcomplex K_j according to the definition of boundary operator, i.e., how the new σ_j is connected with K_j.

Example 5:
1. We again examine Example 2, together with the complete lower-star filtration given in Remark 2 resp. in Example 2.
At the first stage, $\sigma_1 = \langle x_1 \rangle$ creates a new 0-cycle in K_1, i.e. rank $(H_0(K_1)) = 1$, and σ_1 is positive.
At the second stage, $\sigma_2 = \langle x_2 \rangle$ creates a second new 0-cycle in K_2, i.e., it is positive.
At the third stage, $\sigma_3 = \langle x_3 \rangle$ creates a third new 0-cycle in K_3, i.e., it is positive.
At the fourth stage, $\sigma_4 = \langle x_2, x_3 \rangle$ is a 1-simplex with boundary $\langle x_2 \rangle + \langle x_1 \rangle$ that does not create a new 1-cycle. But it destroys the zero cycle $\langle x_3 \rangle$ that appeared at stage 2. Alternatively we could say that the zero cycle $\langle x_2 \rangle$ is destroyed that appeared at stage 2. But in such a case, we always consider the cycle as "destroyed" that was born latest. Thus σ_4 is negative.
Analogously, $\sigma_5 = \langle x_1, x_3 \rangle$ destroys the 0-cycle in K_2 that was born after the 0-cycle in K_1.
The 0-simplex $\sigma_6 = \langle x_4 \rangle$ again creates a new 0-cycle in K_5, i.e., it is positive.
Finally, $\sigma_7 = \langle x_2, x_4 \rangle$ again destroys the newest 0-cycle, i.e., it is negative.

2. Considering Example 3 with the filtration given there involving also a 2-simplex, σ_5 creates and represents the 1-cycle $\langle a_1, a_2 \rangle + \langle a_2, a_3 \rangle + \langle a_1, a_3 \rangle$ which is later destroyed by $\sigma_6 = \langle a_1, a_2, a_3 \rangle$ as $\partial\sigma_6 = \langle a_1, a_2 \rangle + \langle a_2, a_3 \rangle + \langle a_1, a_3 \rangle$. Thus, σ_5 is positive since it creates a new 1-cycle, and σ_6 is negative since it destroys a 1-cycle.

The idea of persistence is the realization that the creation of classes in a homology group corresponding to one stage of the filtration can be related to the destruction of classes in a homology group corresponding to a later stage of the filtration [23]. During a filtration, the creation and destruction can be reflected by the change of Betti numbers at every filtration stage, i.e., adding a positive p-simplex σ_j increases the p-Betti number by one while adding a negative $(p+1)$-simplex σ_j decreases the p-Betti number by one. We thus can define the persistence as follows.

Definition 2.14 (Persistence and pair of simplices [24])**:**
The **persistence** of a p-cycle c is defined as the difference of the birth times of the two related simplices $\alpha(\sigma_j) - \alpha(\sigma_i)$ with $i \leq j$, where σ_j destroys the p-cycle c that was created by σ_i. The corresponding p-simplex σ_i and the $(p+1)$-simplex σ_j form a **persistence pair**. We say that $\alpha(\sigma_j) - \alpha(\sigma_i)$ is the persistence of the pair (σ_i, σ_j).

Remark: 1. Persistence describes, using birth times of σ_j and σ_i, how long c sustains in the filtration. Since a p-cycle always represents a homology class and this p-cycle can be represented by the positive simplex σ that creates the p-cycle, we can say the homology class is represened by σ. Due to the unique correspondence of the homology class and the simplex that creates it, we can claim that persistence of a homology class is also the persistence of its corresponding pair. The persistence definition can also be given by birth and death of a certain class in homology groups obtained during a filtration [24]. Those two definitions are in fact equivalent, since the birth of σ_j in the filtration means the death of the homology class which has been created by σ_i. We will further extend the persistence definition to be given by critical points.
2. Coming back to Example 2, see also Example 5, we find the persistence pairs (σ_3, σ_4), (σ_2, σ_5), (σ_6, σ_7) for dimension 0.

We now describe a formal algorithm for pairing of simplices as follows [24], where β_k^j is the kth Betti number of K_j.

Algorithm 2.15 (Pairing of simplices)**:**

Input: K_i and σ_i, for $i = 1, \cdots, n$.

Initialize: $C = \emptyset$, $P_k = \emptyset$, for all possible k.
For $j = 1, \ldots, n$ **do**
  If σ_j with $\dim \sigma_j = k$ is positive and creates the cycle c_j
    then add c_j into C, i.e., $C = C \cup c_j$.
  elseif σ_j with $\dim \sigma_j = k+1$ is negative and σ_j destroyes c_{i_0} in C,
    then form the pair (c_{i_0}, c_j) and add this pair in to P_k.
  end
end

Output: P_k (as a multiset) of persistence pairs of dimension k in the given filtration.

Example 6:
In Example 3, we get for $k = 1$ one simplex pair (σ_5, σ_6).

2.2.3 Pairing of vertices using lower-star filtration

As described above, the creation and destruction of classes in homology groups of subcomplexes K_{j+1} (resp. the change of Betti numbers of K_{j+1}), can be computed from

those of K_j merely by looking at the type of σ_j and how it is connected with K_j, [20,23]. Let us again look more closely at the (non-complete) lower-star filtration of a simplicial complex. We will now simplify the procedure of pairing for persistence computation. In Subsection 2.2.5 we will explain in more detail the importance of the lower-star filtration in our context.

Definition 2.16 (Local and non-local pairs [25])**:**
Assume that (σ, τ) is a pair of simplices given by Algorithm 2.15, where $\sigma \in \text{St } \sigma_s$ and $\tau \in \text{St } \sigma_t$. We say that σ and τ are **locally paired** if $s = t$ and they are **non-locally paired** if $s \neq t$.

Example 7:
We go back to Example 2, where we have computed already the lower-star filtration in Remark 2 and the persistence pairs for the complete filtration in Example 5, namely (σ_3, σ_4), (σ_2, σ_5), (σ_6, σ_7) for dimension 0. Now, with the lower-star filtration

$$\begin{aligned} &K_0 = \emptyset, \qquad K_1 = \{\langle x_1\rangle\}, \quad K_2 = \{\langle x_1\rangle, \langle x_2\rangle\}, \\ &K_3 = K_2 \cup \{\langle x_3\rangle, \langle x_1, x_3\rangle, \langle x_2, x_3\rangle\}, \\ &K_4 = K_3 \cup \{\langle x_4\rangle, \langle x_2, x_4\rangle\}, \end{aligned}$$

we observe that $(\sigma_3, \sigma_4) = (\langle x_3\rangle, \langle x_2, x_3\rangle)$ is locally paired (with vertex x_3) and similarly, also $(\sigma_6, \sigma_7) = (\langle x_4\rangle, \langle x_2, x_4\rangle)$ is locally paired (with vertex x_4). The only non-local pair is $(\sigma_2, \sigma_5) = (\langle x_2\rangle, \langle x_1, x_3\rangle)$.

A pair whose both simplices exist in a same lower-star is called **trivial**, since when using the (non-complete) lower-star filtration, the corresponding cycle is created and destroyed simultaneously when the lower-star set of the vertex is added in the filtration. In other words, we cannot figure out the existence of this class within the "resolution" of the lower-star filtration we designed. This leads us to consider only the nontrival pairs, i.e., nonlocal pairs, and in the following, we will pair vertices instead of pairing simplices.

Definition 2.17:
Let (σ, τ) be a non-locally paired simplex pair, where $\sigma \in \text{St}_- \sigma_s$ and $\tau \in \text{St}_- \sigma_t$. We define the corresponding persistence pair of vertices as (s, t).

Example 8:
In Example 2, see also the example above, we obtain the simplex pair (σ_2, σ_5), thus the vertices pair (x_2, x_3).

In Subsection 2.2.5, we will see that this definition is closely related with the idea of pairing of critical points of a function. A similar concept will be applied for piecewise linear functions on an interval later in Chapter 4, see e.g. Algorithm 4.3.

2.2.4 Boundary matrix and matrix-based persistence computation

The boundary matrix is used to describe the relation between a simplex and all its faces. It can be also used as a powerful tool for computing persistence pairs.

As indicated by Definition 2.5, we know that a $(p+1)$-simplex can have some p-simplex as its faces. In other words, a p-simplex can either be a face of a $(p+1)$-simplex or not. This relationship between p-simplex and $(p+1)$-simplex of a simplical complex can be described by a binary matrix given in the following definition.

Definition 2.18 (Boundary matrix [22, 42])**:**
For a given filter $\{\sigma_0, \cdots, \sigma_{n-1}\}$ for a simplical complex K as in Definition 2.9, the binary matrix M is called the **boundary matrix** of K, if M satisfies that $M(i,j) = 1 \iff \sigma_i$ is a face of σ_j.

Example 9:
We consider the simplicial complex K in Figure 2.1 and its filter in Example 2 as follows,

σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7
$\langle a_1 \rangle$	$\langle a_2 \rangle$	$\langle a_3 \rangle$	$\langle a_2, a_3 \rangle$	$\langle a_1, a_3 \rangle$	$\langle a_4 \rangle$	$\langle a_2, a_4 \rangle$

.

Then K has the following boundary matrix according to Definition 2.18,

$$M = \begin{array}{c} \\ \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

The boundary matrix has some nice properties. It can be used as a boundary operator if a p-chain is expressed in vector form.

Example 10:
We take the simplicial complex K in Figure 2.1 and the setup in Example 9. We compute the boundary of the 1-chain $c = \langle a_1, a_3 \rangle + \langle a_3, a_2 \rangle = \sigma_4 + \sigma_5$ using the boundary matrix M in Example 9. We first see that this 1-chain can be expressed as the vector below according the given fixed filter,

$$v = \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array},$$

where the binary values 1 at 4th and 5th index indicate that the corresponding simplices exist in the 1-chain. By multiplying M with v^T, we obtain the following vector $\tilde{v}$ which corresponds to c's boundary $\sigma_1 + \sigma_2 = \langle a_1 \rangle + \langle a_2 \rangle = \partial(\langle a_1, a_3 \rangle + \langle a_3, a_2 \rangle)$,

$$\tilde{v} = \begin{array}{c} \\ \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

With the help of the boundary matrix and the lowest index definition below, we can now describe Algorithm 2.15 in a matrix-based form.

Definition 2.19 (Lowest index [22])**:**
For a given binary matrix M with the jth column being denoted by M_j, we define the **lowest index** $\delta_j(M)$ of M_j as the row index of the lowest non-zero element of M_j:

$$\delta_j(M) = \begin{cases} k, & M_j \neq 0, k \text{ is the index of lowest non-zero element of } M_j, \\ 0, & \text{otherwise.} \end{cases}$$

For example, the matrix M in Example 9 yields $\delta_4(M) = 2$, $\delta_5(M) = 1$ and $\delta_7(M) = 2$, while for the zero columns we have $\delta_1(M) = \delta_2(M) = \delta_3(M) = \delta_6(M) = 0$.

In matrix form, Algorithm 2.15 can be written as follows, [22].

Algorithm 2.20 (Matrix-based simplices pairing)**:**

Input: M of size n by n.

Initialize: $M^{(1)} = M$.
For $j = 1, \ldots, n$ **do**
  For $k = 1, \ldots, j$ **do**
    If $\delta_k(M^{(j)}) = \delta_j(M^{(j)})$ then
      $M_j^{(j)} = M_j^{(j)} + M_k^{(j)}$.
    end
    Set $M^{(j+1)} = M^{(j)}$.
  end
end

Output: $M^{(n)}$ contains information on all simplex pairs:
  for each column index p, if $\delta_p(M^{(n)}) = q$, then (σ_p, σ_q) is a pair.

Example 11:
We apply Algorithm 2.20 to M in Example 9 and obtain

$$M^{(1)}: \begin{array}{c} \\ \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \rightarrow M^{(2)}: \begin{array}{c} \\ \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

Repeating the procedure yields no further change, i.e., $M^{(2)} = M^{(3)} = \ldots = M^{(8)}$. In the resulting matrix $M^{(2)}$, we have $\delta_4(M^{(2)}) = 3$, $\delta_5(M^{(2)}) = 2$ and $\delta_7(M^{(2)}) = 6$ which indicate the persistence pairs of simplices (σ_3, σ_4), (σ_2, σ_5) and (σ_6, σ_7), respectively. By Definition 2.17, we obtain the persistence pairs of vertices are (a_3, a_3), (a_2, a_3) and (a_4, a_4). When taking again the interpretation as in Remark 2, this is equivalent with the persistence pairs of vertices (x_3, x_3), (x_2, x_3) and (x_4, x_4), where only the pair (x_2, x_3) is non-local according to Definition 2.16.

2.2.5 Triangulation

All our discussion above was based on simplicial complexes. But functions are usually defined in a multidimensional real space $\mathbb{R}^d$ with $d \geqslant 1$, on manifolds, or even generally in a topological space. The theory which studies the homology properties of a topological space is called singular homology theory. We refer to [32] for more details. Singular homology groups are usually hard to compute. A possible way for simplying the computation of singular homology groups is to simplify the topological space itself to a simplicial complex while keeping some properties. Assuming that this simplification preserves the wanted topological properties of the space suitably, one can make use of the homology theory on simplicial complexes.

The simplification of a topological space is called triangulation. Let us first recall the definition of a homeomorphism.

Definition 2.21 (Homeomorphism)**:**
A function $h : \mathbf{X} \longrightarrow \mathbf{Y}$ between two topological spaces $(\mathbf{X}, T_{\mathbf{X}})$ and $(\mathbf{Y}, T_{\mathbf{Y}})$ is called a **homeomorphism** if it satisfies following properties.
(1) h is a bijection.
(2) h is continuous.
(3) h is invertible and the inverse function h^{-1} is also continuous.

Definition 2.22 (Triangulation)**:**
A **triangulation** of a topological space $\mathbf{X}$ is a simplicial complex K, homeomorphic to X, together with a homeomorphism $h : K \longrightarrow \mathbf{X}$.

Considering one-dimensional functions, the triangulation is simple and the procedure of computing persistence using the lower-star filtration is closely related to the notion of persistence for Morse functions, see [22].

We will not concentrate on triangulation too much but notice the following special one-dimensional case whose triangulation can be itself.

Persistence for piecewise linear functions

Let $f : \mathbb{R} \to \mathbb{R}$ be a piecewise linear function, given by the sequence of knots $(x_k)_{k\in I}$ and with $f(x_k) = y_k$ for $k \in I$, where I denotes a finite index set. As before in example 2, the connected points $a_k = (x_k, y_k)$ can be interpreted as a 0-simplicial complex. We assume that f is non-degenerate, i.e., $f(x_k) \neq f(x_j)$ for $k \neq j$. Applying the lower-star filtration, each vertex x_k will be connected to a filter set St_x_k. This set contains either only the vertex x_k itself, if it is a local minimum of f, or it contains beside the vertex x_k further 1-simplexes (edges), namely exactly one edge if x_k is neither a local minimum nor a local maximum, and two edges if it is a local maximum. In the latter two cases, there will be always local persistence pairs (namely x_k together with the one edge in St_x_k if x_k is neither a local minimum nor a local maximum, or x_k together with one of the two edges in St_x_k if x_k is a local maximum of f).

Therefore, we can simply find the important (non-local) persistence pairs that relate a local minimum vertex with a local maximum vertex.

The following persistence for Morse functions is equivalent to the above pairing procedure, see [22]. We consider all critical points of f, i.e., the vertices x_k that are either local minima or local maxima of f, and order them according to the size of function values starting with the global minimum. Now for the obtained order of corresponding function values $y_{k_l} = f(x_{k_l})$, we consider the sublevel sets $\mathbb{R}_t = f^{-1}(-\infty, t]$. Increasing t, the connectivity of $\mathbb{R}_t$ will change at each value y_{k_l}. A new component will be added at a local minimum value, and two components will merge at a local maximum value. The pairing procedure can now be done according to the following rule. When a new component is introduced at a local minimum value $t = y_j = f(x_j)$ then x_j is said to represent this component. If we pass a local maximum value $y_k = f(x_k)$ then we pair x_k with the younger (higher) of the two local minima that represent the two components being merged.

A similar procedure will be applied in Section 4 to define the new notion of persistence distance of piecewise linear functions on an interval.

3 Discrete total variation and the discrete 1D denoising problem

The total variation model is originally designed for continuous functions with bounded total variation.

3.1 Classical continuous TV restoration model

We first recall the origin of the TV restoration idea for continuous functions [13, 37].

Let $f(x)$ be the measured noise contaminated version of the clean signal $u(x)$, i.e.,

$$f(x) = u(x) + n(x), \tag{3.1}$$

where the noise $n(x)$ possesses mean 0 and variance σ^2,

$$En(x) = 0, \quad En^2(x) = \sigma^2. \tag{3.2}$$

Rudin, Osher and Fatemi [37] invented the TV denoising model which minimizes the total variation

$$TV(u) = \int_\Omega |\nabla u(x)| dx, \tag{3.3}$$

where Ω is the domain of a continuous function u and where

$$\nabla u(x) = \left((u_{x_1}(x))^2 + (u_{x_2}(x))^2\right)^{1/2}$$

is the gradient of u at $x = (x_1, x_2)$. Here we have assumed for simplicity that the original signal u is differentiable, i.e., that the partial derivatives in x_1- and x_2-direction indeed exist. Otherwise one needs to employ a more general definition for $TV(u)$, see e.g. [11].

Then, the assumption (3.2) on noise results in two constraints for the minimization of of the TV norm,

$$\int_\Omega u(x)dx = \int_\Omega f(x)dx, \qquad \frac{1}{|\Omega|}\int_\Omega (u(x)-f(x))^2 dx = \sigma^2.$$

With the constraints above, we can finally turn the TV minimization problem (3.3) into the following energy functional by introducing a Lagrange multiplier λ,

$$J(u) = \int_\Omega |\nabla u(x)|dx + \frac{\lambda}{2}\int_\Omega (u(x)-f(x))^2 dx. \tag{3.4}$$

The Euler-Lagrange equation of J is given by

$$-\nabla\cdot\Big(\frac{\nabla u}{|\nabla u|}\Big) + \lambda(u-f) = 0, \tag{3.5}$$

and steepest descent marching gives

$$\frac{\partial u}{\partial x} = \nabla\cdot\Big(\frac{\nabla u}{|\nabla u|}\Big) + \lambda(f-u).$$

By setting $\lambda = 0$, one obtains the Osher-Rudin TV diffusion

$$\frac{\partial u}{\partial x} = \nabla\cdot\Big(\frac{\nabla u}{|\nabla u|}\Big).$$

It is easy to observe that $|\nabla u|$ in (3.5) can have singularities in flat regions and at extrema. Due to those singularities, $|\nabla u|$ in (3.5) is regularized to

$$|\nabla u|_a = \sqrt{|\nabla u|^2 + a}$$

with some small positive parameter a.

Then one can show that the solution $u(x)$ of the regularized Euler-Lagrange equation (3.5) (with $|\nabla u|$ replaced by $|\nabla u|_a$) minimizes the following regularized version of (3.4)

$$J_a(u) = \int_\Omega |\nabla u(x)|_a dx + \frac{\lambda}{2}\int_\Omega (u(x)-f(x))^2 dx. \tag{3.6}$$

3.2 The discrete TV model

In this subsection, we introduce first the discrete TV model on graphs and the corresponding nonlinear restoration minimization algorithm due to [13]. As a fundament of the next chapter, we also elaborate the one-dimensional discrete TV minimization restoration functional and its minimization algorithm.

Our following considerations summarize the approach by Chan, Osher and Shen [13] to transfer the TV restoration model of Rudin, Osher and Fatemi [37] to the discrete setting.

3.2.1 Digital TV filters on graphs

The necessity of modeling a general digital domain by a graph was proposed by Osher and Shen, Alpert et al. in [2, 33].

A general digital domain can be modeled as a graph $[\Omega, E]$, with a finite set Ω of nodes and an edge dictionary E. If the nodes α and β are linked by an edge, we write $\alpha \sim \beta$. We consider now digital signals u being functions defined on Ω,

$$\mathbf{u}\colon \Omega \longrightarrow \mathbb{R}.$$

The function value at node α is written as u_α or $u(\alpha)$. The **local variation** at a node $\alpha \in \Omega$ is defined as

$$|\nabla_\alpha \mathbf{u}| := \sqrt{\sum_{\beta\sim\alpha} (u_\beta - u_\alpha)^2}. \tag{3.7}$$

For a small positive real number a, the **regularized local variation** at a node α is

$$|\nabla_\alpha \mathbf{u}|_a := \sqrt{\sum_{\beta\sim\alpha} (u_\beta - u_\alpha)^2 + a^2}. \tag{3.8}$$

The **edge derivative** is defined by discretization of the directional derivative in the continuous case. The edge derivative of u along e at α is determined by

$$\left.\frac{\partial \mathbf{u}}{\partial e}\right|_\alpha := u_\beta - u_\alpha, \tag{3.9}$$

where the edge e connects node α with node β, i.e., $\alpha \sim \beta$. This simple determination of the discrete edge derivative is especially senseful if all edges have the same length. Otherwise, one should think about normalizing by edge length, similarly as for first order approximations of derivatives by divided differences. One should notice that the edge derivative of u along e at β is

$$\left.\frac{\partial \mathbf{u}}{\partial e}\right|_\beta = u_\alpha - u_\beta.$$

Then it becomes obvious that

$$\left.\frac{\partial \mathbf{u}}{\partial e}\right|_\alpha = -\left.\frac{\partial \mathbf{u}}{\partial e}\right|_\beta.$$

The discrete gradient in (3.7) can be expressed as

$$|\nabla_\alpha \mathbf{u}| = \sqrt{\sum_{e \vdash \alpha} \Big(\frac{\partial \mathbf{u}}{\partial e}\Big|_\alpha \Big)^2}, \tag{3.10}$$

where $e \vdash \alpha$ means that the sum is taken over all edges possessing the node α.

Given a noisy signal f on Ω,

$$f_\alpha = u_\alpha + n_\alpha, \quad \alpha \in \Omega, \tag{3.11}$$

which has been contaminated by noise n, Chan et al. [13] proposed a non-linear data-dependent **digital TV filter**

$$\mathscr{F}^{\lambda,a} : \mathbf{u} \longrightarrow \mathbf{v},$$

where v is the output signal on Ω, λ is the parameter to balance approximation and smoothing, and a is the regulation parameter in (3.8), which can be of the order 10^{-4} for a typical signal. For simplicity, we denote $\mathscr{F}^{\lambda,a}$ by $\mathscr{F}$. For each node $\alpha \in \Omega$ the filter is given by

$$v_\alpha = \mathscr{F}^{\lambda,a}_\alpha(\mathbf{u}) := \sum_{\beta \sim \alpha} h_{\alpha\beta}(\mathbf{u}) u_\beta + h_{\alpha\alpha}(\mathbf{u}) u^0_\alpha. \tag{3.12}$$

The adaptive filter coefficients in (3.12) are defined by

$$\begin{aligned} h_{\alpha\beta}(\mathbf{u}) &:= \frac{w_{\alpha\beta}}{\lambda + \sum_{\gamma\sim\alpha} w_{\alpha\gamma}}, \\ h_{\alpha\alpha}(\mathbf{u}) &:= \frac{\lambda}{\lambda + \sum_{\gamma\sim\alpha} w_{\alpha\gamma}}, \\ w_{\alpha\beta}(\mathbf{u}) &:= \frac{1}{|\nabla_\alpha \mathbf{u}|_a} + \frac{1}{|\nabla_\beta \mathbf{u}|_a}. \end{aligned} \tag{3.13}$$

It is easy to verify that for any node α

$$h_{\alpha\alpha}(\mathbf{u}) + \sum_{\beta\sim\alpha} h_{\alpha\beta}(\mathbf{u}) = 1.$$

In this sense, $\mathscr{F}$ is a lowpass filter. The proposed algorithm now reads as follows.

Algorithm 3.1:

Input: noisy signal $\mathbf{f}$, parameters λ, a, $N_{\text{iteration}}$

1) Initialize $\mathbf{u}^0 = \mathbf{f}$.
2) **For** $k = 0, \ldots, N_{\text{iteration}}$ **do**
    For $j = 0, \ldots, N$ **do**
        Compute $v_j = \mathscr{F}_j^{\lambda,\alpha}(\mathbf{u}^k) = \sum_{\beta\sim\alpha} h_{\alpha\beta}(\mathbf{u}^k)u_\beta + h_{\alpha\alpha}(\mathbf{u}^k)u_\alpha^0$
    end
    Put $\mathbf{u}^{k+1} := \mathbf{v}$.
end

Output: $\mathbf{u} = \mathbf{u}^{N_{iteration}+1}$.

The iteration sequence $\mathbf{u}^k$ obtained in this algorithm has a fixed point that minimizes a digital version of the functional

$$J_a(u) = \int_\Omega |\nabla u(x)|_a dx + \frac{\lambda}{2}\int_\Omega (u(x) - f(x))^2 dx.$$

in (3.6), where $\int_\Omega |\nabla u(x)|_a dx$ is replaced by the sum of all local variations for all nodes, which is usually defined as **discrete total variation**.

The discretized minimization problem can be minimized by Algorithm (3.1) according to the following theorems, see [13].

Theorem 3.2:
If the filtering process in Algorithm 3.1 converges to some signal $\mathbf{u}$, then $\mathbf{u}$ satisfies

$$\sum_{e\vdash\alpha} \frac{\partial}{\partial e}\frac{-1}{|\nabla \mathbf{u}|_a}\frac{\partial \mathbf{u}}{\partial e}\bigg|_\alpha + \lambda(u_\alpha^0 - u_\alpha) = 0, \quad \text{for all } \alpha \in \Omega. \tag{3.14}$$

Proof: The limit signal $\mathbf{u}$ satisfies

$$u_\alpha = \mathscr{F}_\alpha(\mathbf{u}), \quad \alpha \in \Omega.$$

According to (3.12) and (3.13) it can be written as

$$\Big(\lambda + \sum_{\beta\sim\alpha} w_{\alpha\beta}\Big)u_\alpha = \sum_{\beta\sim\alpha} w_{\alpha\beta}u_\beta + \lambda u_\alpha^0, \qquad \alpha \in \Omega,$$

which equals to

$$\sum_{\beta\sim\alpha} w_{\alpha\beta}(u_\beta - u_\alpha) + \lambda(u_\alpha^0 - u_\alpha) = 0.$$

Thus, the proof is done if we can show that

$$\sum_{\beta\sim\alpha} w_{\alpha\beta}(u_\beta - u_\alpha) = \sum_{e\vdash\alpha} \frac{\partial}{\partial e}\frac{-1}{|\nabla u|_a}\frac{\partial u}{\partial e}\bigg|_\alpha$$

holds true. We compute the right-hand side of the above equation according to the definition of edge derivative (3.9) and find for the edge e linking α to β,

$$\begin{aligned}
\frac{\partial}{\partial e}\frac{-1}{|\nabla u|_a}\frac{\partial u}{\partial e}\bigg|_\alpha &= \Big(\frac{-1}{|\nabla u|_a}\frac{\partial u}{\partial e}\Big)\bigg|_\beta - \Big(\frac{-1}{|\nabla u|_a}\frac{\partial u}{\partial e}\Big)\bigg|_\alpha \\
= \quad & \frac{-1}{|\nabla_\beta u|_a}\Big(\frac{\partial u}{\partial e}\Big)\bigg|_\beta + \frac{1}{|\nabla_\alpha u|_a}\Big(\frac{\partial u}{\partial e}\Big)\bigg|_\alpha \\
= \quad & \frac{-1}{|\nabla_\beta u|_a}(u_\alpha - u_\beta) + \frac{1}{|\nabla_\alpha u|_a}(u_\beta - u_\alpha) \\
= \quad & \Big(\frac{1}{|\nabla_\beta u|_a} + \frac{1}{|\nabla_\alpha u|_a}\Big)(u_\beta - u_\alpha).
\end{aligned}$$

Noticing that the last term is exactly $w_{\alpha\beta}(u_\beta - u_\alpha)$ according to (3.13), we finish the proof. □

Remark: Formula (3.14) can be seen as a digital version of (3.5) on Ω,

$$\nabla \cdot \Big(\frac{\nabla u}{|\nabla u|}\Big) + \lambda(f - u) = 0,$$

where we take the negative on both sides of (3.5).

Theorem 3.3:
If the TV filtering process in Algorithm 3.1 converges, then the limit signal u is the unique minimizer of the digital TV energy functional

$$\widetilde{J}_a(\mathbf{v}) = \sum_{\alpha\in\Omega} |\nabla_\alpha \mathbf{v}|_a + \frac{\lambda}{2}\sum_{\alpha\in\Omega}(v_\alpha - u^0_\alpha)^2.$$

Proof:
It is easy to notice that the left-hand side of (3.14) is exactly the negative gradient of $\widetilde{J}_a(\mathbf{u})$ at node α. On the other hand, it has been shown that $\widetilde{J}_a(\mathbf{u})$ is strictly convex, see [11,33]. Thus, the assertion holds true. □

3.2.2 One-dimensional discrete TV

In this section, we specialize the general digital domain Ω to the one-dimensional discrete case and redescribe briefly the corresponding theorem and algorithm.

Let $\mathbf{X}$ be a **partition** of the interval $[a, b]$ of the form $a = x_0 < x_1 < \cdots < x_N = b$. Further, let us consider a sequence $\mathbf{y} = \{y_j\}_{j=0}^N$ of measured data that corresponds to the partition $\mathbf{X}$. Then (x_j, y_j), $j = 0, \ldots, N$, uniquely define a linear spline function $f : [a, b] \to \mathbb{R}$ with $f(x_j) = y_j$. We denote the space of linear splines with respect to the partition $\mathbf{X}$ by $S_1(\mathbf{X})$. In this special case, we hence have $\Omega = \{x_0, \ldots, x_N\}$ and neighboring nodes x_i and x_{i+1} are linked by an edge, i.e., $x_i \sim x_{i+1}$ for $i = 0, \ldots, N-1$.

Analogously to (3.11), let $\mathbf{u} = (u(x_j))_{j=0}^N$ be a finite digital function on $\mathbf{X}$ that is corrupted with white noise, i.e., we have the given data

$$y_j = f(x_j) = u(x_j) + n(x_j), \qquad j = 0, \ldots, N \tag{3.15}$$

for a partition $\mathbf{X}$, where n has zero mean and (unknown) deviation σ. Using the celebrated discrete ROF-model, a reconstruction of u with respect to $\mathbf{X}$ can be obtained as the minimizer of the functional

$$J(\mathbf{u}) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{\ell=0}^{N-1} |u(x_{\ell+1}) - u(x_\ell)|, \tag{3.16}$$

where the second term denotes the one-dimensional discrete total variation of $\mathbf{u}$. It is defined generally as follows.

Definition 3.4 (Discrete total variation)**:**
With partition $\mathbf{X}$ defined above, the **discrete total variation** of f (resp. $\mathbf{y}$) is defined as the absolute sum of all changes of function values, i.e.,

$$TV(f) := \sum_{j=0}^{N-1} |f(x_{j+1}) - f(x_j)| \tag{3.17}$$

resp.

$$TV(\mathbf{y}) := \sum_{j=0}^{N-1} |y_{j+1} - y_j|.$$

Let us shortly summarize some well-known properties of $TV(f)$ (resp. $TV(\mathbf{y})$), for a proof, we refer e.g. to [10], where a more general discrete TV is defined based on the discrete co-area formula.

Theorem 3.5:
Let $TV(\mathbf{y})$ with $\mathbf{y} = (f(x_j))_{j=0}^N \in \mathbb{R}^{N+1}$ be the discrete total variation of $f \in S_1(\mathbf{X})$. Then we have:

(1) $TV(\mathbf{y})$ is nonnegative and $TV(\mathbf{y}) = 0$ if and only if $\mathbf{y} = c \cdot \mathbf{1}$ with $\mathbf{1} := (1, \dots, 1)^T \in \mathbb{R}^{N+1}$ and $c \in \mathbb{R}$.

(2) $TV(\mathbf{y})$ is positively homogeneous, i.e., $TV(\lambda \cdot \mathbf{y}) = \lambda TV(\mathbf{y})$ for any $\lambda \geq 0$.

(3) $TV(\mathbf{y})$ is invariant by addition of a constant, i.e., $TV(\mathbf{y} + c \cdot \mathbf{1}) = TV(\mathbf{y})$.

(4) $TV(\mathbf{y}) : \mathbb{R}^{N+1} \to \mathbb{R}$ is a continuous functional.

(5) $TV(\mathbf{y})$ is submodular, i.e., for any two functions $f, g \in S_1(\mathbf{X})$ with $\mathbf{y} = (f(x_j))_{j=0}^N$ and $\mathbf{z} = (g(x_j))_{j=0}^N$, we have

$$TV(\mathbf{y}) + TV(\mathbf{z}) \geq TV(\max(\mathbf{y}, \mathbf{z})) + TV(\min(\mathbf{y}, \mathbf{z})),$$

where $\max(\mathbf{y}, \mathbf{z}) := (\max\{y_j, z_j\})_{j=0}^N$ and $\min(\mathbf{y}, \mathbf{z}) := (\min\{y_j, z_j\})_{j=0}^N$.

(6) The discrete total variation is a semi-norm, i.e., for $\mathbf{y}, \mathbf{z} \in \mathbb{R}^{N+1}$,

$$TV(\mathbf{y} + \mathbf{z}) \leq TV(\mathbf{y}) + TV(\mathbf{z}).$$

Proof:

For completeness we give the proof for these properties of the discrete TV that can in our case be partially simplified compared to [10].

(1) $TV(\mathbf{y}) = 0 \iff |y_j - y_{j-1}| = 0$, for $j = 1, \cdots, N \iff \mathbf{y} = c \cdot \mathbf{1}$.

(2) The property holds true, since $|(\lambda \cdot \mathbf{y})_j - (\lambda \cdot \mathbf{y})_{j-1}| = \lambda \cdot |y_j - y_{j-1}|$ holds for all $j = 1, \cdots, N$.

(3) The property holds true, since $|(\mathbf{y} + c \cdot \mathbf{1})_j - (\mathbf{y} + c \cdot \mathbf{1})_{j-1}| = |y_j - y_{j-1}|$ holds for all $j = 1, \cdots, N$.

(4) Assume that $\|\mathbf{y} - \mathbf{z}\|_\infty < \varepsilon$ for some $\varepsilon > 0$, i.e., $|y_j - z_j| < \varepsilon$ for $j = 0, \dots, N$. Then

$$TV(\mathbf{z}) = \sum_{j=1}^{N} |z_j - z_{j-1}| \leq \sum_{j=1}^{N} (|y_j - y_{j-1}| + 2\varepsilon) = TV(\mathbf{y}) + 2N\varepsilon,$$

and thus $|TV(\mathbf{y}) - TV(\mathbf{z})| < 2N\varepsilon$.

(5) For a fixed node $j \in \{1, \dots, N\}$, we consider locally all possible cases for $\mathbf{y}$ and $\mathbf{z}$. Without loss of generality we can assume that $y_{j-1} = \min\{y_{j-1}, y_j, z_{j-1}, z_j\}$. (Otherwise we exchange $\mathbf{y}$ and $\mathbf{z}$ or replace $\mathbf{y}$ by $-\mathbf{y}$.) Now we have to check the following six cases.

a) For $y_{j-1} < z_{j-1} < z_j < y_j$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| = |y_j - z_{j-1}| + |z_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

b) For $y_{j-1} < z_j < z_{j-1} < y_j$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| > |y_j - z_{j-1}| + |z_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

c) For $y_{j-1} < z_{j-1} < y_j < z_j$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| = |z_j - z_{j-1}| + |y_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

d) For $y_{j-1} < z_j < y_j < z_{j-1}$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| > |y_j - z_{j-1}| + |z_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

e) For $y_{j-1} < y_j < z_{j-1} < z_j$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| = |z_j - z_{j-1}| + |y_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

f) For $y_{j-1} < y_j < z_j < z_{j-1}$, we have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| = |z_j - z_{j-1}| + |y_j - y_{j-1}| \\ = \; & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}|. \end{aligned}$$

We thus have

$$\begin{aligned} & |y_j - y_{j-1}| + |z_j - z_{j-1}| \geqslant \\ & |\max\{y_j, z_j\} - \max\{y_{j-1}, z_{j-1}\}| + |\min\{y_j, z_j\} - \min\{y_{j-1}, z_{j-1}\}| \end{aligned}$$

holds true for all j. By summing up those inequalities for all $j = 1, \cdots, N$, we obtain $TV(\mathbf{y}) + TV(\mathbf{z}) \geq TV(\max(\mathbf{y}, \mathbf{z})) + TV(\min(\mathbf{y}, \mathbf{z}))$. Another general proof of this property based on "co-area formula" can be found in [10].

(6) It is obvious that

$$|(\mathbf{y} + \mathbf{z})_j - (\mathbf{y} + \mathbf{z})_{j-1}| = |y_j - y_{j-1} + z_j - z_{j-1}| \leqslant |y_j - y_{j-1}| + |z_j - z_{j-1}|$$

holds for $j = 1, \cdots, N$. Summing up both sides of the inequality over all j, we get $TV(\mathbf{y} + \mathbf{z}) \leq TV(\mathbf{y}) + TV(\mathbf{z})$. □

Let us first derive a very simple first-order iteration filter in order to minimize the discrete ROF functional (3.16) above. For any function u defined with respect to partition $\mathbf{X}$, let $\mathbf{u} = (u_j)_{j=0}^N = (u(x_j))_{j=0}^N$.

We consider the Euler-Lagrange equation and find for $j = 1, \ldots, N-1$,

$$\lambda(u_j - f_j) - \operatorname{sign}(u_{j+1} - u_j) + \operatorname{sign}(u_j - u_{j-1}) = 0,$$

and at the boundary,

$$\begin{aligned} \lambda(u_0 - f_0) - \operatorname{sign}(u_1 - u_0) &= 0, \\ \lambda(u_N - f_N) + \operatorname{sign}(u_N - u_{N-1}) &= 0, \end{aligned}$$

where we define

$$\operatorname{sign}(x) = \begin{cases} \frac{x}{|x|} & x \neq 0, \\ 0 & x = 0. \end{cases}$$

Using the definitions

$$\begin{aligned} g_{1,j}(\mathbf{u}) &:= \begin{cases} \frac{1}{|u_j - u_{j-1}|} & \text{for } u_j - u_{j-1} \neq 0 \\ 0 & \text{else,} \end{cases} \\ g_{2,j}(\mathbf{u}) &:= \begin{cases} \frac{1}{|u_j - u_{j+1}|} & \text{for } u_j - u_{j+1} \neq 0 \\ 0 & \text{else,} \end{cases} \end{aligned}$$

we can write the Euler-Lagrange equation above for $j = 1, \ldots, N-1$ as follows,

$$u_j(g_{1,j}(\mathbf{u}) + g_{2,j}(\mathbf{u}) + \lambda) = g_{j,1}(\mathbf{u})u_{j-1} + g_{j,2}(\mathbf{u})u_{j+1} + \lambda f_j,$$

and derive a simple iteration scheme in the form

$$u_j^{k+1} = \frac{g_{j,1}(\mathbf{u}^k)u_{j-1}^k + g_{j,2}(\mathbf{u}^k)u_{j+1}^k + \lambda f_j}{g_{1,j}(\mathbf{u}^k) + g_{2,j}(\mathbf{u}^k) + \lambda}.$$

At the boundary we obtain the simplifications

$$u_0^{k+1} = \frac{g_{2,0}(\mathbf{u}^k)u_1^k + \lambda f_0}{g_{2,0}(\mathbf{u}^k) + \lambda}, \qquad u_N^{k+1} = \frac{g_{N,1}(\mathbf{u}^k)u_{N-1}^k + \lambda f_N}{g_{1,N}(\mathbf{u}^k) + \lambda}.$$

An alternative to compute a minimizer of the discrete ROF functional is the application of the non-linear filter proposed by Chan, Osher and Shen in [13].

In this case, the **local variation** $|\nabla_j \mathbf{u}|$ at x_j is defined by

$$|\nabla_j \mathbf{u}| := \begin{cases} \sqrt{\big(u_j - u_{j-1}\big)^2 + \big(u_j - u_{j+1}\big)^2}, & j = 1, \cdots, N-1, \\ \sqrt{\big(u_j - u_{j+1}\big)^2}, & j = 0, \\ \sqrt{\big(u_j - u_{j-1}\big)^2}, & j = N. \end{cases}$$

Analogously, we define the regularized version with a small positive number a by

$$|\nabla_j \mathbf{u}|_a := \begin{cases} \sqrt{\big(u_j - u_{j-1}\big)^2 + \big(u_j - u_{j+1}\big)^2 + a^2}, & j = 1, \cdots, N-1, \\ \sqrt{\big(u_j - u_{j+1}\big)^2 + a^2}, & j = 0, \\ \sqrt{\big(u_j - u_{j-1}\big)^2 + a^2}, & j = N. \end{cases} \tag{3.18}$$

For the given noisy signal $\mathbf{u}^0 = \mathbf{f}$, we again consider the non-linear data-dependent **digital TV filter** by Chan et al.,

$$\mathscr{F}^{\lambda,a} : \mathbf{u} \longrightarrow \mathbf{v}$$

where $\mathbf{u}$ is a given signal on $\mathbf{X}$, $\mathbf{v}$ is the output signal on $\mathbf{X}$, λ is the parameter in (3.16) to balance approximation and smoothing, and a is the regulation parameter in (3.18), which can be of the order 10^{-4} for a typical signal. For simplicity, we denote $\mathscr{F}^{\lambda,a}$ by $\mathscr{F}$. For a fixed node $x_j \in \mathbf{X}$, we apply the the TV filter

$$v_j = \mathscr{F}_j^{\lambda,a}(\mathbf{u}) = h_{j,j-1}(\mathbf{u})u_{j-1} + h_{j,j+1}(\mathbf{u})u_{j+1} + h_{j,j}(\mathbf{u})u_j^0, \tag{3.19}$$

where we let the item(s) vanish if the the indices $j-1$ or $j+1$ is not in $\{0, \ldots, N\}$. In the following context, we always assume that an item vanishes if its any index is out of $\{0, \ldots, N\}$.

The adaptive filter coefficients in (3.19) are now given by

$$\begin{aligned} h_{j,k}(\mathbf{u}) &= \frac{w_{j,k}(\mathbf{u})}{\lambda + w_{j,j-1}(\mathbf{u}) + w_{j,j+1}(\mathbf{u})}, \quad \text{for } k = j-1 \ \text{ and } \ k = j+1 \\ h_{j,j}(\mathbf{u}) &= \frac{\lambda}{\lambda + w_{j,j-1}(\mathbf{u}) + w_{j,j+1}(\mathbf{u})}, \\ w_{j,k}(\mathbf{u}) &= \frac{1}{|\nabla_j \mathbf{u}|_a} + \frac{1}{|\nabla_k \mathbf{u}|_a}, \quad \text{for } k = j-1 \ \text{ and } \ k = j+1. \end{aligned}$$

It is easy to check that for any index j

$$h_{j,j}(\mathbf{u}) + h_{j,j-1}(\mathbf{u}) + h_{j,j-1}(\mathbf{u}) = 1.$$

In this sense, $\mathscr{F}$ is a lowpass filter. The complete denoising algorithm in the one-dimensional case applied to the noisy signal f is thus as follows.

Algorithm 3.6:

Input: noisy signal $\mathbf{f}$, parameters λ, a, $N_{\text{iteration}}$.

1) Initialize $\mathbf{u}^0 = \mathbf{f}$.
2) **For** $k = 0, \ldots, N_{\text{iteration}}$ **do**
   For $j = 0, \ldots, N$ **do**
   Compute $v_j = \mathscr{F}_j^{\lambda,a}(\mathbf{u}^k) = h_{j,j-1}(\mathbf{u}^k)u_{j-1}^k + h_{j,j+1}(\mathbf{u}^k)u_{j+1}^k + h_{j,j}(\mathbf{u}^k)u_j^0$
   end
   Put $\mathbf{u}^{k+1} := \mathbf{v}$.
   end

Output: $\mathbf{u} = \mathbf{u}^{N_{iteration}+1}$.

Theorem 3.7:
If the TV filtering process in Algorithm 3.6 converges, then the limit signal $\mathbf{u}$ is the unique minimizer of the TV energy functional (3.16)

$$J(\mathbf{u}) := \frac{\lambda}{2}\sum_{j=0}^{N}|u(x_j) - f(x_j)|^2 + \sum_{\ell=0}^{N-1}|u(x_{\ell+1}) - u(x_\ell)|.$$

3.2.3 Primal-dual minimization method for the one-dimensional ROF model

We briefly summarize the primal-dual minimization method by Chambolle and Pock [12] specilized to the one-dimensional ROF-model.

It has been shown in [12] that the minimization problem of (3.16) can be rewritten as the following primal-dual problem

$$\min_{\mathbf{u}\in\mathbb{R}^{N+1}}\max_{\mathbf{w}\in\mathbb{R}^{N+1}}(\mathbf{w}^T\mathbf{D}\mathbf{u} + \frac{\lambda}{2}\sum_{j=0}^{N}|u(x_j) - f(x_j)|^2 - P_{Y^*}(\mathbf{w})), \tag{3.20}$$

where $Y^* = \{\mathbf{p} \in \mathbb{R}^{N+1} : \| \mathbf{p} \|_\infty \leqslant 1\}$, and

$$P_{Y^*}(\mathbf{w}) := \begin{cases} 0 & \mathbf{w} \in Y^*, \\ +\infty & \mathbf{w} \notin Y^*. \end{cases}$$

Further $\mathbf{D}$ is the $(N+1) \times (N+1)$ matrix

$$\mathbf{D} = \begin{pmatrix} -1 & 1 & \cdots & 0 & 0 \\ & -1 & 1 & \cdots & 0 \\ & & \cdots & & \\ & & & -1 & 1 \\ & & & & 0 \end{pmatrix}$$

such that the total variation of $\mathbf{u}$ can be written as the 1-norm of $\mathbf{Du}$, i.e., $TV(\mathbf{u}) = \|\mathbf{Du}\|_1$.

The saddle-point problem in (3.20) can now be solved using the following iterative procedure, see [12].

Algorithm 3.8:

Input: noisy signal $\mathbf{f}$, parameters λ, $\tau, \sigma > 0$, $\theta \in [0,1]$, $N_{\text{iteration}}$.

1) Initialize $\bar{\mathbf{u}}^0 = \mathbf{y}$, $\mathbf{w}^0 = \mathbf{0}$.

2) **For** $k = 0, \ldots, N_{\text{iteration}}$ **do**

Let $\mathbf{w}^{k+1} = (\mathbf{w}^k + \sigma \mathbf{D}\bar{\mathbf{u}}^k)/(\max(\mathbf{1}, \text{abs}(\mathbf{w}^k + \sigma \mathbf{D}\bar{\mathbf{u}}^k)))$

$\mathbf{u}^{k+1} = \frac{1}{1+\tau\lambda}(\bar{\mathbf{u}}^k - \tau \mathbf{D}^T \mathbf{w}^{k+1} + \tau\lambda \mathbf{f})$

$\bar{\mathbf{u}}^{k+1} = \mathbf{u}^{k+1} + \theta(\mathbf{u}^{k+1} - \mathbf{u}^k)$

end

Output: $\mathbf{u}^{N_{\text{iteration}}+1}$ approximates the minimizer of (3.16).

In the above algorithm 3.8, the operators "/", "abs(·)" and "max(·)" have to be applied componentwisely.

In our numerical experiments in Chapter 5, we will take $\theta = 1$. The benefit of this setup is that one can show the convergence of the iteration procedure, see [12] for more details.

4 Persistence distance and its relation to discrete total variation

In this chapter, we introduce the new concept of persistence distance based on persistence pairs and the corresponding difference of function values of a one-dimensional spline function f on an interval. The persistence distance consists of a sum of distances of function values of f being local extrema of the function f. We will show that the persistence distance possesses a lot of favorable properties. In particular, we show that there exists a close relationship between the persistence distance and the discrete total variation of a continuous one-dimensional function. However, differently from the discrete total variation, where just the absolute differences of neighboring function values are accumulated, the new persistence distance contains more information about the topological structure of the function. The persistence distance and its relation to discrete total variation will be used for establishing a new signal denoising model in Chapter 5. This model can be also described as a new weighted ROF model.

4.1 Persistence distance and its properties

We have already seen in Subsection 2.2.5 that the extremal values (vertices) of a one-dimensional piecewise linear function f play an important role in investigating the topological persistence properties of f. Now we want to derive the notion of persistence distance, based on persistence pairs. We want to get rid of the restriction that f has to be non-degenerate and do not longer assume that the function values $y_j = f(x_j)$, $x_j \in \mathbf{X} := \{x_0, \ldots, x_N\}$ are pairwise different.

For the one-dimensional signal $\mathbf{y} = (f(x_j))_{j=0}^N$ on the partition $\mathbf{X}$ we first define the (one-sided) local maxima and minima as follows.

Definition 4.1:
A knot $x_l \in \mathbf{X} \setminus \{x_0, x_N\}$ is called (left-sided) local minimum knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the local minimum value $y_l = f(x_l)$, if $y_{l-1} = f(x_{l-1}) > f(x_l)$, and if there exists a $\nu \in \mathbb{N}_0$ such that $l + \nu + 1 \leq N$ and

$$f(x_l) = f(x_{l+1}) = \cdots = f(x_{l+\nu}) < f(x_{l+\nu+1}).$$

Analogously, a knot $x_l \in \mathbf{X} \setminus \{x_0, x_N\}$ is called (left-sided) local maximum knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the local maximum value $y_l = f(x_l)$, if $y_{l-1} = f(x_{l-1}) < f(x_l)$, and if there exists a $\nu \in \mathbb{N}_0$ such that $l + \nu + 1 \leq N$ and

$$f(x_l) = f(x_{l+1}) = \cdots = f(x_{l+\nu}) > f(x_{l+\nu+1}).$$

The boundary knot $x_0 \in \mathbf{X}$ is called (left-sided) local minimum (resp. maximum) knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the local minimum (resp. maximum) value $y_0 = f(x_0)$, if there exists a $\nu \in \mathbb{N}_0$ with $\nu \leq N - 1$ such that

$$f(x_0) = f(x_1) = \cdots = f(x_\nu) < f(x_{\nu+1})$$

(resp. $f(x_0) = f(x_1) = \cdots = f(x_\nu) > f(x_{\nu+1})$). The boundary knot $x_N \in \mathbf{X}$ is called local minimum (resp. maximum) knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the local minimum (resp. maximum) value $y_N = f(x_N)$, if $f(x_{N-1}) > f(x_N)$ (resp. $f(x_{N-1}) < f(x_N)$) holds.

We now consider the subsets of $\{y_j : j = 0, \ldots, N\}$,

$$Y_m := \{y_k = f(x_k) : y_k \text{ is a local minimum value of } \mathbf{y}\},$$

$$Y^m := \{y_k = f(x_k) : y_k \text{ is a local maximum value of } \mathbf{y}\},$$

as well as the corresponding subsets of the partition $\mathbf{X}$,

$$X_m := \{x_k : f(x_k) \in Y_m\},$$

$$X^m := \{x_k : f(x_k) \in Y^m\}.$$

Further, let $x_{max} := \max\{X_m, X^m\}$ be the extremum knot with highest index occurring in the set $X_m \cup X^m$. Observe that x_{max} not coincides with x_N if $f(x_\nu) = \ldots = f(x_{N-1}) = f(x_N)$ for some $\nu < N$. For the number of elements in Y_m and Y^m we obviously have the relation

$$\#Y_m - \#Y^m \in \{-1, 0, 1\},$$

since after ordering the knots $x_k \in X_m \cup X^m$ by size, a local minimum (maximum) knot always possesses a local maximum (minimum) as its neighbor.

Definition 4.2:
The knot $x_l \in X_m$ is called global minimum knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the global minimum value $f(x_l)$ if $x_l = \underset{x \in X_m}{\operatorname{argmin}} f(x)$. The knot $x_l \in X^m$ is called global maximum knot of $\mathbf{y} = (f(x_j))_{j=0}^N$ on $\mathbf{X}$ with the global maximum value $f(x_l)$ if $x_l = \underset{x \in X^m}{\operatorname{argmax}} f(x)$.

If the global maximum (or minimum) knot is not uniquely determined by Definition 4.2 then we take the knot x_l with smallest index l. In this way we allow also functions where the global minimum or the global maximum is taken at more than one knot.

We want to derive an algorithm for finding the persistence pairs that simplifies Algorithm 2.15 and particularly does not involve the construction of homology groups. The idea is closely related to the persistence of Morse functions, see [22]. The pairing procedure for a one-dimensional function can be done by investigating of its local maxima and local minima and by pairing them using the following idea: a (local) minimum at t creates and represents a new component of the level set $\mathbb{R}_t = f^{-1}(-\infty, t]$. At a (local) maximum two components of the level set are merged and we pair the higher representer of these two components with the maximum. The new merged component is then represented by the lower minimum. An equivalent description is: when passing a maximum, we pair the maximum with the higher neighboring minimum and pull out the paired values from the set of local extrema, see [22].

We now construct **persistence pairs** (x_k, x_l) of $\mathbf{y} = (f(x_j))_{j=0}^N$ over the partition $\mathbf{X}$ by the following algorithm according to the idea described above. In this algorithm, we no longer require the function to be non-degenerate, since we always can pair a minimum knot with the left maximum knot such that the ambiguity of pairing can be eliminated when two local maximum knots possess the same function value. We pair a maximum knot with the neighboring minimum knot on the right-hand side if the two minima share the same function value.

Algorithm 4.3:

Input: Y_m, Y^m, X_m, X^m for $\mathbf{y} = (f(x_k))_{k=0}^N$.

1) Let $r := \#Y^m$, $P_1 := \emptyset$ and $X_{m,0} := X_m$.
Fix the ordered set $K_0 := \{f(x_{k_1}) \leq \cdots \leq f(x_{k_r})\}$ of all local maximum values in Y^m using the convention that for $f(x_k) = f(x_l) \in Y^m$ we take $f(x_k)$ first if $x_k < x_l$.

2) **For** $l = 1, \ldots, r$ **do**
Consider the l-th entry $f(x_{k_l})$ in the ordered set K_0.
If $x_{k_l} \notin \{x_0, x_{\max}\}$ then find the two spatial neighbors $\tilde{x}_1, \tilde{x}_2 \in X_{m,l-1}$ of x_{k_l}.
Put $\tilde{x} := \operatorname{argmin}_{x \in \{\tilde{x}_1, \tilde{x}_2\}} \mid f(x_{k_l}) - f(x) \mid$, where in case of
$\mid f(x_{k_l}) - f(\tilde{x}_1) \mid = \mid f(x_{k_l}) - f(\tilde{x}_2) \mid$ we take $\tilde{x} = \max\{\tilde{x}_1, \tilde{x}_2\}$.
Then $(\tilde{x},\, x_{k_l})$ resp. $(x_{k_l}, \tilde{x})$ is a persistence pair of f, and we set
$P_1 = P_1 \cup \{(\tilde{x},\, x_{k_l})\}$ and $X_{m,l} := X_{m,l-1} \setminus \{\tilde{x}\}$.
Here we apply the convention that the knots in the persistence pairs are ordered by size, i.e. we write $(\tilde{x}, x_{k_l})$ if $\tilde{x} < x_{k_l}$ and $(x_{k_l}, \tilde{x})$ if $\tilde{x} > x_{k_l}$.

Output: P_1 containing all persistence pairs of $\mathbf{y}$ (resp. f).

With the above procedure, we obtain at least $\#Y^m - 2$ persistence pairs, since each local maximum knot of f (resp. $\mathbf{y}$) that is not at the boundary (i.e. not in $\{x_0, x_{max}\}$) is paired with one local minimum knot by the above algorithm. Observe that in this way also each local minimum knot being not the global minimum knot, is contained in exactly one persistence pair while the global minimum knot is not paired. A boundary knot (i.e., x_0 or x_{max}) occurs as a knot in a persistence pair if it is a local but not the global minimum knot, and it is not contained in any persistence pair if it is a local maximum knot or the global minimum knot.

Remark:
It is easy to see that Algorithm 4.3 applied to Example 2 gives the same pairing result (x_2, x_3) as Algorithm 2.15 with the simplification obtained by the lower-star filtration where only non-local simplex pairs are considered, compare Example 8.

Example 12:
Let us consider the vector $\mathbf{y} = (0, 2, 1, 3, 1, 4, -1, 0, 1)$ on the equidistant partition $\mathbf{X} = \{x_j\}_{j=0}^N$ with $x_j = j$, $j = 0, \ldots, N$, where $N = 8$, see Figure 4.1 (left).

According to the definition, we find the sets

$$\begin{aligned} Y^m &= \{2, 3, 4, 1\}, \qquad Y_m = \{0, 1, 1, -1\} \\ X^m &= \{x_1, x_3, x_5, x_8\}, \qquad X_m = \{x_0, x_2, x_4, x_6\}. \end{aligned}$$

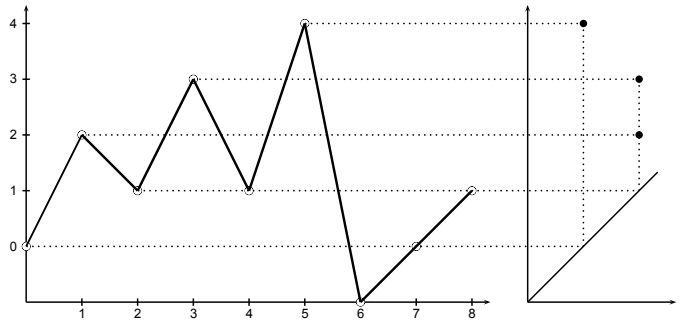


Fig. 4.1: Spline function f in Example 12 (left), corresponding persistence diagram (right).

Algorithm 4.3 provides now with $K_0 = \{1, 2, 3, 4\} = \{f(x_8), f(x_1), f(x_3), f(x_5)\}$ the set of persistence pairs

$$P_1 = \{(x_1, x_2), (x_3, x_4), (x_0, x_5)\}.$$

The global minimum knot x_6 and the local maximum knot x_8 at the boundary do not occur in any persistence pair.

Example 13:
Let us consider a second example with degenerate local extrema. Consider the vector $\mathbf{y} = (1, 0, 1, 0, 1, 0, 1, 0, 1)$ on the equidistant partition $\mathbf{X} = \{x_j\}_{j=0}^{8}$ with $x_j = j$.

According to our definition, we find now for this degenerate case the sets

$$\begin{aligned} Y^m &= \{1, 1, 1, 1, 1\}, \qquad Y_m = \{0, 0, 0, 0\} \\ X^m &= \{x_0, x_2, x_4, x_6, x_8\}, \qquad X_m = \{x_1, x_3, x_5, x_7\}. \end{aligned}$$

Algorithm 4.3 provides now with $K_0 = \{1, 1, 1, 1, 1\} = \{f(x_0), f(x_2), f(x_4), f(x_6), f(x_8)\}$ the set of persistence pairs

$$P_1 = \{(x_2, x_3), (x_4, x_5), (x_6, x_7)\}.$$

The minimum knot x_1 and the local maximum knots x_0 and x_8 at the boundary do not occur in any persistence pair.

Remark 3:
In computational topology, the persistence pairs are usually visualized by barcodes [9] or by a persistence diagram, see e.g. [15, 42]. Each persistence pair (x_k, x_l) corresponds to the point $(f(x_k), f(x_l))$ in the persistence diagram, and the distance of this point to the line $y = x$, i.e., the distance $|f(x_k) - f(x_l)|$ gives us some information about the "topological relevance" of these two local extrema of f. Important features correspond to points being further away from the diagonal, i.e., to persistence pairs (x_k, x_l) with significant distances $|f(x_l) - f(x_k)|$. In Figure 4.1 (right) the persistence diagram for Example 12 is illustrated.

Now, we want to construct a second set of persistence pairs for f (resp. for $\mathbf{y}$) on $\mathbf{X}$. For that purpose, we apply Algorithm 4.3 also to the sequence $\{-f(x_j)\}_{j=0}^N = \{-y_j\}_{j=0}^N$, and obtain a set P_2 of persistence pairs.

Obviously, the transfer from $\{f(x_j)\}_{j=0}^N$ to $\{-f(x_j)\}_{j=0}^N$ switches the roles of the sets Y_m and Y^m (and of X_m and X^m), i.e., using the notations

$$Y_m(-f),\ Y^m(-f),\ X_m(-f),\ X^m(-f)$$

for the sets of extremal values of $\{-f(x_j)\}_{j=0}^N$ and their corresponding knots $\{x_j\}_{j=0}^N$, we have

$$f(x_j) \in Y_m \iff -f(x_j) \in Y^m(-f),$$
$$f(x_j) \in Y^m \iff -f(x_j) \in Y_m(-f),$$

and $X_m(-f) = X^m$, $X^m(-f) = X_m$.

Considering again the Example 12 with $-\mathbf{y} = (0, -2, -1, -3, -1, -4, 1, 0, -1)$, we then obtain a second set of persistence pairs

$$P_2 = \{(x_1, x_2), (x_3, x_4), (x_6, x_8)\}.$$

In particular, we observe that the global maximum knot x_5 of X^m does not occur in any persistence pair of P_2. Analogously, applying the procedure to Example 13 with $-\mathbf{y} = (-1, 0, -1, 0, -1, 0, -1, 0, -1)$, we obtain the second set of persistence pairs

$$P_2 = \{(x_1, x_2), (x_3, x_4), (x_5, x_6), (x_7, x_8)\}.$$

Comparing the sets P_1 and P_2, we note that the persistence pairs found in P_1 and P_2 partially coincide, but usually P_1 and P_2 are not equal. Further, the boundary extremum knots x_0 and x_{max} are included in at most one persistence pair, either in one from P_1 or in one from P_2, since they are not regarded when being a local maximum knot. Indeed, x_0 (resp. x_{max}) will not occur in any persistence pair, i.e., neither in P_1 nor in P_2, if it is a global extremum knot. We are now ready for the following new definition.

Definition 4.4 (Persistence distance):
For a given function $f \in S_1(\mathbf{X})$ respective the vector $\mathbf{y} = (f(x_j))_{x_j \in X}$, we define the **persistence distance** by

$$\|f\|_{per} = \|\mathbf{y}\|_{per} = \|\mathbf{y}|\mathbf{X}\|_{per} :=$$
$$\sum_{(x_k, x_l) \in P_1} |f(x_l) - f(x_k)| + \sum_{(x_k, x_l) \in P_2} |f(x_l) - f(x_k)|,$$

i.e., as the sum over all distances of function values for the persistence pairs in P_1 and P_2.

Observe that for persistence pairs that occur twice, i.e., are contained in $P_1 \cap P_2$, the corresponding absolute difference of function values is added twice. We call a set in which an element can appear more than one time as **multiset**.

Remark 4:
As far as we know, the persistence distance as given in Definition 4.4 has not been regarded before in the homology literature. The idea to consider a so-called p-norm of the persistence diagram of a function $f_t : \mathbb{R}^2 \to \mathbb{R}$ that is obtained by convolving the original function $f : \Omega \to \mathbb{R}$ with the isotropic Gaussian kernel with scale $t > 0$ (in the two-dimensional case), can be found already in [14]. This p-norm takes the p-th root of the sum of the p-th powers of all persistences. In contrast to the p-norm definition of the persistence diagram, we consider the persistence pairs for a function on a bounded interval and have to treat extremal values at the boundary with special care. Further, we consider the persistences for f and for $-f$.

Let us derive some properties of the persistence distance $\|f\|_{per} = \|\mathbf{y}\|_{per}$.

Theorem 4.5:
Let $f \in S_1(\mathbf{X})$ be a spline function with $\mathbf{y} = (f(x_j))_{j=0}^N$ on the partition $\mathbf{X} = \{x_0, \dots, x_N\}$ of $[a, b]$. Then the persistence distance $\|f\|_{per} = \|\mathbf{y}|\mathbf{X}\|_{per} = \|\mathbf{y}\|_{per}$ satisfies the following properties.

(1) $\|\mathbf{y}\|_{per} \geq 0$. We have $\|\mathbf{y}\|_{per} = 0$ if and only if $\mathbf{y} = (y_j)_{j=0}^N$ is monotone.

(2) For each $c \in \mathbb{R}$, we have $\|c\mathbf{y}\|_{per} = |c| \cdot \|\mathbf{y}\|_{per}$.

(3) The persistence distance is invariant under addition of a constant function,

$$\|\mathbf{y} + c\mathbf{1}\|_{per} = \|\mathbf{y}\|_{per},$$

where $\mathbf{1} = (1, \dots, 1)^T \in \mathbb{R}^{N+1}$ and $c \in \mathbb{R}$. In particular, $\|c\mathbf{1}\|_{per} = 0$.

(4) The persistence distance $\|\mathbf{y}\|_{per} : \mathbb{R}^{N+1} \to \mathbb{R}$ is a continuous functional.

(5) The persistence distance $\|\mathbf{y}\|_{per}$ is submodular, i.e., for $f, g \in S_1(\mathbf{X})$ with $\mathbf{y} = (f(x_j))_{j=0}^N$ and $\mathbf{z} = (g(x_j))_{j=0}^N$ we have

$$\|\mathbf{y}\|_{per} + \|\mathbf{z}\|_{per} \geq \|\max(\mathbf{y}, \mathbf{z})\|_{per} + \|\min(\mathbf{y}, \mathbf{z})\|_{per},$$

where $\max(\mathbf{y}, \mathbf{z}) := (\max\{y_j, z_j\})_{j=0}^N$ and $\min(\mathbf{y}, \mathbf{z}) := (\max\{y_j, z_j\})_{j=0}^N$.

(6) There exist $\mathbf{y}, \mathbf{z} \in \mathbb{R}^{N+1}$ such that the persistence distance $\|\mathbf{y}\|_{per}$ does not satisfy the triangle inequality, i.e.,

$$\|\mathbf{y} + \mathbf{z}\|_{per} \leq \|\mathbf{y}\|_{per} + \|\mathbf{z}\|_{per}.$$

Hence, $\|\mathbf{y}\|_{per}$ is not convex.

Proof:

(1) The property $\|\mathbf{y}\|_{per} \geq 0$ is obvious by definition, where $\|\mathbf{y}\|_{per} = 0$ can only occur if there are no persistence pairs, neither for f nor for $-f$, i.e., $P_1 \cup P_2 = \emptyset$. According to Algorithm 4.3, we have $P_1 = \emptyset$, if and only if the set Y^m is a subset of $\{f(x_0),\, f(x_{max})\}$, i.e., there are local maxima only at the boundary. Analogously, $P_2 = \emptyset$, if and only if $Y_m \subset \{f(x_0), f(x_{max})\}$, i.e., there are local minima only at the boundary. Hence, $P_1 \cup P_2 = \emptyset$ is true if and only if $\mathbf{y}$ is monotone.

(2) This property is obvious, where for $c < 0$ the roles of X_m and X^m and hence of P_1 and P_2 are exchanged.

(3) All persistence pairs and hence the persistence distance are invariant under addition of a constant.

(4) Since f is a tame function, this assertion is a direct consequence of the stability of persistence diagrams, see e.g. [15]. In the special case considered here, we can also derive this property directly. Assume first, that the vector $\mathbf{y} = (f(x_j))_{j=0}^N$ is non-degenerate, i.e., that $y_j \neq y_k$ for $j \neq k$. Then, there exists an $\varepsilon > 0$ such that for each $\tilde{\mathbf{y}}$ with $\|\mathbf{y} - \tilde{\mathbf{y}}\|_\infty < \varepsilon$ the sets of minimum and maximum knots for $\mathbf{y}$ and $\tilde{\mathbf{y}}$ coincide, i.e., $X^m = \tilde{X}^m$ and $X_m = \tilde{X}_m$, and such that the order of maximum and minimum values (i.e., the order of the values $f(x_{k_1}), \ldots, f(x_{k_r})$ in the set K_0 in Algorithm 4.3) does not change, and hence all persistence pairs (x_k, x_l) remain the same for $\mathbf{y}$ and $\tilde{\mathbf{y}}$. Hence

$$\begin{aligned}
|\|\mathbf{y}\|_{per} - \|\tilde{\mathbf{y}}\|_{per}| &\leq \sum_{(x_k,x_l)\in P_1} |(|y_l - y_k| - |\tilde{y}_l - \tilde{y}_k|)| \\
&\quad + \sum_{(x_k,x_l)\in P_2} |(|y_l - y_k| - |\tilde{y}_l - \tilde{y}_k|)| \\
&\leq \sum_{(x_k,x_l)\in P_1} |(y_l - \tilde{y}_l) - (\tilde{y}_k - y_k)| + \sum_{(x_k,x_l)\in P_2} |(y_l - \tilde{y}_l) - (\tilde{y}_k - y_k)| \\
&\leq 2N\varepsilon.
\end{aligned}$$

The last inequality follows from the fact that $\#P_1 \leq \#Y^m$ and $\#P_2 \leq \#Y_m$, where Y^m resp. Y_m contain the maximum resp. minimum values of $\mathbf{y}$.

In the case of equal function values in $\mathbf{y}$, the sets $\tilde{P}_1$ and $\tilde{P}_2$ may enlarge for the perturbed vector $\tilde{\mathbf{y}}$. However, for each pair $(x_k, x_l) \in P_1 \cup P_2$ there exists a persistence pair $(x_{k'}, x_{l'}) \in \tilde{P}_1 \cup \tilde{P}_2$, with $y_k - y_{k'} = 0$, $y_l - y_{l'} = 0$ and $y_k - \tilde{y}_{k'} < \varepsilon$, $y_l - \tilde{y}_{l'} < \varepsilon$. Further, the new sets $\tilde{P}_1$ and $\tilde{P}_2$ of $\tilde{\mathbf{y}}$ may contain new persistence pairs, but these are due to components in $\tilde{\mathbf{y}}$ that correspond to equal neighboring values in $\mathbf{y}$ and hence have a distance of at most 2ε. Thus the same estimate as in the first case applies also here.

(5) The proof of submodularity is postponed to Remark 5.

(6) We give a counterexample, where the triangle inequality is not satisfied. Choose $\mathbf{X} = (x_0, x_1, x_2, x_3)$ with $x_j = j$, and let $\mathbf{y}$ and $\mathbf{z}$ be determined by the vectors $\mathbf{y} = (0, 1, -1, 0)^T$ and $\mathbf{z} = (0.6, 1.2, 1.8, 2.4)^T$. For $\mathbf{y}$ we find the sets of persistence pairs $P_1 = \{(x_0, x_1)\}$, $P_2 = \{(x_2, x_3)\}$ and hence $\|\mathbf{y}\|_{per} = |y_1 - y_0| + |y_3 - y_2| = 2$. Since $\mathbf{z}$ is monotone, we find $P_1 = \emptyset$ and $P_2 = \emptyset$ and hence $\|\mathbf{z}\|_{per} = 0$. Finally, for the sum $\mathbf{y} + \mathbf{z} = (0.6, 2.2, 0.8, 2.4)^T$ we obtain $P_1 = \{(x_1, x_2)\}$, $P_2 = \{(x_1, x_2)\}$ yielding $\|\mathbf{y} + \mathbf{z}\|_{per} = 2.8$. Hence, $\mathbf{y}$ and $\mathbf{z}$ do not satisfy the triangle inequality. □

4.2 Relation between discrete total variation and persistence distance

While being not a semi-norm, the persistence distance (together with the sets of persistence pairs) contains a lot of information about the structure of a function $f \in S_1(\mathbf{X})$.

In this section, we show the following close relation to the discrete total variation $TV(f)$.

Theorem 4.6:
Let $\mathbf{X}$ be a partition of the form $a = x_0 < x_1 < \cdots x_N = b$. Then, for each function $f \in S_1(\mathbf{X})$ we have

$$\|f\|_{per} + \max_{x,y \in \mathbf{X}} |f(x) - f(y)| = TV(f),$$

where $TV(f)$ is defined in (3.17). Analogously, for each sequence $\mathbf{y} \in \mathbb{R}^{N+1}$, we have

$$\|\mathbf{y}\|_{per} + \max_{j,k \in \{0,1,\cdots,N\}} |y_j - y_k| = TV(\mathbf{y}).$$

The proof of Theorem 4.6 is based on an iterative topological simplification technique as used e.g. in [5]. Before we can prove this Theorem 4.6, we need the following Lemmata. In the first lemma we show a nesting principle for persistence pairs.

Lemma 4.7:
Let P_1 and P_2 be the two sets of persistence pairs of $f \in S_1(\mathbf{X})$. Let (x_k, x_l) be a persistence pair in $P_1 \cap P_2$. Then, for all $x \in X_m \cup X^m$ with $x_k < x < x_l$, there exists a further knot $\tilde{x} \in X_m \cup X^m$ with $x_k < \tilde{x} < x_l$ such that $(x, \tilde{x})$ or $(\tilde{x}, x)$ is also contained in $P_1 \cap P_2$.

Proof:
The above assertion is in fact a direct conclusion from Algorithm 4.3. Let (x_k, x_l) be a persistence pair in P_1 with $x_k < x_l$, and let us assume without loss of generality, that $x_k \in X_m$, and $x_l \in X^m$, i.e., x_k is a local minimum knot and x_l is a local maximum knot of f. Recalling Algorithm 4.3 it follows that in the iteration step, where $f(x_l) \in Y^m$ is considered, the knot x_k is a direct neighbor of x_l, i.e., there is no other minimum knot $x \in X_{m,\nu}$ left in the interval between x_k and x_l. Hence, if there exists a local maximum knot $x \in X^m$ with $x_k < x < x_l$, then it had been paired with a minimum knot contained

in (x_k, x_l) and pulled out already in an earlier step of the iteration. In particular, it hence corresponds to a local maximum value smaller than (or equal to) $f(x_l)$.

Analogously, since $(x_k, x_l) \in P_2$, the arguments can be repeated for $-f(x_k) \in \tilde{Y}_m$ and $-f(x_l) \in \tilde{Y}^m$. Hence, each knot $x \in X_m$ with $x_k < x < x_l$ is paired with some $\tilde{x} \in X^m$ with $x_k < \tilde{x} < x_l$, and vice versa. Moreover, it cannot happen that one such $x \in X_m \cap (x_k, x_l)$ is paired with different knots $\tilde{x}_1 \neq \tilde{x}_2$ in P_1 and P_2, since the pairing procedure in Algoritm 4.3 is defined uniquely. □

Lemma 4.8:
Let $f \in S_1(\mathbf{X})$ with the sets P_1 and P_2 of persistence pairs, where $P_1 \cap P_2 = \emptyset$. Then, for each persistence pair $(x_k, x_l) \in P_1 \cup P_2$, the values x_k and x_l are neighbor knots in $X_m \cup X^m$.

Proof:
Assume by contrast that there is a persistence pair $(x_k, x_l) \in P_1 \cup P_2$ that does not satisfy this assertion. Without loss of generality let $x_k \in X_m$ and $x_l \in X^m$ and $(x_k, x_l) \in P_1$. Since x_k and x_l are no neighbor knots in $X_m \cup X^m$, there exist (by Lemma 4.7) $\tilde{x}_0 \in X_m$ and $\tilde{x}_1 \in X^m$ with $x_k < \tilde{x}_1 < \tilde{x}_0 < x_l$ that also form a persistence pair $(\tilde{x}_1, \tilde{x}_0)$ in P_1 and such that according to Algorithm 4.3 $f(\tilde{x}_0) \geq f(x_k)$ and $f(\tilde{x}_1) < f(x_l)$. Hence, for $\{-f(x_k)\}_{k=0}^N$, we find similarly as in the proof of Lemma 4.7 that $(\tilde{x}_1, \tilde{x}_0)$ is also a persistence pair in P_2, contradicting the assumption $P_1 \cap P_2 = \emptyset$.

Proof: (of Theorem 4.6)

1. Let $f \in S_1(\mathbf{X})$ be given with local minima and maxima sets X_m, Y_m , X^m, Y^m and sets P_1 and P_2 of persistence pairs. We order all persistence pairs $(x_k, x_l) \in P_1 \cup P_2$ by their distances $|f(x_l) - f(x_k)|$ starting with the smallest. Now we apply the following iterative simplification algorithm to $f^0 := f$. If (x_k, x_l) is the persistence pair in $P_1 \cap P_2$ of f^0 with smallest absolute difference $|f^0(x_l) - f^0(x_k)|$, we determine $f^1 \in S_1(\mathbf{X})$ by

$$f^1(x) = \begin{cases} \dfrac{f^0(x_k) + f^0(x_l)}{2}, & x \in \mathbf{X} \cap [x_k, x_l], \\ f^0(x), & x \in \mathbf{X} \setminus [x_k, x_l]. \end{cases}$$

Hence, since f^0 is monotone in $[x_k, x_l]$, we have changed the total variation by $2|f^0(x_k) - f^0(x_l)|$, i.e.,

$$TV(f^1) = TV(f^0) - 2|f^0(x_k) - f^0(x_l)|.$$

Consider now the change of persistences of f^1. Obviously, since $|f^0(x_l) - f^0(x_k)|$ was the smallest absolute difference, $f^1(x_k)$ and $f^1(x_l)$ are no longer extremal values of f^1 while all other extremal values remain the same compared to f^0. Due to Algorithm 4.3 and Lemma 4.7, f^1 possesses the same persistence pairs as f^0 up to (x_k, x_l), i.e.,

$$\|f^1\|_{per} = \|f^0\|_{per} - 2|f^0(x_k) - f^0(x_l)|.$$

This simplification can now be applied to f^1 removing the next persistence pair (x_k^1, x_l^1) of f^1 with smallest absolute difference $|f(x_k^1) - f(x_l^1)|$ to obtain f^2 etc. If m is the number of persistence pairs in $P_1 \cap P_2$, then, after m simplification steps we obtain $f^m \in S_1(\mathbf{X})$ with

$$\begin{aligned}\|f^m\|_{per} &= \|f^0\|_{per} - 2 \sum_{(x_k,x_l)\in P_1\cap P_2} |f^0(x_k) - f^0(x_l)| \\ &= \sum_{(x_k,x_l)\in P_1\setminus P_2} |f^0(x_k) - f^0(x_l)| + \sum_{(x_k,x_l)\in P_2\setminus P_1} |f^0(x_k) - f^0(x_l)|\end{aligned}$$

and with

$$TV(f^m) = TV(f^0) - 2 \sum_{(x_k,x_l)\in P_1\cap P_2} |f^0(x_k) - f^0(x_l)|. \tag{4.1}$$

Hence, it remains to consider the relation between $\|f\|_{per}$ and $TV(f)$ for a function $f = f^m \in S_1(\mathbf{X})$ with persistence sets P_1, P_2 satisfying $P_1 \cap P_2 = \emptyset$.

2. Assume now that we have $P_1 \cap P_2 = \emptyset$ for $f \in S_1(\mathbf{X})$. By construction of P_1 and P_2 in Algorithm 4.3, we know that each local extremum knot x_k that is not a global extremum and not a boundary knot (i.e., $x_0 < x_k < x_{max}$), occurs in two persistence pairs, one in P_1 and one in P_2. By assumption and Lemma 4.8, these two persistence pairs are different and connect x_k with its two spatial extremum knot neighbors. Further, each boundary knot (x_0 and x_{max}) that is not a global extremum, occurs in one persistence pair, namely in P_1 if it is a local minimum and in P_2 if it is a local maximum. This persistence pair connects the boundary knot with its spatial extremum knot neighbor. If x_k is a global extremum (maximum or minimum) knot of f and not a boundary knot, then it occurs in only one persistence pair (in P_1 if being the global maximum or in P_2 otherwise). By Lemma 4.8, this persistence pair connects x_k with one of its spatial extremum knot neighbors.

But since all other knots are already connected with their spatial extremum knot neighbors by persistence pairs, there is only one "connection" missing, namely between the global minimum knot and the global maximum knot. Hence, we obviously have

$$\|f\|_{per} = TV(f) - \max_{(x,y)\in X} |f(x) - f(y)|.$$

Finally, if x_k is a global extremum of f and a boundary knot, then it does not occur in any persistence pair. Also in this case, we can repeat the argument, observing that only the connection between the global minimum knot and global maximum knot is missing. Together with (4.1) the assertion of the theorem follows. □

Example 14:
We consider $\mathbf{X} = \{x_j\}_{j=0}^8$ with $x_j = j$, and let $f \in S_1(\mathbf{X})$ be given by the vector $\{f(x_j)\}_{j=0}^8 = (1,0,3,2,5,1,4,0,2)$, see Figure 4.2 (a). We obtain the sets

$$\begin{aligned} X_m &= \{x_1, x_3, x_5, x_7\}, \qquad X^m = \{x_0, x_2, x_4, x_6, x_8\}, \\ P_1 &= \{(x_2, x_3), (x_5, x_6), (x_4, x_7)\}, \\ P_2 &= \{(x_2, x_3), (x_5, x_6), (x_7, x_8), (x_0, x_1)\}, \end{aligned}$$

Hence, we find the persistence distance $\|f\|_{per} = (1+3+5)+(1+3+2+1) = 16$, and the discrete total variation $TV(f) = 21$. After simplification of f according to the procedure described in the proof of Theorem 4.6, where the persistence pairs (x_2, x_3) and (x_5, x_6) in $P_1 \cap P_2$ are removed, we obtain f^2 with $\{f^2(x_j)\}_{j=0}^8 = (1, 0, 2.5, 2.5, 5, 2.5, 2.5, 0, 2)^T$, see Figure 4.2 (b). For f^2, we regard

$$\begin{aligned} X_m &= \{x_1, x_7\}, \qquad X^m = \{x_0, x_4, x_8\} \\ P_1 &= \{(x_4, x_7)\}, \qquad P_2 = \{(x_0, x_1), (x_7, x_8)\}, \end{aligned}$$

such that $\|f^2\|_{per} = 5 + 1 + 2 = 8$ and $TV(f^2) = 13$.

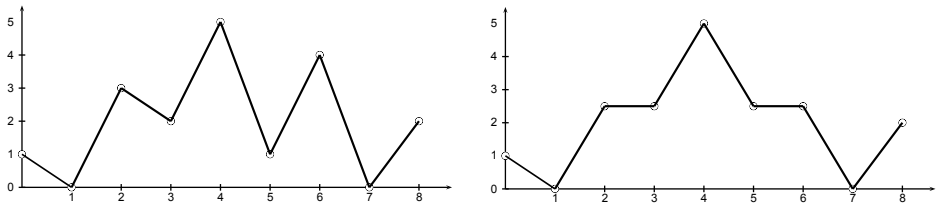


Fig. 4.2: (a) Illustration of the spline functions f and (b) of f^2 in Example 14.

Remark 5:
The submodularity of the persistence distance stated in Theorem 4.5 follows now directly from the submodularity of the discrete total variation in Theorem 3.5. For all $\mathbf{y}, \mathbf{z} \in \mathbb{R}^{N+1}$ we find with $y^m := \max\{y_j : j = 0, \ldots, N\}$, $y_m := \min\{y_j : j = 0, \ldots, N\}$, and $z^m := \max\{z_j : j = 0, \ldots, N\}$, $z_m := \min\{z_j : j = 0, \ldots, N\}$,

$$\begin{aligned} & \|\mathbf{y}\|_{per} + \|\mathbf{z}\|_{per} = TV(\mathbf{y}) + TV(\mathbf{z}) - (y^m - y_m) - (z^m - z_m) \\ \geq\ & TV(\max(\mathbf{y}, \mathbf{z})) + TV(\min(\mathbf{y}, \mathbf{z})) - y^m + y_m - z^m + z_m \\ =\ & TV(\max(\mathbf{y}, \mathbf{z})) + TV(\min(\mathbf{y}, \mathbf{z})) - (\max\{y^m, z^m\} \\ & - \max\{y_m, z_m\}) - (\min\{y^m, z^m\} - \min\{y_m, z_m\}) \\ =\ & \|\max(\mathbf{y}, \mathbf{z})\|_{per} + \|\min(\mathbf{y}, \mathbf{z})\|_{per}. \end{aligned}$$

4.3 Persistence based simplification

The iterative simplification procedure that has been applied in the proof of Theorem 4.6 iteratively removes local minima and maxima of the function f (respective the vector $\mathbf{y} = (f(x_j))_{j=0}^N$). Thus, this procedure simplifies the topological structures of f in a simple manner. At each iteration step, the persistence pair corresponding to the smallest distance of function values is removed, and at the same time, the function will be smoothed by inserting a plateau part, i.e., a piecewise constant part. Comparing the sequence of obtained simplified functions f^j, $j = 0, 1, \ldots$ with $f^0 = f$, the corresponding change of the norm $\|f_j - f_{j+1}\|$ does only depend on the distance of the function values that corresponds to the persistence pair that has been removed at that iteration step. Thus the procedure can be also seen as a smoothing procedure, where the most significant parts will be removed only at the end. Inserting a suitable stopping criteria and removing only the persistence pairs corresponding to a persistence (distance of function values) being smaller that a fixed threshold $\varepsilon > 0$, the idea can be taken already as a denoising method, see e.g. [7, 23, 27].

The simplification approach for functions on surfaces considered in [3] is optimal in the sense that the obtained simplification f_δ has a minimum number of critical points and satisfies the condition $\|f_\delta - f\|_\infty \leq \delta$.

We pay special attention to the simplification of one-dimensional piecewise linear functions f in [5] for the purpose of comparison. The simplification procedure in [5] successively removes the persistence pairs of the set P_1 whose persistence is under certain threshold ε by setting values of all relevant points in the persistence interval to an average of the function values of the pair. A further idea in [5] replaces the persistence pairs by so-called filling pairs and the the average of the two function values $(f^j(x_k) - f^j(x_\ell))/2$ by the filling volume.

5 Application of persistence distance to signal denoising

Having found the close relationship between persistence distance and discrete total variation, we want to explore some ideas of how this relationship can be applied to signal denoising.

We connect Theorem 4.6 with one of the most famous and successful models for signal denoising —the celebrated Rudin-Osher-Fatemi model [37]— by noticing that its regularization term is the discrete total variation.

Considering the discrete setting, let $f \in S_1(\mathbf{X})$ be the noise contaminated version of a clean signal $u \in S_1(\mathbf{X})$, i.e.,

$$f(x_j) = u(x_j) + n(x_j), \qquad x_j \in \mathbf{X},$$

where $(n(x_j))_{x_j \in \mathbf{X}}$ denotes a vector of i.i.d. random variables simulating white noise, with mean value zero and variance σ^2. In order to reconstruct $\mathbf{u} = (u(x_j))_{j=0}^N$, it is proposed to minimize the functional

$$J(\mathbf{u}) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{j=0}^{N-1} |u(x_{j+1}) - u(x_j)|,$$

where the second term coincides with the discrete total variation $TV(u)$ in Definition (3.17). The above functional $J(\mathbf{u})$ is strictly convex but not differentiable. The parameter $\lambda > 0$ balances the regularization term $TV(u)$ and the data fitting term, and a suitable choice of λ is crucial for the success of the method.

The main advantage of the ROF model in comparison to other models involving a smoother regularization term is its ability to preserve sharp changes in the data.

From Theorem 4.6 it follows that the ROF functional can also be written as

$$\begin{aligned} J(\mathbf{u}) &= \frac{\lambda}{2}\sum_{j=0}^{N}|u(x_j)-f(x_j)|^2 + \|u\|_{per} + \max_{x,\tilde{x}\in\mathbf{X}}|u(x)-u(\tilde{x})| \\ &= \frac{\lambda}{2}\sum_{j=0}^{N}|u(x_j)-f(x_j)|^2 + \sum_{(x,\tilde{x})\in P_1}|u(x)-u(\tilde{x})| \\ &\quad + \sum_{(x,\tilde{x})\in P_2}|u(x)-u(\tilde{x})| + \max_{j,k\in\{0,\dots,N\}}|u(x_j)-u(x_k)|. \end{aligned}$$

In contrast to the total variation $TV(u)$, the persistence distance consists of a sum of distances of function values being local extrema of the function u, i.e., describing the topological properties of the function u, where small distances $|u(x)-u(\tilde{x})|$ (being related to small pairs $(x,\tilde{x})$) correspond to oscillatory behavior like noise while the large distances $|u(x)-u(\tilde{x})|$ describe the important features of the function u. Let for simplicity

$$P(u) = P = P_1 \cup P_2 \cup \{(x,\tilde{x})\}$$

be the set of all (persistence) pairs, where $(x,\tilde{x})$ denotes the pair of knots whose corresponding function values are the global minimum and the global maximum of f. Here, as before, pairs in $P_1 \cap P_2$ occur twice in the multiset $P(u)$.

5.1 Weighted ROF-model based on the persistence distance

We propose to consider the new weighted functional

$$\tilde{J}(\mathbf{u}) = \frac{\lambda}{2}\sum_{j=0}^{N}|u(x_j)-f(x_j)|^2 + \sum_{(x_j,\tilde{x}_j)\in P(u)}\alpha_j(\mathbf{u})|u(x_j)-u(\tilde{x}_j)|, \tag{5.1}$$

where $\alpha_j = \alpha_j(\mathbf{u}) = \alpha(u,x_j,\tilde{x}_j)$ depends on the persistence $|u(x_j)-u(\tilde{x}_j)|$. It should be large for small distances $|u(x_j)-u(\tilde{x}_j)|$ and rather small for large distances $|u(x_j)-u(\tilde{x}_j)|$. A suitable choice of weights α_j enables us to ensure that the denoised signal $\mathbf{u}$ obtained by minimization of $\tilde{J}(\mathbf{u})$ keeps the essential features of $\mathbf{f} = (f(x_j))_{j=0}^N$, and in particular preserves the significant extremum values of $\mathbf{f}$. From Lemma 4.7 and Lemma 4.8 it follows that there is a special structure of persistence pairs of a function $u \in S_1(\mathbf{X})$. We regard persistence pairs that occur only once, i.e., being not contained in $P_1(u)\cap P_2(u)$, as **single persistence pairs**, while persistence pairs in $P_1(u)\cap P_2(u)$ are denoted as **double persistence pairs**. Single persistence pairs have a special importance for the function structure, and the corresponding function values can be seen as **significant extremum**

values of u. The pair $(\operatorname{argmin}_{x\in\mathbf{X}_m} u(x), \operatorname{argmax}_{x\in\mathbf{X}^m} u(x))$ of global minimum and maximum knots will be handled like a single persistence pair by introducing the set

$$S(u) := P(u) \setminus (P_1(u) \cap P_2(u)).$$

We observe that for each interval $[x_l, x_{l+1}]$ formed by neighboring knots in the knot set $\mathbf{X}$, there exists a unique chain of interlacing intervals corresponding to persistence pairs, such that

$$[x_l, x_{l+1}] \subseteq [x_1^l, \tilde{x}_1^l] \subset \ldots \subset [x_{r(l)}^l, \tilde{x}_{r(l)}^l], \tag{5.2}$$

where $(x_\nu^l, \tilde{x}_\nu^l) \in P_1(u) \cap P_2(u)$ for $\nu = 1, \ldots, r(l)-1$ and $(x_{r(l)}^l, \tilde{x}_{r(l)}^l) \in S(u)$. If already $(x_l, x_{l+1}) \in S(u)$ then the chain collapses to this one interval and we have $[x_l, x_{l+1}] = [x_1^l, \tilde{x}_1^l]$, i.e., $r(l) = 1$. Analogously, for each double persistence pair $(x_j, \tilde{x}_j)$, there exists a unique chain of interlacing persistence intervals that contains $[x_j, \tilde{x}_j]$. We say that this double persistence pair has the **order** $k = k(x_j, \tilde{x}_j)$ if there are k further "persistence" intervals containing the interval $[x_j, \tilde{x}_j]$. In particular, all pairs in $S(u)$ are of order $k = 0$. We call (5.2) a **pair chain** with an abuse of pair notation $(\cdot, \cdot)$ and its corresponding closed interval $[\cdot, \cdot]$.

Example 15:
Considering again Example 14, see also Figure 4.2, with

$$\begin{aligned}
X_m &= \{x_1, x_3, x_5, x_7\}, \qquad X^m = \{x_0, x_2, x_4, x_6, x_8\},\\
P_1 &= \{(x_2, x_3), (x_5, x_6), (x_4, x_7)\},\\
P_2 &= \{(x_2, x_3), (x_5, x_6), (x_7, x_8), (x_0, x_1)\},\\
P(f) = P &= P_1 \cup P_2 \cup \{(x_1, x_4)\},\\
P_1 \cap P_2 &= \{(x_2, x_3),\ (x_5, x_6)\},\\
S(f) = S &= \{(x_0, x_1), (x_1, x_4), (x_4, x_7), (x_7, x_8)\}.
\end{aligned}$$

Then we obtain the following simple chains of interlacing persistence intervals

l	$r(l)$	chain of $[x_l, x_{l+1}]$
0	1	$[x_0, x_1]$
1	1	$[x_1, x_2] \subset [x_1, x_4]$
2	2	$[x_2, x_3] \subseteq [x_2, x_3] \subset [x_1, x_4]$
3	1	$[x_3, x_4] \subset [x_1, x_4]$
4	1	$[x_4, x_5] \subset [x_4, x_7]$
5	2	$[x_5, x_6] \subseteq [x_5, x_6] \subset [x_4, x_7]$
6	1	$[x_6, x_7] \subset [x_4, x_7]$
7	1	$[x_7, x_8] \subseteq [x_7, x_8]$

Table 5.1: Pair chains of the intervals in Example 15.

For the smoothing algorithm, we want to consider not only the local behavior of the function around the interval $[x_l, x_{l+1}]$ but also the function structure in the corresponding chain. We may have some a-priori information on the structure of the original signal regarding the number of levels in the above chains in order to judge which persistence pairs indeed represent important features. Note that our theoretical observations in Theorem 5.1 will be true for arbitrary choices of the weights $\alpha_j(\mathbf{u})$ in (5.1). One possibility is to choose for every (persistence) pair $(x_j, \tilde{x}_j) \in P(u)$ the weight

$$\alpha_j(\mathbf{u}) = \frac{1}{1 + \eta|u(\tilde{x}_j) - u(x_j)|} \tag{5.3}$$

with some suitable $\eta > 0$. In this way, the weight is rather small for large distances $|u(\tilde{x}_j) - u(x_j)|$ and approximately 1 for small distances. On the other side, we choose the weights in a way such that weights are numerically stable for the whole range of possible distances for a given function. We will obey the above principles to set up also other weight formulas in our experiments.

The proposed new functional $\tilde{J}(u)$ in (5.1) is highly nonlinear. For the numerical evaluation, we show that $\tilde{J}(u)$ can also be seen as a weighted ROF-functional.

Theorem 5.1:
Consider for each interval $[x_l, x_{l+1}]$, $l = 0, \ldots, N-1$, the corresponding complete chain of persistence intervals of u such that $[x_l, x_{l+1}] \subseteq [x_1^l, \tilde{x}_1^l] \subset \ldots \subset [x_{r(l)}^l, \tilde{x}_{r(l)}^l]$, and denote by $\alpha_\nu^l(\mathbf{u}) = \alpha_\nu^l$, $\nu = 1, \ldots, r(l)$, the weight in the functional $\tilde{J}(u)$ in (5.1) corresponding to the persistence pair $(x_\nu^l, \tilde{x}_\nu^l)$. Then the weighted functional $\tilde{J}(\mathbf{u})$ in (5.1) is equivalent to the weighted ROF functional

$$J_w(\mathbf{u}) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{l=0}^{N-1} w_l(\mathbf{u}) |u(x_{l+1}) - u(x_l)|, \tag{5.4}$$

where

$$w_l(\mathbf{u}) = w_l := \sum_{\nu=1}^{r(l)} (-1)^{\nu-1} \alpha_\nu^l. \tag{5.5}$$

Proof:
The following considerations can be carried out for each single pair $(x, \tilde{x})$ in $S(u)$ separately in order to compute the weights $w_l(\mathbf{u})$ for all $[x_l, x_{l+1}] \subseteq [x, \tilde{x}]$. Therefore we restrict ourselves to one interval $I = [x, \tilde{x}]$ with $(x, \tilde{x}) \in S(u)$.

For each $[x_l, x_{l+1}] \subset I$ we consider the corresponding chain $[x_l, x_{l+1}] \subseteq [x_1^l, \tilde{x}_1^l] \subset \ldots \subset [x_{r(l)}^l, \tilde{x}_{r(l)}^l] = [x, \tilde{x}]$ and apply the following procedure. If $r(l) = 1$, we find $w_l(\mathbf{u}) = \alpha_{r(l)}^l = \alpha_1^l$ as the weight corresponding to $(x, \tilde{x})$ that has to be assigned to the term $|u(x_{l+1}) - u(x_l)|$. For $r(l) \geq 2$, we consider the adjacent smaller persistence interval $[x_{r(l)-1}^l, \tilde{x}_{r(l)-1}^l]$ being a subinterval of $[x_{r(l)}^l, \tilde{x}_{r(l)}^l] = [x, \tilde{x}]$. By construction of

the persistence pairs it follows that either $u(x^l_{r(l)}) > u(\tilde{x}^l_{r(l)})$ and $u(x^l_{r(l)-1}) < u(\tilde{x}^l_{r(l)-1})$ or vice versa. Hence, since $|u(x^l_{r(l)}) - u(\tilde{x}^l_{r(l)})| > |u(x^l_{r(l)-1}) - u(\tilde{x}^l_{r(l)-1})|$ we obtain for $x = x^l_{r(l)} < x^l_{r(l)-1} < \tilde{x}^l_{r(l)-1} < \tilde{x}^l_{r(l)} = \tilde{x}$ that

$$\begin{aligned} &|u(x) - u(\tilde{x})| = |u(x^l_{r(l)}) - u(\tilde{x}^l_{r(l)})| \\ &= |u(x^l_{r(l)}) - u(x^l_{r(l)-1})| - |u(x^l_{r(l)-1}) - u(\tilde{x}^l_{r(l)-1})| \\ &+|u(\tilde{x}^l_{r(l)-1}) - u(\tilde{x}^l_{r(l)})|. \end{aligned} \tag{5.6}$$

Multiplying $\alpha^l_{r(l)}(\mathbf{u})$ on both sides of (5.6) it follows that

$$\begin{aligned} & \alpha^l_{r(l)}(\mathbf{u})|u(x^l_{r(l)}) - u(\tilde{x}^l_{r(l)})| \\ = \; & \alpha^l_{r(l)}(\mathbf{u})|u(x^l_{r(l)}) - u(x^l_{r(l)-1})| - \alpha^l_{r(l)}(\mathbf{u})|u(x^l_{r(l)-1}) \\ & -u(\tilde{x}^l_{r(l)-1})| + \alpha^l_{r(l)}(\mathbf{u})|u(\tilde{x}^l_{r(l)-1}) - u(\tilde{x}^l_{r(l)})| \end{aligned}$$

thereby showing how the weighted difference $\alpha^l_{r(l)}(\mathbf{u})|u(x^l_{r(l)}) - u(\tilde{x}^l_{r(l)})|$ can be rewritten for the three subintervals of $[x^l_{r(l)}, \tilde{x}^l_{r(l)}]$. Hence, the new coefficient corresponding to the term $|u(x^l_{r(l)-1}) - u(\tilde{x}^l_{r(l)-1})|$ is now the sum of its original coefficient $\alpha^l_{r(l)-1}(\mathbf{u})$ and $-\alpha^l_{r(l)}(\mathbf{u})$. In case of $r(l) = 2$ we thus obtain the weight $w_l(\mathbf{u}) = \alpha^l_{r(l)-1}(\mathbf{u}) - \alpha^l_{r(l)}(\mathbf{u})$.

For $r(l) > 2$, the same argument can be applied to the persistence pairs $(x^l_{r(l)-1}, \tilde{x}^l_{r(l)-1})$ and $(x^l_{r(l)-2}, \tilde{x}^l_{r(l)-2})$, and we find the new coefficient for the term $|u(x^l_{r(l)-2}) - u(\tilde{x}^l_{r(l)-2})|$ as the sum of $\alpha_{r(l)-2}$ and $-(-\alpha_{r(l)}(\mathbf{u}) + \alpha_{r(l)-1}(\mathbf{u}))$. We repeat this argument until the smallest interval $[x^l_1, \tilde{x}^l_1]$ in the chain of $[x_l, x_{l+1}]$ is reached and obtain the coefficient $w_l(\mathbf{u})$ of $|u(x^l_1) - u(\tilde{x}^l_1)|$ of the form

$$w_l(\mathbf{u}) = \sum_{\nu=1}^{r(l)} (-1)^{\nu-1} \alpha^l_\nu. \tag{5.7}$$

This is exactly the weight that we have to assign to $|u(x_{l+1}) - u(x_l)|$. □

5.2 Numerical algorithm

In this section, we propose a numerical scheme to minimize the nonlinear weighted TV-functional $J_w(\mathbf{u})$ in (5.4). Let $\mathbf{X}$ be an equidistant partition of the interval $[a, b]$, i.e. $x_j := a + (b-a)j/N$, $j = 0, \ldots, N$. Let $f_j := f(x_j)$ be the given noisy values and let $u_j := u(x_j)$ be the values of the wanted denoised signal for $j = 0, \ldots, N$.

We now generalize the simplified first-order iteration filter described in Chapter 2 to the weighted TV-functional $J_w(\mathbf{u})$ in (5.4).

Starting with $\mathbf{u}^0 = (u_j^0)_{j=0}^N := (f_j)_{j=0}^N$, we iteratively apply the following data-dependent filter

$$u_j^{k+1} = \frac{\lambda f_j + g_{1,j}(\mathbf{u}^k)\, u_{j-1}^k + g_{2,j}(\mathbf{u}^k)\, u_{j+1}^k}{\lambda + g_{1,j}(\mathbf{u}^k) + g_{2,j}(\mathbf{u}^k)} \tag{5.8}$$

for $j = 1, \ldots, N-1$, where the filter coefficients $g_{1,j}(\mathbf{u}^k)$ and $g_{2,j}(\mathbf{u}^k)$ are given by

$$g_{1,j}(\mathbf{u}^k) := \begin{cases} w_{j-1}(\mathbf{u}^k)/|u_j^k - u_{j-1}^k| & \text{for } |u_j^k - u_{j-1}^k| \neq 0, \\ 0 & \text{else,} \end{cases}$$

$$g_{2,j}(\mathbf{u}^k) := \begin{cases} w_j(\mathbf{u}^k)/|u_j^k - u_{j+1}^k| & \text{for } |u_j^k - u_{j+1}^k| \neq 0, \\ 0 & \text{else.} \end{cases}$$

For $j = 0$ and $j = N$, the filter (5.8) is simplified by assuming that $u_{-1}^k = u_{N+1}^k = 0$ and $g_{1,0}(\mathbf{u}^k) = g_{2,N}(\mathbf{u}^k) = 0$.

Here $w_j(\mathbf{u}^k)$, $j = 0, \ldots, N-1$, are the weights of the functional $J_w(\mathbf{u})$ in (5.5). In order to stabilize the procedure and to reduce the computational costs, we will compute the weights $w_j(\mathbf{u}^k)$ not in each iteration step but employ an outer iteration to recompute the persistence weights and an inner iteration, where we apply several filtering steps with fixed weights. The complete algorithm is given as follows.

Algorithm 5.2:

Input: noisy vector $\mathbf{f}$, parameters λ and η.

1) Initialize $\mathbf{u}^0 = \mathbf{f}$.
2) **For** $k = 1, \ldots, n_{outer}$ **do**
   Compute the persistence weights $w_j(\mathbf{u}^k)$ in (5.5) and (5.3) for $j = 0, \ldots, N-1$.
   Initialize $\mathbf{u}^{k,0} := \mathbf{u}^k$, i.e., $u_j^{k,0} := u_j^k$ for $j = 0, \ldots, N$.
   For $\ell = 1, \ldots, n_{inner}$ **do**
      Compute for $j = 0, \ldots, N$
      $u_j^{k,\ell+1} = \frac{\lambda f_j + g_{1,j}(\mathbf{u}^{k,\ell})\, u_{j-1}^{k,\ell} + g_{2,j}(\mathbf{u}^{k,\ell})\, u_{j+1}^{k,\ell}}{\lambda + g_{1,j}(\mathbf{u}^{k,\ell}) + g_{2,j}(\mathbf{u}^{k,\ell})}$,
      where as before $g_{1,j}(\mathbf{u}^{k,\ell}) := w_{j-1}(\mathbf{u}^k)/|u_j^{k,\ell} - u_{j-1}^{k,\ell}|$
      for $|u_j^{k,\ell} - u_{j-1}^{k,\ell}| \neq 0$ (and $g_{1,j}(\mathbf{u}^{k,\ell}) := 0$ else)
      $g_{2,j}(\mathbf{u}^{k,\ell}) := w_j(\mathbf{u}^k)/|u_j^{k,\ell} - u_{j+1}^{k,\ell}|$ for $|u_j^{k,\ell} - u_{j+1}^{k,\ell}| \neq 0$
      (and $g_{2,j}(\mathbf{u}^{k,\ell}) := 0$ else).
   end
   Put $\mathbf{u}^{k+1} := \mathbf{u}^{k,n_{inner}}$.
end

Output: $\mathbf{u} = \mathbf{u}^{n_{outer}+1}$ approximates the minimizer of $\min_u J_w(u)$.

Observe that the functional $J_w(\mathbf{u})$ is not necessarily convex. Within the inner iterations, where the weights in the second term of the functional are fixed, we have applied a simple

gradient method that is justified by the next theorem. Of course, the filter in (5.8) for computing $u_j^{k,\ell+1}$ in Algorithm 5.2 can be also be replaced by slightly smoother versions as e.g. in [13].

Theorem 5.3:
Let $\mathbf{X} = \{x_j\}_{j=0}^N$ with $x_j = a + (b-a)j/N$. For given $f \in S_1(\mathbf{X})$ and $u^k \in S_1(\mathbf{X})$, let $J_w(\mathbf{u}, \mathbf{u}^k)$ be the weighted ROF-functional

$$J_w(\mathbf{u}, \mathbf{u}^k) := \frac{\lambda}{2} \sum_{j=0}^{N} |u(x_j) - f(x_j)|^2 + \sum_{l=0}^{N-1} w_l(\mathbf{u}^k)|u(x_{l+1}) - u(x_l)| \tag{5.9}$$

with fixed weights $w_l(\mathbf{u}^k)$. If the sequence $(\mathbf{u}^{k,\ell})_{\ell=0}^{\infty}$ of function vectors obtained by the inner iteration in Algorithm 5.2 converges to some function vector $\mathbf{u}$, then $\mathbf{u}$ satisfies

$$\frac{\partial J_w(\mathbf{u}, \mathbf{u}^k)}{\partial u(x_j)} = 0 \quad \text{for} \quad j = 0, \ldots, N. \tag{5.10}$$

Proof:
We first consider $j = 1, \ldots, N-1$. The limit $\mathbf{u}$ of the sequence $(\mathbf{u}^{k,\ell})_{\ell=0}^{\infty}$ given by the inner iteration in Algorithm 5.2 satisfies

$$[\lambda + g_{1,j} + g_{2,j}]u_j = \lambda f_j + g_{1,j}u_{j-1} + g_{2,j}u_{j+1}$$

with $g_{1,j} := w_{j-1}(\mathbf{u}^k)/|u_j - u_{j-1}|$ for $|u_j - u_{j-1}| \neq 0$ (and $g_{1,j} = 0$ else) as well as $g_{2,j} = w_j(\mathbf{u}^k)/|u_j - u_{j+1}|$ for $|u_j - u_{j+1}| \neq 0$ (and $g_{2,j} = 0$ else). Hence

$$\lambda(u_j - f_j) + g_{1,j}(u_j - u_{j-1}) + g_{2,j}(u_j - u_{j+1}) = 0.$$

Incorporating the definitions of $g_{1,j}$ and $g_{2,j}$ we find

$$\begin{aligned} &\lambda(u_j - f_j) + w_{j-1}(\mathbf{u}^k)\text{sign}(u_j - u_{j-1}) \\ &+w_j(\mathbf{u}^k)\text{sign}(u_j - u_{j+1}) = 0 \end{aligned}$$

and the assertion follows, where for $u_j - u_{j-1} = 0$ (resp. $u_j - u_{j+1} = 0$) the definition of a subdifferential has to be applied. The special cases $j = 0$ and $j = N$ are obtained analogously. □

5.3 Numerical experiments

In this section, we study the performance of our proposed method which in the following is called persistence denoising method. We employ the two test signals consisting of piecewise smooth functions in Figure 5.1 taken from the WaveLab toolbox (see `http://statweb.stanford.edu/~wavelab`), where the original signals of length 1000

are shown in the first row, and the noisy signals (perturbation with white Gaussian noise and uniform noise) are shown in the second row and third row of Figure 5.1, respectively.

In the experiments, we would like to investigate the performance of the new denoising scheme, thereby comparing different setups with respect to noise type and weight strategy. We employ different strategies of setting up the weights, besides the one used in (5.3), and consider two different kinds of noise – Gaussian noise and uniform noise. The following subsections will be arranged according to weight choosing strategies.

We compare the denoising performance of our method (using the persistence-weighted ROF functional minimized with the first-order iteration filter, called persistence denoising) with the non-weighted ROF model minimized by the first-order iteration filter (called FOF method), and with the denoising algorithm proposed in [5] (called persistence simplification). As an alternative minimization method for the ROF model, we also compare with the performance of the primal-dual method by Chambolle and Pock [12].

As mentioned above, we are faced with the problem how to make use of the additional information contained in the hierarchical structures in the pair chains and the persistences corresponding to every pair.

5.3.1 Weight strategy 1 (WS1) of persistence denoising

As already proposed in (5.3), we choose the weights

$$\alpha_j(\mathbf{u}) = \frac{1}{1+\eta|u(\tilde{x}_j) - u(x_j)|}.$$

We can assume that pairs with large persistence $|u(\tilde{x}_j) - u(x_j)|$ usually belong to the main structures of a function u that we want to keep during the iteration while pairs with small persistence are likely caused by noise. Therefore, we set smaller weights $\alpha_j(\mathbf{u})$ in (5.1) for persistence pairs with larger persistence and larger weights for noise-like pairs as in (5.3). Furthermore, in (5.3) the importance of persistences is weighted by η. We call $\alpha_j(\mathbf{u}) = \frac{1}{1+\eta|u(\tilde{x}_j)-u(x_j)|}$ **persistence factor**.

We would like to comment on the principle of setting the regularization parameters λ in (5.1) and η in $\alpha_j(\mathbf{u})$. For decreasing η also the influence of the different persistences decreases. For $\eta = 0$, we simply have $\alpha_j(\mathbf{u}) = 1$ and thus $w_\ell(\mathbf{u}) = 1$ for odd $r(\ell)$ and $w_\ell(\mathbf{u}) = 0$ for even $r(\ell)$. Considering e.g. again Example 15, see also Figure 4.2, this means that we consider the total variation of the simplified function f_2 instead of the total variation of f in the ROF-functional. Thus, for small η we again obtain similar results as for the usual ROF-model. In this case we need to increase λ to maintain the values of peaks.

5.3.2 Weight strategy 2 (WS2) of persistence denoising

Notice that noise is usually of higher oscillation and corresponds to pair chains with higher order while persistence pairs with a lower order indicate the more important structure of the function. We want to make use of the hierarchical information contained in the pair chains in (5.2) and propose to choose the weight for a pair $(x_j, \tilde{x}_j)$ as follows

$$\alpha_j(\mathbf{u}) = (k_j + 1)\tau \frac{1}{1 + \eta|u(\tilde{x}_j) - u(x_j)|}, \tag{5.11}$$

where $k = k_j = k(x_j, \tilde{x}_j)$ is the order of the pair $(x_j, \tilde{x}_j)$ in its chain of pairs and $\tau > 1$. We call the new factor $(k_j + 1)\tau$ **hierarchical factor**. By this strategy, a persistence pair with a higher order k achieves a larger factor $(k + 1)\tau$ (and thus a larger weight) than a persistence pair with lower order k even though those two pairs may share the same persistence. This scheme makes sure that the function values corresponding to the noise-like small pairs with higher order chains can be flattened more effectively and faster during the iteration. In our numerical experiments, $\tau = 3$ has been used, i.e., $\alpha_j(\mathbf{u}) = 3(k_j + 1)\frac{1}{1+\eta|u(\tilde{x}_j)-u(x_j)|}$. By fixing τ we reduce the complexity of optimization problem. Of course, one can choose other values for τ.

5.3.3 Denoising results for Gaussian noise contaminated signals

For signals contaminated by Gaussian noise, Figures 5.2(a) and (b) show the denoising results obtained by the FOF method with 70000 iterations using $\lambda = 0.04$ for the first and $\lambda = 0.15$ for the second test signal. Despite its denoising performance, there remain small oscillations in smooth parts of the signal. Taking a smaller parameter λ, one obtains a stronger smoothing effect while the significant peaks of the signals will be smoothed out even further. Figures 5.2(c) and (d) are obtained by using the persistence simplification in [5] with thresholds 23.2 and 31.5, respectively. It performs well in some constant regions but suffers from strong oscillations at other smooth parts of the signals. This procedure keeps the significant peaks of the signal well and even overshoots, i.e. does not denoise at peak points.

The results obtained by the primal-dual method are shown in Figures 5.2(e) and (f) with $\lambda = 0.03$ and $\lambda = 0.15$, respectively.

The results of our persistence denoising approach with WS1 using Algorithm 5.2 are presented in Figures 5.3 (a) and (b). Figure 5.3(a) is obtained with $\lambda = 0.02$, $\eta = 0.003$ and using 100 outer iterations and 700 inner iterations. Figure 5.3(b) is found with $\lambda = 0.006$, $\eta = 0.57$ with 100 outer and 700 inner iterations. The results of persistence denoising with WS2 are shown in Figures 5.3 (c) and (d) with parameters given in Table 5.3.

Furthermore, we employ the weighted primal-dual method in Algorithm 3.8 with WS1 to denoise the first and the second signal, respectively. Figure 5.7 (e) is obtained with $\lambda = 0.012$ and $\eta = 0.013$. Figure 5.7 (f) is obtained with $\lambda = 0.0795$ and $\eta = 0.021$.

As desired, the new algorithm keeps the significant features of the test signals very well and performs better for denoising in flat regions than the persistence simplification. Unfortunately, some isolated noisy points near jumps remain after denoising. This phenomenon is caused by significant persistence pairs (single pairs) that occur by pairing of local minima and maxima of the noisy function that are beyond the genuine local minima and maxima of the function. The denoising performance of the new algorithm can be strongly improved by application of a local median filter in a post-processing step.

For a better comparison, Figure 5.4 shows the results of the six methods for local regions, particularly for flat regions (see (a) and (c)) and for peak denoising (see (b) and (d)). These illustrations nicely show that our new method is able to perform better than the other two methods regarding denoising in flat regions as well as regarding correct peak preservation.

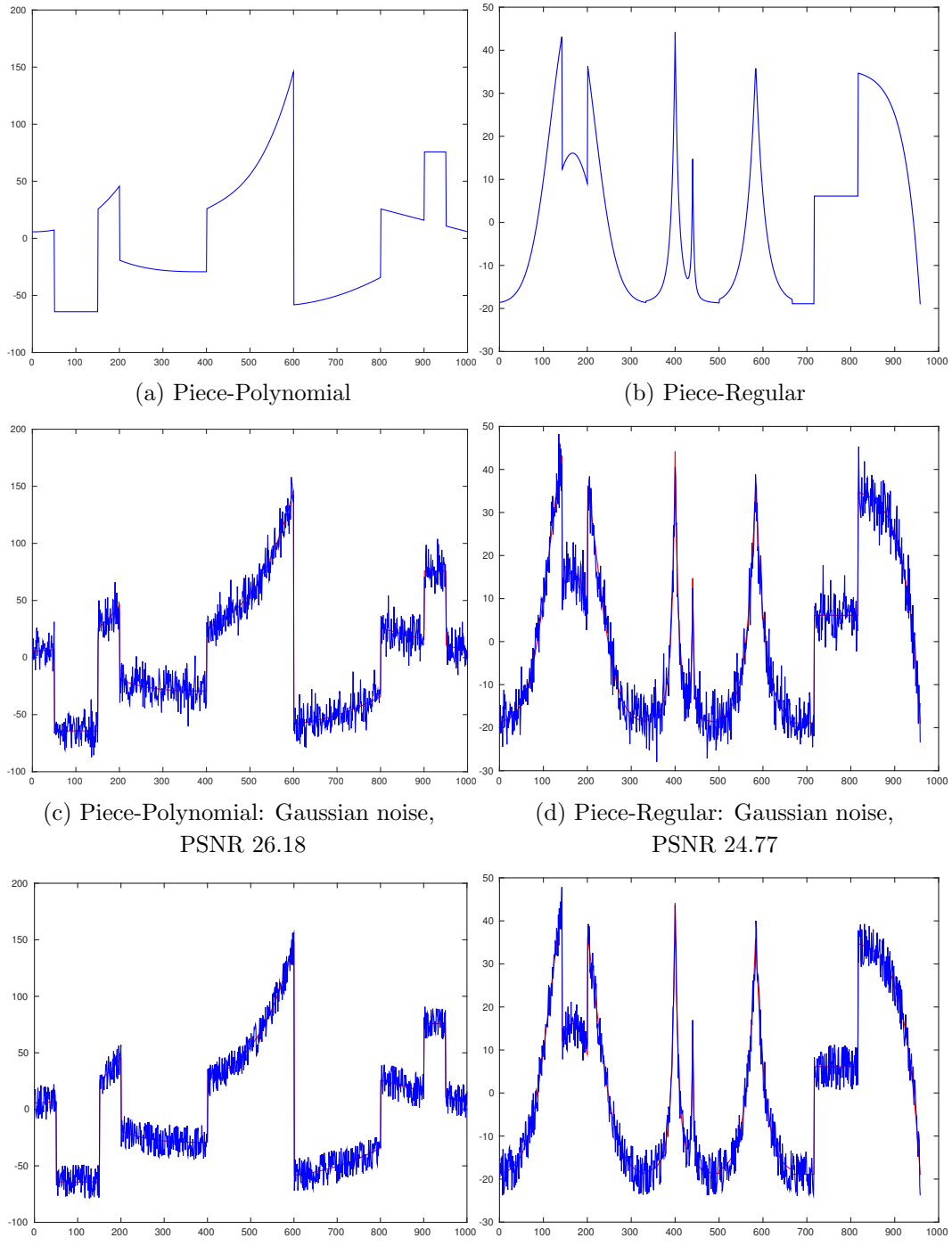
We summarize the denoising parameters and results for Gaussian noisy signals in Tables 5.2 and 5.3.

Method	iteration	λ	η	threshold	PSNR
Noisy signal					26.18
FOF	70000	0.04			33.79
Persistence simplification				23.2	32.08
Primal-dual method	70000	0.03			38.41
Persistence denoising with WS1	70000	0.002	0.003		37.11
Persistence denoising with WS2	70000	0.016	0.014		37.90
Weighted Primal-dual with WS1	70000	0.012	0.013		38.43

Table 5.2: Denoising parameters and denoising results for the first signal with Gaussian noise.

Method	iteration	λ	η	threshold	PSNR
Noisy signal					24.77
FOF	70000	0.15			31.21
Persistence simplification				31.5	30.95
Primal-dual	70000	0.015			32.50
Persistence denoising with WS1	70000	0.006	0.57		32.56
Persistence denoising with WS2	70000	0.3	0.0001		32.22
Weighted Primal-dual with WS1	70000	0.0795	0.021		32.65

Table 5.3: Denoising parameters and denoising results for the second signal with Gaussian noise.



(a) Piece-Polynomial

(b) Piece-Regular

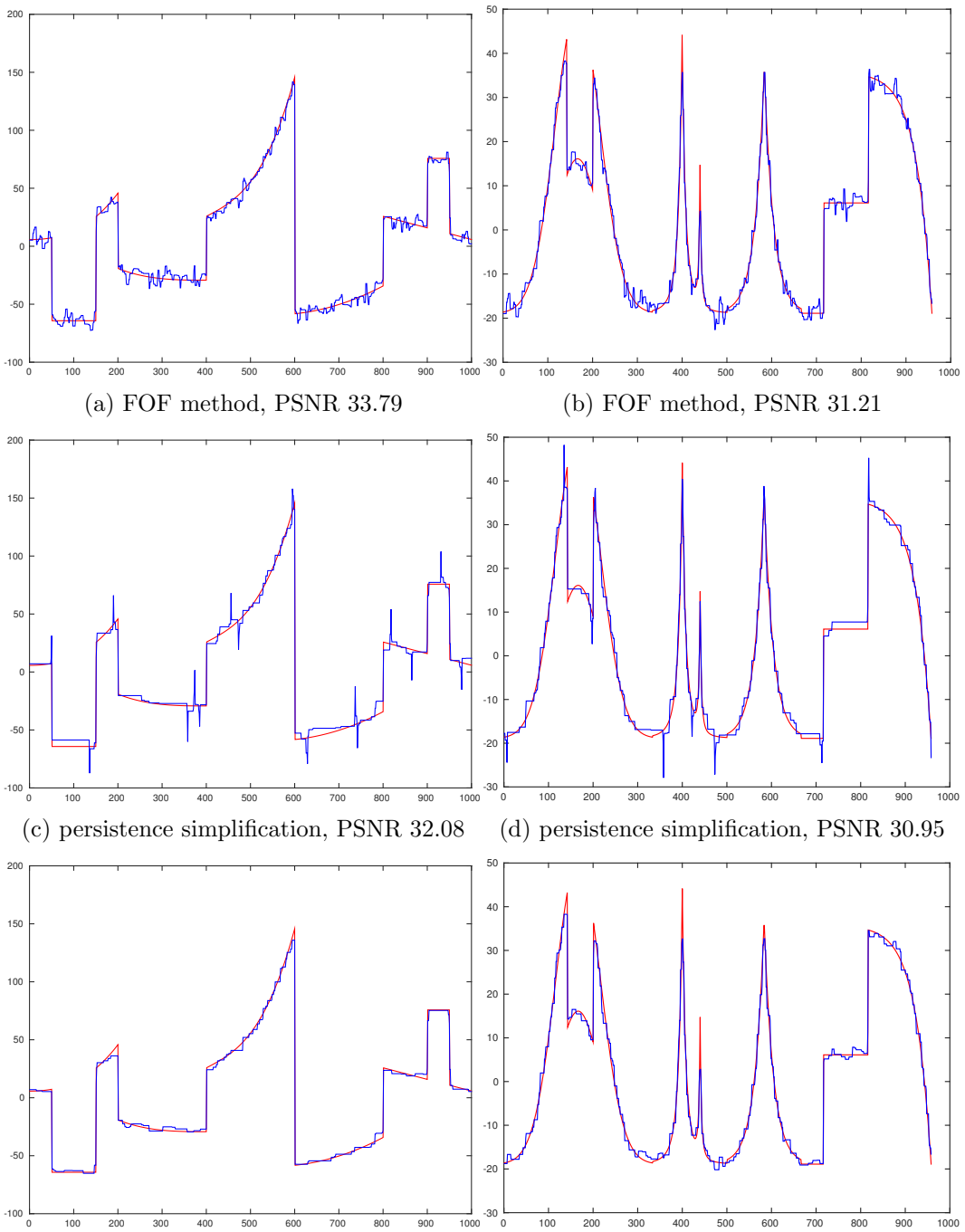
(c) Piece-Polynomial: Gaussian noise, PSNR 26.18

(d) Piece-Regular: Gaussian noise, PSNR 24.77

(e) Piece-Polynomial: uniform noise, PSNR 27.44

(f) Piece-Regular: uniform noise, PSNR 26.05

Fig. 5.1: First row: two test signals; Second row: Gaussian noise; Third row: Uniform noise. Red lines represent original signal, the blue ones represent noisy signals.

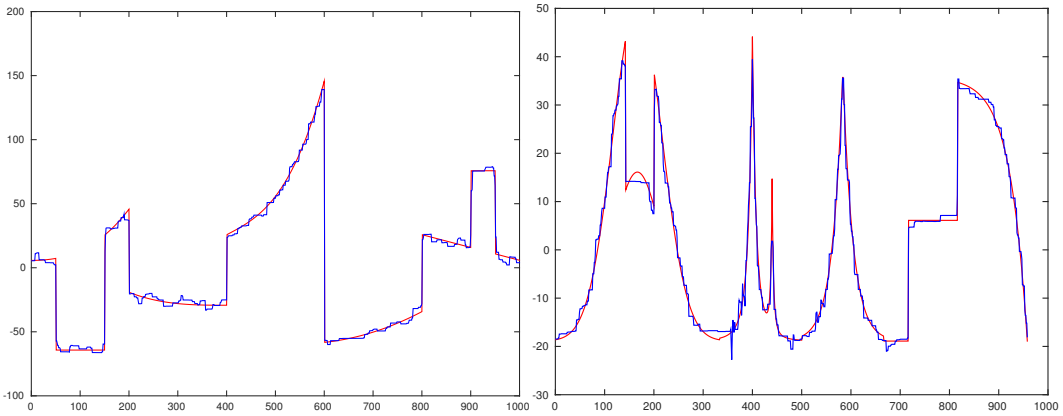


(a) FOF method, PSNR 33.79 (b) FOF method, PSNR 31.21

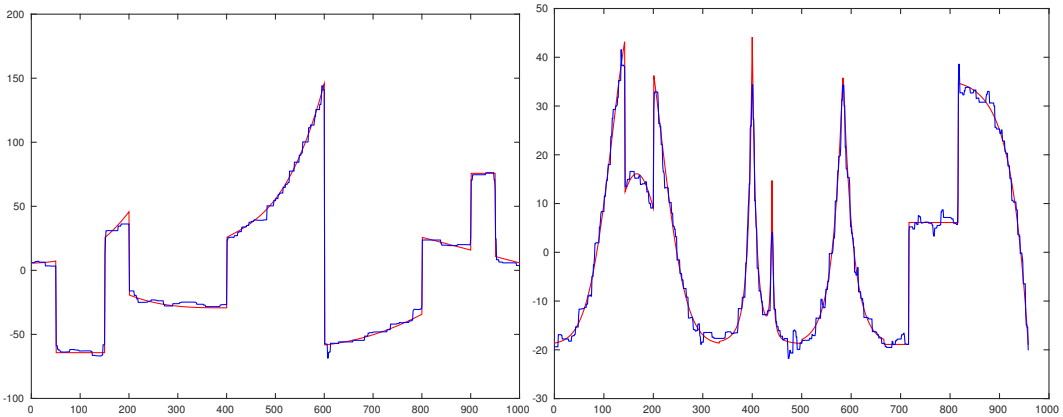
(c) persistence simplification, PSNR 32.08 (d) persistence simplification, PSNR 30.95

(e) Primal-dual denoising, PSNR 38.41 (f) Primal-dual denoising, PSNR 32.56

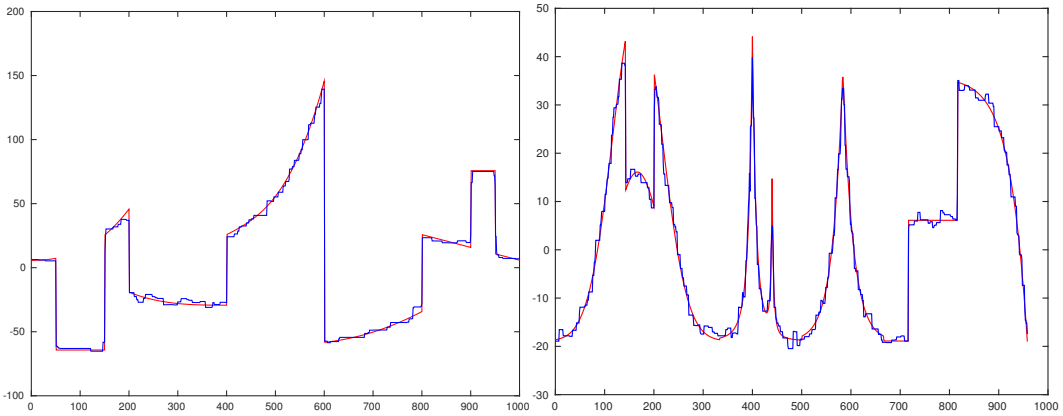
Fig. 5.2: Denoising results for signals with Gaussian noise.
First row: denoising results of FOF method,
Second row: denoising results of persistence simplification,
Third row: denoising results of primal-dual method.
Red lines represent the original signals, blue lines are the denoised results.



(a) WS1 , PSNR 37.11 (b) WS1 , PSNR 32.56



(c) WS2, PSNR 37.90 (d) WS2, PSNR 32.22



(e) Weighted PD with WS1, PSNR 38.43 (f) Weighted PD with WS1, PSNR 32.65

Fig. 5.3: Denoising results for signals with Gaussian noise.
First row: Persistence denoising with WS1,
Second row: Persistence denoising with WS2,
Third row: Weighted PD with WS1.
Red lines represent the original signals, blue lines are the denoised results.

5.3.4 Denoising results for uniform noise contaminated signals

For signals contaminated by uniform noise, Figures 5.5(a) and (b) show the denoising results obtained by the FOF method with 70000 iterations using $\lambda = 0.05$ for the first and $\lambda = 0.15$ for the second test signal. Figures 5.5(c) and (d) are obtained by using the persistence simplification with thresholds 11 and 10, respectively.

The results obtained by the primal-dual method are shown in Figures 5.2(e) and (f) with $\lambda = 0.027$ and $\lambda = 0.14$, respectively.

The results of our method with WS1 using Algorithm 5.2 are shown in Figures 5.6(a) and (b). Figure 5.6 (a) is obtained with $\lambda = 0.015$, $\eta = 0.0009$ and using 100 outer iterations and 700 inner iterations. Figure 5.6(b) is found with $\lambda = 0.019$, $\eta = 0.0004$ with 100 outer and 700 inner iterations. The parameters and result for persistence denoising with WS2 are shown in Figure 5.6 (c)(d), Table 5.4, and 5.5.

For a better comparison, Figure 5.7 shows the results of the six methods for local regions, particularly for flat regions (see (a) and (c)) and for peak denoising (see (b) and (d)).

For quicker accessibility, the parameters and results are summarized in Tables 5.4 and 5.5.

Method	iteration	λ	η	threshold	PSNR
Noisy signal					27.44
FOF	70000	0.05			35.04
Persistence simplification				11	29.25
Primal-dual method	70000	0.027			40.01
Persistence denoising with WS1	70000	0.015	0.0009		36.41
Persistence denoising with WS2	70000	0.09	0.001		37.80

Table 5.4: Denoising parameters and denoising results for the first signal with uniform noise.

Method	iteration	λ	η	threshold	PSNR
Noisy signal					26.05
FOF	70000	0.075			31.87
Persistence simplification				10	33.56
Primal-dual method	70000	0.14			33.43
Persistence denoising with WS1	70000	0.019	0.0004		30.02
Persistence denoising with WS2	70000	0.3	0.0001		33.39

Table 5.5: Denoising parameters and denoising results for the second signal with uniform noise.

Notice that η in WS2 is much smaller than that in WS1 in Table 5.5, it is reasonable since the hierarchical factor plays a more important role in WS2 while persistence factor is still non-neglectable. But this phenomenon is not always true, i.e., the reverse phenomenon

can happen in some other cases. We believe that the massive change of η and λ for different noisy signals reflects the ratio of persistence of noise pairs and the persistence of the pairs which represent the main structure of signal.

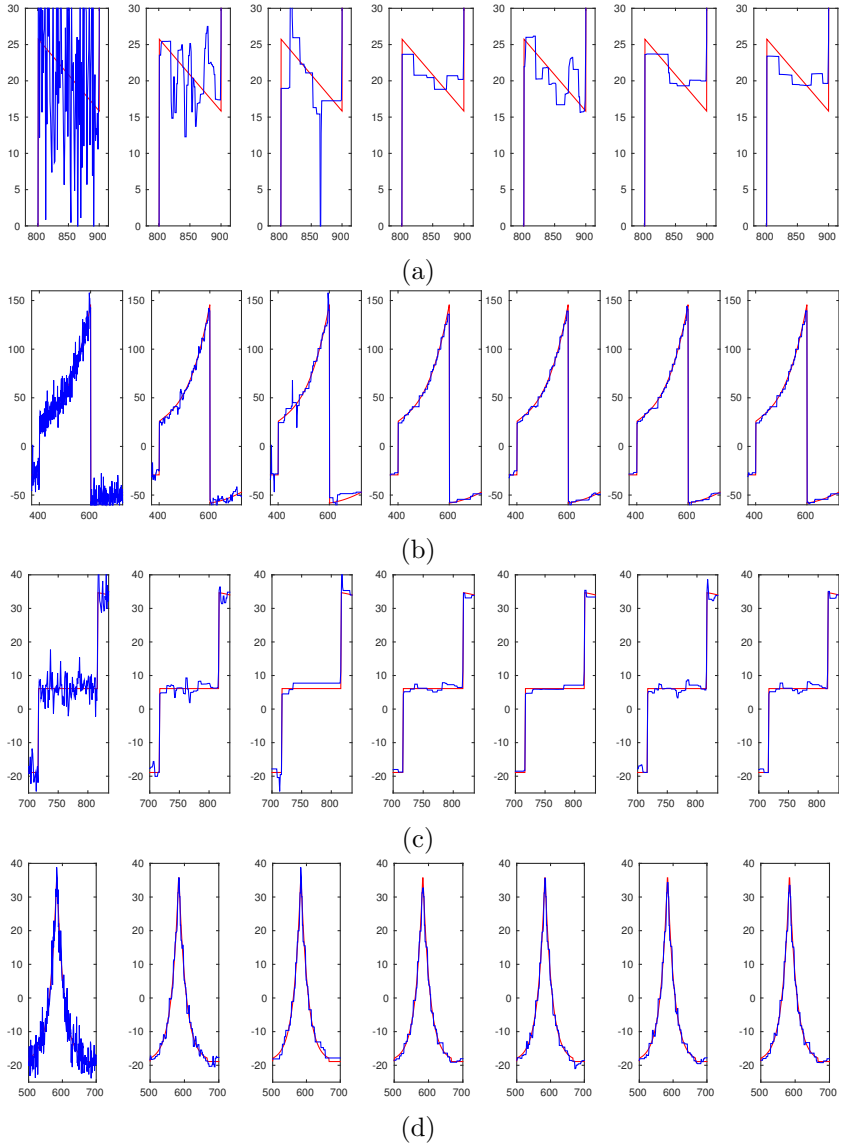
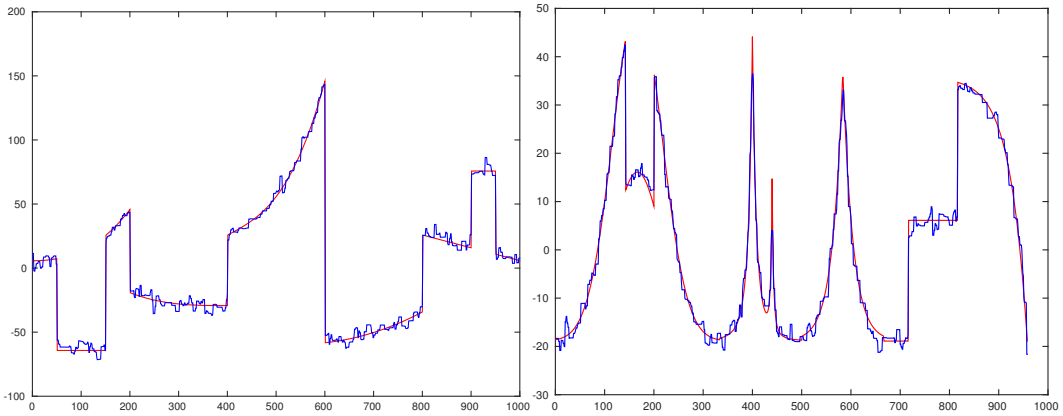
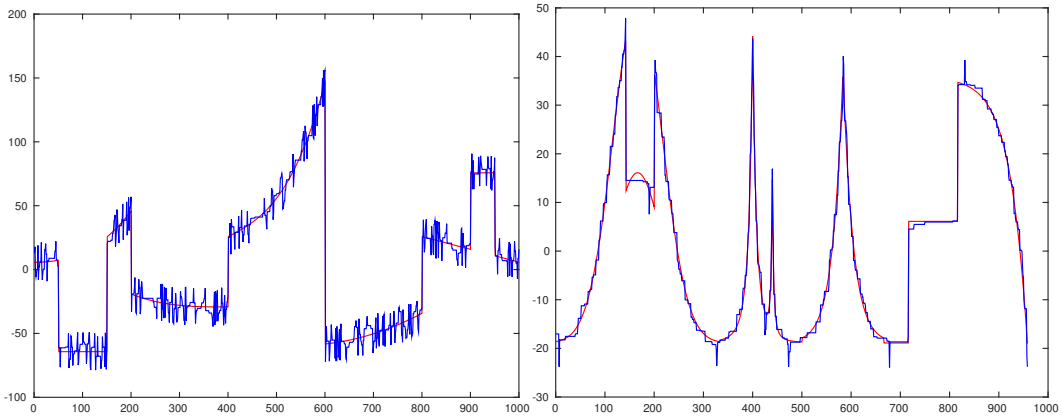


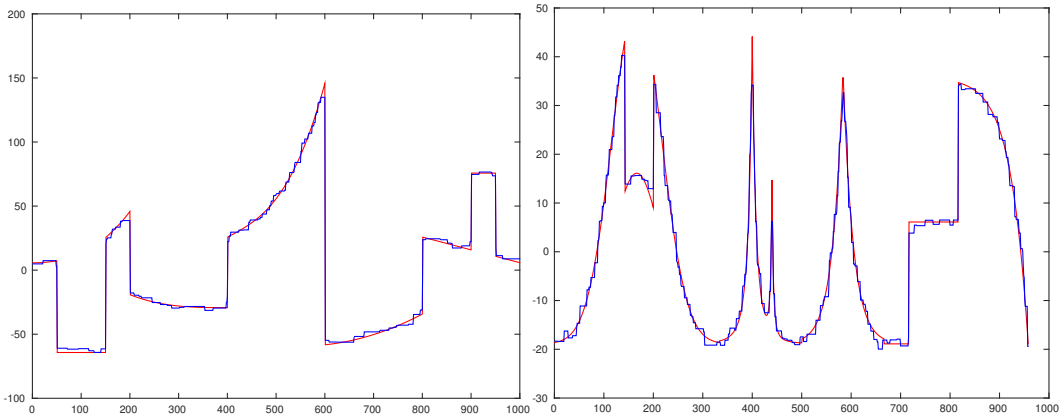
Fig. 5.4: Denoising results for signals with Gaussian noise.
Comparison of the denoising performance of the six methods in flat regions and peak regions; (a) and (b) are subregions of the first test signal, (c) and (d) are subregions of the second test signal; for all subfigures the denoising performance (left to right) is given for noisy signal, FOF denoising, persistence simplification [5], primal-dual, our persistence denosing with WS1 and WS2, and weighted primal-dual with WS1.
Red lines represent the original signal, blue lines are the denoised results.



(a) FOF method, PSNR 35.04 (b) FOF method, PSNR 31.87

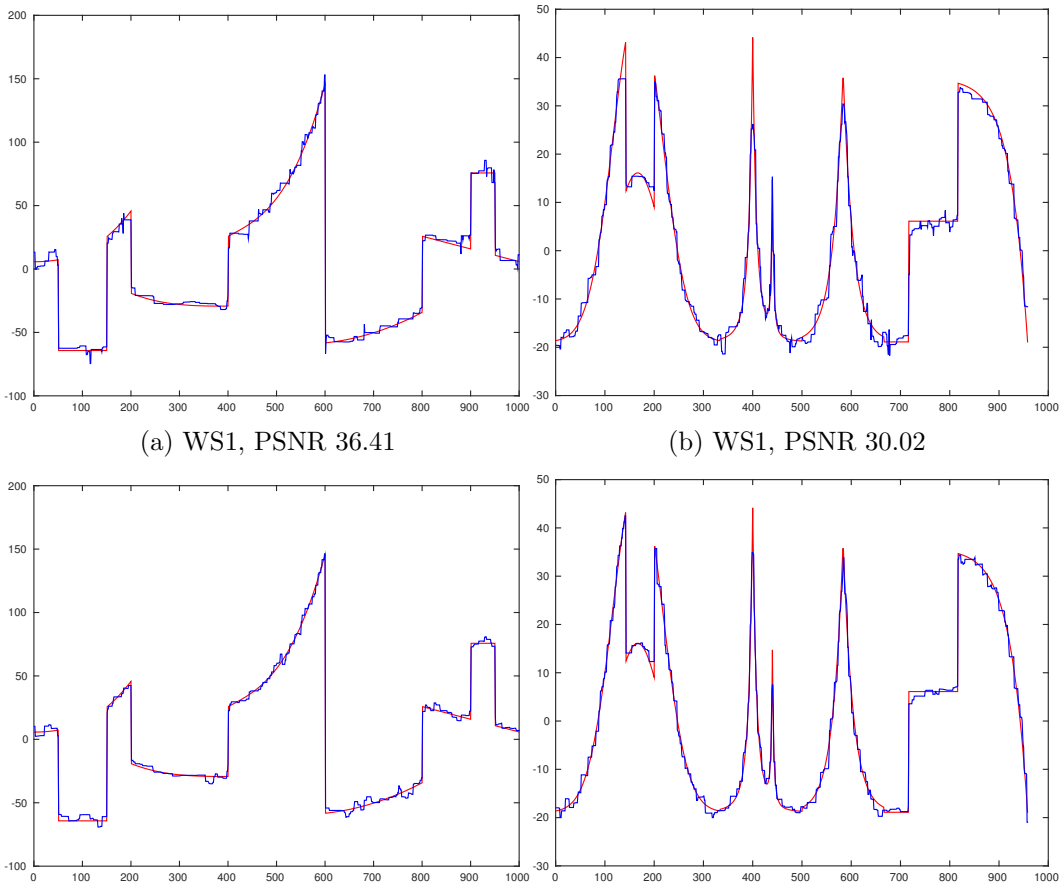


(c) persistence simplification, PSNR 29.25 (d) persistence simplification, PSNR 33.56



(e) Primal-dual denoising, PSNR 40.01 (f) Primal-dual denoising, PSNR 33.43

Fig. 5.5: Denoising results for signals with uniform noise.
First row: denoising results of FOF method,
Second row: denoising results of persistence simplification,
Third row: denoising results of primal-dual method.
Red lines represent the original signals, blue lines are the denoised results.



(a) WS1, PSNR 36.41

(b) WS1, PSNR 30.02

(c) WS2, PSNR 37.80

(d) WS2, PSNR 33.39

Fig. 5.6: Denoising results for signals with uniform noise.
First row: persistence denoising with WS1,
Second row: persistence denoising with WS2.
Red lines represent the original signals, blue lines are the denoised results.

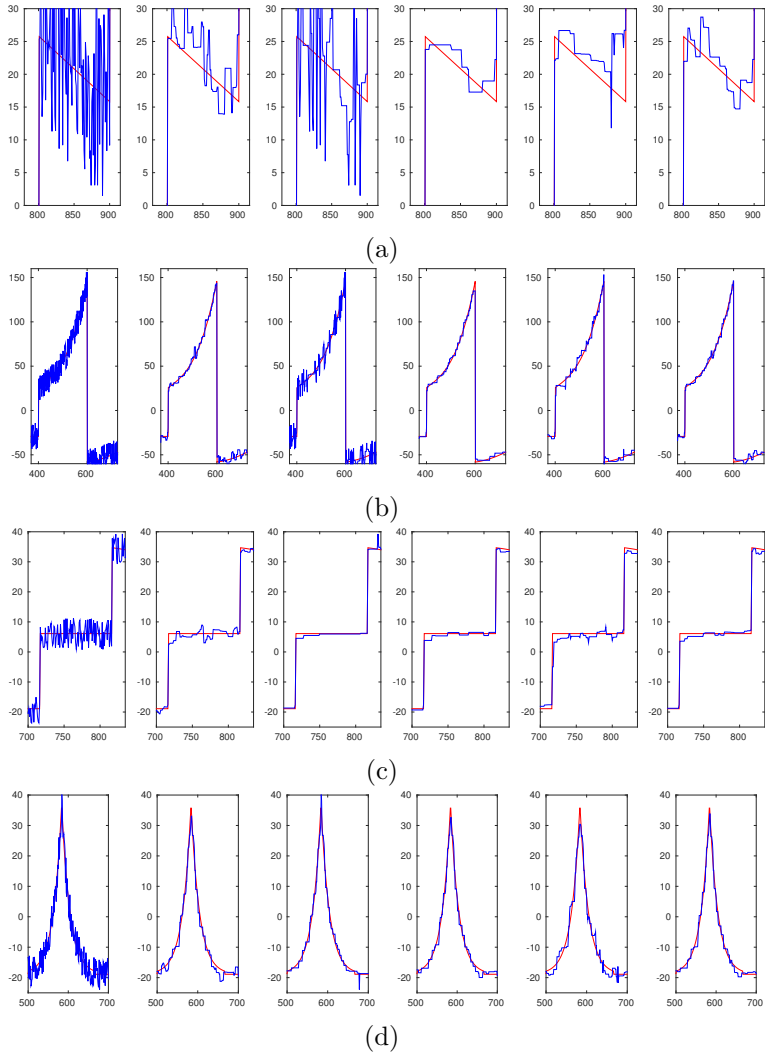


Fig. 5.7: Denoising results for signals with uniform noise.
Comparison of the denoising performance of the six methods in flat regions and peak regions; (a) and (b) are subregions of the first test signal, (c) and (d) are subregions of the second test signal; for all subfigures the denoising performance (left to right) is given for noisy signal, FOF denoising, persistence simplification [5], primal-dual, and our persistence denosing with WS1 and WS2,
red lines represent the original signal, blue lines are the denoised results.

6 Application of persistence distance to image denoising

In this chapter, we want to apply the idea of one-dimensional persistence denoising to image denoising in two ways. One is treating the rows and columns of the image as digital signals and denoising them with our one-dimensional persistence denoising scheme that has been derived in Chapter 5. Another way is utilizing a two-dimensional weighted ROF model where the weights are given using one-dimensional persistences pairs.

Let us first fix the following notations for a discrete image. A matrix $\mathbf{I} \in \mathbb{R}^{(P+1)\times(Q+1)}$ is called a discrete image, where the gray values at the index (i, j) with $i \in \{0, \cdots, P\}$ and $j \in \{0, \cdots, Q\}$, are described by real values in a certain range, e.g. in $[0, 1]$. Often, the range in quantized to 255 different gray values. We denote the i-th row of the matrix $\mathbf{I}$ by $\mathbf{I}_r(i)$ and the j-th column by $\mathbf{I}_c(j)$. Furthermore, the k-th element of $\mathbf{I}_r(i)$ and $\mathbf{I}_c(j)$ are denoted by $\mathbf{I}_r(i)(k) = \mathbf{I}(i, k)$ and $\mathbf{I}_c(j)(k) = \mathbf{I}(k, j)$, respectively. The index set $\{0, \cdots, P\} \times \{0, \cdots, Q\}$ of the image is denoted by Ω.

Remark: For abbreviation, we sometimes denote the double index (i, j) by a greek letter, such as α, β and γ, according to the context.

We assume that the given image $\tilde{\mathbf{I}}$ is contaminated with noise, and we can write

$$\tilde{\mathbf{I}}(\alpha) = \mathbf{I}(\alpha) + n(\alpha), \quad \alpha \in \Omega, \tag{6.1}$$

where the noise n possesses mean 0 and (unknown) variance σ^2, and where $\mathbf{I}$ is the original image. Our goal is to recover the clean image $\mathbf{I}$ from the contaminated version $\tilde{\mathbf{I}}$.

We show in the following sections that this goal can be achieved e.g. by a separable scheme which is based on one-dimensional persistence denoising or by a two-dimensional persistence-weighted ROF model.

6.1 Separable scheme for image denoising

Notice that the rows and columns of the given image $\tilde{\mathbf{I}}$ are in fact digital signals, and we can directly apply our one-dimensional persistence denoising approach to those digital signals. For that purpose, we apply Algorithm 5.2 separately first to all rows then to all columns of $\tilde{\mathbf{I}}$ to recover the clean image $\mathbf{I}$.

To reduce the large effort caused by persistence weights updating, we do not update the persistence weights after every iteration. Instead, we use again an outer-inner iteration structure. More explicitly, we perform as follows. Every outer iteration step contains two stages. Assume that we have obtained the image $\mathbf{I}^{(k)}$ after the k-th outer iteration, where we take the initialization $\mathbf{I}^{(0)} = \tilde{\mathbf{I}}$.

At the first stage, we apply Algorithm 5.2 to all rows $\mathbf{I}_r^{(k)}(i)$, $i = 0, \cdots, P$. That means, we take each row separately. We first compute the persistence weights $w_j(\mathbf{I}_r^{(k)}(i))$ for the i-th row, $i = 0, \ldots, P$, and then apply a fixed number of iteration steps to approximate a signal that minimizes the weighted ROF-functional in (5.4) with these persistence weights. The new row signals that we obtain in this way are taken together to form an "intermediate" image being denoted by $\hat{\mathbf{I}}^{(k+1)}$.

At the second stage, we proceed now with the columns, i.e., we apply Algorithm 5.2 to all columns $\hat{\mathbf{I}}_c^{(k+1)}(j)$, $j = 0, \cdots, Q$ of the previously obtained image $\hat{\mathbf{I}}^{(k+1)}$. Now, we first compute the persistence weights $w_j(\hat{\mathbf{I}}_c^{(k+1)}(j))$ for the j-th row and then again apply a fixed number of iteration steps to approximate a signal that minimizes the weighted ROF-functional in (5.4) with these persistence weights. The new column signals obtained in this way are taken together to form $\mathbf{I}^{(k+1)}$.

For the inner iterations to solve the one-dimensional weighted ROF model, we use here the filter described in Section 5.2 based on a simple iteration scheme. Alternatively, one could use a generalization of the approach of Chan et al. [13] as given in Section 3.2, to the weighted case, or another iterative method as e.g. the primal-dual method by Chambolle and Pock [12].

According to the non-linearity of the problem, it is difficult to derive a general proof of convergence for this algorithm. Taking just the inner iterations, convergence can be shown for fixed weights under certain restrictions. There are several papers using weighted ROF methods with locally adapted weights, some of them with convergence proofs, see e.g. [21, 26, 38, 39].

In our experiments, we apply again the weight strategy 1, i.e., for a given signal $\mathbf{u} = (u(\ell))_{\ell=0}^{N}$ we take the weights

$$\alpha_j(u) = \frac{1}{1 + \eta |u(\tilde{j}) - u(j)|}$$

corresponding to the persistence pair $(\tilde{j}, j)$ in (5.1), and compute the corresponding weights of the weighted ROF functional according to Theorem 5.1.

In particular, we have to fix the regularization parameter λ that determines the influence of the approximation term and the smoothing term in the functional J_w in (5.1) resp. (5.4), the parameter η in the definition of the weights $\alpha_j(u)$.

The following algorithm shows the approach in details.

Algorithm 6.1:

Input: noisy vector $\tilde{\mathbf{I}}$, parameters λ and η.

1) Initialize $\mathbf{I}^0 = \tilde{\mathbf{I}}$.
2) **For** $k = 1, \ldots, n_{outer}$ **do**
  For $i = 0, \ldots, P$ **do**
    Compute the persistence weights $w_j(I_r^k(i))$ in (5.5) and (5.3) for $j = 0, \ldots, Q-1$.
    Initialize $\mathbf{I}_r^{k,0}(i) := \mathbf{I}_r^k(i)$, i.e., $\mathbf{I}_r^{k,0}(i)(j) := \mathbf{I}_r^k(i)(j)$ for $j = 0, \ldots, Q$.
    For $\ell = 1, \ldots, n_{inner}$ **do**
      Compute for $j = 0, \ldots, Q-1$
      $\mathbf{I}_r^{k,\ell+1}(i)(j) = \frac{\lambda \mathbf{I}_r^{k,\ell}(i)(j) + g_{1,j}(\mathbf{I}_r^{k,\ell}(i))\, \mathbf{I}_r^{k,\ell}(i)(j-1) + g_{2,j}(\mathbf{I}_r^{k,\ell}(i))\, \mathbf{I}_r^{k,\ell}(i)(j+1)}{\lambda + g_{1,j}(\mathbf{I}_r^{k,\ell}(i)) + g_{2,j}(\mathbf{I}_r^{k,\ell}(i))}$,
      where $g_{1,j}(\mathbf{I}_r^{k,\ell}(i)) := w_{j-1}(\mathbf{I}_r^{k,\ell}(i))/|\mathbf{I}_r^{k,\ell}(i)(j) - \mathbf{I}_r^{k,\ell}(i)(j-1)|$
      for $|\mathbf{I}_r^{k,\ell}(i)(j) - \mathbf{I}_r^{k,\ell}(i)(j-1)| \neq 0$ (and $g_{1,j}(\mathbf{I}_r^{k,\ell}(i)) := 0$ else)
      $g_{2,j}(\mathbf{I}_r^{k,\ell}(i)) := w_j(\mathbf{I}_r^{k,\ell}(i))/|\mathbf{I}_r^{k,\ell}(i)(j) - \mathbf{I}_r^{k,\ell}(i)(j+1)|$
      for $|\mathbf{I}_r^{k,\ell}(i)(j) - \mathbf{I}_r^{k,\ell}(i)(j+1)| \neq 0$ (and $g_{2,j}(\mathbf{I}_r^{k,\ell}(i)) := 0$ else).
    end
  end
  Initialize matrix $\mathbf{I}^k := ((\mathbf{I}_r^{k,n_{inner}+1}(0))^T, \cdots, (\mathbf{I}_r^{k,n_{inner}+1}(P))^T)^T$,
  For $j = 0, \ldots, Q$ **do**
    Compute the persistence weights $w_i(I_c^k(j))$ in (5.5) and (5.3) for $i = 0, \ldots, P-1$.
    Initialize $\mathbf{I}_c^{k,0}(j) := \mathbf{I}_c^{k,0}(j)$ for $j = 0, \ldots, Q$.
    For $\ell = 1, \ldots, n_{inner}$ **do**
      Compute for $i = 0, \ldots, P$
      $\mathbf{I}_c^{k,\ell+1}(j)(i) = \frac{\lambda \mathbf{I}_c^{k,\ell}(j)(i) + g_{1,i}(\mathbf{I}_c^{k,\ell}(j))\, \mathbf{I}_c^{k,\ell}(j)(i-1) + g_{2,i}(\mathbf{I}_c^{k,\ell}(j))\, \mathbf{I}_c^{k,\ell}(j)(i+1)}{\lambda + g_{1,i}(\mathbf{I}_c^{k,\ell}(j)) + g_{2,i}(\mathbf{I}_c^{k,\ell}(j))}$,
      where as defined $g_{1,i}(\mathbf{I}_c^{k,\ell}(j)) := w_{i-1}(\mathbf{I}_c^{k,\ell}(j))/|\mathbf{I}_c^{k,\ell}(j)(i) - \mathbf{I}_c^{k,\ell}(j)(i-1)|$
      for $|\mathbf{I}_c^{k,\ell}(j)(i) - \mathbf{I}_c^{k,\ell}(j)(i-1)| \neq 0$ (and $g_{1,i}(\mathbf{I}_c^{k,\ell}(j)) := 0$ else)
      $g_{2,i}(\mathbf{I}_c^{k,\ell}(j)) := w_i(\mathbf{I}_c^{k,\ell}(j))/|\mathbf{I}_c^{k,\ell}(j)(i) - \mathbf{I}_c^{k,\ell}(j)(i+1)|$
      for $|\mathbf{I}_c^{k,\ell}(j)(i) - \mathbf{I}_c^{k,\ell}(j)(i+1)| \neq 0$ (and $g_{2,i}(\mathbf{I}_c^{k,\ell}(j)) := 0$ else).
    end
  end
  Put $\mathbf{I}^{k+1} := (\mathbf{I}_c^{k,n_{inner}+1}(0), \cdots, \mathbf{I}_c^{k,n_{inner}+1}(Q))$.
end

Output: $\mathbf{I} = \mathbf{I}^{n_{outer}+1}$ approximates the minimizer of $\min_u J_w(u)$.

6.2 2D persistence-weighted ROF model

We want to propose a second scheme that uses a two-dimensional approximation of the discrete total variation but applies again the persistence weights based on the persistence pairs and their structure along rows and columns of the image.

For a fixed (inner) index $\alpha := (i, j)$ in the index set Ω of the image $\mathbf{I}$, we consider its four neighbors $(i-1, j)$, $(i+1, j)$,$(i, j-1)$ and $(i, j+1)$. We call this set of indices the **neighborhood** of $\alpha = (i, j)$, denoted by $NB(\alpha) = NB(i, j)$, i.e.,

$$NB(\alpha) = NB(i, j) = \{\beta | \beta \sim (\alpha)\} = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\}.$$

For boundary indices $\alpha = (i, j)$, where $i \in \{0, P\}$ or $j \in \{0, Q\}$ we will have only three (or at the corners even only two) neighbor indices and just say $NB(i, j) = \{\beta | \beta \sim (i, j)\} = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\} \cap \Omega$. Now, we consider the intervals generated by (i, j) and its neighbor indices. It is easy to see that the indices $(i, j-1)$, $(i, j+1)$ are in the same row $\mathbf{I}_r(i)$ with (i, j) and the indices $(i-1, j)$, $(i+1, j)$ are in the same column $\mathbf{I}_c(j)$ with (i, j). The two indices $(i, j-1)$ and $(i, j+1)$ in the i-th row $\mathbf{I}_r(i)$ of the image $\mathbf{I}$ generate two intervals with (i, j), respectively, namely $[(i, j-1), (i, j)]$ and $[(i, j), (i, j+1)]$. Applying our weighted ROF approach based on persistence pairs to the i-th row $\mathbf{I}_r(i)$, we assign the weights $w_{j-1}(\mathbf{I}_r(i))$ and $w_j(\mathbf{I}_r(i))$ derived in Theorem 5.1 (see (5.5)) to these two intervals. Similarly, we compute the weights for the two intervals generated in the column $\mathbf{I}_c(j)$. We summarize this idea in the following.

Definition 6.2:
For any index $\alpha := (i, j)$ in the index set Ω of a given image $\mathbf{I}$, α is contained in at most four **intervals** being defined as follows. The **left interval** and **right interval** are the horizontal intervals $[\mathbf{I}_r(i)(j-1), \mathbf{I}_r(i)(j)]$ and $[\mathbf{I}_r(i)(j), \mathbf{I}_r(i)(j+1)]$, respectively, in row $\mathbf{I}_r(i)$. The **up interval** and **down interval** are the vertical intervals $[\mathbf{I}_c(j)(i-1), \mathbf{I}_c(j)(i)]$ and $[\mathbf{I}_c(j)(i), \mathbf{I}_c(j)(i+1)]$, respectively, in row $\mathbf{I}_c(j)$. If $NB(\alpha)$ contains less entries, we have only three (or two) intervals corresponding to α.

The persistence weights for the above intervals can be given as follows.

Definition 6.3:
For each index $\alpha := (i, j) \in \Omega$ of a given image $\mathbf{I}$, we assign the following persistence weights to the interval $[\alpha, \beta]$, where $\beta \in NB(\alpha)$,

$$z_{\alpha\beta}(\mathbf{I}) := \begin{cases} w_{j-1}(\mathbf{I}_r(i)) & \text{for } [\alpha, \beta] = [\mathbf{I}_r(i)(j-1), \mathbf{I}_r(i)(j)] \text{ and } (i, j-1) \in \Omega, \\ w_j(\mathbf{I}_r(i)) & \text{for } [\alpha, \beta] = [\mathbf{I}_r(i)(j), \mathbf{I}_r(i)(j+1)] \text{ and } (i, j+1) \in \Omega, \\ w_{i-1}(\mathbf{I}_c(j)) & \text{for } [\alpha, \beta] = [\mathbf{I}_c(j)(i-1), \mathbf{I}_c(j)(i)] \text{ and } (i-1, j) \in \Omega, \\ w_i(\mathbf{I}_c(j)) & \text{for } [\alpha, \beta] = [\mathbf{I}_c(j)(i), \mathbf{I}_c(j)(i+1)] \text{ and } (i+1, j) \in \Omega, \end{cases}$$

where the weights $w_j(\mathbf{I}_r(i))$, $j = 0, \ldots, Q-1$, for the rows and columns of $\mathbf{I}$ (taken as one-dimensional signals) are computed as given in Theorem 5.1.

Remark: We need not to distinguish the ordering of subindices, since $z_{\alpha\beta} = z_{\beta\alpha}$ always holds true. This can be checked in the following way. For the horizontal interval case, it is easy to see that the interval $[\alpha, \beta]$ formed by α and its east neighbor β coincides with the interval $[\alpha, \beta]$ formed by β and its west neighbor α. The same holds true for the vertical interval case.

Considering the local variation of $\mathbf{I}$ at node $\alpha = (i, j)$ given by (3.7)

$$|\nabla_\alpha \mathbf{I}| = \sqrt{\sum_{\beta\sim\alpha} (\mathbf{I}(\alpha) - \mathbf{I}(\beta))^2},$$

we apply now the persistence weights to obtain the **persistence-weighted local variation** of the image

$$|\tilde{\nabla}_\alpha \mathbf{I}| := \sqrt{\sum_{\beta\sim\alpha} z_{\alpha\beta}(\mathbf{I})(\mathbf{I}(\alpha) - \mathbf{I}(\beta))^2},$$

where the additional (positive) weights $z_{\alpha\beta}$ give more structural information about the intervals $[\alpha, \beta]$. The regularized version is

$$|\tilde{\nabla}_\alpha \mathbf{I}|_a = \sqrt{\sum_{\beta\sim\alpha} z_{\alpha\beta}(\mathbf{I})(\mathbf{I}(\alpha) - \mathbf{I}(\beta))^2 + a^2}.$$

Comparing this new definition with the local variation in (3.7) and the corresponding discrete edge derivative in (3.9), we now determine the **weighted edge derivative** determined by

$$\left.\frac{\partial_w \mathbf{I}}{\partial_w e}\right|_\alpha := z_{\alpha\beta}(\mathbf{I})(\mathbf{I}(\beta) - \mathbf{I}(\alpha)) \tag{6.2}$$

for the edge interval $[\alpha, \beta]$.

Formally, we can now define the **persistence-weighted digital TV filter** as

$$\tilde{\mathscr{F}}_\alpha^{\lambda,a}(\mathbf{I}^k) = \sum_{\beta\sim\alpha} t_{\alpha\beta}(\mathbf{I})\mathbf{I}(\beta) + t_{\alpha\alpha}(\mathbf{I})\mathbf{I}^0(\alpha), \;\; \alpha \in \Omega,$$

where parameters are given by

$$t_{\alpha\beta}(\mathbf{I}) = \frac{p_{\alpha\beta}(\mathbf{I})}{\lambda + \sum_{\gamma\sim\alpha} p_{\alpha\gamma}(\mathbf{I})}, \;\; \text{for } \alpha,\beta \in \Omega$$

$$t_{\alpha\alpha}(\mathbf{I}) = \frac{\lambda}{\lambda + \sum_{\gamma\sim\alpha} p_{\alpha\gamma}(\mathbf{I})},$$

$$p_{\alpha\beta}(\mathbf{I}) = z_{\alpha\beta}(\mathbf{I})\Big(\frac{1}{|\tilde{\nabla}_\alpha \mathbf{I}|} + \frac{1}{|\tilde{\nabla}_\beta \mathbf{I}|}\Big), \;\; \text{for } \alpha,\beta \in \Omega$$

It is easy to check that for any index α

$$t_{\alpha\alpha}(\mathbf{I}) + \sum_{\gamma\sim\alpha} t_{\alpha\gamma}(\mathbf{I}) = 1.$$

With the new TV filter, Algorithm 3.1 with additonal outer iteration can thus be modified as follows.

Algorithm 6.4:

Input: noisy signal $\tilde{\mathbf{I}}$, parameters λ, $a = 0.0001$.

1) Initialize $\mathbf{I}^{0,0} = \tilde{\mathbf{I}}$.
2) **For** $j = 0, \ldots, N_{\text{outer}}$ **do**
   Update $t_{\alpha\beta}(\mathbf{I}^{j,0})$
   For $k = 0, \ldots, N_{\text{inner}}$ **do**
   For $\alpha \in \Omega$ **do**
   Compute $\mathbf{I}^{j,k+1}(\alpha) = \tilde{\mathscr{F}}_{\alpha}^{\lambda,a}(\mathbf{I}^{j,k}) = \sum_{\beta\sim\alpha} t_{\alpha\beta}(\mathbf{I}^{j,0})\mathbf{I}^{j,k}(\beta) + t_{\alpha\alpha}(\mathbf{I}^{j,0})\mathbf{I}^{0,0}(\alpha)$
   end
   end
   Let $\mathbf{I}^{j+1,0} = \mathbf{I}^{j,N_{inner}+1}$
   end

Output: $\mathbf{I} = \mathbf{I}^{N_{iteration}+1}$ approximates the minimizer of $\min_u J(u)$ in (6.3).

For the inner iteration of above algorithm for the weighted ROF-model we can show similar results as for the COS-filter in Subsection 3.2.1, see also [13].

Theorem 6.5:
If the TV filtering process in Algorithm 6.4 converges, then the limit signal $\mathbf{I}$ is the unique minimizer of the TV energy functional

$$J_{\alpha}(\mathbf{I}) = \sum_{\alpha} |\tilde{\nabla}_{\alpha}\mathbf{I}|_a + \frac{\lambda}{2}\sum_{\Omega}(\mathbf{I}(x) - \tilde{\mathbf{I}}(x))^2. \tag{6.3}$$

Theorem 6.6:
If the filtering process in Algorithm 6.4 converges to some image $\mathbf{I}$, then $\mathbf{I}$ satisfies

$$\sum_{e\vdash\alpha} \frac{\partial}{\partial e}\frac{-1}{|\tilde{\nabla}\mathbf{I}|_a}\frac{\partial_w \mathbf{I}}{\partial_w e}\bigg|_{\alpha} + \lambda(\tilde{\mathbf{I}}_{\alpha} - \mathbf{I}_{\alpha}) = 0, \quad \alpha \in \Omega. \tag{6.4}$$

Proof: The limit satisfies

$$\mathbf{I}_{\alpha} = \mathscr{F}_{\alpha}(\mathbf{I}), \quad \alpha \in \Omega.$$

It represented as

$$\Big(\lambda + \sum_{\gamma\sim\alpha} p_{\alpha\gamma}\Big)\mathbf{I}(\alpha) = \sum_{\beta\sim\alpha} p_{\alpha\beta}\mathbf{I}(\beta) + \lambda\tilde{\mathbf{I}}(\alpha),$$

which equals

$$\sum_{\beta\sim\alpha} p_{\alpha\beta}(\mathbf{I}(\beta) - \mathbf{I}(\alpha)) + \lambda(\tilde{\mathbf{I}}(\alpha) - \mathbf{I}(\alpha)) = 0.$$

Thus, the proof is done if we can show that

$$\sum_{\beta\sim\alpha} p_{\alpha\beta}(\mathbf{I}(\beta)-\mathbf{I}(\alpha)) = \sum_{e\vdash\alpha} \frac{\partial}{\partial e}\frac{-1}{|\tilde{\nabla}\mathbf{I}|_a}\frac{\partial_w \mathbf{I}}{\partial_w e}\Big|_\alpha$$

holds true. We compute directly the right-hand side of the above equation according to the definition of the edge derivative (3.9) and the weighted edge derivative (6.2) and find for the edge $\alpha\sim\beta$

$$\begin{aligned}
&\frac{\partial}{\partial e}\frac{-1}{|\tilde{\nabla}\mathbf{I}|_a}\frac{\partial_w \mathbf{I}}{\partial_w e}\Big|_\alpha = \Big(\frac{-1}{|\tilde{\nabla}\mathbf{I}|_a}\frac{\partial_w \mathbf{I}}{\partial_w e}\Big)\Big|_\beta - \Big(\frac{-1}{|\tilde{\nabla}\mathbf{I}|_a}\frac{\partial_w \mathbf{I}}{\partial_w e}\Big)\Big|_\alpha \\
= \;& \frac{-1}{|\tilde{\nabla}_\beta \mathbf{I}|_a}\Big(\frac{\partial_w \mathbf{I}}{\partial_w e}\Big)\Big|_\beta + \frac{1}{|\tilde{\nabla}_\alpha \mathbf{I}|_a}\Big(\frac{\partial_w \mathbf{I}}{\partial_w e}\Big)\Big|_\alpha \\
= \;& \frac{-1}{|\tilde{\nabla}_\beta \mathbf{I}|_a} z_{\alpha\beta}(\mathbf{I})(\mathbf{I}(\alpha)-\mathbf{I}(\beta)) + \frac{1}{|\tilde{\nabla}_\alpha \mathbf{I}|_a} z_{\alpha\beta}(\mathbf{I})(\mathbf{I}(\beta)-\mathbf{I}(\alpha)) \\
= \;& z_{\alpha\beta}(\mathbf{I})\Big(\frac{1}{|\tilde{\nabla}_\beta \mathbf{I}|_a}+\frac{1}{|\tilde{\nabla}_\alpha \mathbf{I}|_a}\Big)(\mathbf{I}(\beta)-\mathbf{I}(\alpha)).
\end{aligned}$$

Noticing that the last term is exactly $p_{\alpha\beta}(\mathbf{I}_\beta - \mathbf{I}_\alpha)$, we have $p_{\alpha\beta}(\mathbf{I}) = z_{\alpha\beta}(\mathbf{I})\Big(\frac{1}{|\tilde{\nabla}_\alpha \mathbf{I}|}+\frac{1}{|\tilde{\nabla}_\beta \mathbf{I}|}\Big)$ which finishes the proof. □

We will show the performance of this persistence-weighted ROF-model in experiments with weight strategy 1 in Section 6.4.

6.3 Comparison with other image denoising methods

We want to compare the performance of our weighted ROF model for image denoising with some other models that have been used in the literature. Beside the usual (unweighted) ROF approach being an example for an image denoising method based on variational methods, we consider for comparison one method based on the anisotropic nonlinear diffusion, namely the four-pixel scheme proposed by Welk, Weickert and Steidl, [41], one method based on computational harmonic analysis, namely shearlet shrinkage, and one of the actually best performing image denoising methods, the hybrid method BM3D proposed by Dabov, Foi, Katkovnik & Egiazarian, [17, 18].

6.3.1 Four-Pixel scheme

There exist several approaches for image denoising that are based on partial differential equations, particularly on nonlinear diffusion, see e.g. [34, 40] and references therein. For our comparisons, we consider the four-pixel scheme in [41] that is based on a nonlinear diffusion process where the corresponding diffusivity function is singular. Singular

diffusivities, as e.g. the TV-diffusion $g(|\nabla u|) = |\nabla u|^{-1}$, have the advantage that no further contrast parameter needs to be fixed, but a suitable numerical implementation of the obtained singular differential equation is difficult. Therefore, in [41] a new numerical method is proposed for the 2D-case, using local analytical solutions of the diffusion equation on 2×2-images. These local solutions serve as building blocks for generating a solution for general $2D$-images, where operator splitting methods are employed. The local analytical solutions for the 2×2-images can also be related to a Haar-wavelet shrinkage procedure with shift-invariant Haar wavelets.

In the experiments, we choose the diffusion funtion as $g(|\nabla u|) = |\nabla u|^{-1}$.

6.3.2 Shearlet-Shrinkage

The shearlet shrinkage procedure for image denoising is based on the representation of the image in the redundant shearlet frame. Shearlets can be seen as a generalization of two-dimensional wavelet approaches, where the elements of the shearlet frame are generated by dilation, shearing and translation of one "mother shearlet". Similarly as for curvelet constructions [8], the dilation is taken differently in x- and y-direction using e.g. a dilation matrix of the form

$$A = \begin{pmatrix} 2 & 0 \\ 0 & \sqrt{2} \end{pmatrix}.$$

This construction enables us to recognize directional structures in images using the shearlet representation. Earlier shearlet constructions are based on frame functions being compactly supported on "wedges" in Fourier domain. For a suitable choice of the mother shearlet, the set of all shearlet functions can form even a Parseval frame, see e.g. [30].

Recently, special shearlet frame constructions have been proposed whose frame elements have compact support in spatial domain. These shearlet frames admit a very efficient implementation of the shearlet transform that can be also generalized to 3D, [28,29,31].

Similarly as for wavelet denoising techniques, shearlet shrinkage is based on the restriction of the image representation to only the most important shearlet coefficients, while small shearlet coefficients are taken as "noise". For shrinking the shearlet coefficients we have used the usual soft-threshold function.

The experiment code with shearlet shringkage used in this thesis is ShearLab (Version 1.0), [43].

6.3.3 BM3D

BM3D is one of the state-of-the-art algorithms for image denoising. Originally proposed by Dabov at al., see [17,18], this hybrid method has been continuously developed further since that time, see e.g. [19].

The method unifies different denoising strategies. In a first step, certain reference image blocks are fixed, and all image blocks being similar to the reference block are grouped together to obtain a 3D block. In the second step a so-called "collaborative filtering" is employed. Here, a 3D-transform is applied together with a shrinkage procedure. The 3D-transform can exploit correlations inside of each of the image blocks but also between the image blocks. As a 3D-transform, one can use a wavelet transform e.g. with a Daubechies or biorthogonal filters. Since the reference blocks are usually overlapping, one can get a first improved denoising result by weighted averaging of the obtained denoised blocks. In a last step, an improved denoising result is obtained by a second block grouping and the application of a Wiener filter procedure.

6.4 Numerical results

This section is contributed to show the performance of our proposed two-dimensional persistence-weighted schemes. We employ the cameraman and peppers images of size 256×256 to illustrate the denoising results for different methods. The original images are presented in the first row of Figure 6.1. The images are contaminated with Gaussian noise where we consider the two different noise levels $\sigma = 0.1$ and $\sigma = 0.15$ (see second and third rows of Figure 6.1, respectively) with corresponding PSNR values. Here the PSNR value is determined by

$$\mathrm{PSNR}(\mathbf{I}, \tilde{\mathbf{I}}) = 10 \log_{10} \frac{1}{\frac{1}{(P+1)\cdot(Q+1)} \sum_{\alpha \in \Omega} (\mathbf{I}(\alpha) - \tilde{\mathbf{I}}(\alpha))^2},$$

where $\mathbf{I} \in \mathbb{R}^{(P+1)\times(Q+1)}$ and $\alpha \in \Omega$. Here we have assumed that the gray values of the images are normalized to the range $[0, 1]$. For computing the PSNR value, $\mathbf{I}$ is the original image and $\tilde{\mathbf{I}}$ is either the noise contaminated image or the denoised image.

The denoising results for noisy cameraman image with $\sigma = 0.1$ and the corresponding parameters used for the various methods are shown in Table 6.1 and in Figure 6.2.

For the second noise level $\sigma = 0.15$, the denoising results and parameters are given in Table 6.2 and Figure 6.3.

It is worth pointing out how the persistence weights updating is done. For a fixed number of total iterations, the updating of persistence weights is done at most 100 times, i.e., after each hundredth of the number of iterations. Taking e.g. a total of 5000 inner iterations, the updating of persistence weigths is performed after 50 inner iterations. In

our numerical examples below we have used only 60 iterations, and persistence weights updating actually takes place for every iteration.

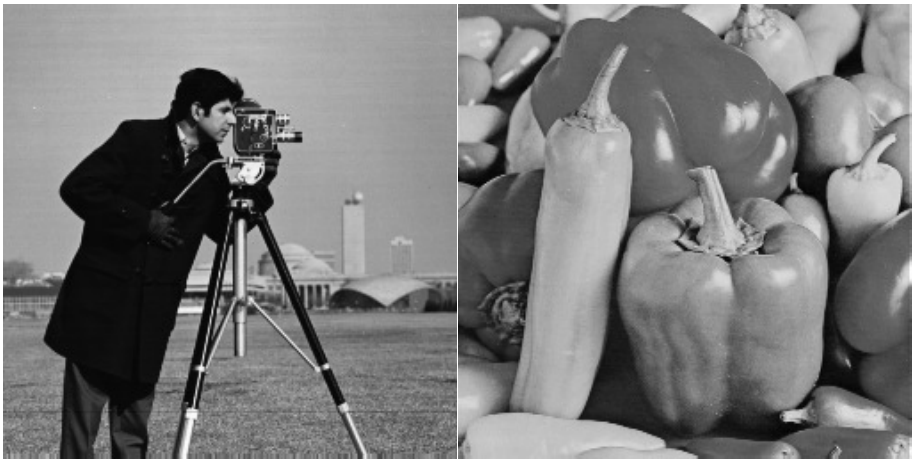
We observe that our non-separated persistence-weighted ROF model with weight strategy 1 (WS1) works slightly better than the separable scheme with the same weight strategy WS1. The achieved PSNR value of or persistence-weighted ROF scheme with WS1 also slightly outperforms the results of the four-pixel scheme and the shearlets denoising result. The higher PSNR achieved by our persistence-weighted ROF scheme benefits from the better contrast at the boundaries which is the expected advantage of taking for significant pairs smaller persistence weights. The obvious drawbacks of the new persistence-weighted ROF scheme and separable scheme are the emerging blocking effects and the more expensive computation of persistence weights. The blocking effects can be reduced either by employing a different smoothing filter for the inner iterations or by applying a smoothing filter in a post-processing step.

The performance of the various methods for denoising the peppers image are similar. The obtained results and parameters for the first noise level $\sigma = 0.1$ are shown in Table 6.3 and in Figure 6.4. For the higher noise level $\sigma = 0.15$, the PSNR values and parameters are summarized in Table 6.4 and in Figure 6.5.

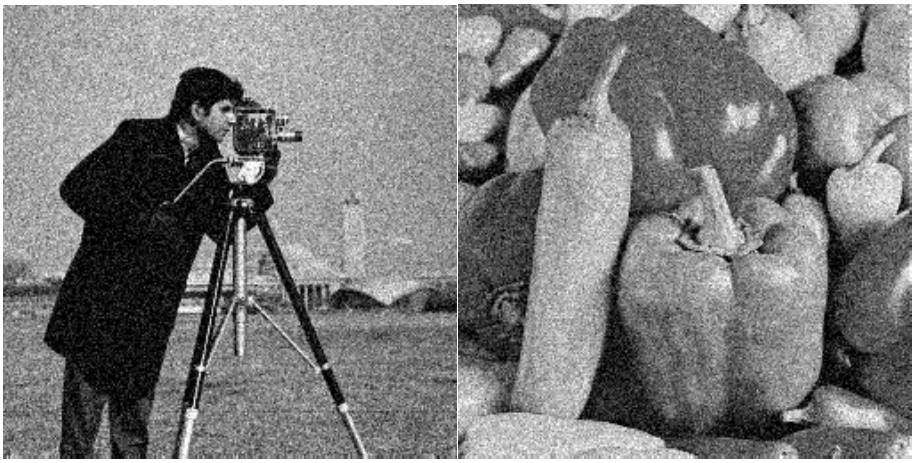
Again, the performance of our persistence-weighted ROF approach is similarly good as that of the four-pixel scheme, and both schemes, the separable scheme with WS1 and the 2D persistence weighted ROF model outperform the shearlet denoising result. For the peppers image, the less good performance of the persistence-weighted ROF scheme in comparison to the four-pixel scheme may be caused by less strong contrast within the image such that keeping the boundaries nicely is less important than approximating the smooth regions.

Method	Parameters	PSNR
Noisy image		19.97
Four-pixel scheme	73 iterations, $\tau = 0.001$	27.64
Shearlets	$\theta = 0.01$, others default	26.07
BM3D	block size 8×8, others default	29.36
Separable scheme WS1	60 iterations, $\lambda = 0.015$, $\beta = 0.006$	27.28
Persistence-weighted ROF WS1	60 iterations, $\lambda = 0.06$, $\beta = 0.0012$	27.72

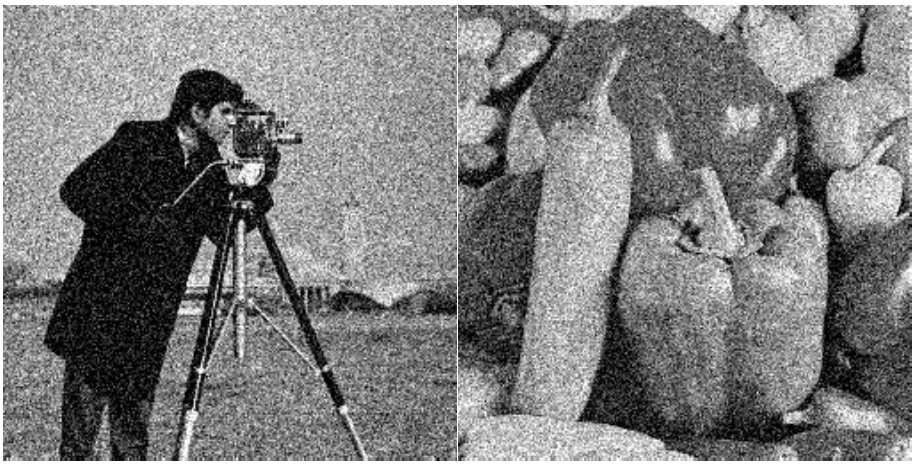
Table 6.1: Denoising parameters and denoising results (cameraman, $\sigma = 0.1$).



(a) Cameraman (b) Peppers



(c) Cameraman, $\sigma = 0.1$, PSNR 19.97 (d) Peppers, $\sigma = 0.1$, PSNR 19.97



(e) Cameraman, $\sigma = 0.15$, PSNR 16.45 (f) Peppers, $\sigma = 0.15$, PSNR 16.45

Fig. 6.1: (Gaussian noise)First row: Original images
Second row: noisy images with $\sigma = 0.1$
Third row: noisy images with $\sigma = 0.15$.

(a) Four-pixel scheme, PSNR 27.64

(b) Shearlets, PSNR 26.07

(c) BM3D, PSNR 29.36

(d) Separable scheme, PSNR 27.28

(e) Persistence-weighted ROF, PSNR 27.72

Fig. 6.2: Denoising results for cameraman image with $\sigma = 0.1$

(a) Four-pixel scheme, PSNR 25.73

(b) Shearlets, PSNR 24.23

(c) BM3D, PSNR 27.50

(d) Separable scheme, PSNR 25.24

(e) Persistence-weighted ROF, PSNR 25.78

Fig. 6.3: Denoising results for cameraman image with $\sigma = 0.15$

(a) Four-pixel scheme, PSNR 28.26

(b) Shearlets, PSNR 26.82

(c) BM3D, PSNR 30.22

(d) Separable scheme, PSNR 27.45

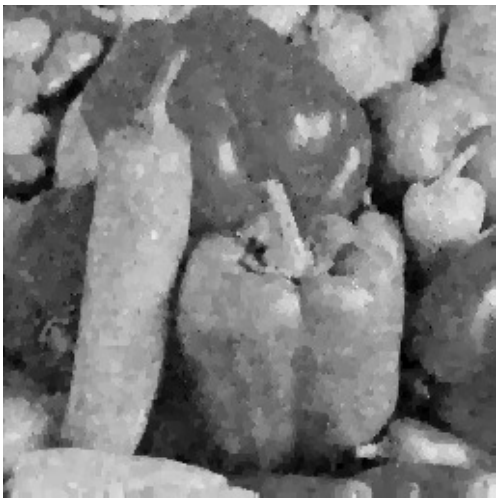
(e) Persistence-weighted ROF, PSNR 28.23

Fig. 6.4: Denoising results of peppers image with $\sigma = 0.1$

(a) Four-pixel scheme, PSNR 26.13

(b) Shearlets, PSNR 25.04

(c) BM3D, PSNR 28.18



(d) Separable scheme, PSNR 25.08

(e) Persistence-weighted ROF, PSNR 26.01

Fig. 6.5: Denoising results for peppers image with $\sigma = 0.15$

Method	Parameters	PSNR
Noisy image		16.45
Four-pixel scheme	122 iterations, $\tau = 0.001$	25.73
Shearlets	$\theta = 0.01$, others default	24.23
BM3D	block size 8×8, others default	27.50
Separable scheme WS1	60 iterations, $\lambda = 0.01$, $\beta = 0.002$	25.24
Persistence-weighted ROF WS1	60 iterations, $\lambda = 0.04$, $\beta = 0.0015$	25.78

Table 6.2: Denoising parameters and denoising results (cameraman, $\sigma = 0.15$).

Method	Parameters	PSNR
Noisy image		19.97
Four-pixel scheme	76 iterations, $\tau = 0.001$	28.26
Shearlets	$\theta = 0.01$, others default	26.82
BM3D	block size 8×8, others default	30.22
Separable scheme WS1	60 iterations, $\lambda = 0.015$, $\beta = 0.005$	27.45
Persistence-weighted ROF WS1	60 iterations, $\lambda = 0.055$, $\beta = 0.002$	28.23

Table 6.3: Denoising parameters and denoising results (peppers, $\sigma = 0.1$).

Method	Parameters	PSNR
Noisy image		16.45
Four-pixel scheme	124 iterations, $\tau = 0.001$	26.13
Shearlets	$\theta = 0.01$, others default	25.04
BM3D	block size 8×8, others default	28.18
Separable scheme WS1	60 iterations, $\lambda = 0.01$, $\beta = 0.001$	25.08
Persistence-weighted ROF WS1	60 iterations, $\lambda = 0.035$, $\beta = 0.001$	26.01

Table 6.4: Denoising parameters and denoising results (peppers, $\sigma = 0.15$).

Bibliography

[1] A. Almansa, C. Ballester, V. Caselles, and G. Haro. A tv based restoration model with local constraints. *J. Sci. Comput.*, 34(3):209–236, 2008.

[2] C.J. Alpert, T.F. Chan, A.B. Kahng, I.L. Markov, and P. Mulet. Faster minimization of linear wirelength for global placement. *IEEE Trans. Computer-Aided Design*, 17(1):3–13, January 1998.

[3] U. Bauer, C. Lange, and M. Wardetzky. Optimal topological simplification of discrete functions on surfaces. *Discrete Comput. Geom.*, 47:347–377, 2012.

[4] U. Bauer, Axel Munk, Hannes Sieling, and Max Wardetzky. Persistence barcodes versus kolmogorov signatures: detecting modes of one-dimensional signals. *preprint, arXiv:1404.1214*, 2014.

[5] U. Bauer, C.-B. Schönlieb, and M. Wardetzky. Total variation meets topological persistence: A first encounter. In *AIP Conf. Proc.*, volume 1281, page 1022, 2010.

[6] G. Bertrand. On topological watersheds. *J. Math. Imaging Vis.*, 22:217–230, 2005.

[7] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. Topological hierarchy for functions on triangulated surfaces. *IEEE Trans. Vis. Comput. Graphics*, 10:385–396, 2004.

[8] E. J. Candès, L. Demanet, D.L. Donoho, and L. Ying. Fast discrete curvelet transforms. *Multiscale Model. Simul.*, 5:861–899, 2006.

[9] G. Carlsson, A. Zomorodian, A. Collins, and L. Guibas. Persistence barcodes for shapes. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 124–135, 2004.

[10] A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximum flows. *Int. J. Comput. Vis.*, 84:288–307, 2009.

[11] A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numer. Math.*, 76(2):167–188, 1997.

[12] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with application to imaging. *J. Math. Imaging Vis.*, 40:120–145, 2011.

[13] T.F. Chan, S. Osher, and J. Shen. The digital tv filter and nonlinear denoising. *IEEE Trans. Image Process.*, 10(2):231–241, 2001.

[14] C. Chen and H. Edelsbrunner. Diffusion runs low on persistence fast. In *IEEE Int. Conference on Computer Vision (ICCV)*, pages 423–430, 2011.

[15] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete Comput. Geom.*, 37:103–120, 2007.

[16] D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and Y. Mileyko. Lipschitz functions have l_p-stable persistence. *Found. Comput. Math.*, 10:127–139, 2010.

[17] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising with block-matching and 3d filtering. In *Proc. SPIE Electronic Imaging*, number no. 6064A-30, San Jose, California, USA, 06.

[18] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3d transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007.

[19] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *IEEE Trans. Image Process.*, 21(4):1715–1728, 2012.

[20] C.J.A. Delfinado and H. Edelsbrunner. An incremental algorithm for betti numbers of simplicial complexes on the 3-sphere. *Comput. Aided Geom. Design*, 12(7):771–784, 1995.

[21] Y. Dong, M. Hintermüller, and M. Monserrat Rincon-Camacho. Automated regularization parameter selection in multi-scale total variation models for image restoration. *J. Math. Imaging Vis.*, 40:82–104, 2011.

[22] H. Edelsbrunner and J. Harer. Persistent homology — a survey. In J.E. Goodman, J. Pach, and R. Pollack, editors, *Surveys on Discrete and Computational Geometry: Twenty Years Later: AMS-IMS-SIAM Joint Summer Research Conference*, volume 453, pages 257–282. American Mathematical Soc., Feb. 29 2008.

[23] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical morse-smale complexes for piecewise linear 2-manifolds. *Discrete Comput. Geom.*, 30:87–107, 2003.

[24] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533, 2002.

[25] H. Edelsbrunner, D. Morozov, and V. Pascucci. Persistence-sensitive simplification functions on 2-manifolds. In *SCG '06 Proceedings of the Twenty-Second Annual Symposium on Computational Geometry*, pages 127–134, 2006.

[26] M. Grasmair. Locally adaptive total variation regularization. *SSVM, Lecture Notes in Computer Science*, 5567:331–342, 2009.

[27] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and Bernd Hamann. Topology-based simplification for feature extraction from 3d scalar fields. In *In Proc. IEEE Conf. Visualization*, pages 275–280, 2005.

[28] P. Kittipoom, G. Kutyniok, and W.-Q Lim. Construction of compactly supported shearlet frames. *Constr. Approx.*, 35:21–72, 2012.

[29] G. Kutyniok, W.-Q Lim, and R. Reisenhofer. Shearlab 3d: Faithful digital shearlet transforms based on compactly supported shearlets. *ACM Trans. Math. Software*, 42(1):to appear, 2015.

[30] G. Kutyniok, W.-Q Lim, and G. Steidl. Shearlets: theory and applications. *GAMM-Mitt.*, 37:259–280, 2014.

[31] W.-Q Lim. The discrete shearlet transform : A new directional transform and compactly supported shearlet frames. *IEEE Trans. Image Process.*, 19:1166–1180, 2010.

[32] J.R. Munkres. *Elements of Algebraic Topology*. Addison-Wesley, Redwood City, 1984.

[33] S. Osher and J. Shen. *"Digitized PDE method for data restoration" in "Handbook of analytic-computational methods in applied mathematics"*. Boca Raton, FL: Chapman & Hall/CRC, 2000.

[34] G. Plonka and J. Ma. Nonlinear regularized reaction-diffusion filters for denoising of images with textures. *IEEE Trans. Image Process.*, 17(8):1283–1294, 2008.

[35] G. Plonka and Y. Zheng. Relation between total variation and persistence distance and its application in signal processing. *preprint, http://na.math.uni-goettingen.de/pdf/plonka_zheng.pdf*, 2014.

[36] J. B. T. M. Roerdink and A. Meijster. The watershed transform: definitions, algorithms, and parallelization strategies. *Fundam. Inform.*, 41:187–228, 2000.

[37] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D.*, 60:259–268, 1992.

[38] D. M. Strong, J.-F. Aujol, and T.F. Chan. Scale recognition, regularization parameter selection, and Meyer's G-norm in total variation regularization. *Multiscale Model. Simul.*, 5(1):275–303, 2006.

[39] D. M. Strong and T. F. Chan. Spatially and scale adaptive total variation based regularization and anisotropic diffusion in image processing. *UCLA Math Department CAM Report*, pages 46–96, 1996.

[40] J. Weickert. *Anisotropic diffusion in image processing*. Teuber Verlag, Stuttgart, 1998.

[41] M. Welk, J. Weickert, and G. Steidl. *A four-pixel scheme for singular differential equations*, volume 3459 of *Scale Space and PDE Methods in Computer Vision*, pages 610–621. 2005.

[42] A. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete Comput. Geom.*, 33:249–274, 2005.

[43] ShearLab-PPFT-1.0. *http://www.shearlab.org/software*, downloaded on 14 May 2014.

Curriculum Vitae

M. Sc. Yi Zheng

Address	Institute for Numerical and Applied Mathematics University of Göttingen Lotzestraße 16-18 37083 Göttingen / Germany
Tel.: (office)	+49/551/20075
Email:	y.zheng@math.uni-goettingen.de
Homepage:	http://na.math.uni-goettingen.de/index.php?section=zheng

---

Personal Details

Gender	Male
Date of birth	18 September 1984
Place of birth	Hanyuan, Ya'an, Sichuan, China
Citizenship	Chinese

Education

09/2011-06/2015	Ph.D. student of mathematics at University of Göttingen (Germany). Supervisor: G. Plonka-Hoch.
09/2010-07/2011	Ph.D. student of computer science at University of Electronic Science and Technology of China (China). Supervisor: J.P. Li.
09/2007-07/2010	Master student of applied mathematics at the Chengdu University of Technology (China). Supervisor: Y.L. Wang.
09/2003-06/2007	Bachelor student of mathematics at Sichuan Normal University (China).
09/1997-06/2003	Middle school and high school education at Hanyuan No.2 middle school (China).

Publications

[1] G. Plonka and Y. Zheng: Relation between total variation and persistence distance and its application in signal processing, preprint.

Participation in Conferences and Workshops

2014	International Conference on Learning and Approximation (Talk), Dec. 08-12 in Shanghai, China.
	Mathematical Signal Processing and Phase Retrieval, Sep. 01-03 in Göttingen.
	24. Rhein-Ruhr-Workshop (Poster), Jan. 31-Feb. 01 in Bestwig.
2013	Final Colloquium of the Research Training Group 1023, Nov. 29-30 in Göttingen.
	Statistical Issues in Compressive Sensing, Nov. 11-13 in Göttingen.
	Advances in Mathematical Image Processing (AIP) (Talk), Sep. 30-Oct. 02 in Annweiler.
2012	Goslar Workshop 2012 (Talk) (Annual Workshop of the RTG 1023), Oct. 17-19 in Goslar.
	Advances in Mathematical Image Processing, Sep 04-06 in Göttingen.
	2012 Intensive Program on Mathematical Models in Seismology, Aug. 27-Sep. 07 in L'Aquila (Italy).
	22. Rhein-Ruhr-Workshop, Feb 03-04 in Bestwig.
2011	Goslar Workshop 2011 (Talk) (Annual Workshop of the RTG 1023), Oct. 12-14 in Goslar.