TOTAL VARIATION REGULARIZATION FOR MANIFOLD-VALUED DATA

ANDREAS WEINMANN† LAURENT DEMARET† AND MARTIN STORATH‡

Abstract. We consider total variation minimization for manifold valued data. We propose a cyclic proximal point algorithm and a parallel proximal point algorithm to minimize TV functionals with ℓ^p-type data terms in the manifold case. These algorithms are based on iterative geodesic averaging which makes them easily applicable to a large class of data manifolds. As an application, we consider denoising images which take their values in a manifold. We apply our algorithms to diffusion tensor images, interferometric SAR images as well as sphere and cylinder valued images. For the class of Cartan-Hadamard manifolds (which includes the data space in diffusion tensor imaging) we show the convergence of the proposed TV minimizing algorithms to a global minimizer.

1. Introduction. Data taking values in a manifold appear naturally in various signal and image processing applications. One example is diffusion tensor imaging where the data live in the Riemannian manifold of positive (definite) matrices; see, e.g., [6, 55]. Other examples are color images based on non-flat color models [16, 69, 46]. Here the data has circle- or sphere-valued components. Data with values on the circle also appear in the context of interferometric SAR images [48]. $SO(3)$ and motion group-valued data were considered in [68].

Processing manifold valued data has gained a lot of interest in recent years. To mention only some examples, wavelet-type multiscale transforms for manifold data have been considered in [68, 39, 71]. Manifold valued partial differential equations are the subject of study in the papers [66, 18, 38]. Furthermore, statistical issues on Riemannian manifolds are topic of [30, 31, 32].

The present paper deals with total variation (TV) minimization for data taking values in a manifold. Our main application is the denoising of such manifold-valued data. For scalar data, TV minimization was shown to be a powerful tool for edge-preserving denoising [59]. For images, the anisotropic version of TV minimization is given by

$$\operatorname*{argmin}_{x\in M^{n\times m}} \Big\{ \frac{1}{p} \sum_{i,j=1}^{n,m} d(x_{ij}, f_{ij})^p + \alpha \sum_{i,j=1}^{n-1,m} d(x_{ij}, x_{i+1,j}) + \alpha \sum_{i,j=1}^{n,m-1} d(x_{ij}, x_{i,j+1}) \Big\}. \quad (1.1)$$

For scalar data, the symbol $d(y,z) = |y-z|$ simply denotes the euclidean distance between y and z. The distance to data f is measured in the ℓ^p norm. For Gaussian noise, $p=2$ is reasonable, whereas, for noise types with heavier tails such as Laplacian noise, $p=1$ is more natural. The natural generalization of the total variation problem for data on a manifold M is given by using the distance d induced by the Riemannian metric on the manifold instead of the euclidean distance in (1.1).

We introduce algorithms to minimize the ℓ^p-TV functional for manifold data and show convergence towards a (global) minimizer for the class of Cartan-Hadamard manifolds where the data space in diffusion tensor imaging is a particular member of. Our experiments show the denoising capability of ℓ^p-TV minimization in the manifold context.

†Helmholtz Center Munich, and Department of Mathematics, Technische Universität München (Germany), andreas.weinmann@tum.de, laurent.demaret@helmholtz-muenchen.de

‡Biomedical Imaging Group, École Polytechnique Fédérale de Lausanne (Switzerland), martin.storath@epfl.ch

1.1. Total variation regularization for scalar and vector-valued data. Total variation regularization was first introduced in the early 1990ies by Rudin, Osher and Fatemi [59]. A central advantage of total variation regularization compared with classical Tykhonov regularization is that it preserves sharp edges [63, 36]. Especially due to this property TV minimization has been used in a vast amount of applications. Examples are biomedical imaging [24], geophysics [2] and computer vision [74, 20], to mention only a few.

Theoretical properties of total variation regularization have been investigated in a series of papers. For instance, results on existence and uniqueness of minimizers have been proved in [13]. Connections to wavelet shrinkage are shown in [56] and equivalences between diffusion techniques, thresholding strategies and TV minimization be found in [61].

A lot of different algorithms for TV minimization of scalar- and vector-valued images have been proposed in the last 20 years. In their original work, Rudin, Osher, Fatemi [59] consider ℓ^2 data terms. They use gradient descent on the Euler-Lagrange equations of the (scalar-valued) total variation functional. Further methods are based on Fenchel duals [12], the alternating direction method of multipliers [73], and split Bregman methods [35].

Several authors have studied the total variation problem with ℓ^1 data terms [1, 51, 15]. Approaches based on the ℓ^1-TV functional enjoy the edge preserving properties of TV regularizers while, in addition, being more robust to outliers. Various solution strategies have been proposed for the ℓ^1-TV problem. To mention some examples, schemes based on smooth approximations are presented in [52, 53]; semi-smooth Newton method based approaches are the topic of [21]; primal-dual methods are proposed in [14, 26].

1.2. Algorithms for TV minimization for manifold-valued data. In this paper we derive algorithms for the TV minimization problem (1.1) on manifold-valued data. Our algorithms are based on iterative geodesic averaging. More precisely, we decompose the TV functional in (1.1) into a sum of functionals in such a way that we are able to explicitly compute the proximal mappings of these functionals on the manifold. We obtain that these proximal mappings are given in terms of points on certain geodesics. So, in order to make the algorithms work on a concrete manifold, the only operations we need are those needed for calculating geodesics. The spaces which frequently occur as data spaces are matrix groups or related symmetric spaces. So usually, there are explicit formulas available for this task.

Our algorithms are iterative schemes. In each iteration, we apply the above proximal mappings of the functionals decomposing the TV functional. The first algorithm is a *cyclic proximal point algorithm*. This means that we successively apply the proximal mappings using the output related to the ith summand as a new input for the proximal mapping of the $(i+1)$th summand. The second algorithm is a *parallel proximal point algorithm*. Here the proximal mappings are calculated for the same initial point and averaged afterwards. Since computing mean values on a manifold is a relatively expensive iterative procedure, we also consider a variant which only does approximative averaging (but yields comparable results). We call this variant *fast parallel proximal point algorithm*. Due to the averaging procedure, the parallel algorithms need more operations in total. However, they have higher potential for parallelization.

Our algorithms belong to the class of proximal splitting methods (for manifold valued data). A survey on proximal splitting methods (for scalar data) is [22]. In

section 7, the paper [22] also describes a related parallel algorithm. Parallel proximal point algorithms were also considered in [9]. Cyclical proximal point algorithms have been studied in [8] (for linear spaces) and in [4] where they are applied for the computation of means and medians in Hadamard spaces.

Principally, our algorithms work for all ℓ^p data terms with $p \geq 1$ as well as for regularizing terms based on qth variation, $q \geq 1$, instead of total variation. This in particular includes the classical Tikhonov regularization which corresponds to $p = q = 2$. For $p, q = 1, 2$ we give nice closed form expressions; in the other cases, one needs the numerical solution of a certain nonlinear equation. We furthermore consider Huber data terms as well as Huber regularizing terms (which are sometimes called "Huber-ROF"). The latter are employed to avoid unwanted staircasing effects; see [14].

For the class of Cartan-Hadamard manifolds, we obtain the convergence of our geodesic averaging based schemes towards a (global) minimizer of (1.1). Cartan-Hadamard manifolds are Riemannian manifolds containing many symmetric spaces such as the data space in diffusion tensor imaging. Our convergence statements also hold true for the more general class of Hadamard spaces and for regularization based on qth variation as well as Huber data and regularization terms.

1.3. Applications. We demonstrate the denoising capabilities of our algorithms on various data spaces. It is the common observation of all experiments that TV minimization reliably removes noise from manifold-valued data while preserving edges.

First, we consider diffusion tensor images. Diffusion tensor imaging (DTI) is a technique to quantify non-invasively the diffusional characteristics of a specimen [6, 41]. Here the underlying data space is the space of positive matrices. According to the model, the diffusivity in direction v is determined by $v^T A v$ where A is a positive matrix representing data. The space of positive matrices becomes a Cartan-Hadamard manifold when equipped with a suitable Riemannian metric. This means that our algorithm provably converge to a minimizer in the DTI setup. We demonstrate the denoising performance with a real diffusion tensor image of a human brain and with synthetic data; see the Figures 1 and 2.

Next, we consider interferometric synthetic aperture radar (InSAR) data. InSAR is an important airborne imaging modality for geodesy [48]. In our concrete example, the InSAR image has the interpretation of a wrapped periodic version of a digital elevation model [58]. Hence the underlying data space is the sphere S^1. From the experiment in Figure 3 we see that total variation minimization is capable of removing almost all the noise from the InSAR image. We further observe that ℓ^1 and Huber data terms are slightly more robust to outliers in the data than the ℓ^2 data term.

Our third application is image denoising in nonlinear color spaces. We consider the LCh space which consists of real-valued luminance and chromaticity components L and C as well as a S^1 valued hue component h. Thus the underlying manifold is the cylinder $\mathbb{R}^2 \times S^1$. We note that, although the underlying manifold $\mathbb{R}^2 \times \mathrm{S}^1$ is a product space, the algorithms cannot be applied separately to the components. In our experiment we obtain a better reconstruction quality by manifold-valued TV minimization in the LCh color space than by classical vectorial TV minimization in the standard RGB space; see Figure 4.

We continue with the sphere S^2. Data with values in S^2 appear in, for example, chromaticity-based image processing [16] and as orientation fields of three dimensional images [57]. We here consider a synthetic example on which we impose von Mises-Fisher noise. Figure 5 shows that, also for data with values in the sphere S^2, the noise is almost perfectly removed and that the edges are not smoothed out.

We conclude with the rotation group SO(3) as data spaces. Data with values in SO(3) appear, for example, in the context of aircraft orientations [68], protein alignments [37], and the tracking of 3D rotational data arising in robotics [27]. We apply our methods to a synthetic time series on which we imposed noise based on a matrix Fisher distribution. The results confirm their denoising capability and they also reveal that a Huber regularizing term is less affected by staircasing effects; cf. Figure 6.

1.4. Organization of the article. We start out by developing algorithms for TV minimization for manifold valued data in Section 2. Then we show the convergence of our algorithms towards (global) minimizers of the TV minimization problem (1.1) in Hadamard spaces in Section 3. In Section 4, we apply our algorithms to denoising data on concrete manifolds.

2. Algorithms for TV minimization for manifold-valued data. In the following we propose two algorithms for total variation minimization for data which take their values in a manifold. We consider ℓ^1 and ℓ^2 as well as Huber data terms. Our algorithms are based on iterative geodesic averaging. The appearing geodesic averages are the minimizers of certain proximal mappings which arise as follows: we split the TV functional into basic building blocks and consider the proximal mappings of these building blocks. The first algorithm performs the iteration of the proximal mappings in a cyclical way whereas the second does so in a parallel way.

2.1. Splitting of the TV functional and proximal mappings. Let us consider the problem of (bivariate) ℓ^p-TV^q minimization

$$\frac{1}{p}\sum_{i,j} d^p(x_{ij}, f_{ij}) + \alpha\frac{1}{q}\sum_{i,j} d^q(x_{ij}, x_{i+1,j}) + \alpha\frac{1}{q}\sum_{i,j} d^q(x_{ij}, x_{i,j+1}) \to \min. \qquad (2.1)$$

The data f_{ij} as well as the arguments x_{ij} to minimize take their values in a Riemannian manifold M. The symbol d^p denotes the pth power of the distance induced by the Riemannian metric. For a bounded (complete) Riemannian manifold, the functional obviously has a minimizer since continuous functions have minima on compact sets. For the unbounded case, we notice that going too far away from the data f leads to high functional values. Hence the set of minimizer candidates is actually confined to a bounded set which brings us back to the already discussed situation and minimizers exist.

Setting $q = 1$ in (2.1), we get the discrete (anisotropic) TV functional with ℓ^p data term. In particular, if $p = 1$, we are in the ℓ^1-TV setting. The case $q = 2$ corresponds to the classical Tikhonov regularization term in the scalar case. We comment on Huber data and regularizing terms in Section 2.5.

Our approaches towards the minimization of (2.1) are based on rewriting (2.1) as a sum of simpler functions. We consider the function $F : M \times \ldots \times M \to \mathbb{R}$ which is the data term given by

$$F(x) = \frac{1}{p}\sum_{i,j=1}^{n,m} d^p(x_{ij}, f_{ij}), \qquad (2.2)$$

as well as, for i, j, the functions $G_{ij}, H_{ij} : M \times \ldots \times M \to \mathbb{R}$ given by

$$\begin{aligned} G_{ij}(x) &= \frac{1}{q} d^q(x_{ij}, x_{i,j+1}), \\ H_{ij}(x) &= \frac{1}{q} d^q(x_{ij}, x_{i+1,j}). \end{aligned} \tag{2.3}$$

Using this notation, the minimization problem (2.1) has the form

$$F(x) + \alpha \sum_{i,j} G_{ij}(x) + \alpha \sum_{i,j} H_{ij}(x) \to \min. \tag{2.4}$$

For each summand in (2.4), we consider its proximal mapping [50, 28, 3]. The proximal mappings of the G_{ij} are defined by the minimization problem

$$\operatorname{prox}_{\lambda G_{ij}} x = \operatorname{argmin}_{y \in M^{n \times m}} \left(\lambda G_{ij}(y) + \frac{1}{2} d^2(x, y) \right), \tag{2.5}$$

where the parameter $\lambda > 0$ and the distance d on the product manifold $M^{n \times m}$ is given by $d^2(x, y) = \sum_{i,j=1}^{n,m} d(x_{ij}, y_{ij})^2$. The proximal mappings of F and the H_{ij} are defined analogously. The crucial point is that, using the splitting (2.4), the proximal mappings of all appearing summands can be explicitly computed as geodesic averages. More precisely, solving the minimization problem of (2.5) reduces to computing points on shortest geodesics joining given points. The same is true for the analogous problems for F and the H_{ij}.

We give a heuristic introductory derivation of these facts now. A mathematically precise statement is formulated as Proposition 1 later on. We assume for the time being (without mentioning) that the points in the Riemannian manifold are sufficiently near to each other such that the following arguments apply. Let us consider the proximal mapping of G_{ij} given by (2.5). A necessary condition for $x' \in M^{n \times m}$ to be a minimizer in (2.5) is that 0 is in the (sub)gradient of $\lambda G_{ij}(x') + \frac{1}{2} d^2(x, x')$. This immediately implies that, for the $(k, l)^{th}$ component of the proximal mapping $\operatorname{prox}_{\lambda G_{ij}} x$, we have

$$(\operatorname{prox}_{\lambda G_{ij} x})_{kl} = x_{kl} \quad \text{for} \quad k \neq i, i+1 \text{ and } l \neq j. \tag{2.6}$$

For $k = i, i+1$ and $l = j$ we use that the gradient of the mapping $M \to \mathbb{R}$, $z \mapsto d^p(z, v)/p$ fulfills (see, e.g., [65, Eq. (2.8)])

$$\nabla_z d^p(z, v)/p = \frac{\exp_z^{-1}(v)}{d^{2-p}(z, v)}.$$

Hence, we get as necessary conditions for a minimizer x' in (2.5) that

$$\begin{aligned} \lambda \tfrac{1}{d^{2-p}(x'_{ij}, x'_{i,j+1})} \exp_{x'_{ij}}^{-1}(x'_{i,j+1}) + \exp_{x'_{ij}}^{-1} x_{ij} = 0, \\ \lambda \tfrac{1}{d^{2-p}(x'_{ij}, x'_{i,j+1})} \exp_{x'_{i,j+1}}^{-1}(x'_{ij}) + \exp_{x'_{i,j+1}}^{-1} x_{i,j+1} = 0. \end{aligned}$$

Here $\exp_z^{-1}$ denotes the inverse of the Riemannian exponential mapping at the point z. Thus, both summands in the first condition are tangent vectors at x'_{ij}. They point in opposite directions, and so, the first condition implies that the three points $x'_{i,j}$, $x'_{i,j+1}$, and $x_{i,j}$ lie on a common geodesic in M. Analogously, the second condition

implies that the three points $x'_{i,j}$, $x'_{i,j+1}$, and $x_{i,j+1}$ also lie on a common geodesic. Hence, the four points must lie on one geodesic. In particular, the points $x'_{i,j}$, $x'_{i,j+1}$ – which are the $(i,j)^{th}$ and $(i,j+1)^{th}$ component of the proximal mapping of G_{ij} applied to x – lie on the geodesic joining $x_{i,j}$ and $x_{i,j+1}$.

Then, after some technical considerations (cf. the proof of Theorem 2), the location of the points $x'_{i,j} = (\mathrm{prox}_{\lambda G_{ij}} x)_{i,j}$ and $x'_{i,j+1} = (\mathrm{prox}_{\lambda G_{ij}} x)_{i,j+1}$ are explicitly given as follows. We have

$$\begin{aligned} (\mathrm{prox}_{\lambda G_{ij}} x)_{ij} &= [x_{ij}, x_{i,j+1}]_t, \\ (\mathrm{prox}_{\lambda G_{ij}} x)_{i,j+1} &= [x_{i,j+1}, x_{ij}]_t, \end{aligned} \tag{2.7}$$

where the symbol $[\cdot,\cdot]_t$ denotes the point reached after time t on the unit speed geodesic starting at the first argument in direction of the second argument. In the TV case ($q = 1$),

$$t = \begin{cases} \lambda, & \text{if } \lambda < \frac{1}{2} d(x_{ij}, x_{i,j+1}), \\ d(x_{ij}, x_{i,j+1})/2, & \text{else.} \end{cases} \tag{2.8}$$

For $q = 2$, which corresponds to quadratic variation, we get

$$t = \frac{\lambda}{1 + 2\lambda} d(x_{ij}, x_{i,j+1}). \tag{2.9}$$

The proximal mappings of the H_{ij} are obtained in a completely analogous way. It remains to find the proximal mapping of F which means finding the proximal mapping of the distance function in M. This is well known and can be found, e.g., in [28]. They can again be written as geodesic averages and are explicitly given by

$$(\mathrm{prox}_{\lambda F})_{ij}(x) = [x_{ij}, f_{ij}]_t, \tag{2.10}$$

where, for the ℓ^2 data term,

$$t = \tfrac{\lambda}{1+\lambda} d(x_{ij}, f_{ij}). \tag{2.11}$$

For the ℓ^1 data term,

$$t = \begin{cases} \lambda, & \text{if } \lambda < d(x_{ij}, f_{ij}), \\ d(x_{ij}, f_{ij}), & \text{else.} \end{cases} \tag{2.12}$$

This corresponds to the equivalent of *soft thresholding* in the context of manifolds.

A mathematically precise formulation of the results of the above heuristic derivation is as follows; its proof is provided in Section 3 below the proof of Theorem 2.

PROPOSITION 1. *Let M be a complete connected Riemannian manifold. Then a solution y^* of the proximal mapping related minimization problem in* (2.5) *is given as follows. For $k \neq i, i+1$ and $l \neq j$, define $y^*_{k,l} = x_{k,l}$; the component $y^*_{i,j}$ is chosen as the point on a shortest geodesic between x_{ij} and $x_{i,j+1}$ starting from x_{ij} reached after time t with t given by* (2.8),(2.9)*; the component $y^*_{i,j}$ is obtained analogously, but starting at $x_{i,j+1}$ on the same geodesic with reverse direction. Analogous statements hold for the H_{ij} and F.*

In particular, if there is only one shortest geodesic (which is always the case for nearby points), then the proximal mappings of F, G_{ij}, H_{ij} are well defined and they are given by (2.10),(2.6),(2.7) *and their analogues for H_{ij}.*

2.2. A cyclic proximal point algorithm for TV minimization for manifold-valued data. The first algorithm we propose for TV minimization for manifold-valued data is a cyclic proximal point algorithm based on geodesic averaging. For vector space data, cyclical proximal point algorithms were considered in [8]. For Hadamard spaces, they were investigated by M. Bačák [4] who applied them to the computation of means and medians.

We now derive a cyclic proximal point algorithm for the minimization of the ℓ^p-TV^q functional (2.1). We consider the problem in the form $F(x)+ \alpha \sum_{i,j} G_{ij}(x) + \alpha \sum_{i,j} H_{ij}(x)$ given by (2.4). We first apply the proximal mapping of F which is given as pointwise geodesic averages of data f_{ij} and the argument of the functional x_{ij}; see (2.10). Then we successively apply the proximal mappings of all the G_{ij}. They are given by (2.6) and (2.7) which is again based on geodesic averaging. As a last step, the analogous operations are executed for the $H_{ij.}$

Iteration of all these steps yields the algorithm which is stated as Algorithm 1. During the iteration, the parameter λ_n of the proximal mappings is successively decreased. In this way, the penalty for deviation from the previous iterate is successively increased. It is chosen in a way such that the sequence λ_n is square-summable but not summable. This is moderate enough not to prevent convergence towards a minimizer; cf. Theorem 2.

Algorithm 1: Cyclic proximal point algorithm for ℓ^p-TV^q for manifold data.

Input: Manifold-valued image $f \in M^{n\times m}$, regularization parameter $\alpha > 0$, parameter sequence for the proximal mappings $\lambda = (\lambda_1, \ldots) \in \ell^2 \setminus \ell^1$.
Output: Minimizer x of the ℓ^p-TV^q problem (2.1).
begin
  $x \leftarrow f$;
  for $r \leftarrow 1, 2, \ldots$ **do**
    for *i ← 1,...,n; j ← 1,...,m* **do**
      $t \leftarrow \text{calc\_t}_F(\lambda_r, p, q, x_{ij}, f_{ij})$; /* Calculate t, see Table 2.2. */
      $x_{ij} \leftarrow [x_{ij}, f_{ij}]_t$; /* Proximal mapping of F. */
    end
    for *i ← 1,...,n; j ← 1,...,m-1* **do**
      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{ij}, x_{i,j+1})$; /* Calculate t, see Table 2.1. */
      $x'_{ij} \leftarrow [x_{ij}, x_{i,j+1}]_t$; /* Proximal mapping of G_{ij}. */
      $x'_{i,j+1} \leftarrow [x_{i,j+1}, x_{ij}]_t$;
      $x_{ij} \leftarrow x'_{ij}$; $x_{i,j+1} \leftarrow x'_{i,j+1}$;
    end
    for *i ← 1,...,n-1; j ← 1,...,m* **do**
      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{ij}, x_{i,j+1})$; /* Calculate t, see Table 2.1. */
      $x'_{ij} \leftarrow [x_{ij}, x_{i+1,j}]_t$; /* Proximal mapping of H_{ij}. */
      $x'_{i+1,j} \leftarrow [x_{i+1,j}, x_{ij}]_t$;
      $x_{ij} \leftarrow x'_{ij}$; $x_{i+1,j} \leftarrow x'_{i+1,j}$;
    end
  end
end

2.3. A parallel proximal point algorithm for TV minimization for manifold-valued data. Parallel proximal point algorithms were, for example, considered in [9, 22, 60]. We here apply a related parallel proximal point algorithm to TV minimization for manifold-valued data. A great advantage of this approach is its

immediate parallelizability.

As for the cyclic algorithm, we split the TV functional into a sum of simpler functionals. But instead of applying the proximal mappings successively , we apply all the proximal mappings to the same initial data. Then the results of the different proximal mappings are averaged.

In order to split the TV functional, we consider the mappings

$$\begin{aligned} G_e &= \sum_{j:j \text{ even}} \sum_i \frac{1}{q} d^q(x_{ij}, x_{i,j+1}), \\ G_o &= \sum_{j:j \text{ odd}} \sum_i \frac{1}{q} d^q(x_{ij}, x_{i,j+1}). \end{aligned} \tag{2.13}$$

The mapping H_e, H_o are defined analogously, exchanging the role of i and j. Then we have that the ℓ^p-TV^q functional can be decomposed into $F + G_e + G_o + H_e + H_o$. Since $G_e = \sum_{j:j \text{ even}} \sum_i G_{ij}$, the proximal mapping of G_e is explicitly given by

$$(\text{prox}_{\lambda G_e} x)_{i,j} = \begin{cases} [x_{ij}, x_{i,j+1}]_{t_1}, & j \text{ even}, \\ [x_{ij}, x_{i,j-1}]_{t_2}, & j \text{ odd}. \end{cases} \tag{2.14}$$

Here t_1 and t_2 are defined by (2.8),(2.9) (cf. the derivation of (2.7)). The proximal mapping of G_o is obtained by exchanging the terms "even" and "odd" in the above formula. For H_e, H_o, one exchanges the roles of i and j.

Equipped with these explicit formulas for the proximal mappings, the next step is to average the results of the application of the proximal mappings. Since our data live in a Riemannian manifold, the usual arithmetic mean in a vector space is not available. However, it is well known (cf. [43, 44, 55, 31]) that

$$z^* = \text{argmin}_{z \in M} \sum_{i=1}^{N} d(z, z_i)^2 \tag{2.15}$$

is the appropriate definition of the mean $z^* = \text{mean}(z_1, \ldots, z_N)$ of the N elements z_i on the manifold M. The mean is in general not globally defined since the minimization problem has no unique solution in general. For the z_i being in a small ball, however, it is unique. The size of the ball depends on the sectional curvature of the manifold M. Details and further information can, e.g., be found in [44, 43].

In order to get the mean in the product manifold $M^{n \times m}$ we just have to compute the component-wise means. Applied to the above proximal mapping, we get one iteration of the parallel algorithm at pixel i, j by

$$\begin{aligned} &x_{ij}^{(k+1)} = \\ &\text{mean}([x_{i,j}^{(k)}, x_{i,j+1}^{(k)}]_{t_1}, [x_{i,j}^{(k)}, x_{i,j-1}^{(k)}]_{t_2}, [x_{i,j}^{(k)}, x_{i+1,j}^{(k)}]_{t_3}, [x_{i,j}^{(k)}, x_{i-1,j}^{(k)}]_{t_4}, [x_{i,j}^{(k)}, f_{i,j}]_{t_5}) \end{aligned} \tag{2.16}$$

where the t_i are computed according to (2.11), (2.12) and (2.8), (2.9), respectively. So the iterate at pixel (i, j) is obtained by the mean of geodesic averages of the old iterate in a neighborhood. The whole algorithm is summed up as Algorithm 2.

In contrast to the euclidean case there is no closed form expression of the intrinsic mean defined by (2.15) in Riemannian manifolds. The methods used for computing the mean are of iterative nature and are thus more time consuming. Perhaps the most well known method for computing the intrinsic mean, is the gradient descent already

Algorithm 2: Parallel proximal point algorithm for ℓ^p-TV^q for manifold data.

Input: Manifold-valued image $f \in M^{n\times m}$, regularization parameter $\alpha > 0$, parameter sequence for the proximal mappings $\lambda = (\lambda_1, \ldots) \in \ell^2 \setminus \ell^1$.

Output: Minimizer x of the ℓ^p-TV^q problem (2.1).

begin

  $x \leftarrow f$;

  for $r \leftarrow 1, 2, \ldots$ **do**

    for $i \leftarrow 1, \ldots, n; j \leftarrow 1, \ldots, m$ **do**

      $t \leftarrow \text{calc\_t}_F(\lambda_r, p, q, x_{i,j}, f_{i,j})$; /* Calculate t, see Table 2.2. */

      $z^{(1)} \leftarrow [x_{ij}, f_{ij}]_t$; /* Proximal mapping of F. */

      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{i,j}, x_{i,j+1})$; /* Calculate t, see Table 2.1. */

      $z^{(2)} \leftarrow [x_{ij}, x_{i,j+1}]_t$; /* Proximal mapping of G_e/G_o. */

      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{i,j}, x_{i,j-1})$; /* Calculate t by (2.8),(2.9) */

      $z^{(3)} \leftarrow [x_{ij}, x_{i,j-1}]_t$; /* Proximal mapping of G_o/G_e. */

      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{i,j}, x_{i+1,j})$; /* Calculate t, see Table 2.1. */

      $z^{(4)} \leftarrow [x_{ij}, x_{i+1,j}]_t$; /* Proximal mapping of H_o/H_e. */

      $t \leftarrow \text{calc\_t}_{GH}(\lambda_r\alpha, p, q, x_{i,j}, x_{i-1,j})$; /* Calculate t, see Table 2.1. */

      $z^{(5)} \leftarrow [x_{ij}, x_{i-1,j}]_t$; /* Proximal mapping of H_o/H_e. */

      $x_{ij} \leftarrow \text{mean}(z^{(1)}, z^{(2)}, z^{(3)}, z^{(4)}, z^{(5)})$; /* Intrinsic mean. */

      Alternative: $x_{ij} \leftarrow \text{approx\_mean}(z^{(1)}, z^{(2)}, z^{(3)}, z^{(4)}, z^{(5)})$; /* Fast approximative variant (cf. (2.21)). */

    end

  end

end

mentioned in Karcher's seminal paper [43]; cf. also [31], for example. The iteration for computing the intrinsic mean of the points $x_1, \ldots, x_N$ is given by

$$x^{(k+1)} = \exp_{x^{(k)}} \sum_{i=1}^{N} \tfrac{1}{N} \exp^{-1}_{x^{(k)}} x_i. \tag{2.17}$$

Also approaches based on Newton's method can be found in the literature; see, e.g., [29].

However, it is reported in the literature and also confirmed by the authors' experience that the gradient descent converges rather fast; in most cases, 5-10 iterations are enough for five points. This might explain why this simple method of gradient descent is widely used.

2.4. Speed-up of the parallel proximal point algorithm. In Algorithm 2, we calculate the intrinsic mean of the five points $z^{(i)}$ in the inner loop. Each step of the gradient descent (2.17) in the computation of the mean of the $z^{(i)}$ takes about half of the time needed for computing the points $z^{(i)}$ by geodesic averaging. Since we typically need at least five iterations for the gradient descent, computing the means is the most time consuming part of Algorithm 2.

In order to reduce the computation time, we propose to replace the mean by another construction (known as geodesic analogues in the subdivision context [70]) which is computationally less demanding. This construction approximates the mean, the results are comparable (cf. Figure 2), and we can also show convergence towards a minimizer (cf. Theorem 7).

Regularizer	**Geodesic path length** (value of calc_t$_{GH}(\lambda, p, q, y, z)$)
TV	$\min(\lambda, d(y,z)/2)$
TV^2	$\frac{\lambda}{1+2\lambda} d(y,z)$
Huber	$\begin{cases} \frac{2\lambda\tau^2}{1+4\lambda\tau^2} d(y,z), & \text{if } d(y,z) < \frac{\omega(1+4\lambda\tau^2)}{\sqrt{2}\tau}, \\ \min\left(d(y,z)/2, \sqrt{2}\lambda\omega\tau\right), & \text{otherwise.} \end{cases}$

Table 2.1: Geodesic path length for proximal point problems associated to G and H for the regularization terms considered in this article.

In order to explain the construction, we rewrite the euclidean mean x of n points $x_1, \ldots, x_n$ as iterative convex combinations of only two points (in a binary tree like fashion):

$$x = \sum \tfrac{1}{n} x_i = \mathrm{conv}_{t_1}(\mathrm{conv}_{t_2}(\ldots, \ldots \mathrm{conv}_{t_l}(x_{i_l}, x_{j_l}) \ldots), \mathrm{conv}_{t_3}(\ldots, \ldots)). \tag{2.18}$$

Here, we use the notation $\mathrm{conv}(y,z)_t$ for the convex combination $(1-t)y+tz$ of points y, z. For example, for $n = 5$ points, we have the following representation

$$x = \mathrm{conv}_{0.2}(\mathrm{conv}_{0.5}(\mathrm{conv}_{0.5}(x_1, x_2), \mathrm{conv}_{0.5}(x_3, x_4)), x_5). \tag{2.19}$$

We note that the above representation is not unique. The idea is now to replace each euclidean convex combination $\mathrm{conv}_t(x,y)$ in (2.18) by the corresponding Riemannian one, i.e., the point $[x,y]_{td(x,y)}$ on the geodesic joining x and y. Then, (2.18) reads

$$x = [[\ldots, \ldots [x_{i_l}, x_{j_l}]_{t'_l} \ldots]_{t'_2}, [\ldots, \ldots]_{t'_3}]_{t'_1}, \tag{2.20}$$

where $t'_k = t_k d_k$ and d_k denotes the distance of the elements in the bracket the d_k is attached to. (This technicality arises since we consider unit speed geodesics.) We consider the above decomposition (2.19) and transport it to the Riemannian setting (2.20). Then, instead of using the mean in Algorithm 2 we propose to use the alternative procedure (called "approx_mean" in Algorithm 2) given by

$$x = [[[z^{(1)}, z^{(2)}]_{0.5d_1}, [z^{(3)}, z^{(4)}]_{0.5d_2}]_{0.5d_3}, z^{(5)}]_{0.2d_4}. \tag{2.21}$$

Here each d_k again denotes the distance of the elements in the bracket the d_i is attached to. The points $z^{(i)}$ are the results of the application of the proximal mappings in Algorithm 2.

The full algorithm is given by Algorithm 2 using the part referred to as "Alternative".

2.5. Huber regularizing and data terms. Total variation regularized images may suffer from the undesirable creation of steps in the solution. This is often called staircasing effect. An effective way to decrease staircasing is to replace the total variation by the *Huber regularizer* sometimes called *Huber-ROF model* [14]. To this end, we replace d^q in the definition of the TV^q functional (2.1) by $h \circ d$ which is the concatenation of the distance on the manifold and the *Huber function* h. The Huber function h is defined, for $s > 0$, by

$$h(s) = \begin{cases} \tau^2 s^2, & \text{for } s < \omega/(\sqrt{2}\tau), \\ \omega\sqrt{2}\tau s - \omega^2/2, & \text{otherwise,} \end{cases} \qquad \tau, \omega > 0. \tag{2.22}$$

Data term	**Geodesic path length** (value of calc_$\mathrm{t}_F(\lambda, p, q, y, z)$)
ℓ^1	$\min(\lambda, d(y,z))$
ℓ^2	$\frac{\lambda}{1+\lambda} d(y,z)$
Huber	$\begin{cases} \frac{2\lambda\tau^2}{1+2\lambda\tau^2} d(y,z), & \text{if } d(y,z) < \frac{\omega(1+2\lambda\tau^2)}{\sqrt{2}\tau}, \\ \min\left(d(y,z), \sqrt{2}\lambda\omega\tau\right), & \text{otherwise.} \end{cases}$

Table 2.2: Geodesic path lengths for the proximal point problem associated to F for the data terms considered in this article.

It is a square function (for small arguments) smoothly glued with an absolute value function (for large arguments). In analogy to (2.3) and (2.4) we write the Huber regularizer as $\sum_{i,j} G^h_{ij} + \sum_{i,j} H^h_{ij}$ with $G^h_{ij}(x) = h \circ d(x_{ij}, x_{i,j+1})$ and $H^h_{ij}(x) = h \circ d(x_{ij}, x_{i+1,j})$. As in (2.6) and (2.7), the proximal mappings of the G^h_{ij} are given by (cf. the proof of Theorem 2)

$$(\mathrm{prox}_{\lambda G^h_{ij}} x)_{kl} = \begin{cases} [x_{ij}, x_{i,j+1}]_t, & k = i \text{ and } l = j, \\ [x_{i,j+1}, x_{ij}]_t, & k = i \text{ and } l = j+1, \\ x_{kl}, & \text{otherwise,} \end{cases} \tag{2.23}$$

where

$$t = \begin{cases} \frac{2\lambda\tau^2}{1+4\lambda\tau^2} d(x_{i,j}, x_{i,j+1}), & \text{if } d(x_{ij}, x_{i,j+1}) < \frac{\omega(1+4\lambda\tau^2)}{\sqrt{2}\tau}, \\ \min\left(d(x_{ij}, x_{i,j+1})/2, \sqrt{2}\lambda\omega\tau\right), & \text{otherwise.} \end{cases} \tag{2.24}$$

So replacing the procedure to calculate the geodesic length t in Algorithm 1 and Algorithm 2 by the procedure to calculate t given by (2.24) yields cyclic and parallel minimization algorithms with the Huber regularizing term.

We can use the Huber function for the data term, i.e., we let $F_h(x, f) = \sum_{i,j} h \circ d(x_{ij}, f_{ij})$. For small distances the Huber data term behaves like the ℓ^2 data term but it is more robust to outliers. The proximal mapping of F_h is given by (cf. the proof of Theorem 2)

$$\begin{aligned} &(\mathrm{prox}_{\lambda F_h})_{ij}(x) = [x_{ij}, f_{ij}]_t, \\ &\text{where } t = \begin{cases} \frac{2\lambda\tau^2}{1+2\lambda\tau^2} d(x_{ij}, f_{ij}), & \text{if } d(x_{ij}, f_{ij}) < \frac{\omega(1+2\lambda\tau^2)}{\sqrt{2}\tau}, \\ \min\left(d(x_{ij}, f_{ij}), \sqrt{2}\lambda\omega\tau\right), & \text{otherwise.} \end{cases} \end{aligned} \tag{2.25}$$

Using the proximal mapping of the Huber data term (2.25) instead of the proximal mapping of F in Algorithm 1 and Algorithm 2, respectively, yields cyclic and parallel TV minimization algorithms with the Huber data term.

3. Convergence in Hadamard spaces. In this section we show the convergence of Algorithm 1 and Algorithm 2 for a certain class of spaces which the TV functionals we consider are convex on.

For general Riemannian manifolds, the ℓ^p-TV^q functional (2.1) is not necessarily convex. The perhaps simplest example where convexity fails is the one-dimensional sphere S^1 (cf. [62]). In the non-convex case, the study of (global) convergence becomes

much more involved and is out of the scope of this paper. Here, we treat the quite large class of Cartan-Hadamard manifolds where we still have convexity. These are complete Riemannian manifolds of nonpositive sectional curvature. Prominent examples are the spaces of positive matrices (which are the data space in diffusion tensor imaging) and the hyperbolic spaces. For details we refer to [25] or to [5].

The proofs in this section work in the more general setup of Hadamard spaces without additional effort. Hadamard spaces are certain metric spaces generalizing the concept of Cartan-Hadamard manifolds which are particular instances. Examples of Hadamard spaces which are not Cartan-Hadamard manifolds are the metric trees of [64]. Since there is no additional effort, we consider Hadamard spaces as underlying spaces in this section.

A Hadamard space is a geodesic space, i.e., for each two points x, y the length of the shortest arc connecting x and y (which exists by definition) equals the distance of the points. Furthermore, there is a certain condition ensuring that the geodesic triangles are "not fat" (Triangles on the sphere are "fat".). This condition providing the essential properties of Cartan-Hadamard manifolds in this metric space setting is as follows. For given x_0, x_1 there is a point y on the geodesic joining them such that for every z, $d^2(z,y) \leq \frac{1}{2}d^2(z,x_0) + \frac{1}{2}d^2(z,x_1) - \frac{1}{4}d^2(x_0,x_1)$. For details we refer to [64] and the references therein or to the book [10].

We note that the ℓ^p-TV^q functionals given by (2.1) are convex in a Hadamard space. This is because the distance function is doubly convex in Hadamard spaces; see [64].

Our first goal is to show the convergence of Algorithm 1 which is the geodesic averaging algorithm based on cyclical application of proximal mappings.

THEOREM 2. *For data in a (locally compact) Hadamard space Algorithm 1 converges towards a minimizer of the ℓ^p-TV^q functional. The statement remains true when using Huber data and regularizing terms based on* (2.22).

Proof. We first show that, in a Hadamard space, the proximal mappings of the functions F, G_{ij} and H_{ij} are given by (2.10),(2.6),(2.7) and their analogues for H_{ij}. We also show that the proximal mappings of the Huber regularizing and data terms are given by (2.23), (2.24) and (2.25), respectively. We start with the mappings G_{ij}. The proximal mapping of G_{ij} is given by

$$\operatorname{prox}_{\lambda G_{ij}} x = \operatorname{argmin}_y \lambda \tfrac{1}{q} d(y_{ij}, y_{i+1,j})^q + \tfrac{1}{2} \sum_{k,l} d(y_{kl}, x_{kl})^2. \tag{3.1}$$

Hence, every minimizer y^* must fulfill $y^*_{kl} = x_{kl}$ for $k \neq i$ and $l \neq j, j+1$. Otherwise, letting $y^*_{kl} = x_{kl}$ for $k \neq i$ and $l \neq j, j+1$ would decrease the functional value which contradicts the minimizer property. This shows (2.6).

Now let y^* be a minimizer of (3.1). We show that the four points $x_{ij}, x_{i,j+1}, y^*_{ij}$, $y^*_{i,j+1}$ lie on one geodesic. We may assume that $d(x_{ij}, y^*_{ij}) \leq d(x_{ij}, x_{i,j+1})$ since, otherwise, setting $y'_{ij} = y'_{i,j+1} = x_{i,j+1}$ would yield a lower functional value in (3.1). By the same argument, we may assume that $d(x_{i,j+1}, y^*_{i,j+1}) \leq d(x_{ij}, x_{i,j+1})$.

We define the point $z = [x_{ij}, x_{i,j+1}]_{t_1}$ as the point reached on the unit speed geodesic starting at x_{ij} after time $t_1 = d(x_{ij}, y^*_{ij})$. Analogously, we let $z' = [x_{i,j+1}, x_{i,j+1}]_{t_2}$ be the point on the same geodesic when starting from x_{k+1} after time $t = d(x_{k+1}, y^*_k)$.

We first consider the case with ordering $x_{ij}, z, z', x_{i,j+1}$ when running on the geodesic starting at x_k (including the case $z = z'$.) Since these points lie on a geodesic,

we have that

$$\begin{aligned} d(x_{ij}, x_{i,j+1}) &= d(x_{ij}, z) + d(z, z') + d(z', x_{i,j+1}) \\ &\leq d(x_{ij}, y^*_{ij}) + d(y^*_{ij}, y^*_{i,j+1}) + d(y^*_{i,j+1}, x_{i,j+1}) \end{aligned} \tag{3.2}$$

Here, the inequality is true since every geodesic is a shortest path in a Hadamard space. By our choice of t_1, t_2, this implies $d(z, z') \leq d(y^*_{ij}, y^*_{i,j+1})$. As a consequence, the functional value $a(z, z') \leq a(y^*_{ij}, y^*_{i,j+1})$ where

$$a(v, v') = \tfrac{1}{2} d(v, x_{ij})^2 + \lambda \tfrac{1}{q} d(v, v')^q + \tfrac{1}{2} d(v', x_{i,j+1})^2. \tag{3.3}$$

This is the essential part of the functional in (3.1) meaning that minimizing the functional in (3.1) is equivalent to minimizing a. Hence, since geodesics are unique in a Hadamard space, $z = y^*_{ij}$ and $z' = y^*_{i,j+1}$. Thus, these four points lie on a geodesic.

Next, we consider the case with ordering x_{ij}, z', z, $x_{i,j+1}$ when running on the geodesic starting at x_k. Then we have that $a(z, z) \leq a(z, z')$, since $d(z, x_{ij}) < d(z, x_{i,j+1})$. Hence, we obtain a lower functional value which means that this situation cannot occur for a minimizer of (3.1).

Summing up, we know that the points $x_{ij}, y^*_{ij}, y^*_{i,j+1}, x_{i,j+1}$ lie on a geodesic in this ordering.

Next, we need the precise position of $y^*_{ij}, y^*_{i,j+1}$ on the geodesic. We consider the real numbers $d = d(x_{ij}, x_{i,j+1})$ and the time points t_1 and t_2 given above. Minimization of a is now equivalent to minimizing, for $0 \leq t_1, t_2 \leq d/2$,

$$a'(t_1, t_2) = \tfrac{1}{2} t_1^2 + \lambda \tfrac{1}{q} (d - t_1 - t_2)^q + \tfrac{1}{2} t_2^2.$$

By symmetry and uniqueness, a minimizer fulfills $t_1 = t_2$. Hence, we have to find a minimizer of

$$a''(t) = t^2 + \tfrac{\lambda}{q} (d - 2t)^q.$$

For $q = 1$, we get the solution

$$t = \min(\lambda, d/2),$$

and for $q = 2$,

$$t = \frac{\lambda d}{2 + 2\lambda}.$$

This shows (2.7) and the subsequent formulas (2.8) and (2.9). The corresponding proof for the H_{ij} is analogous.

Next, we consider the Huber data term F_h based on (2.22). We show that its proximal mapping is given by (2.25). Similar as above, we define the point $z = [x_{ij}, f_{ij}]_t$, where $t = d(x_{ij}, y^*_{ij})$ and y^*_{ij} is the $(i,j)^{th}$ component of the proximal mapping of λF_h at x. Modifying the arguments above, we see that $z = y^*_{ij}$. Thus the three points x_{ij}, y^*_{ij}, f_{ij} lie on one geodesic and it remains to determine t. This leads to minimizing the scalar problem $t \mapsto 2\lambda h(d - t) + t^2$ under the constraint $0 \leq t \leq d$, where we let $d = d(x_{ij}, f_{ij})$. To solve this problem one applies an analogous calculation as the one in Example 4.5 of [17] and concludes (2.25). The corresponding proof of (2.10) for the ℓ^p type data term F is analogous.

It remains to consider the Huber regularizer G^h+H^h from Section 2.5. We proceed analogous to the proof for the regularizer $G+H$ to obtain that the four points $x_{ij}, y^*_{ij}, y^*_{i,j+1}, x_{i,j+1}$ lie on a geodesic in this ordering which shows (2.23). Then proceeding as in the proof for the Huber data term F_h with a similar calculation as in [17] we obtain the formula (2.24).

Since Algorithm 1 only produces convex combinations of the points involved, we have that the iterates produced by the algorithm stay in the convex hull of the data $(f_{i,j})_{i,j}$. Since all functions F, G_{ij}, H_{ij} are continuous, they are Lipschitz on that convex hull. Hence the assumptions of [4, Theorem 3.4] are fulfilled, and the application of this theorem yields the convergence of Algorithm 1. □

Next, using the above techniques, we supply the proof of Proposition 1 which was a statement on proximal mappings in the setup of Riemannian manifolds (not Hadamard spaces).

Proof of Proposition 1. We show the statement for the functions G_{ij}. As in the proof of Theorem 2 we see that $y^*_{k,l} = x_{k,l}$ for $k \neq i, i+1$ and $l \neq j$. Then we consider the components y^*_{ij} and $y^*_{i,j+1}$ of a minimizer. We fix a shortest geodesic connecting x_{ij} and $x_{i,j+1}$. We define the points z, z' on this geodesic as in the proof of Theorem 2. Then we see that we may apply the same arguments as in that proof to reduce to the situation where $x_{ij}, z', z, x_{i,j+1}$ lie on the geodesic in this ordering. Then the estimate (3.2) is true since we chose a shortest geodesic. This implies $d(z, z') \leq d(y^*_{ij}, y^*_{i,j+1})$. On the other hand, y^* is a minimizer of G_{ij}. This implies $d(y^*_{ij}, y^*_{i,j+1}) \leq d(z, z')$. Hence, equality holds and y^*_{ij} and $y^*_{i,j+1}$ lie on a shortest geodesic connecting x_{ij} and $x_{i,j+1}$. The statements for the functionals H_{ij} and F follow analogously. □

The goal of the rest of this section is to show that Algorithm 2 and its fast variant (with the approximate mean calculation from (2.20)) converge in a Hadamard space. To this end, we first show a generic convergence statement for parallel proximal point algorithms.

THEOREM 3. *We consider a convex function g defined on a Hadamard space which has a minimizer. Let $g = g_1 + \ldots + g_n$ and assume that all summands are convex and lower semicontinuous. Assume further that the positive parameter sequence $\lambda = (\lambda_1, \ldots)$ is square-summable but not summable. We consider the iteration*

$$x^{k+1} = \operatorname{mean}\left(\operatorname{prox}_{\lambda_k g_1} x^k, \ldots, \operatorname{prox}_{\lambda_k g_n} x^k\right). \tag{3.4}$$

Here mean *is the intrinsic mean in the Hadamard space defined by* (2.15)*. If there is a constant $L > 0$ such that, for all g_i and all k,*

$$g_i(x^k) - g_i(\operatorname{prox}_{\lambda_k g_1} x^k) \leq L \cdot d(x^k, \operatorname{prox}_{\lambda_k g_1} x^k). \tag{3.5}$$

then the iteration (3.4) *converges to a minimizer of g.*

The proof of this statement is an adaption of the proof of Theorem 3.4 in [4] to the parallel setting. In [4], the applied method of proof is addressed to [8]. We need the following two lemmas which are Lemma 2.6 and Lemma 3.2 of [4].

LEMMA 4. *Let a_k, b_k, c_k be sequences of positive numbers. Assume that $\sum c_k < \infty$ and that, for all k, $a_{k+1} < a_k - b_k + c_k$. Then the sequence a_k converges and $\sum b_k < \infty$.*

LEMMA 5. *Consider a convex and lower semicontinuous function h on a (locally compact) Hadamard space. Then,*

$$h(\operatorname{prox}_{\lambda h} x) - h(y) \leq \tfrac{1}{2\lambda}\left(d(x,y)^2 - d(\operatorname{prox}_{\lambda h} x, y)^2\right), \tag{3.6}$$

for any y in the Hadamard space.

Equipped with these preparations, we show Theorem 3.

Proof of Theorem 3. The function $x \mapsto d(x,y)^2$ is uniformly convex (cf. [64]). Thus, using Jensen's inequality (cf. [64]), we get, for the intrinsic mean in the Hadamard space x^{k+1}, that

$$d(x^{k+1},y)^2 \leq \sum_{i=1}^{n} \tfrac{1}{n} d(\operatorname{prox}_{\lambda_k g_i} x^k, y)^2, \tag{3.7}$$

for all y. In the following, we use the notation $\tilde{x}_i^{k+1} = \operatorname{prox}_{\lambda_k g_i} x^k$ for the proximal mapping of g_i at the previous iterate x^k. Using Lemma 5, we estimate

$$\begin{aligned} d(\tilde{x}_i^{k+1},y)^2 &\leq d(x^k,y)^2 - 2\lambda_k \left(g_i(\tilde{x}_i^{k+1}) - g_i(y)\right) \\ &= d(x^k,y)^2 - 2\lambda_k \left(g_i(x^k) - g_i(y)\right) + 2\lambda_k \left(g_i(x^k) - g_i(\tilde{x}_i^{k+1})\right). \end{aligned} \tag{3.8}$$

We combine the estimates (3.7) and (3.8) to obtain

$$\begin{aligned} &d(x^{k+1},y)^2 \\ &\leq \frac{1}{n}\sum_{i=1}^{n} d(x^k,y)^2 - \frac{2\lambda_k}{n}\sum_{i=1}^{n}\left(g_i(x_n) - g_i(y)\right) + \frac{2\lambda_k}{n}\sum_{i=1}^{n}\left(g_i(x_n) - g_i(\tilde{x}_i^{k+1})\right) \\ &= d(x^k,y)^2 - \frac{2\lambda_k}{n}\left(g(x_n) - g(y)\right) + \frac{2\lambda_k}{n}\sum_{i=1}^{n}\left(g_i(x_n) - g_i(\tilde{x}_i^{k+1})\right). \end{aligned} \tag{3.9}$$

The next goal is to estimate the last summand on the right hand side. To this end, we use that, by assumption,

$$g_i(x^k) - g_i(\tilde{x}_i^{k+1}) \leq L d(x^k, \tilde{x}_i^{k+1}).$$

Furthermore, since $\tilde{x}_i^{k+1}$ minimizes the expression in the definition of the proximal mapping, we obtain

$$g_i(\tilde{x}_i^{k+1}) + \tfrac{1}{2\lambda_k} d(x^k, \tilde{x}_i^{k+1}) \leq g_i(x^k).$$

Applying these estimates successively yields

$$\begin{aligned} g_i(x^k) - g_i(\tilde{x}_i^{k+1}) &\leq L d(x^k, \tilde{x}_i^{k+1}) \\ &\leq L 2\lambda_k \frac{g_i(x^k) - g_i(\tilde{x}_i^{k+1})}{d(x^k, \tilde{x}_i^{k+1})} \leq 2\lambda_k L^2. \end{aligned} \tag{3.10}$$

This allows us to estimate the last summand on the right-hand side of (3.9) by

$$\frac{2\lambda_k}{n}\sum_{i=1}^{n}\left(g_i(x_n) - g_i(\tilde{x}_i^{k+1})\right) \leq \frac{2\lambda_k}{n}\sum_{i=1}^{n} 2\lambda_k L^2 = 4\lambda_k^2 L^2.$$

Thus, (3.9) now reads

$$d(x^{k+1},y)^2 \leq d(x^k,y)^2 - \frac{2\lambda_k}{n}\left(g(x_n) - g(y)\right) + 4\lambda_k^2 L^2. \tag{3.11}$$

Next, we consider a minimizer y^* of g and plug it into (3.11) above. Then $g(x_n) - g(y^*) \geq 0$, and we may apply Lemma 4 to (3.11). This yields that $d(x^k, y^*)$ converges as $n \to \infty$ and that

$$\sum_{n\in\mathbb{N}} \frac{2\lambda_k}{n}\left(g(x_n) - g(y)\right) < \infty. \tag{3.12}$$

Since the parameter sequence $\lambda = (\lambda_1, \ldots)$ is not summable, (3.12) implies that $g(x^{k_l}) \to g(y^*)$ on a subsequence k_l. Since $d(x^k, y^*)$ is bounded and the underlying space is locally compact, we may choose another subsequence k_{l_r} such that x^k converges on this subsequence; call the limit x^*. Then by the lower semicontinuity of g, we have $g(x^*) \leq g(y^*)$. Then, since y^* is a minimizer, also x^* is a minimizer. We apply Lemma 4 for a second time, but now for $y = x^*$. As a result $d(x^k, x^*)$ converges. Moreover, $d(x^k, x^*) \to 0$, since this is true on a subsequence. This completes the proof. □

For the fast variant of the parallel proximal algorithm introduced in Section 2.4 we replaced the intrinsic mean by the approximation (2.20). In order to obtain convergence of the corresponding algorithm, we need the following result.

THEOREM 6. *The statement of Theorem 3 remains true if we replace the intrinsic mean* (2.15) *by its approximation* (2.20).

Proof. We show that (3.7) remains true if we replace the intrinsic mean by the construction given in (2.20). By the convexity of the function $a(z) = d(z, y)^2$ we have that $a([z, y]_{td(z,y)}) \leq (1 - t)a(z) + ta(y)$ for all $z, y \in M$. We successively apply this inequality to every geodesic average in (2.20) (in a top-down fashion). As a first step, we obtain

$$a(x) \leq (1 - t_1')a([\ldots, \ldots [x_{i_l}, x_{j_l}]_{t_l'} \ldots]_{t_2'}) + t_1' a([\ldots, \ldots]_{t_3'}).$$

Proceeding further, we get

$$a(x) \leq c_1 a(x_1) + \ldots + c_n a(x_n), \tag{3.13}$$

where the c_k are products with factors t_r' and $(1-t_r')$. Reversing the construction of the t_r' in (2.18), we see that, for each x_k, the factor c_k equals $1/n$. Plugging the definition $a(z) = d(z, y)^2$ and $c_k = 1/n$ in (3.13) yields (3.7) for the construction (2.20). Then we can follow the rest of the proof of Theorem 3 to conclude the assertion of the present theorem. □

Our main results concerning the convergence of the parallel proximal algorithm are as follows.

THEOREM 7. *The parallel proximal algorithm for ℓ^p-TV^q minimization (Algorithm 2) and its approximative variant converge towards a minimizer in every (locally compact) Hadamard space. The statement remains true when using Huber regularization and Huber data terms based on* (2.22).

Proof. As a first step, we have to show that the proximal mappings of the functions G_e, G_o, H_e, H_o given in and below (2.13) are in fact given by (2.14) and the explanations following (2.14). We only consider G_e since the other cases are analogous. We use the fact that G_e is a sum of the functions G_{ij}. More precisely, $G_e = \sum_{j:j \text{ even}} \sum_i G_{ij}$. Considering this form of G_e, we see that the minimization problem in the definition of the proximal mapping separates into minimization problems which require minimizing expressions of the form (3.3). Hence, the $(i, j)^{th}$ component of the proximal mapping of G_e equals the corresponding component of the proximal mapping of G_{ij}. This proximal mapping has been considered in the proof of Theorem 2. Its $(i, j)^{th}$ component agrees with the expression in (2.14). This shows (2.14).

For the Huber regularizing term, we consider the functionals $G_e^h, G_o^h, H_e^h, H_o^h$ defined in analogy to G_e, G_h, H_e, H_o by replacing d^q by $h \circ d$ where d is the Huber function (2.22) in (2.13). Then following the argument for G_e in the previous paragraph, we see that the $(i, j)^{th}$ component of the proximal mappings of G_e^h agrees with the corresponding component of the proximal mapping of G_{ij}^h given by (2.23),(2.24).

Analogous statements are true for G_o^h, H_e^h, H_o^h. The proximal mappings of the ℓ^p type data term F and the Huber data term F_h have been shown to agree with (2.10) and (2.25), respectively, in the proof of Theorem 2.

The next step is to apply Theorem 3 and Theorem 6 (for the approximative variant). Since the algorithm only produces convex combinations and intrinsic means, the iterates produced by the algorithm stay in the convex hull of the data $(f_{i,j})_{i,j}$. So the involved functions (which are all continuous) are Lipschitz on this convex hull which means that (3.5) is fulfilled. Hence, we may apply Theorem 3 and Theorem 6 and conclude the assertion of the theorem. □

4. Applications. In this section, we apply the algorithms proposed in this paper to concrete manifolds which frequently occur in applications. The manifolds we consider are the space of positive matrices Pos_3, the spheres S^1 and S^2, the product space $S^1 \times \mathbb{R}^2$ (which appears in the context of nonlinear color models) as well as the rotation group.

In order to make Algorithm 1 and Algorithm 2 work in a specific manifold we have to compute geodesics and distances on this manifolds. This is accomplished using the Riemannian exponential mapping and its inverse. Recall that the exponential mapping $\exp_a : T_aM \to M$ returns the point $\exp_a v$ on the manifold which we obtain when following the unit speed geodesic starting at a into the direction of the given tangent vector v for time $\|v\|_a$. Conversely, the inverse of the exponential mapping $\exp_a^{-1} : M \to T_aM$ gives us the tangent vector $\exp_a^{-1} b$ at the point a which leads to the point b when following the geodesic with respect to this tangent vector for time $\|\exp_a^{-1} b\|_a$. Using these mappings, the point $[a, b]_t$ reached on the unit speed geodesic joining a and b after time t is given by

$$[a, b]_t = \exp_a(t \cdot \exp_a^{-1}(b)). \tag{4.1}$$

In order to calculate the geodesic path length t, we further have to calculate distances on the manifold under consideration (cf. Table 2.1 and Table 2.2). To this end, we use that the distance between points a and b is given by the length of the tangent vector $\exp_a^{-1}(b)$, i.e.,

$$d(a, b) = \|\exp_a^{-1}(b)\|_a. \tag{4.2}$$

Here the length is measured with respect to the Riemannian metric in the tangent space of a. Hence, in order to apply our algorithms for a specific data space, we only need to instantiate the exponential mapping and its inverse for the corresponding manifold. For the data spaces considered in this article, the exponential mappings and their inverse have closed expressions involving only basic arithmetic operations such as trigonometric functions or matrix exponentials.

The numerical experiments were conducted on a Macbook using a single core of a 2.6 GHz Intel Core i7 processor. (Parallelized implementations of our algorithms are out of the scope of this article.) For the experiments in Figure 4, we optimized the total variation parameter α with respect to the peak signal-to-noise ratio. In the other experiments, α was determined empirically. A simple choice for the sequence λ_r is $\lambda_r = cr^{-\omega}$ with $c > 0$. The sequence fulfills the condition to be in $\ell^2 \setminus \ell^1$ for each $0.5 < \omega \leq 1$. We here used $\omega = 0.95$ and $c = 3$. We observed only little differences when using different parameter pairs. In order to quantitatively assess the denoising performance of total variation regularization on manifolds we use the *signal-to-noise ratio improvement* (compare [67, p. 244]). We consider a manifold-valued version of

the signal-to-noise ratio improvement which is given by

$$\Delta\mathrm{SNR} = 10\log_{10}\left(\frac{\sum_{ij} d(g_{ij}, f_{ij})^2}{\sum_{ij} d(g_{ij}, x_{ij})^2}\right).$$

Here f is the noisy data, g the ground truth, and x the regularized restoration.

4.1. The space of positive matrices Pos_3 – Diffusion tensor imaging. Diffusion tensor imaging (DTI) is a non-invasive imaging modality based on nuclear magnetic resonance. It allows to quantify the diffusional characteristics of a specimen [6, 41]. Applications are the determination of fiber tract orientations [6] and the detection of brain ischemia [47]. Denoising is an important topic in DTI which has been addressed in various articles, e.g. [19, 55, 7].

In DTI, the diffusivity of water molecules is captured by a *diffusion tensor,* i.e., a (symmetric) positive (definite) 3×3 matrix $S(p)$ sitting at pixel p. It is reasonable to consider the space of diffusion tensors Pos_3 as Riemannian manifold with the Riemannian metric

$$g_D(W, V) = \mathrm{trace}(D^{-\frac{1}{2}} W D^{-1} V D^{-\frac{1}{2}});$$

see [55]. Here the symmetric matrices W, V represent tangent vectors in the point D. Equipped with this Riemannian metric the space of positive matrices becomes a Cartan-Hadamard manifold. Hence, by virtue of Theorem 2 and Theorem 7, the cyclic proximal point algorithm and both variants of the parallel algorithm converge to a global minimizer.

For the space of positive matrices, the Riemannian exponential mapping $\exp_D$ is given by

$$\exp_D(W) = D^{\frac{1}{2}} \exp(D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) D^{\frac{1}{2}}.$$

Here D is a positive matrix and the symmetric matrix W represents a tangent vector in D. The mapping exp is the matrix exponential. Furthermore, there is also a closed form expression for the inverse of the Riemannian exponential mapping: we have, for positive matrices D, E,

$$\exp_D^{-1}(E) = D^{\frac{1}{2}} \log(D^{-\frac{1}{2}} E D^{-\frac{1}{2}}) D^{\frac{1}{2}}.$$

The matrix logarithm log is well-defined since the argument is a positive matrix. The matrix exponential and logarithm can be efficiently computed by diagonalizing the symmetric matrix under consideration and then applying the scalar exponential and logarithm functions to the eigenvalues. The distance between D and E is just the length of the tangent vector $\exp_D^{-1}(E)$ which can be explicitly calculated by

$$d^2(D, E) = \sum_{l=1}^{3} \log(\kappa_l)^2,$$

where κ_l is the l^{th} eigenvalue of the matrix $D^{-\frac{1}{2}} E D^{-\frac{1}{2}}$.

The data actually measured in DTI are so-called diffusion weighted images (DWI) $D_v(p)$ which capture the directional diffusivity in the direction v at pixel p. The relation between the diffusion tensor $S(p)$ and the DWIs $D_v(p)$ at the pixel p is given by the Stejskal-Tanner equation

$$D_v(p) = A_0 e^{-b\ v^T S(p) v} \tag{4.3}$$

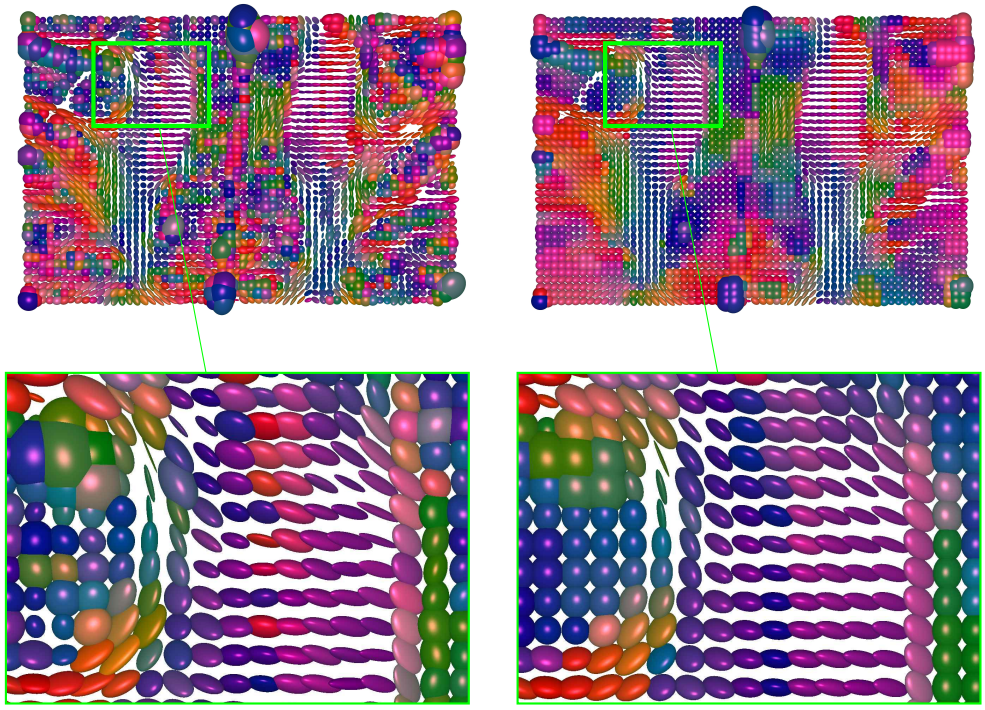


Fig. 1: *Left:* Diffusion tensor image of a human brain (axial cut); *Right:* Total variation denoising with ℓ^2 data term using the cyclic proximal point algorithm (Algorithm 1) using $\alpha = 0.11$. The runtime is 496.0 sec for 4000 iterations. The regularized image is much smoother than the original image. At the same time, strong changes of the orientations are preserved.

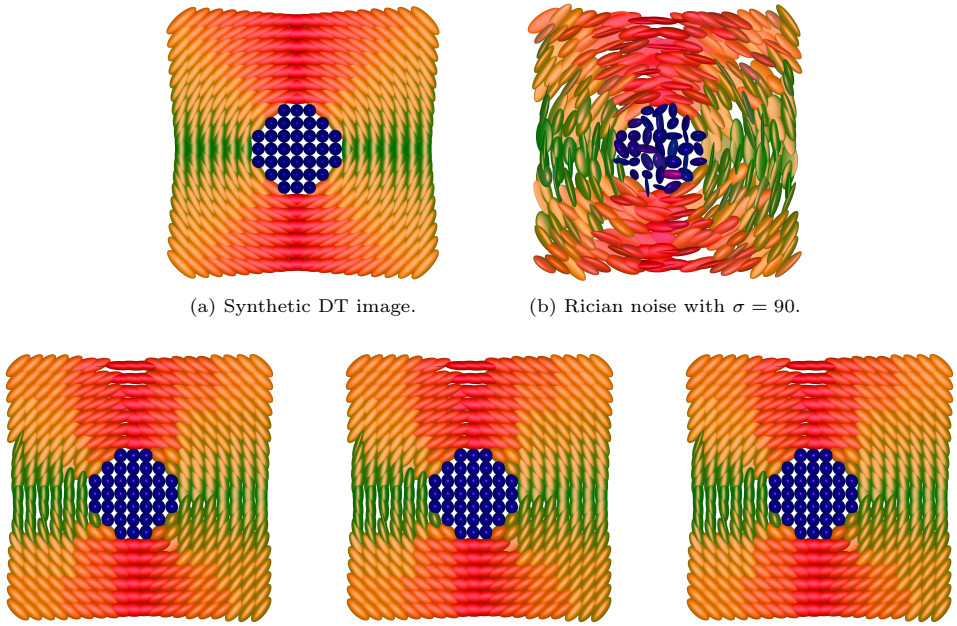
with constants $b, A_0 > 0$. Typically $b = 800$ and $A_0 = 1000$. Usually, 6 to 30 diffusion weighted images D_v (with different directions v) are measured [41, 3. IV C]. Being magnetic resonance images the DWIs are corrupted by Rician noise which arises from complex-valued Gaussian noise in the original frequency domain measurements [7]. This means that assuming the model (4.3) the actual measurement in direction v at pixel p is given by

$$D'_v(p) = \sqrt{(X + D_v(p))^2 + Y^2},$$

with the Gaussian random variables $X, Y \sim N(0, \sigma^2)$. Typically, the tensor $D_v(p)$ is obtained from the DWIs via a least square fit using the Stejskal-Tanner equation (4.3). In our synthetic examples, we impose Rician noise to 15 diffusion weighted images D'_v obtained from a synthetic diffusion tensor image S by (4.3). Then we apply least square fitting to the noisy DWIs to obtain a noisy diffusion tensor image.

In our experiments we visualize the diffusion tensors by the isosurfaces of the corresponding quadratic forms. More precisely, the ellipse visualizing the diffusion tensor $S(p)$ at pixel p are the points x fulfilling $(x - p)^T S(p)(x - p) = c$, for some $c > 0$.

In Figure 1, we apply ℓ^2-TV minimization to real DTI data of a human brain. The data set stems from the Camino project [23] and is freely accessible. We observe that TV minimization removes noise and preserves sharp boundaries between oriented



(a) Synthetic DT image. (b) Rician noise with $\sigma = 90$.

(c) Algorithm 1 (72.2 sec). (d) Algorithm 2 (205.0 sec). (e) Fast variant of Algorithm 2 (129.8 sec).

Fig. 2: ℓ^2-TV regularization of a diffusion tensor image with high noise level. Algorithm 1 as well as Algorithm 2 and its fast variant converge to the same solution. The TV regularization ($\alpha = 0.70$) removes almost all the noise and it preserves the sharp transitions. The signal-to-noise-ratio improvement is $\Delta\text{SNR} = 19.03$ in all three cases. The numbers in brackets denote the CPU time for 4000 iterations.

structures.

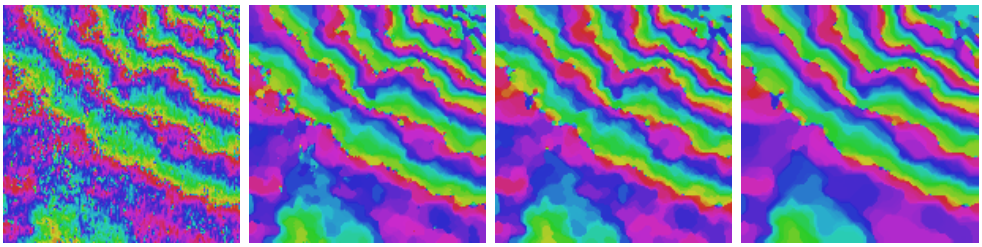
In Figure 2 we apply Algorithm 1 as well as Algorithm 2 and its fast variant for ℓ^2-TV minimization to a synthetic DTI on which we impose Rician noise. We observe the denoising capabilities of the proposed algorithms under a relatively high noise level; minimization of the ℓ^2-TV functional almost completely removes the noise while preserving sharp boundaries at the same time.

4.2. The one-dimensional sphere S^1 – InSAR images. Synthetic aperture radar (SAR) is a radar technique for sensing the earth's surface from measurements taken by aircrafts or satellites. Interferometric synthetic aperture radar (InSAR) images consist of the phase difference between two SAR images, recording a region of interest either from two different angles of view or at two different points in times. Important applications of InSAR are the creation of accurate digital elevation models and the detection of terrain changes; cf. [48, 58].

As InSAR data consists of phase values, the natural data space of InSAR images is the one-dimensional sphere S^1. The exponential mapping and its inverse have a particularly simple form when regarding S^1 as unit circle in the complex plane. Then the exponential mapping is given by

$$\exp_a(v) = e^{i(\theta+v)},$$

where $a = e^{i\theta}$ and $v \in]-\pi;\pi[$. For two non-antipodal points a and b the inverse



(a) InSAR image (real data). (b) ℓ^2-TV ($\alpha = 0.60$). (c) ℓ^1-TV ($\alpha = 0.60$). (d) TV with Huber data term ($\alpha = 0.60$).

Fig. 3: Total variation denoising of an InSAR image of dimension 150×150. The S^1-valued data are visualized as hue component in the HSV color space. Total variation minimization reliably removes the noise while preserving the structure of the image. We observe that ℓ^1 and Huber data terms are slightly more robust to outliers. In all three cases the runtime is about 20 sec for 600 iterations.

exponential map reads

$$\exp_a^{-1}(b) = \arg(b/a),$$

which is the polar angle of the complex number b/a. The distance between two points on the sphere reads $d(a, b) = |\arg(b/a)|$.

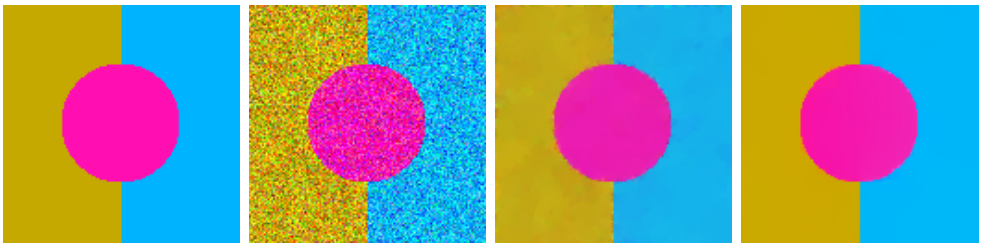
In Figure 3, we apply total variation denoising to a real InSAR image taken from [58]. This experiment shows the different effects of total variation regularization of using different data terms. We use ℓ^2 and ℓ^1 terms as well as the Huber term (with the parameters $\tau = \sqrt{2}$ and $\omega = 1$ in the definition of the Huber function (2.22)). We used 600 iterations of the cyclic proximal point algorithm. We observe that TV regularization reduces the noise significantly. The ℓ^1 data term and the Huber data term appear to be more robust to outliers than the ℓ^2 data term.

4.3. $\mathbb{R}^2 \times S^1$-valued images – Denoising in LCh color space. It was observed that total variation based denoising may give better results when using certain non-flat color models instead of the classical RGB color space [16]. One of these non-flat models is the HSV color space which leads to cylindrical data living in the product space $\mathbb{R}^2 \times S^1$.

We here use the LCh color space. Similar to the HSV space it is a cylindrical space consisting of a luminance component $L \in \mathbb{R}_0^+$, a chroma component $C \in \mathbb{R}_0^+$, and a hue component $h \in S^1$. The difference between HSV and LCh is that the first derives directly from the RGB space, whereas the latter derives from the Lab color space (also called $L^*a^*b^*$ space) which is intended to better match the human visual perception than the technically motivated RGB space. We perform the color space conversions using Matlab's built-in functions. For the hue and range preserving enhancement of color images see [54].

The exponential and the logarithmic mappings are given componentwise by the respective mappings on $\mathbb{R}^2$ and S^1. Note that in spite of this separability property, the proposed algorithm is not equivalent to performing the algorithm on $\mathbb{R}^2$ and S^1 separately (except for $p{=}q{=}2$). The reason is that the path length calculated according to Table 2.1 and Table 2.2 except for $p, q{=}2$ depend nonlinearly on the distance in the product manifold.

In Figure 4 we compare denoising in the RGB space with denoising in the LCh space. The RGB example was computed using the split Bregman method for TV



(a) Original image. (b) Gaussian noise (PSNR: 15.64). (c) ℓ^2-TV in RGB space ($\alpha = 0.22$, PSNR: 23.92). (d) ℓ^2-TV in LCh space ($\alpha = 0.90$, PSNR: 32.13).

Fig. 4: TV denoising in different color spaces. We see that measuring the distance in the non-flat LCh metric can lead to higher reconstruction quality than in the RGB color space. The runtime is 13 sec for 1000 iterations.

denoising which is a state-of-the-art method for vectorial total variation regularization [35, 34]. We optimized the corresponding model parameter with respect to the peak signal to noise ratio (PSNR) given by

$$\mathrm{PSNR}(x) = 10 \log_{10} \left(\frac{3mn \cdot (\max_{i,j,k} |g_{i,j,k}|)^2}{\sum_{i,j,k} |g_{i,j,k} - x_{i,j,k}|^2} \right).$$

where $g \in \mathbb{R}^{n \times m \times 3}$ denotes the ground truth (in RGB space). In Figure 4 we observe that the LCh color space denoising can indeed lead to better results than the vectorial RGB denoising.

4.4. S^2-valued images. We next apply our methods to images taking values in the two-dimensional sphere S^2. For example, spherical data appear in image processing in the context of chromaticity-based color models [16, 69] and as orientation fields of three dimensional images [57].

For a unit vector a on the unit sphere S^2 in $\mathbb{R}^3$ and a non-zero tangent vector v to the sphere at the point a, the exponential mapping is given by

$$\exp_a(v) = a \cdot \cos \|v\| + \frac{a \cdot \sin \|v\|}{\|v\|}.$$

The inverse $\exp_a^{-1}$ of the exponential mapping is well defined for non-antipodal points a and b and given by

$$\exp_a^{-1}(b) = \arccos(\langle a, b \rangle) \cdot \frac{b - \langle b, a \rangle a}{\|b - \langle b, a \rangle a\|},$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product in $\mathbb{R}^3$. The distance between a and b is $d(a, b) = \arccos(\langle a, b \rangle)$.

We test the denoising potential of our algorithm on a (synthetic) spherical-valued image. In the context of directional statistics a popular noise model on S^2 uses the von Mises-Fisher distribution having the probability density

$$f(x) = c(\kappa) \exp(\kappa \langle x, \mu \rangle).$$

Here the parameter $\kappa > 0$ expresses the concentration around the mean orientation $\mu \in S^2$ – the higher κ, the more concentrated the distribution. The constant $c(\kappa)$ is

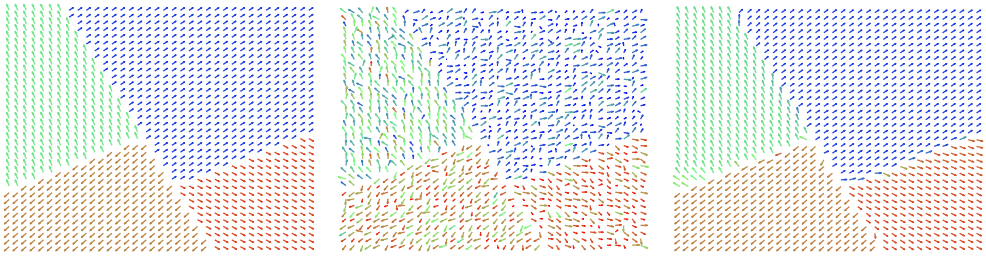


Fig. 5: Denoising of an S^2-valued image. The polar angle is coded both as length of the vectors and as color (red pointing towards the reader, blue away from the reader). *Left:* Original; *Center:* Von Mises-Fisher noise of level $\kappa = 12.7$; *Right:* ℓ^1-TV regularization using $\alpha = 0.5$. The noise is almost completely removed whereas the jumps are preserved (ΔSNR $= 6.85$). The runtime is 12.9 sec for 7000 iterations.

used for normalization to obtain a probability measure. For each data point $x_{ij} \in S^2$, we consider the above distribution with $\mu = x_{ij}$ and draw a sample. For the simulation of the distribution we used the implementation [42]; see [72] for a description of the algorithm. In Figure 5, we observe that the noise is almost completely removed by TV minimization and that the edges are retained.

4.5. SO(3) data. Measurements which involve the orientations of rigid objects in three-dimensional space lead to data which take their values in the rotation group SO(3). Examples of SO(3)-valued data are aircraft orientations [68] and protein alignments [37]. They also appear in the context of tracking 3D rotational data arising in robotics [27]; see also [49] for connections with directional statistics.

The special orthogonal group SO(3) consists of all orthogonal 3×3 matrices with determinant one, i.e.

$$SO(3) = \left\{ Q \in GL(3) \ : \ Q^t Q = I_3, \det Q = 1 \right\}.$$

As usual for matrix groups, we only consider the tangent space to SO(3) in the identity matrix I_3. It is given by the space of 3×3 skew-symmetric matrices so(3). Identifying the tangent space at an arbitrary point P with the tangent space at I_3 (via the differential of the left group action) the exponential mapping $\exp_P : \mathrm{so}(3) \to \mathrm{SO}(3)$ in the point P is given by

$$\exp_P(W) = \exp(W)P.$$

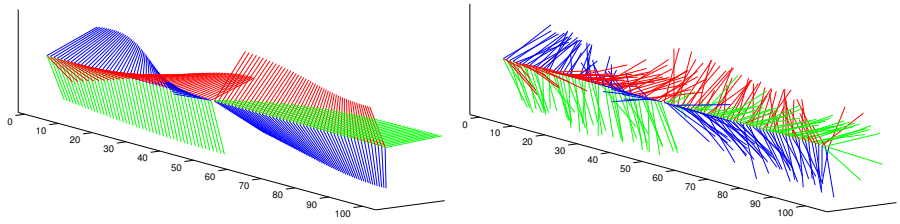
Here exp denotes the matrix exponential. For $P, Q \in SO(3)$, the "inverse" of the above exponential mapping reads

$$\exp_P^{-1}(Q) = \log(QP^t),$$

where log denotes the principal logarithm (which may be viewed as componentwise principal logarithm on the eigenvalues). The distance between P and Q equals the Frobenius norm of $\log(QP^t)$.

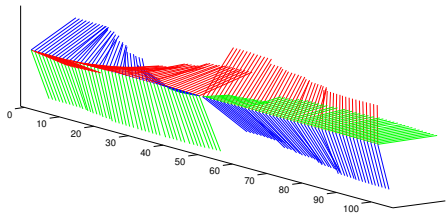
For the matrix operations needed above there are closed form expression available; see e.g. [49]. More precisely, to compute the matrix exponential of a skew-symmetric matrix W we use the Rodrigues formula

$$\exp(W) = I_3 + \frac{\sin(a)}{a} W + \frac{1 - \cos a}{a^2} W^2, \qquad \text{for } a = \sqrt{\mathrm{trace}(W^t W)} > 0.$$

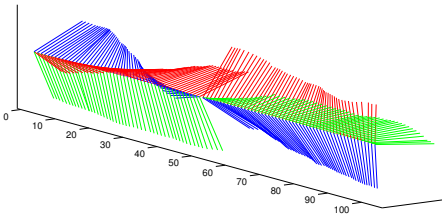


(a) Original.

(b) Fisher noise of level $\kappa = 75$.



(c) ℓ^2-TV ($\alpha = 2.0$, $\Delta\text{SNR} = 6.32$).



(d) ℓ^2-Huber ($\alpha = 0.9$, $\Delta\text{SNR} = 9.15$).

Fig. 6: Denoising of a SO(3)-valued time-series; TV regularization removes the noise and preserves the jump. The Huber regularization term gives even better results with less staircasing effects. The runtimes for TV and Huber regularization are 1.3 sec and 4.7 sec, respectively, for 1000 iterations.

For $a = 0$, we have $\exp(W) = I_3$. Concerning the principal matrix logarithm of a rotation matrix P, we let $\theta = \arccos(\text{trace}(P) - 1)/2$. If $\theta = 0$, then $\log(P) = 0$, the zero matrix. For $|\theta| < \pi$, the principal logarithm of P is given by

$$\log(P) = \frac{\theta \cdot Y}{2\sin(\theta)}, \qquad \text{where } Y = (P - P^t)/2.$$

In Figure 6, we consider a synthetic $1D$ signal consisting of 130 rotation matrices (visualized as tripods). The signal varies smoothly except for a single jump at the 50^{th} matrix. We simulate noisy data using the matrix Fisher distribution [45] which is given by the density

$$f(X) = c_F \exp\left\{\text{trace}(F^t X)\right\}, \; X \in SO(3).$$

The matrix F is a fixed 3×3 parameter matrix which describes the mode and the concentration of the distribution. In the isotropic case, the concentration of the distribution can be described by a single parameter $\kappa > 0$ which can be regarded as noise level. A small value of κ corresponds to a high level of noise. Our simulation uses a method recently introduced in [33] and relies on the sampling of quaternions following a related Bingham distribution. To simulate the latter we used the code from [11] which implements the method in [40]. For details we refer to [33, Chapter 5].

The results in Figure 6 show that the proposed algorithm removes the noise. The resulting signal is smoothed while the jump is preserved. In that experiment, we also compare total variation with Huber regularization terms. We see that the Huber regularization exhibits less staircasing artifacts than TV regularization.

5. Conclusion and future research. In this article we have developed proximal point algorithms for total variation minimization for manifold-valued data. Our

experiments show the denoising capability of the developed algorithms in various manifolds appearing in applications. For Hadamard spaces, we obtain convergence towards a global minimizer of the TV functional.

In future work, we address Blake-Zisserman and Potts functionals for manifold-valued data.

Acknowledgement. This work was supported by the German Federal Ministry for Education and Research under SysTec Grant 0315508 and by the European Research Council (ERC) under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no. 267439. The authors would like to thank Gabriele Steidl for valuable discussions and for pointing out the PPXA algorithm in [22] which led us to consider the parallel proximal algorithms in this paper. They would also like to thank Klaus Hahn for valuable discussions.

REFERENCES

[1] S. Alliney. Digital filters as absolute norm regularizers. *IEEE Transactions on Signal Processing*, 40:1548–1562, 1992.

[2] A. Anagaw and M. Sacchi. Edge-preserving seismic imaging using the total variation method. *Journal of Geophysics and Engineering*, 9:138, 2012.

[3] D. Azagra and J. Ferrera. Proximal calculus on Riemannian manifolds. *Mediterranean Journal of Mathematics*, 2:437–450, 2005.

[4] M. Bačák. Computing medians and means in Hadamard spaces. Preprint, 2013.

[5] W. Ballmann, M. Gromov, and V. Schroeder. *Manifolds of nonpositive curvature*. Birkhäuser, Boston, 1985.

[6] P. Basser, J. Mattiello, and D. LeBihan. MR diffusion tensor spectroscopy and imaging. *Biophysical journal*, 66:259–267, 1994.

[7] S. Basu, T. Fletcher, and R. Whitaker. Rician noise removal in diffusion tensor MRI. In *Medical Image Computing and Computer-Assisted Intervention 2006*, pages 117–125. Springer, 2006.

[8] D. Bertsekas. Incremental proximal methods for large scale convex optimization. *Mathematical Programming*, 129:163–195, 2011.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3:1–122, 2011.

[10] M. Bridson and A. Haefliger. *Metric spaces of non-positive curvature*. Springer, Berlin, 1999.

[11] M. Brubaker, M. Salzmann, and R. Urtasun. A family of MCMC methods on implicitly defined manifolds. In *International Conference on Artificial Intelligence and Statistics*, pages 161–172, 2012.

[12] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20:89–97, 2004.

[13] A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76:167–188, 1997.

[14] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011.

[15] T. Chan and S. Esedoglu. Aspects of total variation regularized L^1 function approximation. *SIAM Journal on Applied Mathematics*, 65:1817–1837, 2005.

[16] T. Chan, S. Kang, and J. Shen. Total variation denoising and enhancement of color images based on the CB and HSV color models. *Journal of Visual Communication and Image Representation*, 12:422–435, 2001.

[17] C. Chaux, P. Combettes, J.-C. Pesquet, and V. Wajs. A variational formulation for frame based inverse problems. *Inverse Problems*, 23:1495–1518, 2007.

[18] C. Chefd'Hotel, D. Tschumperlé, R. Deriche, and O. Faugeras. Regularizing flows for constrained matrix-valued images. *Journal of Mathematical Imaging and Vision*, 20:147–162, 2004.

[19] B. Chen and E. Hsu. Noise removal in magnetic resonance diffusion tensor imaging. *Magnetic Resonance in Medicine*, 54:393–401, 2005.

[20] T. Chen, W. Yin, X. Zhou, D. Comaniciu, and T. Huang. Total variation models for variable

lighting face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1519–1524, 2006.

[21] C. Clason, B. Jin, and K. Kunisch. A semismooth Newton method for L^1 data fitting with automatic choice of regularization parameters and noise calibration. *SIAM Journal on Imaging Sciences*, 3:199–231, 2010.

[22] P. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212, 2011.

[23] P. Cook, Y. Bai, S. Nedjati-Gilani, K. Seunarine, M. Hall, G. Parker, and D. Alexander. Camino: Open-source diffusion-MRI reconstruction and processing. In *14th Scientific Meeting of the International Society for Magnetic Resonance in Medicine*, page 2759, 2006.

[24] N. Dey, L. Blanc-Feraud, C. Zimmer, P. Roux, Z. Kam, J.-C. Olivo-Marin, and J. Zerubia. Richardson-Lucy algorithm with total variation regularization for 3D confocal microscope deconvolution. *Microscopy Research and Technique*, 69:260–266, 2006.

[25] M. do Carmo. *Riemannian Geometry*. Birkhäuser, Boston, 1992.

[26] Y. Dong, M. Hintermüller, and M. Neri. An efficient primal-dual method for l^1-tv image restoration. *SIAM Journal on Imaging Sciences*, 2:1168–1189, 2009.

[27] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:932–946, 2002.

[28] O. Ferreira and P. Oliveira. Proximal point algorithm on Riemannian manifolds. *Optimization*, 51:257–270, 2002.

[29] R. Ferreira, J. Xavier, J. Costeira, and V. Barroso. Newton algorithms for riemannian distance related problems on connected locally symmetric manifolds. *IEEE Journal of Selected Topics in Signal Processing*, 7:634–645, 2013.

[30] P. Fletcher. Geodesic regression and the theory of least squares on Riemannian manifolds. *International Journal of Computer Vision*, 105:171–185, 2013.

[31] P. Fletcher and S. Joshi. Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Processing*, 87:250–262, 2007.

[32] P. Fletcher, C. Lu, S. Pizer, and S. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, 23:995–1005, 2004.

[33] A. Ganeiber. *Estimation and simulation in directional and statistical shape models*. PhD thesis, University of Leeds, 2012.

[34] P. Getreuer. Rudin-Osher-Fatemi total variation denoising using split Bregman. *Image Processing On Line*, 2012. `http://dx.doi.org/10.5201/ipol.2012.g-tvd`.

[35] T. Goldstein and S. Osher. The split Bregman method for L^1-regularized problems. *SIAM Journal on Imaging Sciences*, 2:323–343, 2009.

[36] Y. Gousseau and J.-M. Morel. Are natural images of bounded variation? *SIAM Journal on Mathematical Analysis*, 33:634–648, 2001.

[37] P. Green and K. Mardia. Bayesian alignment using hierarchical models, with applications in protein bioinformatics. *Biometrika*, 93:235–254, 2006.

[38] P. Grohs, H. Hardering, and O. Sander. Optimal a priori discretization error bounds for geodesic finite elements. Preprint, 2013.

[39] P. Grohs and J. Wallner. Interpolatory wavelets for manifold-valued data. *Applied and Computational Harmonic Analysis*, 27:325–333, 2009.

[40] P. Hoff. Simulation of the matrix Bingham-von Mises-Fisher distribution, with applications to multivariate and relational data. *Journal of Computational and Graphical Statistics*, 18:438–456, 2009.

[41] H. Johansen-Berg and T. Behrens. *Diffusion MRI: From quantitative measurement to in-vivo neuroanatomy*. Academic Press, London, 2009.

[42] S. Jung. Random number generation from von Mises–Fisher distribution, 2010. `http://www.unc.edu/~sungkyu/manifolds/randvonMisesFisherm.m`.

[43] H. Karcher. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics*, 30:509–541, 1977.

[44] W. Kendall. Probability, convexity, and harmonic maps with small image I: uniqueness and fine existence. *Proceedings of the London Mathematical Society*, 3:371–406, 1990.

[45] C. Khatri and K. Mardia. The von Mises-Fisher matrix distribution in orientation statistics. *Journal of the Royal Statistical Society. Series B*, 39:95–106, 1977.

[46] R. Kimmel and N. Sochen. Orientation diffusion or how to comb a porcupine. *Journal of Visual Communication and Image Representation*, 13:238–248, 2002.

[47] D. Le Bihan, J.-F. Mangin, C. Poupon, C. Clark, S. Pappata, N. Molko, and H. Chabriat. Diffusion tensor imaging: Concepts and applications. *Journal of Magnetic Resonance Imaging*, 13:534–546, 2001.

[48] D. Massonnet and K. Feigl. Radar interferometry and its application to changes in the earth's surface. *Reviews of Geophysics*, 36:441–500, 1998.

[49] M. Moakher. Means and averaging in the group of rotations. *SIAM journal on matrix analysis and applications*, 24:1–16, 2002.

[50] J.-J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math.*, 255:2897–2899, 1962.

[51] M. Nikolova. Minimizers of cost-functions involving nonsmooth data-fidelity terms. Application to the processing of outliers. *SIAM Journal on Numerical Analysis*, 40:965–994, 2002.

[52] M. Nikolova. A variational approach to remove outliers and impulse noise. *Journal of Mathematical Imaging and Vision*, 20:99–120, 2004.

[53] M. Nikolova, M. Ng, S. Zhang, and W. Ching. Efficient reconstruction of piecewise constant images using nonsmooth nonconvex minimization. *SIAM Journal on Imaging Sciences*, 1:2–25, 2008.

[54] M. Nikolova and G. Steidl. Fast hue and range preserving histogram specification: Theory and new algorithms for color image enhancement. Preprint, 2013.

[55] X. Pennec, P. Fillard, and N. Ayache. A Riemannian framework for tensor computing. *International Journal of Computer Vision*, 66:41–66, 2006.

[56] P. Petrushev, A. Cohen, H. Xu, and R. DeVore. Nonlinear approximation and the space $BV(\mathbb{R}^2)$. *American Journal of Mathematics*, 121:587–628, 1999.

[57] R. Rezakhaniha, A. Agianniotis, J. Schrauwen, A. Griffa, D. Sage, C. Bouten, F. Van de Vosse, M. Unser, and N. Stergiopulos. Experimental investigation of collagen waviness and orientation in the arterial adventitia using confocal laser scanning microscopy. *Biomechanics and Modeling in Mechanobiology*, 11:461–473, 2012.

[58] F. Rocca, C. Prati, and A. Ferretti. An overview of SAR interferometry. In *Proceedings of the 3rd ERS Symposium on Space at the Service of our Environment, Florence*, 1997. `http://earth.esa.int/workshops/ers97/program-details/speeches/rocca-et-al`.

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

[60] S. Setzer, G. Steidl, and T. Teuber. On vector and matrix median computation. *Journal of Computational and Applied Mathematics*, 236:2200–2222, 2012.

[61] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk. On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization, and SIDEs. *SIAM Journal on Numerical Analysis*, 42:686–713, 2004.

[62] E. Strekalovskiy and D. Cremers. Total variation for cyclic structures: Convex relaxation and efficient minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1905–1911, 2011.

[63] D. Strong and T. Chan. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse problems*, 19:S165, 2003.

[64] K.-T. Sturm. Probability measures on metric spaces of nonpositive curvature. In *Heat kernels and analysis on manifolds, graphs, and metric spaces*, volume 338 of *Contemp. Math.*, pages 357–390. American Mathematical Society, Providence, 2003.

[65] R. Tron, B. Afsari, and R. Vidal. Riemannian consensus for manifolds with bounded curvature. *IEEE Transactions on Automatic Control*, 58:921–934, 2013.

[66] D. Tschumperlé and R. Deriche. Diffusion tensor regularization with constraints preservation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I948–I953, 2001.

[67] M. Unser and P. Tafti. *An introduction to sparse stochastic processes*. 2013, to appear.

[68] I. Ur Rahman, I. Drori, V. Stodden, D. Donoho, and P. Schröder. Multiscale representations for manifold-valued data. *Multiscale Modeling and Simulation*, 4:1201–1232, 2005.

[69] L. Vese and S. Osher. Numerical methods for p-harmonic flows and applications to image processing. *SIAM Journal on Numerical Analysis*, 40:2085–2104, 2002.

[70] J. Wallner and N. Dyn. Convergence and C^1 analysis of subdivision schemes on manifolds by proximity. *Computer Aided Geometric Design*, 22:593–622, 2005.

[71] A. Weinmann. Interpolatory multiscale representation for functions between manifolds. *SIAM Journal on Mathematical Analysis*, 44:162–191, 2012.

[72] A. Wood. Simulation of the von Mises-Fisher distribution. *Communications in Statistics-Simulation and Computation*, 23:157–164, 1994.

[73] J. Yang, Y. Zhang, and W. Yin. A fast alternating direction method for TVL1-L2 signal reconstruction from partial Fourier data. *IEEE Journal of Selected Topics in Signal Processing*, 4:288–297, 2010.

[74] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L^1 optical flow. In *Pattern Recognition*, pages 214–223. Springer, 2007.