Deflated preconditioned Conjugate Gradient methods for noise filtering of low-field MR images

Shan, Xiujie; van Gijzen, Martin B.

DOI
10.1016/j.cam.2021.113730

Publication date
2022

Document Version
Final published version

Published in
Journal of Computational and Applied Mathematics

Journal of Computational and Applied Mathematics 400 (2022) 113730

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Deflated preconditioned Conjugate Gradient methods for noise filtering of low-field MR images

Xiujie Shan [a,b,*], Martin B. van Gijzen [b]

[a] Harbin Institute of Technology, School of Mathematics, 150001, Harbin, China
[b] Delft University of Technology, Delft Institute of Applied Mathematics, 2628 CD, Delft, The Netherlands

ARTICLE INFO

Article history:
Received 9 December 2020
Received in revised form 7 May 2021

Keywords:
Low-field MRI
DPCG
PDE
Image denoising

ABSTRACT

We study efficient implicit methods to denoise low-field MR images using a nonlinear diffusion operator as a regularizer. This problem can be formulated as solving a nonlinear reaction–diffusion equation. After discretization, a lagged-diffusion approach is used which requires a linear system solve in every nonlinear iteration. The choice of diffusion model determines the denoising properties, but it also influences the conditioning of the linear systems. As a solution method, we use Conjugate Gradient (CG) in combination with a suitable preconditioner and deflation technique. We consider four different preconditioners in combination with subdomain deflation. We evaluate the methods for four commonly used denoising operators: standard Laplace operator, two Perona–Malik type operators, and the Total Variation (TV) operator. We show that a Discrete Cosine Transform (DCT) preconditioner works best for problems with a slowly varying diffusion coefficient, while Jacobi preconditioning with subdomain deflation works best for a strongly varying diffusion, as happens for the TV operator. This research is part of a larger effort that aims to provide low-cost MR imaging capabilities for low-resource settings. We have evaluated the algorithms on low-field MRI images using inexpensive commodity hardware. With a suitable preconditioner for the chosen diffusion model, we are able to limit the time to denoise three-dimensional images of more than 2 million pixels to less than 15 s, which is fast enough to be used in practice.

1. Introduction

1.1. The context of the research

Many people benefit from the availability of MRI scanners for medical diagnostic purposes. However, conventional MRI scanners are expensive and difficult to operate and maintain, and therefore are out of reach in many low- and middle-income countries. To provide affordable MRI systems, there has been an increase in research to develop low-cost MRI scanners [1,2]. We are involved in a project [3–6] that aims to design a low-field MRI device for imaging the head, primarily to aid in the treatment of hydrocephalus, a condition that affects many newborns in Africa. The system [4] developed by the Leiden University Medical Center (LUMC), uses a Halbach-array-based permanent magnet to eliminate the need for expensive super-conducting magnets that are typically used in conventional MRI systems. In the same way as for the

* Corresponding author at: Harbin Institute of Technology, School of Mathematics, 150001, Harbin, China.
E-mail address: xiujieshan@gmail.com (X. Shan).

https://doi.org/10.1016/j.cam.2021.113730
0377-0427/

traditional MRI systems, the inverse Fourier transform is applied to convert the k-space signal into a complex-valued image.

For conventional general-purpose MRI systems, also the image processing software, and its maintenance and improvement costs, are relatively high [1]. This cost can be reduced by using commodity hardware, i.e., standard laptop or desktop computers, and open-source software. This, however, provides constraints on the processing algorithms that we can use: they should require limited computational resources while still being efficient. Low-field MR images are typically noisy and blurred. The focus of this paper is therefore on the development of effective denoising and edge-preserving algorithms, that are suitable for commodity hardware.

1.2. Denoising models

The maximum a posterior (MAP) model for image denoising is formulated as, see, e.g., [7]

$$u = \underset{u}{\operatorname{argmin}}\, \mathcal{R}(u) + \mu \mathcal{F}(u, f).$$

in which $\mathcal{R}(u)$ is the regularization term associated with an image prior, which promotes certain regularity properties of the image, $\mathcal{F}(u, f)$ is the fidelity term which guarantees the difference between the denoised image u and the initial, noisy image f is not too big, such that the main features of the image f are preserved, and the fidelity parameter μ determines the trade-off between the two terms. The well-known denoising Total Variation (TV) model, first proposed by Rudin, Osher, and Fatemi [8], belongs to this class. The variational model can be minimized by solving the diffusion equation using a gradient descent flow method. Chen et al. [9] study a generalization of the TV functional with a variable exponent, which provides a model for image denoising, enhancement, and restoration.

The diffusion model in image processing interprets pixel intensities as a diffusion process in the image. Standard heat diffusion was the first model of this type used for image denoising. The disadvantage of the method is that it smooths out image edges and therefore results in a blurred image. In the early 90s, Perona and Malik [10] proposed a nonlinear diffusion model (PM model) for image processing. The magnitude of the gradient of the image is assumed to be a good indication of the location of the image edge. By replacing the constant-diffusion coefficient by a gradient-based coefficient, the model is able to preserve edges while removing noise. However, the PM model suffers from ill-posedness of the solution. To overcome the ill-posedness of the PM model, Alvarez, Lions and Morel [11] have introduced a regularization PM model that makes the problem well-posed. A general class of diffusivities for image denoising is proposed in [12] and the authors further utilize a numerical method to avoid piecewise constant structures in the numerical solution, which may happen when evolving the TV and other diffusion models.

Several papers have appeared on denoising of three-dimensional MR images. Considering the original anisotropic diffusion PM model, Gerig et al. [13] proposed a PDE-based filtering method. Golshan et al. [14] presented an LMMSE-based method for denoising of three-dimensional images. In [15], the authors applied a three-dimensional anisotropic diffusion process to MRI data and compared the efficiency and effectiveness of two parallel preconditioners: sparse approximate inverse preconditioners and block-diagonal preconditioners in combination with the General Minimal Residual method.

1.3. Motivation

In [16], Nordström shows that a global edge detection algorithm based on variational regularization can be seen as a biased anisotropic diffusion method. The problem can be formulated as follows:

$$\begin{aligned} u_t = \nabla \cdot (c_n(\|\nabla u\|)\nabla u) + \mu(f-u) \quad &\text{in} \quad \Omega \times (0,T), \\ \frac{\partial u}{\partial \vec{n}} = 0 \quad &\text{on} \quad \partial\Omega \times (0,T), \\ u(x,0) = f \quad &\text{in} \quad \Omega, \end{aligned} \tag{1}$$

where $\Omega \subseteq \mathbb{R}^d$ for $d = 2$ or 3, T is the stopping time, u is the pixel value (which is complex for MR images), f is the noisy image, μ is the fidelity parameter and c_n is a non-negative monotonically decreasing function with $c_n(0) = 1$ and $c_n(\infty) \to 0$. The bias term $(f-u)$ (also called fidelity term) ensures that the filtered image u does not drift too far from the original image f. The author further states that because of the bias term, a steady state solution exists for the diffusion model. In the remainder of the paper, we adopt this widely used model for image denoising of low-field MR images. But instead of integrating the equation in time, we directly solve for the steady state of the model. Formulated in this way, Eq. (1) can be interpreted as the Euler–Lagrange equation of a variational model.

The numerical discretization of our denoising model leads to a nonlinear system

$$A(\mathbf{u})\mathbf{u} = \mathbf{b}$$

where the operator $A(\mathbf{u})$ depends on $\mathbf{u}$ if the diffusion coefficient is solution-dependent. This system can be solved using the following Picard iteration

$$A(\mathbf{u}^n)\mathbf{u}^{n+1} = \mathbf{b}.$$

This iteration was first introduced by Vogel and Oman in [17], and is known as the lagged-diffusion method. In each iteration a large sparse system of linear equations

$$A\mathbf{u} = \mathbf{b} \tag{2}$$

needs to be solved. Here A is a symmetric and positive-definite matrix. For such systems, the CG method is the method of choice. If A is ill-conditioned, i.e., has a large condition number κ, convergence may be unacceptably slow. The standard way to improve the rate of convergence is to apply CG to the preconditioned system $M^{-1}A\mathbf{u} = M^{-1}\mathbf{b}$, which yields the Preconditioned CG (PCG) algorithm. The preconditioner M should be chosen such that it resembles A, and that systems with M are easy to solve. A complementary way to speed up convergence is to combine PCG with deflation. This method is called Deflated Preconditioned CG (DPCG). In 1987, Nicolaides [18] chose the deflation vectors to be piecewise constant, where the nonzeros correspond to a partitioning of the domain into subdomains. This technique is particularly efficient if the spectrum of the preconditioned matrix contains a limited number of small eigenvalues.

1.4. Review of preconditioners for the lagged-diffusion linear system

Several preconditioners have been proposed for the linear system (2). Vogel and Oman [17] combined PCG with a multigrid method as a preconditioner for solving the TV model. Our approach is closely related to theirs. While they restricted themselves to the solution of the TV model with multigrid, we extend this by investigating different types of preconditioners in combination with different diffusion models. In 2007, Duarte-Carvajalino et al. [19] considered image denoising for hyperspectral images using the regularized PM model. They employed the Additive Operator Splitting (AOS) [20] and Alternating Direction Implicit schemes [21] as preconditioners for the PCG linear solvers. In 2010, Bertaccini et al. [22] proposed an updating strategy for the (incomplete) Cholesky preconditioners for the sequence of lagged-diffusion linear systems. In [23], the authors presented a simple, directionally-split, semi-implicit method for anisotropic diffusion which is linearly stable for large timesteps. The authors also suggested that this method might be able to serve as an effective preconditioner to further accelerate an unsplit iterative method. In 2013, a novel kind of regularization of the classical Perona–Malik model was proposed in [24]. The authors used the Krylov subspace spectral methods to implement the diffusion model. In 2014, Arridge et al. [25] derived a factorization-free preconditioned LSQR algorithm for solving large-scale linear inverse imaging problems, regularized with a nonlinear, edge-preserving penalty term such as Total Variation or the Perona–Malik technique. The method is aimed at problems defined on unstructured meshes, where such regularizers naturally arise in unfactorized form as a stiffness matrix of an anisotropic diffusion operator and factorization is prohibitively expensive.

1.5. Contributions of this paper

The goal of this paper is to analyze and evaluate suitable preconditioners in combination with deflation to solve Eq. (2). We consider three preconditioners that are representative for different classes. The first is Jacobi preconditioning which is a classical preconditioner that is based on a regular splitting of the matrix. The second is the AOS preconditioner, which is an operator splitting method that splits the two- or three-dimensional problem into a sequence of one-dimensional problems. The third preconditioner approximates the variable diffusion operator by the standard Laplace operator. The action of the inverse of the (shifted) Laplace operator is computed using the Discrete Cosine Transform (DCT). All three methods are trivial to parallelize and can be implemented matrix-free. For reasons of comparison, we also consider unpreconditioned CG. We analyze the preconditioners for four different denoising models and provide upper bounds on the number of CG iterations.

We combine the methods with deflation, which can be seen as a coarse-grid correction. We have evaluated our methods on a wide range of low-field MR images. Our analysis and the numerical results show that the best choice of preconditioner depends on the denoising model. If the diffusion is modeled by a smoothly varying function, the DCT preconditioner is best, while for a strongly varying diffusion, Jacobi preconditioning combined with deflation works best.

1.6. Structure of this paper

This paper starts in Section 2 with the description of the denoising models. It also explains the structure of the linear system that has to be solved in every Picard iteration. Section 3 describes the DPCG method that is used to solve these linear systems. It explains the deflation operation and presents the preconditioners we consider: Jacobi preconditioning, AOS, and DCT. Section 4 gives numerical experiments on low-field MR images. We present the denoised images and give the numerical results for CG, PCG, and DPCG for the different denoising models. These results show that the relative performance of the preconditioners depends on the denoising model. Finally, we end with conclusions in Section 5.

1.7. Notation

We use the following notations. Vectors are denoted by bold characters, matrices by capitals, and scalars by regular characters. The superscript H denotes the conjugate transpose, and T the normal transpose. Norm is denoted by $\|\cdot\|$. $\|\cdot\|$ with a subscript denotes a specific norm. Norms without subscripts are standard 2-norms. $\otimes$ stands for the Kronecker product.

2. Mathematical model and numerical algorithms

2.1. The denoising models

The specific models we consider are formulated as follows:

$$\begin{aligned} \nabla\cdot(c_n(\|\nabla u\|)\nabla u)+\mu(f-u)=0 \quad &\text{in} \quad \Omega\times(0,T),\\ \frac{\partial u}{\partial \vec{n}}=0 \quad &\text{on} \quad \partial\Omega\times(0,T),\\ u(x,0)=f \quad &\text{in} \quad \Omega, \end{aligned} \tag{3}$$

where $\Omega\subseteq\mathbb{R}^2$ or $\mathbb{R}^3$. Choices for the function c_n we consider are:

$$c_1=1, \tag{4}$$

$$c_2(\|\nabla u\|)=\frac{1}{1+\left(\frac{\|\nabla u\|}{K}\right)^2}, \tag{5}$$

$$c_3(\|\nabla u\|)=\mathrm{e}^{-(\|\nabla u\|/K)^2}, \tag{6}$$

$$c_4(\|\nabla u\|)=\frac{1}{\|\nabla u\|}, \tag{7}$$

where K is a damping parameter.

The above equations define four different denoising models, which we denote as Models 1–4 according to the coefficients c_1-c_4. Some of them have also been used in [26] for edge detections. Choosing c_1 yields the stationary standard heat equation with source term $\mu(u_0-u)$. It models the stationary solution of heat flow, which makes it efficient in removing noise but tends to blur the edges of the image. c_2 and c_3 are choices that were already proposed in the classic paper by Perona and Malik [10]. c_2 privileges wide regions over smaller ones and c_3 privileges high-contrast edges over low-contrast ones. Taking c_4, (3) corresponds to the Euler–Lagrange equation of the TV model [8]. In the original paper [8], the authors suggested to use the gradient descent method to solve the evolution equation. Instead of using the gradient descent method, Chan et al. [27] solved the steady state for the TV model directly, which is the approach we take in this paper.

In [28], the authors explained in Section 2.7.6 that, under some assumptions, the solution $u(t,\cdot)$ of the time-dependent form of model (3) should approximate a minimizer $u(\cdot)$ of model (3) as t increases. Eq. (3) can also be viewed as one time step of the implicit Euler time-integration method applied to the diffusion equation $u_t=\nabla\cdot(c_n(\|\nabla u\|)\nabla u)$. A single implicit Euler time step applied to the diffusion equation is given by $\frac{u^1-u^0}{\tau}=\nabla\cdot(c_n(\|\nabla u^1\|)\nabla u^1)$, which can be rewritten as $\nabla\cdot(c_n(\|\nabla u^1\|)\nabla u^1)+\frac{1}{\tau}(u^0-u^1)=0$. The initial value of u is f, which means we can choose $u^0=f$ and we set $u^1=u$ and $\mu=\frac{1}{\tau}$. This exactly matches the first equation in (3). Therefore, solving the steady-state equation with the fidelity term is equivalent to applying one implicit Euler step to the transient diffusion equation without fidelity term.

2.2. Numerical discretization

We use the standard finite difference method to discretize (3) in space, see also [10]. For ease of presentation, we assume that the number of pixels is equal to N in each direction and that the images are defined on the unit domain. This implies that for the step size we have

$$h=\frac{1}{N}.$$

We first consider the one-dimensional case. The one-dimensional discretization of $\nabla(c_n\cdot\nabla u)$ at point x_i is given by

$$\nabla(c_n\cdot\nabla u)_{x_i}\approx c_{i+\frac{1}{2}}\frac{(u_{i+1}-u_i)}{h^2}-c_{i-\frac{1}{2}}\frac{(u_i-u_{i-1})}{h^2},$$

where $c_{i\pm\frac{1}{2}}=\frac{c_{i\pm1}+c_i}{2}$. $c_i:=c(|u_x|_i)=c(|\frac{u_{i+1}-u_{i-1}}{2h}|)$ for $0\le i\le N-1$. Due to the Neumann boundary conditions, we have that $u_{-1}=u_0$ and $u_{N-1}=u_N$.

The one-dimensional diffusion matrix is given by

$$C^{1d} = \frac{1}{h^2}\begin{pmatrix} -c_{\frac{1}{2}} & c_{\frac{1}{2}} & & & \\ c_{\frac{1}{2}} & -(c_{\frac{1}{2}} + c_{1+\frac{1}{2}}) & c_{1+\frac{1}{2}} & & \\ & \ddots & \ddots & \ddots & \\ & & c_{N-\frac{5}{2}} & -(c_{N-\frac{5}{2}} + c_{N-\frac{3}{2}}) & c_{N-\frac{3}{2}} \\ & & & c_{N-\frac{3}{2}} & -c_{N-\frac{3}{2}} \end{pmatrix}.$$

Note that the diffusion matrix is diagonally equivalent, which means that for every row the absolute value of the main diagonal is equivalent to the sum of the absolute values of the sub-diagonal elements. Moreover, the matrix is symmetric and negative definite.

For the two-dimensional case, we write our matrix as

$$C^{2d} = C_x + C_y$$

where C_l describes the diffusion in the l direction, $l = x,\ y$. C_l is a block-diagonal matrix, in which each block corresponds to a one-dimensional diffusion matrix. We refer to, e.g., [20] for the precise definition of C^{2d}. The resulting discretized equation is given by

$$C(\mathbf{u})\mathbf{u} + \mu(\mathbf{f} - \mathbf{u}) = 0. \tag{8}$$

where

$$\mathbf{u} = (u_{0,0}, \ldots, u_{N-1,0}, \ldots, u_{0,N-1}, \ldots, u_{N-1,N-1})^T,$$
$$\mathbf{f} = (f_{0,0}, \ldots, f_{N-1,0}, \ldots, f_{0,N-1}, \ldots, f_{N-1,N-1})^T.$$

in two-dimensional case. The three-dimensional case can be defined similarly.

Eq. (8) can be rewritten as

$$(I - \frac{1}{\mu}C(\mathbf{u}))\mathbf{u} = \mathbf{f}. \tag{9}$$

We solve the nonlinear system (9) with the lagged diffusion Picard iteration as proposed in [17]:

$$(I - \frac{1}{\mu}C(\mathbf{u}^n))\mathbf{u}^{n+1} = \mathbf{f}. \tag{10}$$

In every Picard iteration a linear system of the form

$$A\mathbf{u} = \mathbf{b} \tag{11}$$

needs to be solved. The matrix A is symmetric and positive definite. However, because of possible discontinuities in the diffusion parameter c, the matrix A may be ill-conditioned.

3. Linear system solver

3.1. Deflated preconditioned conjugate gradients

We use the DPCG to solve the system (11). We first explain the deflation technique as proposed in [18]. This technique splits the solution $\mathbf{u}$ into two parts, one in the range of the deflation subspace and another in its complement. To this end, we define the projector P by

$$P = I - AZ(Z^TAZ)^{-1}Z^T, \quad Z \in \mathbb{R}^{n\times m}$$

where I is the identify matrix, and $Z = [z_1\ z_2\ \cdots\ z_m]$ is the deflation matrix of rank m. The columns of Z span the deflation subspace. Since $\mathbf{u} = (I - P^T)\mathbf{u} + P^T\mathbf{u}$ and

$$(I - P^T)\mathbf{u} = Z(Z^TAZ)^{-1}Z^TA\mathbf{u} = ZA_c^{-1}Z^T\mathbf{b},$$

we only need to calculate $P^T\mathbf{u}$. We solve the deflated system as follows:

$$PA\tilde{\mathbf{u}} = P\mathbf{b}.$$

We use the DPCG method to get $\tilde{\mathbf{u}}$, and multiply $\tilde{\mathbf{u}}$ by P^T to obtain $P^T\tilde{\mathbf{u}}$ which is equal to $P^T\mathbf{u}$.

The initial residual is $\mathbf{r}_0 = P\mathbf{b} - PA\mathbf{u}^n$. M is the preconditioner. Note that the matrices are real-valued, and that the vectors are complex-valued.

$k = 0$.
while $\|\mathbf{r}_k\| < \epsilon\|\mathbf{r}_0\|$ **do**
  Solve $\mathbf{z}_k = M^{-1}\mathbf{r}_k$
  $k = k + 1$
  if $k = 1$ **then**
    $\mathbf{p}_1 = \mathbf{z}_0$
  else
    $\beta_k = \mathbf{r}_{k-1}^H \mathbf{z}_{k-1} / (\mathbf{r}_{k-2}^H \mathbf{z}_{k-2})$
    $\mathbf{p}_k = \mathbf{z}_{k-1} + \beta_k \mathbf{p}_{k-1}$
  end
  $\alpha_k = \mathbf{r}_{k-1}^H \mathbf{z}_{k-1} / (\mathbf{p}_k^H PA\mathbf{p}_k)$
  $\tilde{\mathbf{u}}_k = \tilde{\mathbf{u}}_{k-1} + \alpha_k \mathbf{p}_k$
  $\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k PA\mathbf{p}_k$
end
$\mathbf{u} = Z(A_c)^{-1}Z^T\mathbf{b} + P^T\tilde{\mathbf{u}}_k$.

Algorithm 1: The DPCG algorithm for solving (11).

In [18], Nicolaides defined Z based on a decomposition of the domain Ω. The idea is first to decompose the domain Ω into m nonoverlapping subdomains Ω_i, $i = 1, 2, \ldots, m$. Then choosing vectors z_i for $i \in \{1, 2, \ldots, m\}$ such that $z_i = 1$ on Ω_i and $z_i = 0$ on Ω_j, $j \neq i$, $j \in \{1, 2, \ldots, m\}$. With this choice of matrix Z, the resulting deflation technique is called subdomain deflation [18]. It can be interpreted as a two-level multigrid method [29], where the projection matrix P corresponds to a coarse-grid correction using a piece-wise constant interpolation with extreme coarsening. In this paper, we use squares or cubes of equal size as subdomains for deflation.

3.2. Preconditioners

We consider four different preconditioning techniques. Each preconditioner has a low computational complexity of (approximately) $O(N^d)$, in which d is the dimension of the problem, both to construct and to apply and is easily parallelisable. In this section, we will describe the preconditioners and give for each preconditioner an upper bound on the condition number of the preconditioned matrix. Since both M and A are symmetric positive definite, the condition number of the preconditioned matrix, κ_{prec} is equal to the ratio of the largest and smallest eigenvalue of $M^{-1}A$,

$$\kappa_{prec} = \frac{\lambda_{\max}(M^{-1}A)}{\lambda_{\min}(M^{-1}A)}.$$

Using our upper bounds on the condition numbers, we will derive upper bounds on the required number of iterations for the different combinations of preconditioners and diffusion models. In our analysis, we frequently need the absolute value maximum of main diagonal element of a matrix B. To simplify notation we therefore define

$$r(B) = \max_i |B_{i,i}|$$

to denote this value.

3.2.1. No preconditioning

In case no preconditioner is applied, we can simply bound the eigenvalues $\lambda(A)$ of A using Gershgorin's theorem. By using that $C(\mathbf{u}^n)$ is diagonally equivalent, we know that the eigenvalues are bounded by

$$|\lambda(A) - (1 + \frac{1}{\mu} r(C(\mathbf{u}^n)))| \leq \frac{1}{\mu} r(C(\mathbf{u}^n))$$

and, by using that A is symmetric and hence has real eigenvalues, we obtain

$$1 \leq \lambda(A) \leq 1 + \frac{2}{\mu} r(C(\mathbf{u}^n)),$$

and hence that the condition number κ_{unprec} of the unpreconditioned matrix is bounded by

$$\kappa_{unprec} \leq 1 + \frac{2}{\mu} r(C(\mathbf{u}^n)).$$

We furthermore can make use of the definition of the diffusion coefficients $c_{i,j}(\mathbf{u}^n)$ for the different models. Clearly

$$0 \leq |c_{i,j}(\mathbf{u}^n)| \leq 1 \text{ and hence } r(C(\mathbf{u}^n)) \leq \frac{2d}{h^2}$$

for diffusion coefficients (4)–(6). Making use of this, we obtain that

$$\kappa_{unprec} \leq 1 + \frac{4d}{h^2\mu}.$$

However, for c_4, the entries of $C(\mathbf{u}^n)$ may become arbitrarily large for flat regions in an image, and hence for the TV model $r(C(\mathbf{u}^n))$ may become unbounded.

3.2.2. Jacobi preconditioning

A simple and popular preconditioning method is to use the main diagonal of A as a preconditioning matrix M. This technique is known as Jacobi preconditioning. The clear advantages are that it is easy to implement, parallelize, and apply. Using Gershgorin's theorem we obtain

$$|\lambda(M^{-1}A) - 1| \leq \frac{\frac{1}{\mu}r(C(\mathbf{u}^n))}{1 + \frac{1}{\mu}r(C(\mathbf{u}^n))}$$

and hence

$$1 - \frac{\frac{1}{\mu}r(C(\mathbf{u}^n))}{1 + \frac{1}{\mu}r(C(\mathbf{u}^n))} \leq \lambda(M^{-1}A) \leq 1 + \frac{\frac{1}{\mu}r(C(\mathbf{u}^n))}{1 + \frac{1}{\mu}r(C(\mathbf{u}^n))}.$$

For the condition number of the Jacobi-preconditioned A, κ_{Jacobi}, we obtain

$$\kappa_{Jacobi} \leq 1 + \frac{2}{\mu}r(C(\mathbf{u}^n))$$

which is the same as the bound for the condition number of the unpreconditioned matrix. However, an advantage of Jacobi preconditioning is that, in case of strongly varying coefficients, the spectrum only contains a few small eigenvalues. When this is the case, Jacobi preconditioning combined with deflation can be quite efficient, see for example [30].

3.2.3. AOS

Assuming the image domain to be d dimensional, we can rewrite Eq. (10) as

$$\left[\frac{1}{d}\Sigma_{l=1}^{d}(I - \frac{d}{\mu}C_l(\mathbf{u}^n))\right]\mathbf{u}^{n+1} = \mathbf{f}. \tag{12}$$

In the one-dimensional case, (12) is a tridiagonal linear system that can be solved by the Thomas algorithm in linear time. In [20], the authors proposed a method called Additive Operator Splitting to approximately solve equation (12) in the multi-dimensional case, by solving a sequence of tridiagonal linear systems. The AOS method is defined by

$$\mathbf{u}^{n+1} \approx \frac{1}{d}\Sigma_{l=1}^{d}\left[I - \frac{d}{\mu}C_l(\mathbf{u}^n)\right]^{-1}\mathbf{f}.$$

The operators $I - \frac{d}{\mu}C_l(\mathbf{u}^n)$ describe the one-dimensional diffusion operators along the x_l axes. They are strictly diagonally dominant tridiagonal matrices (if properly ordered), hence systems with these matrices can be solved with linear complexity using the Thomas algorithm. Moreover, the one-dimensional systems along the same direction can be solved in parallel. The AOS preconditioner is then defined as

$$M^{-1} = \frac{1}{d}\Sigma_{l=1}^{d}\left[I - \frac{d}{\mu}C_l(\mathbf{u}^n)\right]^{-1}.$$

To bound the condition number κ_{AOS} for the AOS-preconditioner, we will again first bound the eigenvalues of the preconditioned matrix $M^{-1}A$. This matrix can be written as

$$\begin{aligned}
M^{-1}A &= \left(\frac{1}{d}\Sigma_{l_1=1}^{d}\left[I - \frac{d}{\mu}C_{l_1}(\mathbf{u}^n)\right]^{-1}\right)\left(\frac{1}{d}\Sigma_{l_2=1}^{d}\left[I - \frac{d}{\mu}C_{l_2}(\mathbf{u}^n)\right]\right) \\
&= \frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1}^{d}\left[I - \frac{d}{\mu}C_{l_1}(\mathbf{u}^n)\right]^{-1}\left[I - \frac{d}{\mu}C_{l_2}(\mathbf{u}^n)\right] \\
&= I + \frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1,l_2\neq l_1}^{d}\left[I - \frac{d}{\mu}C_{l_1}(\mathbf{u}^n)\right]^{-1}\left[\frac{d}{\mu}(C_{l_1}(\mathbf{u}^n) - C_{l_2}(\mathbf{u}^n))\right].
\end{aligned}$$

We now use the Rayleigh quotient of the preconditioned matrix

$$R_{M^{-1}A}(\mathbf{x}) = \frac{\mathbf{x}^H A\mathbf{x}}{\mathbf{x}^H M\mathbf{x}}$$

to bound its eigenvalues

$$\min_{\mathbf{x}} R_{M^{-1}A}(\mathbf{x}) \leq \lambda(M^{-1}A) \leq \max_{\mathbf{x}} R_{M^{-1}A}(\mathbf{x}),$$

by deriving upper and lower bounds on $R_{M^{-1}A}(\mathbf{x})$. First we note that

$$\frac{\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))}{1-\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))} \leq \frac{\mathbf{x}^H(\frac{d}{\mu}C_{l_1}(\mathbf{u}^n))\mathbf{x}}{\mathbf{x}^H(I-\frac{d}{\mu}C_{l_1}(\mathbf{u}^n))\mathbf{x}} < 0$$

and that

$$0 \leq -\frac{\mathbf{x}^H(\frac{d}{\mu}C_{l_2}(\mathbf{u}^n))\mathbf{x}}{\mathbf{x}^H(I-\frac{d}{\mu}C_{l_1}(\mathbf{u}^n))\mathbf{x}} \leq -\frac{d}{\mu}\lambda_{\min}(C_{l_2}(\mathbf{u}^n)).$$

Here we used that C_{l_1} and C_{l_2} are negative semi-definite, and that

$$\mathbf{x}^H(I-\frac{d}{\mu}C_{l_1}(\mathbf{u}^n))\mathbf{x} \geq \mathbf{x}^H\mathbf{x}.$$

Combining these bounds, we obtain

$$\frac{\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))}{1-\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))} \leq \frac{\mathbf{x}^H(\frac{d}{\mu}(C_{l_1}(\mathbf{u}^n)-C_{l_2}(\mathbf{u}^n)))\mathbf{x}}{\mathbf{x}^H(I-\frac{d}{\mu}C_{l_1}(\mathbf{u}^n))\mathbf{x}} \leq -\frac{d}{\mu}\lambda_{\min}(C_{l_2}(\mathbf{u}^n)).$$

We use this result to bound the Rayleigh quotient $R_{M^{-1}A}(\mathbf{x})$, and with that the eigenvalues of the preconditioned matrix:

$$1+\frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1,l_2\neq l_1}^{d}\frac{\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))}{1-\frac{d}{\mu}\lambda_{\min}(C_{l_1}(\mathbf{u}^n))} \leq \lambda(M^{-1}A)$$
$$\leq 1-\frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1,l_2\neq l_1}^{d}\frac{d}{\mu}\lambda_{\min}(C_{l_2}(\mathbf{u}^n)).$$

Applying Gersgorin's theorem yields

$$1-\frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1,l_2\neq l_1}^{d}\frac{\frac{2d}{\mu}r(C_{l_1}(\mathbf{u}^n))}{1+\frac{2d}{\mu}r(C_{l_1}(\mathbf{u}^n))} \leq \lambda(M^{-1}A)$$
$$\leq 1+\frac{1}{d^2}\Sigma_{l_1=1}^{d}\Sigma_{l_2=1,l_2\neq l_1}^{d}\frac{2d}{\mu}r(C_{l_2}(\mathbf{u}^n)).$$

For Models 1–3, for which the diffusion coefficients are given by (4)–(6), this can be simplified to

$$1-(d-1)\frac{\frac{4}{\mu h^2}}{1+\frac{4d}{\mu h^2}} \leq \lambda(M^{-1}A) \leq 1+(d-1)\frac{4}{\mu h^2}.$$

and for the condition number, we obtain

$$\kappa_{AOS} \leq \frac{1+(d-1)\frac{4}{\mu h^2}}{1+\frac{4}{\mu h^2}}(1+\frac{4d}{\mu h^2}).$$

Note that this upper bound is sharp for $d = 1$, and for $d = 2$ it is the same as for no-preconditioning and Jacobi preconditioning.

3.2.4. Discrete Cosine Transform preconditioner

The standard Laplace operator for the Neumann problem can be diagonalized using the Discrete Cosine Transform. This means that it is easy to solve a system involving the shifted standard and scaled Laplace operator $I-\frac{1}{\mu}C_{heat}$, where C_{heat} denotes the standard Laplace operator for the Neumann problem. This operator is therefore potentially a good preconditioner for the varying-coefficient matrix $I-\frac{1}{\mu}C(\mathbf{u}^n)$, if the coefficients vary smoothly. The idea of exploiting DCT of the standard Laplace operator as preconditioner was first proposed in [31].

The DCT preconditioner uses knowledge of the eigenpairs of the discrete Laplace operators C_{heat}, which we now briefly review. The one-dimensional standard Laplace operator with Neumann boundary is given by

$$C_{heat}^{1d} = \frac{1}{h^2}\begin{pmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{pmatrix}.$$

The eigenvalues and eigenvectors of this matrix are well-known and can be found for example in [32]. The eigenvalues matrix are given by

$$\lambda_i(C^{1d}_{heat}) = -\frac{1}{h^2}(2 - 2\cos\frac{2i\pi}{N})$$

for $i = 0, \ldots, N-1$. The corresponding orthogonal eigenvectors are

$$\mathbf{v}^j = [v^i], \quad v^i = \cos(j + \frac{1}{2})\frac{i\pi}{N}, \quad \text{for } j = 0, \ldots, N-1.$$

The eigenvector matrix

$$V^{1d}_{heat} = \left[\ \mathbf{v}^0, \mathbf{v}^1, \ldots, \ldots, \mathbf{v}^{N-1} \ \right]$$

is exactly the one-dimensional DCT operator, and its adjoint $[V^{1d}_{heat}]^T$ the inverse DCT. Multiplication with the matrices V^{1d}_{heat} and $[V^{1d}_{heat}]^T$ only requires $N \cdot O(\log(N))$ operations.

We can express the two-dimensional standard Laplace operator in terms of one-dimensional matrices by $C^{2d}_{heat} = (C^{1d}_{heat} \otimes I_N) + (I_N \otimes C^{1d}_{heat})$, where I_N is the identity matrix. The eigenvalues are given by $\lambda_{(i,j)} = \lambda_i + \lambda_j$ for all pairs (i, j). The corresponding eigenvectors are $\mathbf{v}^{i,j} = \mathbf{v}^i \otimes \mathbf{v}^j$. The eigenvalues and eigenvectors of the three-dimensional case can be obtained in the same way.

We can exploit the above theory by taking the matrix $M = I - \frac{1}{\mu}C_{heat}$ as a preconditioner. This matrix has the same eigenvectors as C_{heat}. The eigenvalues are given by

$$\lambda(M) = 1 - \frac{1}{\mu}\lambda(C_{heat}).$$

Operation with M^{-1} can be performed by taking sequences of N one-dimensional DCTs in the x, y, and z directions, followed by multiplication with the inverse of a diagonal matrix with the eigenvalues of M on the diagonal, followed by sequences of N one-dimensional inverse DCTs in z, y, and x direction.

In order to bound the condition number κ_{DCT} for the preconditioned matrix $M^{-1}A$, we first write

$$A = I - \frac{1}{\mu}(C_{heat} + C(\mathbf{u}^n) - C_{heat}).$$

Therefore we have

$$M^{-1}A = I + \frac{1}{\mu}M^{-1}(C(\mathbf{u}^n) - C_{heat}).$$

We can bound the eigenvalues of the matrix $(C(\mathbf{u}^n) - C_{heat})$ using Gershgorin's theorem

$$0 \leq \lambda(C(\mathbf{u}^n) - C_{heat}) \leq 2|-r(C(\mathbf{u}^n)) + \frac{2d}{h^2}|.$$

For Models 1–3, we know that $0 \leq r(C(\mathbf{u}^n)) \leq r(C_{heat}) = 2d/h^2$, so

$$0 \leq \lambda(C(\mathbf{u}^n) - C_{heat}) \leq \frac{4d}{h^2}.$$

Using the Rayleigh quotient to bound the eigenvalues of this matrix yields

$$1 \leq \lambda(M^{-1}A) \leq 1 + \max_{\mathbf{x}} \frac{\frac{1}{\mu}\mathbf{x}^H(C(\mathbf{u}^n) - C_{heat})\mathbf{x}}{\mathbf{x}^H(I - \frac{1}{\mu}C_{heat})\mathbf{x}}.$$

Since $\mathbf{x}^H(I - \frac{1}{\mu}C_{heat})\mathbf{x} \leq \mathbf{x}^H\mathbf{x}$, we obtain

$$\kappa_{DCT} \leq 1 + \frac{4d}{h^2\mu},$$

which is the same bound that we obtained for κ_{unprec} and for κ_{Jacobi}. Of course this bound is very pessimistic, when $C(\mathbf{u}^n) - C_{heat}$ is small. For Model 1, $C(\mathbf{u}^n) = C_{heat}$, and we have $\kappa_{DCT} = 1$. For Models 2 and 3, the diffusion coefficients are close to 1, if $\|\nabla u\|^2/K^2$ is small, in which case we can expect fast convergence.

A bound on the number of iterations to reduce the residual norm below a certain tolerance ϵ is given as below. A classical upper bound on the CG error is, see for example [33],

$$\|\mathbf{u}_k - \mathbf{u}\|_A \leq 2\left(\frac{\sqrt{\kappa_{prec}} - 1}{\sqrt{\kappa_{prec}} + 1}\right)^k \|(\mathbf{u}^n - \mathbf{u})\|_A. \tag{13}$$

By referring to [34] Lemma 2.3.2, we have

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} \leq \sqrt{\kappa_{unprec}}\frac{\|\mathbf{u}_k - \mathbf{u}\|_A}{\|\mathbf{u}^n - \mathbf{u}\|_A}. \tag{14}$$

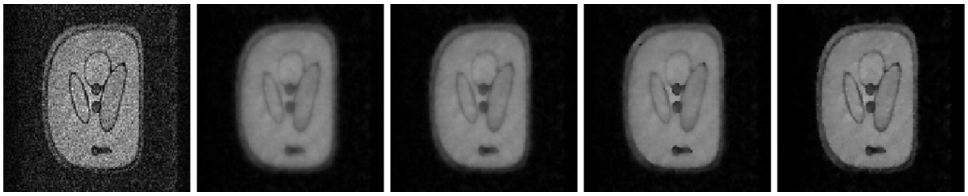

Fig. 1. From left to right: Noisy image, Denoised images by Model 1 ($\mu = 5e3$), Model 2 and Model 3 ($K = 15$, $\mu = 5e3$), and Model 4 ($\mu = 2e3$) with 8^2 deflation vectors.

Giving the tolerance ϵ,

$$2\sqrt{\kappa_{unprec}}\left(\frac{\sqrt{\kappa_{prec}}-1}{\sqrt{\kappa_{prec}}+1}\right)^k = \epsilon,$$

and taking the logarithm, we then obtain

$$k = \ln\left(\frac{\epsilon}{2\sqrt{\kappa_{unprec}}}\right)/\ln\left(\frac{\sqrt{\kappa_{prec}}-1}{\sqrt{\kappa_{prec}}+1}\right). \tag{15}$$

4. Numerical experiments

The numerical tests have been performed on a MacBook Air computer equipped with an Apple M1 CPU and 16 GB of memory. The M1 chip contains 8 cores, four for performance and four for energy efficiency. The algorithms have been implemented in F90/F95, and parallelized using OpenMP. The numerical methods have been tested on a wide range of images. We discuss the results for two representative images in this section. Appendix B presents the results for two additional images.

4.1. Images

We show numerical results for two representative images: a two-dimensional Shepp–Logan (SL) image of 128×128 pixels, and a three-dimensional melon of $128 \times 128 \times 128$ pixels. Both images have been obtained with the MRI scanner described in [4]. The image data are complex valued, and we apply our algorithms directly to these complex data.

Figs. 1 and 2 show the raw images and the denoised images for the four different diffusion models. We have used $K = 15$ for the parameter in Models 2 and 3. For the fidelity parameter, we have taken $\mu = 5e3$ for Models 1–3, and for Model 4 we have taken $\mu = 2e3$ for the Shepp–Logan image, and $\mu = 1.5e3$ for the melon. These parameters have been selected to give good visual results and are used in all numerical experiments. The results show that all diffusion models successfully denoise the images. Model 1, which is equivalent to a standard Gaussian filter, as expected, does not preserve the edges well. The images become shaper from left to right. Model 4, Total Variation, gives visually the clearest images (see Figs. 1 and 2).

4.2. Numerical results

This section evaluates the numerical techniques on the two test images. For each image and diffusion model, we report the number of Picard iterations to denoise the image, and for each Picard iteration, we give the number of DPCG iterations. We present the results for the four different preconditioners, and for 4×4, 8×8, and 16×16 deflation vectors for the Shepp–Logan image, and for $4 \times 4 \times 4$ and $8 \times 8 \times 8$ deflation vectors for the melon. As a convergence criterion for DPCG we use

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} < 10^{-5}.$$

The convergence criterion for the Picard iteration is

$$\frac{\|\mathbf{u}^n - \mathbf{u}^{n-1}\|}{\|\mathbf{f}\|} < 10^{-2}.$$

We take the solution of the previous Picard iteration as an initial guess for DPCG.

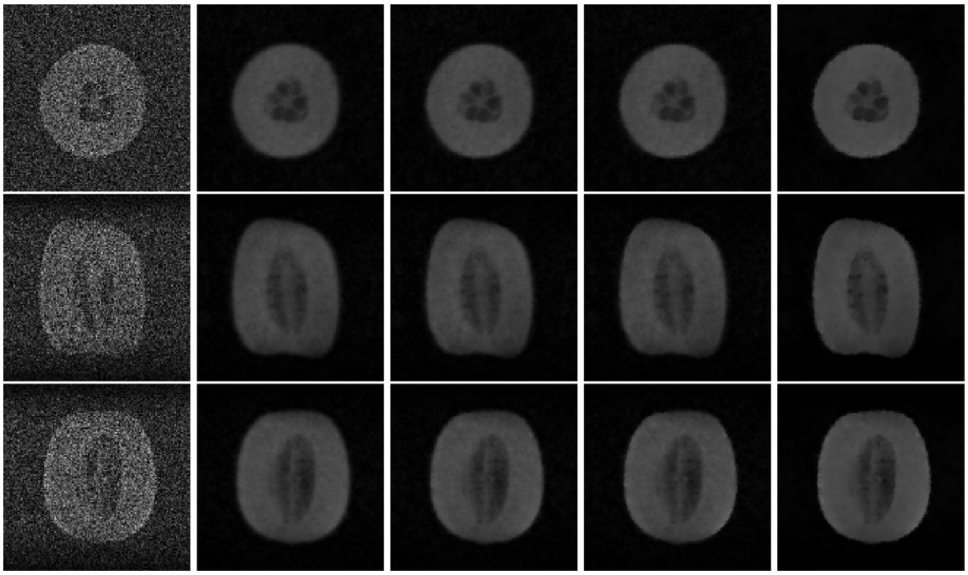

Fig. 2. From left to right: Noisy image, Denoised images by Model 1 ($\mu = 5e3$), Model 2 and Model 3 ($K = 15$, $\mu = 5e3$), and Model 4 ($\mu = 1.5e3$) with 8^3 deflation vectors. From top to bottom: the center slices from x, y and z directions.

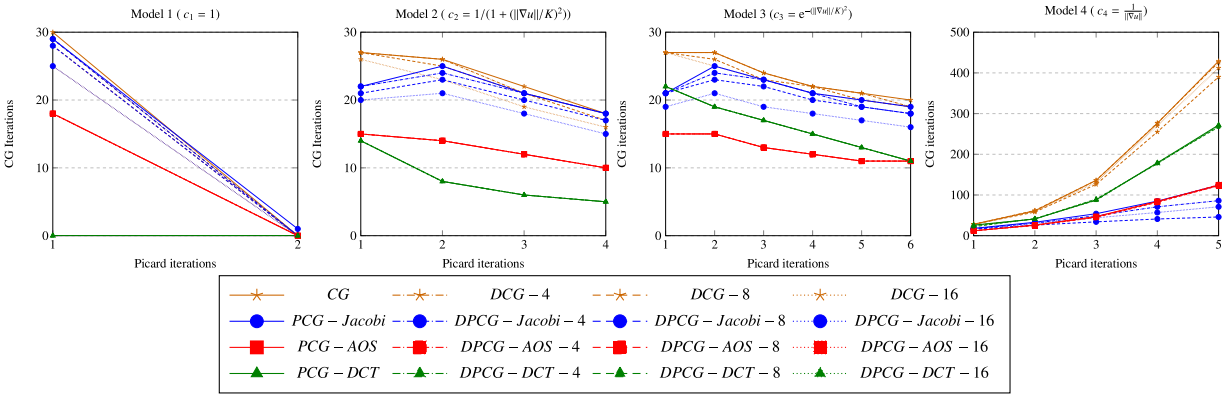

Fig. 3. Measured Shepp–Logan (128 × 128). Picard and CG iteration numbers for different models with different preconditioners.

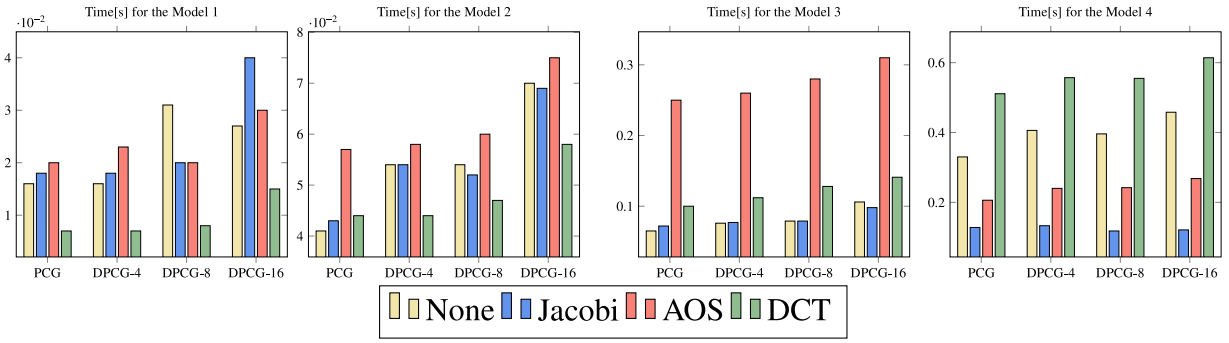

Fig. 4. Measured Shepp–Logan (128 × 128). Run time for different models with different preconditioners.

4.2.1. Shepp–Logan

Fig. 3 gives for each Picard iteration the number of DPCG iterations, and Fig. 4 the elapsed times for the Shepp–Logan image.

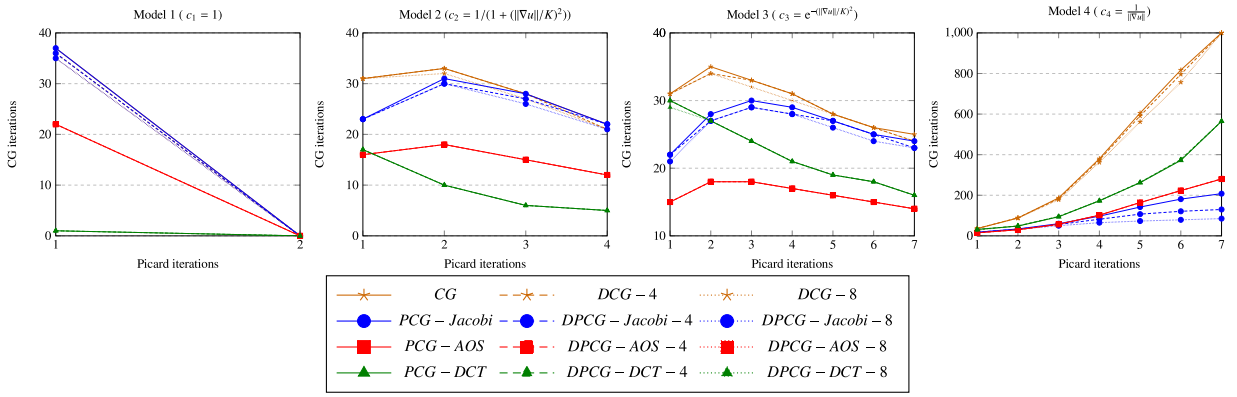

Fig. 5. Melon ($128 \times 128 \times 128$). Picard and CG iteration numbers for different models with different preconditioners.

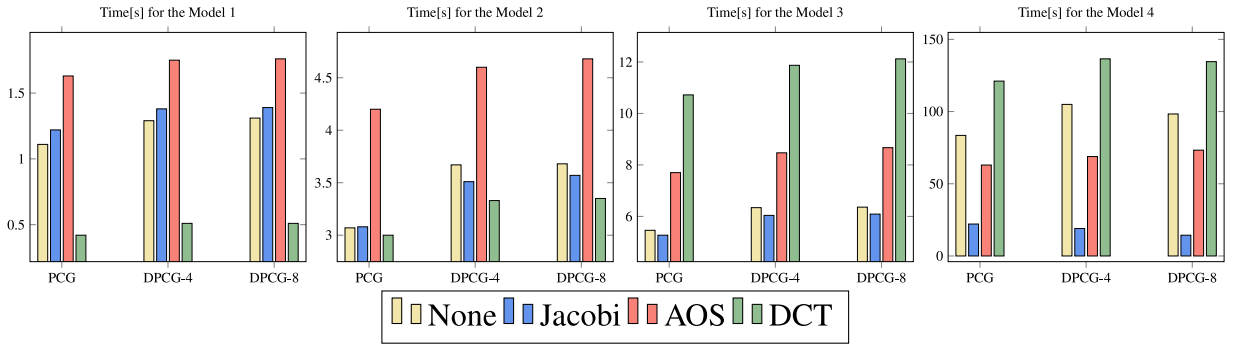

Fig. 6. Melon ($128 \times 128 \times 128$). Run time for different models with different preconditioners.

Model 1 is linear and therefore takes two Picard iterations to converge. This is the minimum since the termination criterion is based on the change in the solution. The DCT preconditioner is a direct method for this model. This is confirmed by the fact that DPCG immediately terminates at the exact solution. For the other preconditioners, the number of PCG iterations is less or equal to 30, which is in agreement with the upper bounds on the number of iterations which are given in the previous section of 36 iterations. We furthermore notice that deflation does not give a significant improvement. The DCT preconditioner without deflation is the fastest method for this model and takes only 7 ms.

Of the three nonlinear diffusion models, Model 2, the rational Perona–Malik model, yields the smoothest variations in the diffusion coefficient. Only four Picard iterations are needed to reach convergence. Also for this model, the DCT preconditioner, although no longer a direct method, yields the lowest number of iterations and computing time. For all preconditioners, the number of iterations is below 30, which agrees with the upper bounds on the number of iterations. Also for this model, deflation does not give a significant improvement.

Model 3, the exponential Perona–Malik model, yields larger variations in the diffusion coefficient than Model 2, which translates into a higher number of seven Picard iterations to reach convergence. Also, DCT preconditioner is no longer the best preconditioner. AOS takes the least number of PCG iterations. In time, however, Jacobi preconditioning is the fastest method. We remark that DCT becomes relatively better than the other preconditioners when the Picard iterations proceed. This is because the images become smoother, and as a consequence, the diffusion coefficients will be almost constant in large patches of the image, which favors DCT. As was the case for Models 1 and 2, deflation does not give a significant improvement for Model 3. The convergence bounds are satisfied. The number of iterations for unpreconditioned CG is even almost equal to the upper bound in the second Picard iteration, which shows that the upper bounds are reasonably sharp.

Model 4, Total Variation, yields the largest variations in the diffusion coefficients, and for this model, we do not have a priori bounds on the number of PCG-iterations. The performance of the different techniques is quite different from the other models. For this model, the number of CG iterations sharply increases when the Picard iterations proceed, since the diffusion coefficients become unbounded in flat regions. The combination of Jacobi preconditioning with deflation gives the best performance. The increase in the number of DPCG-iterations is greatly reduced by the use of deflation. In particular, in the last Picard iteration, a reduction of a factor of three in the number of Jacobi-PCG iterations can be observed, if the method is combined with deflation. Due to the computational overhead of deflation, however, there is no significant reduction of computing time for this example.

Table A.1
SSOR for Model 1.

Image	Preconditioner	Method	Picard		Time[s]
			1	2	
Shepp–Logan	SSOR($\omega = 1$)	PCG	11	2	0.019
		DPCG-4	11	2	0.022
		DPCG-8	10	2	0.034
		DPCG-16	9	2	0.026
	SSOR($\omega = 1.5$)	PCG	8	2	0.016
		DPCG-4	8	2	0.021
		DPCG-8	8	2	0.018
		DPCG-16	8	2	0.027
Melon	SSOR($\omega = 1$)	PCG	13	2	2.093
		DPCG-4	13	2	2.198
		DPCG-8	13	2	2.201
	SSOR($\omega = 1.5$)	PCG	9	2	1.591
		DPCG-4	9	2	1.682
		DPCG-8	9	2	1.690

Table A.2
SSOR for Model 2.

Image	Preconditioner	Method	Picard				Time[s]
			1	2	3	4	
Shepp–Logan	SSOR($\omega = 1$)	PCG	8	9	8	7	0.049
		DPCG-4	8	9	8	7	0.056
		DPCG-8	8	9	7	6	0.052
		DPCG-16	8	8	7	6	0.070
	SSOR($\omega = 1.5$)	PCG	8	7	6	5	0.046
		DPCG-4	8	7	6	5	0.052
		DPCG-8	8	7	6	5	0.049
		DPCG-16	8	7	6	5	0.072
Melon	SSOR($\omega = 1$)	PCG	8	11	10	8	5.408
		DPCG-4	8	11	10	8	5.659
		DPCG-8	8	11	10	8	5.692
	SSOR($\omega = 1.5$)	PCG	8	7	6	5	4.069
		DPCG-4	8	7	6	5	4.299
		DPCG-8	8	7	6	5	4.320

4.2.2. Melon

Next, we consider the melon image. The main difference with the Shepp–Logan image is that the melon image is three-dimensional. While for the two-dimensional Shepp–Logan image, all computing times were in the order of seconds at most, and therefore more of academic interest than of practical importance, we can expect much longer computing times for the three-dimensional melon image. Fig. 5 gives for each Picard iteration the number of DPCG iterations, and Fig. 6 the elapsed times for this image.

For Models 1–3 we can make the same observations as for the Shepp–Logan image. For these models, the upper bound on the number of iterations for no-preconditioning, Jacobi preconditioning, and DCT is 45, and for AOS 62. The observed number of PCG iterations to reach convergence is always below 37. For Models 1 and 2 the DCT preconditioner is the most efficient with computing times of less than half second for Model 1, and less than 4 s for Model 2. AOS is the most efficient for Model 3 and takes about 9 s. Deflation does not improve the convergence. The numbers of iterations are basically the same with or without deflation.

For Model 4, however, the number of PCG iterations grows again for all methods to many hundreds of iterations when the Picard iterations proceed. The resulting computing times are factors larger than for the other models. Jacobi preconditioning is the most efficient preconditioner, and the combination with deflation speeds up the convergence, and also reduces the computing time by a factor of two, to less than 15 s.

5. Conclusions

We have investigated an efficient implicit method for denoising using diffusion filtering. We have considered four different models for the diffusion: constant diffusion, the rational and the exponential diffusion models proposed by Perona and Malik [10], and the Total Variation model. We solved the discretized equations using a lagged-diffusion Picard iteration. The linear systems were solved with Preconditioned CG in combination with deflation. We have evaluated the

Table A.3
SSOR for Model 3.

Image	Preconditioner	Method	Picard							Time[s]
			1	2	3	4	5	6	7	
Shepp–Logan	SSOR($\omega = 1$)	PCG	8	9	8	8	7	7	–	0.071
		DPCG-4	8	9	8	8	7	7	–	0.081
		DPCG-8	8	8	8	7	7	7	–	0.078
		DPCG-16	7	8	7	7	6	6	–	0.105
	SSOR($\omega = 1.5$)	PCG	8	7	7	6	6	5	–	0.067
		DPCG-4	8	7	7	6	6	5	–	0.079
		DPCG-8	8	7	7	6	6	5	–	0.074
		DPCG-16	8	7	7	6	6	5	–	0.094
Melon	SSOR($\omega = 1$)	PCG	8	10	11	10	10	9	9	9.654
		DPCG-4	8	10	11	10	10	9	8	10.00
		DPCG-8	8	10	10	10	9	9	8	9.801
	SSOR($\omega = 1.5$)	PCG	8	8	7	7	6	6	6	7.348
		DPCG-4	8	8	7	7	6	6	6	7.747
		DPCG-8	8	8	7	7	6	6	6	7.800

Table A.4
SSOR for Model 4.

Image	Preconditioner	Method	Picard							Time[s]
			1	2	3	4	5	6	7	
Shepp–Logan	SSOR($\omega = 1$)	PCG	7	12	20	31	45	–	–	0.145
		DPCG-4	7	12	18	26	32	–	–	0.133
		DPCG-8	7	11	16	21	26	–	–	0.118
		DPCG-16	6	10	13	17	19	–	–	0.124
	SSOR($\omega = 1.5$)	PCG	8	10	17	28	36	–	–	0.130
		DPCG-4	8	10	16	23	28	–	–	0.126
		DPCG-8	8	10	15	21	24	–	–	0.122
		DPCG-16	7	10	13	16	18	–	–	0.124
Melon	SSOR($\omega = 1$)	PCG	7	12	21	34	49	63	72	32.81
		DPCG-4	7	12	20	29	37	42	45	25.86
		DPCG-8	7	12	18	23	26	28	30	20.09
	SSOR($\omega = 1.5$)	PCG	8	9	14	22	32	41	46	22.42
		DPCG-4	8	9	14	19	25	28	30	18.42
		DPCG-8	8	9	12	16	18	21	24	15.39

numerical methods on two noisy images that have been obtained with an inexpensive MRI scanner based on permanent magnets. Also for the evaluation of our methods we used an inexpensive computer.

Our conclusions are as follows. For models with constant or slowly varying diffusion coefficients (Models 1 and 2) the DCT preconditioner is most efficient. For Model 3, the AOS preconditioner is best. For all these three models the diffusion coefficients have bounded values between zero and one, which allowed us to give an upper bound on the number of PCG iterations. For these models the computing times are low, a few seconds at most, low enough for practical purposes. Model 4, Total Variation, gives the best image quality. However, denoising with the TV model is computationally challenging, due to the much larger jumps in the diffusion coefficients. For this model, it turns out that a simple Jacobi preconditioning combined with subdomain deflation yields a fast and robust method.

The techniques that we have described in this paper will be used for processing low-field MRI images in low-resource settings. For this, we have focused on an implementation on the inexpensive commodity hardware. On such hardware, we are able to denoise, with the techniques that we have examined, images with a resolution of 128^3 pixels in less than 15 s. This is fast enough for operational purposes.

Acknowledgments

The authors thank the LUMC team for collaborative discussions and providing the low-field MR images. This work is partly funded by NWO, The Netherlands WOTRO under grant W07.303.101 and by the China Scholarship Council.

Appendix A. SSOR as a preconditioner

Symmetric Successive Over Relaxation (SSOR) is a classical preconditioner based on a regular splitting of the matrix. It belongs to the same family as the Jacobi preconditioner. For many problems the number of iterations for SSOR is smaller than for Jacobi preconditioning, in particular if the relaxation parameter is optimized. A drawback of the method is that it

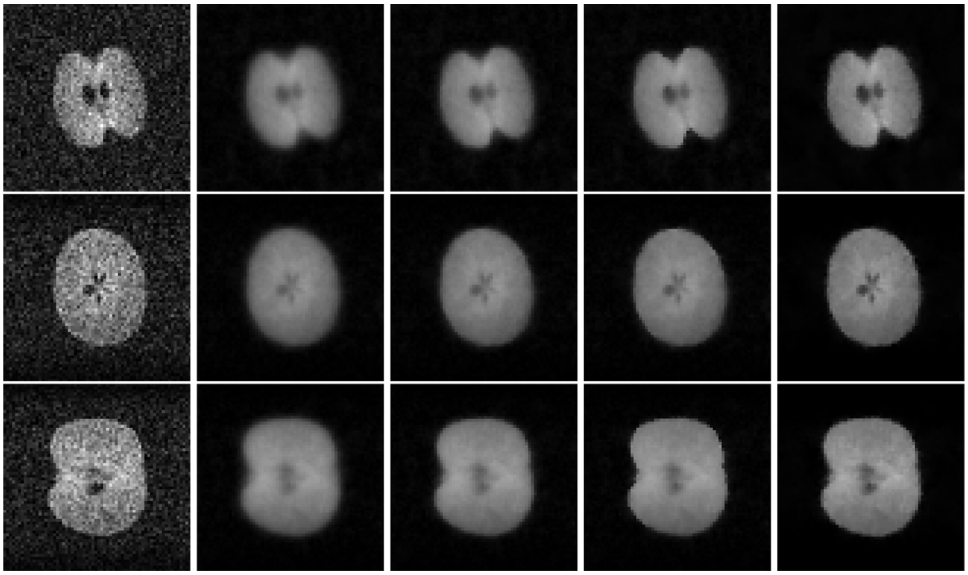

Fig. B.7. From left to right: Noisy image, Model 1 ($\mu = 2e3$), Model 2 and Model 3 ($K = 10$ and $\mu = 2e3$). Model 4 ($\mu = 1e3$), Residual image (Noisy image - Model 4 image) calculated by DPCG with deflation vector number 8^3. From up to down: the center slices from x, y and z directions.

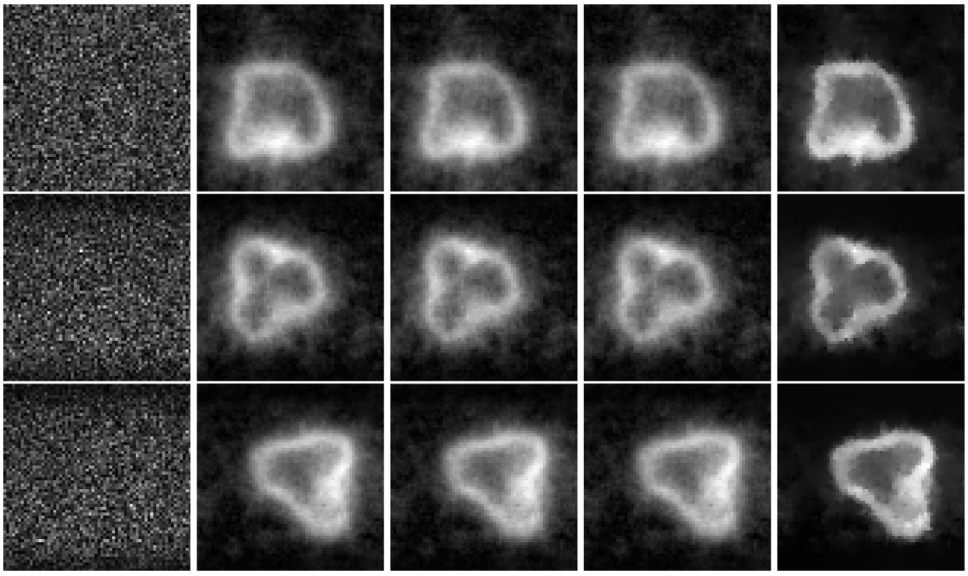

Fig. B.8. From left to right: Noisy image, Model 1 ($\mu = 2e2$), Model 2 and Model 3 ($K = 10$ and $\mu = 2e2$). Model 4 ($\mu = 4e2$) calculated by DPCG with deflation vector number 8^3. From up to down: the center slices from x, y and z directions.

requires a back- and forward substitution operation on triangular matrices to be applied. These operations are not suited for parallel computing.

In this part, we present the result for DPCG with SSOR as a preconditioner. We present the results for relaxation parameter $\omega = 1$, and for the average optimized value $\omega = 1.5$. Our results show that, although the number of iterations is reduced considerably in comparison to Jacobi preconditioning, the timings for SSOR are slightly worse due to the poor parallel performance of the method (see Tables A.1–A.4).

Appendix B. Additional experimental results

We present additional experiments on two three-dimensional images of $64 \times 64 \times 64$ pixels. The first image is of an apple, and the second of a bell pepper. The second image has a very poor signal-to-noise ratio, and without noise filtering

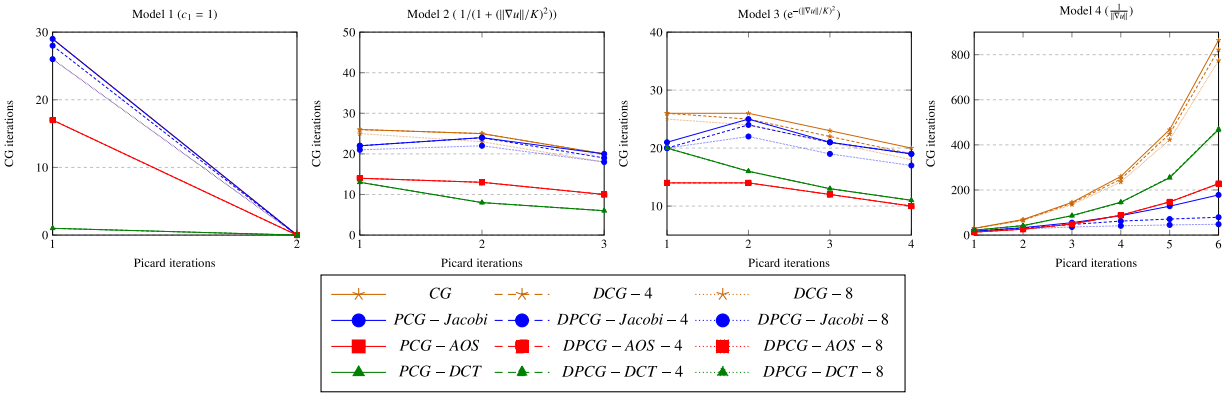

Fig. B.9. Apple (64 × 64 × 64). Picard and CG iteration numbers for different models with different preconditioners.

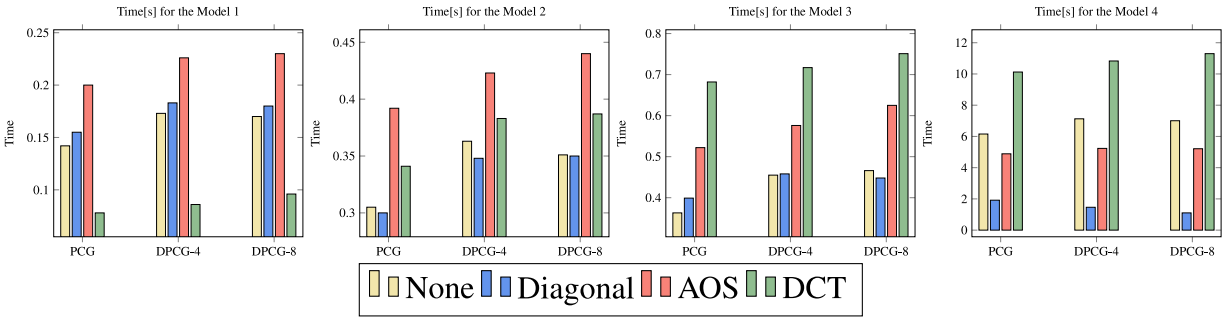

Fig. B.10. Apple (64 × 64 × 64). Run time for different models with different preconditioners.

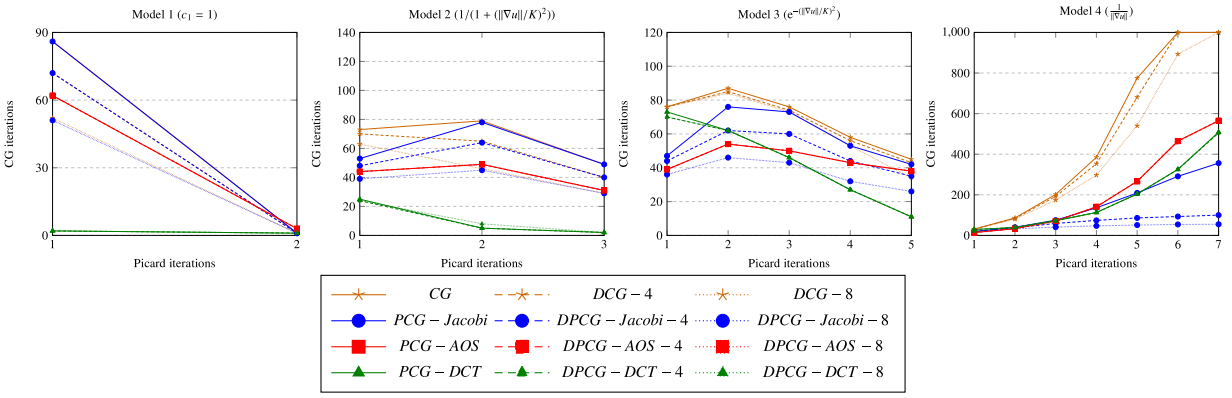

Fig. B.11. Bell pepper (64 × 64 × 64). Picard and CG iteration numbers for different models with different preconditioners.

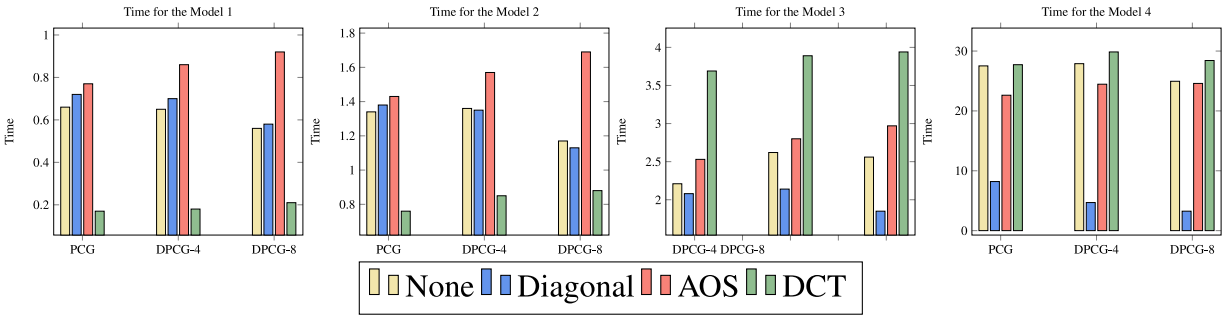

Fig. B.12. Bell pepper (64 × 64 × 64). Run time for different models with different preconditioners.

it is impossible to recognize the bell pepper. The results, given in Figs. B.7–B.12, confirm the conclusions that we drew before.

References

[1] L. Wald, P. McDaniel, T. Witzel, J. Stockmann, C. Cooley, Low-cost and portable MRI, J. Magn. Reson. Imaging (2019).
[2] T. O'Reilly, W.M. Teeuwisse, D. de Gans, K. Koolstra, A.G. Webb, In vivo 3d brain and extremity MRI at 50 mT using a permanent magnet Halbach array, Magn. Reson. Med. 85(1) (2020) 495–505.
[3] M.L. de Leeuw den Bouter, M.B. van Gijzen, R.F. Remis, Conjugate Gradient variants for L_p-regularized image reconstruction in Low-field MRI, SN Appl. Sci. 1 (12) (2019) 1736.
[4] T. O'Reilly, W. Teeuwisse, A. Webb, Three-dimensional MRI in a homogenous 27.cm diameter bore Halbach array magnet, J. Magn. Reson. 307 (2019) 106578.
[5] J. Obungoloch, J. Harper, S. Consevage, I. Savukov, T. Neuberger, S. Tadigadapa, S. Schiff, Design of a sustainable prepolarizing magnetic resonance imaging system for infant hydrocephalus, Magn. Reson. Mater. Phys. Biol. Med. 31 (2018) 665–676.
[6] M.L. de Leeuw den Bouter, D. Gecmen, A. Meijer, L.M. D. de Gans, R. Remis, M.B. van Gijzen, Description of a Low-Field MRI scanner based on permanent magnets, in: CEUR Workshop Proceedings, Vol. 2688, 2020, p. 15.
[7] T. Chan, J. Shen, Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods, Society for Industrial and Applied Mathematics, Philadelpia, 2005.
[8] S.O. Leonid I. Rudin, E. Fatemi, Nonlinear total variation based noise removal algorithm, Physica D 60 (1992) 1–4.
[9] Y. Chen, S. Levine, M. Rao, Variable exponent, linear growth functionals in image restoration, SIAM J. Appl. Math. 66 (4) (2006) 1383–1406.
[10] P. Perona, J. Malik, Scale space and edge detection using anisotropic diffusion, IEEE Trans. Pattern Anal. Mach. Intell. 12 (7) (1990) 629–639.
[11] L. Alvarez, P.-L. Lions, J.-M. Morel, Image selective smoothing and edge detection by nonlinear diffusion. II, SIAM J. Numer. Anal. 29 (3) (1992) 845–866.
[12] M. Welk, J. Weickert, G. Steidl, A four-pixel scheme for singular differential equations, in: Scale Space and PDE Methods in Computer Vision, Springer Berlin Heidelberg, 2005, pp. 610–621.
[13] G. Gerig, O. Kubler, R. Kikinis, F.A. Jolesz, Nonlinear anisotropic filtering of MRI data, IEEE Trans. Med. Imaging 11 (2) (1992) 221–232.
[14] H.M. Golshan, R.P. Hasanzadeh, S.C. Yousefzadeh, An MRI denoising method using image data redundancy and local SNR estimation, Magn. Reson. Imaging 31 (7) (2013) 1206–1217.
[15] N. Kang, J. Zhang, E.S. Carlson, Parallel simulation of anisotropic diffusion with human brain DT-MRI Data, Comput. Struct. 82 (28) (2004) 2389–2399.
[16] K.N. Nordström, Biased anisotropic diffusion: a unified regularization and diffusion approach to edge detection, Image Vis. Comput. 8 (4) (1990) 318–327.
[17] C.R. Vogel, M.E. Oman, Iterative methods for total variation denoising, SIAM J. Sci. Comput. 17 (1) (1996) 227–238.
[18] R.A. Nicolaides, Deflation of Conjugate Gradients with applications to boundary value problems, SIAM J. Numer. Anal. 24 (2) (1987) 355–365.
[19] J. Duarte-Carvajalino, P. Castillo, M. Velez-Reyes, Comparative study of semi-implicit schemes for nonlinear diffusion in hyperspectral imagery, IEEE Trans. Image Process. 16 (2007) 1303–1314.
[20] J. Weickert, B.M.T.H. Romeny, M.A. Viergever, Efficient and reliable schemes for nonlinear diffusion filtering, IEEE Trans. Image Process. 7 (3) (1998) 398–410.
[21] D. Barash, T. Schlick, M. Israeli, R. Kimmel, Multiplicative operator splittings in nonlinear diffusion: from spatial splitting to multiple timesteps, J. Math. Imaging Vision 19 (1) (2003) 33–48.
[22] D. Bertaccini, F. Sgallari, Updating preconditioners for nonlinear deblurring and denoising image restoration, Appl. Numer. Math. 60 (10) (2010) 994–1006.
[23] P. Sharma, G.W. Hammett, A fast semi-implicit method for anisotropic diffusion, J. Comput. Phys. 230 (12) (2011) 4899–4909.
[24] P. Guidotti, Y. Kim, J. Lambers, Image restoration with a new class of forward-backward-forward diffusion equations of Perona–Malik type with applications to satellite image enhancement, SIAM J. Imaging Sci. 6 (3) (2013) 1416–1444.
[25] S.R. Arridge, M.M. Betcke, L. Harhanen, Iterated preconditioned LSQR method for inverse problems on unstructured grids, Inverse Problems 30 (7) (2014) 075009.
[26] V.B.S. Prasath, A. Singh, Edge detectors based anisotropic diffusion for enhancement of digital images, in: 2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing, 2008, pp. 33–38.
[27] T.F. Chan, S. Osher, J. Shen, The digital TV filter and nonlinear denoising, IEEE Trans. Image Process. 10 (2) (2001) 231–241.
[28] L.A. Vese, C.L. Guyader, Variational Methods in Image Processing, CRC Press, 2015.
[29] C.B. Jenssen, P.A. Weinerfelt, A coarse grid correction scheme for implicit multi block Euler calculations, AIAA J. 33 (1995) 1816–1821.
[30] X. Shan, M.B. van Gijzen, Deflated Preconditioned Conjugate Gradients for nonlinear diffusion image enhancement, Lect. Not. Comput. Sci. Eng. 139 (2021) 459–468.
[31] R.H. Chan, M.K. Ng, Conjugate Gradient methods for Toeplitz systems, SIAM Rev. 38 (3) (1996) 427–482.
[32] G. Strang, The Discrete Cosine Transform, SIAM Rev. 41 (1) (1999) 135–147.
[33] J.A. Meijerink, H.A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, Math. Comp. 31 (1977) 148–162.
[34] C.T. Kelley, Iterative Methods for Linear and Nonlinear Equations, Society for Industrial and Applied Mathematics, Philadelphia, USA, 1995.