Fast optimization methods for image registration in adaptive radiation therapy

Qiao, Y.

4

An efficient preconditioner for stochastic gradient descent optimization of image registration

This chapter was adapted from:

Y. Qiao, B.P.F. Lelieveldt and M. Staring. **An efficient preconditioner for stochastic gradient descent optimization of image registration**, submitted

Abstract

Stochastic gradient descent (SGD) is commonly used to solve (parametric) image registration problems. In case of ill-conditioned problems, SGD however only exhibits sublinear convergence properties. In this chapter we propose an efficient preconditioner estimation method to improve the convergence rate of SGD. Based on the observed distribution of voxel displacements in the registration, we estimate the diagonal entries of a preconditioning matrix, thus rescaling the optimization cost function. The preconditioner is suitable for stochastic and not only deterministic optimization. It is efficient to compute and employ, and can be used for mono-modal as well as multi-modal cost functions, in combination with different transformation models like the rigid, affine and B-spline model. Experiments on different clinical data sets show that the proposed method indeed improves the convergence rate compared to SGD with speedups around 5 in all tested settings, while retaining the same level of registration accuracy.

4.1 Introduction

Image registration is widely used in medical image analysis and has ample application, e.g. in radiation therapy and segmentation [19, 2, 3]. This procedure can be used to align images from different modalities or different time points following a continuous deformation strategy. The strategy can be formulated as a (parametric) optimization problem to minimize the dissimilarity between a d-dimensional fixed image I_F and moving image I_M:

$$\widehat{\boldsymbol{\mu}} = \arg\min_{\boldsymbol{\mu}} \mathscr{C}(I_F, I_M \circ \boldsymbol{T}_{\boldsymbol{\mu}(\boldsymbol{x})}), \tag{4.1}$$

in which $\boldsymbol{T}_{\boldsymbol{\mu}}(\boldsymbol{x})$ is a coordinate transformation parameterized by $\boldsymbol{\mu}$. An iterative scheme is typically used to solve this problem:

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k - \gamma_k \boldsymbol{d}_k, \tag{4.2}$$

where k is the iteration number, γ_k is the step size at iteration k, and $\boldsymbol{d}_k$ is a search direction in the parameter space. Commonly used methods to determine the search direction $\boldsymbol{d}_k$ are of first order (gradient descent) or second order (Newton or quasi-Newton) descent type. Gradient descent, however, only achieves a sublinear convergence rate for nonconvex problems or a linear convergence rate for convex problems [79, 25]. Especially for badly scaled cost functions these methods converge slowly. Second order derivative methods such as the quasi-Newton method converge faster, however, the computation of the Hessian matrix update is very time consuming, especially when the number of image voxels and transformation parameters are large [80]. To overcome these shortcomings, preconditioning techniques were proposed to turn a badly scaled cost function into a properly scaled cost function, considering the curvature of the cost function [79, 81, 82, 83]. The construction of these preconditioners can however be computationally expensive in themselves, which can easily mitigate the positive effect of faster convergence.

Two major groups of preconditioning techniques are widely used in iterative optimization. One, sometimes named variable preconditioning, uses the update rule:

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k - \gamma_k \boldsymbol{P}_k \boldsymbol{g}_k. \tag{4.3}$$

The preconditioner $\boldsymbol{P}_k$ is updated at each iteration (or at least regularly) to adapt to the local shape of the cost function [84, 85, 86, 87, 88, 89, 90]. This group of methods is typically used in machine learning to solve a linear system [91, 92, 85, 87, 93, 94], but is also popular in image registration [95, 96, 25, 97]. Popular preconditioners, such as Newton or quasi-Newton methods [82, 89], indeed exhibit superior convergence rate compared to the standard gradient descent methods. These improvements, however, come at a cost of the estimation of the inverse Hessian, which alleviates some of the advantages and can even lead to a net deceleration. Zikic *et al.* [89] proposed a diagonal preconditioner for Demons registration. They applied the preconditioner before the dense gradient of the energy function using the inverse of the gradient magnitude. Besides its extra computation efforts at each iteration, its performance mainly depends on the choice of parameter ρ. This parameter is problem specific for different dissimilarity measures, different modalities and different transformation models, which may limit its practicality.

Another group of preconditioning techniques, sometimes called traditional preconditioning, use a static $\boldsymbol{P}$, i.e. the preconditioner $\boldsymbol{P}$ is only calculated once before the start of the optimization [83, 79, 85]. The Krylov subspace method, sparse approximate inverse and Jacobi preconditioning techniques are often used [83]. Klein *et al.* proposed a preconditioner construction method only suitable for mono-modal image registration [70], which approximates the Hessian matrix of the cost function based on an assumption that the intensity difference between moving image and fixed image is zero after a perfect registration. This method is additionally very time-consuming when the number of transformation parameters and image size increase: the required matrix decomposition of the Hessian matrix takes more than 3 hours for $\sim 10^5$ parameters.

As image registration is time crucial for several clinical applications, for example online adaptive radiation therapy [98], it is advantageous to find an efficient way to obtain a search direction $\boldsymbol{d}_k$ and its preconditioner $\boldsymbol{P}$. For registration problems with large degrees of freedom and of large images, it is not very efficient to calculate the search direction in a deterministic way [25] (i.e. using all voxels to compute the gradient). Klein *et al.* proposed a stochastic gradient descent method for image registration, which approximates the gradient by only using a random subset of the image samples [15]. This approximation is much more efficient to compute, thereby outperforming deterministic gradient descent and even quasi-Newton methods [25]. For ill-conditioned problems, however, SGD does not provide a solution and would still suffer from a deteriorated convergence rate.

In this chapter, we consider the preconditioned stochastic gradient descent method (PSGD) that calculates the preconditioner only once. Based on a connection between the incremental displacement of a voxel and the gradient change between iterations, we propose an efficient method to construct a diagonal preconditioner for stochastic gradient descent methods. The chapter is organized as follows. The background and proposed method are given in Section 4.2 and Section 4.3. The dataset used to evaluate the proposed method is described in Section 4.4. This is followed by the experimental setup in Section 4.5 and the results in Section 4.6. The discussion and conclusion are given in Section 4.7 and Section 4.8.

4.2 Background

4.2.1 Preconditioned stochastic gradient descent

The preconditioned stochastic gradient descent method is established as:

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k - \gamma_k \boldsymbol{P} \tilde{\boldsymbol{g}}_k, \tag{4.4}$$

where γ_k is the step size, $\tilde{\boldsymbol{g}}_k$ is a stochastic gradient evaluated on a random subset of the image samples Ω_F^s and $\boldsymbol{P}$ is a positive definite $N_P \times N_P$ matrix, with N_P the number of parameters that model the transformation, i.e. $|\boldsymbol{\mu}|$. When $\boldsymbol{P} = \boldsymbol{I}$, PSGD will be reduced to the standard SGD method.

The convergence of PSGD is guaranteed when (1) $\boldsymbol{P}$ is positive definite; and (2) the step size sequence is a non-increasing and non-zero sequence with $\sum_{k=1}^{\infty} \gamma_k = \infty$ and $\sum_{k=1}^{\infty} \gamma_k^2 < \infty$ [99, 86, 85]. The step size sequence used here is defined as follows

[70]:

$$\begin{aligned} \gamma_k &= \frac{\eta}{(t_k+1)/A+1}, \\ t_k &= \max(0, t_{k-1} + \text{sigmoid}(-\tilde{\boldsymbol{g}}_{k-1}^T \boldsymbol{P} \tilde{\boldsymbol{g}}_{k-2})), \end{aligned} \tag{4.5}$$

in which η is a noise compensation factor and A controls the decay speed of the step size sequence and is typically set to 20. The noise introduced by the stochastic procedure will influence the convergence rate, so inspired from [70, 100] we use the following compensation factor:

$$\eta = \frac{E\|\boldsymbol{g}^T \boldsymbol{P} \boldsymbol{g}\|}{E\|\tilde{\boldsymbol{g}}^T \boldsymbol{P} \tilde{\boldsymbol{g}}\|} = \frac{E\|\boldsymbol{g}^T \boldsymbol{P} \boldsymbol{g}\|}{E\|\boldsymbol{g}^T \boldsymbol{P} \boldsymbol{g}\| + E\|\boldsymbol{\epsilon}^T \boldsymbol{P} \boldsymbol{\epsilon}\|}, \tag{4.6}$$

in which $\boldsymbol{g}$ is the exact gradient evaluated on all voxels in the image, $\boldsymbol{\epsilon}$ the random noise added to the exact gradient and $E\|\cdot\|$ is the expectation of the norm.

4.2.2 Related work

There are two related methods to estimate a preconditioner:

1. Hessian-type preconditioner (PSGD-H). The theoretical optimal choice for the preconditioner is the inverse Hessian at the optimal parameter $\hat{\boldsymbol{\mu}}$. However, it is impossible to obtain the exact inverse Hessian beforehand because $\hat{\boldsymbol{\mu}}$ is unknown [70]. Based on the assumption that the moving image is the same as the fixed image after successful registration: $\boldsymbol{F} \approx \boldsymbol{M}(\boldsymbol{T}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}))$, and the assumption that the deformation is small: $\partial \boldsymbol{T} / \partial \boldsymbol{\mu} \approx \boldsymbol{I}$, Klein *et al.* proposed a method to approximate the Hessian-type preconditioner [70]. This method requires an implementation to calculate a Hessian matrix and a decomposition to construct the preconditioner. This method is only suitable for mono-modal image registration. Moreover, the computation time of this preconditioner is very long when solving large scale problems, which defeats the improvements in the convergence.

2. Jacobi-type preconditioner (PSGD-J). For rigid and affine registration problems, Klein *et al.* [70] assumed that the rotation parameters were scaled by the average voxel displacement caused by a small perturbation of the rotation angle, and proposed a method to construct a diagonal Jacobi-type preconditioner for PSGD. The elements p_i of the diagonal preconditioner $\boldsymbol{P}$ are calculated as follows:

$$p_i = \left(\int_{\Omega_F} \left\| \frac{\partial \boldsymbol{T}}{\partial \theta_i}(\boldsymbol{x}; \boldsymbol{\mu}_0) \right\|^2 \mathrm{d}x \Big/ \int_{\Omega_F} \mathrm{d}x \right)^{-\frac{1}{2}}. \tag{4.7}$$

   This method can be used for multi-modal image registration, however, it was proposed for rigid and affine registration only.

4.3 Method

4.3.1 Preliminaries

The aim of the preconditioner $\boldsymbol{P}$ is to scale the parameter space to make ill-posed cost functions easier to optimize. The ideal preconditioner should take care of the relative scaling between the parameters. Construction of a suitable preconditioner is a

challenge for a given problem. First, different transformation models and different dissimilarity measures result in different characteristic of the cost function, making the determination of a preconditioner problem-specific. Second, the computation of the preconditioner should be efficient performance-wise, otherwise the overhead of the preconditioner computation will defeat the advantage in runtime reductions obtained from the improvements of the convergence rate.

To find a suitable approximation of $\boldsymbol{P}$ in a computationally efficient way, and robust for different cost functions, a diagonal preconditioner $\boldsymbol{P} = \text{diag}(\boldsymbol{p})$, with $\boldsymbol{p} = (p_1, \ldots, p_N)$, is preferred. In this chapter, we propose a novel way to construct this diagonal preconditioner, which is suitable for both stochastic and deterministic optimization and can be used for mono-modal as well as multi-modal cost functions, in combination with different transformation models like the rigid, affine and B-spline model.

The intuition of the proposed preconditioner is that a gradient change will result in incremental voxel displacements, which is inspired by [15, 100]. In the following we will derive the i-th entry p_i of the preconditioner corresponding to the i-th entry of the transformation parameters $\boldsymbol{\mu}$, such that the displacement induced by a change in that parameter is equal to a predefined value δ. The incremental displacement of a voxel $\boldsymbol{x}_j$ in the fixed image domain Ω_F between iteration k and $k+1$ for an iterative optimization scheme is defined as:

$$\boldsymbol{d}_k(\boldsymbol{x}_j) = \boldsymbol{T}\left(\boldsymbol{x}_j, \boldsymbol{\mu}_{k+1}\right) - \boldsymbol{T}\left(\boldsymbol{x}_j, \boldsymbol{\mu}_k\right), \quad \forall \boldsymbol{x}_j \in \Omega_F. \tag{4.8}$$

We approximate the incremental displacement $\boldsymbol{d}_k$ using the first-order Taylor expansion around $\boldsymbol{\mu}_k$:

$$\begin{aligned} \boldsymbol{d}_k(\boldsymbol{x}_j) &\approx \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{\mu}}\left(\boldsymbol{x}_j, \boldsymbol{\mu}_k\right) \cdot \left(\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k\right) \\ &= \boldsymbol{J}(\boldsymbol{x}_j) \cdot \left(\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k\right), \end{aligned} \tag{4.9}$$

in which $\boldsymbol{J}(\boldsymbol{x}_j) = \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{\mu}}\left(\boldsymbol{x}_j, \boldsymbol{\mu}_k\right)$ is the Jacobian matrix of size $d \times N_P$. Using the optimization scheme (4.4), we obtain $\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k = -\gamma_k \boldsymbol{P}\boldsymbol{g}_k$, and we can rewrite $\boldsymbol{d}_{k+1}$ as:

$$\boldsymbol{d}_k(\boldsymbol{x}_j) \approx -\gamma_k \boldsymbol{J}(\boldsymbol{x}_j)\boldsymbol{P}\boldsymbol{g}_k. \tag{4.10}$$

4.3.2 Diagonal preconditioner estimation

From Equation (4.10), we notice that the preconditioner matrix $\boldsymbol{P}$ can be estimated prior to registration, i.e. at iteration $k = 0$. After choosing $\gamma_0 = 1$, we obtain $\boldsymbol{d}_1(\boldsymbol{x}_j) \approx -\boldsymbol{J}(\boldsymbol{x}_j)\text{diag}(\boldsymbol{p})\boldsymbol{g}_0$. In the remainder of the chapter, we use the notation $\boldsymbol{d}$ and $\boldsymbol{g}$ for simplification.

The Jacobi-type preconditioner from Equation (4.7) can be rewritten to:

$$p_i = \left(E\|\boldsymbol{J}^i(\boldsymbol{x}_j)\|^2\right)^{-1/2}, \tag{4.11}$$

where $\boldsymbol{J}^i(\boldsymbol{x}_j)$ denotes the i-th column of the Jacobian matrix. Inspired by Equation (4.11) we inspect the displacement $\|\boldsymbol{d}^i\|$ that is induced by a change $\triangle\boldsymbol{\mu}_i$ in the i-th transformation parameter, i.e. the displacement generated by g^i only:

$$\|\boldsymbol{d}^i(\boldsymbol{x}_j)\| \approx \left\| -\boldsymbol{J}^i(\boldsymbol{x}_j)\, p_i\, g^i \right\| = p_i \left\| -\boldsymbol{J}^i(\boldsymbol{x}_j)\, g^i \right\|, \tag{4.12}$$

Algorithm 2 Proposed preconditioner estimation

Require: N_s the number of samples, δ the maximum allowed voxel displacement, τ the regularization factor, $\kappa_{\max}$ the maximum condition number

Compute the gradient $\boldsymbol{g}$ of size N_P
Randomly take N_s samples $\{\boldsymbol{x}_j\}$ from the fixed image
$\boldsymbol{p} = \boldsymbol{I}$, $\boldsymbol{t} = \boldsymbol{0}$, $\boldsymbol{z} = \boldsymbol{0}$, $\boldsymbol{y} = \boldsymbol{0}$ ▷ initialization
for $j = 1, 2, \ldots, N_s$ **do** ▷ loop over the samples $\boldsymbol{x}_j$
    Calculate the Jacobian $\boldsymbol{J}(\boldsymbol{x}_j)$
    for $i = 1, 2, \ldots, N_P$ **do** ▷ loop over the parameters
        $s_i = \|\boldsymbol{J}^i(\boldsymbol{x}_j) g^i\|$
        Regularize s_i with τ using Section 4.3.3
        $z_i = z_i + s_i$ ▷ update for the mean
        $y_i = y_i + s_i^2$ ▷ update for the variance
        $t_i = t_i + 1$ ▷ increase counter
for $i = 1, 2, \ldots, N_P$ **do** ▷ loop over the parameters
    $q_i = z_i / t_i + 2\sqrt{(y_i / t_i) - (z_i / t_i)^2}$
    $p_i = \delta / q_i$
    Constrain the condition number of p_i using $\kappa_{\max}$ (see Section 4.3.4)
Return $\boldsymbol{p}$

in which $\|\cdot\|$ used in this chapter is the ℓ_1 norm. In medical image registration, we expect a continuous and homogenous transformation and moreover assume that the voxel displacement $\boldsymbol{d}^i$ is to be not larger than δ: i.e $\|\boldsymbol{d}^i(\boldsymbol{x}_j)\| \leq \delta, \quad \forall \boldsymbol{x}_j \in \Omega_F$. Based on the distribution of the voxel displacements, there is a weakened form for this assumption: $P(\|\boldsymbol{d}^i(\boldsymbol{x}_j)\| > \delta) < \rho$, where ρ is a small probability value often 0.05. According to the Vysochanskij-Petunin inequality [46], we have the following expression:

$$E\|\boldsymbol{d}^i(\boldsymbol{x}_j)\| + 2\sqrt{Var\|\boldsymbol{d}^i(\boldsymbol{x}_j)\|} \leq \delta, \quad \forall \boldsymbol{x}_j \in \Omega_F. \tag{4.13}$$

Combined with Equation (4.12), we obtain the relationship between the i-th entry p_i of the preconditioner and the maximum voxel displacement as follows:

$$p_i \left(E\left\| s_i(\boldsymbol{x}_j) \right\| + 2\sqrt{Var\left\| s_i(\boldsymbol{x}_j) \right\|} \right) \leq \delta, \tag{4.14}$$

where $s_i(\boldsymbol{x}_j) = \| - \boldsymbol{J}^i(\boldsymbol{x}_j) g^i \|$. The i-th entry of the preconditioner is then defined as:

$$p_i = \frac{\delta}{E\left\| s_i(\boldsymbol{x}_j) \right\| + 2\sqrt{Var\left\| s_i(\boldsymbol{x}_j) \right\|}}. \tag{4.15}$$

Finally, the full preconditioner $\boldsymbol{P}$ is obtained by repeating the above procedure for each p_i. The procedure is sketched in Algorithm 2.

4.3.3 Regularization

The assumption used to approximate a preconditioner, that all transformation parameters should independently induce a maximum voxel displacement δ, may be too strict

or too sensitive to noise in the measurements. For the B-spline transformation, fore example, this assumption forces all regions to have a displacement δ, even regions that do not require registration. Noise could come from an insufficient number of samples $\boldsymbol{x}_j$ used for the estimation, or from inexact evaluation of the gradient. This could result in differences in the estimated entries of the preconditioner that are expected to have similar value. For the B-spline transformation model one would expect that nearby control points would be scaled similarly, without sudden sharp transitions. For the affine transformation on the other hand, one would expect that scalings related to translation parameters are more similar than those related to rotational parameters.

We therefore propose to optionally regularize the procedure from Section 4.3.2, such that the i-th entry p_i of the preconditioner is not treated completely independent, but also takes into account the estimates of the related parameters. Related parameters are those jointly affected by a voxel $\boldsymbol{x}_j$ (for an affine transformation these are all parameters; for the B-spline only parameters in the compact support region of $\boldsymbol{x}_j$), and secondly by their similarity in Jacobian contribution (for the affine transformation, intuitively rotations and translations are to be treated separately). The proposed regularization procedure is as follows:

$$s_i(\boldsymbol{x}_j) = \tau \cdot s_i(\boldsymbol{x}_j) + (1-\tau) \cdot \underbrace{\frac{1}{\sum \omega_m} \sum_{m \neq i} s_m(\boldsymbol{x}_j) \cdot \omega_m}_{\text{regularization term}}, \tag{4.16}$$

where ω_m weighs the contributions of similar parameters and τ balances the contribution of entry i with the contributions of the other parameters. The weights ω_m are calculated using a Gaussian function:

$$\omega_m = \exp\left(-\frac{(\|\boldsymbol{J}^i(\boldsymbol{x}_j)\| - \|\boldsymbol{J}^m(\boldsymbol{x}_j)\|)^2}{2\sigma^2} \right), \tag{4.17}$$

in which σ is chosen as $\min(\|\boldsymbol{J}^i(\boldsymbol{x}_j)\| - \|\boldsymbol{J}^m(\boldsymbol{x}_j)\|) / \max(\|\boldsymbol{J}^i(\boldsymbol{x}_j)\| - \|\boldsymbol{J}^m(\boldsymbol{x}_j)\|)$, $\forall m \neq i$.

While for the B-spline transformation model such a choice would also be valid, a simplification is possible. For the B-spline model the displacement of a voxel is only determined by the control points in its support region. Furthermore, we expect the influence on the displacement to be almost equal for each control point in the support region. We therefore assume for the B-spline model that the weights $\omega_m = 1$, simplifying Equation (4.16) to $s_i(\boldsymbol{x}_j) = \tau \times s_i(\boldsymbol{x}_j) + (1-\tau) \cdot \|\sum(\boldsymbol{J}^i(\boldsymbol{x}_j) g^i)\|$.

4.3.4 Condition number

Even if the resulting preconditioner is symmetric and positive definite, it could be ill-conditioned, especially for nonrigid image registration problems. The convergence rate of the algorithm can be measured by the so-called condition number:

$$\kappa = \lambda_{\max} / \lambda_{\min}, \tag{4.18}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalue of $\boldsymbol{P}$, respectively. It is common to constrain the eigenvalues, such that the condition number will be closer to 1 [70, 88]. We introduce a user-defined maximum condition number $\kappa_{\max}$ for this purpose.

Define a diagonal eigenvalue matrix $\Lambda = diag(\lambda_1, \ldots, \lambda_{N_p})$ for the preconditioner $\boldsymbol{P}$. In this study, as our preconditioner $\boldsymbol{P}$ is diagonal, the entries of $\boldsymbol{P}$ are equal to the eigenvalues of Λ: $p_i = \lambda_i, \forall i$. To constrain the eigenvalues, we replace small eigenvalues of $\boldsymbol{P}$ that make $\kappa > \kappa_{\max}$ using the following equation:

$$p_i = \begin{cases} \lambda_{\max}/\kappa_{\max}, & \text{if } \lambda_{\max}/\lambda_i > \kappa_{\max}, \\ \lambda_i, & \text{otherwise.} \end{cases} \tag{4.19}$$

The thus constrained matrix constitutes then the finally proposed static preconditioner. Combined with Equation (4) this defines the Fast Preconditioned Stochastic Gradient Descent method (FPSGD).

4.4 Data sets

The proposed FPSGD method is tested on mono-modal as well as multi-modal data. An overview of the used data sets is presented in Table 4.1.

4.4.1 Mono-modal lung data: SPREAD

3D lung Computed Tomography (CT) images of 19 patients were acquired during the SPREAD study [77]. A follow-up scan was acquired for each patient after the baseline scan with image sizes around $450 \times 300 \times 150$ and voxel sizes around $0.7 \times 0.7 \times 2.5$ mm. The ground truth consists of 100 anatomical corresponding points, which were semi-automatically extracted using Murphy's method [49]. The algorithm first automatically selects 100 evenly distributed landmarks at characteristic locations in the baseline image, and then predicts the corresponding points in the follow-up image. The corresponding points are then inspected and corrected by two experts using a graphical user interface [50].

4.4.2 Multi-modal brain data: RIRE and BrainWeb

Two multi-modal datasets are used to evaluate the performance of the proposed method.

4.4.2.1 RIRE brain data

This brain dataset was acquired during the Retrospective Image Registration Evaluation (RIRE) project. CT scans and Magnetic Resonance Imaging (MRI-T1) are available for 9 patients. The CT images have sizes of $512 \times 512 \times 50$ with voxel sizes of $0.45 \times 0.45 \times 3$ mm, while the MRI-T1 image is of size $256 \times 256 \times 50$ with voxel sizes of $0.85 \times 0.85 \times 3$ mm. Fiducial markers were implanted in each patient and served as a ground truth [47]. These markers were manually erased from the images and replaced with a simulated background pattern.

4.4.2.2 BrainWeb simulated brain data

T1 and T2 weighted 3D brain MR images were created using the Simulated Brain Database from BrainWeb [101]. To generate brain image pairs, default settings provided by BrainWeb were used with 3% noise and 20% intensity non-uniformity. The brain images are of sizes $181 \times 217 \times 181$ and a voxel spacing of 1 mm isotropically. A mask of the brain was extracted from the T1 image by FSL-BET [102] and the same mask was used for the T2 image. 100 randomly generated displacement vector fields (DVFs) serve as the ground truth deformation fields. The DVFs are isotropically

Table 4.1: Overview of data sets and experimental setup.

	Mono-modal	Multi-modal	
Dataset	SPREAD	RIRE	BrainWeb
Anatomy	Lung	Brain	Brain
Modality	CT	CT and 1.5T MR T1	MR T1 and T2
Dimensions	450 × 300 × 130	CT: 512 × 512 × 50 MR: 256 × 256 × 50	181 × 217 × 181
Voxel size (mm)	~ 0.7 × 0.7 × 2.5	CT: 0.45 × 0.45 × 3 MR: 0.85 × 0.85 × 3	1 × 1 × 1
Number of patients	21	9	1 × 100
Similarity measure	MSD	MI	MI
Transformation	Affine, B-spline	Rigid	B-spline
B-spline control point grid spacing (mm)	10 × 10 × 10	-	10 × 10 × 10
Number of parameters (last resolution)	~ 90k	6	36300
Ground truth	100 corresponding points	8 corner points	100 simulated deformations
Evaluation measure	Euclidean distance	Euclidean distance	Residuals

generated in three dimensions within the brain mask and the maximum magnitude of DVFs is chosen as 8 and 15 mm. These DVFs are then smoothed by a Gaussian filter with a standard deviation between 10 and 30 mm.

4.5 Experiments

In this section, experimental settings are given to test the performance of the proposed method. The proposed FPSGD method is compared with the following methods:

1. Fast adaptive stochastic gradient descent (FASGD) [100], which is a state-of-the-art first order stochastic optimization method that does not use preconditioning. For rigid and affine registration, the diagonal of the preconditioner $\boldsymbol{P}$ is chosen as 1 for the translational parameters and 1/100000 for the others. This reflects that the parameters $\boldsymbol{\mu}$ corresponding to rotation have in general a much smaller range than parameters corresponding to translation.

2. Jacobi-type preconditioner (PSGD-J) [70], where a diagonal preconditioner is chosen according to Equation (4.7). This method was only proposed for rigid and affine registration.

3. Hessian-type preconditioner (PSGD-H) [70], see Section 4.2.2. This preconditioner is only suitable for mono-modal registration, and therefore only implemented for the mean squared intensity difference (MSD) dissimilarity measure.

All these methods, including the proposed method, were implemented in C++ and are available as open source software via the `elastix` package [10]. All experiments were performed on a workstation with an Ubuntu Linux OS, which has 8 cores running at 2.4 GHz and 24 GB of memory. Detailed settings are presented in Section 4.5.3 and 4.5.4, and an overview of the experimental setup is given in Table 4.1.

4.5.1 Experimental setup

To validate the generality of the proposed preconditioner, the experiments are performed on mono-modal as well as multi-modal image registration. For each group, different transformation models are used, namely the rigid, affine and B-spline transformation models [10]. For rigid and affine image registration, only one resolution of 500 iterations is used, to be able to more easily compare convergence properties. For B-spline image registration, a three-level multi-resolution framework is used on the SPREAD data with a standard deviation of the Gaussian smoothing filter of 2, 1 and 0.5 mm, and 500 iterations for each resolution. For the BrainWeb data, we used only one resolution of 1000 iterations.

The number of samples used for computing $\tilde{\boldsymbol{g}}$ was the same for all methods and set to 5000 [100]. Different methods used different number of samples for the preconditioner estimation. For FPSGD 50000 samples were used at each resolution. For PSGD-H the number of samples were 100000, 100000 and 500000 for the three resolutions, respectively. For PSGD-J, 1000 samples were used. To estimate the step size of FASGD, the number of samples was chosen equal to the number of transformation parameters $\|\boldsymbol{\mu}\|$ at each resolution, for instance in the SPREAD experiment around 4000, 15000 and 90000 samples for the three resolutions, respectively. The user pre-defined value δ for FASGD and FPSGD is chosen as the mean length of the voxel size. $A = 20$ is used for all tested methods.

In Section 4.3, we introduced two free parameters of the proposed FPSGD method: the regularization factor τ and the maximum condition number $\kappa_{\max}$. To assess the influence of these two parameters on the results, we first vary the regularization factor τ while using a fixed $\kappa_{\max}$, and then vice versa. The regularization factor τ was selected between 0 and 1, using increments of 0.2, so there were 6 variations. For these tests, $\kappa_{\max} = 2$ was chosen for the B-spline registration, while for rigid and affine registration no restriction is needed on the condition number, i.e. $\kappa_{\max} = \infty$. In the second group of tests, a fixed $\tau = 0.6$ was chosen and $\kappa_{\max} \in \{1, 2, 4, 8, 16\}$ were tested for the B-spline registrations of the SPREAD data and the BrainWeb data. The results are reported in Section 4.6.1.

4.5.2 Convergence and runtime performance

The performance of the tested methods is first evaluated in terms of the convergence rate and the resulting speed-up in runtime. To measure the convergence rate, the dissimilarity measure (MSD or MI) was calculated at each 5^{th} iteration. This calculation was performed deterministically using all samples from the fixed image. FASGD is chosen as the baseline method and we compare the exact cost function value of all other methods against the exact cost function value of FASGD at its final solution $\widehat{\boldsymbol{\mu}}_{\text{ref}}$. For each method, we counted the number of iterations I required to obtain a cost function value that is equal to or smaller than that of the baseline method using $\mathscr{C}(\boldsymbol{\mu}_k) \leq \mathscr{C}(\widehat{\boldsymbol{\mu}}_{\text{ref}})$ for the first time.

To assess runtime performance, several computations are timed and recorded: the time t_{est} it takes to estimate the preconditioner $\boldsymbol{P}$ and the time t_{iter} each iteration takes. When I equals the number of iterations needed for reaching the same cost function value as FASGD, then the pure registration time is defined as $t_{\text{pure}} = t_{\text{iter}} \cdot I$. The total registration time is then $t_{\text{total}} = t_{\text{est}} + t_{\text{pure}}$. The time t_{est} consists of the time to estimate the preconditioner and/or the step size γ_0 for the different methods, i.e. for FASGD t_{est} is the estimation time of the step size, for PSGD-J and PSGD-H both are included and for FPSGD t_{est} is the estimation time of the preconditioner.

4.5.3 Mono-modal image registration: SPREAD

In this experiment we compare the proposed method compared to all three alternative methods: FASGD, PSGD-H and PSGD-J. The baseline and follow-up image were treated as fixed image and moving image, respectively. The Euclidean distance of the 100 corresponding points is computed to evaluate the registration accuracy using $\text{ED} = \frac{1}{100}\sum_{i=1}^{100} \| \boldsymbol{T}_{\widehat{\boldsymbol{\mu}}}(\boldsymbol{p}_F^i) - \boldsymbol{p}_M^i \|$, with $\boldsymbol{p}_F$ and $\boldsymbol{p}_M$ the corresponding points, and $\boldsymbol{T}$ the transformation at iteration I. A Wilcoxon signed-rank test on the registration accuracy is used to evaluate statistical differences of these methods compared to FASGD method.

We use the mean squared intensity difference (MSD) as a dissimilarity measure, and test for affine as well as B-spline transformations.

4.5.4 Multi-modal image registration: RIRE and BrainWeb

For multi-modal image registration, real clinical brain data is used for rigid registration and simulated brain data is used for nonrigid registration. These datasets are used to compare the performance of FPSGD with FASGD and PSGD-J, as PSGD-H is not suitable for multi-modal registration.

4.5.4.1 RIRE brain data

We registered the MR T1 image (moving image) to the CT image (fixed image) using the rigid transformation model and mutual information (MI) dissimilarity measure. The registration accuracy is evaluated using $\text{ED} = \frac{1}{8}\sum_{i=1}^{8} \| \boldsymbol{T}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{p}_F^i) - \boldsymbol{p}_M^i \|$, with $\boldsymbol{p}_F$ and $\boldsymbol{p}_M$ the corner points defined by RIRE and annotated in the fixed and moving image, respectively.

4.5.4.2 BrainWeb simulated brain data

Pairwise B-spline registration was performed using these randomly generated DVFs as the initial transformation $\boldsymbol{T}_{\text{init}}$. The registration accuracy is evaluated using the average residual deformation inside the brain mask Ω_F [103]:

$$Residual(\boldsymbol{T}_{\text{init}}, \boldsymbol{T}_{\hat{\boldsymbol{\mu}}}) = \frac{1}{|\Omega_F|} \sum_{\boldsymbol{x}_i \in \Omega_F} \| \boldsymbol{T}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{T}_{\text{init}}(\boldsymbol{x}_i) - \boldsymbol{x}_i) \|. \tag{4.20}$$

The statistical differences of FASGD and PSGD-J compared to FASGD method were evaluated using a Wilcoxon signed-rank test on the registration accuracy.

4.6 Results

4.6.1 Parameter sensitivity analysis

4.6.1.1 Selection of the regularization factor τ

For all datasets we varied the parameter τ. The results can be found in Table 4.2, Table 4.3, Table 4.4, and Table 4.5. It can be seen that the regularization factor $\tau = 1.0$ (no regularization) gave the worst performance for rigid and affine registration on all datasets. For B-spline registration $\tau = 1.0$ did work for the SPREAD data, but failed again on the BrainWeb data. Setting the regularization factor $\tau = 0.0$ is another extreme meaning that the regularization term completely determines the estimation of the preconditioner. From the results in the tables, it can be seen that the convergence rate is much slower than for other choices of τ, even though the registration accuracy is almost similar.

The experimental results on the different datasets show that there is no statistical difference between the different choices of τ $(0.0 < \tau < 1.0)$ regarding the accuracy. However, the convergence rate is improved when taking a larger value of τ. We therefore conclude that a regularization factor τ between 0.6 and 0.8 gives the best results. In the remainder of the chapter we use $\tau = 0.6$.

4.6.1.2 Influence of the condition number $\kappa_{\max}$

The maximum condition number $\kappa_{\max}$ is especially important for non-rigid registration. Table 4.6 presents the registration accuracy with respect to $\kappa_{\max}$ for the SPREAD study. As we can see, different $\kappa_{\max}$ obtained the similar accuracy. However, less iterations were needed for a larger $\kappa_{\max}$. From the convergence plot in Figure 4.1, it can be observed that the optimization converged faster for $\kappa_{\max} \geq 2$. However, for $\kappa_{\max} \geq 8$, the plot exhibits more oscillating behavior, suggesting a less stable optimization.

For the BrainWeb data in Table 4.7, we again see that registration accuracy is similar for different $\kappa_{\max}$. In Figure 4.2, all choices of $\kappa_{\max}$ converged faster than FASGD, while for $\kappa_{\max} \geq 2$ the convergence rate does not improve further. From Table 4.7 and Figure 4.2, we can see that $\kappa_{\max} = 2$ or 4 gave the best results, which is

Table 4.2: Overall results of affine registration for the SPREAD lung CT data. We used the MSD dissimilarity measure, 3 resolutions, 500 iterations and $\kappa_{\max} = \infty$.

Optimizer	Iterations I avg ± std	t_{est} (s) avg ± std	t_{pure} (s) avg ± std	t_{total} (s) avg ± std	Speed-up avg ± std	ED (mm) avg ± std	p-value
FASGD	489 ± 12	0.11 ± 0.03	1.83 ± 0.19	1.95 ± 0.21	-	4.99 ± 3.42	-
PSGD-J	153 ± 75	0.11 ± 0.02	0.55 ± 0.28	0.66 ± 0.29	3.5 ± 1.6	5.51 ± 3.83	0.036
PSGD-H	31 ± 24	4.14 ± 0.76	0.11 ± 0.09	4.25 ± 0.79	0.5 ± 0.1	4.75 ± 3.21	0.049
FPSGD $\tau = 0.0$	80 ± 92	0.25 ± 0.02	0.29 ± 0.34	0.55 ± 0.34	4.3 ± 1.4	5.02 ± 3.64	0.520
FPSGD $\tau = 0.2$	52 ± 54	0.25 ± 0.02	0.19 ± 0.20	0.43 ± 0.21	5.0 ± 1.3	5.04 ± 3.72	0.260
FPSGD $\tau = 0.4$	39 ± 23	0.25 ± 0.02	0.14 ± 0.08	0.39 ± 0.09	5.2 ± 1.0	5.08 ± 3.60	1.000
FPSGD $\tau = 0.6$	41 ± 21	0.25 ± 0.02	0.15 ± 0.08	0.40 ± 0.09	5.1 ± 1.2	5.14 ± 3.53	0.355
FPSGD $\tau = 0.8$	50 ± 43	0.26 ± 0.03	0.18 ± 0.16	0.44 ± 0.17	4.9 ± 1.5	5.05 ± 3.36	0.658
FPSGD $\tau = 1.0$	109 ± 141	0.25 ± 0.03	0.40 ± 0.53	0.65 ± 0.53	4.2 ± 1.7	4.99 ± 3.24	0.469

Table 4.3: Overall results of B-spline registration for the SPREAD lung CT data. We used the MSD dissimilarity measure, 3 resolutions, 500 iterations and $\kappa_{\max} = 4$.

	Optimizer	Iterations I avg ± std	t_{est} (s) avg ± std	t_{pure} (s) avg ± std	t_{total} (s) avg ± std	Speed-up avg ± std	ED (mm) avg ± std	p-value
Resolution 1	FASGD	496 ± 0	0.92 ± 0.12	11.0 ± 0.16	11.9 ± 0.22	-		
	PSGD-H	23 ± 6	24.2 ± 4.14	0.73 ± 0.22	25.0 ± 4.19	0.5 ± 0.1		
	FPSGD $\tau = 0.0$	414 ± 61	1.15 ± 0.14	9.15 ± 1.29	10.3 ± 1.33	1.2 ± 0.1		
	FPSGD $\tau = 0.2$	358 ± 75	1.14 ± 0.13	7.90 ± 1.72	9.03 ± 1.78	1.4 ± 0.2		
	FPSGD $\tau = 0.4$	294 ± 66	1.15 ± 0.13	6.50 ± 1.49	7.66 ± 1.53	1.6 ± 0.3		
	FPSGD $\tau = 0.6$	225 ± 50	1.13 ± 0.13	4.97 ± 1.10	6.10 ± 1.15	2.0 ± 0.4		
	FPSGD $\tau = 0.8$	157 ± 37	1.14 ± 0.14	3.47 ± 0.85	4.61 ± 0.93	2.7 ± 0.5		
	FPSGD $\tau = 1.0$	97 ± 37	1.13 ± 0.13	2.14 ± 0.90	3.27 ± 0.98	3.8 ± 0.8		
Resolution 2	FASGD	496 ± 0	0.98 ± 0.13	11.3 ± 0.13	12.2 ± 0.21	-		
	PSGD-H	9 ± 15	174 ± 80	1.36 ± 2.47	175 ± 80	0.1 ± 0.0		
	FPSGD $\tau = 0.0$	306 ± 101	1.38 ± 0.16	6.99 ± 2.25	8.36 ± 2.29	1.6 ± 0.5		
	FPSGD $\tau = 0.2$	258 ± 88	1.38 ± 0.19	5.87 ± 1.99	7.25 ± 2.05	1.8 ± 0.5		
	FPSGD $\tau = 0.4$	205 ± 79	1.36 ± 0.16	4.67 ± 1.76	6.03 ± 1.78	2.2 ± 0.6		
	FPSGD $\tau = 0.6$	145 ± 63	1.37 ± 0.16	3.32 ± 1.46	4.68 ± 1.46	2.8 ± 0.8		
	FPSGD $\tau = 0.8$	103 ± 47	1.38 ± 0.17	2.36 ± 1.09	3.75 ± 1.11	3.5 ± 1.0		
	FPSGD $\tau = 1.0$	116 ± 103	1.36 ± 0.17	2.64 ± 2.30	4.00 ± 2.33	3.6 ± 1.3		
Resolution 3	FASGD	491 ± 17	1.33 ± 0.18	12.0 ± 0.43	13.3 ± 0.47	-	1.67 ± 1.68	-
	PSGD-H	11 ± 22	9959 ± 7269	22.2 ± 45.3	9981 ± 7265	0.0 ± 0.0	1.59 ± 1.59	0.445
	FPSGD $\tau = 0.0$	398 ± 94	2.52 ± 0.36	9.78 ± 2.35	12.3 ± 2.38	1.1 ± 0.3	1.64 ± 1.65	0.243
	FPSGD $\tau = 0.2$	319 ± 101	2.52 ± 0.38	7.85 ± 2.44	10.4 ± 2.40	1.4 ± 0.3	1.64 ± 1.64	0.295
	FPSGD $\tau = 0.4$	260 ± 95	2.52 ± 0.36	6.42 ± 2.37	8.94 ± 2.31	1.6 ± 0.4	1.62 ± 1.61	0.049
	FPSGD $\tau = 0.6$	215 ± 76	2.53 ± 0.36	5.28 ± 1.86	7.81 ± 1.82	1.8 ± 0.5	1.58 ± 1.56	0.007
	FPSGD $\tau = 0.8$	202 ± 117	2.52 ± 0.37	4.99 ± 2.87	7.51 ± 2.82	2.1 ± 1.1	1.52 ± 1.48	0.004
	FPSGD $\tau = 1.0$	215 ± 184	2.51 ± 0.37	5.25 ± 4.48	7.76 ± 4.39	2.5 ± 1.6	1.48 ± 1.41	0.006

Table 4.4: Overall results for the RIRE brain dataset. We used the MI dissimilarity measure, 3 resolutions, 500 iterations and $\kappa_{\max} = \infty$.

$\mathscr{C}, \boldsymbol{T}$	Optimizer	Iterations I avg ± std	$t_{\text{est}}(s)$ avg ± std	$t_{\text{pure}}(s)$ avg ± std	t_{total} (s) avg ± std	Speed-up avg ± std	ED (mm) avg ± std	p-value
MI, Rigid	FASGD	485 ± 22	0.26 ± 0.02	2.77 ± 0.16	3.02 ± 0.17	-	5.88 ± 3.64	-
	PSGD-J	65 ± 19	0.26 ± 0.02	0.38 ± 0.11	0.65 ± 0.11	4.8 ± 0.8	4.81 ± 2.88	0.004
	FPSGD $\tau = 0.0$	27 ± 14	0.34 ± 0.02	0.16 ± 0.08	0.49 ± 0.08	6.2 ± 0.9	4.02 ± 2.56	0.004
	FPSGD $\tau = 0.2$	27 ± 14	0.35 ± 0.03	0.17 ± 0.08	0.52 ± 0.08	6.0 ± 1.0	3.95 ± 2.75	0.008
	FPSGD $\tau = 0.4$	32 ± 17	0.34 ± 0.02	0.19 ± 0.10	0.54 ± 0.10	5.8 ± 0.9	4.08 ± 2.93	0.004
	FPSGD $\tau = 0.6$	38 ± 20	0.34 ± 0.03	0.23 ± 0.13	0.57 ± 0.14	5.5 ± 1.2	3.69 ± 2.69	0.004
	FPSGD $\tau = 0.8$	50 ± 29	0.34 ± 0.01	0.30 ± 0.17	0.63 ± 0.16	5.0 ± 1.1	4.16 ± 2.96	0.004
	FPSGD $\tau = 1.0$	117 ± 132	0.34 ± 0.03	0.67 ± 0.76	1.01 ± 0.76	3.9 ± 1.5	8.56 ± 12.3	0.129

Table 4.5: Overall results for the BrainWeb dataset. We used the MI dissimilarity measure, 3 resolutions, 1000 iterations and $\kappa_{\max} = 4$. For $\tau = 1.0$, most registrations failed.

$\mathscr{C}, \boldsymbol{T}$	Optimizer	Iterations I avg ± std	$t_{\text{est}}(s)$ avg ± std	$t_{\text{pure}}(s)$ avg ± std	$t_{\text{total}}(s)$ avg ± std	Speed-up avg ± std	Residuals avg ± std	p-value
MI, BSpline	FASGD	996 ± 0	1.72 ± 0.06	49.3 ± 0.90	51.05 ± 0.90	-	2.48 ± 1.43	-
	FPSGD $\tau = 0.0$	289 ± 130	2.29 ± 0.07	13.8 ± 6.17	16.06 ± 6.16	3.6 ± 1.1	2.48 ± 1.42	0.518
	FPSGD $\tau = 0.2$	234 ± 92	2.29 ± 0.07	11.4 ± 4.52	13.66 ± 4.53	4.1 ± 1.2	2.48 ± 1.43	0.921
	FPSGD $\tau = 0.4$	202 ± 62	2.31 ± 0.07	9.89 ± 2.99	12.20 ± 2.99	4.4 ± 1.0	2.48 ± 1.40	0.264
	FPSGD $\tau = 0.6$	211 ± 115	2.36 ± 0.09	11.0 ± 6.42	13.32 ± 6.43	4.2 ± 1.1	2.50 ± 1.39	0.031
	FPSGD $\tau = 0.8$	354 ± 218	2.41 ± 0.05	19.5 ± 12.0	21.95 ± 12.0	3.0 ± 1.4	2.76 ± 1.35	0.000
	FPSGD $\tau = 1.0$	-	-	-	-	-	-	

Table 4.6: The influence of $\kappa_{\max}$ on B-spline registration for the SPREAD study. We used the MI dissimilarity measure, 3 resolutions, 500 iterations, and $\tau = 0.6$.

Optimizer	Resolution 1 Iterations I avg ± std	Resolution 2 Iterations I avg ± std	Resolution 3 Iterations I avg ± std	ED (mm) avg ± std	p-value
FASGD	496 ± 0	496 ± 0	489 ± 14	1.67 ± 1.68	-
FPSGD $\kappa_{\max} = 1$	440 ± 72	426 ± 82	481 ± 44	1.71 ± 1.70	0.001
FPSGD $\kappa_{\max} = 2$	294 ± 85	292 ± 66	378 ± 120	1.66 ± 1.64	0.968
FPSGD $\kappa_{\max} = 4$	180 ± 87	226 ± 50	241 ± 74	1.58 ± 1.56	0.003
FPSGD $\kappa_{\max} = 8$	149 ± 86	224 ± 68	202 ± 73	1.52 ± 1.49	0.000
FPSGD $\kappa_{\max} = 16$	153 ± 89	222 ± 68	200 ± 74	1.51 ± 1.49	0.001

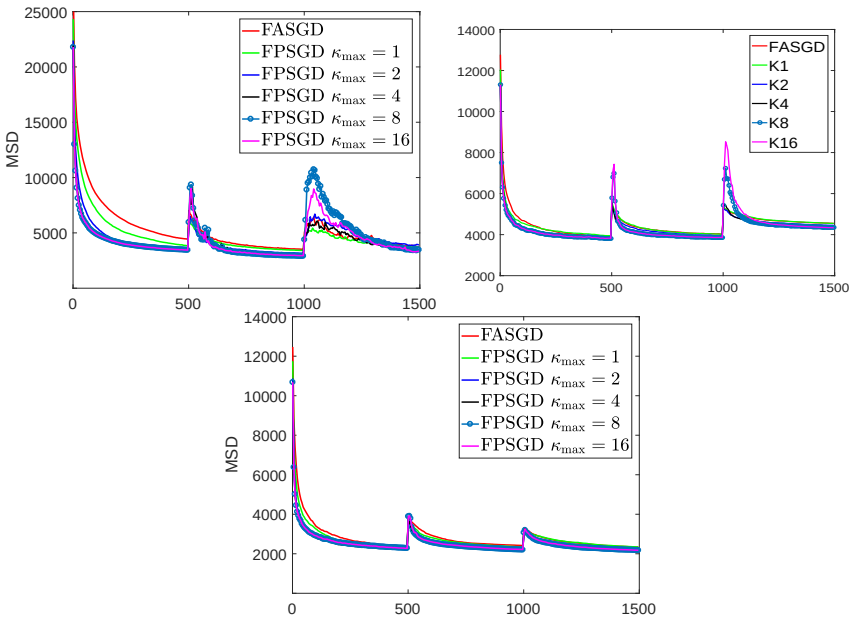

Figure 4.1: Convergence plots for three different patients in the experiments of different $\kappa_{\max}$ for the SPREAD dataset, showing the cost function value (MSD) against the iteration number for different $\kappa_{\max}$ using $\tau = 0.6$.

consistent with the results in [70]. In the remainder of the chapter we set $\kappa_{\max} = 4$ for B-spline registration (and $\kappa_{\max} = \infty$ for rigid and affine registration).

4.6.2 Results of mono-modal image registration

4.6.2.1 Affine registration

The overall results of the experiments on affine registration for the SPREAD lung CT data are given in Table 4.3. It shows that the proposed FPSGD method took less iterations to obtain the same cost function value $\mathscr{C}(\widehat{\boldsymbol{\mu}}_{\text{ref}})$ than FASGD and PSGD-J. The speed-up in terms of number of iterations of FPSGD is about 10. The improvements

Table 4.7: The influence of $\kappa_{\max}$ on B-spline registration for the BrainWeb study. We used the MI dissimilarity measure, 1 resolution, 500 iterations, and $\tau = 0.6$.

Optimizer	Iterations I avg ± std	Speed-up avg ± std	Residuals avg ± std	p-value
FASGD	996 ± 0	1.0 ± 0.0	2.48 ± 1.43	-
FPSGD $\kappa_{\max} = 1$	333 ± 151	3.5 ± 1.3	2.48 ± 1.43	0.561
FPSGD $\kappa_{\max} = 2$	230 ± 85	4.8 ± 1.5	2.49 ± 1.42	0.080
FPSGD $\kappa_{\max} = 4$	211 ± 115	5.4 ± 1.6	2.50 ± 1.39	0.031
FPSGD $\kappa_{\max} = 8$	208 ± 118	5.6 ± 1.9	2.50 ± 1.37	0.017
FPSGD $\kappa_{\max} = 16$	220 ± 128	5.3 ± 1.8	2.53 ± 1.37	0.001

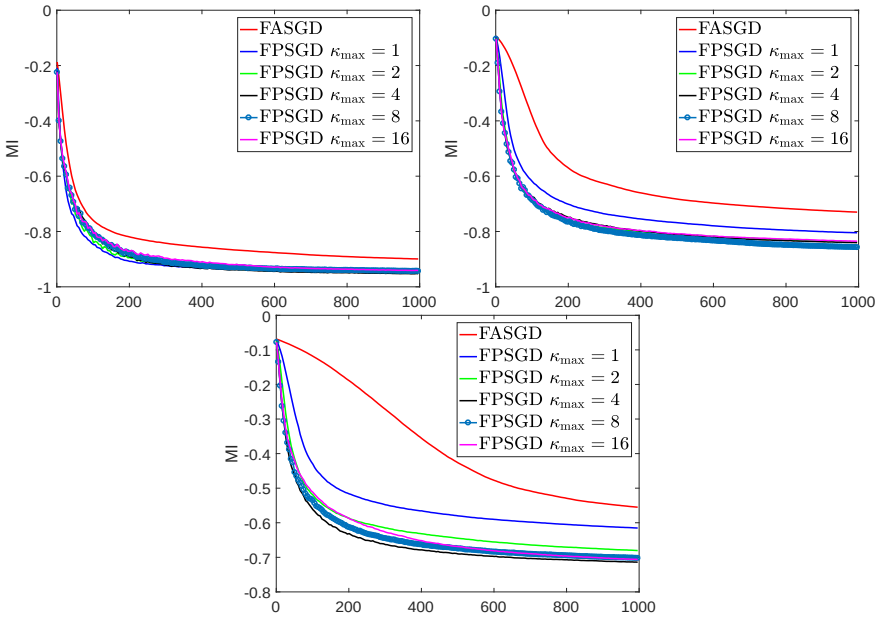

Figure 4.2: Convergence plots for three different patients for the BrainWeb dataset, showing the cost function value (negative MI dissimilarity measure) against the iteration number, using $\tau = 0.6$.

of FPSGD compared to FASGD and PSGD-J in the convergence rate are also shown in Figure 4.5a and Figure 4.5b. These methods have the same runtime per iteration (~3.5 ms). PSGD-H required less iterations than the proposed FPSGD method. The computation of the preconditioner however took somewhat longer, resulting in an overall decrease in performance. For the affine consistently use transformation the runtime per iteration is similar for PSGD-H and FPSGD (~2 ms and ~1 ms, respectively). The overall speed-up in terms of runtime is about 5 for FPSGD, compared to 0.5 for PSGD-H.

It can be seen from Table 4.2 that the Euclidean distance error of all methods is around 5 mm. The p-value of the Wilcoxon signed-rank test of PSGD-J and PSGD-H

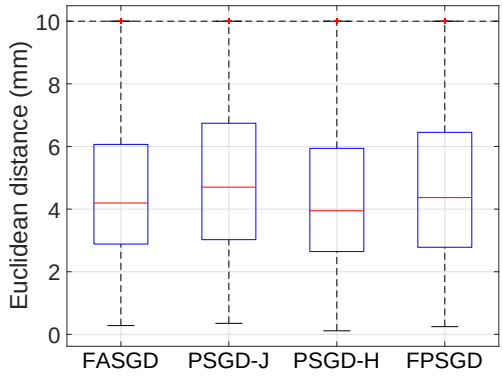

Figure 4.3: Euclidean distance error in mm for the different methods with the SPREAD lung CT data. The experiments were performed using MSD dissimilarity measure and affine transformation model. For FPSGD, $\tau = 0.6$ and $\kappa_{\max} = \infty$ are used.

compared to FASGD is smaller than 0.05, indicating a statistically significant difference. Although significant, the differences are very small, i.e. less than 0.5 mm. The Wilcoxon signed-rank tests of FPSGD (all settings of τ) compared to FASGD show no statistical difference ($p > 0.05$). A boxplot of the Euclidean distance error of 100 corresponding points is given in Figure 4.3, using $\tau = 0.6$ for FPSGD.

4.6.2.2 B-spline registration

The overall results of the experiments on B-spline registration for the SPREAD lung CT data are given in Table 4.3. For all three resolutions, the proposed method took less iterations to obtain the same cost function value as FASGD. Although the proposed method took somewhat longer to estimate the preconditioner compared to FASGD, less iterations were required, resulting in an overall improvement of runtime. For FPSGD ($\tau = 0.6$), the overall speed-up is of a factor of 2. The number of iterations used for PSGD-H to obtain the same cost function value is less than both FASGD and FPSGD, which can also be observed from the convergence plots in Figure 4.5c and Figure 4.5d. However, the overhead of computing the preconditioner increased substantially for the PSGD-H method: around 10^4 seconds for $\sim 10^5$ parameters in resolution 3, while the FPSGD method required $\sim$2s.

The ED errors in Table 4.3 are evaluated at the end of resolution 3. All three methods FASGD, PSGD-H and FPSGD obtained a mean ED error around 1.65 mm, which is within one voxel. The p-value of the Wilcoxon signed-rank test of PSGD-H and FPSGD compared to FASGD is 0.445 and 0.968, respectively, indicating no statistical difference. Figure 4.4 presents the boxplot of the Euclidean distance error for the different methods, where for FPSGD we used $\tau = 0.6$ and $\kappa_{\max} = 4$.

4.6.3 Results of multi-modal image registration

4.6.3.1 RIRE brain data

Table 4.4 presents the runtime differences and the mean Euclidean distance error of the RIRE experiments for all methods. We can observe that much less iterations are required for PSGD-J and FPSGD compared to FASGD. The speed-up in iterations is

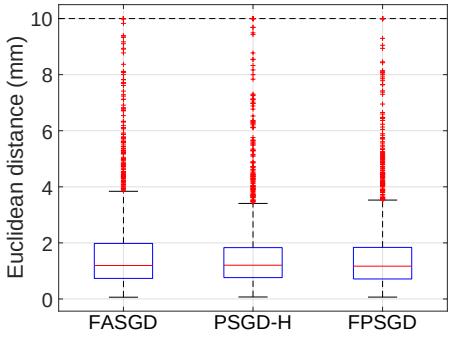

Figure 4.4: Euclidean distance error in mm for the different methods with the SPREAD lung CT data. The experiments were performed using MSD dissimilarity measure and B-spline transformation model. For FPSGD, $\tau = 0.6$ and $\kappa_{\max} = 4$ are used.

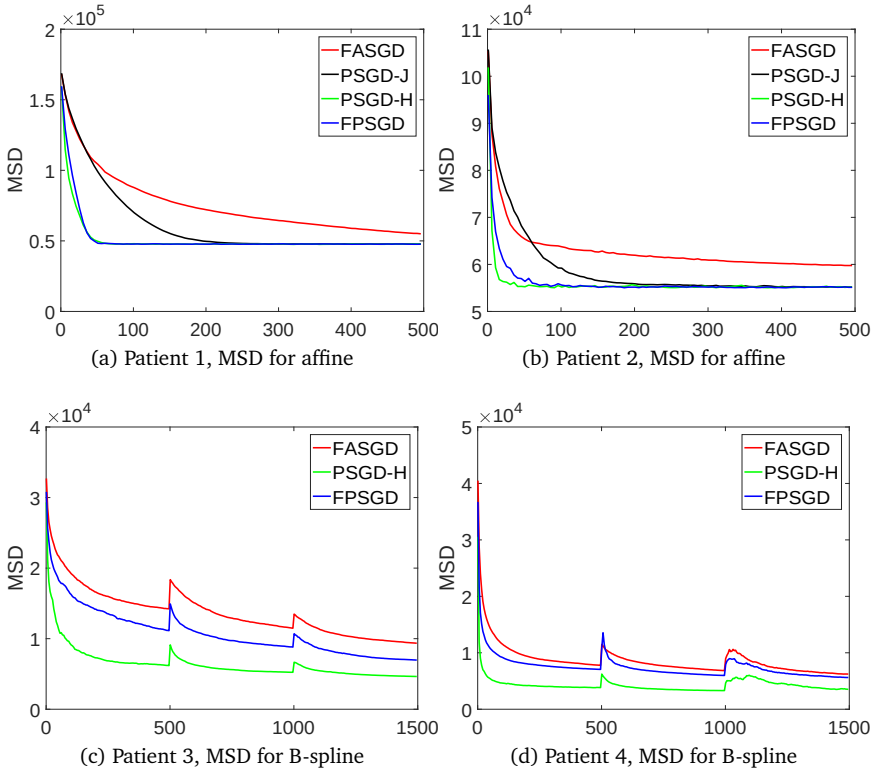

Figure 4.5: Convergence plots of four cases in the experiments of the SPREAD lung CT data, showing the cost function value (MSD) against the iteration number. For B-spline registration of FPSGD, $\tau = 0.6$ and $\kappa_{\max} = 4$ are used.

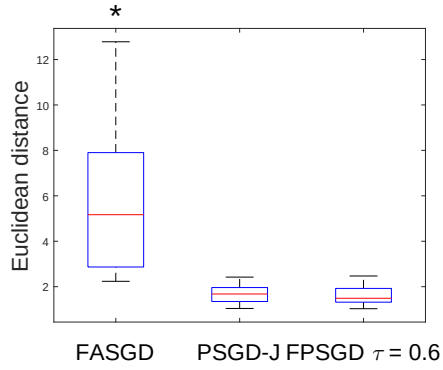

Figure 4.6: Euclidean distance error in mm for different methods with the RIRE brain data. The experiments were performed using MI dissimilarity measure and rigid transformation model. For FPSGD, $\tau = 0.6$ and $\kappa_{\max} = \infty$ are used.

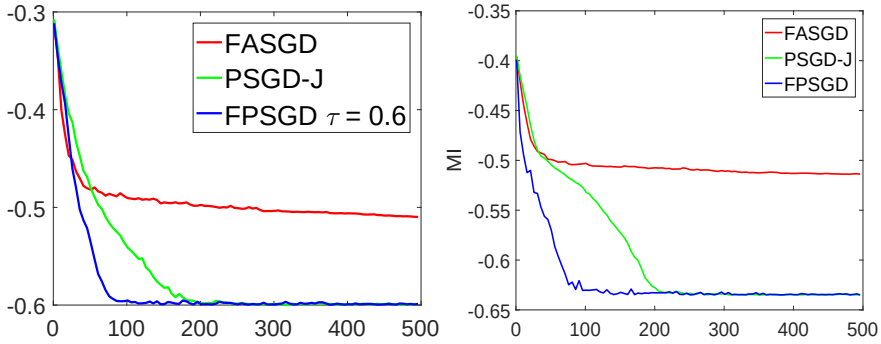

Figure 4.7: Convergence plots of two patients of the RIRE brain dataset, showing the cost function value (negative mutual information measure) against the iteration number. For FPSGD, $\tau = 0.6$ and $\kappa_{\max} = \infty$ are used.

a factor of 10. It can also be seen that the speedup in runtime is around 5 for the FPSGD method. The convergence plots in Figure 4.7 show substantial improvement in convergence rate for FPSGD.

The boxplots of the Euclidean distance error for the RIRE data are shown in Figure 4.6. The median Euclidean distance of 9 patients before registration is 21.7 mm. As we can see, the FASGD method that manually chooses a scaling factor is inferior to the other two methods. From Table 4.4, it can be seen that the Wilcoxon signed-rank tests between FASGD and FPSGD with different τ show significant statistical differences ($p < 0.05$), except for $\tau = 1.0$.

4.6.3.2 BrainWeb simulated brain data

The results of the BrainWeb experiment are shown in Table 4.5, Figure 4.8 and Figure 4.9. The number of iterations for FPSGD ($\tau = 0.6$) to obtain the same cost function value (MI) as FASGD is around 200, resulting in a runtime speed-up of about a factor of 5, as can be seen in Table 4.5. These improvements can also be observed from the

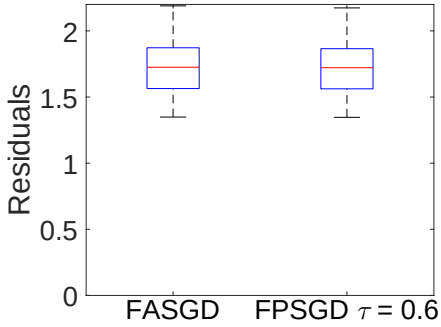

Figure 4.8: Residuals in mm for the different methods with the BrainWeb simulated brain data. The experiments were performed using MI dissimilarity measure and B-spline transformation model. The parameter settings of FPSGD are $\tau = 0.6$ and $\kappa_{\max} = 4$.

convergence plots in Figure 4.9.

The mean residuals of the different methods show a similar result. The Wilcoxon signed-rank test between FASGD and FPSGD ($\tau = 0.6$) shows a significant statistical difference ($p = 0.031$). However, from Table 4.5, it can be seen that the difference is very small (around 0.02). Increasing the regularization factors τ can achieve a faster convergence rate, however, most registrations failed for $\tau = 1.0$. The boxplots of the residuals of both FASGD and FPSGD ($\tau = 0.6$, $\kappa_{\max} = 4$) are shown in Figure 4.8.

4.7 Discussion

The experimental results show that the proposed FPSGD method works well in both mono-modal as well as multi-modal image registration, in combination with different transformation models and dissimilarity measures, showing that the proposed method is generic for different registration problems. The proposed FPSGD method can be used for different transformation models, unlike PSGD-J which was proposed only for rigid and affine registration problems. Compared to FASGD which is not preconditioned, the proposed FPSGD method not only obtains the same registration accuracy, moreover improves the convergence. Without the computational burden of the Hessian matrix calculation and decomposition, the proposed FPSGD method takes much less time than PSGD-H to construct a preconditioner. Additionally, the proposed method requires only a cost function gradient and a set of transformation Jacobians, while PSGD-H also needs the implementation of the self-Hessian. Most importantly, the proposed FPSGD method is more generic for different modalities and not limited to mono-modal problems like PSGD-H.

Compared to FASGD, the main improvement of the proposed FPSGD method is in the convergence rate, inducing a speedup in runtime of a factor of 2.0-6.0 depending on the application. Specifically, the proposed FPSGD method used half a second to obtain the same registration accuracy as FASGD for the affine registration on the SPREAD lung CT with image sizes of $450 \times 300 \times 130$, while FASGD took 2 seconds. The proposed FPSGD method needs much less computation time for the preconditioner estimation than PSGD-H: ~2 seconds vs ~10^4 seconds for ~10^5 transformation parameters, see

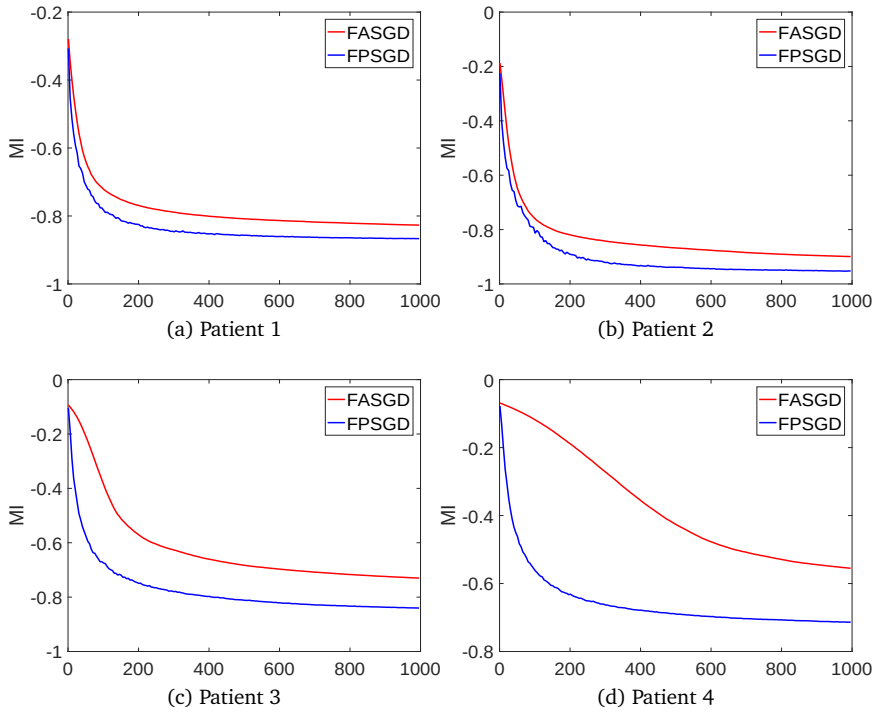

Figure 4.9: Convergence plots of the experiment of the BrainWeb simulated dataset, showing the cost function value (negative MI dissimilarity measure) against the iteration number. For FPSGD, $\tau = 0.6$ and $\kappa_{\max} = 4$ are used.

Table 4.3. This large difference between different methods in the computation time of preconditioner estimation can be attributed to the complexity of different methods. For PSGD-H, the complexity is highly due to the Cholesky decomposition of $\mathcal{O}(N_p^3)$, i.e. depending on the number of transformation parameters, while for the FPSGD method the complexity is only linear in the number of samples $\mathcal{O}(N_p)$. In addition, the runtime per iteration for the PSGD-H method increased to ~5 seconds for $N_P \approx 10^5$ transformation parameters, due to the multiplication of a full matrix $\boldsymbol{P}$ instead of only a diagonal matrix for FPSGD (~24 ms per iteration for MI dissimilarity measurement). We therefore conclude that the proposed FPSGD method converges faster than the FASGD method and is more time-efficient than the PSGD-H method.

There are two parameters that influence the performance of the proposed FPSGD method: the regularization factor τ and the maximum condition number $\kappa_{\max}$. We validated the influence of both parameters experimentally. We showed that the extreme cases ($\tau = 0$ and $\tau = 1$) yielded suboptimal results, indicating that regularization of the preconditioner is required. The proposed regularization method performs a Gaussian smoothing, considering entries with a similar Jacobian response. This choice reflects the observation that transformation parameters that have a similar effect on the displacement, require similar preconditioning, and vice versa. For example,

for the affine transformation rotation and translation require different scaling. The experiments showed that the choice $\tau = 0.6$ yielded good results for all applications.

For ill-conditioned problems, $\kappa_{\max}$ serves as a safe guard to prevent extreme values in the preconditioner. In the experiment on the SPREAD data, different $\kappa_{\max}$ obtained a similar registration accuracy, however, the convergence has some oscillations for $\kappa_{\max} > 4$ in the second and third resolution in Figure 4.1. For the BrainWeb data, best results were acquired with $\kappa_{\max} = 4$ and the convergence plots are also very stable. Overall, the best choice of $\kappa_{\max}$ is between 2 and 4 for nonrigid registration, while $\kappa_{\max} = \infty$ can be used for rigid and affine registration.

To further improve the proposed FPSGD method the following may be considered. Firstly, the proposed preconditioning scheme detailed in Algorithm 2 is very suitable for further acceleration on a Graphics Processing Unit (GPU). It could be easily applied for the parallel computing of the gradient and the preconditioner [22], therefore this will be beneficial when going to variable preconditioning. Secondly, our method can be combined with the variable preconditioning techniques for difficult problems where the curvature of the cost function changes iteratively. Instead of estimating the preconditioner once at the beginning of each resolution, we may regularly update it. A GPU implementation is then warranted to keep the runtime per iteration low. Furthermore, a stopping condition other than the number of iterations will be required to practically take advantage of the convergence improvements. An interesting option suitable in a stochastic setting is a moving average of the noisy gradients over a few iterations.

4.8 Conclusion

In this chapter, we proposed a generic preconditioner estimation method for the stochastic gradient descent optimizers used in medical image registration. Based on the observed distribution of the voxel displacements, this method automatically constructs a diagonal preconditioner, avoiding the computationally complex calculation of the Hessian matrix. All tested methods obtained a similar final registration accuracy in all tested datasets. The proposed FPSGD optimizer, however, outperforms FASGD and PSGD-J in terms of convergence rate, while yielding a similar computational overhead. While a previous method (PSGD-H) even further reduces the required number of iterations, it comes at a substantial overhead in computing the preconditioner, especially for high dimensional transformations. Additionally, PSGD-H can only be used in mono-modal problems and requires the implementation of a Hessian matrix computation.

We conclude that the proposed method can act as a generic preconditioner for optimization in registration methods, yielding similar accuracy as gradient descent routines while substantially improving the convergence rate.

Acknowledgments

This research was supported by ZonMw, the Netherlands Organization for Health Research and Development [grant number 104003012]; and the China Scholarship Council [grant number 201206130066]. The RIRE project is acknowledged for providing a platform for rigid registration evaluation. Dr. M.E. Bakker and J. Stolk are acknowledged for providing a ground truth for the SPREAD study data used in this chapter.