Computing (2007) 81: 179–197
DOI 10.1007/s00607-007-0248-9
Printed in The Netherlands

Computing

A generic approach to diffusion filtering of matrix-fields

B. Burgeth[1]*, **S. Didas**[1]**, **L. Florack**[2] and **J. Weickert**[1]

[1]Department of Mathematics and Computer Science, Saarland University, Saarbrücken, Germany

[2]Department of Biomedical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands

Received 2 February 2007; Accepted 22 August 2007; Published online 6 November 2007

Summary

Diffusion tensor magnetic resonance imaging, is a image acquisition method, that provides matrix-valued data, so-called matrix fields. Hence image processing tools for the filtering and analysis of these data types are in demand. In this article, we propose a generic framework that allows us to find the matrix-valued counterparts of the Perona–Malik PDEs with various diffusivity functions. To this end we extend the notion of derivatives and associated differential operators to matrix fields of symmetric matrices by adopting an operator-algebraic point of view. In order to solve these novel matrix-valued PDEs successfully we develop truly matrix-valued analogs to numerical solution schemes of the scalar setting. Numerical experiments performed on both synthetic and real world data substantiate the effectiveness of our novel matrix-valued Perona–Malik diffusion filters.

AMS Subject Classifications: Primary 35K55, 15A57; Secondary 35Q80.

Keywords: diffusion tensor magnetic resonance imaging; DT-MRI; Jordan product; Perona–Malik diffusion; matrix fields.

1. Introduction

Matrix-fields are used, for instance, in civil engineering to describe anisotropic behaviour of physical quantities. Stress and diffusion tensors are prominent examples. The output of diffusion tensor magnetic resonance imaging (DT-MRI) [22] are symmetric 3×3-matrix fields as well. In medical sciences this image acquisition technique has become an indispensable diagnostic tool in recent years.

There is an increasing demand to develop image processing tools for the filtering and analysis of such matrix-valued data.

* The Dutch Organization NWO is gratefully acknowledged for financial support.
** The German Organization DFG is gratefully acknowledged for financial support.

Correspondence: Bernhard Burgeth, Department of Mathematics and Computer Science, Saarland University, 66041 Saarbrücken, Germany (E-mail: burgeth@mia.uni-saarland.de)

In modern image processing d-dimensional scalar images $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ have been denoised, segmented and/or enhanced successfully with various filters described by nonlinear parabolic PDEs. In this article we focus on one of the most prominent examples of PDEs used in image processing, the Perona–Malik equation [21]. The corresponding initial boundary value problem is given by

$$\begin{aligned} \partial_t u - \operatorname{div}(g_\lambda(|\nabla u|^2) \cdot \nabla u) &= 0 \quad \text{in } I \times \Omega, \\ \partial_n u &= 0 \quad \text{in } I \times \partial\Omega, \\ u(x,0) &= f(x) \quad \text{in } \Omega, \end{aligned} \tag{1}$$

where $\Omega \subset \mathbb{R}^d$ is the image domain and $I = [0, T[$ a potentially unbounded time interval.

The diffusivity function g_λ with parameter $\lambda > 0$ is positive, decreasing on the interval $[0, +\infty[$ with $g_\lambda(0) = 1$ and $\lim_{x\to+\infty} g_\lambda(x) = 0$. Practically relevant are diffusivities such as the Perona–Malik diffusivity [20]:

$$g_\lambda(s^2) = \frac{1}{1 + (\frac{s}{\lambda})^2} \tag{2}$$

or the family of Weickert diffusivities [29]

$$g_{\lambda,p}(s^2) = 1 - \exp\left(-\frac{c_p}{(\frac{s}{\lambda})^{2p}}\right), \tag{3}$$

where c_p is a normalising constant such that

$$\frac{d}{ds}(s \cdot g_\lambda(s^2)) \mid_{s=\lambda} = 1 - \frac{1 + 2pc_p}{\exp(c_p)} = 0.$$

For $p = 4$ one obtains $c_4 = 3.31488$. Noticing that c_p depends logarithmiclly on $p > 0$ it is not hard to see that in the limit we get a 0–1-diffusivity

$$g_{\lambda,\infty}(s) := \lim_{p\to+\infty} g_{\lambda,p}(s) = \begin{cases} 1 & \text{for } 0 \le s \le \lambda \\ 0 & \text{for } \lambda < s\,. \end{cases} \tag{4}$$

In effect the diffusivities (2,3) entail a forward diffusion in the image at locations where $|\nabla u| < \lambda$ and a backward diffusion where $|\nabla u| > \lambda$. This accounts for the well-known edge-preserving or even edge-enhancing properties of this nonuniform process, since edges are locii of high grey value variations. Hence the visually impressive denoising results when these filter type is applied do not come as a surprise. However, backward diffusion is an ill-posed process and hence some unwanted effects appear such as the creation of artificial edges known as staircaising. Theory has not yet progressed so far to be able to predict where these discontinuities appear during Perona–Malik diffusion in dimension $d \ge 2$. Investigations even in the case of the continuous Perona–Malik diffusion in one (spatial) dimension proved to be extremely difficult [9], [18], [28], [31], [16], [23], [12], [2], [19], [4], [34]. Nevertheless, in practice Perona–Malik-type diffusion provides a successful method to smooth noisy images while preserving important contour information [17], [31], [25]. Extensions of nonlinear PDEs from scalar grey value to vectorial colour images have been proposed in [14], [27], [10], [30]. There, in essence, the summation over the structure tensors stemming from the vector components ensure an appropriate channel coupling.

The goal of this article is to extend the important Perona–Malik diffusion process to matrix-valued images or matrix fields, for short. Here a matrix field is considered as a mapping

$$F : \Omega \subset \mathbb{R}^d \longrightarrow M_n(\mathbb{R}),$$

from a d-dimensional image domain into the set of $n \times n$-matrices with real entries, $F(x) = (f_{p,q}(x))_{p,q=1,\dots,n}$. Of particular importance for us is the subset of symmetric matrices $\mathrm{Sym}_n(\mathbb{R})$. The set of positive (semi-)definite matrices, denoted by $\mathrm{Sym}_n^{++}(\mathbb{R})$ ($\mathrm{Sym}_n^{+}(\mathbb{R})$, resp.), consists of all symmetric matrices A with

$$\langle v, Av\rangle := v^\top Av > 0 \quad (\geq 0, \text{resp.,}) \quad \text{for } v \in \mathbb{R}^n \setminus \{0\}\,.$$

This set is interesting for applications since DT-MRI acquisition technique produces data with this property. Note that at each point the matrix $F(x)$ of a field of symmetric matrices can be diagonalised, respectively, decomposed into its spectral components yielding

$$F(x) = V(x)^\top D(x)V(x) = \sum_{i=1}^{n} \lambda_i(x)\; v_i(x)v_i^\top(x).$$

Here $x \mapsto V(x) \in O(n)$ is a matrix field of orthogonal matrices $V(x)$ with column vectors $v_i(x)$, $i = 1, \dots, n$, while $x \mapsto D(x)$ is a matrix field of diagonal matrices with entries $\lambda_i(x)$, $i = 1, \dots, n$. In the sequel we will denote $n \times n$ – diagonal matrices with entries $\lambda_1, \dots, \lambda_n \in \mathbb{R}$ from left to right simply by $\mathrm{diag}(\lambda_i)$, and $O(n)$ stands for the matrix group of orthogonal $n \times n$-matrices.

Nonlinear partial differential equations have been employed to process matrix fields in [13] and more recently in [24]. Some extensions of scalar PDEs to matrices proposed in these works rely on generalisations of the so-called structure tensor. The considerations in [32], [6] spearheaded these generalisations of structure-tensor concepts.

Other approaches to positive definite matrix field filtering with a differential geometric background have been suggested in [26], [11]. In their setting the set of positive definite matrices is endowed with a structure of a manifold and the methodology is geared towards application to DT-MRI data. Comprehensive survey articles on the analysis of matrix fields using a wide range of different techniques can be found in [33] and the literature cited therein.

The path we take in this article is a different one. We will develop a general generic framework for deriving matrix-valued counterparts for scalar PDEs by adopting an operator-algebraic point of view. This means that we are not just deriving systems of PDEs which can be written in matrix form. Instead we will exploit the operator-algebraic properties of (symmetric) matrices to establish truly matrix-valued PDEs. We consider the symmetric matrices as a natural generalisation of real numbers with a rich algebraic structure. For this work we concentrate on the matrix-valued analogs of the Perona–Malik PDE for a proof-of-concept. It is also worth mentioning that in contrast to [13], [24], [5] our framework does not rely on a notion of a structure tensor. Nevertheless, the proposed concept ensures an appropriate and desirable coupling of channels. The methodology to be developed will enable us to transfer numerical schemes from the scalar to the matrix-valued setting as well.

The article is organised as follows: Sect. 2 provides the basic definitions necessary for our framework, such as functions of a matrix, partial derivatives, and generalised gradient of a matrix field. In Sect. 3, we turn first to the simple linear diffusion for matrix fields for the sake of later comparison. The Perona–Malik PDE requires the definition of a symmetrised multiplication for symmetric matrices. We will focus on two possibilities and study their influence on the evolution process later on. Within this framework we then formulate the matrix-valued counterparts of the Perona–Malik diffusion equation. By considering the already rather complicated one-dimensional case, first properties of the matrix-valued Perona–Malik diffusion processes are inferred.

The transition from scalar numerical solution schemes to matrix-valued algorithms for the solutions of the new diffusion equations is made in Sect. 4. Exemplary applications of the proposed framework to synthetic data as well as real DT-MRI data are presented in Sect. 5. We conclude with a summary in Sect. 6. Some results related to this work have been presented at a conference [8]. However, the investigations presented here encompass a more detailed analysis of the suitable symmetric matrix products, the enhancement properties of nonlinear diffusion processes, and a significantly extended experimental validation.

2. Generic framework for matrix-valued PDEs

In this section, we provide the key definitions for the formulation of matrix-valued PDEs. The basic idea is that to a certain extend symmetric matrices can be regarded as a generalisation of real numbers. Hence we transfer notions from scalar calculus to the the matrix-valued setting: as instigated in [7] we define functions of matrices and especially derivatives and gradients of such functions.

We juxtapose the corresponding basic definitions in Table 1, and comment on them in the remarks below. We assume the matrix field $U(x)$ to be diagonisable with $U = (u_{ij})_{ij} = V^\top \mathrm{diag}(\lambda_1, \ldots, \lambda_n)V$, where $V \in O(n)$ and $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$.

Comments:

(1) The proposed notions for a calculus on symmetric matrix fields are extensions of the calculus of scalar multivariate functions. As such it must be possible to regain the scalar calculus from the newly introduced matrix-valued framework by specification. There are two ways to view scalar calculus as a special case of the matrix calculus: clearly, setting $n = 1$ turns the matrix field into a scalar function. However, one can also embed the set of real numbers $\mathbb{R}$ into the set of symmetric matrices $\mathrm{Sym}_n(\mathbb{R})$ by the identification $\mathbb{R} \ni r \longleftrightarrow r \cdot I_n$ with the $n \times n$ identity matrix I_n. Hence, asides from having a certain simplicity, it is mandatory that the proposed extensions collapse to the scalar calculus when making the transition from scalar functions to matrix fields in one way or the other.

Table 1. Extensions of elements of scalar valued calculus (*middle*) to the matrix-valued setting (*right*)

Setting	Scalar valued	Matrix-valued
Function	$h : \begin{cases} \mathbb{R} \longrightarrow \mathbb{R} \\ x \mapsto h(x) \end{cases}$	$h : \begin{cases} \mathrm{Sym}_n(\mathbb{R}) \longrightarrow \mathrm{Sym}_n(\mathbb{R}) \\ U \mapsto V^\top \mathrm{diag}(h(\lambda_1), \ldots, h(\lambda_n)) V \end{cases}$
Partial derivatives	$\partial_\omega u,$ $\omega \in \{t, x_1, \ldots, x_d\}$	$\overline{\partial}_\omega U := (\partial_\omega u_{ij})_{ij},$ $\omega \in \{t, x_1, \ldots, x_d\}$
Higher derivatives	$\partial^k_\omega u,$ $\omega \in \{t, x_1, \ldots, x_d\}$	$\overline{\partial}^k_\omega U := \left(\partial^k_\omega u_{ij}\right)_{ij},$ $\omega \in \{t, x_1, \ldots, x_d\}$
Laplacian	$\Delta u := \sum_{i=1}^d \partial^2_{x_i} u$	$\overline{\Delta} U := \sum_{i=1}^d \overline{\partial}^2_{x_i} U$
Hessian	$\mathcal{H} u(x) := \left(\partial_{x_i}\partial_{x_j}\, u(x)\right)_{i,j=1,\ldots,d},$ $\mathcal{H} u(x) \in \mathrm{Sym}_d(\mathbb{R})$	$\overline{\mathcal{H}} U(x) := (\overline{\partial}_{x_i}\overline{\partial}_{x_j}\, U(x))_{i,j=1,\ldots,d},$ $\overline{\mathcal{H}} U(x) \in \mathrm{Sym}_d(\mathrm{Sym}_n(\mathbb{R}))$
Gradient	$\nabla u(x) := (\partial_{x_1}\, u(x), \ldots, \partial_{x_d}\, u(x))^\top,$ $\nabla u(x) \in \mathbb{R}^d$	$\overline{\nabla} U(x) := (\overline{\partial}_{x_1}\, U(x), \ldots, \overline{\partial}_{x_d}\, U(x))^\top,$ $\overline{\nabla} U(x) \in (\mathrm{Sym}_n(\mathbb{R}))^d$
Divergence	$\mathrm{div}\, (a(x))^\top := \sum_{i=1}^d \partial_{x_i} a_i(x),$ $a(x) := (a_1(x), \ldots, a_d(x))$	$\overline{\mathrm{div}}\, (A(x))^\top := \sum_{i=1}^d \overline{\partial}_{x_i} A_i(x),$ $A(x) := (A_1(x), \ldots, A_d(x))$
Length	$\|w\|_p := \sqrt[p]{\|w_1\|^p + \ldots + \|w_d\|^p},$ $\|w\|_p \in [0, +\infty[$	$\|W\|_p := \sqrt[p]{\|W_1\|^p + \ldots + \|W_d\|^p},$ $\|W\|_p \in \mathrm{Sym}^+_n(\mathbb{R})$
Product	$a \cdot b$	$A \bullet_P B := A^{\frac{1}{2}} B A^{\frac{1}{2}},$ $A \bullet_J B := \frac{1}{2}(AB + BA)$

(2) **Functions of matrices.** The definition of a function h on $\mathrm{Sym}_n(\mathbb{R})$ is standard [15]. As an important example, we emphasise that $|U|$ denotes the matrix-valued equivalent of the *absolute value* of a real number, $|U| = V^\top \mathrm{diag}(|\lambda_1|, \ldots, |\lambda_n|) V \in \mathrm{Sym}^+_n(\mathbb{R})$, not to be confused with the determinant $\det(U)$ of U. Note that $|U| = \sqrt{U^2}$ is in complete accordance with the scalar case.

(3) **Partial derivatives.** The *componentwise* definition of the partial derivative for matrix fields is a natural extension of the scalar case:

$$\begin{aligned}\overline{\partial}_\omega U(\omega_0) &= \lim_{h\to 0} \frac{1}{h}\,(U(\omega_0 + h) - U(\omega_0)) \\ &= \left(\lim_{h\to 0} \frac{u_{ij}(\omega_0 + h) - u_{ij}(\omega_0)}{h}\right)_{i,j} = \left(\partial_\omega u_{ij}(\omega_0)\right)_{i,j}.\end{aligned}$$

In this way higher-order partial differential operators, such as the Laplacian, or other more sophisticated operators, find their natural counterparts in the matrix-valued framework. It is worth mentioning that for the operators $\overline{\partial}_\omega$ a *product rule* holds:

$$\overline{\partial}_\omega(A(\omega_0) \cdot B(\omega_0)) = (\overline{\partial}_\omega A(\omega_0)) \cdot B(\omega_0)) + A(\omega_0) \cdot (\overline{\partial}_\omega B(\omega_0)).$$

(4) **Generalised gradient of a matrix field.** The definition of a generalised gradient is somewhat different from the one that might be expected when viewing a matrix as a tensor (of second order). The rules of differential geometry would tell us that derivatives are tensors of third order. Instead, we adopt a more operator-algebraic point of view: The matrices are self-adjoint operators that can be added, multiplied with a scalar, and concatenated. Thus, they form an algebra, and we aim at consequently replacing the field $\mathbb{R}$ by the algebra $\mathrm{Sym}_n(\mathbb{R})$ in the scalar, that is, $\mathbb{R}$-based formulation of PDEs used in image processing. Hence, the generalised gradient $\overline{\nabla}U(x)$ at a voxel x is regarded as an element of the module $(\mathrm{Sym}_n(\mathbb{R}))^d$ over $\mathrm{Sym}_n(\mathbb{R})$ in close analogy to the scalar setting where $\nabla u(x) \in \mathbb{R}^d$.
In the sequel we will call a mapping from R^d into $(\mathrm{Sym}_n(\mathbb{R}))^d$ a module field rather than a vector field.

(5) **Generalised Hessian.** The generalised Hessian of a field of symmetric matrices is a $nd \times nd$ block matrix with blocks of size $n \times n$. If the entries of each of the matrices of the matrix field are twice continuously differentiable then the Hessian is a symmetric matrix, just as its smaller counterpart derived from a multivariate scalar function.

(6) **Generalised divergence of the module field.** The generalisation of the divergence operator div acting on a vector field to an operator $\overline{\mathrm{div}}$ acting on a module field A is straightforward, and is in accordance with the formal relation

$$\overline{\Delta}U = \overline{\mathrm{div}}\,\overline{\nabla}U = \overline{\nabla}.\overline{\nabla}U$$

known in its scalar form from standard vector analysis.

(7) **Generalised length in $(\mathbf{Sym}_n(\mathbb{R}))^d$.** Considering the formal definition in the table the length of a element of a module field A is close at hand. Moreover, it results in a positive semidefinite matrix as the direct counterpart of a nonnegative real number as the length of a vector in $\mathbb{R}^d$. However, one cannot assume that this generalised length fulfils a triangle inequality with respect to the Loewner ordering.

(8) **Symmetrised product of symmetric matrices.** The product of two symmetric matrices $A, B \in \mathrm{Sym}_n(\mathbb{R})$ is not symmetric unless the matrices commute. Among the numerous options to define a symmetrised matrix product we focus on two specific ones: The first is inspired from pre-conditioning of symmetric linear equation systems.

$$A \bullet_P B = A^{\frac{1}{2}} B A^{\frac{1}{2}} \quad \text{for } A \in \mathrm{Sym}_n^+(\mathbb{R}),\ B \in \mathrm{Sym}_n(\mathbb{R}). \tag{5}$$

The following short list of properties is easily verified: it is neither associative, nor commutative, and distributive only in the second argument. However, if A is non-singular, the so-called *signature* $s = (s_+, s_-, s_0)$ of B is preserved, where s_+, s_-, and s_0, stand for the number of positive, negative, and vanishing eigenvalues of B, respectively. This implies in particular that the positive definiteness of B is preserved. A multiplication rule for the determinant holds,

$$\det(A \bullet_P B) = \det(A) \cdot \det(B).$$

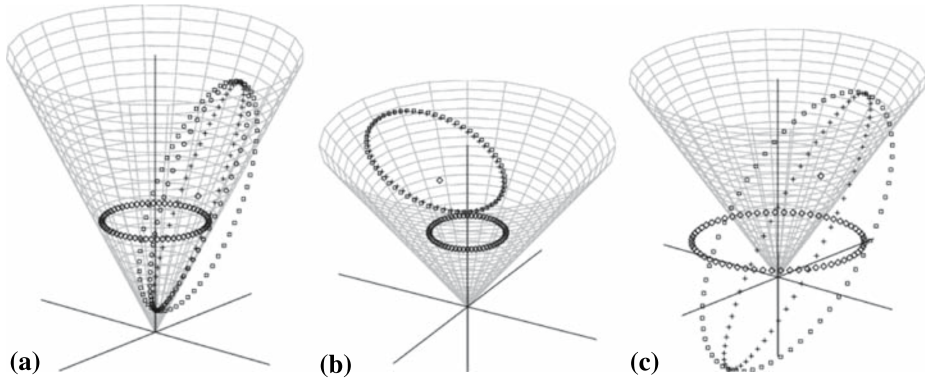

Fig. 1. The cone of all positive semidefinite 2×2-matrices is displayed in all three pictures. Each matrix of a set of positive semidefinite-matrices $\{A_i\}$ (*ring of black diamonds*) is multiplied by a fixed matrix B (*single black diamond*) utilising the Jordan product $A_i \bullet_J B$ (*line of boxes*), the preconditioning product $A_i \bullet_P B$ (*line of crosses*) and, if applicable, the logarithmic product (*line of circles*). (**a**) *Left* the matrices A_i, B are positive definite but have one small eigenvalue. The *boxes* indicate the matrices produced by the Jordan-product, they lie outside the cone. Hence $\bullet_J$ does not preserve positive semidefiniteness. As expected the preconditioning product (*crosses*) and the logarithmic product (*circles*) preserve positive semidefiniteness. (**b**) *Middle* if the two eigenvalues of each matrix A_i, B are positive and comparable in magnitude (corresponding points are in the vicinity of the center axis of the cone), the three types of products are very similar to each other. (**c**) *Right* The products $\bullet_J$ and $\bullet_P$ are produce quite different results if the one of the matrices multiplied is indefinite. Note that the logarithmic product is not defined in this case

Furthermore, for commuting matrices A, B we have $A \bullet_P B = A \cdot B$. Note that the first argument has to be positive semidefinite.

The second choice is well-known from algebra and called *Jordan product*:

$$A \bullet_J B = \frac{1}{2}(AB + BA) \qquad \text{for } A, B \in \mathrm{Sym}_n(\mathbb{R}). \tag{6}$$

This product is commutative and distributive but not associative. It is one half of the anti-commutator of A and B, but due to its additive structure no determinant product rule holds. Most important, it does not preserve the positive semidefinitness of its arguments. Again, for commuting A and B we have $A \bullet_J B = A \cdot B$.

It should be mentioned that the logarithmic multiplication introduced in [1] and given by $A \bullet_L B := \exp(\log(A) + \log(B))$ is defined only for positive definite matrices. However, the matrix-valued Perona–Malik diffusion proposed here requires the multiplication to be able to cope with at least one factor matrix being indefinite. Furthermore matrix fields that are not necessarily positive semidefinite should also be within the reach of our PDE-based filtering. Hence the logarithmic multiplication is not suitable for our purpose.

For a better comparison of the products we represent symmetric 2×2-matrices by points in $\mathbb{R}^3$ via the mapping [7]

$$\begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix} \longleftrightarrow \frac{1}{\sqrt{2}}(2\beta, \gamma - \alpha, \gamma + \alpha).$$

The set $\mathrm{Sym}_2^+(\mathbb{R})$ of positive semidefinite matrices then appears as a cone in $\mathbb{R}^3$, the cone corresponding to the Loewner ordering, see Fig. 1, where a comparison of the products is displayed: a set of matrices $\{A_i : i \in I\}$ with constant trace is multiplied by a single matrix utilising the Jordan-, the preconditioning, and, as long as the matrices are positive definite, the logarithmic product.

3. Diffusion equations for matrix fields

3.1 Matrix-valued linear diffusion

The linear diffusion equation $\partial_t u = \sum_{i=1}^d \partial_{x_i} \partial_{x_i} u = \sum_{i=1}^d \partial_{x_i x_i} u = \overline{\Delta} u$ on $\mathbb{R}^d \times [0, \infty[$ is directly extended to the matrix-valued setting:

$$\overline{\partial}_t U = \sum_{i=1}^{d} \overline{\partial}_{x_i} \overline{\partial}_{x_i} U = \sum_{i=1}^{d} \overline{\partial}_{x_i x_i} U = \overline{\Delta} U \tag{7}$$

with initial condition $U(x, 0) = F(x)$. The diffusion process described by this equation acts on each of the components of the matrix independently. It is not immediately clear that positive (semi-)definiteness of the initial matrix field F is indeed bequeathed to U for all times. Let us denote the i-th real eigenvalue of U, resp., F, as λ_i, resp., λ_i^F, numbered according to decreasing value.

Proposition: *The following inequality holds for all* $(x, t) \in \mathbb{R}^d \times [0, \infty[$:

$$\sup_x \lambda_1^F(x) \geq \lambda_i(x, t) \geq \inf_x \lambda_n^F(x).$$

Especially the positive (semi)definiteness of the initial field F is preserved in U.

Proof: We infer from the linearity of the differential operators $\overline{\partial}_t$ and $\overline{\Delta}$ that for any fixed unit vector $w \in \mathbb{R}^n$ the scalar diffusion equation

$$\partial_t \langle w, Uw \rangle = \Delta \langle w, Uw \rangle$$

holds, with initial condition $\langle w, U(x, 0)w \rangle = \langle w, F(x)w \rangle$. Hence, the Rayleigh coefficient $\langle w, U(x, t)w \rangle$ is a scalar function obeying a max-min-principle leading to the estimates

$$\sup_x \lambda_1^F(x) \geq \sup_x \langle w, F(x)w \rangle \geq \langle w, U(x, t)w \rangle \geq \inf_x \langle w, F(x)w \rangle \geq \inf_x \lambda_n^F(x)$$

valid for all $(x, t) \in \mathbb{R} \times [0, \infty[$ and unit vectors w. Choosing w as the eigenvector corresponding to the eigenvalue $\lambda_i(x, t)$ we can ensure the equality

$$\langle w, U(x, t)w \rangle = \lambda_i(x, t)$$

which proves the claim. □

3.2 Matrix-valued Perona–Malik diffusion equations

The scalar Perona–Malik diffusion Eq. (1) requires the multiplication of the components of a vector (namely ∇u) with a scalar (namely $g(|\nabla u|^2)$). In the matrix-valued setting the components of $\overline{\nabla} U$, that is, $\overline{\partial}_{x_i} U$, $i = 1, \ldots, d$, its generalised length $|\overline{\nabla} U|_2 =: |\overline{\nabla} U|$ and hence $g(|\overline{\nabla} U|^2)$ are symmetric matrices. We opted for two possibilities: The Jordan product $\bullet_J$ and the preconditioning product $\bullet_P$ as defined in Eqs. (6) and (5), respectively.

With these definitions we are now in the position to state the matrix-valued counterpart of the Perona–Malik PDE Eq. (1). It is given by

$$\overline{\partial}_t U = \overline{\operatorname{div}}\,(g(|\overline{\nabla} U|^2) \bullet \overline{\nabla} U) \tag{8}$$

which becomes manifest in the following two versions:

$$\overline{\partial}_t U = \sum_{i=1}^{d} \overline{\partial}_{x_i} \left(\sqrt{g((\overline{\nabla} U)^2)} \cdot (\overline{\partial}_{x_i} U) \cdot \sqrt{g(|\overline{\nabla} U|^2)} \right), \tag{9}$$

$$\overline{\partial}_t U = \sum_{i=1}^{d} \overline{\partial}_{x_i} \, (g((\overline{\nabla} U)^2) \cdot (\overline{\partial}_{x_i} U) + (\overline{\partial}_{x_i} U) \cdot g(|\overline{\nabla} U|^2)), \tag{10}$$

depending on the usage of the preconditioning Eq. (9) or the Jordan Eq. (10) product.

3.3 Enhancement properties/diffusion properties

In this section we will show that the matrix-valued Perona–Malik diffusion process can be expected to have the same properties as their scalar counterparts. This is an important confirmation of the validity of the proposed generic approach to matrix-valued PDEs. We restrict ourselves for the moment to the case of one spatial dimension ($d = 1$): $U : \mathbb{R} \longrightarrow \mathrm{Sym}_n(\mathbb{R})$, that is, to matrix-valued signals since then simplifications occur. Only one spatial derivative appears and the expressions containing the matrix $\overline{\partial}_x U$ commute. Hence, in those expressions the symmetric multiplication "$\bullet$" collapses to "$\cdot$", facilitating the analysis. The equation for the matrix-valued Perona–Malik diffusion in one space dimension simplifies to

$$\overline{\partial}_t U = \overline{\partial}_x (g((\overline{\partial}_x U)^2) \cdot \overline{\partial}_x U).$$

However, even in this simplified setting matrix-valued data exhibit directional (through eigenvectors) as well as shape information (through eigenvalues) which allows for the appearance of new phenomena.

The partial derivative $\overline{\partial}_x$ of a signal U of symmetric matrices results again in symmetric matrices, $\overline{\partial}_x U(x) \in \mathrm{Sym}_n(\mathbb{R})$. Hence

$$\overline{\partial}_x U(x) = \tilde{V}^\top(x) \mathrm{diag}(\tilde{\lambda}_i(x)) \tilde{V}(x)$$

with $\tilde{V}(x) \in O(n)$ for all $x \in \Omega$. We observe that $g((\overline{\partial}_x U)^2)$ is also diagonalised by $\tilde{V}$, and it follows that

$$g((\overline{\partial}_x U)^2) \cdot \overline{\partial}_x U = \tilde{V}^\top \mathrm{diag}(g(\tilde{\lambda_i}^2) \cdot \tilde{\lambda_i})\, \tilde{V}\,.$$

In allusion to the analysis of the Perona–Malik equation in [29] we introduce a flux function Φ by

$$\Phi(s) := s \cdot g(s^2)$$

which gives $\frac{\mathrm{d}\Phi}{\mathrm{d}s}(s) = \Phi'(s) = 2s^2 g'(s^2) + g(s^2)$ at least for $s > 0$. The product rule for matrix-valued functions then yields, if we suppress the explicit dependence of $\tilde{V}$ and $\tilde{\lambda}_i$ on x notationally:

$$\begin{aligned}
&\overline{\partial}_x (g((\overline{\partial}_x U)^2) \cdot \overline{\partial}_x U)\\
&= \overline{\partial}_x \tilde{V} \mathrm{diag}(g(\tilde{\lambda_i}^2) \cdot \tilde{\lambda_i})\, \tilde{V}^\top + \tilde{V} \mathrm{diag}(g(\tilde{\lambda_i}^2) \cdot \tilde{\lambda_i})\, \overline{\partial}_x \tilde{V}^\top + \tilde{V} \mathrm{diag}(\partial_x [g(\tilde{\lambda_i}^2) \cdot \tilde{\lambda_i}])\, \tilde{V}^\top\\
&= (\overline{\partial} \tilde{V}^\top, \tilde{V}^\top, \tilde{V}^\top) \underbrace{\begin{pmatrix} \mathrm{diag}(g(\tilde{\lambda_i}^2)\, \tilde{\lambda_i}) & 0 & 0 \\ 0 & \mathrm{diag}(g(\tilde{\lambda_i}^2)\, \tilde{\lambda_i}) & 0 \\ 0 & 0 & \mathrm{diag}(\Phi'(\tilde{\lambda_i})\, \partial_x \tilde{\lambda_i}) \end{pmatrix}}_{=:M} \begin{pmatrix} \tilde{V} \\ \overline{\partial} \tilde{V} \\ \tilde{V} \end{pmatrix}\\
&= (\overline{\partial} \tilde{V}^\top, \tilde{V}^\top, \tilde{V}^\top) \mathrm{diag}(g(\tilde{\lambda_i}^2); g(\tilde{\lambda_i}^2); \Phi'(\tilde{\lambda_i}))\, \mathrm{diag}(\tilde{\lambda_i}; \tilde{\lambda_i}; \partial_x \tilde{\lambda_i}) \begin{pmatrix} \tilde{V} \\ \overline{\partial} \tilde{V} \\ \tilde{V} \end{pmatrix}, \qquad (11)
\end{aligned}$$

where the 3×3-block-matrix M has been decomposed into a product of the block-matrices

$$\mathrm{diag}(h(\tilde{\lambda_i}^2); h(\tilde{\lambda_i}^2); \Phi'(\tilde{\lambda_i})) := \begin{pmatrix} \mathrm{diag}(h(\tilde{\lambda_i}^2)) & 0 & 0 \\ 0 & \mathrm{diag}(h(\tilde{\lambda_i}^2)) & 0 \\ 0 & 0 & \mathrm{diag}(\Phi'(\tilde{\lambda_i})) \end{pmatrix},$$

$$\mathrm{diag}(\tilde{\lambda_i}; \tilde{\lambda_i}; \partial_x \tilde{\lambda_i}) := \begin{pmatrix} \mathrm{diag}(\tilde{\lambda_i}) & 0 & 0 \\ 0 & \mathrm{diag}(\tilde{\lambda_i}) & 0 \\ 0 & 0 & \mathrm{diag}(\partial_x \tilde{\lambda_i}) \end{pmatrix}.$$

Hence, the matrix-valued version of the Perona–Malik diffusion equation takes on the form

$$\overline{\partial}_t U = (\overline{\partial}_x \tilde{V}^\top, \tilde{V}^\top, \tilde{V}^\top) \mathrm{diag}(g(\tilde{\lambda_i}^2); g(\tilde{\lambda_i}^2); \Phi'(\tilde{\lambda_i})) \mathrm{diag}(\tilde{\lambda_i}; \tilde{\lambda_i}; \partial_x \tilde{\lambda_i}) \begin{pmatrix} \tilde{V} \\ \overline{\partial}_x \tilde{V} \\ \tilde{V} \end{pmatrix}, \qquad (12)$$

while the matrix-valued linear diffusion equation can be cast into the form

$$\overline{\partial}_t U = (\overline{\partial}_x \tilde{V}^\top, \tilde{V}^\top, \tilde{V}^\top) \mathrm{diag}(\tilde{\lambda_i}; \tilde{\lambda_i}; \partial_x \tilde{\lambda_i}) \begin{pmatrix} \tilde{V} \\ \overline{\partial}_x \tilde{V} \\ \tilde{V} \end{pmatrix}. \qquad (13)$$

Juxtaposing this pairing with their *scalar* versions rewritten in this fashion, the Perona–Malik equation turns out to be

$$\begin{aligned}\partial_t u &= \Phi'(\partial_x u) \cdot \partial_{xx} u \\ &= (\partial_x 1, 1, 1)\,\mathrm{diag}(g((\partial_x u)^2), g((\partial_x u)^2), \Phi'(\partial_x u))\mathrm{diag}\,(\partial_x u, \partial_x u, \partial_x \partial_x u) \begin{pmatrix} 1 \\ \partial_x 1 \\ 1 \end{pmatrix},\end{aligned}$$

whereas the standard scalar linear diffusion equation in 1D reads

$$\partial_t u = \partial_{xx} u = (\partial_x 1, 1, 1)\,\mathrm{diag}\,(\partial_x u, \partial_x u, \partial_x \partial_x u) \begin{pmatrix} 1 \\ \partial_x 1 \\ 1 \end{pmatrix}.$$

What distinguishes Perona–Malik diffusion from the linear one is the multiplicative factor $\mathrm{diag}\big(g((\partial_x u)^2), g((\partial_x u)^2), \Phi'(\partial_x u)\big)$ in the scalar case as opposed to $\mathrm{diag}(g(\tilde{\lambda}_i^2); g(\tilde{\lambda}_i^2); \Phi'(\tilde{\lambda}_i))$ in the matrix-valued case.

This comparison brings to light the complete analogy between the scalar setting and the matrix-valued framework as outlined above, down to the correspondence

$$(\overline{\partial}_x \tilde{V}^\top, \tilde{V}^\top, \tilde{V}^\top)(\tilde{V}, \overline{\partial}_x \tilde{V}, \tilde{V})^\top = \overline{\partial}_x(\tilde{V}^\top \tilde{V}) + \tilde{V}^\top \tilde{V} = 0 + I$$

and its scalar counterpart $(\partial_x 1, 1, 1)\,(1, \partial_x 1, 1) = 0 + 1$, with ± 1 being the only two orthogonal 1×1 matrices. In the scalar setting the sign of $\Phi'(\partial_x u)$ decides on the direction of the diffusion: a negative sign if $|\partial_x u| > \lambda$ results in a *backward* diffusion whereas small gradients $|\partial_x u| < \lambda$ entail a positive sign and hence a *forward* diffusion. The role of $\Phi'(\partial_x u)$ in the scalar setting is played in the matrix case by the $n \times n$-matrix $\mathrm{diag}(\Phi'(\tilde{\lambda}_i))$ and we infer that forward diffusion occurs in those eigen-directions where the corresponding eigenvalue λ_i satisfies $\lambda_i < \lambda$, and backward diffusion in those eigen-directions where $\lambda_j > \lambda$. It is remarkable that the difference between linear and Perona–Malik diffusion for both data types is made by multiplicative factors which correspond to each other perfectly: $\mathrm{diag}\big(g((\partial_x u)^2), g((\partial_x u)^2), \Phi'(\partial_x u)\big)$ in the scalar case, and $\mathrm{diag}(g(\tilde{\lambda}_i^2); g(\tilde{\lambda}_i^2); \Phi'(\tilde{\lambda}_i))$ in the matrix-valued setting.

Remarks:

(1) *Considering the PDEs Eqs. (12) and (13) for matrix-valued Perona–Malik and linear diffusion suggests that they inherit the smoothing and enhancing properties of their scalar counterparts. So we may expect from Perona–Malik-type matrix-valued diffusion good denoising qualities combined with edge-preserving features.*

(2) *However, the matrix-valued data allow for a new phenomenon: Unlike in the scalar setting, a matrix carries* directional *information conveyed through the eigenvectors as well as* shape *information mediated via eigenvalues. The evolution process described in Eq. (11) displays a coupling between shape and directional information by virtue of the simultaneous occurrence of terms containing $\overline{\partial}_x \tilde{V}(x)$ and $\partial_x \tilde{\lambda}(x)$. Clearly there is no equivalent for this in the scalar setting.*

4. Matrix-valued numerical solution schemes

In the previous sections the guideline to infer matrix-valued PDEs from scalar ones was, roughly speaking, analogy by making a transition from the real field $\mathbb{R}$ to the vector space $\mathrm{Sym}_n(\mathbb{R})$ endowed with some "symmetric" product "$\bullet$". We follow this very guideline also in the issue of numerical schemes for matrix-valued PDEs. For the sake of brevity we restrict ourselves to the numerical scheme for two space dimensions ($d = 2$). The necessary extensions to dimensions $d \geq 3$ are immediate. A possible scheme for the scalar Perona–Malik diffusion can be cast into the form

$$\begin{aligned}\frac{\mathrm{d}u(i,j)}{\mathrm{d}t} &\\ = \frac{1}{h_1}&\left(g\left(i+\frac{1}{2},j\right)\cdot\frac{u(i+1,j)-u(i,j)}{h_1} - g\left(i-\frac{1}{2},j\right)\cdot\frac{u(i,j)-u(i-1,j)}{h_1}\right)\\ +\frac{1}{h_2}&\left(g\left(i,j+\frac{1}{2}\right)\cdot\frac{u(i,j+1)-u(i,j)}{h_2} - g\left(i,j-\frac{1}{2}\right)\cdot\frac{u(i,j)-u(i,j-1)}{h_2}\right),\end{aligned}$$

where $g(i,j)$ and $u(i,j)$ are samples of the diffusivity g and of u at pixel $(i\,h_1, j\,h_2)$ and, for example, $g(i \pm \frac{1}{2}, j) := \frac{g(i\pm1,j)+g(i,j)}{2}$. In the numerical implementation we approximate the time derivative by one-sided finite difference and the set $h_1 = h_2 = 1$. According to our preparations in Sects. 2 and 3 its matrix-valued extension to solve the Perona–Malik diffusion equation in the matrix setting reads

$$\begin{aligned}\frac{\mathrm{d}U(i,j)}{\mathrm{d}t} &\\ = \frac{1}{h_1}&\left(G\left(i+\frac{1}{2},j\right)\bullet\frac{U(i+1,j)-U(i,j)}{h_1} - G\left(i-\frac{1}{2},j\right)\bullet\frac{U(i,j)-U(i-1,j)}{h_1}\right)\\ +\frac{1}{h_2}&\left(G\left(i,j+\frac{1}{2}\right)\bullet\frac{U(i,j+1)-U(i,j)}{h_2} - G\left(i,j-\frac{1}{2}\right)\bullet\frac{U(i,j)-U(i,j-1)}{h_2}\right).\end{aligned}$$

The arithmetic mean $G(i \pm \frac{1}{2}, j) := \frac{G(i\pm1,j)+G(i,j)}{2} \in \mathrm{Sym}_n(\mathbb{R})$ approximates the diffusivity $G(|\overline{\nabla}U|^2)$ between the pixels $(i \pm 1, j)$ and (i, j).

5. Experiments

In our experiments we used both artificial and real-world data. Figure 2 shows a 2D artificial data set consisting of a 16×16 field of matrices. The data are represented as ellipsoids via the level sets of the quadratic form $\{x^\top A^{-2}x = const. : x \in \mathbb{R}^3\}$ associated with a matrix $A \in \mathrm{Sym}^+(3)$. By using A^{-2} the length of the semi-axes of the ellipsoid correspond directly with the three eigenvalues of the matrix. To demonstrate the denoising capabilities, we have added random positive definite matrices to the data. The eigenvalues of this noise were obtained by choosing Gaussian-distributed numbers with standard deviation $\sigma = 1{,}000.0$ and taking the absolute value for positive definiteness. The high standard deviation can be explained by the

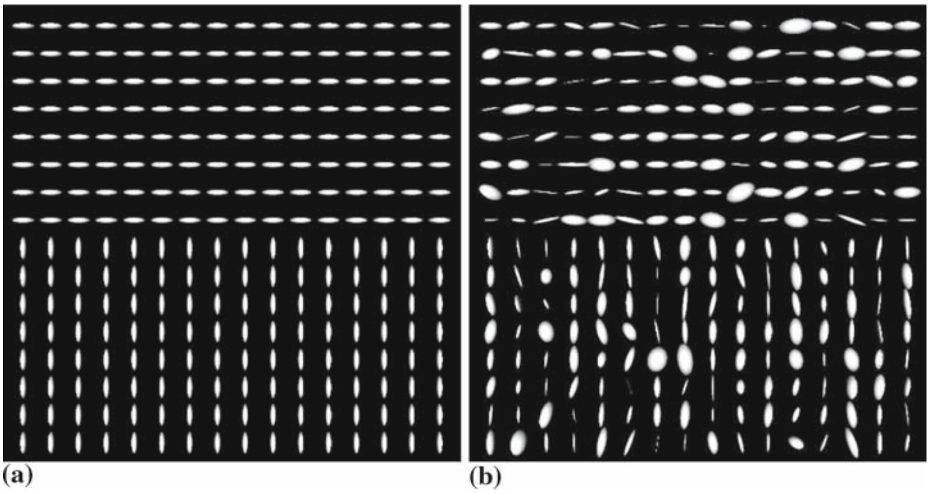

Fig. 2. Artificial test data. (**a**) *Left* original data set, 16 × 16 pixels. (**b**) *Right* data (**a**) with additive noise of average Frobenius norm 1,430 per pixel

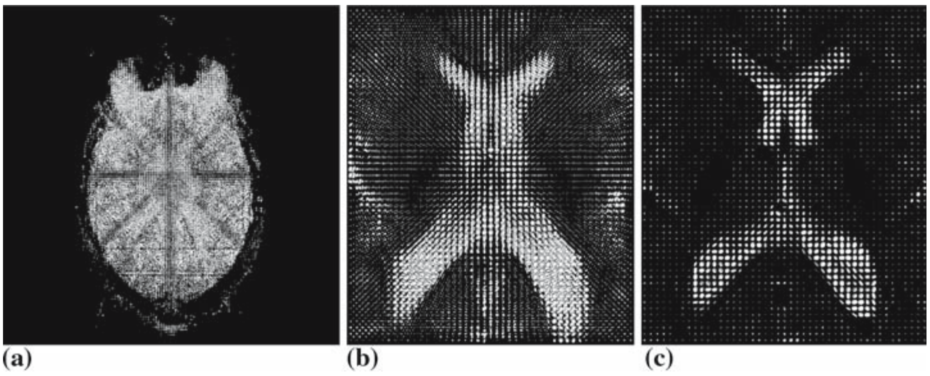

Fig. 3. Real-world data set. (**a**) *Left* original data set, 128 × 128 × 30 voxels. (**b**) *Middle* 3D section of (**a**) with 45 × 53 × 5 voxels. (**c**) *Right* 2D section of (**a**) with 45 × 53 × 1 pixels

fact that in real-world data the typical eigenvalues are in the order of magnitude of 1,000. The eigenvectors of the artificial noise result in choosing three uniformly distributed angles and rotating the matrix by these angles around the coordinate axes. The resulting data is shown in Fig. 2.

Besides the artificial data, we also use a real-world 3D DT-MRI data set of a human head consisting of a $128 \times 128 \times 30$-field of positive definite matrices, see Fig. 3. We compare the results $\tilde{U}$ and $\tilde{\tilde{U}}$ of the filtering processes differing in the selection of product or diffusivity function by considering the matrix field $\delta(x)$ of absolute differences in the matrix sense (Sect. 2, Comment 1)

$$\delta(x) := \left|\tilde{U}(x) - \tilde{\tilde{U}}(x)\right|.$$

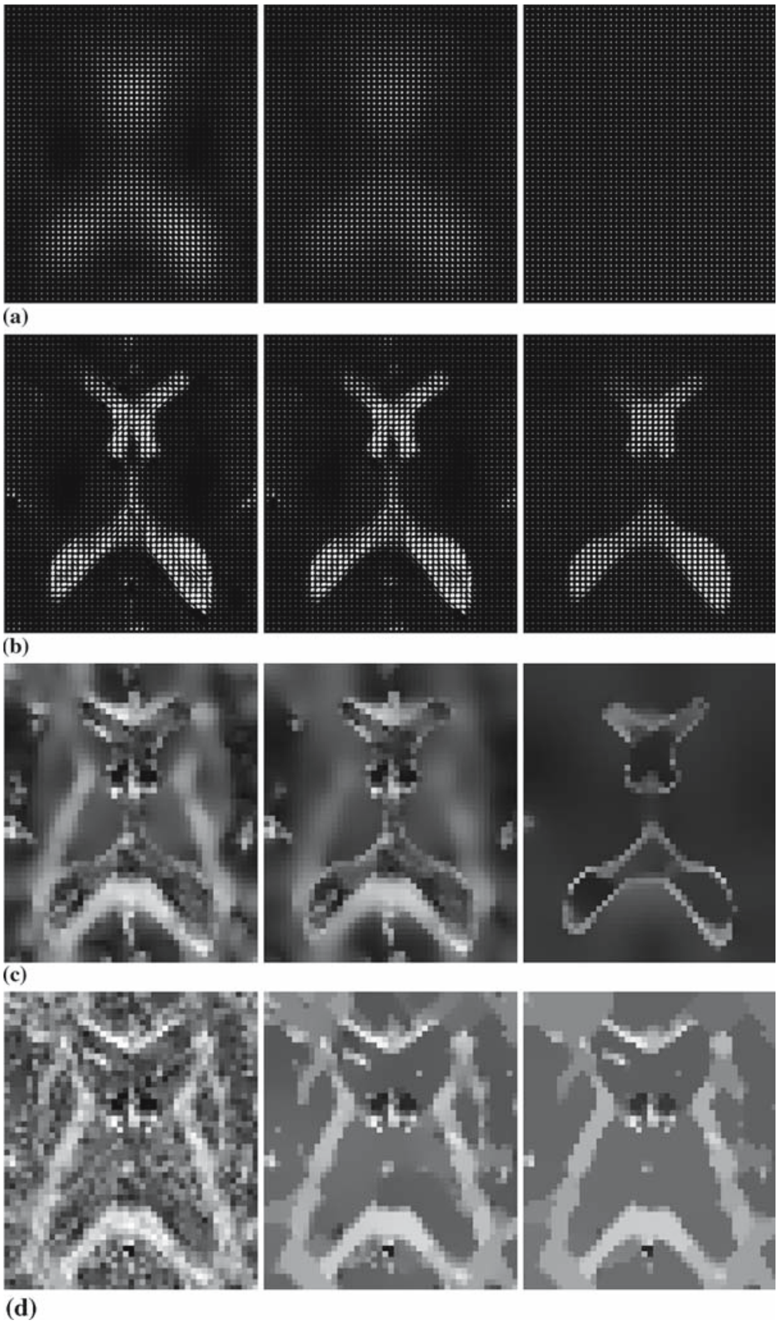

Fig. 4. Image simplification properties with 2D real DT-MRI data. (**a**) *Top row* linear matrix-valued diffusion at diffusion times $t = 5, 10, 100$. (**b**) *Second row* matrix-valued Perona–Malik diffusion with classical diffusivity g_λ and $\lambda = 100$ at diffusion times $t = 5, 10, 100$ using the *Jordan product*. (**c**) *Third row* corresponding evolution of fractional anisotropy under matrix-valued Perona–Malik diffusion. (**d**) *Bottom row* scalar Perona–Malik diffusion of the grey value image displaying the fractional anisotropy of the original data, $\lambda = 1, t = 5, 10, 100$

In Fig. 4 below, we compare the results of matrix-valued linear diffusion and Perona–Malik diffusion. The edge preserving quality of the Perona–Malik filtering is observable as can be expected following the discussion in Sect. 3.3. Figure 4 also makes clear the importance of filtering the matrix data directly: the so called fractional anisotropy (FA), a scalar quantity important in medical imaging [3], is defined via the eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$ of the matrix at a voxel x by

$$\mathrm{FA}(x) := \sqrt{\frac{(\lambda_1-\tilde{\lambda})^2+(\lambda_2-\tilde{\lambda})^2+(\lambda_3-\tilde{\lambda})^2}{\lambda_1^2+\lambda_2^2+\lambda_3^2}},$$

with the average $\tilde{\lambda} = \frac{1}{3}(\lambda_1+\lambda_2+\lambda_3)$. Obtaining the FA image from the *filtered* images gives a higher quality result than calculating the scalar FA image from the original matrix field and then filtering this grey value image with the scalar Perona–Malik process. It is clearly visible that for larger diffusion times, the FA of the *filtered* image is getting smaller, while filtering the FA directly converges towards the average FA in the initial data.

In Fig. 5, the influence of the choice of multiplication, Jordan or preconditioning product, on the denoising capabilities of Perona–Malik filtering with classical diffusivity function g_λ is accented. In both instances the noise is removed while the edge is preserved, in very good agreement with the well-known denoising properties of their scalar predecessors. The influence of the type of multiplication is not very prominent as the high magnification factor ($\times 15$) in the difference field confirms.

A more detailed experimental analysis of the effect of the diffusivity function during the evolution process is depicted in Fig. 6 where the Perona–Malik and the exponentially decaying Weickert diffusivity functions g_λ, $g_{\lambda,4}$ are employed. The difference matrix field emphasises the influences of the choice of the diffusivity function on the evolution process. These influences are magnified for visualisation purposes. Nevertheless, if the PDE methods should be used as pre-processing step in a larger application framework, it might be worthwhile to quantify the differences more precisely.

Finally, we investigate the behaviour of our filtering methods for negative definite or even indefinite matrices. For this purpose we have subtracted a factor times the identity matrix from all matrices in the noisy artificial data set shown in Fig. 2. To obtain indefinite data, we have chosen the factor as the mean between largest and smallest eigenvalue in the data set. Negative definite data has been obtained by subtracting the largest eigenvalues appearing in the whole data set. After filtering, the same values have been added again to the results to visualise them. Figure 7 shows that the filters are invariant under the addition of scaled identity matrices. They have exactly the same behaviour independent of the definiteness properties of the initial data.

6. Conclusion

In this article we have presented a novel and generic framework for the extension of the Perona–Malik PDE to *not necessarily* positive definite symmetric matrix fields in

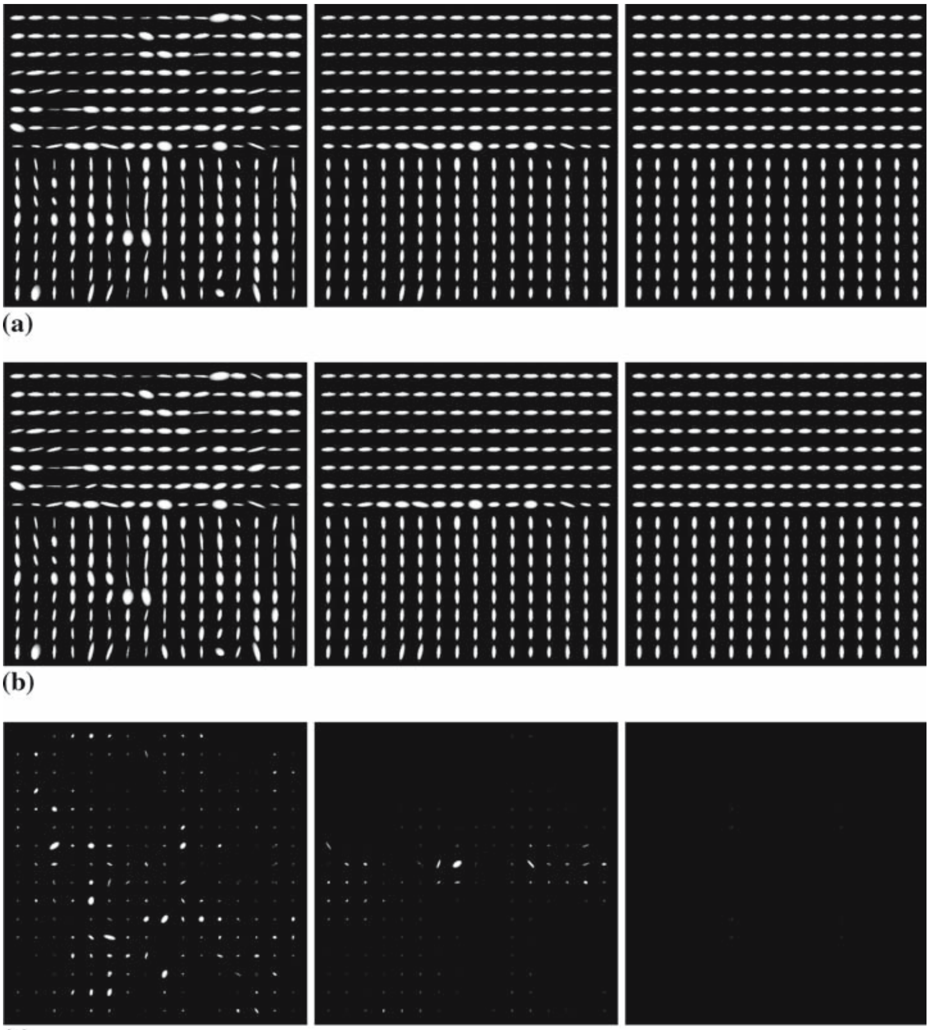

Fig. 5. Comparison between Jordan and preconditioning product with artificial noisy data. (**a**) *Top row* Perona–Malik diffusion with classical diffusivity ($\lambda = 100$) at diffusion times $t = 5, 20, 100$ using the *Jordan product*. (**b**) *Middle row* the same, except using the *preconditioning product*. (**c**) *Bottom row* the absolute value of the difference of the two filtered matrix fields at times $t = 5, 20, 100$ magnified by a factor 15

any spatial dimension. The approach assumes an operator-algebraic point of view by emphasising the fact that symmetric matrices are finite-dimensional instances of self-adjoint Hilbert space operators. Two reasonable types of a symmetric multiplication for symmetric matrices have been considered ensuring appropriate channel interaction. The different products cause only a slightly different evolution in the associated Perona–Malik process. Also different types of diffusivities steering the diffusion have been considered, the classical one with polynomial decay, one with exponential decay, and finally a 1-0-diffusivity. The influence of the choice of the

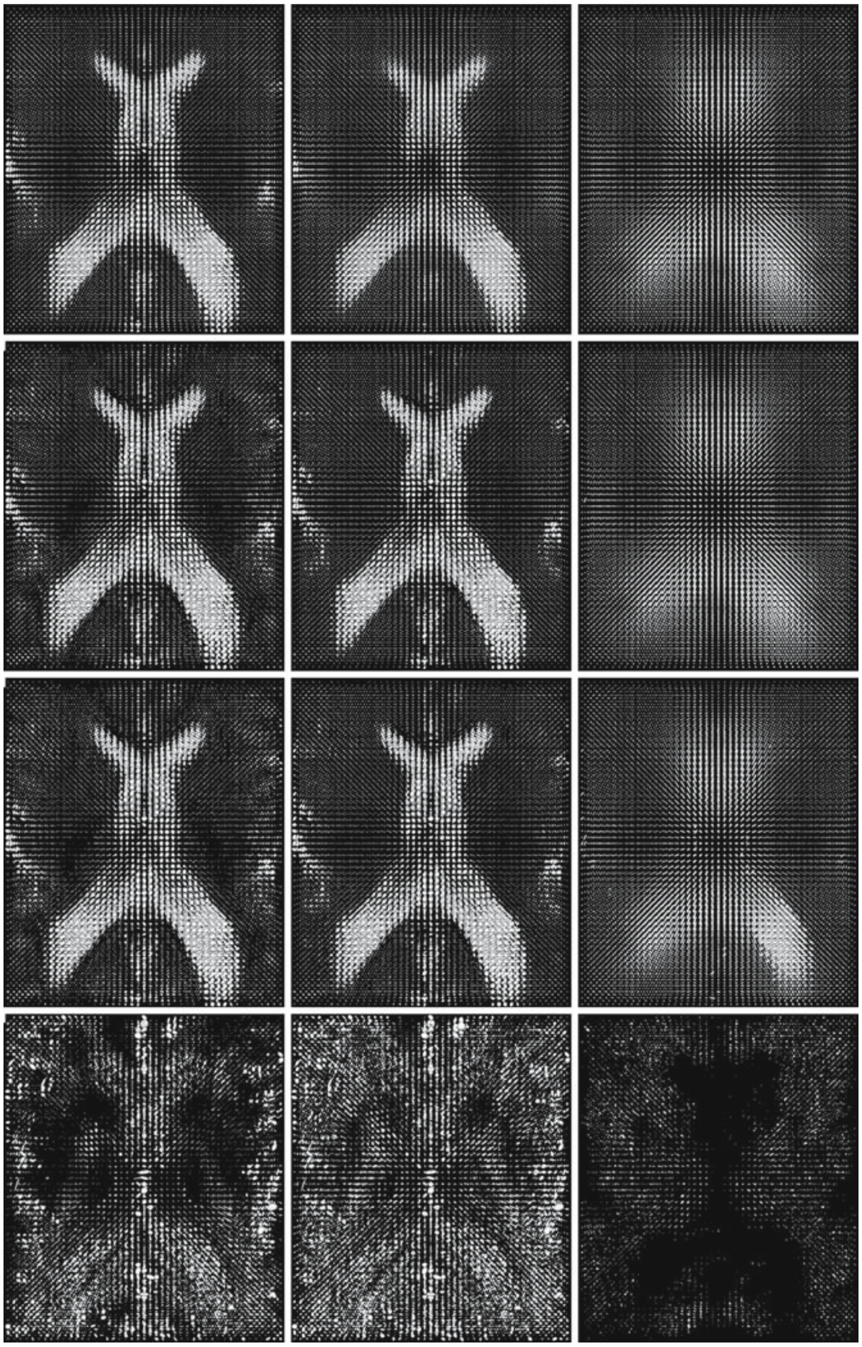

Fig. 6. Comparison between diffusivity functions with 3D real DT-MRI data, stopping time $t = 10$. *Top row* Perona–Malik diffusivity, $\lambda = 100, 200, 1000$. *Second row* Weickert diffusivity, $p = 4$, $\lambda = 100, 200, 1{,}000$. *Third row* 0–1 diffusivity, $g_{\lambda,\infty}$, $\lambda = 100, 200, 1{,}000$. *Bottom row* absolute difference between the results for $\lambda = 100, t = 10$, scaled by the factor 5. *Bottom left* Perona–Malik and Weickert diffusivities. *Bottom middle* Perona–Malik and $g_{\lambda,\infty}$. *Bottom right* Weickert and $g_{\lambda,\infty}$

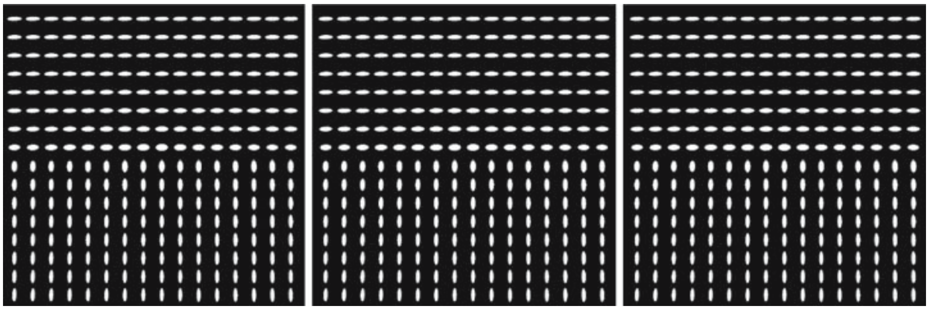

Fig. 7. Results of filtering of indefinite or negative definite matrices if the evolution process is forced to "detour" through indefinite data (*middle image*) and negative definite data (*right image*)

diffusivity on the evolution of the Perona–Malik diffusion is clearly noticeable as experiments confirm.

Experiments on positive semidefinite DT-MRI and on indefinite/negative definite artificial data also illustrate that the matrix-valued Perona–Malik diffusion inherits desirable characteristic properties of their scalar valued predecessors, e.g., very good denoising capabilities combined with feature preserving qualities. In future work we will investigate how this framework can help to extend other scalar PDEs and more sophisticated numerical solution concepts in image processing to the matrix-valued setting.

Acknowledgements

DT-MRI data set is a courtesy of Anna Vilanova i Bartoli, Eindhoven University of Technology.

References

[1] Arsigny, V., Fillard, P., Pennec, X., Ayache, N.: Fast and simple calculus on tensors in the log-Euclidean framework. In: (Duncan, J., Gerig, G., eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005, part I, pp. 115–122. LNCS, vol. 3749. Springer, Heidelberg (2005)

[2] Aubert, G., Kornprobst, P.: Mathematical problems in image processing: partial differential equations and the calculus of variations. Applied Mathematical Sciences, vol. 147. Springer, New York (2002)

[3] Basser, P. J., Pierpaoli, C.: Microstructural and physiological features of tissues elucidated by quantitative-diffusion-tensor mri. J Magn Reson Series B **111**, 209–219 (1996)

[4] Bellettini, G., Novaga, M., Paolini, E.: Global solutions to the gradient flow equation of a nonconvex functional. SIAM J Math Anal **37**(5), 1657–1687 (2006)

[5] Brox, T., Weickert, J.: Nonlinear matrix diffusion for optic flow estimation. In: (Van Gool, L., ed.) Pattern recognition, pp. 446–453. Lecture Notes in Computer Science, vol. 2449. Springer, Berlin (2002)

[6] Brox, T., Weickert, J., Burgeth, B., Mrázek, P.: Nonlinear structure tensors. Image Vis Comput **24**(1), 41–55 (2006)

[7] Burgeth, B., Bruhn, A., Didas, S., Weickert, J., Welk, M.: Morphology for matrix-data: ordering versus PDE-based approach. Image Vis Comput **25**(4), 496–511 (2007)

[8] Burgeth, B., Didas, S., Florack, L., Weickert, J.: Singular PDEs for the processing of matrix-valued data. In: (Sgallari, F., Paragios, N., Murli, A., eds.) Scale space and variational methods in computer vision, pp. 556–567. Lecture Notes in Computer Science, vol. 4485. Springer, Berlin (2007)

[9] Catté, F., Lions, P.-L., Morel, J.-M., Coll, T.: Image selective smoothing and edge detection by nonlinear diffusion. SIAM J Numer Anal **32**, 1895–1909 (1992)

[10] Chambolle, A.: Partial differential equations and image processing. In: Proc. 1994 IEEE International Conf. on Image Processing, vol. 1, pp. 16–20. Austin, TX, November 1994. IEEE Computer Society Press (1994)

[11] Chefd'Hotel, C., Tschumperlé, D., Deriche, R., Faugeras, O.: Constrained flows of matrix-valued functions: application to diffusion tensor regularization. In: (Heyden, A., Sparr, G., Nielsen, M., Johansen, P., eds.) Computer Vision – ECCV 2002, pp. 251–265. Lecture Notes in Computer Science, vol. 2350. Springer, Berlin (2002)

[12] Esedoglu, S.: An analysis of the Perona-Malik scheme. Commun Pure Appl Math **54**, 1442–1487 (2001)

[13] Feddern, C., Weickert, J., Burgeth, B., Welk, M.: Curvature-driven PDE methods for matrix-valued images. Int J Comput Vis **69**(1), 91–103 (2006)

[14] Gerig, G., Kübler, O., Kikinis, R., Jolesz, F. A.: Nonlinear anisotropic filtering of MRI data. IEEE Trans Med Imag **11**, 221–232 (1992)

[15] Horn, R. A., Johnson, C. R.: Matrix analysis. Cambridge University Press, Cambridge (1990)

[16] Kawohl, B., Kutev, N.: Maximum and comparison principle for one-dimensional anisotropic diffusion. Math Ann **311**, 107–123 (1998)

[17] Kichenassamy, S.: The Perona–Malik paradox. SIAM J Appl Math **57**, 1343–1372 (1997)

[18] Morel, J.-M., Solimini, S.: Variational methods in image segmentation. Birkhäuser, Basel (1994)

[19] Mrázek, P., Navara, M.: Selection of optimal stopping time for nonlinear diffusion filtering. Int J Comput Vis **52**(2/3), 189–203 (2003)

[20] Perona, P., Malik, J.: Scale space and edge detection using anisotropic diffusion. In: Proc. IEEE Computer Society Workshop on Computer Vision, pp. 16–22. IEEE Computer Society Press, Miami Beach, FL, November (1987)

[21] Perona, P., Malik, J.: Scale space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal Mach Intel **12**, 629–639 (1990)

[22] Pierpaoli, C., Jezzard, P., Basser, P. J., Barnett, A., Di Chiro, G.: Diffusion tensor MR imaging of the human brain. Radiology **201**(3), 637–648 (1996)

[23] Rosati, M., Schiaffino, A.: Some remarks about Perona–Malik equation. Technical Report 4/1999, Instituto per le Applicazioni del Calcolo "Mauro Picone", Rome, Italy (1999)

[24] Schultz, T., Burgeth, B., Weickert, J.: Flexible segmentation and smoothing of DT-MRI fields through a customizable structure tensor. In: (Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Nefian, A. V., Gopi, M., Pascucci, V., Zara, J., Molineros, J., Theisel, H., Malzbender, T., eds.) Proc. Int. Symp. on Visual Computing, pp. 454–464. Lecture Notes in Computer Science, vol. 4291. Springer, Berlin (2006)

[25] Smolka, B.: Combined forward and backward anisotropic diffusion filtering of color images. In: (Van Gool, L., ed.) Pattern recognition, pp. 314–320. Lecture Notes in Computer Science, vol. 2449. Springer, Berlin (2002)

[26] Tschumperlé, D., Deriche, R.: Diffusion tensor regularization with constraints preservation. In: Proc. 2001 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. 948–953. IEEE Computer Society Press, Kauai, HI, December (2001)

[27] Weickert, J.: Scale-space properties of nonlinear diffusion filtering with a diffusion tensor. Technical Report 110, Laboratory of Technomathematics, University of Kaiserslautern, Germany, October (1994)

[28] Weickert, J.: Theoretical foundations of anisotropic diffusion in image processing. Computing (Suppl) **11**, 221–236 (1996)

[29] Weickert, J.: Anisotropic diffusion in image processing. Teubner, Stuttgart (1998)

[30] Weickert, J.: Coherence-enhancing diffusion of colour images. Image Vis Comput **17**(3–4), 199–210 (1999)

[31] Weickert, J., Benhamouda, B.: A semidiscrete nonlinear scale-space theory and its relation to the Perona–Malik paradox. In: (Solina, F., Kropatsch, W. G., Klette, R., Bajcsy, R., eds.) Advances in computer vision, pp. 1–10. Springer, Wien (1997)

[32] Weickert, J., Brox, T.: Diffusion and regularization of vector- and matrix-valued images. In: (Nashed, M. Z., Scherzer, O., eds.) Inverse problems, image analysis, and medical imaging, pp. 251–268. Contemporary Mathematics, vol. 313. AMS, Providence (2002)

[33] Weickert, J., Hagen, H. (eds.): Visualization and processing of tensor fields. Springer, Berlin (2006)

[34] Zhang, K.: Existence of infinitely many solutions for the one-dimensional Perona–Malik model. Calc Var **26**(2), 171–199 (2006)