Journal of Statistical Software

October 2020, Volume 95, Issue 6. doi: 10.18637/jss.v095.i06

Patch-Wise Adaptive Weights Smoothing in R

Jörg Polzehl
Weierstrass Institute

Kostas Papafitsoros
Weierstrass Institute

Karsten Tabelow
Weierstrass Institute

Abstract

Image reconstruction from noisy data has a long history of methodological development and is based on a variety of ideas. In this paper we introduce a new method called patch-wise adaptive smoothing, that extends the propagation-separation approach by using comparisons of local patches of image intensities to define local adaptive weighting schemes for an improved balance of reduced variability and bias in the reconstruction result. We present the implementation of the new method in an R package **aws** and demonstrate its properties on a number of examples in comparison with other state-of-the art image reconstruction methods.

Keywords: image denoising, patch-wise structural adaptive smoothing, total variation, non-local means, R.

1. Introduction

Impairment of images by noise is one of the problems in image processing. Noise is often inherently connected with the image acquisition process, deteriorates the image quality and hinders image analysis. While the notion of images in general refers to two-dimensional data, the problem of noise reduction also occurs in connection with data in higher dimensions, especially in the context of medical imaging problems.

There is a vast literature on different noise reduction techniques. They typically employ an assumption on the spatial structure of the imaging data. A common and simple assumption is, e.g., that the data is characterized by spatially extended regions of homogeneity that are separated by discontinuities. A more sophisticated assumption replaces local homogeneity by local smoothness (Polzehl and Spokoiny 2008; Polzehl and Tabelow 2012). Alternatively, geometric characterizations using orientation spaces or channels in feature space are discussed, e.g., in Felsberg (2012); Felsberg, Forssén, and Scharr (2006); Florack (2012); Duits, Fuehr, and Janssen (2012); Franken (2008).

Typically, noise reduction methods must balance the variability reduction and the bias of the reconstruction results, realizing some kind of edge preservation. They are based on a large variety of basic methodology, achieving this goal to a different degree. They range from kernel smoothing and local polynomials (Wand and Jones 1995; Fan and Gijbels 1996; Simonoff 1996; Bowman and Azzalini 1997), the filter proposed by Lee (1980), bilateral filtering (Tomasi and Manduchi 1998), scale space methods (Chaudhuri and Marron 2000), to non-linear diffusion methods (Perona and Malik 1990; Scharr and Krajsek 2012; Weickert 1998) in a rather incomplete list.

The pointwise adaptive methods introduced in Lepski and Spokoiny (1997); Spokoiny (1998) and Polzehl and Spokoiny (2003) aim in each design point for an optimal selection of a maximal local bandwidth or largest window of positive weights in kernel or local polynomial regression while controlling the bias of the estimates. A numerically efficient refinement of these methods is provided in Astola, Katkovnik, and Egiazarian (2006) and Foi, Katkovnik, and Egiazarian (2007).

In this paper we consider a class of noise reduction methods that has been introduced under the name adaptive weights smoothing (AWS, Polzehl and Spokoiny 2000) and later refined as propagation-separation approach (PS, Polzehl and Spokoiny 2006), which generalizes several of the concepts above in non-parametric regression. PS has been extended to cover locally smooth images (Polzehl and Spokoiny 2008) or color images (Polzehl and Tabelow 2007). Furthermore, it has been successfully applied to a number of imaging problems in neuroimaging, e.g., in functional magnetic resonance imaging (fMRI, Tabelow, Polzehl, Voss, and Spokoiny 2006; Polzehl, Voss, and Tabelow 2010) or in diffusion-weighted magnetic resonance imaging (dMRI, Tabelow, Polzehl, Spokoiny, and Voss 2008; Becker, Tabelow, Voss, Anwander, Heidemann, and Polzehl 2012; Becker, Tabelow, Mohammadi, Weiskopf, and Polzehl 2014). PS combines local comparisons of image intensities to define adaptive weighting schemes with a multiscale approach which iteratively inspects scale space from very local to large scales.

Buades, Coll, and Morel (2005) introduced an adaptive denoising method also extending some of the filtering concepts above. Instead of comparing pairs of single local image intensities to define adaptive weighting schemes, it uses non-local comparisons of larger patches of intensities. Denoted as non-local means (NLM) it has been successfully applied to many imaging problems, e.g., denoising MRI data (Manjón, Coupé, Martí-Bonmatí, Collins, and Robles 2009). Non-local Bayesian noise reduction methods that improve on the original NLM algorithm are considered, e.g., in Lebrun, Buades, and Morel (2013).

The purpose of this paper is to combine the strength of the multiscale approach of PS and the patch-wise comparison of image intensities of NLM to present a new algorithm which we call patch-wise adaptive smoothing (PAWS). We will demonstrate its performance on a series of two- and three-dimensional images. This will show that PAWS overcomes two of the major drawbacks of AWS (or PS), the occurrence of artificial structure within areas of smooth intensity changes and non-smooth borders between neighboring regions with significantly different intensity.

Furthermore, we will compare the results with those obtained considering the image reconstruction as an energy minimizing problem with penalization, i.e., the total variation (TV, Rudin, Osher, and Fatemi 1992) and the total generalized variation (TGV, Bredies, Kunisch, and Pock 2010) approach. Even though these two approaches belong to a different family of denoising methods than those based on propagation-separation, the comparison is neverthe-

less very relevant. Indeed, there is a clear analogy of the relationship between TV and TGV on the one hand and AWS and PAWS on the other. As we will discuss in the next section, AWS employs a structural assumption of piecewise constant image intensity for the noise-free data. This is very similar to TV regularization where the minimization of the ℓ_1 norm of the gradient leads to its sparsity and thus to piecewise constant reconstructions. Hence, TV and AWS suffer from the same type of artifacts, i.e., cartoon-like structures, which are undesirable when the underlying ground truth image contains also piecewise smooth parts. TGV is a higher order extension of TV which was introduced with the aim to eliminate these artifacts. This is done by incorporating an infimal convolution type combination of the ℓ_1 norms of first and second order derivatives resulting in piecewise affine reconstructions. As we see in the following sections, local polynomial AWS (lpAWS) and the new method PAWS improve AWS in a very similar way by imposing a locally smooth structure on the denoised image. Thus, they improve AWS in the same way that TGV improves TV, leading to more naturally looking denoised images.

In this paper, we give an overview on the package **aws** (Polzehl 2020; Polzehl and Spokoiny 2006; Polzehl, Papafitsoros, and Tabelow 2018)[1] for the R (R Core Team 2020) language and environment for statistical computing and the implementation of the above-mentioned methods for adaptive noise reduction including the new PAWS. We will explain the usage of the corresponding functions by extensive examples with two- and three-dimensional data.

The outline of the paper is as follows: We first review the basic principles of the PS approach and the NLM method. Then, we introduce the new PAWS algorithm. Furthermore, we shortly review the pointwise adaptive estimation procedure and the variational approaches (TV and TGV) before presenting the examples to demonstrate properties of the different methods in various situations. Data and R scripts used to perform the analysis of the examples and to produce the corresponding tables and figures are provided in the electronic supplements.

2. Adaptive weights smoothing

Adaptive weights smoothing for the restoration of images from noisy data was originally introduced in Polzehl and Spokoiny (2000) and refined under the term propagation-separation approach in Polzehl and Spokoiny (2006). It employs a structural assumption on the data, more specifically, a local constant parameter model. It is designed as an iterative multi-scale approach that inspects scale space from local to global and simultaneously infers on both the parameter value and its spatial structure.

2.1. Local constant adaptive weights smoothing (AWS)

Let us assume that data $Y_i \in \mathcal{Y}$ is observed at positions $x_i = (x_{i_1}, \ldots, x_{i_d})$ in a bounded subset $\mathcal{X}$ of a d-dimensional metric space. We assume Y_i to be distributed as $Y_i \sim P_{\theta_i}$, where P_{θ_i}, with density $p(y, \theta_i)$, depends on some local parameter θ_i (typically from $\mathbb{R}^p$) and is a probability distribution with support in $\mathcal{Y}$ from some parametric (typically exponential) family $P_{\theta_i} \in \mathcal{P}_\Theta$.

The structural assumption is formulated such that there exists a partitioning $\mathcal{X} = \bigcup_{n=1,\ldots,N} \mathcal{X}_n$

[1] Polzehl *et al.* (2018) is a prior version of this manuscript that contains numerical results obtained with MATLAB (The MathWorks Inc. 2019) implementations of non-local means and total-variation based algorithms. In this manuscript they are replaced by an own implementation in R to achieve full reproducibility.

into N subsets with $\mathcal{X}_n \cap \mathcal{X}_l = \emptyset$ if $n \neq l$ and $\theta_i \equiv \theta_j$ if $x_i \in \mathcal{X}_n$ and $x_j \in \mathcal{X}_n$ for some n. Literally speaking we assume that within any subset $\mathcal{X}_n$ the parameter θ as a function of x is constant.

The method employs both a distance $\delta(x_i, x_j)$ in design space $\mathcal{X}$ as well as a distance $\eta(\theta_i, \theta_j)$ in parameter space $\mathcal{Y}$. A common choice in case of $\mathcal{X} \subset \mathbb{R}^d$ is the Euclidean distance

$$\delta(x_i, x_j) = \|x_i - x_j\|_2$$

in design space and

$$\eta(\theta_i, \theta_j) = \mathcal{KL}(P_{\theta_i}, P_{\theta_j}) = \int_{\mathcal{Y}} p(y, \theta_i) \log \frac{p(y, \theta_i)}{p(y, \theta_j)} dy$$

in parameter space which is the Kullback Leibler divergence between the probability distributions with parameters θ_i and θ_j at locations x_i and x_j, respectively. Henceforth, we abbreviate the locations with indices i and j.

In general, adaptive weights smoothing can be defined for regular as well as irregular designs $\mathcal{X}$. However, the patch-wise smoothing algorithm that we propose in this paper requires $\mathcal{X}$ to be a one-, two- or three-dimensional grid. Common examples include 2D or 3D images, where $\mathcal{X} \subset \mathbb{R}^d$ $(d = 2, 3)$ is a cube and image intensities Y_i are sampled at rectangular grid points x_i. Let $\mathcal{X}_G$ denote the set of grid points in $\mathcal{X}$ and $\mathcal{I}_G$ a set of indices enumerating the points in $\mathcal{X}_G$.

Adaptive weights smoothing employs an iterative scheme with a sequence of increasing bandwidths $h^{(k)}$ for steps $k = 0, \ldots, k^\star$ alternating the computation of weighted maximum likelihood estimates

$$\hat{\theta}_i^{(k)} = \arg\max_\theta l(Y, W_i^{(k)}; \theta) = \arg\max_\theta \sum_{j \in \mathcal{I}_G} w_{ij}^{(k)} \log(p(Y_j, \theta))$$

and the determination of adaptive weighting schemes $W_i^{(k)} = \{w_{ij}^{(k)}, j \in \mathcal{I}_G\}$. Specifically, the weights $w_{ij}^{(k)}$ at iteration step k are given as the product of two terms: a kernel weight $K_{\text{loc}}(l_{ij}^{(k)})$ with $l_{ij}^{(k)} = \delta(x_i, x_j)/h^{(k)}$ and a component $K_{\text{st}}(s_{ij}^{(k)})$ depending on

$$s_{ij}^{(k)} = N_i^{(k-1)} \cdot \eta(\hat{\theta}_i^{(k-1)}, \hat{\theta}_j^{(k-1)})/\lambda.$$

denoted as statistical penalty for two kernel functions K_{loc} and K_{st}. The term $N_i^{(k-1)} = \sum_j w_{ij}^{(k-1)}$ serves as a proxy for the variance reduction achieved for $\hat{\theta}_i^{(k-1)}$. Note, that the noise variance σ^2 typically enters the function η and needs to be known or has to be estimated. The adaptive weights smoothing (AWS) is summarized by Algorithm 1.

The sequence of bandwidths $h^{(k)}$ is chosen such that for $\lambda = \infty$ the variance of the estimate $\hat{\theta}_i^{(k)}$ is reduced by a factor of c_h compared to $\hat{\theta}_i^{(k-1)}$. The specific value for c_h is not very important, cf. Li, Gilmore, Wang, Styner, Lin, and Zhu (2012). However, $c_h = 1.25$ turned out to be a good compromise between sufficient increase of variance reduction between steps and careful increase of $h^{(k)}$ in order to obtain sufficiently neat coverage of the scale space. The kernels K_{loc} and K_{st} are monotone non increasing functions on $\mathbb{R}^+ \mapsto \mathbb{R}^+$ preferably with compact support. Our default[2] choice is

$$K_{\text{loc}}(x) = \max(0, 1 - x^2), \quad \text{and} \quad K_{\text{st}}(x) = \max(0, \min(1, 4/3(1 - x))).$$

[2] K_{loc} is up to a scale factor the Epanechnikov kernel, see, e.g., Fan and Gijbels (1996). K_{st} is a truncated triangular kernel that exhibits a plateau leading to a stabilization of estimates in the iterative algorithm.

Algorithm 1: Formal outline of adaptive weights smoothing (AWS).

Data: Observations $Y_i \in \mathcal{Y}$ at locations $x_i \in \mathcal{X}_G$.
Initialization: Set $k = 0$, $h^{(0)} = 1$, $w_{ij}^{(0)} = K_{\text{loc}}(\delta(x_i, x_j))$ and initialize $\hat{\theta}_i^{(0)}$ as weighted likelihood or least squares estimate.
while $k \leq k^\star$ **do**

For all locations i and j define

$$w_{ij}^{(k)} = K_{\text{loc}}(l_{ij}^{(k)}) K_{\text{st}}(s_{ij}^{(k)})$$

with

$$l_{ij}^{(k)} = \delta(x_i, x_j)/h^{(k)}$$

and

$$s_{ij}^{(k)} = N_i^{(k-1)} \cdot \eta(\hat{\theta}_i^{(k-1)}, \hat{\theta}_j^{(k-1)})/\lambda.$$

For all i define estimates

$$\hat{\theta}_i^{(k)} = \arg\max_\theta l(Y, W_i^{(k)}; \theta)$$

and calculate

$$N_i^{(k)} = \sum_{j \in \mathcal{I}_G} w_{ij}^{(k)}.$$

end
Result: Adaptively denoised parameters $\hat{\theta}_i^{(k^\star)}$.

The main parameters of the procedure are the number k^* of iterations and the scale parameter λ of the statistical penalty. λ can be chosen independently from the data at hand by checking a so-called propagation condition for simulated data, see Becker and Mathé (2013); Becker (2014). For $\lambda = 0$ the data is not changed during the iteration, the choice $\lambda = \infty$ corresponds to a non-adaptive kernel estimate with kernel function K_{loc} and bandwidth $h^{(k^*)}$. An optimal λ will lie between these two extremes and provide a nearly nonadaptive kernel estimate in case of a globally constant parameter θ.

If the structural assumption is valid the AWS approach possesses interesting properties (Polzehl and Spokoiny 2006): Within the interior of any homogeneous region $\mathcal{X}_n$ the final estimate $\hat{\theta}_i^{(k^\star)}$ is similar to a nonadaptive kernel smoother with a bandwidth $h^{(k^*)}$ as specified by the propagation condition (Becker and Mathé 2013; Becker 2014). On the other hand, two different regions $\mathcal{X}_k$ and $\mathcal{X}_l$ of the partition are separated, i.e., $w_{ij}^{(k^\star)} \simeq 0$ if $x_i \in \mathcal{X}_n$, $x_j \in \mathcal{X}_l$, and the contrast $\eta(\theta_i, \theta_j)$ exceeds some critical value that depends on the size of the two regions. For details, see Polzehl and Spokoiny (2006).

2.2. Local polynomial adaptive weights smoothing (lpAWS)

The final estimate $\hat{\theta}_i^{(k^\star)}$ stabilizes for $k^* \to \infty$. If the structural assumption of a local constant parameter function $\theta(x)$ is not valid the algorithm nonetheless enforces a local constant parameter map, which leads to a cartoon-like appearance for the final estimate. In order to overcome

this drawback and to relax the structural assumption on the data the propagation-separation approach has been generalized for locally smooth functions $\theta(x)$ (Polzehl and Spokoiny 2008; Polzehl and Tabelow 2012). Due to the increasing complexity of the algorithm we restrict ourselves to the case $d = 2$ of a two-dimensional design space, i.e., 2D imaging data.

Specifically, we extend the structural assumption such that within a homogeneous region $x_i, x_j \in \mathcal{X}_n$, the data Y_j can be modeled with additive Gaussian errors $\varepsilon_j \sim N(0, \sigma_j^2)$ as

$$Y_j = \theta_i^\top \Psi(x_{j_1} - x_{i_1}, x_{j_2} - x_{i_2}) + \varepsilon_j,$$

where the components of $\Psi(\xi_1, \xi_2)$ contain values of basis functions

$$\psi_{m_1,m_2}(\xi_1, \xi_2) = \xi_1^{m_1} \cdot \xi_2^{m_2}$$

for integers $m_1, m_2 \geq 0$, $m_1 + m_2 \leq p$ and some polynomial order p.

For a given local model $W_i^{(k)}$ at iteration step k estimates $\hat{\theta}_i^{(k)}$ of θ_i are obtained by local least squares as

$$\hat{\theta}_i^{(k)} = \left(B_i^{(k)}\right)^{-1} \sum_{j \in \mathcal{I}_G} w_{ij}^{(k)} \Psi(x_{j_1} - x_{i_1}, x_{j_2} - x_{i_2}) Y_j,$$

with

$$B_i^{(k)} = \sum_{j \in \mathcal{I}_G} w_{ij}^{(k)} \Psi(x_{j_1} - x_{i_1}, x_{j_2} - x_{i_2}) \Psi(x_{j_1} - x_{i_1}, x_{j_2} - x_{i_2})^\top.$$

At each position x_i (and x_j) the estimates $\hat{\theta}_i$ (and $\hat{\theta}_j$) are given in terms of a local set of basis functions ψ. In order to make $\hat{\theta}_j$ and $\hat{\theta}_i$ comparable we perform a simple linear (coordinate) transformation for $\hat{\theta}_j$ using the local model and denote the result by $\hat{\theta}_{ji}$. At iteration step k the statistical penalty in the Gaussian model above can then be defined as

$$s_{ij}^{(k)} = \frac{1}{\lambda 2 \sigma_i^2} (\hat{\theta}_i^{(k-1)} - \hat{\theta}_{ji}^{(k-1)})^\top B_i^{(k-1)} (\hat{\theta}_i^{(k-1)} - \hat{\theta}_{ji}^{(k-1)}),$$

where σ_i^2 is the (local) variance of ε_i. We refer to Polzehl and Spokoiny (2008) or Polzehl and Tabelow (2012) for a more detailed description and examples using image data.

In general, local polynomial AWS improves the reconstruction results compared to the original AWS procedure in case of locally smooth images. As by lpAWS the cartoon-like appearance of the result is avoided, it enables larger variance reduction, i.e., the use of a larger k^*. However, it also increases model flexibility and therefore requires more extended homogeneous regions $\mathcal{X}_n$ to adapt to small contrasts in θ, i.e., to separate two regions $\mathcal{X}_n$ and $\mathcal{X}_l$ in terms of the adaptive weights $w_{ij}^{(k^\star)} \simeq 0$.

2.3. Non-local means (NLM)

The non-local means filter has been introduced in Buades *et al.* (2005) and is related to the adaptive weights smoothing described above. It requires a regular grid $\mathcal{X}_G$ as design, we will assume an isotropic grid. Instead of using only the data at locations i and j to define adaptive weights w_{ij} it uses vectorized values in vicinities of the locations for comparison. In order to formalize the basic idea we introduce a local patch, or vicinity, $V_i = \{v_l(i) : \|v_l(i) - i\|_1 \leq s\}$ of a design point i, see Figure 1. It contains all $n_s = (2s + 1)^d$ grid points $x_{v_l(i)}$ within a

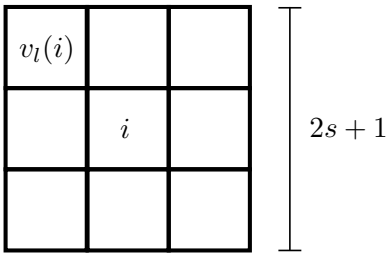

Figure 1: Schematic example of a patch V_i of size s of a location i. It contains all locations $v_l(i)$ with a maximum l_1-distance of s from i.

d-dimensional cube of side length $2s$. The index $l = 1, \ldots, n_s$ varies over the locations $v_l(i)$ in the patch V_i. We denote by Y_{V_i} the vectorized data $(Y_{v_1(i)}, \ldots, Y_{v_{n_s}(i)})$.

The method then defines local adaptive weights

$$w_{ij} = K_{\text{st}}\left(\|Y_{V_j} - Y_{V_i}\|_2/(\sqrt{n_s}\lambda\sigma)\right), \tag{1}$$

where K_{st} is typically chosen as a Gaussian kernel, σ denotes the standard deviation of Y and λ is some bandwidth controlling adaptivity. Thus, NLM is a single step AWS method ($k^\star = 1$) comparing Y_{V_j} and Y_{V_i} instead of Y_j and Y_i with an adaptation bandwidth λ, a uniform location kernel K_{loc}, and $h^{(1)} = \infty$, giving rise to the notion "non-local". Estimation is performed by a weighted mean. For a discussion on the choice of λ we refer to Coupé, Yger, Prima, Hellier, Kervrann, and Barillot (2008).

Two common refinements to reduce the computational burden of the NLM algorithm are to restrict positive weights w_{ij} to a search cube $R_i = \{j : \|j - i\|_1 \leq r\}$ and to reduce the dimension of the comparisons in (1) by using principal component analysis (PCA) on the matrix of patch vectors $(Y_{V_i})_{i \in \mathcal{I}_G}$ and using the first pc principal components instead of the full patch vector (Y_V).

Several further extensions and refinements of the basic method with a large number of applications in the medical imaging context have been proposed and utilized. We refer the reader to the extensive literature on the topic. Examples are the efficient optimized blockwise non-local means denoising filter (ONLM; Coupé *et al.* 2008) and the adaptive multi resolution non-local means filter (MRONLM; Coupé, Manjon, Robles, and Collins 2012). For both reference implementations in MATLAB are available.

3. Patch-wise adaptive weights smoothing (PAWS)

The major drawback of the original and the local polynomial AWS is that their final estimates for large $k^\star$ reflect the strict structural assumption, but do not incorporate any information on the smoothness of boundaries between homogeneous regions. Due to the construction of the method, large noise realizations in some location of the data can be mistaken for structure and lead to a speckled appearance of the final estimate. Here, we thus develop a new extension of AWS by combining its multi-scale approach with the use of information about the local spatial structure in terms of the local patches V_i as defined for NLM.

Algorithm 2: Formal outline of patch-wise adaptive weights smoothing (PAWS).

Data: Observations $Y_i \in \mathcal{Y}$ at locations $x_i \in \mathcal{X}_G$.
Initialization: Set $k = 0$, $h^{(0)} = 1$, $w_{ij}^{(0)} = K_{\text{loc}}(\delta(x_i, x_j))$ and initialize $\hat{\theta}_i^{(0)}$ as weighted likelihood or least squares estimate.
while $k \leq k^\star$ **do**

For all locations i and j define

$$w_{ij}^{(k)} = K_{\text{loc}}(l_{ij}^{(k)}) K_{\text{st}}(s_{ij}^{(k)})$$

with

$$l_{ij}^{(k)} = \delta(x_i, x_j)/h^{(k)}$$

and

$$s_{ij}^{(k)} = \max_{l=1,\ldots,n_s} N_{v_l(i)}^{(k-1)} \cdot \eta(\hat{\theta}_{v_l(i)}^{(k-1)}, \hat{\theta}_{v_l(j)}^{(k-1)})/\lambda.$$

For all i define estimates

$$\hat{\theta}_i^{(k)} = \arg\max_\theta l(Y, W_i^{(k)}; \theta)$$

and calculate

$$N_i^{(k)} = \sum_{j \in \mathcal{I}_G} w_{ij}^{(k)}.$$

end
Result: Adaptively denoised parameters $\hat{\theta}_i^{(k^\star)}$.

This patch-wise adaptive weights smoothing (PAWS) procedure will employ a new form of the statistical penalty $s_{ij}^{(k)}$ based on patches V_i. The variability of the estimates $\hat{\theta}_i$ at iteration step k depends on the (local) weighting schemes $W_i^{(k)}$. This is taken into account in the definition of the statistical penalty $s_{ij}^{(k)}$ by the use of the sum of weights $N_i^{(k)}$. Depending on the unknown underlying structure $\mathcal{X}_n$ the variability of the estimates $\hat{\theta}_{v_l(i)}$ may vary considerably over grid points $v_l(i) \in V_i$. Thus, when we extend the definition $s_{ij}^{(k)}$ to comparisons between patches, it should consider accuracy of the parameter estimates reflected by $N_{v_l(i)}$ as achieved in former iteration steps. We thus define a suitable statistical penalty for PAWS by

$$s_{ij}^{(k)} = \max_{l=1,\ldots,n_s} N_{v_l(i)}^{(k-1)} \cdot \eta(\hat{\theta}_{v_l(i)}^{(k-1)}, \hat{\theta}_{v_l(j)}^{(k-1)})/\lambda.$$

Taking the maximum over all locations $l = 1, \ldots, n_s$ in the patch enables to balance spatial differences in the variance of the estimates.

As for AWS the adaptation bandwidth λ in $s_{ij}^{(k)}$ depends only on the parametric family $P_{\theta_i} \in \mathcal{P}_\Theta$, the dimension of the design space d and, additionally, on the patch size s. We choose it by a propagation condition, see Becker (2014). The algorithm is summarized in Algorithm 2.

We illustrate the propagation of the weights $W_i^{(k)}$ with k for several ground truth situations of the data. The example image is composed of four quadrants with a constant function, two

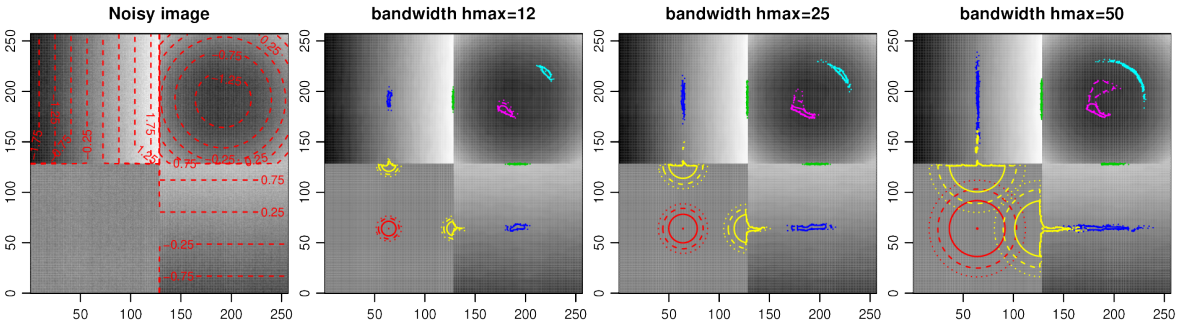

Figure 2: Propagation of weights for PAWS. Left: Noisy image with contour lines of the original structure added. 2nd, 3rd and 4th image (from left): reconstruction results for $k^* = 26, 32$, and 38 ($h_{\max} = 12, 25$, and 50) and patch size $s = 2$ overlayed with contour lines, levels 0.1 (dotted), 0.4 (dashed) and 0.7 (solid), of weighting schemes in nine selected locations. Color labeling corresponds to varying scenarios, see text.

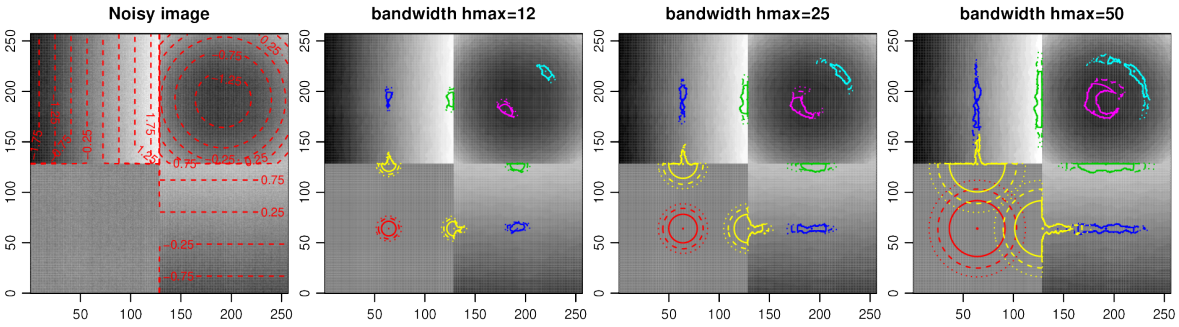

Figure 3: Propagation of weights for AWS. Left: Noisy image with contour lines of the original structure added. 2nd, 3rd and 4th image (from left): reconstruction results for $k^* = 26, 32$, and 38 ($h_{\max} = 12, 25$, and 50) overlayed with contour lines, levels 0.1 (dotted), 0.4 (dashed) and 0.7 (solid), of weighting schemes in nine selected locations. Color labeling corresponds to varying scenarios, see text.

linear functions with different gradient orientation and strength and a quadratic function, plus additive Gaussian noise. In the left of Figure 2 we plotted the noisy image and contour lines of the noise-free image for guidance. The other three plots provide the reconstruction results $\theta^{(k)}$ after $k = 26$, 32 and 38 iteration steps of the PAWS algorithm with a patch size $s = 2$. For these iteration steps and for nine selected locations i we overlay contour lines of the weights $w_{ij}^{(k)}$ corresponding to weights of 0.1 (dotted), 0.4 (dashed) and 0.7 (solid). We use different colors to indicate different scenarios: red for a location within a local constant region, blue in a region with a constant gradient, yellow for locations without intensity contrast at the border of two quadrants, green for locations at discontinuities and cyan/magenta for two locations with quadratic intensity profile but differing distance from the intensity minimum.

It can be seen that within a local constant intensity region the weights propagate isotropically in all directions as for a non-adaptive kernel smoother (red). For locations within a region with constant gradient the weighting schemes are more concentrated in gradient direction while freely extending in the orthogonal direction (blue). The behavior at discontinuities

depends on the image contrast and the distance to the discontinuity. We observe either propagation within the homogeneous region (yellow) or only along the discontinuity if the distance is less than s (green). In case of a nonlinear intensity map the weighting schemes extend and concentrate along level sets with propagation restricted by their curvature.

Figure 3 provides the corresponding illustration for the original AWS procedure. Within the local smooth regions we observe, with increasing bandwidth $h^{(k)}$, the formation of a local constant image reconstruction and the propagation of weights within its constant segments.

4. Pointwise adaptive methods

Pointwise adaptive methods were introduced in 1D nonparametric regression in Lepski, Mammen, and Spokoiny (1997) and Lepski and Spokoiny (1997). These methods are based on an adaptive choice of bandwidths from a sequence of increasing bandwidths $h_0 < h_1 < \cdots < h_{k^*}$ in each design point x_i. Specifically, they select the largest bandwidth h_k for which the kernel estimates $\hat{\theta}_{h_k}(x_i)$ do not significantly differ from any kernel estimate $\hat{\theta}_{h_l}(x_i)$ with $l < k$. The selected bandwidth $h(x_i) = h_k$ then reflects the local smoothness of the regression curve.

This basic principle was modified in Spokoiny (1998) replacing the bandwidth selection with a search for a largest $V(x_i)$ from some set $\mathcal{V}(x_i)$ of intervals (windows) containing x_i where the estimate $\hat{\theta}_{V(x_i)}(x_i)$, employing a uniform kernel on $V(x_i)$, does not significantly differ from any estimate $\hat{\theta}_U(x_i)$ with $U \subset V(x_i)$ and $U \in \mathcal{V}(x_i)$. This approach was generalized to image denoising in 2D in Polzehl and Spokoiny (2003). The algorithm provided was computationally expensive due to the complicated set of utilized windows.

A numerical efficient modification was introduced under the term intersecting confidence intervals (ICI) in Astola *et al.* (2006), see also Foi *et al.* (2007) for a more concise description. In 1D the approach in principle follows Spokoiny (1998). For images the method relies on the partition of a circular region into a specified number S of sectors. For each sector kernel estimates are used, where the support of the kernel is restricted to the specified sector. These estimates can, for a specified bandwidth h, be efficiently computed using convolution by fast Fourier transforms.

The method employs a sequence of bandwidths $h_0 < h_1 < \cdots < h_{k^*}$ and selects, for each sector s, the largest k such that

$$\hat{\theta}^{(s)}_{h_k}(x_i) \in I_l(x_i) \quad \forall l < k,$$

where $I_l(x_i) = [\underline{\theta}^{(s)}_{h_l}(x_i), \bar{\theta}^{(s)}_{h_l}(x_i)]$ is a confidence interval for $\hat{\theta}^{(s)}_{h_l}(x_i)$ at a specified significance level α. This leads, in each design point x_i and for each sector s, to a bandwidth $h^{(s)}(x_i)$ and a corresponding estimate $\hat{\theta}^{(s)}_{h^{(s)}(x_i)}(x_i)$. In an optional step the estimates are stabilized by local median filtering of the estimated bandwidths in each sector. In a final step the estimates obtained for the S sectorial kernels are combined as a weighted mean, with weights proportional to the variance of the individual estimates or, alternatively, selecting the least variable sectorial kernel estimate. Figure 4 illustrates the properties of the method for a noisy parrot image used in Section 7.1. The figure shows, for 10 locations x_i, the support of 12 sectorial Epanechnikov kernels with estimated local bandwidths.

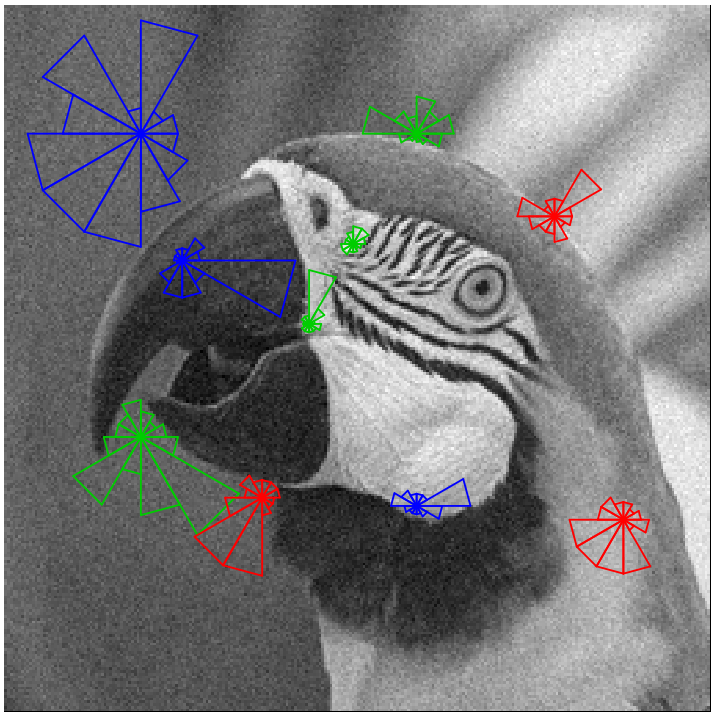

Figure 4: ICI: adaptive kernel support.

5. Adaptive smoothing based on regularization

For data Y on regular Cartesian grids in two dimension, i.e., classical images, approaches based on the numerical solution of optimization problems have been established. They often employ an ℓ_1-regularization term that enforces sparsity of structures where the qualitative local assumptions are violated. The penalties account for deviations from a constant image intensity or constant gradient image. The optimization problem considered below codes similar structural assumptions as the AWS procedures considered in the last section.

5.1. Total variation regularization (TV)

Total variation is a classical energy minimizing method (Rudin *et al.* 1992), where the denoised image U is obtained as a minimizer of the energy

$$\min_U \frac{1}{2}\|U - Y\|_2^2 + \mathrm{TV}_\alpha(U). \tag{2}$$

Here $\|U - Y\|_2^2$ denotes the square of the Euclidean distance between U and Y (discrepancy term), which corresponds to the fact that the data Y is assumed to be corrupted by Gaussian noise. For an image U defined on a Cartesian grid $d_1 \times d_2$, the term $\mathrm{TV}_\alpha(U)$ is the total variation of U which in this discrete formulation reads

$$\mathrm{TV}_\alpha(U) = \alpha\|\nabla U\|_1 = \alpha \sum_{i_1=1}^{d_1}\sum_{i_2=1}^{d_2}((U_{i_1+1,i_2} - U_{i_1,i_2})^2 + (U_{i_1,i_2+1} - U_{i_1,i_2})^2)^{1/2}.$$

The scalar constant $\alpha > 0$ balances the two terms in (2) and determines the amount of filtering. Minimizing the ℓ_1 norm of the gradient results in sparsity of the gradient of the

denoised image. As a result, total variation, like adaptive weights smoothing with a local constant structural assumption, promotes piecewise constant reconstructions which on one hand leads to edge preservation and sharp images. On the other hand it also leads to blocky artifacts (staircasing effect) which is often undesirable in natural images. TV reconstructions are typically accompanied by a certain loss of contrast. There is a vast literature concerning TV minimization. Here we refer the reader to Caselles, Chambolle, and Novaga (2007) and Ring (2000) for analytical properties, and to Hintermüller, Rautenberg, Wu, and Langer (2017) for parameter selection as well as to the references therein.

5.2. Total generalized variation regularization (TGV)

Total generalized variation (Bredies *et al.* 2010) has been proposed as a higher order extension of TV, aiming to avoid the staircasing effect and on the same time to retain the ability of TV to preserve edges. In the case of Gaussian noise, one minimizes the functional

$$\min_U \frac{1}{2}\|U - Y\|_2^2 + \mathrm{TGV}_{\alpha,\beta}(U),$$

where

$$\mathrm{TGV}_{\alpha,\beta}(U) = \min_W \ \alpha\|\nabla U - W\|_1 + \beta\|\mathcal{E}W\|_1.$$

Here, $\mathcal{E}W$ is the symmetrized gradient of W, i.e., $\mathcal{E}W = \frac{1}{2}(\nabla W + \nabla W^\top)$, and $\alpha, \beta > 0$. If $W = 0$ then $\mathrm{TGV}_{\alpha,\beta}(U) = \mathrm{TV}\alpha(U)$. If $W = \nabla U$ then $\mathrm{TGV}_{\alpha,\beta}(U) = \beta\|\nabla^2 U\|_1$. Thus, the TGV functional can be interpreted as an optimal balance between first and second order ℓ_1-type regularization. TGV minimization promotes piecewise affine reconstructions rather than piecewise constant, typically resulting in more visually pleasing results than TV. One drawback of TGV minimization is the presence of two regularization parameters that need to be balanced, see Papafitsoros and Bredies (2015); De los Reyes, Schönlieb, and Valkonen (2017), as well as the increased computational cost in comparison to TV. If the ground truth is unknown a bilevel optimization problem in the spirit of Hintermüller *et al.* (2017) can be devised in order to automatically select the regularization parameters.

6. The package aws

The package **aws** (Polzehl 2020) emerged as a reference implementation of structural adaptive smoothing algorithms (Polzehl and Spokoiny 2000, 2006, 2008) and now provides a broad collection of adaptive smoothing methods in 1D, 2D and 3D. All methods assume the data to be given as a one-, two- or three-dimensional array, which we name `nimg` in the following.

The first class of methods consists of the structural adaptive smoothing methods introduced in Sections 2.1, 2.2 and 3 and implemented in functions `aws`, `lpaws` (1D, 2D only) and `paws`[3]. Smoothing of a 1D, 2D or 3D image `nimg` using a maximum bandwidth `hmax` is performed by

```
R> imghataws <- aws(nimg, hmax)
R> imghatlpaws <- lpaws(nimg, degree = 1, hmax)
R> imghatpaws <- paws(nimg, hmax, patchsize = 1)
```

[3]The package also contains a function `pawsm` restricting all computations to a mask specified as its second argument `mask`.

The functions carry arguments to specify the location kernel K_{loc} (`lkern`), with the default specifying the (spherical) Epanechnikov kernel, the parametric family $\mathcal{P}_\Theta$ (`family`) and a variance estimate (`sigma2`) in case of Gaussian image intensities. The default adaptation bandwidth λ fulfills an propagation condition (Becker and Mathé 2013) at level $\alpha =$`1e-4`[4].

If `sigma2` is not provided we use the interquartile range of consecutive intensity differences, i.e.,

```
R> sigma2 <- (IQR(diff(as.vector(nimg)))/1.908)^2
```

as an estimate that is robust with respect to discontinuities in the image. Functions `vaws` and `vpaws` provide implementations for vector valued images. The local polynomial method `lpaws` is restricted to one- and two-dimensional data. The result is an object of S4 class '`aws`' with component `theta` containing the smoothed image.

Special functions exist for the case of irregular design $\mathcal{X}_G$ (`aws.irreg`) and Gaussian data with mean dependent variance (`aws.gaussian`). Function `aws.segment` provides the structural adaptive segmentation algorithm introduced in Polzehl *et al.* (2010).

The non-local means filter, see Section 2.3, is employed using

```
R> imghatNLM <- nlmeans(nimg, lambda, sigma)
```

where `lambda` specifies the bandwidth λ of kernel K_{st} and `sigma` specifies an estimate of the error standard deviation. The function returns a list of class '`nlmeans`' with component `theta` containing the one-, two- or three-dimensional reconstructed image.

The adaptive weights and non-local means algorithms are computationally demanding and use Fortran code parallelized using **openMP**.

The regularization methods TV and TGV, see Section 5, are provided by `TV_denoising` and `TGV_denoising`:

```
R> imghtTV <- TV_denoising(nimg, alpha, iter)
R> imghatTGV <- TGV_denoising(nimg, alpha, beta, iter)
```

where `alpha` (and `beta`) are the regularization parameters for TV and TGV as described above. Iterations of the optimization algorithm are terminated if either the ℓ_1-norm or ℓ_∞-norm of the difference of two consecutive image reconstructions are less than their specified tolerance values, or the maximum number `iter` of iterations is reached. The implementation is restricted to two-dimensional data. Both functions return a matrix.

A fourth class of methods enables pointwise adaptive smoothing using the intersection of confidence intervals (ICI) method introduced in Astola *et al.* (2006):

```
R> imghatICIC <- ICIcombined(nimg, hmax, thresh, kern, sigma, nsector,
+    presmooth)
R> imghatICIS <- ICIsmooth(nimg, hmax, thresh, kern, sigma, nsector,
+    sector, presmooth)
R> imghatksm <- kernsm(nimg, h, kern, nsector, sector)
```

[4]We do not recommend changing the default.

`ICIcombined` computes the ICI estimated described in Section 4. The function `ICIsmooth` performs the bandwidth selection within a single circular sector, while the function `kernsm` provides kernel estimates restricted to the specified sector. The default parameters `nsector = 1, sector = 1` provide a non-adaptive estimate with the specified kernel or derivative kernel. The parameter `hmax` refers to the maximum bandwidth h_{k^*}, `thresh` is the quantile of a Gaussian distribution that determines the coverage probabilities of the confidence intervals, `kern` determines the type of kernel (defaults to `"Gaussian"`), `nsector` is the number of sectors to use (2 in 1D, any positive integer in 2D, 8 in 3D) and `presmooth` (logical) determines if spatial median filtering is applied on the estimated bandwidths. The functions create S4 objects of class '`ICIsmooth`' or '`kernsm`' with reconstructed images in component `yhat`.

Additionally the package contains the function `qmeasures` to evaluate the criteria listed in the beginning of Section 7 and `summary`, `print` and `plot` methods for the reconstructed images. For more information see the extensive documentation of the package.

We end this section with a (certainly incomplete) overview over the capabilities and utilized methodologies of alternative packages: One-dimensional denoising can be performed in R using a large variety of methods that are implemented in R, i.e., packages **stats** and **splines** (R Core Team 2020), and recommended or contributed packages like, e.g., **KernSmooth** (Wand 2020). There is only a small number of additional packages that enable denoising of 2D and 3D images available from the Comprehensive R Archive Network (CRAN) and Neuroconductor (Muschelli *et al.* 2019) websites. They can be categorized by the class of models they use.

The package **mmand** (Clayden 2020) contains a function `gaussianSmooth` that performs kernel smoothing using a Gaussian kernel, similar to our function `kernsm` with default arguments. Non-adaptive smoothing is also available within the interface packages **ANTsR** (Avants 2020) and **fslr** (Muschelli, Sweeney, Lindquist, and Crainiceanu 2015; Jenkinson, Beckmann, Behrens, Woolrich, and Smith 2012) available from Neuroconductor. Edge preserving smoothing of 2D images by anisotropic diffusion is offered in package **imager** (Barthelme 2020) with function `blur_isotropic`.

Total variation regularization for 2D images is also implemented in the package **tvR** (You 2019). Image denoising using wavelets can be done with packages **wavethresh** (Nason 2016, 2D and 3D) and **waveslim** (Whitcher 2020, 2D only). Denoising using a discrete cosine transform (DCT) in 2D is implemented in **imagerExtra** (Ochi 2018). The orthogonal series based approaches can adapt to the image structure and show, in our experience, a performance similar to AWS and pointwise adaptive methods. These methods rely on numerically efficient algorithms and are, compared to non-local means and adaptive weights algorithms, less computationally intensive.

7. Examples

We illustrate and discuss the performance of our patch-wise adaptive weights smoothing (PAWS) algorithm in comparison to the other adaptive smoothing procedures (AWS, lpAWS, NLM, ICI, TV, TGV) in three examples with artificial Gaussian noise. Data and R scripts to reproduce these examples are published alongside this paper. For the evaluation of the reconstruction $\hat{U}$ in comparison to the ground truth image U we use the following common criteria:

- Peak signal to noise ratio (PSNR):

$$\mathrm{PSNR}(\hat{U}, U) = 20 \log_{10}(\max(U) - \min(U)) - 10 \log_{10}(\mathrm{var}(\hat{U} - U))$$

- Mean absolute error (MAE):

$$\mathrm{MAE}(\hat{U}, U) = \mathrm{mean}(|\hat{U} - U|)$$

- Structural similarity (SSIM), see Wang, Bovik, Sheikh, and Simoncelli (2004):

$$\mathrm{SSIM}(\hat{U}, U) = \frac{(2 \cdot \mathrm{mean}(\hat{U})\,\mathrm{mean}(U) + c_1)(2 \cdot \mathrm{cov}(\hat{U}, U) + c_2)}{(\mathrm{mean}(\hat{U})^2\,\mathrm{mean}(U)^2 + c_1)(\mathrm{var}(\hat{U})^2\,\mathrm{var}(I)^2 + c_2)}$$

  with $c_1 = 10^{-4}(\max(U) - \min(U))^2$ and $c_2 = 9c_1$.

- Mean absolute gradient error (MAGE):

$$\mathrm{MAGE}(\hat{U}, U) = \mathrm{mean}(|\nabla\hat{U} - \nabla U|)$$

  using a numeric approximation of the gradient.

- Root mean squared gradient error (RMSGE):

$$\mathrm{RMSGE}(\hat{U}, U) = \sqrt{\mathrm{var}(\nabla\hat{U} - \nabla U)}.$$

To access and manipulate the grayscale and color images in Examples 1 and 2 we exploit the package **adimpro** (Tabelow and Polzehl 2019; Polzehl and Tabelow 2007).

7.1. Example 1 – Grayscale parrot image

The first example uses a grayscale image (resolution 256×256) extracted from `kodim23.png` (`http://r0k.us/graphics/kodak/`), see the left image in Figure 5. Spatially independent Gaussian noise with four different standard deviations, $\sigma = 0.04, \sigma = 0.08$, $\sigma = 0.16$ and $\sigma = 0.32$ was added after standardizing the image to a range $[0, 1]$. For each noise level we computed reconstructions using AWS, with k^* optimized with respect to PSNR; lpAWS, with polynomial degree $p = 1$; TV, TGV, NLM, ICI and our new PAWS method with patch sizes $s = 1$ and 2. For PAWS and lpAWS the number of iterations k^* used was 18, 22, 24 and 24, corresponding to a maximum bandwidth $h^{(k^*)}$ of 4.9, 7.6, 9.5 and 9.5, for the four noise levels. For the adaptation bandwidth λ we use the defaults in the functions `aws`, `lpaws` and `paws`[5]. Parameters for TV (α), TGV (α and β), NLM (λ) and ICI (`hmax`, `thresh`, `nsector`) were optimized with respect to PSNR for each noise level. The error standard deviation was provided.

Numerical results with respect to PSNR, MAE and SSIM are provided in Tables 1–3. They suggest a superior behavior of PAWS especially in case of low SNR. Figures 6 and 7 provide the reconstructions achieved by AWS, PAWS with $s = 2$, lpAWS with a degree $p = 1$, TV, TGV, NLM and ICI, for medium and low SNR ($\sigma = 0.08, 0.16$), together with the noisy source images. All images are projected into the range of the original for display in order to improve comparability.

[5]These values are determined by simulation to obey a propagation condition (Becker and Mathé 2013; Becker 2014) at level $\alpha =$ `1e-4`.

Figure 5: Example 2D images: grayscale image and color (sRGB) image.

	noisy	TV	TGV	AWS	PAWS_1	PAWS_2	lpAWS_1	NLM	ICI
$\sigma = 0.04$	27.4	33.4	33.7	32.2	33.8	33.8	33.1	34.1	34.0
$\sigma = 0.08$	21.3	29.8	30.1	28.3	30.6	30.6	29.7	30.5	30.3
$\sigma = 0.16$	15.3	26.4	26.7	25.7	27.4	27.8	25.9	26.7	26.2
$\sigma = 0.32$	9.3	23.4	23.6	23.0	23.8	24.5	22.8	22.4	22.1

Table 1: PSNR of parrot reconstructions for the different noise levels. The index for PAWS corresponds to the patch size ($s = 1, 2, 3$).

	noisy	TV	TGV	AWS	PAWS_1	PAWS_2	lpAWS_1	NLM	ICI
$\sigma = 0.04$	0.0320	0.0141	0.0135	0.0166	0.0131	0.0131	0.0138	0.0132	0.0132
$\sigma = 0.08$	0.0641	0.0201	0.0191	0.0237	0.0182	0.0180	0.0191	0.0198	0.0195
$\sigma = 0.16$	0.1281	0.0287	0.0272	0.0297	0.0253	0.0244	0.0279	0.0303	0.0304
$\sigma = 0.32$	0.2563	0.0404	0.0384	0.0401	0.0379	0.0363	0.0420	0.0504	0.0470

Table 2: As Table 1 but reporting MAE.

	noisy	TV	TGV	AWS	PAWS_1	PAWS_2	lpAWS_1	NLM	ICI
$\sigma = 0.04$	0.9843	0.9960	0.9962	0.9948	0.9964	0.9964	0.9957	0.9966	0.9965
$\sigma = 0.08$	0.9400	0.9908	0.9913	0.9872	0.9923	0.9924	0.9907	0.9922	0.9918
$\sigma = 0.16$	0.7966	0.9795	0.9807	0.9766	0.9841	0.9855	0.9772	0.9806	0.9783
$\sigma = 0.32$	0.4947	0.9574	0.9598	0.9551	0.9629	0.9684	0.9521	0.9467	0.9440

Table 3: As Table 1 but reporting SSIM.

7.2. Example 2 – Color image in 2D

The AWS and PAWS algorithms can be easily extended to handle color images using $\eta(\theta_i, \theta_j) = (\theta_i - \theta_j)^\top \Sigma_i^{-1} (\theta_i - \theta_j)$ where $\theta \in \mathbb{R}^3$ is a vector of intensities in the RGB channels and the corresponding Σ_i the error covariance matrix, see Polzehl and Tabelow (2007). The extension of TV and TGV to color images is straightforward as well.

For a color image example we used the right image in Figure 5. We added Gaussian noise with standard deviation equal to 15% of the intensity range in each of the RGB channels of the image. The reconstructions using TV and TGV were obtained using optimized parameters

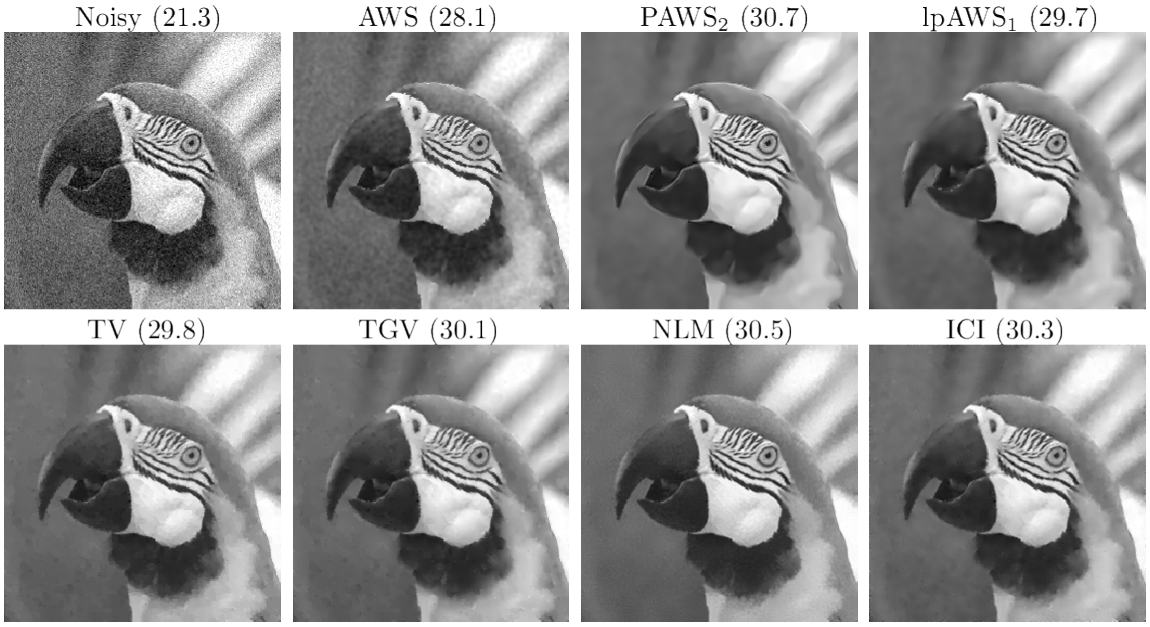

Figure 6: Reconstructions for noisy parrot image with $\sigma = 0.08$. The numbers in parentheses are the PSNR values.

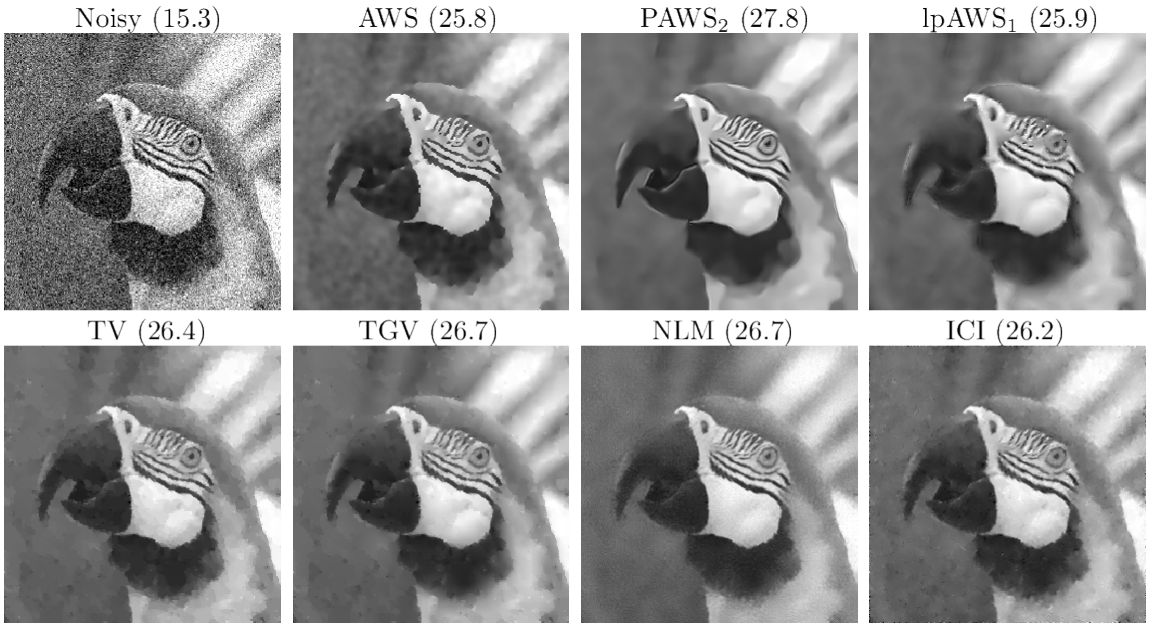

Figure 7: Reconstructions for noisy parrot image with $\sigma = 0.16$. The numbers in parentheses are the PSNR values.

α (and β) with respect to PSNR. For AWS and PAWS we used $k^* = 24$ (corresponding to $h^{(k^*)} = 9.5$) and adjusted the parameter λ to maximize PSNR. Figure 8 shows a zoom of the noisy image and its reconstructions. The intensity values have again been projected into the

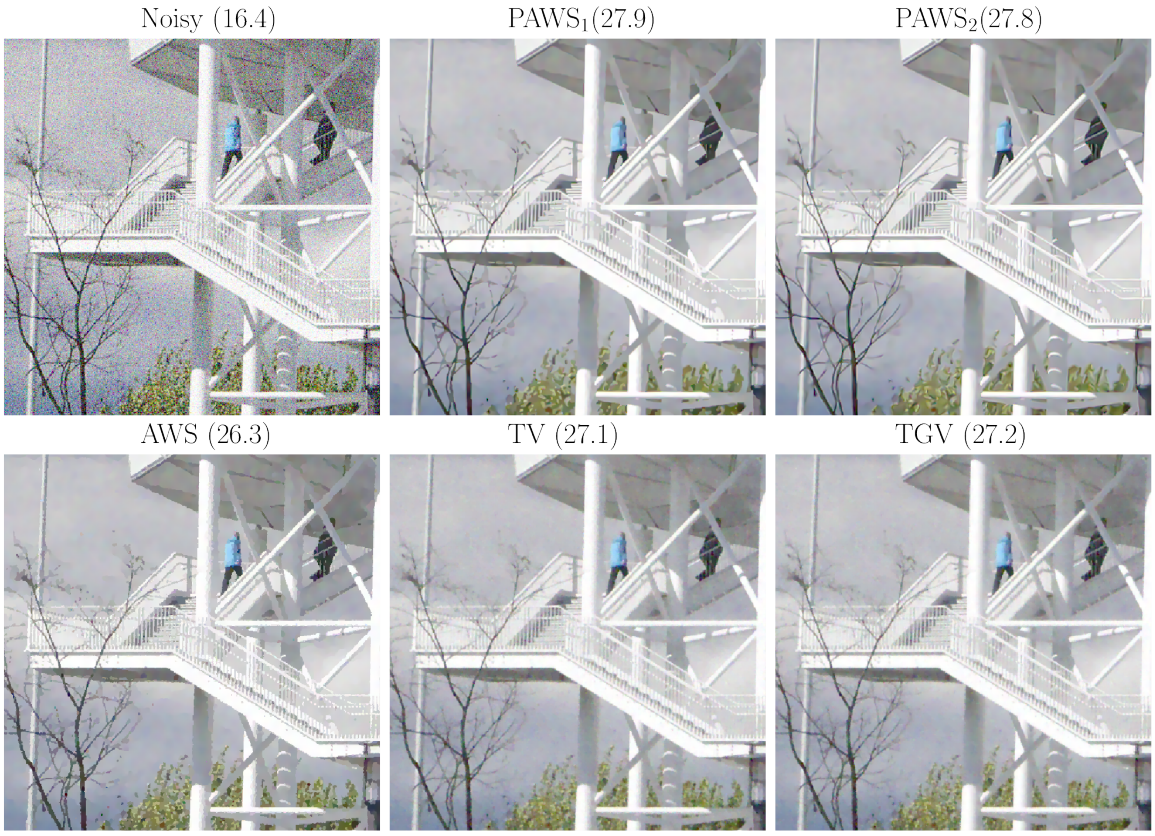

Figure 8: Reconstruction results for the color image. For more details a zoomed image is show. PSNR values are in parentheses, see also Table 4.

	noisy	AWS	PAWS_1	PAWS_2	TV	TGV
PSNR	16.4	26.3	27.9	27.8	27.1	27.2
MAE	7850	1920	1660	1680	1900	1880

Table 4: PSNR and MAE of the reconstruction results for the color image example.

range $[0, 1]$ in each RGB channel for better comparability. PSNR and MAE results reported in the headings of each image correspond to the full image size of 1700×1400 pixel. Note, that the assumption of a local constant model is enforced in the AWS reconstruction in regions with smoothly changing intensities. This effect is by far less prominent with both the PAWS and TV/TGV reconstructions.

7.3. Example 3 – 3D brain image

Our third example uses a 3D T1-weighted image volume with a 1mm isotropic voxel resolution from BrainWeb (`http://brainweb.bic.mni.mcgill.ca/cgi/brainweb1`; Kwan, Evans, and Pike 1996; Collins *et al.* 1998; Kwan, Evans, and Pike 1999), see Figure 9. The package **oro.nifti** (Whitcher, Schmid, and Thornton 2011; Whitcher, Schmid, Thornton, and Muschelli 2020) is employed to read the image. The image dimension is $181 \times 217 \times 181$, image intensity ranges between 0 and 4095 (12 Bit).

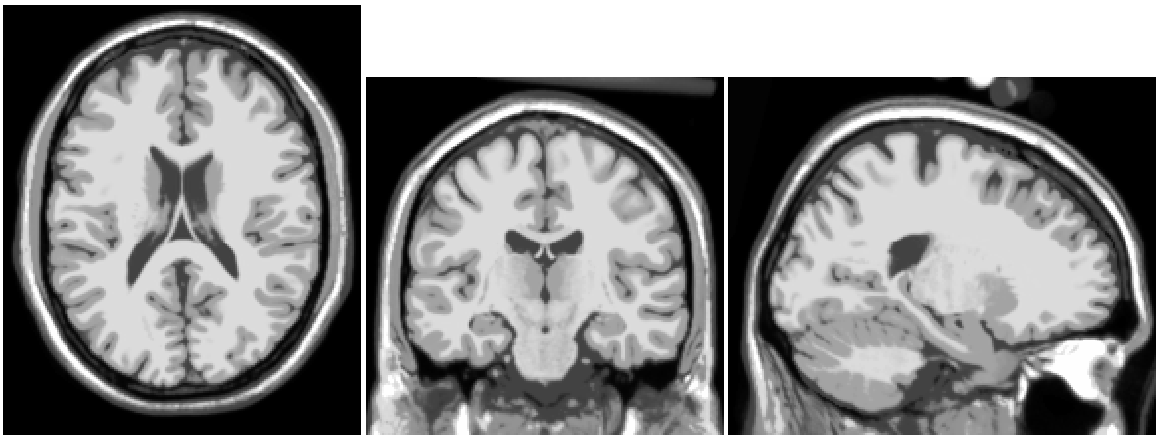

Figure 9: Example 3D brain images (T1w) from BrainWeb: axial (slice #91), coronal (slice #109) and sagittal (slice #70) view.

	noisy	AWS	PAWS_1	PAWS_2	NLMeans	NLMeans wPCA
$\sigma = 200$	26.2	32.7	34.8	35.0	31.2	32.9
$\sigma = 400$	20.2	28.5	32.5	32.2	30.6	31.9
$\sigma = 800$	14.2	25.7	29.3	29.6	26.4	27.7

Table 5: PSNR of brain reconstructions for the different noise levels.

	noisy	AWS	PAWS_1	PAWS_2	NLMeans	NLMeans wPCA
$\sigma = 200$	159.7	70.5	51.3	51.1	80.5	64.0
$\sigma = 400$	319.1	112.4	68.3	71.9	89.1	74.8
$\sigma = 800$	638.3	148.1	99.6	98.5	150.0	126.9

Table 6: MAE of brain reconstructions for the different noise levels.

Images corrupted with noise were obtained by adding spatially independent Gaussian noise with a standard deviation 200 (high SNR), 400 (medium SNR) and 800 (low SNR) to each voxel intensity. For reconstruction we used AWS and PAWS with default values of λ chosen according to a propagation condition (Becker and Mathé 2013). The value of k^* for AWS was selected to provide best results in terms of PSNR (specifically 11, 13 and 20 for the three situations). For PAWS we used $k^* = 23$ ($h^{(k^*)} = 3.85$) for high SNR, $k^* = 26$ ($h^{(k^*)} = 4.81$) for medium SNR and $k^* = 28$ ($h^{(k^*)} = 5.6$) in case of low SNR. For comparison we represent results obtained with the non-local means implementation in package **aws** using a patch size of 27 (patch half width 1; NLMeans) and with optimum number *pc* of principal components (NLMeans wPCA). Parameters `lambda` and `searchhw` were optimized for each image with respect to PSNR. A comparison with results using the original MATLAB implementations of the efficient optimized blockwise non-local means denoising filter (ONLM; Coupé *et al.* 2008) and the adaptive multi resolution non-local means filter (MRONLM; Coupé *et al.* 2012) can be found in Polzehl *et al.* (2018).

Figure 10 illustrates the quality of reconstruction for the various methods for the central axial slice. The values of PSNR and MAE are reported in Tables 5 and 6 and refer to all voxel within a 3D brain mask obtained by thresholding the original 3D BrainWeb image.

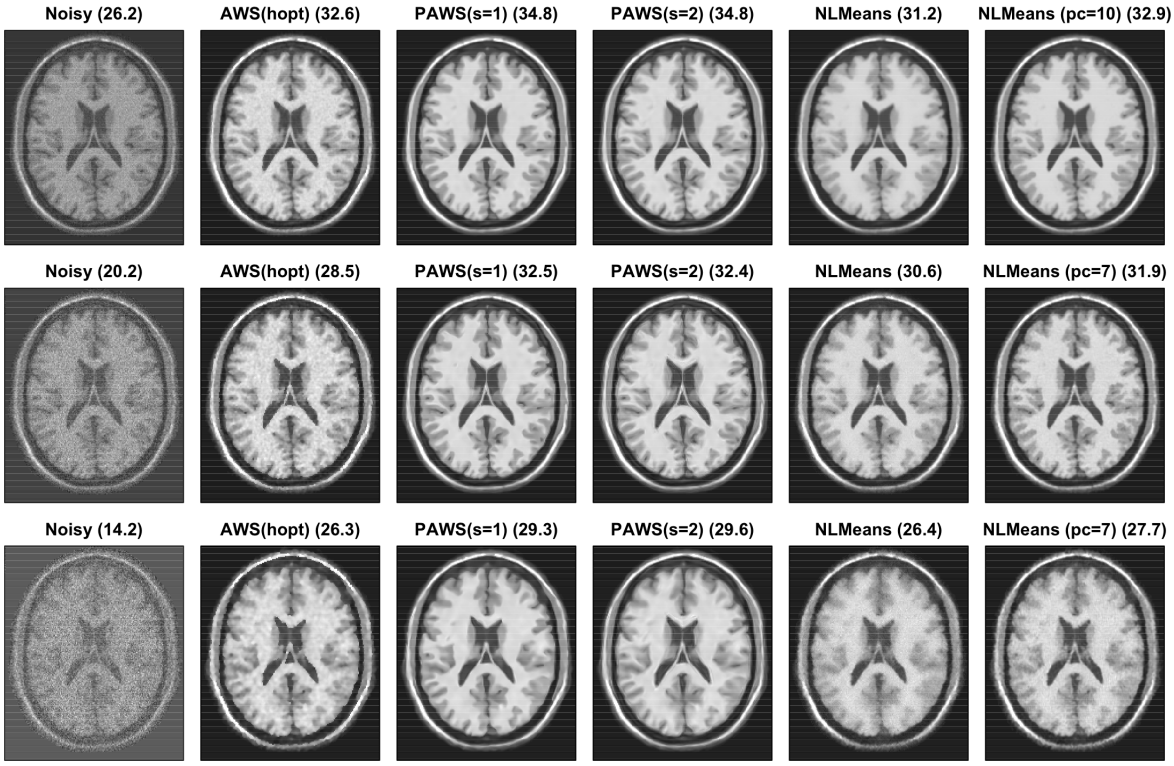

Figure 10: Example 3D brain images (T1w) from BrainWeb: axial view (slice 91) of noisy image, AWS, PAWS ($s = 1, 2$) and NLMeans reconstructions without and with dimension reduction in patch space by PCA. Rows show results for differing noise levels. The numbers is parentheses refer to the PSNR of the 3D reconstruction within the brain mask. Numerical values MAE over all voxel within a brain mask are reported in Tables 5 and 6.

	noisy	AWS	PAWS_1	PAWS_2	NLMeans	NLMeans wPCA
$\sigma = 200$	170.0	60.5	33.3	33.1	48.9	39.5
$\sigma = 400$	362.0	92.7	41.0	42.9	61.4	50.8
$\sigma = 800$	779.0	105.0	58.6	58.0	100.0	82.8

Table 7: MAGE of brain reconstructions for the different noise levels.

	noisy	AWS	PAWS_1	PAWS_2	NLMeans	NLMeans wPCA
$\sigma = 200$	200.0	75.6	53.0	51.3	80.5	65.5
$\sigma = 400$	418.0	115.0	64.4	65.5	81.3	68.8
$\sigma = 800$	879.0	154.0	91.8	88.7	126.0	107.0

Table 8: RMSGE of brain reconstructions for the different noise levels.

In Figure 11 we illustrate the accuracy of edge estimation. We show the central axial slice of a 3D image that contains the norm of the standard numerical gradient approximation as voxel intensity.

Non-local means (with optimized parameters) performs best for high SNR. Using dimension reduction by PCA in patch space consistently improved results with respect to all considered

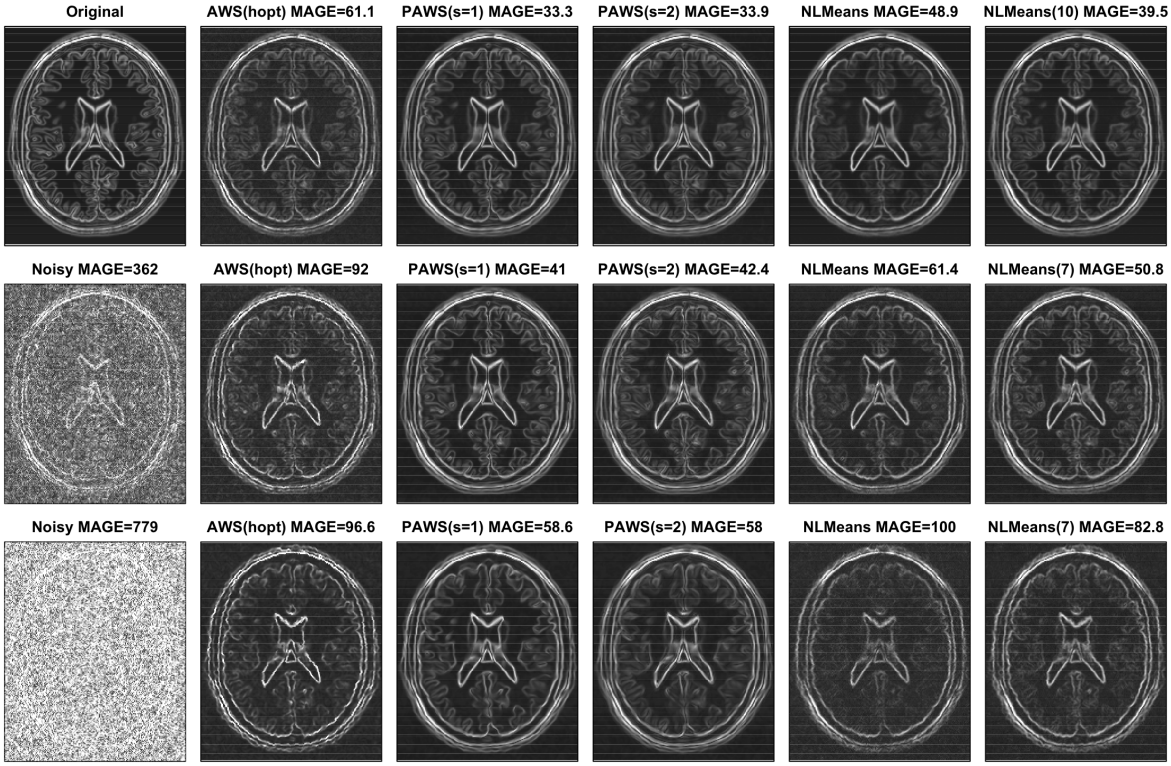

Figure 11: Example 3D brain images (T1w) from BrainWeb: axial view (slice 91) of 3D edge indicators for (from left to right) noisy image (2nd and 3rd row, original is used in the first row for comparison), AWS, PAWS ($s = 1, 2$) and NLMeans reconstructions. Rows show results for differing noise levels. Numerical values in parenthesis refer to MAGE. Numerical values for MAGE and RMSGE are reported in Tables 7 and 8.

criteria. The patch-wise AWS algorithm outperforms the other methods in case of medium and low SNR.

8. Discussion and conclusion

In this paper, we presented a new noise reduction algorithm PAWS based on the propagation-separation approach that combines the multiscale approach of PS with the definition of adaptive weighting schemes based on comparisons of patches of image intensities. It can be easily applied for data in any dimension d. The new PAWS method overcomes the problem of singular locations with extreme image intensity in the reconstruction and the roughness of the boundaries of regions with homogeneous image intensity. The usage of the maximum statistics in the definition of the statistical penalty automatically takes the different variability of the estimates from the previous iteration step into account. PAWS shows also improved performance if the image data is characterized by local smooth instead of local constant intensity regions. It outperforms lpAWS in these cases and is, in contrast to lpAWS, easily applicable for d-dimensional data with $d > 2$.

We also described an implementation of PAWS as well as AWS and lpAWS within the R package **aws** which is freely available under the GPL from CRAN at https://CRAN.R-project.org/package=aws. We demonstrated, that the combination of both ideas, the multiscale approach and the patch-wise comparison, leads to improved reconstruction results in comparison to methods based on a single ingredient. We also demonstrated how the method compares with regularization-based methods like TV and TGV, which in general show similar properties and performance to the PS methods. Additional comparisons include the non-local means filter and the pointwise-adaptive ICI method. We provided a concise description of the principles and algorithmic details of all methods used in the examples. All required functionality is implemented in the R package **aws** that provides a variety of adaptive smoothing methods in 1D, 2D and 3D.

Acknowledgments

Jörg Polzehl would like to thank the Isaac Newton Institute for Mathematical Sciences for support and hospitality during the programme "Variational methods and effective algorithms for imaging and vision" when work on this paper was undertaken. This work was supported by: EPSRC grant numbers EP/K032208/1 and EP/R014604/1.

Kostas Papafitsoros acknowledges the support of the Einstein Foundation Berlin within the ECMath project CH12.

References

Astola J, Katkovnik V, Egiazarian K (2006). *Local Approximation Techniques in Signal and Image Processing*, volume PM157. SPIE Press Monograph. doi:10.1117/3.660178.

Avants BB (2020). ***ANTsR**: **ANTs** in R: Quantification Tools for Biomedical Images*. R package version 0.5.6.2, URL https://neuroconductor.org/package/ANTsR.

Barthelme S (2020). ***imager**: Image Processing Library Based on **CImg***. R package version 0.42.3, URL https://CRAN.R-project.org/package=imager.

Becker S (2014). *The Propagation-Separation Approach – Theoretical Study and Application to Magnetic Resonance Imaging*. Ph.D. thesis, Humboldt-University Berlin, Dept. of Mathematics.

Becker S, Mathé P (2013). "A Different Perspective on the Propagation-Separation Approach." *Electronic Journal of Statistics*, **7**, 2702–2736. doi:10.1214/13-ejs860.

Becker S, Tabelow K, Mohammadi S, Weiskopf N, Polzehl J (2014). "Adaptive Smoothing of Multi-Shell Diffusion-Weighted Magnetic Resonance Data by msPOAS." *NeuroImage*, **95**, 90–105. doi:10.1016/j.neuroimage.2014.03.053.

Becker S, Tabelow K, Voss HU, Anwander A, Heidemann RM, Polzehl J (2012). "Position-Orientation Adaptive Smoothing of Diffusion Weighted Magnetic Resonance Data (POAS)." *Medical Image Analysis*, **16**(6), 1142–1155. doi:10.1016/j.media.2012.05.007.

Bowman AW, Azzalini A (1997). *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-PLUS Illustrations*. Oxford University Press.

Bredies K, Kunisch K, Pock T (2010). "Total Generalized Variation." *SIAM Journal on Imaging Sciences*, **3**(3), 492–526. doi:10.1137/090769521.

Buades A, Coll B, Morel JM (2005). "A Review of Image Denoising Algorithms, with a New One." *Multiscale Modeling & Simulation*, **4**(2), 490–530. doi:10.1137/040616024.

Caselles V, Chambolle A, Novaga M (2007). "The Discontinuity Set of Solutions of the TV Denoising Problem and Some Extensions." *Multiscale Modeling & Simulation*, **6**(3), 879–894. doi:10.1137/070683003.

Chaudhuri P, Marron JS (2000). "Scale Space View of Curve Estimation." *The Annals of Statistics*, **28**(2), 408–428. doi:10.1214/aos/1016218224.

Clayden J (2020). ***mmand**: Mathematical Morphology in Any Number of Dimensions*. R package version 1.6.1, URL https://CRAN.R-project.org/package=mmand.

Collins DL, Zijdenbos AP, Kollokian V, Sled JG, Kabani NJ, Holmes CJ, Evans AC (1998). "Design and Construction of a Realistic Digital Brain Phantom." *IEEE Transactions on Medical Imaging*, **17**(3), 463–468. doi:10.1109/42.712135.

Coupé P, Manjon JV, Robles M, Collins DL (2012). "Adaptive Multiresolution Non-Local Means Filter for Three-Dimensional Magnetic Resonance Image Denoising." *IET Image Processing*, **6**(5), 558–568. doi:10.1049/iet-ipr.2011.0161.

Coupé P, Yger P, Prima S, Hellier P, Kervrann C, Barillot C (2008). "An Optimized Blockwise Non-Local Means Denoising Filter for 3-D Magnetic Resonance Images." *IEEE Transactions on Medical Imaging*, **27**(4), 425–441. doi:10.1109/tmi.2007.906087.

De los Reyes JC, Schönlieb CB, Valkonen T (2017). "Bilevel Parameter Learning for Higher-Order Total Variation Regularisation Models." *Journal of Mathematical Imaging and Vision*, **57**(1), 1–25. doi:10.1007/s10851-016-0662-8.

Duits R, Fuehr H, Janssen B (2012). "Left Invariant Evolutions on Gabor Transforms." In L Florack, MC Van Lieshout, R Duits, L Davies, G Jongbloed (eds.), *Mathematical Methods for Signal and Image Analysis and Representation*, volume 41 of *Computational Imaging and Vision*, chapter 8, pp. 137–158. Springer-Verlag. doi:10.1007/978-1-4471-2353-8_8.

Fan J, Gijbels I (1996). *Local Polynomial Modelling and Its Applications*. Chapman & Hall.

Felsberg M (2012). "Adaptive Filtering Using Channel Representations." In L Florack, MC Van Lieshout, R Duits, L Davies, G Jongbloed (eds.), *Mathematical Methods for Signal and Image Analysis and Representation*, volume 41 of *Computational Imaging and Vision*, chapter 2, pp. 31–48. Springer-Verlag. doi:10.1007/978-1-4471-2353-8_2.

Felsberg M, Forssén PE, Scharr H (2006). "Channel Smoothing: Efficient Robust Smoothing of Low-Level Signal Features." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(2), 209–222. doi:10.1109/tpami.2006.29.

Florack L (2012). "Scale Space Representations Locally Adapted to the Geometry of Base and Target Manifold." In L Florack, MC Van Lieshout, R Duits, L Davies, G Jongbloed (eds.), *Mathematical Methods for Signal and Image Analysis and Representation*, volume 41 of *Computational Imaging and Vision*, chapter 9, pp. 159–172. Springer-Verlag. doi:10.1007/978-1-4471-2353-8_9.

Foi A, Katkovnik V, Egiazarian K (2007). "Pointwise Shape-Adaptive DCT for High-Quality Denoising and Deblocking of Grayscale and Color Images." *IEEE Transactions on Image Processing*, **16**(5), 1395–1411. doi:10.1109/tip.2007.891788.

Franken EM (2008). *Enhancement of Crossing Elongated Structure in Images*. Ph.D. thesis, Eindhoven University of Technology, Dept. of Biomedical Engineering. URL http://alexandria.tue.nl/extra2/200910002.pdf.

Hintermüller M, Rautenberg CN, Wu T, Langer A (2017). "Optimal Selection of the Regularization Function in a Weighted Total Variation Model. Part II: Algorithm, Its Analysis and Numerical Tests." *Journal of Mathematical Imaging and Vision*, **59**(3), 515–533. doi:10.1007/s10851-017-0736-2.

Jenkinson M, Beckmann CF, Behrens TEJ, Woolrich MW, Smith SM (2012). "**FSL**." *NeuroImage*, **62**(2), 782–790. doi:10.1016/j.neuroimage.2011.09.015.

Kwan RKS, Evans AC, Pike GB (1996). "An Extensible MRI Simulator for Post-Processing Evaluation." In *Visualization in Biomedical Computing*, volume 1131 of *Lecture Notes in Computer Science*, pp. 135–140. Springer-Verlag. doi:10.1007/bfb0046947.

Kwan RKS, Evans AC, Pike GB (1999). "MRI Simulation-Based Evaluation of Image-Processing and Classification Methods." *IEEE Transactions on Medical Imaging*, **18**(11), 1085–1097. doi:10.1109/42.816072.

Lebrun M, Buades A, Morel JM (2013). "A Nonlocal Bayesian Image Denoising Algorithm." *SIAM Journal on Imaging Sciences*, **6**(3), 1665–1688. doi:10.1137/120874989.

Lee JS (1980). "Digital Image Enhancement and Noise Filtering by Use of Local Statistics." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-2**(2), 165–168. doi:10.1109/tpami.1980.4766994.

Lepski OV, Mammen E, Spokoiny VG (1997). "Optimal Spatial Adaptation to Inhomogeneous Smoothness: An Approach Based on Kernel Estimates with Variable Bandwidth Selector." *The Annals of Statistics*, **25**(3), 929–947. doi:10.1214/aos/1069362731.

Lepski OV, Spokoiny VG (1997). "Optimal Pointwise Adaptive Methods in Nonparametric Estimation." *The Annals of Statistics*, **25**(6), 2512–2546. doi:10.1214/aos/1030741083.

Li Y, Gilmore JH, Wang J, Styner M, Lin W, Zhu H (2012). "TwinMARM: Two-Stage Multiscale Adaptive Regression Methods for Twin Neuroimaging Data." *IEEE Transactions on Medical Imaging*, **31**(5), 1100–1112. doi:10.1109/tmi.2012.2185830.

Manjón JV, Coupé P, Martí-Bonmatí L, Collins DL, Robles M (2009). "Adaptive Non-Local Means Denoising of MR Images with Spatially Varying Noise Levels." *Journal of Magnetic Resonance Imaging*, **31**(1), 192–203. doi:10.1002/jmri.22003.

Muschelli J, Gherman A, Fortin JP, Avants B, Whitcher B, Clayden JD, Caffo BS, Crainiceanu CM (2019). "Neuroconductor: An R Platform for Medical Imaging Analysis." *Biostatistics*, **20**(2), 218–239. doi:10.1093/biostatistics/kxx068.

Muschelli J, Sweeney E, Lindquist M, Crainiceanu C (2015). "**fslr**: Connecting the **FSL** Software with R." *The R Journal*, **7**(1), 163–175. doi:10.32614/rj-2015-013.

Nason G (2016). ***wavethresh**: Wavelets Statistics and Transforms*. R package version 4.6.8, URL https://CRAN.R-project.org/package=wavethresh.

Ochi S (2018). ***imagerExtra**: Extra Image Processing Library Based on **imager***. R package version 1.3.2, URL https://CRAN.R-project.org/package=imagerExtra.

Papafitsoros K, Bredies K (2015). "A Study of the One Dimensional Total Generalised Variation Regularisation Problem." *Inverse Problems & Imaging*, **9**(2), 511–550. doi:10.3934/ipi.2015.9.511.

Perona P, Malik J (1990). "Scale-Space and Edge Detection Using Anisotropic Diffusion." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(7), 629–639. doi:10.1109/34.56205.

Polzehl J (2020). ***aws**: Adaptive Weights Smoothing*. R package version 2.5, URL https://CRAN.R-project.org/package=aws.

Polzehl J, Papafitsoros K, Tabelow K (2018). "Patch-Wise Adaptive Weights Smoothing." *Technical Report 2520*, WIAS Berlin. doi:10.20347/wias.preprint.2520.

Polzehl J, Spokoiny V (2000). "Adaptive Weights Smoothing with Applications to Image Restoration." *Journal of the Royal Statistical Society B*, **62**(2), 335–354. doi:10.1111/1467-9868.00235.

Polzehl J, Spokoiny V (2003). "Image Denoising: Pointwise Adaptive Approach." *The Annals of Statistics*, **31**(1), 30–57. doi:10.1214/aos/1046294457.

Polzehl J, Spokoiny V (2006). "Propagation-Separation Approach for Local Likelihood Estimation." *Probability Theory and Related Fields*, **135**(3), 335–362. doi:10.1007/s00440-005-0464-1.

Polzehl J, Spokoiny V (2008). "Structural Adaptive Smoothing by Propagation-Separation Methods." In C Chen, W Härdle, A Unwin (eds.), *Handbook of Data Visualization*, Series Handbooks of Computational Statistics, pp. 471–492. Springer-Verlag. doi:10.1007/978-3-540-33037-0_19.

Polzehl J, Tabelow K (2007). "Adaptive Smoothing of Digital Images: The R Package **adimpro**." *Journal of Statistical Software*, **19**(1), 1–17. doi:10.18637/jss.v019.i01.

Polzehl J, Tabelow K (2012). "Structural Adaptive Smoothing: Principles and Applications in Imaging." In L Florack, MC Van Lieshout, R Duits, L Davies, G Jongbloed (eds.), *Mathematical Methods for Signal and Image Analysis and Representation*, volume 41 of *Computational Imaging and Vision*, chapter 4, pp. 65–81. Springer-Verlag. doi:10.1007/978-1-4471-2353-8_4.

Polzehl J, Voss HU, Tabelow K (2010). "Structural Adaptive Segmentation for Statistical Parametric Mapping." *NeuroImage*, **52**(2), 515–523. doi:10.1016/j.neuroimage.2010.04.241.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Ring W (2000). "Structural Properties of Solutions to Total Variation Regularization Problems." *ESAIM: Mathematical Modelling and Numerical Analysis*, **34**(4), 799–810. doi:10.1051/m2an:2000104.

Rudin LI, Osher S, Fatemi E (1992). "Nonlinear Total Variation Based Noise Removal Algorithms." *Physica D*, **60**(1–4), 259–268. doi:10.1016/0167-2789(92)90242-f.

Scharr H, Krajsek K (2012). "A Short Introduction to Diffusion-Like Methods." In L Florack, MC Van Lieshout, R Duits, L Davies, G Jongbloed (eds.), *Mathematical Methods for Signal and Image Analysis and Representation*, volume 41 of *Computational Imaging and Vision*, chapter 1, pp. 1–30. Springer-Verlag. doi:10.1007/978-1-4471-2353-8_1.

Simonoff J (1996). *Smoothing Methods in Statistics*. Springer-Verlag. doi:10.1007/978-1-4612-4026-6.

Spokoiny V (1998). "Estimation of a Function with Discontinuities via Local Polynomial Fit with an Adaptive Window Choice." *The Annals of Statistics*, **26**(4), 1356–1378. doi:10.1214/aos/1024691246.

Tabelow K, Polzehl J (2019). ***adimpro**: Adaptive Smoothing of Digital Images*. WIAS Berlin. R package version 0.9.2, URL https://CRAN.R-project.org/package=adimpro.

Tabelow K, Polzehl J, Spokoiny V, Voss HU (2008). "Diffusion Tensor Imaging: Structural Adaptive Smoothing." *NeuroImage*, **39**(4), 1763–1773. doi:10.1016/j.neuroimage.2007.10.024.

Tabelow K, Polzehl J, Voss HU, Spokoiny V (2006). "Analyzing fMRI Experiments with Structural Adaptive Smoothing Procedures." *NeuroImage*, **33**(1), 55–62. doi:10.1016/j.neuroimage.2006.06.029.

The MathWorks Inc (2019). *MATLAB – The Language of Technical Computing, Version R2019a*. Natick. URL http://www.mathworks.com/products/matlab/.

Tomasi C, Manduchi R (1998). "Bilateral Filtering for Gray and Color Images." In *Proceedings of the 1998 IEEE International Conference on Computer Vision*, pp. 839–846. doi:10.1109/iccv.1998.710815.

Wand M (2020). ***KernSmooth**: Functions for Kernel Smoothing Supporting Wand & Jones (1995)*. R package version 2.23-17, URL https://CRAN.R-project.org/package=KernSmooth.

Wand MP, Jones MC (1995). *Kernel Smoothing*. Chapman and Hall.

Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004). "Image Quality Assessment: From Error Visibility to Structural Similarity." *IEEE Transactions on Image Processing*, **13**(4), 600–612. doi:10.1109/tip.2003.819861.

Weickert J (1998). *Anisotropic Diffusion in Image Processing*. ECMI. Teubner-Verlag. URL https://www.mia.uni-saarland.de/weickert/Papers/book.pdf.

Whitcher B (2020). ***waveslim**: Basic Wavelet Routines for One-, Two- And Three-Dimensional Signal Processing*. R package version 1.8.2, URL https://CRAN.R-project.org/package=waveslim.

Whitcher B, Schmid V, Thornton A, Muschelli J (2020). ***oro.nifti**: Rigorous – NIfTI + ANALYZE + AFNI: Input / Output*. R package version 0.10.3, URL https://CRAN.R-project.org/package=oro.nifti.

Whitcher B, Schmid VJ, Thornton A (2011). "Working with the DICOM and NIfTI Data Standards in R." *Journal of Statistical Software*, **44**(6), 1–28. doi:10.18637/jss.v044.i06.

You K (2019). ***tvR**: Total Variation Regularization*. R package version 0.3.1, URL https://CRAN.R-project.org/package=tvR.

Affiliation:

Jörg Polzehl, Kostas Papafitsoros, Karsten Tabelow
Weierstrass Institute
Mohrenstr. 39
10117 Berlin, Germany
E-mail: joerg.polzehl@wias-berlin.de,
kostas.papafitsoros@wias-berlin.de,
karsten.tabelow@wias-berlin.de
URL: https://www.wias-berlin.de/people/polzehl/
https://www.wias-berlin.de/people/papafitsoros/
https://www.wias-berlin.de/people/tabelow/

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
October 2020, Volume 95, Issue 6 *Submitted:* 2018-07-30
doi:10.18637/jss.v095.i06 *Accepted:* 2019-07-25