SPHERICAL FUNCTION REGULARIZATION FOR PARALLEL MRI RECONSTRUCTION

Yonggui Zhu∗ Tuomo Valkonen†

2018-02-04

Abstract From the optimization point of view, a difficulty with parallel MRI with simultaneous coil sensitivity estimation is the multiplicative nature of the non-linear forward operator: the image being reconstructed and the coil sensitivities compete against each other, causing the optimization process to be very sensitive to small perturbations. This can, to some extent, be avoided by regularizing the unknown in a suitably “orthogonal” fashion. In this paper, we introduce such a regularization based on spherical function bases. To perform this regularization, we represent efficient recurrence formulas for spherical Bessel functions and associated Legendre functions. Numerically, we study the solution of the model with non-linear ADMM. We perform various numerical simulations to demonstrate the efficacy of the proposed model in parallel MRI reconstruction.

Key words Parallel MRI, spherical function, regularization, coil sensitivity, ADMM

1 INTRODUCTION

1.1 PARALLEL MAGNETIC RESONANCE IMAGING

Parallel magnetic resonance imaging (p-MRI) increases acquisition speed by simultaneously using multiple radio frequency (RF) detector coils. This helps avoid some of the time-consuming phase-encoding steps in the MRI process. Although less k-space data is received for each coil, this is compensated by data being available from multiple coils. The first approach to p-MRI was based on an arrangement of \( J \) surface coils around the object for MR imaging, one for each \( k \)-space line to be acquired \([28]\). The p-MRI method to achieve routine use was sensitivity encoding, SENSE \([44, 46]\). In this approach, a discrete Fourier transform is used to reconstruct an aliased image for each element in the array. Then the full full-of-view image is generated from the individual sets of images.

Generally, the MRI signal acquired by receiver coil \( j \) is given by

\[
 s_j(\hat{k}) = \int u(\hat{x})c_j(\hat{x})\exp(i\hat{k} \cdot \hat{x})d\hat{x}, \quad j = 1, 2, \ldots, J.
\]

∗School of Science, Communication University of China, P. R. China (ygzhu@cuc.edu.cn)
†Department of Mathematical Sciences, University of Liverpool, United Kingdom (tuomo.valkonen@iki.fi).
Here $u$ is the excited proton density function, $c_j$ the sensitivity profile of the $j$th coil at $\bar{x}$, and $\bar{k}$ is the chosen $k$-space trajectory. In discrete form (1.1) can be written

$$s_j(k_m, k_n) = \sum_{m=1}^{N} \sum_{n=1}^{N} u(m, n)c_j(m, n)\exp(ik_m m)\exp(ik_n n), \quad j = 1, 2, \ldots, J.$$ (1.2)

If the coil sensitivity profiles $c_j(m, n), \quad j = 1, 2, \ldots, J,$ are known, the system of equations (1.2) can be numerically inverted for $u$ with relative ease [24, 52]. An early direct method to invert (1.2) is to decouple the system of equations in image space under regular sub-sampled pattern like SENSE [36, 45, 46]. Another direct approach is to approximate a sparse inverse by using the coil data in $k$-space as in SMASH [51] and g-SMASH [11]. SMASH is a partial p-MRI method using multiple coils to speedup acquisition in the course of imaging. Whereas g-SMATH is generalized SMASH method that reconstructs image with the coil data in $k$-space. However, the MRI signal equation will be increasingly ill-conditioned when the acceleration factor becomes large. The acceleration factor is the ratio of the amount of $k$-space data required for a fully sampled image to the amount collected in accelerated acquisition. When ill-conditioned, the inversion of the linear system (1.2) will lead to the amplification of noise present in the MRI signal $s_j$. Therefore regularization methods are required to improve reconstruction quality. Historically employed regularization methods include the truncated singular value decomposition (TSVD) and damped least-squares (DLS) [37].

If the coil sensitivities are not known, it is common to acquire sensitivity information by using a calibration step [23]. For example, the coil sensitivity profiles can be obtained directly from the reference lines in autocalibrating SENSE [41]. The GRAPPA method [22] is the most widely used autocalibrating technique in the determination of coil sensitivities. The coils sensitivities are generally determined from the center of the $k$-space rather than using all available information. Due to small errors, this leads to residual aliasing artifacts in the reconstruction because. Nonlinear inversion with the joint the estimation of the coil sensitivities $c_j$ and the determination of the proton density image $u$, can improve reconstruction quality [5, 52, 53, 59].

### 1.2 NONLINEAR INVERSION FOR p-MRI

Parallel MR imaging can be formulated as a nonlinear inverse problem with a nonlinear forward operator $\mathcal{F}$, which maps the proton density $u$ and the coil sensitivities $c = (c_1, c_2, \ldots, c_J)^T$ to the measured $k$-space data $g$ as

$$\mathcal{F}(u, c) := (PF(u \cdot c_1), PF(u \cdot c_2), \ldots, PF(u \cdot c_J))^T = g.$$ (1.3)

Here $P$ is the binary sub-sampling mask, $F$ is the discrete 2D Fourier transform, and $g = (g_1, g_2, \ldots, g_J)^T$ the acquired $k$-space measurements for $J$ receiver coils. As shown in [34, 52], the problem (1.3) can be solved by the iteratively regularized Gauss-Newton (IRGN) method [4, 9, 19, 26]. The discrepancy principle is used to obtain a suitable level of regularization. In [35] the authors furthermore expanded IRGN method with variational regularization terms to improve reconstruction quality. The method works as follows. Writing $v = (u, c_1, c_2, \ldots, c_J)^T$, and starting from an initial guess $v^0$, we solve on each step for $\Delta v$ from the linearised problem

$$\min_{\Delta v} \frac{1}{2}||\mathcal{F}'(v^k)\Delta v + \mathcal{F}(v^k) - g||_2^2 + \frac{\alpha_k}{2}R_c(c^k + \Delta c) + \beta_k R_u(u^k + \Delta u).$$ (1.4)
Then we update \( v^{k+1} := v^k + \Delta v \). Here \( R_c(c) \) is a regularization functional for penalizing the high Fourier coefficients of the coils \( c_j, j = 1, 2, \ldots, J \), and \( R_u \) regularizes the image. The regularization parameters \( \alpha_k \) and \( \beta_k \) are updated by the formulas \( \alpha_{k+1} := q_x \alpha_k \) and \( \beta_{k+1} := q_x \beta_k \) with \( 0 < q_x, q_y < 1 \). More details can be found in [35].

There are many options for the regularisers \( R_u \) and \( R_c \) in the inverse problems literature. The most basic regularization is the simple \( L^2 \) penalty \( R_u(u) = \frac{1}{2} \|u\|_2^2 \). This is used in \([34, 35, 52]\). Another conventional choice for the image \( u \) is \( R_u(u) = \text{TV}(u) \), the Total Variation \([12, 25, 39, 49, 63, 64]\). The are two common variants of the total variation, dependent on the choice of pointwise norm used. Restricting ourselves to the finite-dimensional setting, with the two-norm we obtain the isotropic total variation

\[
\text{TV}_I(u) = \sum_{m=1}^{N} \sum_{n=1}^{N} \sqrt{|\nabla_1 u(m, n)|^2 + |\nabla_2 u(m, n)|^2},
\]

while with the 1-norm we obtain the computationally easier but anisotropic total variation

\[
\text{TV}_I(u) = \sum_{m=1}^{N} \sum_{n=1}^{N} |\nabla_1 u(m, n)| + |\nabla_2 u(m, n)|.
\]

In both cases we have used the forward-differences

\[
\nabla_1 u(m, n) = \begin{cases} u(m + 1, n) - u(m, n), & m < N \\ 0, & m = N \end{cases}, \quad \nabla_2 u(m, n) = \begin{cases} u(m, n + 1) - u(m, n), & n < N \\ 0, & n = N. \end{cases}
\]

If the penalty parameter \( \beta_k \) becomes large, TV regularization will generate staircasing artefacts. This can be avoided through the use of second-order Total Generalized Variation (TGV) \([10, 33, 56]\).

For the regularization term \( R_c(c) \), one choice from \([4, 9, 19, 26]\) is to take \( R_c(c) = \|w \cdot F c\|_2^2 \), where \( w \) is an weighting operator that penalizes high Fourier coefficients. It is well known that coil sensitivities are generally rather smooth functions that vary only slowly and do not have sharp edges. This supports the use of quadratic regularization of the gradients the Tikhonov-regularized model in \([7]\). Specifically, instead of the iteratively regularized IRGN approach (1.4), the authors directly solve for \( \hat{v} = (\hat{u}, \hat{c}_1, \hat{c}_2, \ldots, \hat{c}_J) \) the variational model

\[
(1.5) \quad \min_{v = (u, c_1, c_2, \ldots, c_J)^T} \frac{1}{2} \sum_{j=1}^{J} \|PF(G(v))_j - g_j\|_2^2 + \alpha_0 R_u(u) + R_c(c),
\]

where \( G(v) = (uc_1, uc_2, \ldots, uc_J)^T \), \( R_u(u) = \text{TV}_I(u) \), and \( R_c(c) = \sum_{j=1}^{J} \alpha_j \|\nabla c_j\|_{2,2} \).

1.3 CONTRIBUTIONS

From the optimization point of view, a difficulty with both models (1.4) and (1.5) is the multiplicative nature of \( G(v) \). It can cause \( u \) and \( c_j \) to compete against each other. Therefore, besides physical considerations, one goal in the design of the regularisers \( R_u \) and \( R_c \) would be to try
to make $u$ and $c_j$ in some vague sense “orthogonal”, to avoid this competition. One approach to such vague orthogonality is to force $u$ piecewise constant and $c_j$ smooth. This is roughly performed by the TV and $H^1$ regularisers in [7]. Another approach is for $u$ and $c_j$ have very different sparsity structure. This is what we will do in this paper.

Specifically, we will assume that the coil sensitivities can be sparsely represented in a spherical function basis $\{f_l^+\}$, which we introduce in detail in Section 2. Then, with $c_j = \sum_{l=1}^{L} a_l^{(j)} f_l^+$, we will in the variational model (1.5) promote sparsity by taking

$$R_c(c) = \alpha R_a(a) \quad \text{for} \quad R_a(a) = \sum_{j=1}^{J} \sum_{l=1}^{L} |a_l^{(j)}|^2.$$

Therefore, we consider the model

$$\min_{v=(u,a)} \frac{1}{2} \sum_{j=1}^{J} \left\| P \mathcal{T}(G(v))_j - g_j \right\|^2 + \alpha_0 R_u(u) + \alpha R_a(a),$$

for an appropriate definition of $G$ that we provide in Section 4.

In order to make this model practical, we propose in Section 3 an efficient approach to compute the spherical basis functions based on spherical Bessel functions of the first kind and spherical harmonics. For the computation of spherical Bessel functions, we develop recurrence formulas. Based on the recurrence formula, all other spherical Bessel functions are efficiently calculated via the first two Bessel functions. For the computation of the spherical harmonics, we also provide a means to efficiently compute the associated Legendre functions by establishing in Section 3 a recurrence formula with only four terms. In Section 4 we then present a numerical method for (1.5) with (1.6), based on the alternating direction method of multipliers (ADMM), and study the practical reconstruction performance in Section 5.

2 SPHERICAL BASIS FUNCTION REPRESENTATION OF COIL SENSITIVITIES

According to the principle of reciprocity [27, 29, 30, 62], the coil sensitivity maps can be evaluated from transmit radio frequency field profiles $B_l^+$. We therefore start by briefly introducing the theory of $B_l^+$ fields. Let $\omega, \mu, \sigma,$ and $\epsilon$ denote the Larmour frequency, the magnetic permeability, the conductivity, and the dielectric permittivity of the material, respectively. The radio frequency (RF) field is denoted by $\vec{B}(\vec{r}) = (B_x(\vec{r}), B_y(\vec{r}), B_z(\vec{r}))^T$ with $\vec{B}(\vec{r}) \in \mathbb{C}^3$ and $\vec{r} \in \mathbb{R}^3$. In the positively rotating frame given in [27, 30, 50], the transmit RF field is

$$B_1^+(\vec{r}) \equiv \frac{B_x(\vec{r}) + iB_y(\vec{r})}{2}.$$

For the detailed introduction of positively rotating frame, see [27]. These fields can be approximated [18, 32, 43, 58] by

$$B_l^+(\vec{r}) \approx \sum_{l=1}^{L} a_l f_l^+(\vec{r}).$$
where \( f^+_1(\vec{r}) \) are the spherical basis functions, \( L \) is a small natural number and \( a_l \) are complex coefficients. The overall magnetic field \( \vec{B} \) can be reduced to the Helmholtz equation \([13]\)

\[
\nabla^2 \vec{B}(\vec{r}) + \zeta^2 \vec{B}(\vec{r}) = 0, \quad \text{where} \quad \zeta^2 = \epsilon \mu \omega^2 - i \sigma \omega \mu.
\]

In spherical coordinates \((\rho, \theta, \phi)\), the equation (2.2) has the solution \([31]\)

\[
(2.3) \quad B_x(\vec{r}) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \alpha_n^m f_n^m(\vec{r}), \quad B_y(\vec{r}) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \beta_n^m f_n^m(\vec{r}), \quad B_z(\vec{r}) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \gamma_n^m f_n^m(\vec{r}),
\]

where \( f_n^m \) are so-called spherical functions. They can be written

\[
(2.4) \quad f_n^m(\rho, \theta, \phi) = j_n(\xi \rho)Y_n^m(\theta, \phi),
\]

where \( j_n \) is the spherical Bessel function of the first kind of order \( n \), and \( Y_n^m \) is the spherical harmonic of order \( n \) and degree \( m \). The spherical functions form a basis for the \( B_l^+ \) fields by setting

\[
f_l^+(\rho, \theta, \phi) = f_n^m(\rho, \theta, \phi) \quad \text{with} \quad l = n^2 + n + m + 1, \ |m| \leq n \text{ and } 0 \leq n \leq \tilde{n}.
\]

When signals or objects of approximately spherical shape are considered, fast convergence is expected, so the complex coefficients \( \alpha_n^m, \beta_n^m, \) and \( \gamma_n^m \) in (2.3) should be negligible for \( n > \tilde{n} \) with \( \tilde{n} \) being a small natural number. We can therefore also expect fast convergence for the spherical function approximation of the \( B_l^+ \) field, and by extension the coil sensitivities.

### 3 EFFICIENT COMPUTATION OF THE SPHERICAL BASIS FUNCTIONS

Our task in the present section is develop efficient recurrence formulas for the computation of the spherical basis functions (2.4). As discussed, this will be based on formulas for the Bessel functions and spherical harmonics.

#### 3.1 A RECURRENCe RELATION FOR THE SPHERICAL BESSEL FUNCTIONS

Following \([1]\), we now develop a recurrence formula for the spherical Bessel functions \( j_n \). We start by recalling that the Bessel function of the first kind, for arbitrary order \( \alpha \in \mathbb{R} \), is defined as

\[
J_\alpha(x) = \sum_{s=0}^{\infty} \frac{(-1)^s}{s!(s+\alpha+1)!} \left( \frac{x}{2} \right)^{\alpha+2s}.
\]

Since the \( \Gamma \) satisfies \( \Gamma(n) = (n-1)! \) for integral \( n \geq 0 \) we can in particular write

\[
J_n(x) = \sum_{s=0}^{\infty} \frac{(-1)^s}{s!((n+s))!} \left( \frac{x}{2} \right)^{n+2s}.
\]

To compute \( J_n \) for negative integers, we can use the relationship

\[
J_{-n}(x) = (-1)^n J_n(x).
\]
In addition, we have the recurrence relationship [1]

\[ J_{n-1}(x) + J_{n+1}(x) = \frac{2n}{x} J_n(x). \]

We now finally define the spherical Bessel function

\[ j_n(x) = \sqrt{\frac{\pi}{2x}} J_{n+1/2}(x). \]

From the recurrence relation (3.1) for the Bessel functions \( J_n \), we then obtain

\[ j_{n-1}(x) + j_{n+1}(x) = \frac{2n + 1}{x} j_n(x). \]

This can more conveniently be rewritten

\[ j_{n+1}(x) = \frac{2n + 1}{x} j_n(x) - j_{n-1}(x). \]

This is a three-term recurrence relation, so if \( j_0 \) and \( j_1 \) are known, then any higher-order \( j_n \) can be computed from (3.2).

To compute \( j_0 \), we recall that Legendre’s duplication formula for the \( \Gamma \) function states

\[ \Gamma(1 + z)\Gamma(z + \frac{1}{2}) = 2^{-2z} \sqrt{\pi} \Gamma(2z + 1). \]

For integral \( z \) therefore

\[ z!(z + \frac{1}{2})! = 2^{-2z-1} \sqrt{\pi} (2z + 1)!. \]

Consequently

\[ j_n(x) = 2^n x^n \sum_{s=0}^{\infty} \frac{(-1)^s (s + n)!}{s! (2s + 2n + 1)!} x^{2s}. \]

When \( n = 0 \), we find from (3.3) that

\[ j_0(x) = \sum_{s=0}^{\infty} \frac{ (-1)^s }{ (2s)! x^{2s} } = \frac{\sin x}{x}. \]

To compute \( j_1 \), we set

\[ n_n(x) = (-1)^{n+1} \sqrt{\frac{\pi}{2x}} J_{n-1/2}(x), \quad \text{and} \quad h_n^{(1)}(x) = j_n(x) + in_n(x). \]

In [1] it is shown

\[ h_n^{(1)}(x) = (-i)^{n+1} \frac{e^{ix}}{x} \sum_{s=0}^{n} \frac{i^s (n + s)!}{s! (2x)^s (n - s)!}. \]
where the associated Legendre function $P_n(x)$ which for $n = 1$ gives

$$h_1^{(1)}(x) = e^{ix} \left( -\frac{1}{x} + \frac{i}{x^2} \right).$$

As the real part of $h_1^{(1)}(x)$, we then obtain for $j_1(x)$ the expression

$$j_1(x) = \frac{\sin x}{x^2} - \frac{\cos x}{x}.$$

Based on the recurrence (3.2) and the expressions (3.4) and (3.5) for $j_0$ and $j_1$, we can now compute any higher-order $j_n$. We list the first few in Table 1.

### 3.2 EFFICIENT COMPUTATION OF THE ASSOCIATED LEGENDRE FUNCTIONS

The spherical harmonics $Y_n^m$ are defined in spherical coordinates as [1]

$$Y_n^m(\theta, \phi) \equiv (-1)^m \sqrt{\frac{2n + 1}{4\pi} \frac{(n - m)!}{(n + m)!}} P_n^m(\cos \theta) e^{im\phi}, \quad 0 \leq \theta \leq \pi, 0 \leq \phi \leq 2\pi,$$

where the associated Legendre function

$$P_n^m(x) = (1 - x^2)^{m/2} \frac{d^m}{dx^m} P_n(x), \quad 0 \leq m \leq n,$$

and $P_n(x)$ are the $n$th-order Legendre polynomials. They are defined as

$$P_n(x) = \sum_{k=0}^{[n/2]} (-1)^k \frac{(2n - 2k)!}{2^mk!(n-k)!(n-2k)!} x^{n-2k}$$

with $[n/2] = n/2$ for $n$ even, $(n - 1)/2$ for $n$ odd. In particular, it is easy to see that

$$P_0(x) = 1 \quad \text{and} \quad P_1(x) = x.$$

For $-n \leq m < 0$, using Leibniz’ differentiation formula, we can find that $P_n^m(x)$ and $P_n^{-m}(x)$ are related by [1]

$$P_n^{-m}(x) = (-1)^m \frac{(n - m)!}{(n + m)!} P_n^m(x).$$
For $|m| > n$, $P_n^m(x) = 0$.

From (3.7), we have

$$P_n^0(x) = P_n(x).$$

Thus $P_0^0(x) = P_0(x) = 1$, $P_1^0(x) = P_1(x) = x$, $P_1^1(x) = (1 - x^2)^{1/2} \frac{d}{dx} P_1(x) = (1 - x^2)^{1/2}$. By (3.9), we obtain $P_1^{-1}(x) = -\frac{1}{2}(1 - x^2)^{1/2}$. For the sake of the convenience of developing the recurrence relation to find all the $P_n^m$ effectively, we put $P_1^{-1}$, $P_0^0$, $P_1^0$, $P_1^1$ together as

$$\begin{align*}
(3.10) & \quad P_0^0(x) = 1, \\
& \quad P_1^{-1}(x) = -\frac{1}{2}(1 - x^2)^{1/2}, \\
& \quad P_0^1(x) = x, \\
& \quad P_1^1(x) = (1 - x^2)^{1/2}.
\end{align*}$$

Let us define the polynomials $P_{s+m}^m(x) = P_{s+m}^m(x) = (1 - x^2)^{-m/2}$, $m \geq 0$. Then the generating function $[1, (12.83)]$

$$g_m(x, t) \equiv \frac{(2m)!}{2^mm!(1 - 2tx + t^2)^{m+1/2}} = \sum_{s=0}^{\infty} P_{s+m}^m(x)t^s.$$ 

Furthermore, following $[1, \S 12.5]$, we have the recursion

$$P_{s+m}^m(x) = 2xP_{s+m-1}^m(x) - P_{s+m-2}^m(x) + (2m - 1)P_{s+m-1}^{m-1}(x).$$

Therefore

\begin{align*}
P_{s+m}^m(x) &= (1 - x^2)^{-m/2}P_{s+m}^m(x) \\
&= (1 - x^2)^{-m/2}(2xP_{s+m-1}^m(x) - P_{s+m-2}^m(x) + (2m + 1)P_{s+m-1}^{m-1}(x)) \\
&= 2(1 - x^2)^{-m/2}xP_{s+m-1}^m(x) - (1 - x^2)^{m/2}P_{s+m-2}^m(x) \\
&\quad + (2m - 1)(1 - x^2)^{m/2}P_{s+m-1}^{m-1}(x) \\
&= 2(1 - x^2)^{m/2}x(1 - x^2)^{-m/2}P_{s+m}^m(x) - (1 - x^2)^{m/2}P_{s+m-2}^m(x) \\
&\quad + (2m - 1)(1 - x^2)^{m/2}P_{s+m-1}^{m-1}(x) \\
&= 2xP_{s+m-1}^m(x) - P_{s+m-2}^m(x) + (2m - 1)(1 - x^2)^{1/2}P_{s+m-1}^{m-1}(x).
\end{align*}$$

That is

$$\begin{align*}
(3.11) & \quad P_{s+m}^m(x) = 2xP_{s+m-1}^m(x) - P_{s+m-2}^m(x) + (2m - 1)(1 - x^2)^{1/2}P_{s+m-1}^{m-1}(x).
\end{align*}$$

Now, for $m \geq 0$, using (3.11), we can compute effectively all the $P_n^m(x)$ by starting with (3.10). For $m < 0$, we can then use (3.9). We list the first few Legendre functions are listed in Table 2. Recalling (3.6), we are in particular interested in the case $x = \cos \theta$, which we also list.

Using the recurrence (3.11) for $P_n^m(\cos \theta)$, and the explicit solutions in Table 2, we now easily find all the spherical harmonics $Y_n^m(\theta, \phi)$ by the formula (3.6). Table 3 shows some of the low-order ones.
Table 2: First few associated Legendre functions as functions of \( x \) and of \( x = \cos \theta \).

<table>
<thead>
<tr>
<th>( P_m^n(x) )</th>
<th>( \text{fn. of } x )</th>
<th>( \text{fn. of } \theta )</th>
<th>( P_m^n(x) )</th>
<th>( \text{fn. of } x )</th>
<th>( \text{fn. of } \theta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_0^0(x) )</td>
<td>1</td>
<td>1</td>
<td>( P_1^1(x) )</td>
<td>( -\frac{1}{2} (1 - x^2)^{1/2} )</td>
<td>( -\frac{1}{2} \sin \theta )</td>
</tr>
<tr>
<td>( P_1^1(x) )</td>
<td>( x )</td>
<td>( \cos \theta )</td>
<td>( P_1^1(x) )</td>
<td>(1 - ( x^2 ))^{1/2}</td>
<td>( \sin \theta )</td>
</tr>
<tr>
<td>( P_2^{-1}(x) )</td>
<td>( \frac{1}{2} (1 - x^2) )</td>
<td>( \frac{1}{2} \sin^2 \theta )</td>
<td>( P_2^{-1}(x) )</td>
<td>( -\frac{1}{4} x (1 - x^2)^{1/2} )</td>
<td>( -\frac{1}{2} \cos \theta \sin \theta )</td>
</tr>
<tr>
<td>( P_2^{1}(x) )</td>
<td>( \frac{1}{2} x^2 - \frac{1}{2} )</td>
<td>( \frac{3}{2} \cos^2 \theta - \frac{1}{2} )</td>
<td>( P_2^{1}(x) )</td>
<td>3(1 - ( x^2 ))^{1/2}</td>
<td>3 ( \cos \theta \sin \theta )</td>
</tr>
<tr>
<td>( P_2^{3}(x) )</td>
<td>3(1 - ( x^2 ))</td>
<td>3 ( \sin^2 \theta )</td>
<td>( P_3^{1}(x) )</td>
<td>( -\frac{1}{16} (1 - x^2)^{3/2} )</td>
<td>( -\frac{1}{4} \sin^3 \theta )</td>
</tr>
<tr>
<td>( P_3^{-2}(x) )</td>
<td>( \frac{1}{8} x (1 - x^2) )</td>
<td>( \frac{1}{4} \cos \theta \sin^2 \theta )</td>
<td>( P_3^{-2}(x) )</td>
<td>( -\frac{7}{8} (5x^2 - 1)(1 - x^2)^{1/2} )</td>
<td>( \frac{1}{2} (5 \cos^2 \theta - 1) \sin \theta )</td>
</tr>
<tr>
<td>( P_3^{0}(x) )</td>
<td>( \frac{1}{8} x (5x^2 - 3) )</td>
<td>( \frac{1}{2} \cos \theta (5 \cos^2 \theta - 3) )</td>
<td>( P_3^{0}(x) )</td>
<td>( \frac{3}{8} (5x^2 - 1)(1 - x^2)^{1/2} )</td>
<td>( \frac{1}{2} (5 \cos^2 \theta - 1) \sin \theta )</td>
</tr>
<tr>
<td>( P_3^{2}(x) )</td>
<td>15x(1 - ( x^2 ))</td>
<td>15 ( \cos \theta \sin^2 \theta )</td>
<td>( P_3^{2}(x) )</td>
<td>15(1 - ( x^2 ))^{3/2}</td>
<td>15 ( \sin^3 \theta )</td>
</tr>
</tbody>
</table>

Table 3: Some low-order spherical harmonics

\[
\begin{align*}
Y_0^0(\theta, \phi) &= \frac{1}{\sqrt{4\pi}} \\
Y_1^0(\theta, \phi) &= \frac{1}{\sqrt{4\pi}} \cos \theta \\
Y_1^{-1}(\theta, \phi) &= \sqrt{\frac{3}{4\pi}} \sin \theta e^{-i\phi} \\
Y_1^{1}(\theta, \phi) &= -\sqrt{\frac{3}{4\pi}} \sin \theta e^{i\phi} \\
Y_2^{-2}(\theta, \phi) &= \sqrt{\frac{15}{32\pi}} \sin^2 \theta e^{-2i\phi} \\
Y_2^{-1}(\theta, \phi) &= \sqrt{\frac{15}{8\pi}} \sin \theta \cos \theta e^{-i\phi} \\
Y_2^{1}(\theta, \phi) &= \sqrt{\frac{15}{8\pi}} \sin \theta \cos \theta e^{i\phi} \\
Y_2^{0}(\theta, \phi) &= \sqrt{\frac{15}{32\pi}} \sin^2 \theta e^{2i\phi} \\
Y_3^{-3}(\theta, \phi) &= \sqrt{\frac{35}{64\pi}} \sin^3 \theta e^{-3i\phi} \\
Y_3^{-1}(\theta, \phi) &= \sqrt{\frac{105}{32\pi}} \cos \theta \sin^2 \theta e^{-i\phi} \\
Y_3^{1}(\theta, \phi) &= \sqrt{\frac{35}{64\pi}} (5 \cos^2 \theta - 1) \sin \theta e^{-i\phi} \\
Y_3^{0}(\theta, \phi) &= \sqrt{\frac{7}{16\pi}} \cos \theta (5 \cos^2 \theta - 3) \\
Y_3^{3}(\theta, \phi) &= \sqrt{\frac{35}{64\pi}} \sin^3 \theta e^{3i\phi} \\
Y_3^{2}(\theta, \phi) &= \sqrt{\frac{105}{32\pi}} \cos \theta \sin^2 \theta e^{2i\phi}
\end{align*}
\]

3.3 A METHOD TO COMPUTE THE SPHERICAL BASIS FUNCTIONS

We recall te presentation of \( f_1^+ \) and the presentation (2.4) of \( f_n^m \). Because \( l = n^2 + n + m + 1 \) and \( |m| \leq n \), we find that \( l \) is bounded by \( l_{\text{max}}^{(n)} = (n+1)^2 \) for \( n \). From \( f_1^+(\rho, \theta, \phi) = f_n^m(\rho, \theta, \phi) \), we easily get for the low-order functions the relationships in Table 4. Using \( f_n^m(\rho, \theta, \phi) = j_n(\xi \rho) Y_m^n(\theta, \phi) \), we then obtain the basis functions \( f_1^+ \). Table 5 lists some \( f_1^+ \) corresponding to low-order \( n \).

From the above analysis, we find that if \( j_n \) and \( P_m^n \) are given only for \( n = 0, 1 \), then rather than directly evaluating the series (3.3) and (3.8), we can quickly find all the basis functions \( f_1^+ \) from (3.2) and (3.11). Starting with \( j_0(\xi \rho) = \frac{\sin(\xi \rho)}{\xi \rho}, j_1(\xi \rho) = \frac{\sin(\xi \rho)}{\xi \rho^2} - \frac{\cos(\xi \rho)}{\xi \rho}, P_0^0(\cos \theta) = 1, P_1^{-1}(\cos \theta) = -\frac{1}{2} \sin \theta, P_1^0(\cos \theta) = \cos \theta, P_1^1(\cos \theta) = \sin \theta \), we can compute all the \( f_1^+ \) with \( |m| \leq n, l = 1, 2, \ldots, l_{\text{max}}^{(n)} \). The numerical computation is outlined in Algorithm 1. In its practical application, the radius \( \rho \), the polar angle \( \theta \) and the azimuth angle \( \phi \) are computed by Cartesian coordinates \( (x, y, z) \), which are as follows:

\[
\rho_{l,j} = \sqrt{x_j^2 + y_j^2 + z_j^2},
\]

\[
\theta_{l,j} = \arccos\left(\frac{z_j}{\rho_{l,j}}\right),
\]

9
The $\ell$-field approximation (two.oldstyle/one.oldstyle) can be expressed in matrix-vector form

$$
\begin{align*}
&n = 0 & f_0^+ = j_0(\zeta \rho) Y_0^0(\theta, \phi), \\
n = 1 & f_1^+ = j_1(\zeta \rho) Y_1^0(\theta, \phi), f_2^+ = j_1(\zeta \rho) Y_1^1(\theta, \phi), \\
n = 2 & f_2^+ = j_2(\zeta \rho) Y_2^0(\theta, \phi), f_3^+ = j_2(\zeta \rho) Y_2^1(\theta, \phi), f_4^+ = j_2(\zeta \rho) Y_2^2(\theta, \phi), \\
n = 3 & f_3^+ = j_3(\zeta \rho) Y_3^0(\theta, \phi), f_4^+ = j_3(\zeta \rho) Y_3^1(\theta, \phi), f_5^+ = j_3(\zeta \rho) Y_3^2(\theta, \phi), f_6^+ = j_3(\zeta \rho) Y_3^3(\theta, \phi), \\
&\vdots & \vdots
\end{align*}
$$

where $z_0$ is fixed, $x_i, y_j$ are discrete results of $x, y, i, j = 1, 2, \ldots, N$. Hence

$$
(f_i^+)_{i,j} = j_n(\zeta \rho l_{i,j}) Y_n^m(\theta_{i,j}, \phi_{i,j}) \quad \text{with} \quad l = n^2 + n + m + 1, |m| \leq n.
$$

The $B_i^+$ field approximation (2.1) can be expressed in matrix-vector form $B_i^+ \approx FA$, where for $\ell = l_{\text{max}}^{(n)}$ we have

$$
F = \begin{pmatrix}
(f_i^+)_1,1 & (f_i^+)_1,2 & \ldots & (f_i^+)_1,n \\
(f_i^+)_2,1 & (f_i^+)_2,2 & \ldots & (f_i^+)_2,n \\
\vdots & \vdots & \ddots & \vdots \\
(f_i^+)_n,1 & (f_i^+)_n,2 & \ldots & (f_i^+)_n,n
\end{pmatrix}, \quad \text{and} \quad A = \begin{pmatrix}
a_1 \\
a_2 \\
\vdots \\
(a_1 a_2 \ldots a_{l_{\text{max}}^{(n)}})
\end{pmatrix}.
Algorithm 1 The computation for the basis functions $f_i^+$

Given $\omega, \mu, \sigma, \varepsilon, \rho, \theta, \phi$ and $\hat{n}$.

Calculate $\zeta = \sqrt{\mu \omega^2 - i\sigma \omega}$.

Initialize $j_0(\zeta \rho) = \frac{\sin(\zeta \rho)}{\zeta \rho}, j_1(\zeta \rho) = \frac{\sin(\zeta \rho)}{(\zeta \rho)^2} - \cos(\zeta \rho) \frac{1}{\zeta \rho}$, $P^{(0)}_0(\cos \theta) = 1, P^{(1)}_1(\cos \theta) = -\frac{1}{2} \sin \theta$.

$P^{(l)}_l(\cos \theta) = \cos \theta, P^{(l+1)}_l(\cos \theta) = \sin \theta$.

$Y_{l}^{0}(\theta, \phi) = \frac{1}{\sqrt{\pi \rho}}, Y_{l}^{1}(\theta, \phi) = \sqrt{\frac{3}{\pi \rho}} \sin \theta e^{-i\phi}$.

$Y_{l}^{0}(\theta, \phi) = \sqrt{\frac{2}{\pi}} \cos \theta, Y_{l}^{1}(\theta, \phi) = -\sqrt{\frac{2}{\pi}} \sin \theta e^{i\phi}$.

The computation for the basis functions $f_{l}^{\pm}$.

For $n = 2 : \hat{n}$

- calculate $j_0(\zeta \rho)$ and $P^{(m)}_m(\cos \theta)$ using (3.2) and (3.11), respectively.
- calculate $Y^{(m)}_{m}(\theta, \phi)$ with (3.6).
- calculate $f_{l}^{\pm} = j_0(\zeta \rho) Y^{(m)}_{m}(\theta, \phi), f_{l}^{\pm} = j_0(\zeta \rho) Y^{(m)}_{m}(\theta, \phi)$.

End

Return $\{f_{l}^{\pm} \mid l = 1, 2, \ldots, \hat{n}\}$.

4 THE NEW REGULARIZATION MODEL AND ITS NUMERICAL REALIZATION

We now present in detail our spherical function based regularization model for p-MRI reconstruction, as well as a method for its numerical realization.

4.1 REGULARIZATION BY SPARSE PRESENTATION IN SPHERICAL BASIS

Replacing the coil sensitivities $c_j$ and their regularization $R_c$ in the model (1.5) by the spherical functions representation $\sum_{l=1}^{\hat{l}} a_{l}^{(j)} f_{l}^{+}, j = 1, 2, \ldots, J$, and $R_{a}(a) = \alpha \sum_{j=1}^{J} \sum_{l=1}^{\hat{l}} |a_{l}^{(j)}|$, respectively, we obtain the model (1.7), that is

$$
\min_{u = (a; u)} \frac{1}{2} \sum_{j=1}^{J} \left\| P\mathcal{F}(G(u))_j - g_j \right\|_2^2 + \alpha_0 R_u(u) + \alpha R_a(a),
$$

where

$$
G(v) = \left( u \sum_{l=1}^{l_{\max}} a_{l}^{(1)}, u \sum_{l=1}^{l_{\max}} a_{l}^{(2)}, \ldots, u \sum_{l=1}^{l_{\max}} a_{l}^{(J)} f_{l}^{+} \right)^T, \quad R_{a}(a) = \sum_{j=1}^{J} \sum_{l=1}^{l_{\max}} |a_{l}^{(j)}|,
$$

and

$$
a = \left( a_{1}^{(1)}, a_{2}^{(1)}, \ldots, a_{l_{\max}}^{(1)}, a_{1}^{(2)}, a_{2}^{(2)}, \ldots, a_{l_{\max}}^{(2)}, \ldots, a_{1}^{(J)}, a_{2}^{(J)}, \ldots, a_{l_{\max}}^{(J)} \right)^T \in \mathbb{C}^{l_{\max} J}.
$$

For the image $u$, we use the same isotropic total variation regularization $R_u(u) = TV_{I}(u)$ as in the model (1.5) from [7], while our $R_a$ yields a different regularization $R_c$ for the coil sensitivities.
The complex numbers $a_i^{(j)}$ are the coefficients corresponding to the spherical function representation. The spherical basis functions are smooth enough to mainly encode low-frequency information for low $l$, so should not pick up important image features. Apart from this, we will not need to store the coil sensitives, which are themselves relatively high-dimensional images, or differentiate and perform their updates in a numerical optimization algorithm, as would be the case with the mode (1.7) in [7]. Instead we work with the relatively low-dimensional $a$. Thus the updated model can be expected to be effective for parallel MRI reconstruction. We will validate this with the numerical experiments in Section 5, but now we need to construct the optimization algorithm to solve the model (4.1), that is (1.7).

### 4.2 THE ADMM FOR CONVEX PROBLEMS

We now start building a method for solving variational problems of type (1.7), i.e., (4.1). Note that due to the structure of $G$, these problems are non-convex. We therefore follow the non-linear ADMM approach of [7], itself motivated by the non-linear primal–dual hybrid gradient method (modified; PDHGM) of [14, 54]. Here we simplify the derivations from [7] to our specific problem form, and squared Hilbert space distances in place of the general Bregman distances employed in [7].

To motivate the algorithm that we will use, we start by considering convex problems

$$\inf_{v \in V} \{E(v) + F(Bv)\},$$

where $E : V \rightarrow \mathbb{R}$ and $F : H \rightarrow \mathbb{R}$ are two proper lower semi-continuous convex functions, $B$ is a linear operator from $V$ into $H$, and $V$ and $H$ are the real Hilbert spaces equipped with the inner products $\langle \cdot, \cdot \rangle_V$ and $\langle \cdot, \cdot \rangle_H$. The augmented dual problem for (4.4) is

$$\sup_{\mu} \phi_\delta(\mu) := \inf_{v \in V} \left( E(v) + \inf_{q \in H} \left( F(q) + \langle \mu, Bv - q \rangle_H + \frac{\delta}{2} \| Bv - q \|_H^2 \right) \right).$$

where, $\delta > 0$, and $\phi_\delta : H \rightarrow [\mathbb{R}]^\ast$.

We can solve (4.4) and the dual (4.5) by finding on $V \times H \times H$ a saddle point of the augmented Lagrangian function defined by

$$\mathcal{L}_\delta(v, q, \mu) = F(q) + E(v) + \langle \mu, Bv - q \rangle_H + \frac{\delta}{2} \| Bv - q \|_H^2.$$

**Theorem 4.1.** If $\{u, p, \lambda\}$ is a saddle-point of $\mathcal{L}_\delta(v, q, \mu)$ on $V \times H \times H$ for $\delta > 0$, then $u$ is a solution of (4.4), and we have $p = Bu$.

**Proof.** The proof is a direct consequence of Theorem 2.1 in Chapter III [21].

In view of Theorem 4.1, in order to calculate the saddle-points of $\mathcal{L}_\delta(v, q, \mu)$ on $V \times H \times H$, we employ an algorithm of the Uzawa type: given $\lambda^0 \in H$, determine $\{u^{k+1}, p^{k+1}\}$ from

$$\min_{\{v, q\} \in V \times H} \mathcal{L}_\delta(v, q, \lambda^k),$$

and then update

$$\lambda^{k+1} = \lambda^k + \delta(Bu^{k+1} - p^{k+1}).$$

12
For more details we refer to [2]. The direct solution to (4.7) is due to the coupling of \( v \) and \( q \). For this reason, we are led to the Alternating Direction Method of Multipliers (ADMM) algorithm introduced in [2] to solve (4.7). This algorithm approximates the pair \((u^{k+1}, p^{k+1})\) for (4.7) via decoupled minimization over \( v \) and \( q \) as

\[
\begin{align*}
(u^{k+1}, p^{k+1}) & \in \text{argmin}_{v \in V} \mathcal{L}_\delta(v, p^k, \lambda^k), & p^{k+1} & \in \text{argmin}_{q \in H} \mathcal{L}_\delta(u^{k+1}, q, \lambda^k).
\end{align*}
\]

The ADMM algorithm for (4.4), described by (4.6), (4.8), and (4.9), can thus be summarized

\[
\begin{align*}
(u^{k+1}, p^{k+1}, \lambda^{k+1}) & \leftarrow (u^k, p^k, \lambda^k) \\
& \leftarrow \left( \text{argmin}_{v \in V} F(p^k) + E(v) + \langle \lambda^k, Bu^k - p^k \rangle_H + \frac{\delta}{2} \| Bu^k - p^k \|_H^2, \right. \\
& \left. \text{argmin}_{q \in H} F(q) + E(u^{k+1}) + \langle \lambda^{k+1}, Bu^{k+1} - q \rangle_H + \frac{\delta}{2} \| Bu^{k+1} - q \|_H^2, \right) \\
& \leftarrow (u^{k+1}, p^{k+1}, \lambda^{k+1}) \left( \text{argmin}_{v \in V} F(p^k) + E(v) + \langle \lambda^k, Bu^k - p^k \rangle_H + \frac{\delta}{2} \| Bu^k - p^k \|_H^2, \right. \\
& \left. \text{argmin}_{q \in H} F(q) + E(u^{k+1}) + \langle \lambda^{k+1}, Bu^{k+1} - q \rangle_H + \frac{\delta}{2} \| Bu^{k+1} - q \|_H^2, \right).
\end{align*}
\]

4.3 PROXIMAL MINIMIZATION

When \( V \) and \( H \) are \( \mathbb{R}^n \), the sub-problems (4.10a) and (4.10b) can be solved by the proximal minimization algorithm that we now describe. Let us consider the convex problem

\[
\min_{x \in \mathbb{R}^n} W(x),
\]

where \( W : \mathbb{R}^n \to [\text{-}, +\infty] \) is a proper, lower semi-continuous convex function. The Moreau–Yosida envelope of \( W(x) \) is defined as

\[
W_\rho(x) = \inf_{y \in \mathbb{R}^n} \left\{ W(y) + \frac{1}{2\rho} \| x - y \|^2 \right\}.
\]

As proved in [42], \( W_\rho(x) \) is convex and differentiable, and has the same set of minimizers, and the same optimal value, as \( W \). This leads to the proximal minimization algorithm proposed by Martinet in [40]. Namely, we solve (4.11) by iterating

\[
x^{k+1} = \arg\min_{x \in \mathbb{R}^n} \left\{ W(x) + \frac{1}{2\rho_k} \| x - x^k \|^2 \right\},
\]

where the initial point \( x^0 \in \mathbb{R}^n \), \( \{\rho_k\}_{k=1}^\infty \) is a sequence of positive numbers. The convergence of this algorithm has been proved by Rockafellar in [47, 48]. For more discussion on proximal methods, see [15, 38, 55].

4.4 PRECONDITIONED PROXIMAL MINIMIZATION

Following the ideas given in [60], we can improve the performance of the proximal minimization method (4.12) by preconditioning. Specifically, we pick some positive definite symmetric matrix \( Q \), and replace (4.12) by

\[
x^{k+1} = \arg\min_{x \in \mathbb{R}^n} \left\{ W(x) + \rho_k^{-1} \| x - x^k \|^2_Q \right\}.
\]
Here we define \( \| x - x^k \|_Q := \sqrt{(Q(x - x^k), x - x^k)} \). Thus, picking \( Q_k^h \) and \( Q_q^h \) positive semi-definite, and incorporating the corresponding Moreau–Yosida regularization into (4.10), we obtain the preconditioned ADMM (cf. [20, 61])

\[
\begin{align*}
    u^{k+1} &= \arg\min_{v \in \mathbb{R}^n} F(p^k) + E(v) + \langle \lambda^k, Bv - p^k \rangle + \frac{\delta}{2} \| Bu - p^k \|^2 + \rho_{k+1}^{-1} \| v - u^k \|^2, \\
    p^{k+1} &= \arg\min_{q \in \mathbb{R}^n} F(q) + E(u^{k+1}) + \langle \lambda^k, Bu^{k+1} - q \rangle + \frac{\delta}{2} \| Bu^{k+1} - q \|^2 + \rho_{k+1}^{-1} \| q - p^k \|^2, \\
    \lambda^{k+1} &= \lambda^k + \delta(Bu^{k+1} - p^{k+1}).
\end{align*}
\]

4.5 A COMPUTATIONAL METHOD FOR THE PROPOSED MODEL

In order to cast the model (1.7) in the preconditioned ADMM framework, let us first study the augmented Lagrangian for (1.7). For \( v = (u; a)^T \), we define

\[
B(v) := (G(v), \nabla u, a),
\]

where \( G \) and \( a \) are as in (4.2) and (4.3), respectively. We represent the image as a vector \( u \in \mathbb{R}^{N^2} \), and write \( \nabla u = (\nabla_1 u; \nabla_2 u) \in \mathbb{R}^{2N^2} \). with \( E = 0 \) and

\[
F(B(v)) = \frac{1}{2} \sum_{j=1}^J \| Pf_j(G(v)) - g_j \|^2 + \alpha_0 R_u(u) + \alpha R_a(a),
\]

the problem (1.7) has the form (4.4) except for \( B \) being nonlinear operator. It follows from the above analysis and the preceding sections that an augmented Lagrangian naturally associated with the problem (1.7) is given by

\[
\mathcal{L}_\delta(v, q, \mu) = F(q) + \langle \mu, B(v) - q \rangle + \frac{\delta}{2} \| B(v) - q \|^2.
\]

By analogue, following [7], we extend the preconditioned ADMM (4.13) to non-linear \( B \) as

\[
\begin{align*}
    u^{k+1} &= \arg\min_{v} F(p^k) + \langle \lambda^k, B(v) - p^k \rangle + \frac{\delta}{2} \| B(v) - p^k \|^2 + \rho_{k+1}^{-1} \| v - u^k \|^2, \\
    p^{k+1} &= \arg\min_{q} F(q) + \langle \lambda^k, B(u^{k+1}) - q \rangle + \frac{\delta}{2} \| B(u^{k+1}) - q \|^2 + \rho_{k+1}^{-1} \| q - p^k \|^2, \\
    \lambda^{k+1} &= \lambda^k + \delta(B(u^{k+1}) - p^{k+1}).
\end{align*}
\]

For our specific problem, the minimizations are over \( v \in \mathbb{R}^{N^2 + J_s \max} \), and \( q \in \mathbb{R}^{J_s N^2 + 2s N^2 + J_q \max} \).

To make the proximal minimizations (4.14a) and (4.14b) easier, we linearise the operator \( B \). Let \( \tilde{F}_1(v) = B(v) - p^k \) and \( \tilde{F}_2(q) = B(u^{k+1}) - q \). Since these functions are smooth,

\[
\tilde{F}_1(v) \approx \tilde{F}_1(u^k) + \tilde{F}_1'(u^k)(v - u^k), \quad \text{and} \quad \tilde{F}_2(q) \approx \tilde{F}_2(p^k) + \tilde{F}_2'(p^k)(q - p^k),
\]

\[14\]
We now study the reconstruction performance of \((/one./seven)\) in comparison to the model \((/one.five)\) from \([/seven]\).

We also set \(\vec{v}\) where still \(\tilde{\lambda}\) where

Algorithm 2 Linearised preconditioned ADMM for (1.7)

**Initialization** \(u^0, p^0, \lambda^0\) and \(\delta\).

Set \(\tilde{\lambda}^0 = \mu^0\).

while "stopping criterion is not satisfied" do

\(J^k_v = \hat{F}_1'(u^k)\)

Choose \(\tau^k_v\) such that \(\tau^k_v \delta < \frac{1}{\|J^k_v\|^2}\)

Compute \(u^{k+1}\) by (4.17)

Compute \(J^k_q = \hat{F}_2'(p^k)\)

Choose \(\tau^k_q\) such that \(\tau^k_q \delta < \frac{1}{\|J^k_q\|^2}\)

Compute \(p^{k+1}\) by (4.18)

Compute \(\lambda^{k+1}\) by (4.14c)

Compute \(\tilde{\lambda}^{k+1} = 2\lambda^{k+1} - \lambda^k\)

end while

**Return** \(u^k, p^k, \lambda^k\) and \(\tilde{\lambda}^k\)

where \(\hat{F}_1'(u^k)\) and \(\hat{F}_2'(p^k)\) are the Fréchet derivative of \(\hat{F}_1\) at \(u^k\) and \(\hat{F}_2\) at \(p^k\). For the sake of clarity, we set \(J^k_v := \hat{F}_1'(u^k)\), \(J^k_q := \hat{F}_2'(p^k)\), \(r^k_v := \hat{F}_1'(u^k)u^k - \hat{F}_1(u^k)\) and \(r^k_q := \hat{F}_2'(p^k)p^k - \hat{F}_2(p^k)\).

Using (4.15), (4.14a), and (4.14b) are replaced by

\[
(4.16a) \quad u^{k+1} = \arg\min_v \langle \lambda^k, J^k_q v \rangle + \frac{\delta}{2} \| J^k_q v - r^k_q \|^2 + \rho_{k+1}^{-1} \| v - u^k \|^2_{Q^k_v},
\]

\[
(4.16b) \quad p^{k+1} = \arg\min_q F(q) + \langle \lambda^k, J^k_q v \rangle + \frac{\delta}{2} \| J^k_q v - r^k_q \|^2 + \rho_{k+1}^{-1} \| q - p^k \|^2_{Q^k_q},
\]

where still \(v \in \mathbb{R}^{N^2 + J*|l|m}^\text{out}\), and \(q \in \mathbb{R}^{J* N^2 + 2N^2 + J*|l|m}^\text{out}\).

In order to simplify the linearisation (4.16a) and (4.16b), we will seek \(Q^k_v\) and \(Q^k_q\) by \(J^k_v\) and \(J^k_q\).

For \(\tau^k_v \delta < \frac{1}{\|J^k_v\|^2}\) and \(\tau^k_q \delta < \frac{1}{\|J^k_q\|^2}\), we specifically let \(Q^k_v := \tau^k_v I - \delta J^k_v J^k_v\) and \(Q^k_q := \tau^k_q I - \delta J^k_q J^k_q\).

We also set \(\tilde{\lambda}^k := 2\lambda^k - \lambda^{k-1}\) and \(\rho_{k+1} = 2\). It follows easily from (4.16a) and (4.16b)

\[
(4.17) \quad u^{k+1} = u^k - \tau^k_v J^k_v^* \tilde{\lambda}^k,
\]

\[
(4.18) \quad p^{k+1} = \left( I + \tau^k_q J^k_q \right)^{-1} \left( p^k - \tau^k_q J^k_q \lambda^k \right) + \delta \left( B(u^{k+1}) - p^k \right).
\]

Finally, (4.17), (4.18) and (4.14c) yields Algorithm 2 for the solution of (1.7). Its convergence is studied in [7] based on the results of [54].

5 NUMERICAL EXPERIMENTS

We now study the reconstruction performance of (1.7) in comparison to the model (1.5) from [7].
5.1 TECHNICAL DETAILS

In the numerical simulations, we introduce regularization parameters $\alpha_j$ into the fidelity term $\frac{1}{2} \sum_{j=1}^{J} \| P^T (G(v))_j - g_j \|_2^2$ in the model (1.7). With this, the objective functional becomes

$$F(Bv) = \frac{1}{2} \sum_{j=1}^{J} \alpha_j \| P^T (G(v))_j - g_j \|_2^2 + \alpha_0 R_u(u) + \alpha R_d(a)$$

$$= \frac{1}{2} \sum_{j=1}^{J} \alpha_j \| P^T (G(v))_j - g_j \|_2^2 + \alpha_0 R_u(u) + \alpha \sum_{j=1}^{J} \sum_{l=1}^{l_{\text{max}}} |a_l^{(j)}|.$$ 

We decompose $F(Bv)$ into $F_j(v) = \frac{1}{2} \alpha_j \| P^T (G(v))_j - g_j \|_2^2$ for $j = 1, 2, \ldots, J$, as well as $F_{j+1}(u) = \alpha_0 R_u(u)$, and $F_{j+2}(a) = \alpha \sum_{j=1}^{J} \sum_{l=1}^{l_{\text{max}}} |a_l^{(j)}|$. That is $F = \sum_{j=1}^{J} F_j + F_{j+1} + F_{j+2} = \sum_{j=1}^{J+2} F_j$. We compute the corresponding resolvents explicitly as

$$\left( I + \gamma_k \partial F_j \right)^{-1} (x) = \mathcal{T}^{-1} \left( \frac{\mathcal{T} x_j + \alpha_0 \tau_q^k \mathcal{P}^T g_j}{1 + \alpha_0 \tau_q^k \text{diag}(\mathcal{P}^T \mathcal{P})} \right), \quad j = 1, 2, \ldots, J,$$

$$\left( I + \gamma_k \partial F_{j+1} \right)^{-1} (x) = \frac{x_{j+1}}{TV_l(x_{j+1})} \max \left( TV_l(x_{j+1}) - \alpha_0 \tau_q^k, 0 \right),$$

$$\left( I + \gamma_k \partial F_{j+2} \right)^{-1} (x) = \frac{(x_{j+2})_i}{(x_{j+2})_i} \max \left( (x_{j+2})_i - \alpha \tau_q^k, 0 \right), \quad i = 1, 2, \ldots, J * l_{\text{max}}^2.$$ 

5.2 EXPERIMENTAL SETUP

Our numerical experiments are based on the synthetic brain phantom from [3, 8], depicted in Section 5.2 and of dimension $190 \times 190$. It contains several tissues, such as cerebrospinal fluid (CSF), gray matter (GM), white matter (WM) and cortical bone. In the numerical simulations, we set the number of coils $J = 8$. For the generation of $k$-space measurement data $g_j$, $j = 1, 2, \ldots, J$, we use the approach of [7]. We generate 8 coil sensitivity maps, based on a measurement of a water bottle with an 8-channel head coil array. These measurements are in Figure 6. We then multiply the brain phantom with each of these coil sensitivity maps separately, and convert the result to $k$-space data with the Fourier transform. Then we apply the 25% subsampling mask shown in Section 5.2. Finally, we add Gaussian noise with standard deviation $\sigma$ to the sub-sampled data.

We also demonstrate the robustness of the proposed approach in the p-MRI reconstruction by perturbing the coil sensitivity maps obtained from the water bottle. This is done by adding the 1st spherical basis function multiplied by factor $\gamma = 4$ to the water bottle measurements. The resulting maps are shown in Figure 12.

In numerical experiments, for the number of spherical basis functions “levels”, we choose either $n = 2$ or $n = 5$. So the number of spherical basis functions is either $l_{\text{max}} = 9$ or $l_{\text{max}} = 36$. In the Cartesian coordinate system, we set $z_0 = 0.5$, and $x, y$ in MATLAB are discretised by

$$\frac{2 \times \text{step} \times (1 : 190)}{190} - 10,$$
We perform a fixed number of iterations of Algorithm two.oldstyle. With the noise level $\tilde{\sigma} = n\text{step}$ where $n\text{step}$ = 2. All algorithms have been implemented in MATLAB, and the test hardware is an Intel Core i7-6700 HQ CPU 2.60GHz with 8GB RAM. We evaluate the results of the proposed approach in terms of the peak signal-to-noise (PSNR) that is available in the image processing toolbox in MATLAB and the Structural SIMilarity (SSIM) given in [57]. In the computation of the spherical basis function $f_j^*$, we use the Larmour frequency $\omega = 42.58$. The conductivity $\sigma$ and the dielectric permittivity $\varepsilon$ are the optimal $(\sigma, \varepsilon)$ for the heterogeneous model in [50] with $\sigma = 0.6, \varepsilon = 50$. The magnetic permeability for water is $\mu = 1.2566 \times 10^{-6}$.

### 5.3 Quality Measures and Parameter Selection

All algorithms have been implemented in MATLAB, and the test hardware is an Intel Core i7-6700 HQ CPU 2.60GHz with 8GB RAM. We evaluate the results of the proposed approach in terms of the peak signal-to-noise (PSNR) that is available in the image processing toolbox in MATLAB and the Structural SIMilarity (SSIM) given in [57]. In the computation of the spherical basis function $f_j^*$, we use the Larmour frequency $\omega = 42.58$. The conductivity $\sigma$ and the dielectric permittivity $\varepsilon$ are the optimal $(\sigma, \varepsilon)$ for the heterogeneous model in [50] with $\sigma = 0.6, \varepsilon = 50$. The magnetic permeability for water is $\mu = 1.2566 \times 10^{-6}$.

### 5.4 Numerical Reconstructions and Comparison Between (1.7) and (1.5)

We perform experiments with the noise level $\tilde{\sigma} = 0.05$. We initialize the Algorithm 2 with $u^0 = 0 \in \mathbb{R}^{N^2}, a^0 = (1, \ldots , 1) \in \mathbb{R}^{J_0^{\text{init}}}$, and take as step lengths $\tau_k^p = 1/8, \tau_q^k = 23, \text{ and } \delta = 1/24$ and the remaining $\sigma^0, \lambda^0, \bar{\lambda}^0$ are all also initialized to zero in the two algorithms. For numerical reconstruction corresponding to (1.5), we use the codes from [7], available from [6].

We take as regularization parameters $\alpha_j = 0.4018, (j = 1, \ldots , J), \alpha_0 = 0.0062$ and $\alpha = 0.2149$. We perform a fixed number of iterations of Algorithm 2. With $\hat{n} = 2$, i.e., $l_{\text{max}} = 9$, stopping after 1000, 1200, and 1500 iterations, the reconstruction results for the model (1.7) are shown in Figure 3. We also perform the numerical simulations with $\hat{n} = 5$, and the same number of iterations 1000, 1200, and 1500. The reconstruction results for the model (1.7) are shown in Figure 4, and for the model (1.5) in Figure 5. In Table 6 we report the PNSR and SSIM [57] values. From these results we can observe that the reconstruction quality of the model (1.7) is much better than the model (1.5) when iterations are 1000, 1200, and 1500. The reconstruction results can be further improved due to the regularization parameters $\alpha$ not being optimally chosen; for a truly fair comparison of the potential of the two models distinct, parameter learning strategies

![Figure 1: (a) shows the brain phantom described in Section 5.2. (b) shows the spiral-shaped 25% $k$-space sub-sampling mask.](image)

where $\text{step} = 10$. We plot in Figure 2 the first 36 spherical basis functions corresponding to $\hat{n} = 5$. For $\hat{n} = 2$ only the 9 first are used.
should be used \([16, 17]\). What we can with reasonable confidence say based on our experiments here is that the non-linear ADMM converges faster for the model \((1.7)\) than for \((1.5)\). This is important in practical applications.

We also report the absolute values of the coefficients \(a^{(j)}_l\) in Table 7 when the stopping number of iterations is 1500 for \(\tilde{n} = 2\). Similarly, Table 8 shows the absolute values for the resulted coefficients \(a^{(j)}_l\) for \(\tilde{n} = 5\) when we stop at 1500 iterations. While for \(\tilde{n} = 2\) the last rows of the coefficient pyramid for each coil still have high coefficient values, for \(\tilde{n} = 5\) the coefficients on the last row have decayed to below 1% of the main coefficient on the first row; often 0.1% or less. This supports our starting intuition that a sparse approximation of the coil sensitivities with relatively few coefficients is sufficient for a high-quality reconstruction.

Using the discovered coefficients \(a^{(j)}_l\), and the known spherical basis functions, we can recon-
Figure 3: Reconstructed brain slice using (1.7) and $n = 2$.

Figure 4: Reconstructed brain slice using (1.7) and $n = 5$.

To test robustness, we show in Figures 10 and 11 for $n = 2$ and $n = 5$, respectively, the reconstructions results for the alternative coil sensitivity maps in Figure 12. The number of iterations is 1500. The PSNR and SSIM values are reported in Table 10. Comparing to Figures 3 and 4 and Table 6, we can see that the results remain stable under this perturbation of coil sensitivities, being virtually identical. By contrast, the reconstructed coil sensitivities in Figures 13 and 14 have changed, corresponding to the change in true coil sensitivities.
In this paper, we have established a new model for parallel MRI reconstruction based on sparse regularization of coil sensitivities in spherical basis function bases. We have developed efficient recurrence formulas for the computation of these functions. We have then applied the nonlinear ADMM from [7] to numerically solve our model (1.7). By numerical reconstructions and comparison between (1.7) and (1.5), we think that the reconstruction quality for proposed model (1.7) is better than the model (1.5). In addition, the reconstruction for our model (1.7) for the alternative coils sensitivity maps is very robust. That has an important significance in practical applications. In the future, we will study the optimal choice among the regularization parameters $\alpha_j$, $\alpha_0$, and $\alpha$ to improve reconstruction quality furthermore via parameter learning strategies in [16, 17].

ACKNOWLEDGMENTS

Y. Zhu was supported by the National Natural Science Foundation of China No. 11571325, Science Research Project of CUC No. 3132016XNL1612. Towards the end of this research, T. Valkonen.
Table 7: The absolute values of coefficients of $f_1^+$ with $\tilde{n} = 2$ for 1500 iterations. $j$ is the ordinal number of coils.

<table>
<thead>
<tr>
<th>$j$</th>
<th>$f_1^+$, $f_2^+$, $f_3^+$, $f_4^+$</th>
<th>$f_1^+$, $f_2^+$, $f_3^+$, $f_4^+$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f_5^+$, $f_6^+$, $f_7^+$, $f_8^+$</td>
<td>$f_5^+$, $f_6^+$, $f_7^+$, $f_8^+$</td>
</tr>
<tr>
<td>1</td>
<td>5.4801 0.9382, 0.0092, 4.7704 1.0213, 0.0075, 5.3347, 0.0034, 3.8693</td>
<td>3.4487 0.2859, 0.0096, 3.6825 1.1578, 0.0075, 3.1660, 0.0059, 3.2907</td>
</tr>
<tr>
<td>2</td>
<td>4.8840 0.3632, 0.0059, 4.7055 1.2463, 0.0085, 4.6682, 0.0083, 4.3803</td>
<td>3.978 0.1909, 0.0085, 3.4326 0.6340, 0.0060, 3.7131, 0.0048, 3.2575</td>
</tr>
<tr>
<td>3</td>
<td>3.4124 0.2777, 0.0098, 4.3643 1.7627, 0.0072, 2.2884, 0.0066, 4.1581</td>
<td>3.7714 0.2415, 0.0016, 3.0790 1.4061, 0.0070, 4.1169, 0.0045, 3.8595</td>
</tr>
<tr>
<td>4</td>
<td>2.6890 0.1031, 0.0094, 4.2809 1.4676, 0.0065, 2.5603, 0.0058, 3.7494</td>
<td>3.3760 0.2552, 0.0065, 3.2423 1.6636, 0.0093, 2.7457, 0.0075, 3.7089</td>
</tr>
</tbody>
</table>

has been supported by the EPSRC First Grant EP/P021298/1, “PARTIAL Analysis of Relations in Tasks of Inversion for Algorithmic Leverage”.

We would like to thank Florian Knoll for the water bottle measurement, and Martin Benning for making his codes [6] available.

A DATA STATEMENT FOR THE EPSRC

All data and source codes will be publicly deposited when the final accepted version of the manuscript is submitted.

REFERENCES


Figure 6: Coil sensitivities generated by the measurements of a water bottle with 8-channel head coil array.


Figure 7: Reconstructed coil sensitivities for the 8 different coils when the number of iterations is 1500 for $\tilde{n} = 2$.


Reconstructed coil sensitivities for the 8 different coils when the number of iterations is 1500 for $\bar{n} = 5$.


Figure 9: Reconstructed coil sensitivities for the 8 different coils when the number of iterations is 500 for (1.5).

(a) 1200 iterations  (b) 1500 iterations  (c) 1800 iterations

Figure 10: Reconstruction when data is generated with the alternative coil sensitivities from Figure 12 with $\tilde{n} = 2$.


Figure 11: Reconstruction when data is generated with the alternative coil sensitivities from Figure 12 with $\tilde{n} = 5$.

Figure 12: Alternative Perturbed coil sensitivity maps.


Figure 13: Reconstructed coil sensitivities when data is generated with the alternative coil sensitivities from Figure 12. The number of iterations is 1500 and $\tilde{n} = 2$.


Reconstructed coil sensitivities when data is generated with the alternative coil sensitivities from Figure 12. The number of iterations is 1500 and $\tilde{n} = 5$.


Table 8: The absolute values of coefficients of $f_j^*$ with $\bar{n} = 5$ for 1500 iterations. $j$ is the ordinal number of coils.

| j   | $f_1^*$ | $f_2^*$ | $f_3^*$ | $f_4^*$ | $f_5^*$ | $f_6^*$ | $f_7^*$ | $f_8^*$ | $f_9^*$ | $f_{10}^*$ | $f_{11}^*$ | $f_{12}^*$ | $f_{13}^*$ | $f_{14}^*$ | $f_{15}^*$ | $f_{16}^*$ | $f_{17}^*$ | $f_{18}^*$ | $f_{19}^*$ | $f_{20}^*$ | $f_{21}^*$ | $f_{22}^*$ | $f_{23}^*$ | $f_{24}^*$ | $f_{25}^*$ | $f_{26}^*$ | $f_{27}^*$ | $f_{28}^*$ | $f_{29}^*$ | $f_{30}^*$ |
|-----|---------|---------|---------|---------|---------|---------|---------|---------|---------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| 5   |        |         |         |         |        |         |         |         |         |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |
| 6   |        |         |         |         |        |         |         |         |         |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |
| 7   |        |         |         |         |        |         |         |         |         |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |
| 8   |        |         |         |         |        |         |         |         |         |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |          |
### Table 9: Reconstruction quality comparison between (1.7) with $\tilde{n} = 2, 5$ and (1.5) for 1500 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>coil No.</th>
<th>PSNR (dB)</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>using (1.7) with $\tilde{n} = 2$</td>
<td>coil 1</td>
<td>8.5038</td>
<td>0.9864</td>
</tr>
<tr>
<td></td>
<td>coil 2</td>
<td>8.9329</td>
<td>0.9878</td>
</tr>
<tr>
<td></td>
<td>coil 3</td>
<td>8.9895</td>
<td>0.9873</td>
</tr>
<tr>
<td></td>
<td>coil 4</td>
<td>10.7526</td>
<td>0.9918</td>
</tr>
<tr>
<td></td>
<td>coil 5</td>
<td>10.2633</td>
<td>0.9900</td>
</tr>
<tr>
<td></td>
<td>coil 6</td>
<td>10.6595</td>
<td>0.9917</td>
</tr>
<tr>
<td></td>
<td>coil 7</td>
<td>9.0644</td>
<td>0.9869</td>
</tr>
<tr>
<td></td>
<td>coil 8</td>
<td>11.2573</td>
<td>0.9929</td>
</tr>
<tr>
<td>using (1.7) with $\tilde{n} = 5$</td>
<td>coil 1</td>
<td>7.2807</td>
<td>0.9849</td>
</tr>
<tr>
<td></td>
<td>coil 2</td>
<td>8.2247</td>
<td>0.9878</td>
</tr>
<tr>
<td></td>
<td>coil 3</td>
<td>7.4537</td>
<td>0.9850</td>
</tr>
<tr>
<td></td>
<td>coil 4</td>
<td>9.3979</td>
<td>0.9903</td>
</tr>
<tr>
<td></td>
<td>coil 5</td>
<td>9.1837</td>
<td>0.9891</td>
</tr>
<tr>
<td></td>
<td>coil 6</td>
<td>10.3233</td>
<td>0.9919</td>
</tr>
<tr>
<td></td>
<td>coil 7</td>
<td>8.5967</td>
<td>0.9876</td>
</tr>
<tr>
<td></td>
<td>coil 8</td>
<td>10.1097</td>
<td>0.9920</td>
</tr>
<tr>
<td>using (1.5)</td>
<td>coil 1</td>
<td>5.9826</td>
<td>0.9759</td>
</tr>
<tr>
<td></td>
<td>coil 2</td>
<td>5.9328</td>
<td>0.9754</td>
</tr>
<tr>
<td></td>
<td>coil 3</td>
<td>5.9124</td>
<td>0.9748</td>
</tr>
<tr>
<td></td>
<td>coil 4</td>
<td>6.3114</td>
<td>0.9779</td>
</tr>
<tr>
<td></td>
<td>coil 5</td>
<td>6.1844</td>
<td>0.9762</td>
</tr>
<tr>
<td></td>
<td>coil 6</td>
<td>6.1754</td>
<td>0.9765</td>
</tr>
<tr>
<td></td>
<td>coil 7</td>
<td>6.3423</td>
<td>0.9772</td>
</tr>
<tr>
<td></td>
<td>coil 8</td>
<td>6.4319</td>
<td>0.9774</td>
</tr>
</tbody>
</table>

### Table 10: The PSNR and SSIM values for reconstruction with alternative coil sensitivities using (1.7) with $\tilde{n} = 2$ and $\tilde{n} = 5$

<table>
<thead>
<tr>
<th>Method</th>
<th>stopping itr. k</th>
<th>PSNR (dB)</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>using (1.7) with $\tilde{n} = 2$</td>
<td>1200</td>
<td>27.0202</td>
<td>0.9998</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>24.2239</td>
<td>0.9995</td>
</tr>
<tr>
<td></td>
<td>1800</td>
<td>24.9633</td>
<td>0.9996</td>
</tr>
<tr>
<td>using (1.7) with $\tilde{n} = 5$</td>
<td>1200</td>
<td>30.8382</td>
<td>0.9999</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>30.7620</td>
<td>0.9999</td>
</tr>
<tr>
<td></td>
<td>1800</td>
<td>30.6448</td>
<td>0.9999</td>
</tr>
</tbody>
</table>