PAPER

GPU-accelerated parallel image reconstruction strategies for magnetic particle imaging

IOP Publishing Phys. Med. Biol. 69 (2024) 135005 https://doi.org/10.1088/1361-6560/ad5510

Physics in Medicine & Biology

IPEM Institute of Physics and Engineering in Medicine

RECEIVED
26 December 2023

REVISED
3 June 2024

ACCEPTED FOR PUBLICATION
6 June 2024

PUBLISHED
24 June 2024

PAPER

GPU-accelerated parallel image reconstruction strategies for magnetic particle imaging

Klaus N Quelhas[1,2,*], **Mark-Alexander Henn**[3,4], **Ricardo Farias**[2], **Weston L Tew**[1] and **Solomon I Woods**[1]

[1] Physical Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, United States of America
[2] Systems Engineering and Computer Science Program, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
[3] Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, United States of America
[4] University of Maryland, College Park, MD, United States of America
[*] Author to whom any correspondence should be addressed.

E-mail: klaus.quelhas@nist.gov

Keywords: magnetic particle imaging, image reconstruction, parallel computing, graphical processing unit, CUDA.

Abstract

Objective. Image reconstruction is a fundamental step in magnetic particle imaging (MPI). One of the main challenges is the fact that the reconstructions are computationally intensive and time-consuming, so choosing an algorithm presents a compromise between accuracy and execution time, which depends on the application. This work proposes a method that provides both fast and accurate image reconstructions. *Approach*. Image reconstruction algorithms were implemented to be executed in parallel in *graphics processing units* (GPUs) using the CUDA framework. The calculation of the model-based MPI calibration matrix was also implemented in GPU to allow both fast and flexible reconstructions. *Main results*. The parallel algorithms were able to accelerate the reconstructions by up to about 6,100 times in comparison to the serial Kaczmarz algorithm executed in the CPU, allowing for real-time applications. Reconstructions using the OpenMPIData dataset validated the proposed algorithms and demonstrated that they are able to provide both fast and accurate reconstructions. The calculation of the calibration matrix was accelerated by up to about 37 times. *Significance*. The parallel algorithms proposed in this work can provide single-frame MPI reconstructions in real time, with frame rates greater than 100 frames per second. The parallel calculation of the calibration matrix can be combined with the parallel reconstruction to deliver images in less time than the serial Kaczmarz reconstruction, potentially eliminating the need of storing the calibration matrix in the main memory, and providing the flexibility of redefining scanning and reconstruction parameters during execution.

1. Introduction

Magnetic particle imaging (MPI) is an emerging technique that allows visualizing the concentration of magnetic nanoparticles (MNPs) in space (Gleich and Weizenecker 2005). It was conceived as a biomedical imaging method, complementary to more traditional methods like magnetic resonance imaging (MRI) and computed tomography (CT), exhibiting significant advantages in terms of spatial resolution, sensitivity, acquisition time and absence of ionizing radiation (Knopp and Buzug 2012). More recently, MPI techniques found use in several other applications, either passive, like targeted imaging and thermometry, or active, like targeted drug delivery and magnetic hyperthermia (Weaver *et al* 2009, Knopp *et al* 2017, Rao *et al* 2019, Tay *et al* 2019, Bui *et al* 2023).

MPI works by remotely detecting the nonlinear magnetization response of MNPs exposed to static and oscillating magnetic fields. Properly placed static gradient fields generated by pairs of permanent magnets or electromagnets force the orientation of the MNPs within it to saturation, except in the region where these static fields cancel each other. In this region, the MNPs can respond to an oscillating excitation field, and their magnetization can be detected by pickup coils. Such a region is called *field-free point* (FFP) or *field-free*

line (FFL), depending on the configuration of the gradient fields. The FFP (or FFL) allows the selective measurement of the particles within it, and with additional magnetic fields or mechanical movement for spatial encoding, it is possible to measure the MNPs in one, two or three dimensions.

Image reconstruction is an essential step in MPI. The voltage signals measured by the pickup coils are translated into discretized representations of the true distributions of the MNPs in space. Over the years that followed the development of MPI, the two dominant image reconstruction methods in the literature were the *x-space* and the *system function*. The *x-space* is an analytical method that takes assumptions like uniform drive fields, linear gradient fields and uniform detection coil sensitivities, among others, to derive a simplified model that allows for the direct mapping of the measured signal in space (i.e. gridding), providing estimates of the spatial concentration by properly scaling the signal (Goodwill and Conolly 2010, 2011). The reconstructed images, however, are still blurred due to the effects of the *point spread function* (PSF), which represents the interaction between the gradient field and the MNP. This interaction can include relaxation effects, since the particles respond to the applied field in a finite time (Croft *et al* 2012). Both effects can be removed through deconvolution, by solving linear systems of equations (Henn *et al* 2022).

The *system function* method is based on a calibration matrix (or *system matrix*) that contains the characteristic response of the MNPs to a specific field sequence generated by a particular MPI system. The field sequence usually is such that the FFP (or FFL) scans the measurement field of view (FOV) following a 2D or 3D Lissajous trajectory. The calibration matrix is obtained by measuring the response of a small calibration sample to this field sequence in each position within the FOV, which correspond to the *voxels* of the reconstructed image (Knopp *et al* 2009). Since the calibration matrix accounts for the effects of the PSF and particle relaxation, as well as for imperfections in the measurement systems, the method provides state-of-the-art reconstruction accuracy. The cost for that is a time consuming calibration process, and the solution of massive systems of equations. The modeling of MPI scanners, their various components and how they interact (Knopp *et al* 2010a, 2010b, Quelhas *et al* 2022) allows generating model-based calibration matrices that save the calibration time, but they still are less accurate than the measured ones. Nevertheless, it is shown in Grüttner *et al* (2013) that both *x-space* and *system function* models are mathematically equivalent, since they are based on the same imaging equation, while it has been shown in Schomberg (2010) and März and Weinmann (2016) that data obtained through any arbitrary (but known) trajectory can be reconstructed by organizing this data in a grid in space and then performing a deconvolution.

Regardless of the chosen method, optimal reconstructions require solving systems of equations of the form $Ax = b$, where A is either the calibration matrix or a convolution kernel representing the PSF or the particle relaxation, b is the measured signal or the gridded representation of it, and x is the unknown concentration vector. Depending on the measurement parameters and the desired resolution, the size of A in the memory can easily reach several gigabytes, which is a serious limitation factor. For model-based approaches, the cost of computing the calibration matrix is of the same order as the reconstruction itself or even higher, which requires the storage of the matrix in the memory (RAM or disk) and prevents changes in the scanning and reconstruction parameters, such as bandwidth, temperature and resolution. In other words, any change in the measurement setup requires a new calibration matrix to be measured or calculated. Additionally, the presence of noise in b makes this system severely ill-posed, requiring the use of techniques such as Tikhonov regularization. Over the years, several algorithms have been studied to provide the most accurate and efficient reconstructions (Knopp *et al* 2008, 2010, Goodwill and Conolly 2011), and amongst the most frequently used in the literature we can mention (not in order of importance, and not limited to) the singular value decomposition (SVD) (Weizenecker and Borget 2007, Knopp *et al* 2008, Vogel *et al* 2016), the Kaczmarz algorithm (Weizenecker *et al* 2009, Knopp *et al* 2010a, 2010b) and the conjugate gradient (CG) method (Knopp *et al* 2009, 2010). The speed of the algorithms and convergence rates are of critical importance when pursuing real-time image reconstructions (Weizenecker *et al* 2009, Lampe *et al* 2012, Knopp and Hofmann 2016, Vogel *et al* 2016, 2017), and in this context, the simplicity and fast-converging properties of the Kaczmarz algorithm resulted in this method being the most adopted one within the MPI community.

In the last few years, a number of authors suggested that one strategy for obtaining high resolution images with high throughput rates should be through the acceleration of the reconstruction algorithms by implementing them to be executed in parallel in *graphics processing units* (GPUs) (Storath *et al* 2017, Vogel *et al* 2017, Chen *et al* 2021). GPU-accelerated algorithms have been employed for other biomedical imaging modalities like MRI and CT (Stone *et al* 2008, Eklund *et al* 2013, Flores *et al* 2014), but no advances in this field have been reported for MPI until recently, when we presented parallel implementations of image reconstruction algorithms, including the Kaczmarz algorithm and the conjugate gradient normal residual

(CGNR) with no regularization in GPU using the CUDA[5] (*Compute Unified Device Architecture*) framework (Quelhas *et al* 2023) for reconstructing 2D images from simulated signals without noise. The study demonstrated that the Kaczmarz algorithm shows poor performance when executed in parallel, and that the parallel implementation of the CGNR algorithm for 2D images resulted in a *speedup* of 10 × when compared to the serial implementation of Kaczmarz, complementing the results reported in a previous study on the subject (Elble *et al* 2009). The poor performance of the parallel Kaczmarz algorithm reported in our first publication has already been confirmed in Shen *et al* (2023), where a parallel implementation has been developed for accelerating multi-frame batch reconstructions. More recently, it has also been demonstrated that a similar implementation applied to 3D temperature reconstructions without noise using the *multi-color* method resulted in executions 40× faster than the ones employing the serial implementation (Quelhas *et al* 2023b).

This work demonstrates for the first time that it is possible to accelerate reconstructions of single MPI frames by up to three orders of magnitude, allowing for actual real-time, high-resolution, 3D MPI reconstructions, even in the presence of noise. This was accomplished by implementing regularized MPI image reconstruction algorithms to be executed in parallel in GPU using CUDA, and by validating the proposed methods with the publicly available `OpenMPIData` dataset (Knopp *et al* 2020). Additionally, we show the first parallel implementation of the model-based calibration matrix computation in GPU, demonstrating that the combination of the parallel calculation of the calibration matrix with parallel reconstruction algorithms would deliver reconstructions faster than executing the serial Kaczmarz algorithm in the CPU. The structure of the paper is as follows: section 2 shows the MPI principles, section 3 describes the studied algorithms, section 4 describes the implementation of the parallel algorithms and the experiments performed, section 5 shows the results of the analysis of the algorithms, section 6 presents the validation of the parallel reconstruction algorithms using the `OpenMPIData` dataset (Knopp *et al* 2020), section 7 discusses the reported results, and section 8 concludes our study.

2. MPI signal generation

MPI exploits the nonlinear magnetization response of MNPs, most often *superparamagnetic iron oxides* (SPIOs). When exposed to a static magnetic field, ideal single-domain non-interacting MNPs exhibit a magnetization that can be described by the Langevin function:

$$M(H) = m\rho\left\{\coth\left(\frac{\mu_0 mH}{kT}\right) - \left(\frac{\mu_0 mH}{kT}\right)^{-1}\right\}, \tag{1}$$

where m is the particle magnetic moment in $(\mathrm{A}\cdot\mathrm{m}^2)$, ρ is the particle concentration in (m^{-3}), μ_0 is the vacuum permeability in $(\mathrm{N}/\mathrm{A}^2)$, H is the applied magnetic field magnitude in (A/m), k is the Boltzmann constant in (J/K) and T is the temperature in (K). In a typical MPI system, a gradient field produced by magnets and/or electromagnets oriented in opposite directions saturates all the particles except those within the region where the fields cancel each other, so when exposed to oscillating magnetic fields, only the MNPs inside the FFP (or FFL) produce a time-varying magnetization response, which can be detected by inductive coils. Scanning is accomplished by either additional electromagnets that shift the gradient field, or mechanical movement of the sample, such that the resulting applied field can be expressed as a function of the FFP trajectory $r(t)$:

$$\mathbf{H}(x,t) = \mathbf{G}(x)\,(x - r(t))\,, \tag{2}$$

where $\mathbf{G}$ is the static gradient field in (T/m). Therefore, the time-dependent signal generated by the particles can be described as a convolution of the time derivative of the Langevin function over the spatial distribution of MNPs in space:

$$s(t) = m\int_{\mathbb{R}^3}\rho(x)\,\mathbf{R}(x)\,\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L}\{\xi\,||\mathbf{H}(x,t)\,||\}\,\frac{\mathbf{H}(x,t)}{||\mathbf{H}(x,t)\,||}\mathrm{d}x, \tag{3}$$

where $\mathbf{R}(x)$ is the sensitivity of the inductive coil in (T/A) and $\xi = \mu_0 m/kT$. The convolution in (3) comes from the influence of the PSF, represented by the derivative of the Langevin function, that is steered over the sample volume and causes blurring in images obtained by direct gridding of the measured signal. Since the

[5] Reference is made to commercial products to adequately specify the experimental procedures involved. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that these products are the best for the purpose specified.

blurred image contains, at each position, contributions from the surrounding particles, deconvolution is required to remove the effect of the PSF by numerically solving a linear system of equations. In the *system function* method, the calibration matrix already accounts for the effect of the PSF, and intrinsically removes the blurring, also by solving a linear system of equations. A deeper discussion of the MPI signal generation modeling, including the particle dynamic response, is provided in Kluth (2018).

3. Reconstruction algorithms

Both *x-space* and *system function* methods require, for optimal accuracy, the numerical solution of systems of equations. In this work we use a model-based calibration matrix representing the continuous scanning of a FFP over a 3D MNP sample following a Lissajous trajectory on a traditional MPI system, as it is a method that provides more accurate results, being also the most computationally demanding approach. Equation (3) can be rewritten in a discrete form as

$$s(t_i) = m\sum_{j=0}^{N} c_j R(x_j)\,\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L}\left\{\xi\,||\mathbf{H}(t_i,x_j)\,||\right\}\frac{\mathbf{H}(t_i,x_j)}{||\mathbf{H}(t_i,x_j)\,||} \tag{4}$$

where the indices i and j represent a particular measurement and a voxel of the reconstruction grid, respectively, and c_j are the discretized values representing the spatial particle concentration. The derivative has been evaluated both analytically and numerically, and the two approaches give the same result. By isolating the particle concentration, the equation can be expressed as a linear system of equations:

$$K \cdot c = s, \tag{5}$$

where s is the vector of size M containing the measured signals, c is the unknown 3D concentration vector of size N, and K is the M by N convolution matrix (the calibration, or *system* matrix):

$$K_{i,j} = mR(x_j)\,\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L}\left\{\xi\,||\mathbf{H}(t_i,x_j)\,||\right\}\frac{\mathbf{H}(t_i,x_j)}{||\mathbf{H}(t_i,x_j)\,||} \tag{6}$$

where M is the number of measurements in each scan sequence, and N is the number of *voxels* of the reconstructed grid. Since the measured signal contains noise, Tikhonov regularization is required to avoid overfitting, keeping the solution stable. The function to be minimized is

$$||K \cdot c - s||_2^2 + \lambda||c||_2^2, \tag{7}$$

where λ is the regularization parameter and $||.||_2$ is the Euclidean norm of the vectors. Several algorithms and normalization schemes have been studied for the solution of (5) in addition to a least squares approach (Knopp *et al* 2017, Chen *et al* 2021), including total variation (Storath *et al* 2017), L1 regularization (Kluth 2020) and total least squares (Kluth *et al* 2017), but this work will focus on minimizing the Tikhonov regularization functional in (7). The algorithms chosen for performing this minimization were Kaczmarz, CGNR, the *Steepest Descent* and SVD.

3.1. Kaczmarz algorithm

The Kaczmarz algorithm was introduced in Kaczmarz (1937) and it has been widely used as an iterative solver since then. Also known as algebraic reconstruction technique (ART), it is a robust method that works by, at each iteration, projecting the current solution over the corresponding line of the calibration matrix, computing the residual with relation to the measured signal, and then correcting the solution accordingly. For equation (7), the $(i+1)$th estimate is given by:

$$c^{(i+1)} = c^{(i)} + \frac{s_i - K_i \cdot c^{(i)} - \lambda \cdot r_i}{||K_i||^2 + \lambda^2}K_i^T, \tag{8}$$

where s_i is the ith element of s, K_i is the ith row of K, and r_i is the ith element of a residual vector. One sweep over all M rows of K is considered one Kaczmarz iteration, which has complexity $O(N^2)$. The convergence of the algorithm strongly depends on the orthogonality of the matrix rows. A version of the Kaczmarz algorithm in which the rows are selected randomly, has been proposed in Strohmer and Vershynin (2006), in order to improve the convergence rate when the rows are nearly parallel. Alternatively, at the end of each update, it is possible to impose constraints to the solution, such as its positivity. Algorithm 1 shows a typical implementation of the Kaczmarz algorithm for the solution of the Tikhonov regularization functional (7). Variants of the Kaczmarz algorithm include the Cimmino or SIRT (*simultaneous iterative reconstruction*

technique) and the fixed-block ART, or block-Kaczmarz. While the former averages the corrections computed in the right-hand side of (8) for all the rows of K, the latter is a hybrid solution in which the problem is divided into blocks, and each block runs (8) over its portion of the data (Censor and Zenios 1997, Aster *et al* 2013, Quelhas *et al* 2023).

Algorithm 1. Regularized Kaczmarz algorithm.

$c^{(0)} \leftarrow 0$ ▷Initialize solution vector
$r \leftarrow 0$ ▷Initialize residual vector
for $i = 0 \rightarrow M$ **do**
  $\beta \leftarrow \left(s_i - K_i \cdot c^{(i)} - \lambda \cdot r_i\right) / \left(||K_i||^2 + \lambda^2\right)$
  $c^{(i+1)} \leftarrow c^{(i)} + \beta \cdot K_i^T$
  if $c^{(i+1)} < 0$ **then**
    $c^{(i+1)} \leftarrow 0$ ▷Positivity constraint
  end if
  $r_i \leftarrow r_i + \beta \cdot \lambda$
end for

3.2. Conjugate gradient normal residual—CGNR

The conjugate gradient (CG) method is an iterative algorithm that has been developed for solving systems of equations of the form $Ax = b$ in which the matrix A is symmetric and positive definite (Hestenes and Stiefel 1952). The algorithm works by iteratively determining a sequence of vectors $p^{(i)}$ that are mutually conjugate with relation to A, used as search directions, ideally converging in N iterations, where N is the size of A, although in general it does not converge exactly due to accumulated round-off errors (Shewchuk 1994, Aster *et al* 2013). In general the MPI convolution matrix in (5) is not symmetric, but the CG method can still be applied to the normal equation:

$$K^T \cdot K \cdot c = K^T \cdot s \tag{9}$$

in the so-called *conjugate gradient normal residual* (CGNR) method, also known as *conjugate gradient least squares* (CGLS) or *conjugate gradient normal equation* (CGNE). The algorithm does not require the explicit computation of $K^T K$, which reduces round-off errors and saves memory space and computation time, but the increased condition number of the matrix might lead to slower convergence. When applied for the minimization of the Tikhonov regularization functional (7), the CGNR method gives as the next estimate:

$$c^{(i+1)} = c^{(i)} + \alpha \cdot p^{(i)}, \tag{10}$$

where

$$\alpha = \frac{||\tilde{K}^T \cdot r^{(i)}||^2}{||\tilde{K} \cdot p^{(i)}||^2}, \tag{11}$$

$$p^{(i)} = \tilde{K}^T \cdot r^{(i)} + \frac{||\tilde{K}^T \cdot r^{(i)}||^2}{||\tilde{K}^T \cdot r^{(i-1)}||^2} p^{(i-1)}, \tag{12}$$

$$r^{(i)} = s - \tilde{K} \cdot c^{(i)}, \tag{13}$$

and

$$\tilde{K} = \begin{vmatrix} K \\ \lambda I \end{vmatrix} \tag{14}$$

is the augmented convolution matrix, which requires the use of extended vectors for dealing with the λ-residuals. Algorithm 2 shows the implementation of the regularized CGNR method, which has complexity $O(N^2)$. Alternatives for improving CG convergence include matrix preconditioning and the use of a weighting matrix (Knopp *et al* 2010). The CGNR method relies on the conjugacy between the subsequent search directions for optimum convergence, which prevents the adoption of a positivity constraint, as implemented in the Kaczmarz algorithm.

Algorithm 2. Regularized CGNR.

$c^{(0)} \leftarrow 0$ ▷Initialize solution vector
$r^{(0)} \leftarrow s - K \cdot c^{(0)}$ ▷Initialize residual vector
$r_\lambda^{(0)} \leftarrow -\lambda \cdot c^{(0)}$ ▷Initialize λ-residual vector
$z^{(0)} \leftarrow K^T \cdot r^{(0)} + \lambda \cdot r_\lambda^{(0)}$
$p^{(0)} \leftarrow z^{(0)}$ ▷Define first search direction
for $i = 0 \rightarrow N$ **do**
  $w^{(i)} \leftarrow K \cdot p^{(i)}$
  $w_\lambda^{(i)} \leftarrow \lambda \cdot p^{(i)}$
  $\alpha^{(i)} \leftarrow ||z^{(i)}||^2 / (||w^{(i)}||^2 + ||w_\lambda^{(i)}||^2)$
  $c^{(i+1)} \leftarrow c^{(i)} + \alpha^{(i)} \cdot p^{(i)}$
  $r^{(i+1)} \leftarrow r^{(i)} - \alpha^{(i)} \cdot w^{(i)}$
  $r_\lambda^{(i+1)} \leftarrow r_\lambda^{(i)} - \alpha \cdot w_\lambda^{(i)}$
  $z^{(i+1)} \leftarrow K^T \cdot r^{(i+1)} + \lambda \cdot r_\lambda^{(i+1)}$
  $\beta^{(i+1)} \leftarrow ||z^{(i+1)}||^2 / ||z^{(i)}||^2$
  $p^{(i+1)} \leftarrow z^{(i+1)} + \beta^{(i+1)} \cdot p^{(i)}$
end for

3.3. Steepest descent

The steepest descent method is a simple first-order algorithm for minimization problems, widely used in deep-learning applications (Soumare *et al* 2021). The method uses the inverse of the gradient of the function (its residual) being minimized as the search direction in each iteration. As with CGNR, the method is applied to the normal equation (9), and when used to minimize (7), the next iteration is given by:

$$c^{(i+1)} = c^{(i)} + \alpha \cdot z^{(i)}, \tag{15}$$

where

$$\alpha = \frac{||z^{(i)}||^2}{||\tilde{K} \cdot z^{(i)}||^2}, \tag{16}$$

and

$$z^{(i)} = \tilde{K}^T \cdot \left(s - \tilde{K} \cdot c^{(i)}\right) \tag{17}$$

The similarity with the CG method is not a coincidence, as CG is an improvement of the steepest descent method. In fact, the first search direction of the CG method is given by the residual of the function. The steepest descent method applied to (7) is detailed in Algorithm 3. As in the CGNR method, the residuals are updated together with the estimate, without the need of the calculation of new ones. If one wishes to enforce a constraint, such as non-negativity, the new residuals must be computed, adding a matrix-vector multiplication to the algorithm, slowing down its execution. The complexity of the steepest descent method is also $O(N^2)$.

Algorithm 3. Regularized Steepest descent.

$c^{(0)} \leftarrow 0$ ▷Initialize solution vector
$r^{(0)} \leftarrow s - K \cdot c^{(0)}$ ▷Initialize residual vector
$r_\lambda^{(0)} \leftarrow -\lambda \cdot c^{(0)}$
for $i = 0 \rightarrow N$ **do**
  $z^{(i)} \leftarrow K^T \cdot r^{(i)} + \lambda \cdot r_\lambda^{(i)}$ ▷Update search direction
  $w^{(i)} \leftarrow K \cdot z^{(i)}$
  $w_\lambda^{(i)} \leftarrow \lambda \cdot z^{(i)}$
  $\alpha^{(i)} \leftarrow ||z^{(i)}||^2 / (||w^{(i)}||^2 + ||w_\lambda^{(i)}||^2)$
  $c^{(i+1)} \leftarrow c^{(i)} + \alpha^{(i)} \cdot z^{(i)}$
  $r^{(i+1)} \leftarrow r^{(i)} - \alpha^{(i)} \cdot w^{(i)}$
  $r_\lambda^{(i+1)} \leftarrow r_\lambda^{(i)} - \alpha \cdot w_\lambda^{(i)}$
end for

3.4. Singular value decomposition—SVD

The singular value decomposition (SVD) falls into the category of direct solvers rather than iterative solvers. The SVD is a factorization method for solving ill-posed least squares problems (Aster *et al* 2013). It works by decomposing the $M \times N$ matrix K into three matrices, U, Σ and V, where U is a $M \times M$ unitary orthogonal matrix, Σ is a $M \times N$ rectangular diagonal matrix containing the singular values σ_i of K, and V is a $N \times N$ orthogonal matrix. These three matrices are used to determine the so-called pseudoinverse of K, which allows finding the solution of the system of equations. However, the presence of noise corrupts the decomposition, leading to unstable solutions, and the implementation of the SVD to the Tikhonov regularization provides the following solution:

$$c_\lambda = \sum_{i=0}^{m} \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \frac{U_i^T \cdot s}{\sigma_i} V_i. \tag{18}$$

In the expression above, the regularization parameter λ acts as a low-pass filter, cancelling the contributions from small singular values, similarly to the truncated singular value decomposition (TSVD). The use of SVD for image reconstructions in MPI has been previously studied in Knopp *et al* (2008), and in Knopp *et al* (2010) it has been shown that its execution time is significantly longer than the Kaczmarz and the CGNR algorithms. Indeed, the factorization of K is the most expensive operation in the SVD method, with complexity $O(N^3)$, and requires a significant amount of memory space to store temporary values, but once it is done, the pseudoinverse can be kept in the memory and the computation of the solution is fast and straightforward (Vogel *et al* 2016, 2017). Algorithm 4 shows a simple implementation of the regularized SVD reconstruction, where f_i are the so-called *filter factors* of equation (18).

Algorithm 4. Regularized SVD reconstruction.

Input matrices: $U_{M\times M}$, $V_{N\times N}$, $\Sigma_{M\times N}$ ▷SVD decomposition of K
$c^{(0)} \leftarrow 0$ ▷Initialize solution vector
$p = U^T \cdot s$
for $i = 0 \rightarrow$N **do**
  $f_i = \sigma_i / (\sigma_i^2 + \lambda^2)$
  $p_i = p_i \cdot f_i$
end for
$c = V \cdot p$ ▷Compute solution vector

4. Experiments

We compared multiple parallel algorithms implemented in GPU with the serial Kaczmarz algorithm implemented in CPU, to investigate whether the GPU algorithms could deliver superior speed and accuracy. In our simulation experiments, we tested all the algorithms using identical conditions for the MNP parameters, MPI scanner parameters, and noise to enable meaningful comparisons. Using the input parameter conditions and a simulated concentration phantom, we first used an analytical forward model to generate the simulated signals and then applied the reconstruction algorithms to these signals to generate images that could be compared with the known simulated phantom.

4.1. Parallel implementation in GPU

Depending on the selected parameters, the execution of the reconstruction algorithms can be extremely time-consuming. More specifically, the sampling rate that will define the number of measurements for a fixed scan time, or rows of the calibration matrix, and the resolution of the reconstructed image, or the number of columns of the calibration matrix, determine the size of the problem to be solved. In order to yield faster reconstructions, several strategies have been adopted, including the reduction of the data dimension by filtering or row selection (Vogel *et al* 2017), data compression (Lampe *et al* 2012) and low-rank approximations (Kluth and Jin 2019). In this work we show the parallelization of these algorithms to be executed in the GPU and the significant performance gains that result from these implementations.

At the same time, the computation of the model-based calibration matrix is another expensive operation—each of the M rows of K is computed as the PSF scaled by the space-dependent receive coil sensitivity and the FFP scanning speed at a time t, and analogously, each of the N columns of K is the characteristic signal response generated by a particle delta sample positioned in a particular *voxel*—with complexity $O(N^2)$. The computational cost of determining the calibration matrix prevents its calculation during execution time (e.g. when one wishes to change the scanning parameters, which requires a new matrix), so K has to be constantly stored in the memory and the amount of available space becomes a serious

Table 1. MPI scanner parameters.

Symbol	Description	Value
M_S	Particle saturation magnetization	450 kA/m
d_C	Particle core diameter	20 nm
T	Temperature	295 K
G	Gradient field strength $[x, y, z]$	$[-8, 4, 4]$ T/m
B	Drive field amplitude $[x, y, z]$	$[40, 20, 20]$ mT
f_B	Base frequency	250 Hz
f_M	Frequency multipliers $[x, y, z]$	$[101, 100, 99]$
t_S	Scan time	4 ms
S_R	Sampling rate	1.25 MHz
R	Receive coil sensitivity	8.38×10^{-4} T/A
c	particle concentration*	$3.168 \times 10^{20}\text{m}^{-3}$
S	Sample grid size $[x, y, z]$	$[10, 10, 10]$ mm
D	Sample grid dimensions $[x, y, z]$	$[20, 20, 20]$ *voxels*

*Concentration determined to be equivalent to 5 mg(Fe)/ml.

limitation factor. To address these two problems, we also propose a parallel implementation of the computation of the calibration matrix.

The code was developed using standard C++ language and the CUDA framework, for better performance. The computations were performed using double-precision floating point operations both in the CPU and GPU. The reconstruction algorithms were implemented in parallel using the *cuBLAS* library for most matrix and vector operations, such as matrix-vector multiplications, dot products and norm calculations. For the SVD factorization of the calibration matrix, the *cuSOLVER* library was employed. A number of custom *kernels* were implemented for the operations that were not efficiently addressed by either library. One example of such a *kernel* is the parallel computation and application of the filter factors within the *for* loop of the SVD reconstruction in algorithm 4, in which each thread i computes its corresponding f_i and p_i values. The parallel calculation of the calibration matrix has been implemented also as a customized *kernel*, in which each *thread* computes one of the M rows of K, as shown in equation (6). As long as the calibration matrix is already stored in the GPU memory, the costs of data transfers are negligible, being limited to one transfer of the measured signal (CPU $\rightarrow$ GPU) and one transfer of the reconstructed image (GPU $\rightarrow$ CPU) per execution.

4.2. Setup studied

The algorithms were employed to reconstruct signal data generated by simulating a hypothetical MPI scanner using an analytical forward model. Table 1 shows the parameters of the simulated measurement system. The simulated phantom consisted of a $20 \times 20 \times 20$ *voxel* grid containing 20 nm magnetite particles distributed to have the shape of a question mark, as shown in figure 1. Both the calibration matrix and the signal data were generated and processed in time domain, since the solution of the system in frequency domain would be similar. There was no filtering and no row selection, and all the data were used in the reconstructions, since the goal of this work is to demonstrate the reconstruction performance when processing large data sets.

4.3. Execution

Signals were simulated and reconstructed for different sampling rates (number of samples, or rows), noise levels and resolutions (number of *voxels*, or columns), to allow assessing the scalability of the parallel implementation, as well as to avoid running the reconstructions using the same model used to generate the data (Wirgin 2018). The adopted sampling rates were 2.5 MHz, 1.25 MHz, 625 kHz and 312.5 kHz, which correspond to $10,000$, $5,000$, $2,500$ and $1,250$ samples per each one of the x, y and z channels for a full 4 ms scan, respectively. The reconstruction resolutions were $10 \times 10 \times 10$, $15 \times 15 \times 15$, $20 \times 20 \times 20$, $25 \times 25 \times 25$ and $30 \times 30 \times 30$ *voxels*. The adopted noise levels were 5 μV (~ 0.1 % of the peak signal or -60 dB) and 50 μV (~ 1 % or -40 dB), with the corresponding regularization parameters 5×10^5 and 1×10^6, respectively, which were chosen by analyzing the residuals from the reconstructions and by visual inspection of the reconstructed images.

The simulations and reconstructions were executed on a desktop computer equipped with an Intel Xeon W-2145 CPU, 32 GB of RAM memory, and a NVIDIA Quadro RTX 4000 GPU containing 36 streaming multiprocessors (*SMs*) with 64 cores each, totaling 2304 CUDA cores capable of running simultaneous threads, and 8 GB of dedicated RAM memory. The serial algorithms were executed using a single thread in the CPU. For all iterative algorithms, it was considered that one iteration is completed when all the rows of K were evaluated, so for Kaczmarz algorithm it meant M updates of the solution vector, while for CGNR and

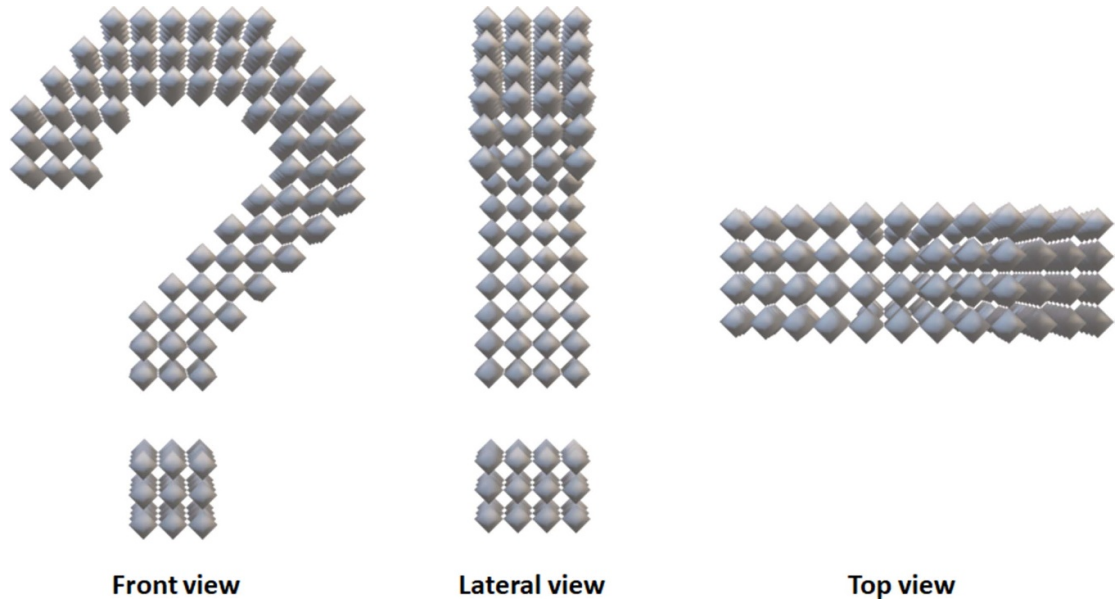

Figure 1. Simulated phantom grid used for signal generation. The *voxels* are represented by 3D diamonds to allow visualization through the surface, with the size of the diamonds proportional to the *voxel*'s particle concentration. In this simulated phantom, all filled *voxels* have maximum concentration, as given by table 1.

steepest descent it meant one update that used the whole calibration matrix. Reconstructions were performed and recorded for, 1, 5, 10, 20, 30, 40, 50, 100, 150 and 200 iterations using the Kaczmarz algorithm running in the CPU with and without positivity constraint, and the CGNR and steepest descent running in the GPU. Since the SVD is a direct solver, only one execution of the algorithm is performed at a time. In Quelhas *et al* (2023), it has already been demonstrated that both the parallel implementation of the Kaczmarz algorithm and the serial implementation of CGNR are not efficient, so they were not included in this study. The number of iterations was chosen to ensure that most algorithms would converge, and the reported executions times are the average of at least three runs.

5. Results

This work aims to find more efficient reconstruction algorithms while ensuring optimum accuracy. For all combinations of configurations (sampling rate, resolution, noise level and number of iterations), the efficiency of the studied algorithms was assessed by measuring the execution times and the reconstruction errors in two spaces (or domains). In signal space, the mean square error (MSE) and peak signal-to-noise ratio (PSNR) were used to assess the reconstruction errors. In image space, for reconstructed images with the same resolution of the phantom grid ($20 \times 20 \times 20$ *voxels*), the MSE and SSIM (*Structural Similarity Index Measure*, Wang *et al* 2004) were employed.

The reported times account for the whole reconstruction, and include data transfers between CPU and GPU (sending the input signal and retrieving the resulting image). The time for the parallel decomposition of the calibration matrix is not included in the reported SVD times, since it only needs to be performed once, and it ranged from 0.63 s to 100 s, depending on the simulation and reconstruction parameters.

5.1. Convergence and accuracy

Figure 2 shows in the left two plots the MSEs relative to the initial MSE values (MSE_0), and on the right it shows the calculated PSNR values, calculated with relation to the measured signals of the $20 \times 20 \times 20$ reconstructions for 5 μV (0.1%, top two plots) and 50 μV (1%, bottom two plots) of noise. We can see how efficient the convergence of the Kaczmarz (KACZ_CPU) algorithm is—after a single iteration (180 ms) the relative MSE dropped to less than 10^{-2} (or a PSNR $>$ 30 dB), and after 10 iterations (1.8 s) it reached a value lower than 10^{-3} (PSNR $>$ 40 dB), even for 1% of noise. The constrained Kaczmarz reconstruction (KACZ_PC_CPU) started with higher MSE values and lower PSNR values than the non-constrained version, but reached better results after 20 iterations (5 s). After a modest convergence in the first iteration, the CGNR (CGNR_GPU) decreased the MSEs and increased the PSNR consistently and eventually reached better values than Kaczmarz after about 40 iterations (248 ms for CGNR and 10 s for Kaczmarz). The steepest descent (SD_GPU) method, however, showed very poor convergence, and after 200 iterations (1.2 s) did not even

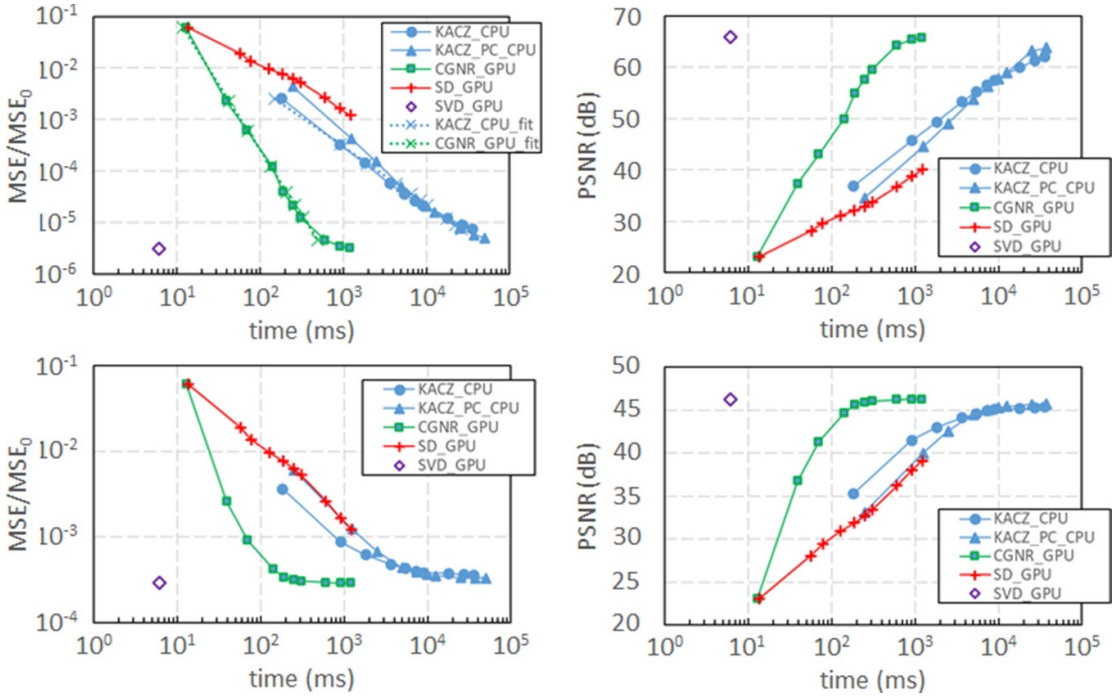

Figure 2. MSE values (left) relative to the initial MSEs (MSE_0) and PSNR values (right) of the reconstructions with relation to the measured signals as functions of time, for 5 μV (0.1% or −60 dB, top) and 50 μV (1% or −40 dB, bottom) of noise. Connected points correspond to values after 1, 5, 10, 20, 30, 40, 50, 100, 150 and 200 iterations for the serial Kaczmarz algorithm (blue circles), the serial Kaczmarz algorithm with positivity constraint (blue triangles), the parallel CGNR (green squares) and the parallel steepest descent (red crosses). Results with SVD (purple diamonds) are shown for a single execution. The blue and green dashed lines and X's in the top left plot are the fits of the relative MSEs of the Kaczmarz algorithm and CGNR, respectively, as discussed in section 5.2.

reach the accuracy of Kaczmarz after 5 iterations (1.2 s) or CGNR after 10 iterations (70 ms). The SVD (SVD_GPU) reconstruction delivered slightly better MSE and PSNR values than all other algorithms, regardless of the noise level. In general, the PSNR values found reflected the MSE values in a different scale, and confirmed the interpretation of the results in signal space.

In real life imaging applications, the only way to evaluate the accuracy of the reconstruction is by taking the residuals $(s - K \cdot c)$, since typically the ground truth is unknown. For our modeled $20 \times 20 \times 20$ reconstruction, it is possible to compare the concentration values of the reconstructed grid with the ground truth, i.e. the simulated phantom grid. Figure 3 shows the MSE and SSIM values with relation to the ground truth for 50 μV (1%) of noise. Clearly, in image space, the constrained Kaczmarz reconstruction delivers a more accurate reconstruction than the non-constrained algorithms, including SVD. The MSE evaluation was confirmed by the SSIM assessment, which also reflected the MSE values but in a different scale, showing that the constrained Kaczmarz algorithm delivered an image closer to the ground truth (an SSIM of 1 means that the two images being compared are identical). Figure 4 shows the resulting 3D reconstructions using the different algorithms, illustrating how the constrained Kaczmarz algorithm delivers better reconstructions (see figure 4(b)), with less artifacts around the main volume. It is also interesting that, for the non-constrained minimization of the Tikhonov regularization functional, all the algorithms converge to the same solution—the steepest descent would as well, with enough iterations.

We found that one single iteration of the constrained Kaczmarz algorithm is capable of significantly improving the accuracy of the non-constrained Tikhonov regularization solution, as also shown in figures 3 and 4(f). The image reconstructed using SVD was subject to one additional iteration of the constrained Kaczmarz algorithm, resulting in a improvement of the MSE with relation to the ground truth to a level equivalent to the constrained Kaczmarz solution, and to the reduction of the amount of artifacts around the image. Naturally, the same approach can be applied to reconstructions obtained with the CGNR algorithm, as the solution of the minimization of the Tikhonov regularization functional is the same for all algorithms. The same approach has been employed for the reconstruction of signals with 5% and 10% of noise, and the robustness of the method has been confirmed for these higher noise levels.

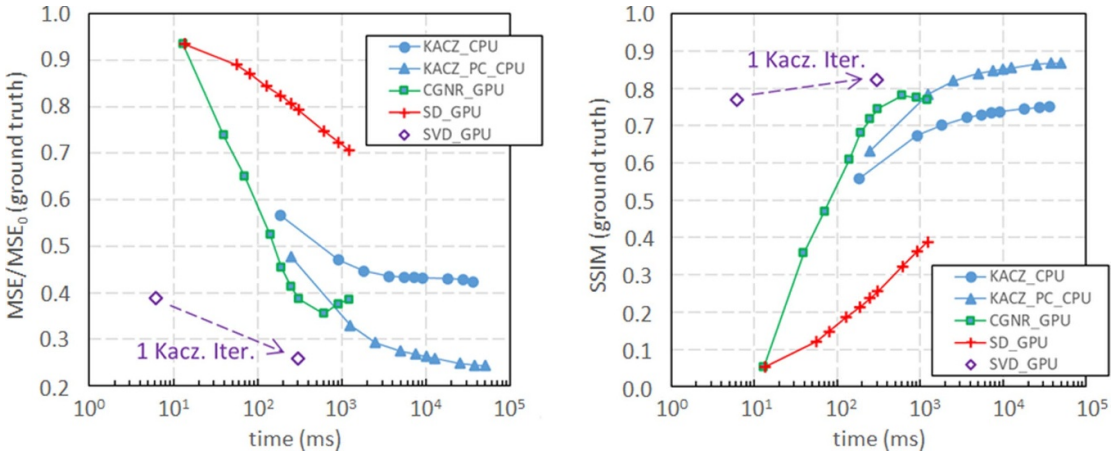

Figure 3. MSE values (left) relative to the initial MSEs (MSE_0) and SSIM values (right) of the reconstructions with relation to the ground truth (the phantom grid) as function of time, for 50 μV (1% or −40 dB) of noise. Connected points correspond to values after 1, 5, 10, 20, 30, 40, 50, 100, 150 and 200 iterations for the serial Kaczmarz algorithm (blue circles), the serial Kaczmarz algorithm with positivity constraint (blue triangles), the parallel CGNR (green squares) and the parallel steepest descent (red crosses). SVD plot (purple diamonds) show the resulting MSE after a single execution, and the dashed line connects it to the results of the SVD followed by a single iteration of Kaczmarz.

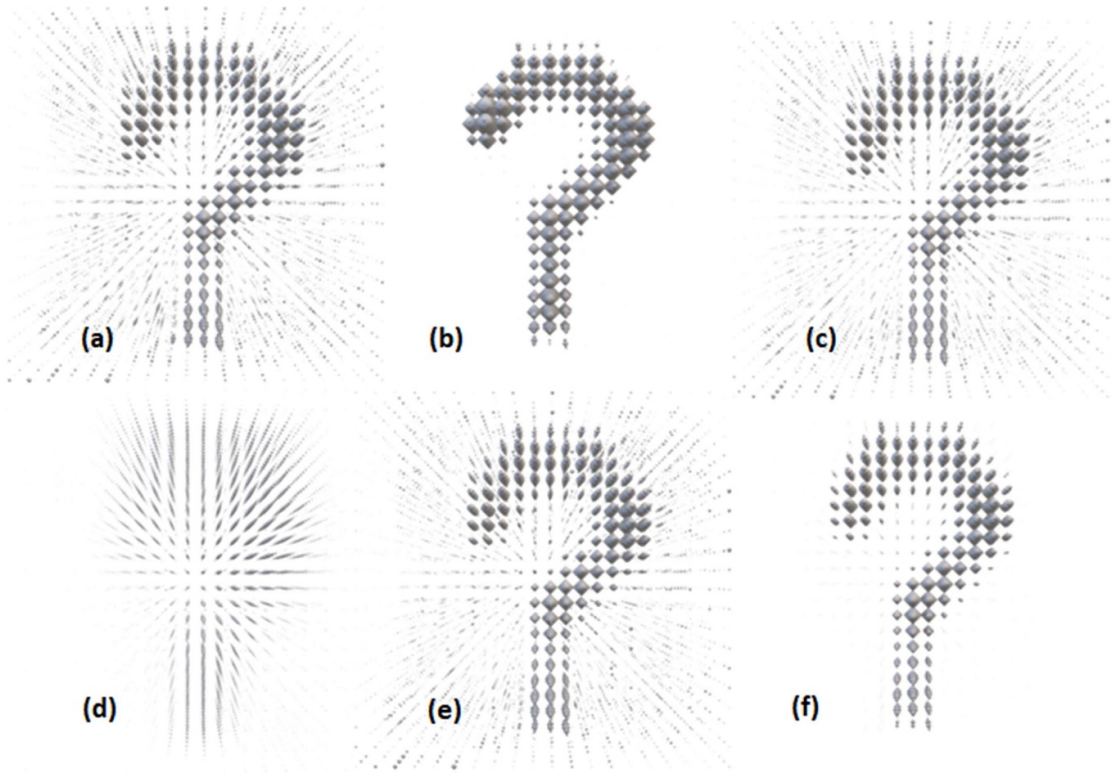

Figure 4. Reconstruction results after 200 iterations for a simulated signal with 50 μV (1% or −40 dB) of noise using (a) the serial Kaczmarz algorithm, (b) the serial Kaczmarz algorithm with positivity constraint, (c) the parallel CGNR algorithm, (d) the parallel steepest descent algorithm, (e) the parallel SVD and (f) the parallel SVD followed by a single iteration of the Kaczmarz algorithm with positivity constraint.

5.2. Computation performance

Figures 2 and 3 also show the execution times of the algorithms studied in this work. The parallel CGNR is in general more than one order of magnitude faster than the serial Kaczmarz algorithm, judged by time per iteration. By the moment Kaczmarz finished its first iteration, CGNR had already executed 30 iterations, and reached significantly better metric values, both with relation to the input signal (MSE and PSNR) and with relation to the ground truth (MSE and SSIM). Typically, the Kaczmarz algorithm is employed with a small number of iterations, no more than 10, for lower execution times. If we take the execution time and results of running 10 Kaczmarz iterations as reference, CGNR would need 20 iterations to reach the same MSE and

Table 2. Reconstruction times as function of required accuracy.

MSE/MSE$_0$	Kaczmarz (CPU)	CGNR (GPU)	SVD (GPU)
10^{-3}	345 ms	59 ms	6.2 ms
10^{-4}	2.7 s	147 ms	6.2 ms
10^{-5}	21 s	362 ms	6.2 ms

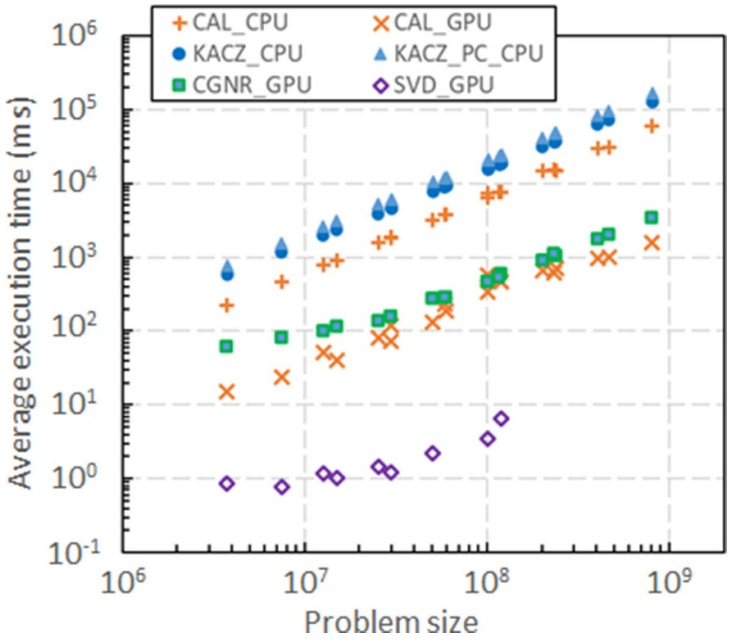

Figure 5. Execution times as functions of the problem size (number of *voxels* times the number of measurements) for the serial calculation of the calibration matrix (orange crosses), the parallel calculation of the calibration matrix in the GPU (orange X's), the serial Kaczmarz algorithm in the CPU (dark blue circles), the serial Kaczmarz algorithm with positivity constraint in the CPU (light blue triangles), the parallel CGNR in the GPU (green squares) and the parallel SVD in the GPU (purple diamonds). The results with the steepest descent were omitted due to their poor accuracy.

PSNR with relation to the input signal, doing it 13 × faster. If we take all the 200 iterations, CGNR would be on average 30 × faster than the Kaczmarz algorithm. Although it runs as fast as the parallel CGNR, the parallel steepest descent exhibited very poor convergence, which does not justify its use in practical applications.

As iterative algorithms with different convergence rates are being compared to a direct SVD method that executes only once, a suitable way of comparing their performances could be taking the times spent by the algorithms to reach a certain desired error level. Therefore, by fitting the linear intervals of the top left log plots shown in figure 2 (i.e. the MSE evaluation for 0.1% of noise), for 10^{-3} of relative MSE error, the serial Kaczmarz algorithm would take about 345 ms, while the parallel CGNR would take about 59 ms, meaning 5.8 × of speedup. For the parallel SVD, its single execution delivers better accuracy, and does so in 6.2 ms in average, meaning a speedup of approximately 56 ×. For 10^{-4} of relative MSE, the speedup of the parallel CGNR would be 18 ×, while for SVD it would be 430 ×, approximately. Finally, for 10^{-5} of relative MSE, the speedups would be about 57 × for CGNR and $3,338\times$ for SVD. Table 2 summarizes the execution times as function of the required accuracy. Similar conclusions can be drawn using the PSNR values instead of MSEs for the evaluation.

The serial Kaczmarz algorithm with positivity constraint may be more expensive to be executed, but the improved convergence and accuracy justifies its use, since it delivers the best results among the algorithms studied in this work. However, as shown in figure 3, it is possible to improve the accuracy of the SVD (and also CGNR) reconstruction by adding a single iteration of the Kaczmarz algorithm with positivity constraint. In this case the average execution time of the parallel SVD would increase to 298 ms, meaning speedups of 1.2 ×, 9.1 × and 71 × when the desired relative MSEs with relation to the measured signals were 10^{-3}, 10^{-4} and 10^{-5}, respectively.

The scalability of the parallel implementations is depicted in figure 5, which shows the evolution of the execution times of the implemented algorithms (the parallel CGNR, the parallel SVD and also the parallel calculation of the calibration matrix) as the size of the problem increased. The problem size is the size of the

Table 3. Speedups of the parallel implementations.

Algorithm	Minimum	Maximum	Average
Calibration matrix[a]	12.7 ×	37.2 ×	21.8 ×
CGNR[b]	9.6 ×	36.8 ×	29.2 ×
SVD[b]	883 ×	6,101×	3,482×
Calibration matrix + CGNR[b]	7.7 ×	25.1 ×	18.1 ×

[a] *Speedups* relative to its serial implementation in the CPU.
[b] *Speedups* relative to the serial Kaczmarz algorithm.

calibration matrix, and is equal to the number of *voxels* times the number of measurements. For the iterative algorithms the times were taken for 200 iterations, while for the calibration matrix calculation and the SVD the times taken correspond to one execution. The plot shows clearly how the parallel algorithms consistently outperform their serial counterparts, regardless of the problem size. Since the steepest descent algorithm showed very poor accuracy, its execution times were not included. The parallel SVD was evaluated for a smaller subset of problem sizes, since the GPU used in this study would run out of memory for bigger problem sizes. It can be derived from figure 5 that it is possible to calculate the calibration matrix and perform the parallel reconstruction using CGNR in significantly less time than the time needed to perform the serial Kaczmarz reconstruction, which is a very interesting result, that would allow for real-time or partial computations of the calibration matrix, without the need of storing the whole matrix in the memory. Table 3 gives the speedups of the parallel implementations studied in this work taken from the data shown in figure 5, including the parallel calculation of the calibration matrix followed by a CGNR reconstruction. The parallel computation of the calibration matrix was compared to its serial version, while the reconstruction algorithms were compared to the serial Kaczmarz algorithm without positivity constraint.

6. GPU-accelerated reconstructions applied to the OpenMPI dataset

To confirm the promising results obtained using simulated data, the image reconstruction algorithms studied in this work were applied to the `OpenMPIData` dataset (Knopp *et al* 2020), widely used in the literature for validation of reconstruction methods. The datasets studied correspond to the experiment number 3, and they consisted of the 3D calibration matrix and the measured signals obtained with the *Shape*, *Resolution* and *Concentration* phantoms for a 38 × 38 × 19 mm field of view and a 19 × 19 × 19 reconstruction resolution.

Both the calibration matrix and the signals were filtered to remove frequency components below 80 kHz, and the size of the resulting matrix was 3.86 GB, which corresponds to 75 615 rows (25 205 per each of the x, y and z channels) by 6859 columns, a problem size of approximately 5.18×10^8, lying within the range of problem sizes analyzed in the previous section. The full calibration matrix occupies too much space in the GPU memory, and because of this the SVD factorization was not possible. To allow assessing the SVD reconstructions as well, the calibration data of a single receive coil, instead of the three coils, was used for the reconstructions. Therefore, all algorithms were tested for data from a single receive coil, and the Kaczmarz and CGNR were also tested for the full dataset using three coils. The noise levels of the signals measured with the three phantoms were estimated by taking, for each element of the signal vectors, the standard deviation of the 1000 repetitions of the measurement sequence. The averages of the standard deviations over the whole signal vectors were taken as estimates of the noise, and correspond to 0.14 % (or −56.8 dB) for the shape phantom, 0.32 % (or −49.9 dB) for the resolution phantom, and 0.36 % (or −48.8 dB) for the concentration phantom, which lies within the range of noise levels studied in the previous section.

For all reconstruction algorithms, a regularization parameter of 3.5×10^4 was found to deliver the best compromise between level of details and noise by analyzing the reconstruction residuals and visual inspection of the reconstructed images. Since the knowledge about the true MNP spatial distribution of the phantoms is limited, the procedure for the assessment of the reconstructed images was the same for assessing the reconstruction of the simulated data in signal space, so the metrics used for measuring the quality of the reconstructions were the MSE and PSNR with relation to the input signals. Preliminary results showed that for the reconstructions of all three phantoms, both metrics degraded significantly when positivity constraint was applied. We postulate that the signal filtering prevents the application of the constraint, and there is precedent in the literature to support this. As can be seen in Lu *et al* (2013), Henn *et al* (2022), the loss of the fundamental frequency by filtering is equivalent to having a DC offset in the reconstructed MPI image. Therefore, forcing the positivity constraint in the algorithms would lead to incorrect solutions and consequently poorer quality metrics. For this reason, only the non-constrained reconstruction algorithms were analyzed in this section.

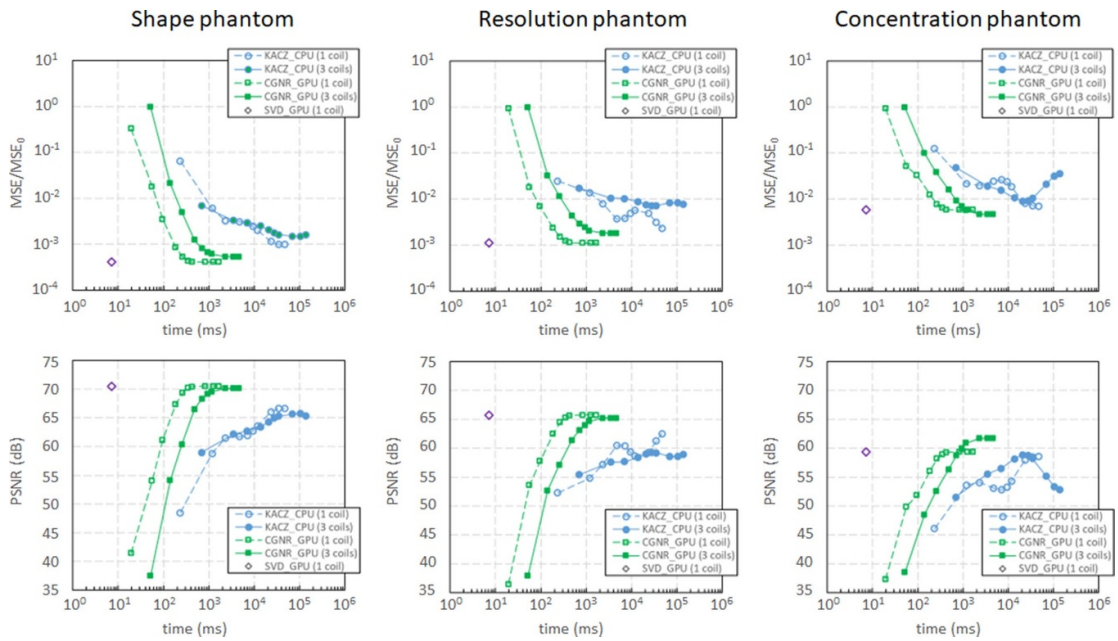

Figure 6. MSE values (top) relative to the initial MSEs (MSE_0) and PSNR values (bottom) of the reconstructions with relation to the measured signals as functions of time, for the shape phantom (left), resolution phantom (center) and concentration phantom (right). Connected points correspond to values after 1, 5, 10, 20, 30, 40, 50, 100, 150 and 200 iterations for the serial Kaczmarz algorithm using data of one receive coil (open blue circles) and the three receive coils (solid blue circles), and the parallel CGNR using data of one receive coil (open green squares) and the three receive coils (solid green squares). Results with SVD using data of one receive coil (open purple diamonds) are shown for a single execution.

Table 4. Speedups, relative to the serial Kaczmarz algorithm, of the `OpenMPIData` reconstructions as function of required accuracy.

	Shape phantom			Resolution phantom			Concentration phantom		
	Single coil		All coils	Single coil		All coils	Single coil		All coils
MSE/MSE_0	CGNR	SVD	CGNR	CGNR	SVD	CGNR	CGNR	SVD	CGNR
10^{-2}	17.8	210	0.9	17.6	297	18.0	82.2	2,879	22.8
5×10^{-3}	31.5	502	4.6	54.9	1,266	*	—	—	—
10^{-3}	118	3,820	*	—	—	—	—	—	—

* Serial Kaczmarz algorithm did not reach this error level.
— Neither algorithms reached this error level.

Figure 6 shows in the top plots the relative MSEs of the reconstructions for the three phantoms, while the bottom plots show their corresponding PSNRs values, for the Kaczmarz algorithm running in the CPU, and for the CGNR and SVD running in the GPU. The plots show that the parallel algorithms clearly outperform the serial Kaczmarz algorithm in all scenarios. The reconstructions of the shape phantom obtained the lowest MSE and highest PSNR values, presumably because the noise level of the measured signal (−56.8 dB) was the lowest among the three samples. The reconstructions with the concentration phantom obtained the worst results, and it is also attributed to the noise level of its measured signal. Regardless of the noise level, the CGNR consistently converged to the optimal solution after about 50 iterations, while the Kaczmarz algorithm, despite the good initial results, oscillated and did not converge even after 200 iterations.

The accelerations of the parallel algorithms ranged from one to three orders of magnitude, depending on the desired accuracy, as also found with the simulated data. Similarly to what has been done with the reconstructions with simulated data, the linear intervals of the MSE plots for reconstructions with the three phantoms were fitted in order to allow estimating the speedups as function of the desired error level, and the results are summarized in table 4. These results demonstrate that the performance gains resulting from the adoption of the parallel algorithms are directly proportional to the desired accuracy, especially when the noise level of the input signal is low and greater accuracy is possible.

Figure 7 shows the cross sections of the 3D images reconstructed using the algorithms studied in this work, comparing them to the images presented on the `OpenMPIData` website (Knopp *et al* 2020). The degree of agreement between the images is excellent, and demonstrates that the proposed parallel algorithms are both fast and accurate, allowing for real-time reconstructions.

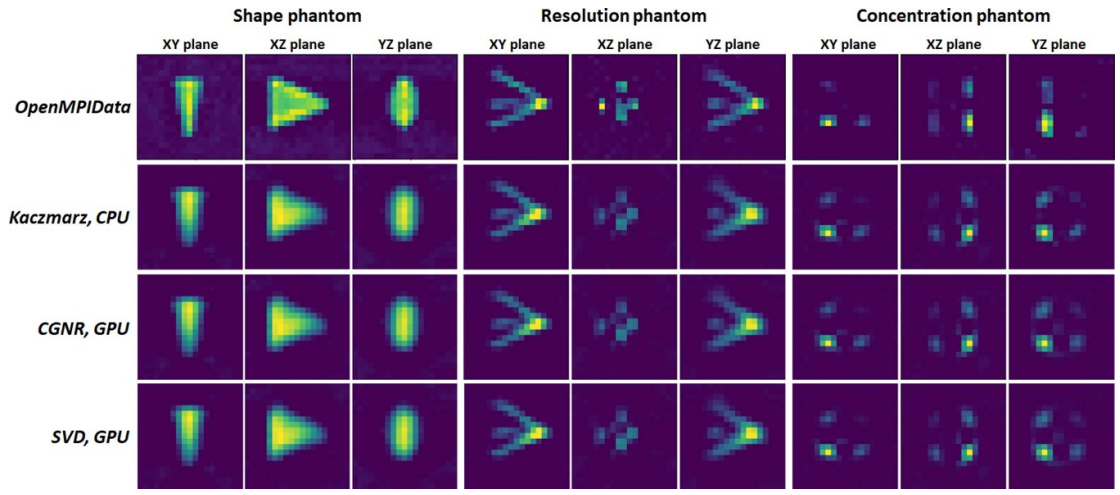

Figure 7. Comparison of the cross sections of the images reconstructed using the studied algorithms with the cross sections of the images presented on the `OpenMPIData` website (Knopp *et al* 2020). Adapted from Knopp *et al* (2020). CC BY 4.0.

7. Discussion

The results presented in sections 5 and 6 show the potential of GPU-accelerated algorithms to speed up MPI reconstructions by an order of magnitude or more, depending on the application, the needed accuracy level and the desired image resolution. When comparing the algorithms in terms of a specific reconstruction error level, the parallel CGNR algorithm performed up to 57 × faster than the serial Kaczmarz algorithm in the analysis using the simulated signals, and up to 118 × faster than Kaczmarz in the analysis using real measured signals. Also in terms of a specific error level, the parallel SVD performed up to 3338 × faster than Kaczmarz for simulated signals and up to 3820 × faster for real measured signals. When comparing simply in terms of execution time, the parallel CGNR is up to 36.8 × and the parallel SVD is up to 6101 × faster than the serial Kaczmarz algorithm.

It had been shown previously that storing the pseudoinverse matrix computed with SVD in the memory would result in lower computation times (Vogel *et al* 2017). We showed that, even for problems up to three orders of magnitude bigger, the parallel implementation delivered faster reconstructions. It finds good use in applications in which images are required at high frame rates and resolution is not critical (e.g. search scans) since the parallel SVD is capable of delivering images at rates of 156–1,250 frames per second, depending on the size of the problem. When improved resolution is needed and memory space becomes a limitation (the SVD reconstruction for 20 × 20 × 20 resolution and 1.25 MHz of sampling rate required 5.4 GB of GPU memory space, for example), the parallel CGNR algorithm still is about 30 × faster than the serial Kaczmarz algorithm, on average.

The results also show that, when the full calibration matrix and signal are available, applying a positivity constraint on the reconstruction algorithms results in even more accurate reconstructions. Among the algorithms studied in this work, only the Kaczmarz algorithm allows applying such a constraint, and in this case, combinations of the different algorithms could be employed for delivering real-time reconstructions for different purposes. For applications in which the system's parameters are fixed and high frame rates are required, the approach employing the parallel SVD might be the best choice, and when the user needs to improve the accuracy of a particular frame (i.e. taking a screenshot), it is possible to run one or more constrained Kaczmarz iterations. Since most of the current CPU architectures contain multiple cores, this calculation could be performed in a separate thread, without even disturbing the scanning and display execution. However, the results with the filtered calibration matrix and signals demonstrated that applying a positivity constraint to reconstructions of such signals would lead to the wrong solutions. In this case, the approach proposed in Henn *et al* (2022) could numerically recover the lost information and allow applying the constraint to the reconstructions, and therefore should be investigated.

A significant finding of this work was to demonstrate that it is possible to compute the calibration matrix and numerically evaluate the system of equations in time domain in the GPU using the parallel reconstruction algorithms in much less time than the serial Kaczmarz evaluation alone, which opens the door for a number of potential real-time applications. It would be possible, for example, to reconstruct images with increased resolutions and sampling rates without worrying about memory space, by simply calculating parts of the calibration matrix as needed. In another scenario, the user might wish to start the

execution performing a broader scan over a bigger field of view, and during the execution, change the scanning parameters to focus on some point of interest, reducing the field of view (zooming in) and increasing the image resolution to reveal more details. In this case the new calibration matrix would be progressively calculated as the system zooms in, without disturbing the frame rate. Another potential application of these algorithms would be to accelerate the computation of the temperature-dependent convolution matrices using the recently reported harmonic PSF approach (Bui *et al* 2023) for 3D temperature imaging. In this case, the dynamic and intensive calculation of PSFs for different temperatures would be required to reconstruct non-uniform temperature maps. The greatest advantage of the parallel computation of the calibration matrix is the flexibility of changing the system's parameters (including the temperature) during the execution. The parallel implementation of the calibration matrix calculation presented in this work is a proof of concept, and more sophisticated implementations, including in frequency domain, can be developed to deliver even faster and more flexible calculations in the future.

To better illustrate the potential of the approach presented in this work, we give an example based on real life applications. Again, for a $20 \times 20 \times 20$ image resolution and a sampling rate of 1.25 MHz, a full scan according to the parameters shown in table 1 would take 4 ms and 15 000 measurements (x, y and z signals), so the problem size would be 1.2×10^8. Assuming 0.1% of noise level and a required accuracy of 10^{-4} with relation to the measured signal, according to table 2 and figure 2 the serial Kaczmarz algorithm would need about 15 iterations and take about 2.7 s to run. The parallel CGNR would need about 22 iterations and take about 140 ms to run, a *speedup* of 19 $\times$. The SVD would take 6 ms, a *speedup* of 450 $\times$. If one chooses to add one Kaczmarz iteration with positivity constraint for improved accuracy (if both the calibration matrix and the signal have not been filtered), the execution times and *speedups* would be 340 ms (7.9 $\times$) and 206 ms (13 $\times$) for the CGNR and SVD executions, respectively. The serial computation of the calibration matrix would take 7.6 s while the parallel computation would take 470 ms, a *speedup* of 16 $\times$. Therefore, the parallel computation of the calibration matrix followed by 22 iterations of CGNR would take 610 ms (4.4 $\times$ faster), and the parallel computation of the calibration matrix followed by 22 CGNR iterations and one Kaczmarz iteration would take 810 ms, still 3.3 $\times$ faster than running 15 iterations of Kaczmarz in the CPU, while showing superior accuracy.

The application of the parallel algorithms to real measurement data provided by the `OpenMPIData` dataset allowed validating the proposed method and demonstrated that it can be applied to measurements performed either in the time or frequency domains, whether or not the signals have been filtered. The execution times shown in figure 5 demonstrated that the parallel implementations are more efficient even for small data sets, which means that it would be advantageous to combine them with techniques such as filtering and row selection, that accelerate reconstructions by reducing the size of the problem. It would also be interesting to explore parallel implementations of other algorithms and regularization schemes reported in the literature, including but not limited to L1 regularization, total variation, and total least squares (Kluth *et al* 2017, Storath *et al* 2017, Kluth 2020). The approach presented in this work can easily be adopted to these and other applications.

In comparison to similar works, this work significantly improved the proposal presented in Quelhas *et al* (2023) by applying it to 3D reconstructions using signals in the presence of noise, and by applying SVD to obtain speedups two orders of magnitude larger. While the proposal in Shen *et al* (2023) implements parallel algorithms to reconstruct multiple frames at the same time, which requires waiting for the signals of several scans before running a batch reconstruction, the approach presented in this work is the first to comprehensively accelerate reconstruction of single frames, allowing for actual real-time, high-resolution, 3D image reconstructions, with great potential for practical applications.

It is very important to highlight that the results reported in this work were obtained using an average GPU (Nvidia Quadro RTX 4000), and much higher speedups are expected from higher performance boards. The adoption of discrete GPUs is a simple solution and a low cost investment relative to the cost of the whole MPI setup, and nowadays they are available with several gigabytes of RAM memory and thousands of cores, allowing for the treatment of more complex and bigger problems in less time.

8. Conclusion

This work presented the parallel implementation in GPU of two fundamental steps in model-based MPI reconstructions: the calculation of the calibration matrix and the image reconstructions algorithms. We investigated the CGNR algorithm, the steepest descent algorithm and SVD. The execution times of the parallel algorithms were compared to the times obtained running the serial calculation of the calibration matrix in the CPU and the serial Kaczmarz algorithm in the CPU without and with positivity constraint. While the parallel steepest descent showed very poor convergence rate despite the high execution speeds, substantial accelerations and good convergence were obtained with the other parallel algorithms, with

reported speedups of up to 37 × for the parallel computation of the calibration matrix and the parallel CGNR, and up to 6,100× for the parallel reconstruction using the pseudoinverse calibration matrix computed with SVD in the GPU. The performance and accuracy of the parallel algorithms was validated by applying them over real measurement data provided by the `OpenMPIData` dataset, with excellent results, demonstrating that the proposed approach is suitable to be applied to data taken either in time or frequency domain, providing single-frame MPI reconstructions in real time. We have found that, if the full calibration matrix and signal are available, a positivity constraint leads to better solutions. Since the Kaczmarz algorithm is the only one that allows adopting such constraint among the algorithms studied in this work, a single constrained Kaczmarz iteration after the parallel SVD or CGNR reconstructions yields equivalent results in much less time, with a speedup of up to 67 × with relation to the serial Kaczmarz algorithm. However, our results showed that filtering the fundamental excitation frequency prevents the adoption of such a constraint, although future work may overcome this limitation. The parallel computation of the calibration matrix reported in this work is so efficient that the computation of the matrix followed by a full CGNR reconstruction is up to 25 × faster than the serial Kaczmarz reconstruction, meaning that it is now possible to compute the calibration matrix in execution time, potentially eliminating the need to store the matrix in memory, and paving the way for much more flexible and sophisticated applications, in which the scanning parameters are changed without interrupting the scan. Future efforts will focus on implementing more efficient algorithms for the calculation of the calibration matrix, including in frequency domain, aiming for dynamic temperature-dependent reconstructions using the harmonic PSF approach for 3D temperature maps, and the parallel implementation of other algorithms and regularization schemes, including the use of matrix preconditioning for the CGNR reconstructions.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary information files).

Acknowledgment

This work was supported by NIST's Innovations in Measurement Science (IMS) grant.

ORCID iDs

Klaus N Quelhas https://orcid.org/0000-0001-9295-4757
Mark-Alexander Henn https://orcid.org/0000-0001-5048-6836
Ricardo Farias https://orcid.org/0000-0002-2411-3101
Weston L Tew https://orcid.org/0000-0002-9979-9136
Solomon I Woods https://orcid.org/0000-0002-1037-3025

References

Aster R C, Borchers B and Thurber C H 2013 *Parameter Estimation and Inverse Problems - 2nd ed* (Elsevier)
Bui T Q, Henn M-A, Tew W L, Catterton M A and Woods S I 2023 Harmonic dependence of thermal magnetic particle imaging *Sci. Rep.* **13** 15762
Censor Y and Zenios S S 1997 *Parallel Optimization: Theory, Algorithms and Applications* (Oxford University Press)
Chen X, Jiang Z, Han X, Wang X and Tang X 2021 The reconstruction of magnetic particle imaging: current approaches based on the system matrix *Diagnostics* **11** 773
Croft L R, Goodwill P W and Conolly S M 2012 Relaxation in x-space magnetic particle imaging *IEEE Trans. Med. Imaging* **31** 2335–42
Eklund A, Dufort P, Forsberg D and LaConte S M 2013 Medical image processing on the GPU—past, present and future *Med. Image Anal.* **17** 1073–94
Elble J M, Sahinidis N V and Vouzis P 2009 GPU computing with Kaczmarz's and other iterative algorithms for linear systems *Parallel Comput.* **36** 215–31
Flores L A, Vidal V, Mayo P, Rodenas F and Verdú G 2014 Parallel CT image reconstruction based on GPUs *Radiat. Phys. Chem.* **95** 247–50
Gleich B and Weizenecker J 2005 Tomographic imaging using the nonlinear response of magnetic particles *Nature* **435** 1214–7
Goodwill P W and Conolly S M 2010 The x-space formulation of the magnetic particle imaging process: 1-D signal, resolution, bandwidth, SNR, SAR and magnetostimulation *IEEE Trans. Med. Imaging* **29** 1851–9
Goodwill P W and Conolly S M 2011 Multidimensional x-space magnetic particle imaging *IEEE Trans. Med. Imaging* **30** 1581–90
Grüttner M, Knopp T, Franke J, Heidenreich M, Rahmer J, Halkola A, Kaethner C, Borgert J and Buzug T M 2013 On the formulation of the image reconstruction problem in magnetic particle imaging *Biomed. Tech.* **58** 583–91
Henn M-A, Quelhas K N, Bui T Q and Woods S I 2022 Improving model-based MPI image reconstructions: baseline recovery, receive coil sensitivity, relaxation and uncertainty estimation *Int. J. Magn. Part. Imaging* **8** 2208001
Hestenes M R and Stiefel E 1952 Methods of conjugate gradients for solving linear systems *J. Res. Natl Bur. Stand.* **49** 409
Kaczmarz S 1937 Angenäherte auflösung von systemen linearer Gleichungen *Bull. Int. Acad. Pol. Sic. Lett.* **35** 355–7

Kluth T 2018 Mathematical models for magnetic particle imaging *Inverse Problems* **34** 083001
Kluth T 2020 L1 data fitting for robust reconstruction in magnetic particle imaging: quantitative evaluation on Open MPI dataset *Int. J. Magn. Part. Imaging* **6** 2009068
Kluth T and Jin B 2019 Enhanced reconstruction in magnetic particle imaging by whitening and randomized SVD approximation *Phys. Med. Biol.* **64** 125026
Kluth T and Maass P 2017 Model uncertainty in magnetic particle imaging: nonlinear problem formulation and model-based sparse reconstruction *Int. J. Magn. Part. Imaging* **3** 1707004
Knopp T, Biederer S and Sattel T F 2010b 2D model-based reconstruction for magnetic particle imaging *Med. Phys.* **37** 485–91
Knopp T, Biederer S, Sattel T and Buzug T M 2008 Singular Value Analysis for Magnetic Particle Imaging *IEEE Nuclear Science Symp.* (IEEE)
Knopp T, Biederer S, Sattel T, Weizenecker J, Gleich B, Borget J and Buzug M 2009 Trajectory analysis for magnetic particle imaging *Phys. Med. Biol.* **54** 385–97
Knopp T and Buzug T M 2012 *Magnetic Particle Imaging* (Springer)
Knopp T, Gdaniec N and Moddel M 2017 Magnetic particle imaging: from proof of principle to preclinical applications *Phys. Med. Biol.* **62** R124–78
Knopp T and Hofmann M 2016 Online reconstruction of 3D magnetic particle imaging data *Phys. Med. Biol.* **61** N257–67
Knopp T, Rahmer J, Sattel T F, Biederer S, Weizenecker J, Gleich B Borgert J and Buzug T M 2010 Weighted iterative reconstruction for magnetic particle imaging *Phys. Med. Biol.* **55** 1577–89
Knopp T, Sattel T F, Biederer S, Rahmer J, Weizenecker J, Gleich B, Borgert J and Buzug T M 2010a Model-based reconstruction for magnetic particle imaging *IEEE Trans. Med. Imaging* **29** 12–18
Knopp T, Szwargulski P, Griese F and Greaser M 2020 OpenMPIData: an initiative for freely accessible magnetic particle imaging data *Data Brief* **28** 7
Lampe J, Bassoy C, Rahmer J, Weizenecker J, Voss H, Gleich B and Borgert J 2012 Fast reconstruction in magnetic particle imaging *Phys. Med. Biol.* **57** 1113–34
Lu K, Goodwill P W, Saritas E U, Zheng B and Conolly S M 2013 Linearity and shift invariance for quantitative magnetic particle imaging *IEEE Trans. Med. Imaging* **32** 1565–75
März T and Weinmann A 2016 Model-based reconstruction for magnetic particle imaging in 2D and 3D *Inverse Problems Imaging* **10** 1087–110
Quelhas K N, Henn M-A, Bui T Q, Wages H R, Tew W L and Woods S I 2022 Flexible software for rigorous simulations of magnetic particle imaging systems *Int. J. Magn. Part. Imaging* **8** 2203081
Quelhas K N, Henn M-A, Farias R C, Tew W L and Woods S I 2023 Parallel MPI image reconstructions in GPU using CUDA *Int. J. Magn. Part. Imaging* **9** 2303043 .
Quelhas K N, Henn M-A, Farias R, Tew W L and Woods S I 2023b Parallel 3D temperature image reconstruction using multi-color magnetic particle imaging (MPI) *Temperature* (accepted)
Rao S *et al* 2019 Remotely controlled chemomagnetic modulation of targeted neural circuits *Nat. Nanotechnol.* **14** 967–73
Schomberg H 2010 Magnetic particle imaging: model and reconstruction *2010 IEEE Int. Symp. on Biomedical Imaging: From Nano to Macro* pp 992–5
Shen Y, Zhang L, Shang Y, Jia G, Yin L, Zhang H, Tian J, Yang G and Hui H 2023 An adaptive multi-frame parallel iterative method for accelerating real-time magnetic particle imaging reconstruction *Phys. Med. Biol.* **68** 245016
Shewchuk J R 1994 An introduction to the conjugate gradient method without the agonizing pain *Technical Report* (Carnegie Mellon University)
Soumare H, Benkahla A and Gmati N 2021 Deep learning regularization techniques to genomics data *Array* **11** 100068
Stone S, Haldar J, Tsao S, Hwu W, Sutton B and Liang Z-P 2008 Accelerating advanced MRI reconstructions on GPUs *J. Parallel Distrib. Comput.* **68** 1307–18
Storath M, Brandt C, Hofmann M, Knopp T, Salamon J, Weber A and Weinmann A 2017 Edge preserving and noise reducing reconstruction for magnetic particle imaging *IEEE Trans. Med. Imaging* **36** 74–85
Strohmer T and Vershynin R 2006 A randomized solver for linear systems with exponential convergence *Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques* 499–507
Tay Z W *et al* 2019 Magnetic particle imaging-guided heating *in vivo* using gradient fields for arbitrary localization of magnetic hyperthermia therapy *ACS Nano* **12** 3699–713
Vogel P, Herz S, Kampf T, Rückert M A, Bley T A and Behr V C 2017 Low latency real-time reconstruction for MPI systems *Int. J. Magn. Part. Imaging* **3** 1707002
Vogel P, Kampf T, Rückert M A and Behr V C 2016 Flexible and dynamic patch reconstruction for traveling wave magnetic particle imaging *Int. J. Magn. Part. Imaging* **2** 1611001
Wang Z, Bovik A, Sheikh H and Simoncelli E 2004 Image quality assessment: from error visibility to structural similarity *IEEE Trans. Image Process.* **13** 600–12
Weaver J B, Rauwerdink A M and Hansen E W 2009 Magnetic nanoparticle temperature estimation *Med. Phys.* **36** 1822–9
Weizenecker J, Borget J Gleich B 2007 A simulation study on the resolution and sensitivity of magnetic particle imaging *Phys. Med. Biol.* **52** 6363–74
Weizenecker J, Gleich B, Rahmer J, Dahnke H and Borget J 2009 Three-dimensional real-time *in vivo* magnetic particle imaging *Phys. Med. Biol.* **54** L1–L10
Wirgin A 2018 The inverse crime (arXiv:math-ph/0401050)