

**Scaled projected-directions methods with application to transmission tomography**

G. Guillaume, Y. Goussard,  
D. Orban

G-2019-60

August 2019

La collection *Les Cahiers du GERAD* est constituée des travaux de recherche menés par nos membres. La plupart de ces documents de travail a été soumis à des revues avec comité de révision. Lorsqu'un document est accepté et publié, le pdf original est retiré si c'est nécessaire et un lien vers l'article publié est ajouté.

**Citation suggérée :** G. Guillaume, Y. Goussard, D. Orban (Août 2019). Scaled projected-directions methods with application to transmission tomography, Rapport technique, Les Cahiers du GERAD G-2019-60, GERAD, HEC Montréal, Canada.

**Avant de citer ce rapport technique**, veuillez visiter notre site Web (<https://www.gerad.ca/fr/papers/G-2019-60>) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

The series *Les Cahiers du GERAD* consists of working papers carried out by our members. Most of these pre-prints have been submitted to peer-reviewed journals. When accepted and published, if necessary, the original pdf is removed and a link to the published article is added.

**Suggested citation:** G. Guillaume, Y. Goussard, D. Orban (August 2019). Scaled projected-directions methods with application to transmission tomography, Technical report, Les Cahiers du GERAD G-2019-60, GERAD, HEC Montréal, Canada.

**Before citing this technical report**, please visit our website (<https://www.gerad.ca/en/papers/G-2019-60>) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2019  
– Bibliothèque et Archives Canada, 2019

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2019  
– Library and Archives Canada, 2019

GERAD HEC Montréal  
3000, chemin de la Côte-Sainte-Catherine  
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053  
Télec. : 514 340-5665  
info@gerad.ca  
www.gerad.ca



# Scaled projected-directions methods with application to transmission tomography

Guillaume Mestdagh<sup>a</sup>

Yves Goussard<sup>a</sup>

Dominique Orban<sup>b</sup>

<sup>a</sup> *Institute of Biomedical Engineering, Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

<sup>b</sup> *GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

guillaume.mestdagh@polymtl.ca

yves.goussard@polymtl.ca

dominique.orban@polymtl.ca

August 2019

Les Cahiers du GERAD

G–2019–60

Copyright © 2019 GERAD, Guillaume, Goussard, Orban

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs. Les auteurs conservent leur droit d'auteur et leurs droits moraux sur leurs publications et les utilisateurs s'engagent à reconnaître et respecter les exigences légales associées à ces droits. Ainsi, les utilisateurs:

- Peuvent télécharger et imprimer une copie de toute publication du portail public aux fins d'étude ou de recherche privée;
- Ne peuvent pas distribuer le matériel ou l'utiliser pour une activité à but lucratif ou pour un gain commercial;
- Peuvent distribuer gratuitement l'URL identifiant la publication.

Si vous pensez que ce document enfreint le droit d'auteur, contactez-nous en fournissant des détails. Nous supprimerons immédiatement l'accès au travail et enquêterons sur votre demande.

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*. Copyright and moral rights for the publications are retained by the authors and the users must commit themselves to recognize and abide the legal requirements associated with these rights. Thus, users:

- May download and print one copy of any publication from the public portal for the purpose of private study or research;
- May not further distribute the material or use it for any profit-making activity or commercial gain;
- May freely distribute the URL identifying the publication.

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

**Abstract:** Statistical image reconstruction in X-Ray computed tomography yields large-scale regularized linear least-squares problems with nonnegativity bounds, where the memory footprint of the operator is a concern. Discretizing images in cylindrical coordinates results in significant memory savings but deteriorates the conditioning of the operator. We improve the Hessian conditioning by way of a block-circulant scaling operator and we propose a strategy to handle nondiagonal scaling in the context of projected-directions methods for bound-constrained problems. We describe our implementation of the scaling strategy using two algorithms: TRON, a trust-region method with exact second derivatives, and L-BFGS-B, a linesearch method with a limited-memory quasi-Newton Hessian approximation. We compare our approach with one where a change of variable is made in the problem. On two reconstruction problems, our approach converges faster than the change of variable approach, and achieves much tighter accuracy than a first-order method.

**Keywords:** X-Ray CT reconstruction, projected-directions methods, scaling

---

**Acknowledgments:** The work presented in this paper was funded by the Natural Sciences and Engineering Research Council of Canada.

# 1 Introduction

We consider the bound-constrained problem

$$\min f(\mathbf{x}) \quad \text{s.t. } \mathbf{x} \geq 0, \quad (1)$$

where  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is convex and  $\mathcal{C}^2$ . We assume that  $\nabla^2 f$  cannot be stored explicitly or in factorized form. We are particularly interested in the case where (1) is large and badly scaled. Our main motivation is to solve efficiently statistical image reconstruction problems arising from X-Ray Computed Tomography (CT) (Herman, 2009). Whereas cartesian coordinates are typical, discretizing such problems in cylindrical coordinates yields large savings in storage, but results in badly scaled problems and, without proper scaling, off-the-shelf solvers usually fail.

In this paper we employ a scaling strategy that exploits the structure of  $f$  combined with the trust-region projected Newton method of Lin and Moré (1999) and with the line search limited-memory BFGS for bound-constrained problems of Byrd, Lu, Nocedal, and Zhu (1995) to maintain satisfaction of the bound constraints at all times.

## Motivation and previous work

Our main interest resides in statistical image reconstruction problems arising from X-Ray Computed Tomography (CT) (Herman, 2009). Compared to analytical methods such of the filtered backprojection family (Feldkamp, Davis, and Kress, 1984), statistical reconstruction results in less noisy images but is more computationally expensive (Fessler, 2000).

Sauer and Bouman (1993) show that an image  $\mathbf{x}$  can be estimated from the measurements  $\mathbf{b}$  by solving

$$\min \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_{\mathbf{V}}^2 + \lambda R(\mathbf{x}) \quad \text{s.t. } \mathbf{x} \geq 0, \quad (2)$$

where  $\mathbf{A}$  is a large sparse matrix,  $\lambda > 0$  is a regularization parameter,  $R : \mathbb{R}^n \rightarrow \mathbb{R}$  is a convex regularization function and  $\mathbf{V}$  is a diagonal weight matrix with  $V_{ii} = \exp(-b_i)$ . In (2), the objective is composed of a least-squares data-fitting term, and a regularization term to discourage large differences between adjacent pixels. Here, we focus on situations where one wishes to solve problem (2) precisely (with a low tolerance level), which corresponds to instances where the information present in the data should be fully exploited, as, e.g., in low-dose CT reconstruction or for imaging thin structures in micro-CT (Hamelin, Goussard, Dussault, Cloutier, Beaudoin, and Soulez, 2010).

While reconstructed images are usually discretized using a cartesian grid of voxels (Fessler, 2000, Equation (6)), we use cylindrical coordinates (Thibaut, Leroux, Fontaine, and Lecomte, 2013). This discretization is in adequation with the circular geometry of the data acquisition process and results in block-circulant  $\mathbf{A}$ , which has far lower storage requirements than the operator resulting from cartesian coordinates (Goussard, Golkar, Wagner, and Voorons, 2013).

In statistical image reconstruction, first-order methods are often preferred due to their low storage and computational demands, and their simplicity (Fessler, 2000). Usual reconstruction methods include the expectation-maximization algorithm (Lange and Fessler, 1995; Ahn, Fessler, Blatt, and Hero, 2006), coordinate-descent methods (Sauer and Bouman, 1993; Noo, Hahn, Schöndube, and Stierstorfer, 2016) and gradient-based methods (Erdoğan and Fessler, 1999a; Kim, Ramani, and Fessler, 2014). This last category is probably the most studied in image reconstruction. Such methods are often improved by using ordered subsets (Erdoğan and Fessler, 1999b; Hudson and Larkin, 1994) or Nesterov momentum (Nesterov, 1983; Kim et al., 2014; Xu, Yang, Tan, Sawatzky, and Anastasio, 2016; Choi, Wang, Zhu, Suh, Boyd, and Xing, 2010; Jensen, Jørgensen, Hansen, and Jensen, 2012). Recently, problem-splitting and proximal approaches have been proposed in the context of sparse reconstruction (Sidky, Jørgensen, and Pan, 2013; Nien and Fessler, 2015; Xu et al., 2016).

For imaging applications, diagonal scaling is usually sufficient for first-order methods to perform well (Pock and Chambolle, 2011; Bonettini, Zanella, and Zanni, 2008). In CT reconstruction, diagonal

scaling is sometimes referred to as the *separable quadratic surrogate method* (Erdoğan and Fessler, 1999b). However, in cylindrical coordinates, widely different voxel sizes make  $\mathbf{A}$  badly scaled, and diagonal scaling is no longer appropriate.

To improve the conditioning of (2), we follow Golkar (2013) and use a block-circulant scaling operator  $\mathbf{C}$  that exploits the block-circulant property of  $\mathbf{A}$  and of the finite-difference matrices that appear in the regularization term  $R(\mathbf{x})$ . The scaling operator can be written

$$\mathbf{C} = \mathbf{F}^* \mathbf{T} \mathbf{F},$$

where  $\mathbf{T}$  is diagonal,  $\mathbf{F}$  is a discrete block Fourier transform, and a star indicates the conjugate transpose. Thus,  $\mathbf{C}$  and its inverse can be applied to a vector at the cost of a fast Fourier transform, namely in  $O(s_b n_b \log n_b)$  operations, where  $s_b$  is the size of a square block, and  $n_b$  is the number of blocks.

If  $\mathbf{C}$  satisfies  $\mathbf{C}^T \nabla^2 f(\mathbf{x}) \mathbf{C} \approx \mathbf{I}$ , the change of variable  $\mathbf{x} = \mathbf{C} \mathbf{u}$  transforms (1) into the scaled problem

$$\min \frac{1}{2} \|\mathbf{A} \mathbf{C} \mathbf{u} - \mathbf{b}\|_{\mathbf{V}}^2 + \lambda R(\mathbf{C} \mathbf{u}) \quad \text{s.t. } \mathbf{C} \mathbf{u} \geq 0, \quad (3)$$

in which the objective Hessian is better conditioned than in (2). However, (3) features linear inequality constraints instead of simple bounds.

McLaughlin (2017) solves (3) with **Cflash**, a variant of TRON (Lin and Moré, 1999) adapted to linear inequality constraints. Each iteration of **Cflash** requires projecting candidate iterates  $\mathbf{u}$  into the feasible set by solving

$$\min_{\mathbf{v}} \|\mathbf{v} - \mathbf{u}\| \quad \text{s.t. } \mathbf{C} \mathbf{v} \geq 0, \quad (4)$$

which represents a significant amount of computation. In **Cflash**, the above projection is computed efficiently by solving the dual problem, which is a bound-constrained linear least-squares problem with operator  $\mathbf{C}$ , iteratively. Even though (4) can be solved efficiently thanks to the structure of  $\mathbf{C}$ , it remains substantially more costly than projecting into simple bounds.

Instead of solving (3), we propose to solve (2) with a scaled quasi-Newton and Newton method in order to reproduce the effect of a change of variable without actually performing it. An advantage of scaled directions is that we can choose the scaling at each iteration.

Bonettini et al. (2008) describe diagonally-scaled projection methods in which both the gradient and projection subproblems are scaled. Bonettini, Landi, Piccolomini, and Zanni (2013) extend the approach to block-diagonal scaling in the context of image deblurring, a problem where the system operator is a 2D convolution, which is block-circulant with circulant blocks. Their scaling depends on the active constraints at the current iterate, as in the projected Newton method described by Bertsekas (1982), which allows them to apply a nondiagonal scaling while preserving simple projections. They also investigate a quasi-Newton method where the Hessian approximation is a truncated spectral decomposition of the system matrix. In the partially-diagonal scaling approach, Bonettini et al. apply the method of Landi and Loli Piccolomini (2008) and solve the linear system inexactly using the conjugate gradient method (CG) of Hestenes and Stiefel (1952). Both methods are presented as scaled gradient projection methods in which the scaling operator is inspired from the problem Hessian or from a quasi-Newton matrix.

Our approach extends that of Bonettini et al. (2008) to more complex methods for bound-constrained problems. Because we apply the scaling to higher-order methods, we restrict our study to the situation where the scaling operator can be applied or inverted easily. The problem Hessian does not enter this category, because it is expensive to apply and we need to use CG to solve a linear system involving it. Unlike Bonettini et al., we consider the use of partially-diagonal operators in the context of a change of scalar product, independently from the solution method. We then describe the impact of the scaling on families of general-purpose optimization algorithms. Specifically, we consider the limited-memory

BFGS method for bound-constrained problems, and a trust-region projected Newton method. The convergence properties of both methods rest on the computation of a Cauchy point, i.e., an approximate minimizer in the negative gradient direction. Our implementation of L-BFGS-B differs from that of Zhu, Byrd, Lu, and Nocedal (1997) in that we compute an inexact Cauchy point and restrict the computation of a step to an active face by way of restriction operators. Our implementation of a projected Newton method follows the design of TRON (Lin and Moré, 1999), in which projected gradient steps are performed to identify an inexact Cauchy point and a candidate active set, followed by a sequence of Newton steps on the active face of the feasible set globalized by a trust-region mechanism. Such an approach has been shown to be able to add and remove numerous bounds from the active set at a time and to be particularly appropriate for large-scale problems. We illustrate the performance of both methods on synthetic images, and we compare it to that of the projected spectral gradient method, a classic method in imaging.

**Notation** Lowercase and uppercase bold Latin letters represent vectors and matrices, respectively. Light face Latin letters represent integers, such as iteration counters, and functions. In addition, the  $i$ -th component of  $\mathbf{x}$  is denoted  $x_i$  and the  $(i, j)$ -th element of  $\mathbf{A}$  is  $A_{ij}$ . Light face Greek letters represent scalars. The Euclidean scalar product on  $\mathbb{R}^n$  is denoted  $\langle \cdot, \cdot \rangle$ . The  $i$ -th partial derivative of function  $f$  at  $x$  is denoted  $\partial_i f(x)$ .

## 2 A scaling strategy for bound-constrained problems

In this section we describe the effect of scaling on procedures that are common between the two methods we present. After a short presentation of the scaling strategy, we present two procedures we use in L-BFGS-B and TRON to compute a Cauchy point and a descent step in the active face respectively. We then apply the scaling on the limited-memory quasi-Newton matrix that appears in L-BFGS-B. We integrate these procedures into the chosen algorithms in [Section 3](#).

### 2.1 Overview of the strategy

Our scaling strategy consists in using a metric in which the problem is well scaled and the projections can be computed with a direct formula.

A linear change of variables is equivalent to changing the scalar product in the original space. Indeed, if  $\mathbf{x} = \mathbf{C}\mathbf{u}$  and  $\mathbf{z} = \mathbf{C}\mathbf{w}$ ,

$$\langle \mathbf{u}, \mathbf{w} \rangle = \langle \mathbf{x}, \mathbf{P}^{-1} \mathbf{z} \rangle, \quad \mathbf{P} := \mathbf{C}\mathbf{C}^T.$$

This equivalence makes it possible to import geometric elements from the scaled space into the original space. From now on, we use  $\mathcal{X}$  to denote the original space and  $\mathcal{U}$  for the scaled space. Every  $\mathbf{x} \in \mathcal{X}$  corresponds to a  $\mathbf{u} \in \mathcal{U}$  such that  $\mathbf{x} = \mathbf{C}\mathbf{u}$ . We denote  $\bar{f}$  the objective function of (1) in the scaled space, i.e., for  $\mathbf{u} \in \mathcal{U}$ ,

$$\bar{f}(\mathbf{u}) := f(\mathbf{C}\mathbf{u}), \quad \nabla \bar{f}(\mathbf{u}) = \mathbf{C}^T \nabla f(\mathbf{C}\mathbf{u}), \quad \nabla^2 \bar{f} = \mathbf{C}^T \nabla^2 f(\mathbf{C}\mathbf{u}) \mathbf{C}.$$

The first element we transform is the gradient direction. Indeed, due to the choice of scaling, the gradient direction in  $\mathcal{U}$  is expected to be a more promising descent direction than the gradient direction in  $\mathcal{X}$ . If  $\mathbf{u} \in \mathcal{U}$ ,  $\mathbf{x} = \mathbf{C}\mathbf{u}$ , and  $\alpha > 0$ ,

$$\mathbf{x}(\alpha) := \mathbf{C}(\mathbf{u} - \alpha \nabla \bar{f}(\mathbf{u})) = \mathbf{x} - \alpha \mathbf{C} \nabla \bar{f}(\mathbf{u}) = \mathbf{x} - \alpha \mathbf{P} \nabla f(\mathbf{x}).$$

In other words, a step along the negative gradient in  $\mathcal{U}$  is equivalent to a step in the direction

$$\mathbf{q} = -\mathbf{P} \nabla f(\mathbf{x}) \tag{5}$$

in  $\mathcal{X}$ . We use (5) instead of  $-\nabla f(\mathbf{x})$  in the hope that the scaled search direction better captures natural problem curvature.

## 2.2 Projected directions

In the context of projected methods, it is often necessary to work on a face of the feasible set, or to take projected gradient steps.

### 2.2.1 Projected gradient steps

Because the method of Lin and Moré (1999) is related to that of Bertsekas (1982), it is useful to review certain common basic concepts.

A standard projected gradient step from  $\mathbf{x}$  can be described by  $\text{Proj}(\mathbf{x} - \alpha \nabla f(\mathbf{x})) - \mathbf{x}$  where  $\alpha > 0$ . A scaled projected gradient step has the form  $\mathbf{x}(\alpha) - \mathbf{x}$ , where

$$\mathbf{x}(\alpha) = \text{Proj}(\mathbf{x} + \alpha \mathbf{d}),$$

and where  $\mathbf{d}$  is a linear transformation of  $\nabla f(\mathbf{x})$ . The direction  $\mathbf{d}$  must be a descent direction in the sense that there exists  $\bar{\alpha} > 0$  such that  $f(\mathbf{x}(\alpha)) < f(\mathbf{x})$  for all  $\alpha \in (0, \bar{\alpha}]$ .

Bertsekas (1982, section 2) explains that (5) might not be such a descent direction. We define the *binding* constraints at  $\mathbf{x}$  as those with indices in

$$I_+(\mathbf{x}) = \{i \mid x_i = 0 \text{ and } \partial_i f(\mathbf{x}) > 0\},$$

where  $\partial_i f(\mathbf{x})$  is the  $i$ -th component of  $\nabla f(\mathbf{x})$ . We introduce the subspace

$$F = \{\mathbf{a} \in \mathbb{R}^n \mid a_i = 0 \text{ for all } i \in I_+(\mathbf{x})\},$$

and the set  $F_+ = F \cap \mathbb{R}_+^n$ , called the face of the feasible set exposed by  $-\nabla f(\mathbf{x})$ . Bertsekas (1982, Proposition 1) establishes that

$$\mathbf{d} = -\bar{\mathbf{P}} \nabla f(\mathbf{x}), \tag{6}$$

where the matrix  $\bar{\mathbf{P}}$  is defined by

$$\bar{\mathbf{P}}_{ij} = \begin{cases} \mathbf{P}_{ij} & \text{if } i, j \notin I_+(\mathbf{x}) \\ 0 & \text{otherwise,} \end{cases}$$

is a descent direction in the sense defined above.

To compute (6), we decompose  $\mathbb{R}^n = F \oplus G$  where  $G = F^\perp$ , and we write

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \mathbf{g}_F \\ \mathbf{g}_G \end{pmatrix} \quad \mathbf{P} = \begin{pmatrix} \mathbf{P}_{FF} & \mathbf{P}_{FG} \\ \mathbf{P}_{GF} & \mathbf{P}_{GG} \end{pmatrix},$$

where  $\mathbf{g}_F = (\partial_i f(\mathbf{x}))_{i \notin I_+(\mathbf{x})}$  and  $\mathbf{g}_G = (\partial_i f(\mathbf{x}))_{i \in I_+(\mathbf{x})}$ .

The gradient is first projected onto the subset  $F$ . Then the scaling is made on the projected gradient  $\mathbf{g}_F$  by applying the principal submatrix  $\mathbf{P}_{FF}$ . This submatrix is obtained by keeping only the rows and columns whose indices are not in  $I_+(\mathbf{x})$ . Finally,

$$\mathbf{d} = - \begin{pmatrix} \mathbf{P}_{FF} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{g}_F \\ \mathbf{g}_G \end{pmatrix} = \begin{pmatrix} -\mathbf{P}_{FF} \mathbf{g}_F \\ 0 \end{pmatrix}.$$

### 2.2.2 Conjugate gradient in a face of the feasible set

The same procedure can be used to modify conjugate gradient directions inside a face of the feasible set. Consider the quadratic problem

$$\min_{\mathbf{x} \in F} \frac{1}{2} \mathbf{x}^T \mathbf{B} \mathbf{x} + \mathbf{x}^T \mathbf{g}, \tag{7}$$

To solve (7), the Conjugate Gradient method of Hestenes and Stiefel (1952) is applied to the equivalent reduced problem

$$\min_{\bar{\mathbf{x}} \in \mathbb{R}^{\dim F}} \frac{1}{2} \bar{\mathbf{x}}^T \mathbf{B}_{FF} \bar{\mathbf{x}} + \bar{\mathbf{x}}^T \bar{\mathbf{g}}.$$

The directions  $\bar{\mathbf{p}}_1, \bar{\mathbf{p}}_2, \dots$  generated by the procedure are conjugate with respect to the principal submatrix  $\mathbf{B}_{FF}$  of  $\mathbf{B}$  (Hestenes and Stiefel, 1952). In particular, at the  $k$ -th iteration, the next direction is defined as

$$\bar{\mathbf{p}}_{k+1} = \bar{\mathbf{r}}_{k+1} + \beta_k \bar{\mathbf{p}}_k,$$

where  $\bar{\mathbf{r}}_{k+1} = \bar{\mathbf{g}} - \mathbf{B}_{FF} \bar{\mathbf{x}}_{k+1}$  is the residual and  $\beta_k$  is chosen so that the conjugacy condition  $\bar{\mathbf{p}}_{k+1}^T \mathbf{B}_{FF} \bar{\mathbf{p}}_k = 0$  is verified.

To improve the convergence of CG, we use a scaled residual to generate the new direction. The direction update formula becomes

$$\bar{\mathbf{p}}_{k+1} = \mathbf{P}_{FF} \bar{\mathbf{r}}_{k+1} + \beta'_k \bar{\mathbf{p}}_k,$$

where  $\beta'_k$  is chosen to respect the conjugacy condition. Note that applying the scaling in the case of CG, is equivalent to preconditioning CG with  $\mathbf{P}_{FF}$ .

### 2.3 Limited memory quasi-Newton matrices

In a quasi-Newton method, the objective function is approximated about the current iterate  $\mathbf{x}_k$  by the quadratic model

$$m_k(\mathbf{x}_k + \mathbf{z}) = f_k + \mathbf{g}_k^T \mathbf{z} + \frac{1}{2} \mathbf{z}^T \mathbf{B}_k \mathbf{z}, \quad (8)$$

where  $f_k$  and  $\mathbf{g}_k$  are the objective value and gradient at  $\mathbf{x}_k$  respectively, and  $\mathbf{B}_k = \mathbf{B}_k^T$  is an approximation of  $\nabla^2 f(\mathbf{x}_k)$ . Secant methods are a special case in which  $\mathbf{B}_k$  must satisfy the secant equation

$$\mathbf{B}_k \mathbf{s}_{k-1} = \mathbf{y}_{k-1},$$

where  $\mathbf{s}_{k-1} := \mathbf{x}_k - \mathbf{x}_{k-1}$  and  $\mathbf{y}_{k-1} := \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})$ . Secant methods typically define  $\mathbf{B}_k$  as rank-one or rank-two update of  $\mathbf{B}_{k-1}$  involving  $\mathbf{s}_{k-1}$  and  $\mathbf{y}_{k-1}$ . This has the disadvantage that  $\mathbf{B}_k$  is almost always dense even though  $\nabla^2 f(\mathbf{x}_k)$  might be sparse. Therefore, the entire matrix  $\mathbf{B}_k$  must be stored, which is unrealistic in large-scale applications. However, at least conceptually, a product between  $\mathbf{B}_k$  and a vector could be computed without storing  $\mathbf{B}_k$  if the initial approximation  $\mathbf{B}_0$  along with all the pairs  $\{\mathbf{s}_i, \mathbf{y}_i\}_{0 \leq i \leq k-1}$  are stored instead.

In a limited-memory context, we store an initial matrix  $\mathbf{B}_0$  along with the  $m$  most recent pairs  $\{\mathbf{s}_i, \mathbf{y}_i\}_{k-m \leq i \leq k-1}$ , where  $m$  is the memory. The procedure uses the information from  $\mathbf{B}_0$  and from the  $m$  pairs to update and compute a product with  $\mathbf{B}_k$ . Even though  $\mathbf{B}_k$  would still be dense if it were materialized, it is only represented implicitly.

Because the memory  $m$  is often small compared to the problem dimension, quasi-Newton updates can only contribute a limited amount of information to  $\mathbf{B}_k$ . For this reason, the choice of  $\mathbf{B}_0$  is critical to obtain a good approximation of the objective Hessian. In particular, when the Hessian is ill conditioned, choosing  $\mathbf{B}_0$  as a multiple of the identity might lead to poor performance.

The best-known limited-memory quasi-Newton method is probably the limited-memory BFGS method (Nocedal, 1980), which additionally ensures that  $\mathbf{B}_k$  is positive definite provided that  $\mathbf{B}_{k-1}$  is positive definite and  $\mathbf{s}_{k-1}^T \mathbf{y}_{k-1} > 0$ . Several procedures exist to compute a product between  $\mathbf{B}_k$  or its inverse and a vector, including the two-loop recursion (Nocedal, 1980), and a variant based on compact storage (Byrd, Nocedal, and Schnabel, 1994). We present the second one because it was designed to handle bound constraints.

The pairs  $\{\mathbf{s}_i, \mathbf{y}_i\}_{k-m \leq i \leq k-1}$  are stored in two matrices

$$\mathbf{S} = (\mathbf{s}_{k-m} \quad \cdots \quad \mathbf{s}_{k-1}) \quad \text{and} \quad \mathbf{Y} = (\mathbf{y}_{k-m} \quad \cdots \quad \mathbf{y}_{k-1}),$$

and we define

$$\mathbf{D} = \begin{pmatrix} \mathbf{s}_{k-m}^T \mathbf{y}_{k-m} & & & \\ & \ddots & & \\ & & \mathbf{s}_{k-1}^T \mathbf{y}_{k-1} & \end{pmatrix} \quad \text{and} \quad \mathbf{L} = \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ \mathbf{s}_{k-m+1}^T \mathbf{y}_{k-m} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{s}_{k-1}^T \mathbf{y}_{k-m} & \cdots & \mathbf{s}_{k-1}^T \mathbf{y}_{k-2} & 0 \end{pmatrix}.$$

In the compact formula,  $\mathbf{B}_k$  is stored implicitly as  $\mathbf{B}_0$ ,

$$\mathbf{W} = (\mathbf{Y} \quad \mathbf{B}_0 \mathbf{S}) \quad \text{and} \quad \mathbf{M} = \begin{pmatrix} -\mathbf{D} & \mathbf{L}^T \\ \mathbf{L} & \mathbf{S}^T \mathbf{B}_0 \mathbf{S} \end{pmatrix}^{-1}, \quad (9)$$

such that

$$\mathbf{B} = \mathbf{B}_0 - \mathbf{W} \mathbf{M} \mathbf{W}^T. \quad (10)$$

where  $\mathbf{B}_0$  is positive definite.

In most implementations,  $\mathbf{B}_0$  is chosen as

$$\mathbf{B}_0 = \theta \mathbf{I} \quad \text{with} \quad \theta = \frac{\mathbf{y}_{k-1}^T \mathbf{y}_{k-1}}{\mathbf{y}_{k-1}^T \mathbf{s}_{k-1}}, \quad (11)$$

where  $\theta$  is a scaling parameter (Byrd et al., 1995). A diagonal  $\mathbf{B}_0$  leads to very efficient operations with the L-BFGS compact formula. However, it might be inappropriate for approximating ill-conditioned Hessians.

We choose  $\mathbf{B}_0$  so that the L-BFGS operator in  $\mathcal{X}$  reproduces the behavior of a standard L-BFGS operator with initial approximation (11) in  $\mathcal{U}$ .

Assume that, in  $\mathcal{U}$ ,  $\bar{f}$  is approximated about the current scaled iterate  $\mathbf{u}_k$  by the quadratic model

$$m'_k(\mathbf{u}_k + \mathbf{w}) = \bar{f}(\mathbf{u}_k) + \nabla \bar{f}(\mathbf{u}_k)^T \mathbf{w} + \frac{1}{2} \mathbf{w}^T \mathbf{B}'_k \mathbf{w}, \quad (12)$$

where  $\mathbf{B}'_k$  is a L-BFGS operator with initial approximation (11). The pairs  $\{\bar{\mathbf{s}}_i, \bar{\mathbf{y}}_i\}$  in  $\mathcal{U}$  are related to the pairs  $\{\mathbf{s}_i, \mathbf{y}_i\}$  in  $\mathcal{X}$  via

$$\bar{\mathbf{s}}_i = \mathbf{C}^{-1} \mathbf{s}_i \quad \bar{\mathbf{y}}_i = \mathbf{C}^T \mathbf{y}_i, \quad i = k-m, \dots, k-1. \quad (13)$$

We replace  $\{\mathbf{s}_i, \mathbf{y}_i\}$  with  $\{\bar{\mathbf{s}}_i, \bar{\mathbf{y}}_i\}$  in (10) and (11), and obtain

$$\mathbf{B}'_k = \bar{\theta} \mathbf{I} - (\mathbf{C}^T \mathbf{Y} \quad \bar{\theta} \mathbf{C}^{-1} \mathbf{S}) \begin{pmatrix} -\mathbf{D} & \mathbf{L}^T \\ \mathbf{L} & \bar{\theta} \mathbf{S}^T \mathbf{P}^{-1} \mathbf{S} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{Y}^T \mathbf{C} \\ \bar{\theta} \mathbf{S} \mathbf{C}^{-T} \end{pmatrix}, \quad (14)$$

where

$$\bar{\theta} = \frac{\bar{\mathbf{y}}_{k-1}^T \bar{\mathbf{y}}_{k-1}}{\bar{\mathbf{s}}_{k-1}^T \bar{\mathbf{y}}_{k-1}} = \frac{\mathbf{y}_{k-1}^T \mathbf{P} \mathbf{y}_{k-1}}{\mathbf{y}_{k-1}^T \mathbf{s}_{k-1}}. \quad (15)$$

We use (8) to approximate  $f$  in  $\mathcal{X}$ . A comparison between (8) and (12) yields

$$\mathbf{B}'_k = \mathbf{C}^T \mathbf{B}_k \mathbf{C}.$$

Finally  $\mathbf{B}$  is a L-BFGS operator with initial approximation

$$\mathbf{B}_0 = \bar{\theta} \mathbf{C}^{-T} \mathbf{C}^{-1} = \bar{\theta} \mathbf{P}^{-1} \quad (16)$$

Apart from the storage of  $\mathbf{P}$ , this modification does not require more storage in the compact L-BFGS formula. Instead of storing  $\mathbf{S}$  and  $\mathbf{S}^T \mathbf{S}$ , we store  $\mathbf{P}^{-1} \mathbf{S}$  and  $\mathbf{S}^T \mathbf{P}^{-1} \mathbf{S}$ , while  $\mathbf{L}$  and  $\mathbf{D}$  remain unchanged. The L-BFGS update only requires one product with  $\mathbf{P}^{-1}$  to compute  $\mathbf{P}^{-1} \mathbf{s}_k$  and one scalar product defined by  $\mathbf{P}$  to compute  $\bar{\theta}$ .

### 3 Modified algorithms

In this section, we present the salient elements of the L-BFGS and TRON algorithms, and of our implementations. Then we describe the modification we brought to apply the scaling strategy.

#### 3.1 The L-BFGS-B algorithm

The L-BFGS-B algorithm of Byrd et al. (1995) is a popular quasi-Newton method for bound-constrained problems. Its standard implementation exploits the compact representation of limited-memory quasi-Newton operators, a diagonal  $\mathbf{B}_0$ , and the Sherman-Morrison-Woodbury formula to solve linear systems whose coefficient is a principal submatrix corresponding to inactive indices. In our application,  $\mathbf{B}_0$  is nondiagonal and its principal submatrices are not structured, so the Sherman-Morrison-Woodbury approach would be inefficient.

##### 3.1.1 Presentation of the algorithm

At the beginning of an iteration, we compute the Cauchy point  $\mathbf{x}_k^C$  as the *exact* first local minimizer of (8) along the piecewise affine path

$$t \mapsto \text{Proj}(\mathbf{x}_k - t \mathbf{g}_k), \quad (17)$$

where  $\mathbf{g}_k$  is the objective gradient at the current iterate  $\mathbf{x}_k$ . The Cauchy point is obtained by successively examining the quadratic model (8) on each segment of (17). On a segment between two breakpoints, the model is a second-order polynomial function of the nonnegative parameter  $t$ . If the polynomial is nonincreasing on the segment, then the procedure moves to the next segment. Otherwise a minimizer is computed on the current segment and returned as the Cauchy point (Byrd et al., 1995).

For clarity, we now drop the iteration index  $k$ . The Cauchy point  $\mathbf{x}^C$  yields the set of fixed indices

$$I_+^C = \left\{ i \mid x_i^C = 0 \text{ and } \partial_i f(\mathbf{x}) > 0 \right\},$$

the associated subspace

$$F = \left\{ \mathbf{x} \in \mathbb{R}^n \mid \forall i \in I_+^C \quad x_i = x_i^C \right\}, \quad (18)$$

and the active face  $F_+ = F \cap \mathbb{R}_+^n$ . In order to compute a minimizer of (8) over  $F$  and obtain a descent direction, we set  $\mathbf{B} = \mathbf{B}_k$  and  $\mathbf{g} = \mathbf{g}_k$  in (7) and solve by way of the Sherman-Morrison-Woodbury formula. If we partition

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_F \\ \mathbf{W}_G \end{pmatrix},$$

where  $\mathbf{W}$  is defined in (9), the inverse of a principal submatrix of  $\mathbf{B}$  is

$$\mathbf{B}_{FF}^{-1} = \mathbf{B}_{0,FF}^{-1} - \mathbf{B}_{0,FF}^{-1} \mathbf{W}_F \left( \mathbf{M}^{-1} - \mathbf{W}_F^T \mathbf{B}_{0,FF}^{-1} \mathbf{W}_F \right)^{-1} \mathbf{W}_F^T \mathbf{B}_{0,FF}^{-1}. \quad (19)$$

With  $\mathbf{B}_0 = \theta \mathbf{I}$ , we have

$$\mathbf{B}_{FF}^{-1} = \theta^{-1} \mathbf{I} - \theta^{-2} \mathbf{W}_F \left( \mathbf{M}^{-1} - \theta^{-1} \mathbf{W}_F^T \mathbf{W}_F \right)^{-1} \mathbf{W}_F^T.$$

Finally, we determine the next iterate by strong Wolfe linesearch between the current iterate and the solution of (7), subject to the bound constraints.

Algorithm 1 shows an overview of the L-BFGS-B method.

---

**Algorithm 1:** Overview of the standard L-BFGS-B algorithm.

---

**Data:**  $\mathbf{x}_0$ , parameters  
**for**  $k = 0, 1, 2, \dots$  **do**  
    Compute an exact Cauchy Point  $\mathbf{x}_k^C$  along the projected path  $t \mapsto \text{Proj}(\mathbf{x}_k - t \nabla f(\mathbf{x}_k))$   
    Identify the active face  $F_+$   
    Find a minimizer of the model (8) over the affine subspace (18) by the Sherman-Morrison-Woodbury formula  
    Perform a strong Wolfe linesearch to find the next iterate  $\mathbf{x}_{k+1}$   
    Update the L-BFGS operator.  
**end**

---

### 3.1.2 Implementation in Matlab

Our MATLAB implementation of L-BFGS-B,<sup>1</sup> `lbfgsb.m`, solves (7) with CG instead of the Sherman-Morrison-Woodbury formula. Indeed, though `lbfgsb.m` uses (11) as an initial matrix, our implementation should work with (16) and (15). When  $\mathbf{B}_0$  is nondiagonal, computing one of the products between  $\mathbf{B}_{0,FF}^{-1}$  and a vector in (19) requires to solve a linear system and might be as expensive as computing a product with  $\mathbf{B}_{FF}^{-1}$  by CG.

We validated our implementation against a C translation of the original Fortran implementation (Zhu et al., 1997; Morales and Nocedal, 2011) provided by Stephen Becker<sup>2</sup> on a collection of standard problems. The C version uses the Sherman-Morrison-Woodbury formula to solve (7). Our benchmark comprises 128 bound-constrained problems from the CUTEst library (Gould, Orban, and Toint, 2015). The tests ran on a 3GHz Intel Core i7-5960X with 64GB of RAM. We report our results in the form of performance profiles in logarithmic scale (Dolan and Moré, 2002) in Figure 1.

Because we use MATLAB instead of C, `lbfgsb.m` is slower than the original version. However, the results are similar in terms of number of objective evaluations and iterations, even though a slight degradation in performance on standard problems is somewhat expected and matches the observations of Byrd et al. (1995).

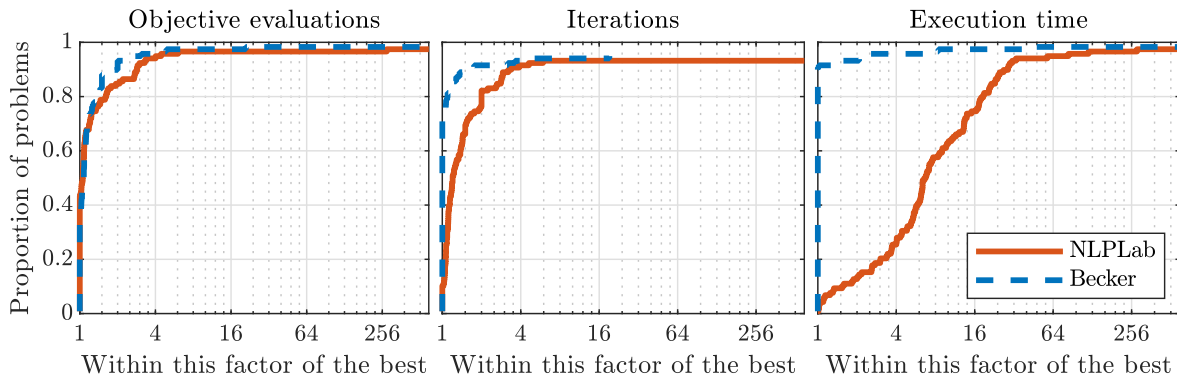


Figure 1: Log-scale performance profiles of our MATLAB implementation versus the C interface. The values compared are the number of objective evaluations (left), the number of iterations (middle) and the execution time (right).

### 3.1.3 Modifications related to scaling

In order to obtain a better Hessian approximation, we use (15) and (16) in the L-BFGS operator. A non-diagonal  $\mathbf{B}_0$  requires several modifications in the algorithm, as the procedures presented by Byrd et al. (1995) owe their efficiency to the diagonal structure of  $\mathbf{B}_0$ .

<sup>1</sup> Available online at <https://github.com/optimizers/NLPLab>

<sup>2</sup> Available online at <https://github.com/stephenbecker/L-BFGS-B-C>

Finding the Cauchy point requires examining up to  $n$  segments defined by the breakpoints along the projected gradient path. The feasibility of an exact search relies on the absence of any operation with complexity worse than  $O(n)$  in the update of the quadratic model derivatives along each segment. In particular, the scalar product between a row of  $\mathbf{B}_0$  and a vector is required for each segment visited. This operation is acceptable if applying  $\mathbf{B}_0$  to a vector is cheap, and in particular for diagonal  $\mathbf{B}_0$ . In our case, applying  $\mathbf{B}_0$  to a vector costs  $O(n \log n)$  operations and quickly becomes time consuming. Instead, we use an Armijo-like backtracking search, similar to that implemented in TRON (Lin and Moré, 1999, Section 6). In the inexact procedure, the Cauchy step must only satisfy the sufficient decrease condition

$$m_k(\mathbf{x}^C) \leq f(\mathbf{x}_k) + \mu_0 \mathbf{g}^T(\mathbf{x}^C - \mathbf{x}_k), \quad (20)$$

where  $0 < \mu_0 < \frac{1}{2}$ .

Our implementation is summarized in [Algorithm 2](#). In the next sections refer to it as `scaled-lbfgsb.m`.

---

**Algorithm 2:** Overview of the modified L-BFGS-B algorithm.

---

```

Data:  $\mathbf{x}_0$ , parameters
for  $k = 0, 1, 2, \dots$  do
    Identify the binding set  $I_+(\mathbf{x}_k)$ 
    Compute an inexact Cauchy point  $\mathbf{x}^C$  along the projected path  $t \mapsto \text{Proj}(\mathbf{x}_k - t \bar{\mathbf{P}} \nabla f(\mathbf{x}_k))$ 
    Identify the active face  $F_+$ 
    Find a minimizer of the model (8) over the affine subspace (18) using preconditioned CG
    Perform a strong Wolfe linesearch to find the next iterate  $\mathbf{x}_{k+1}$ 
    Update the L-BFGS operator.
end
```

---

## 3.2 A trust-region Newton method

Because second-order derivatives are available in our reconstruction problem, we also describe our scaling strategy in the context of a Newton method.

### 3.2.1 Presentation of the algorithm

TRON is a trust-region Newton method proposed by Lin and Moré (1999). As in L-BFGS-B, a general iteration includes the identification of an active face via a Cauchy point and the minimization of a quadratic model over an affine subspace corresponding to the free indices.

The quadratic model (8) now uses  $\mathbf{B}_k = \nabla^2 f(\mathbf{x}_k)$ . Due to the high cost of evaluating the quadratic model, we only compute an inexact Cauchy point satisfying (20).

Computing of a Newton direction requires finding an approximate solution of the trust-region subproblem

$$\begin{aligned} \min \quad & \frac{1}{2} \mathbf{x}^T \mathbf{B}_k \mathbf{x} + \mathbf{x}^T \mathbf{g} \\ \text{s.t. } \quad & \mathbf{x} \in F_+, \quad \|\mathbf{x} - \mathbf{x}_k\| \leq \Delta_k, \end{aligned} \quad (21)$$

where  $\Delta_k > 0$  is the trust-region radius. We generate successive minor iterates  $\mathbf{x}^j$  starting with  $\mathbf{x}^0 = \mathbf{x}^C$ , that satisfy

$$m_k(\mathbf{x}^{j+1}) \leq m_k(\mathbf{x}^j) + \min(0, \mu_0 \langle \mathbf{x}^{j+1} - \mathbf{x}^j, \nabla m_k(\mathbf{x}^j) \rangle). \quad (22)$$

To compute the first minor iterate  $\mathbf{x}^1$ , we solve (7) in the whole subspace  $F^0 = F$  with CG. If  $\hat{\mathbf{x}}$  is the solution and  $\mathbf{z} = \hat{\mathbf{x}} - \mathbf{x}^C$ , we perform a projected search along the path  $t \mapsto \text{Proj}(\mathbf{x}^C + t\mathbf{z})$  to determine  $\mathbf{x}^1$ . Then, we set the new active face to

$$F^1 = \left\{ \mathbf{x} \in F^0 \mid x_i^1 = 0 \Rightarrow x_i = 0 \right\}, \quad (23)$$

and we launch CG again to minimize the quadratic model in  $F^1$ . The procedure terminates when negative curvature is found in CG, when a CG iterate falls outside the trust-region or when a minor iterate satisfies a sufficient decrease condition of type

$$\|\mathbf{x}_k - \text{Proj}(\mathbf{x}_k - \nabla f(\mathbf{x}_k))\| \leq \varepsilon \|\mathbf{x}^j - \text{Proj}(\mathbf{x}^j - \nabla m_k(\mathbf{x}^j))\|. \quad (24)$$

When one of those stopping conditions is met, the decrease of the quadratic model and the decrease of the objective function at the last minor iterate are compared. Depending on this information, the last minor iterate is accepted as  $\mathbf{x}_{k+1}$  or rejected, and the trust-region radius is updated.

A summary of a TRON iteration is given in [Algorithm 3](#).

---

**Algorithm 3:** Overview of the standard TRON algorithm.

---

```

Data:  $\mathbf{x}_0$ , parameters
for  $k = 0, 1, 2, \dots$  do
    Compute an inexact Cauchy Point  $\mathbf{x}^C$  along the projected path  $t \mapsto \text{Proj}(\mathbf{x}_k - t \nabla f(\mathbf{x}_k))$ 
    Identify the active face  $F_+$ 
     $j \leftarrow 0$ ,  $F^0 \leftarrow F$ ,  $\mathbf{x}^0 \leftarrow \mathbf{x}^C$ 
    while (24) is not satisfied do
        Compute a minimizer  $\hat{\mathbf{x}}$  of the quadratic model over  $F^j$  subject to the trust-region constraint with CG
        Perform a projected search between  $\mathbf{x}^j$  and  $\hat{\mathbf{x}}$  to compute  $\mathbf{x}^{j+1}$ 
        Update  $F^{j+1}$  with formula (23)
         $j \leftarrow j + 1$ 
    end
    Accept or reject  $\mathbf{x}^j$  as the new iterate and update the trust-region radius
end

```

---

### 3.2.2 Implementation in Matlab

In the original TRON Fortran implementation, the conjugate gradient is preconditioned using an incomplete Cholesky factorization of  $\mathbf{B}_k$ . Such a factorization is not appropriate for large problems because the matrix coefficients are not explicitly available.

We use **Bcflash**, a Matlab implementation of TRON provided by Friedlander and Orban,<sup>3</sup> without preconditioning and where  $\mathbf{B}_k$  is only used as an operator.

For validation, we test **Bcflash** against the Fortran TRON implementation from which the incomplete Cholesky factorization was removed. The profiles in [Figure 2](#) show the performance results on 127 problems from the CUTEst library (Gould et al., 2015). The profiles show that **Bcflash** is more efficient in terms of function evaluations and Hessian products than the Fortran version. Moreover, the Matlab implementation is more robust. These results confirm the validity of Bcflash as an implementation of TRON.

**Bcflash** is competitive with the Fortran implementation in terms of execution time, whereas the difference is larger for L-BFGS-B. This can be partially explained as follows. In TRON, the bulk of the computation resides in Hessian-vector products. In both implementations the latter are computed by the CUTEst infrastructure, so this part of the computation is common between them. In L-BFGS-B, the objective function and gradient are only called at the beginning of the iteration and during the line search. Moreover, computations related to using and updating the limited-memory operator are difficult to vectorize efficiently, as they include operating on triangular matrices and reordering matrix columns. Thus, **lbfgsb.m** is at a disadvantage because those computations are implemented in Matlab.

A summary of the scaled variant of TRON appears in [Algorithm 4](#). From here on, we refer to it as **scaled-Bcflash**.

---

<sup>3</sup> Available online at <https://github.com/optimizers/NLPLab>

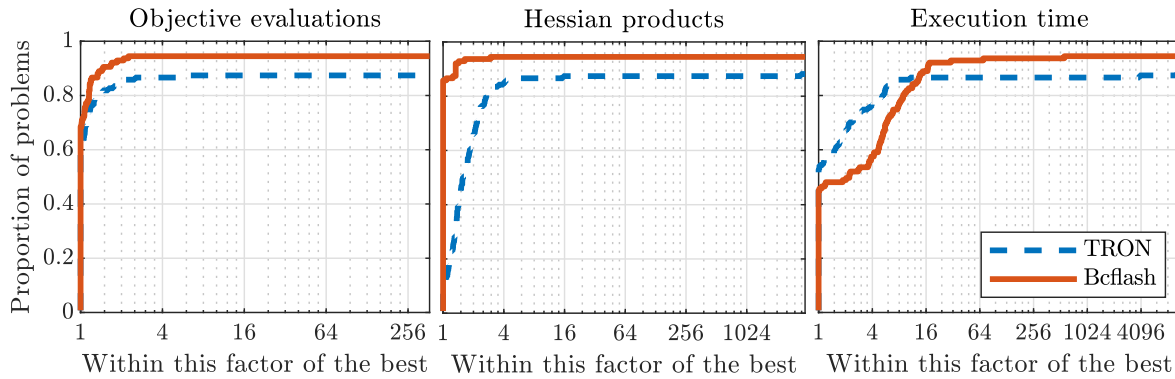


Figure 2: Performance profiles of Bcflash (Matlab) versus TRON without factorization (Fortran).

---

**Algorithm 4:** Overview of the modified TRON algorithm.

---

```

Data:  $\mathbf{x}_0$ , parameters
for  $k = 0, 1, 2, \dots$  do
    Identify the binding set  $I_+(\mathbf{x}_k)$ 
    Compute an inexact Cauchy Point  $\mathbf{x}^C$  along the projected path  $t \mapsto \text{Proj}(\mathbf{x}_k - t \bar{\mathbf{P}} \nabla f(\mathbf{x}_k))$ 
    Identify the active face  $F_+$ 
     $j \leftarrow 0$ ,  $F^0 \leftarrow F$ ,  $\mathbf{x}^0 \leftarrow \mathbf{x}^C$ 
    while (24) is not satisfied do
        Compute a minimizer  $\hat{\mathbf{x}}$  of the quadratic model over  $F^j$  subject to the trust-region constraint with CG
        preconditioned with  $\mathbf{P}_{FF}$ 
        Perform a projected search between  $\mathbf{x}^j$  and  $\hat{\mathbf{x}}$  to compute  $\mathbf{x}^{j+1}$ 
        Update  $F^{j+1}$  with formula (23)
         $j \leftarrow j + 1$ 
    end
    Accept or reject  $\mathbf{x}^j$  as the new iterate and update the trust-region radius
end

```

---

## 4 Numerical results

We now evaluate the performance of Algorithm 2 and Algorithm 4 on two image reconstruction problems. For both problems, we compare the performance of `scaled-lbfgsb.m` and `lbfgsb.m`, and that of `scaled-Bcflash` and `Bcflash`. We also compare `scaled-Bcflash` with a change-of-variable approach, by using `Cflash`, an implementation of TRON adapted to polyhedral constraints, to solve (3). In `Cflash`, projections are made onto the feasible set by solving a quadratic problem with Krylov methods (McLaughlin, 2017). Performances are compared in terms of projected gradient norm decrease and cumulated number of CG iterations along the reconstruction procedure.

First, to compare the convergence properties of the algorithms, we solve a simplified reconstruction problem, which is quadratic and better scaled than (2). In this first test, we also measure the fraction of execution time spent computing products with  $\mathbf{A}$  and  $\mathbf{C}$  for `Bcflash`, `scaled-Bcflash` and `Cflash`, in order to emphasize the high cost of constraints management in the third method compared to that in the two other methods.

In a second test, we use our methods on a real reconstruction problem. We evaluate the convergence acceleration caused by the use of scaled directions in this more complex case. To justify the choice of higher-order methods in image reconstruction, we also compare the performance of `scaled-Bcflash` with that of a first-order method, the spectral projected gradient (SPG) of Birgin and Martínez (2002).

In the following tests, we reconstruct images from a  $672 \times 1 \times 1160$  synthetic sinogram. In order to keep reasonable reconstruction times, we only consider 2D images. The data were created from a XCAT phantom (Segars, Mahesh, Beck, Frey, and Tsui, 2008) of size  $512 \times 512$  in cartesian coordinates.

We reconstruct a discretized image using polar coordinates, with 226 radial subdivisions and 1160 angular subdivisions. This discretization provides a sufficient resolution to obtain, after conversion, a  $512 \times 512$  cartesian image. Thus, the data creation and the image reconstruction are made using different procedures. In this problem,  $\mathbf{A}$  has 779,520 rows and 262,160 columns, and the initial guess is  $\mathbf{x}_0 = 0$ .

All results below are produced on an Intel<sup>®</sup> Xeon<sup>®</sup> E5-2637 v4 processor at 3.50 GHz and 32 GB of RAM.

#### 4.1 Simplified problem

We first consider the regularized linear least-squares problem

$$\min_{\mathbf{x} \geq 0} \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|^2 + \frac{1}{2} \lambda \|\mathbf{Kx}\|^2, \quad (25)$$

where  $\mathbf{Kx}$  models the differences between adjacent pixels of  $\mathbf{x}$ . In this simplified reconstruction problem, we drop the weight matrix  $\mathbf{V}$ , which is equivalent to assuming that all attenuation measures have the same variance, and we choose a  $L_2$  regularization to keep the problem quadratic. We set  $\lambda = 10^{-2}$  because it provides reasonable image quality and convergence speed.

Figure 3 shows the comparison between `lbfgsb.m` and `scaled-lbfgsb.m`. The left and right plots compare the decrease of the optimality residual and the cumulated number of CG iterations, respectively. We observe that the projected gradient norm decreases faster in the scaled case, especially in the first iterations, and that `scaled-lbfgsb.m` requires about half as many CG iterations per outer iteration as `lbfgsb.m`. The use of a nondiagonal  $\mathbf{B}_0$  yields L-BFGS approximations that are closer to the problem Hessian and lead to better progress at each step. For this reason, we see on the left plot that `scaled-lbfgsb.m` decreases the projected gradient norm more than `lbfgsb.m` while doing less outer iterations. Moreover, each outer iteration has a lower cost in terms of CG iterations due to the use of a preconditioner when solving (7).

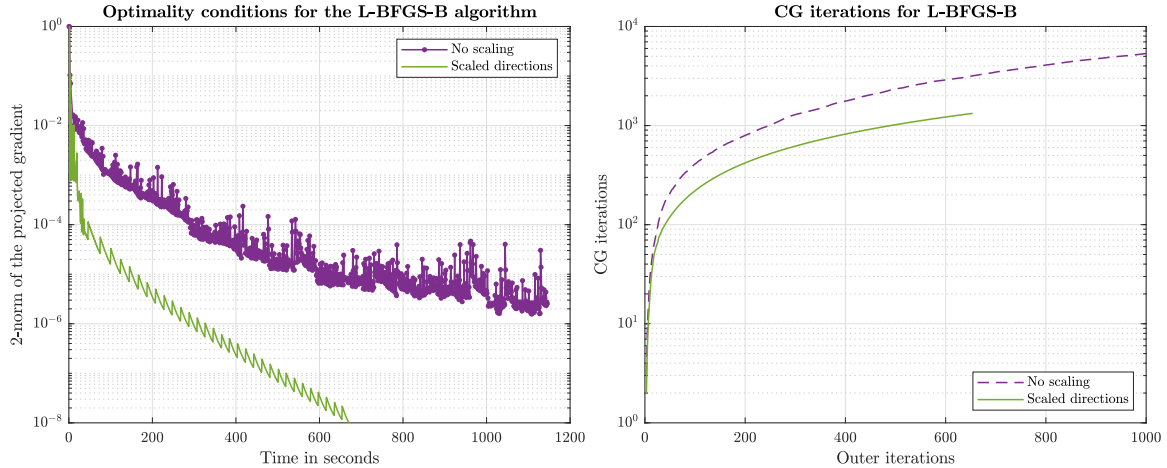


Figure 3: Convergence results for L-BFGS-B on (25).

However, the performance of `scaled-lbfgsb.m` is not sufficient. Though much progress is achieved during the 30 first seconds, the convergence seems to switch to a linear behavior at some point, and it takes more than two minutes to decrease the projected gradient norm by a factor of  $10^5$ .

Figure 4 reports corresponding results for TRON, where we compare `Bcflash`, `Cflash` and `scaled-Bcflash`. Both `Cflash` and `scaled-Bcflash` decrease the projected gradient much faster than `Bcflash`. Even though `scaled-Bcflash` requires more iterations than `Cflash` for the same gradient decrease, it converges faster.

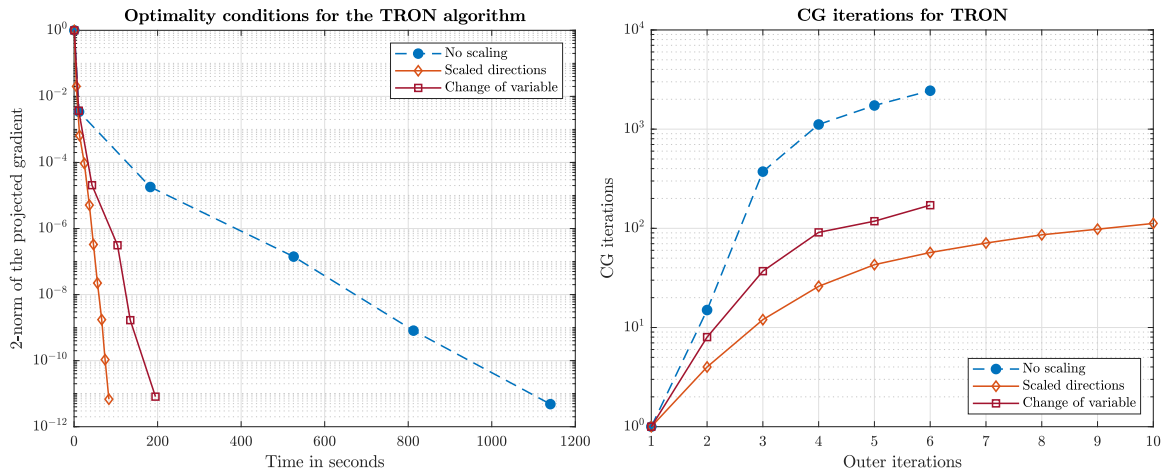


Figure 4: Convergence results for TRON on (25).

Table 1 gathers some statistics about the execution of TRON. Due to the problem size, a significant part of the execution time is associated with products with  $\mathbf{A}$  or its transpose. We see in the table that this time fraction is similar for *Bcflash* and *scaled-Bcflash*, both of which work in the original space, while it is much smaller for *Cflash*, which works in the scaled space. This difference is associated with the time spent computing orthogonal projections onto affine constraints in *Cflash*. Indeed, 23% of the solve time is spent computing products with  $\mathbf{C}$  or its transpose, most of which are computed during orthogonal projections. In *scaled-Bcflash*, time is saved on projections while the cost of conjugate gradient iterations remains similar.

Table 1: Execution statistics for the three versions of the TRON algorithm: fraction of time spent doing products with  $\mathbf{A}$  and  $\mathbf{C}$ .

| TRON variant                             | <i>Bcflash</i> | <i>scaled-Bcflash</i> | <i>Cflash</i> |
|------------------------------------------|----------------|-----------------------|---------------|
| Time fraction for $\mathbf{A}$ -products | 97 %           | 95 %                  | 51 %          |
| Time fraction for $\mathbf{C}$ -products | 0 %            | 1 %                   | 23 %          |

## 4.2 Reconstruction problem

In the second test, we compare Algorithm 2 and Algorithm 4 on the reconstruction problem

$$\min_{\mathbf{x} \geq 0} \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_{\mathbf{V}}^2 + \lambda \phi(\mathbf{K}\mathbf{x}), \quad (26)$$

where  $\phi : \mathbf{q} \mapsto \sum_i \sqrt{\delta^2 + q_i^2}$  is an edge-preserving  $L_2/L_1$  penalty with  $\delta > 0$ , and  $\mathbf{V}$  is the statistical weight matrix defined in (2).

Problem (26) should be more difficult to solve than (25), even for scaled methods. Indeed, the addition of weights deteriorates the Hessian conditioning, and the penalty is not quadratic. The objective Hessian in (26) has the form

$$\nabla^2 f(\mathbf{x}) = \mathbf{A}^T \mathbf{V} \mathbf{A} + \mathbf{K}^T \mathbf{N}(\mathbf{K}\mathbf{x}) \mathbf{K},$$

where  $\mathbf{N}(\mathbf{K}\mathbf{x})$  is a diagonal matrix that depends on  $\mathbf{K}\mathbf{x}$ . This Hessian is not block-circulant and is less likely to be well approximated by a block-circulant  $\mathbf{P}$  than that of (25).

We choose  $\lambda = 10^{-4}$  and  $\delta = 10^{-1}$ . We determine these parameters by comparing the quality of reconstructed images for several values of  $\lambda$  and  $\delta$  in terms of noise and blurring (Hamelin, 2009, section 4.5). To measure the noise level, we choose a zone that is uniform in the original phantom, and

compare the variance of the corresponding pixels in the reconstructed images. To evaluate blurring, we assume the reconstructed image is obtained from the true phantom by a linear space-invariant process (Gonzalez and Woods, 2008, section 5.6.1). According to this model, the true phantom  $\mathbf{x}_{\text{true}}$  and the reconstructed image  $\mathbf{x}^*$  satisfy  $\mathbf{x}^* = \mathbf{x}_{\text{true}} \star \mathbf{h} + \mathbf{n}$  where  $\mathbf{h}$  is the point-spread function,  $\mathbf{n}$  is a noise term and  $\star$  is the 2D convolution operator. For each image, we use the width of  $\mathbf{h}$  as a measure of blurring.

Figure 5 and Figure 6 show convergence results for L-BFGS-B and TRON, respectively. The solve time is longer than on (25) for all solvers, and the impact of the scaled solver is not as pronounced as in (25). The `scaled-lbfgsb.m` decreases the projected gradient by a factor of  $10^4$  about 9 times faster than `lbfgsb.m` on (25), but only 4 times faster on (26).

In the case of TRON, the advantage of `scaled-Bcflash` over `Cflash` is larger on (26) than on (25). The `scaled-Bcflash` decrease the projected gradient by a factor of  $10^7$  about 2.4 times faster than `Cflash` on (25), and 4.2 times faster on (26). So `scaled-Bcflash` is less affected by the scaling deterioration than `Cflash`.

Figure 7 shows a comparison between the convergence of `scaled-Bcflash` and the spectral projected gradient (SPG) of Birgin and Martínez (2002). SPG is sometimes used in imaging applications (Bonettini et al., 2008) and is appealing because the cost of each iteration is low. We modify SPG to employ the same scaling strategy as Algorithm 2 and Algorithm 4. SPG decreases the objective function decrease faster than `scaled-Bcflash` in the first iterations, which is to be expected from a first-order method. However, after a few iterations, the SPG projected gradient norm decrease slowly, whereas the decrease rate is superlinear for TRON. With a 3 minute time limit, `scaled-Bcflash` decreases the projected gradient by a factor of  $10^{10}$  while SPG only achieves a reduction of about  $10^5$ . We conclude that `scaled-Bcflash` is a more appropriate for solving the reconstruction problem at tolerances stricter than  $10^{-5}$ .

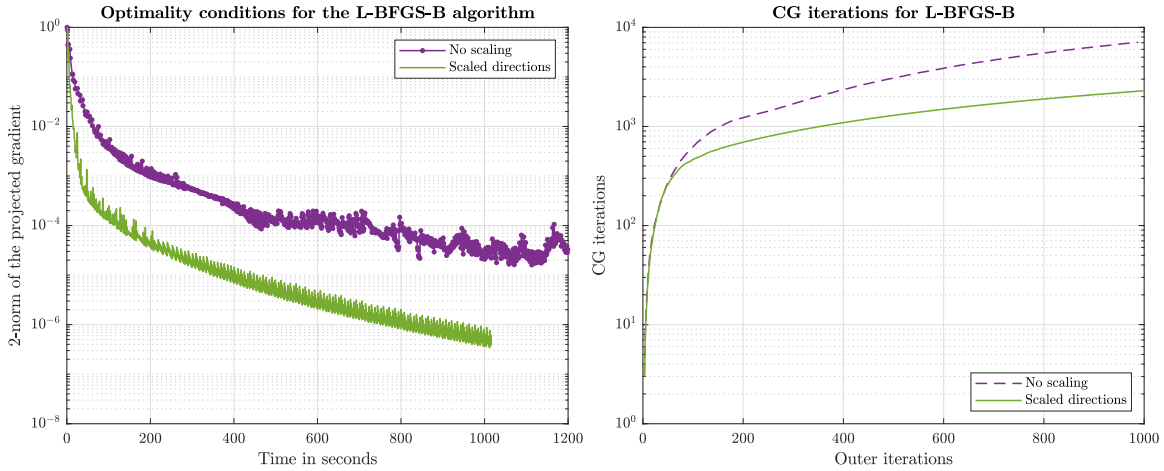


Figure 5: Convergence results for L-BFGS-B on (26).

Figure 8 shows images obtained by solving (26) with SPG with tolerance ranging from  $10^{-1}$  to  $10^{-5}$  on the projected gradient norm, and the `scaled-Bcflash` reconstruction with tolerance  $10^{-10}$ , which serves as reference. The picture obtained by SPG with tolerance  $10^{-5}$  is the only one that does not present artifacts compared to the reference. This shows that in order to obtain good image quality, we should solve (26) with a tolerance stricter than or equal to  $10^{-5}$ . We see in Figure 7 that `scaled-Bcflash` can achieve such accuracy faster than SPG.

These numerical results show that the scaling strategy brings the expected performance improvements for the problems we are interested in. In the case of TRON, this improvement is better than

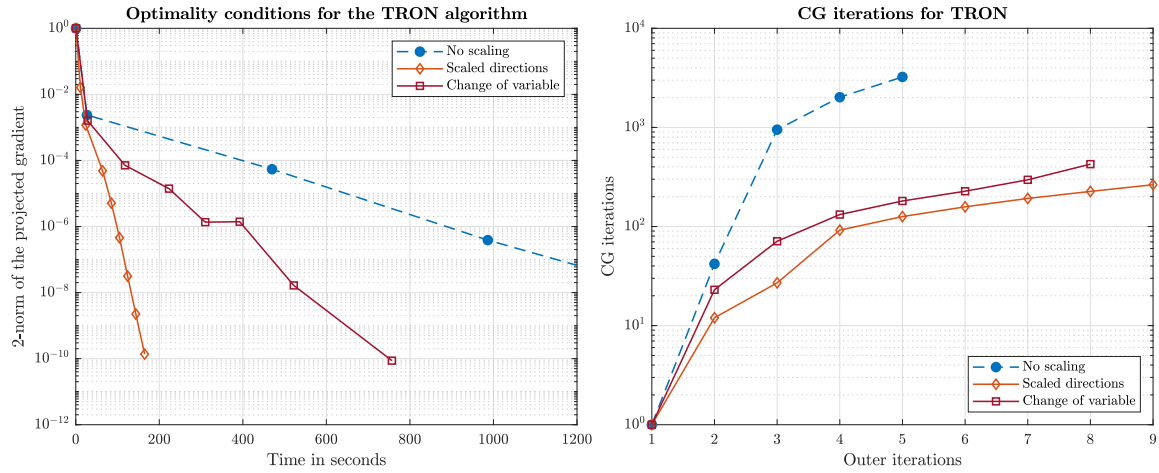


Figure 6: Convergence results for TRON on (26).

that we obtain with a change of variable, due to the high cost of orthogonal projections in the second approach.

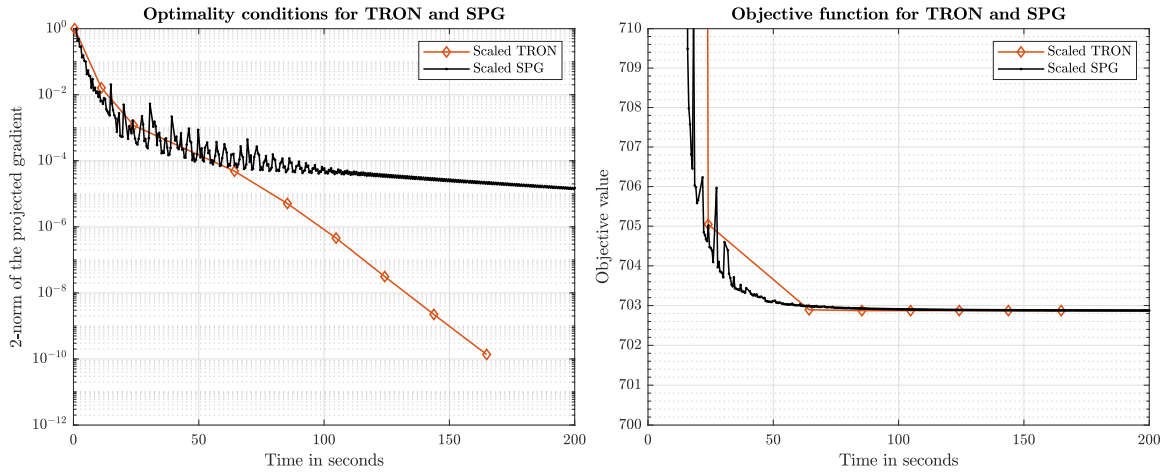
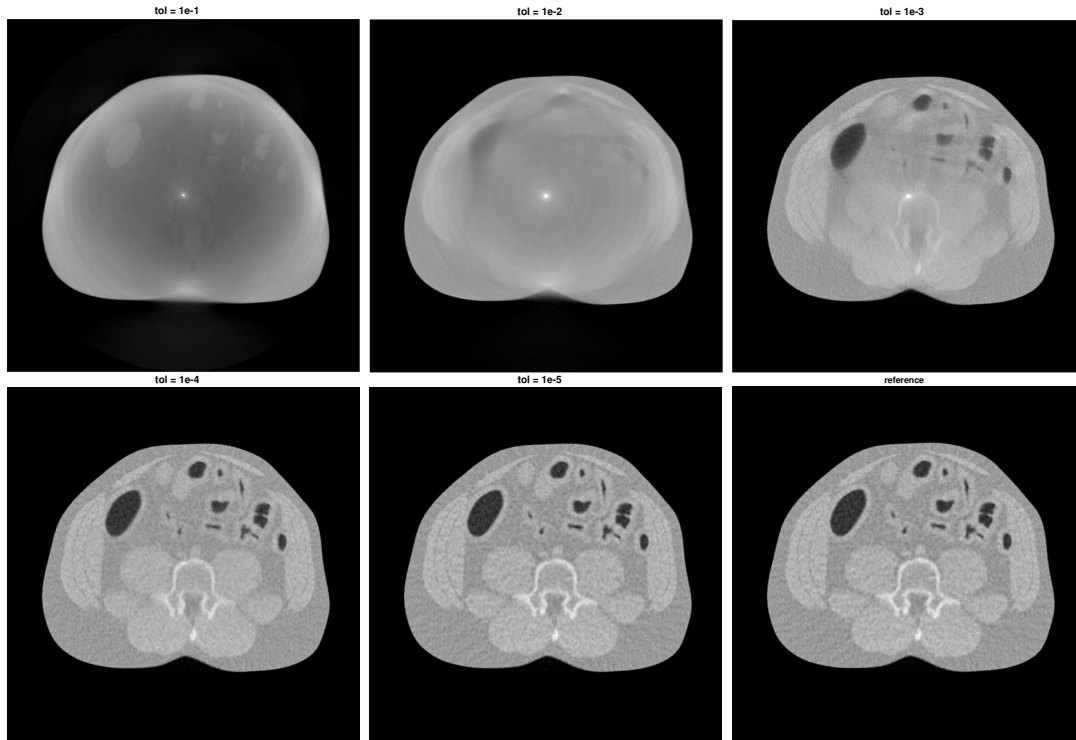


Figure 7: Comparison of TRON and SPG on (26)

## 5 Conclusion

We presented a scaling strategy for bound-constrained problems inspired from a change of variable, and integrated it into two projected-directions algorithms, L-BFGS-B and TRON. Though, our strategy can be implemented into most projected-directions algorithms for large bound-constrained problems with little code modifications. In this paper, we adopted a practical point of view, as we gave details about the implementation of scaling for each subroutine of the algorithms, including the preconditioning of CG to solve quadratic subproblems that appear in higher-order methods. The numerical tests on badly scaled image reconstruction problems show that this approach gives better results than a change of variable, especially because the management of constraints is cheaper.

These results are promising for applications in X-Ray CT reconstruction, as they show the feasibility of reconstructing images in cylindrical coordinates. The partially diagonal scaling ensures the efficiency of the procedure, making fast and memory-saving reconstructions possible. In particular, TRON is a good candidate for applications which require to solve the reconstruction problem with a tight



**Figure 8:** Images obtained with SPG for several tolerances. The reference image is obtained with TRON with tolerance  $10^{-10}$ .

tolerance. As TRON can solve (26) with tolerance  $10^{-10}$ , the reconstructed image is very close to the problem solution, and can be used as a reference to evaluate the convergence speed of other algorithms.

Here we have implemented the scaling into generic methods for large bound-constrained problems and solved generic reconstruction problems. In a future work, we will produce scaled methods for specific applications in medical image reconstruction, in order to combine the memory savings provided by the cylindrical coordinates with performance compatible with clinical applications. In particular, cylindrical coordinates are appropriate when the source and the detector follow a circular trajectory around the object to investigate, like in cone-beam computed tomography or in nondestructive testing. Structured system matrices can appear for other acquisition protocols, as long as the image discretisation yields geometrical invariances. Thus, block-circulant system matrices may also appear in helical computed tomography by using an helical discretization.

Circulant structures also appear in other imaging problems, for which our strategy can be applied. In general, our methods can prove to be useful in applications which lead naturally to non-diagonal scaling operators, like partial differential equations and optimal control, or when the optimization problem is too badly scaled for a diagonal scaling to be efficient.

## References

- S. Ahn, J. A. Fessler, D. Blatt, and A. O. Hero, III. Convergent incremental optimization transfer algorithms: Application to tomography. *IEEE Trans. Medical Imaging*, 25(3):283–296, Mar. 2006. DOI: [10.1109/TMI.2005.862740](https://doi.org/10.1109/TMI.2005.862740).
- D. P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM J. Control Optimization*, 20(2):221–246, 1982. DOI: [10.1137/0320018](https://doi.org/10.1137/0320018).
- E. G. Birgin and J. M. Martínez. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Comput. Optim. Appl.*, 23:101–125, 2002.

- S. Bonettini, R. Zanella, and L. Zanni. A scaled gradient projection method for constrained image deblurring. *Inverse Probl.*, 25(1):015002, 2008. DOI: [10.1088/0266-5611/25/1/015002](https://doi.org/10.1088/0266-5611/25/1/015002).
- S. Bonettini, G. Landi, E. L. Piccolomini, and L. Zanni. Scaling techniques for gradient projection-type methods in astronomical image deblurring. *International Journal of Computer Mathematics*, 90(1):9–29, 2013. DOI: [10.1080/00207160.2012.716513](https://doi.org/10.1080/00207160.2012.716513).
- R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited memory methods. *Math. Program.*, 63(1-3):129–156, 1994. DOI: [10.1007/BF01582063](https://doi.org/10.1007/BF01582063).
- R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, 1995. DOI: [10.1137/0916069](https://doi.org/10.1137/0916069).
- K. Choi, J. Wang, L. Zhu, T. Suh, S. Boyd, and L. Xing. Compressed sensing based cone-beam computed tomography reconstruction with a first-order method. *Math. Program.*, 37(9):5113–5125, 8 2010. DOI: [10.1118/1.3481510](https://doi.org/10.1118/1.3481510).
- E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program. A*, 91:201–213, 2002. DOI: [10.1007/s101070100263](https://doi.org/10.1007/s101070100263).
- H. Erdoğan and J. A. Fessler. Monotonic algorithms for transmission tomography. *IEEE Trans. Medical Imaging*, 18(9):801–814, Sep. 1999a. DOI: [10.1109/SSBI.2002.1233986](https://doi.org/10.1109/SSBI.2002.1233986).
- H. Erdoğan and J. A. Fessler. Ordered subsets algorithms for transmission tomography. *Phys. Med. Biol.*, 44(11):2835–2851, Nov. 1999b. DOI: [10.1088/0031-9155/44/11/311](https://doi.org/10.1088/0031-9155/44/11/311).
- L. A. Feldkamp, L. C. Davis, and J. W. Kress. Practical cone-beam algorithm. *J. Opt. Soc. Am. (A)*, 1(6):612–619, Jun. 1984. DOI: [10.1364/JOSAA.1.000612](https://doi.org/10.1364/JOSAA.1.000612).
- J. A. Fessler. Statistical image reconstruction methods for transmission tomography. In J. M. Fritzpatrick and M. Sonka, editors, *Handbook of Medical Imaging*, volume 2, chapter 1, pages 1–70. SPIE Press, Bellingham, WA, 2000.
- M. Golkar. Fast Iterative Reconstruction in X-Ray Tomography Using Polar Coordinates. PhD thesis, École Polytechnique de Montréal, 2013.
- R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Perarson Education, Inc., Upper Saddle River, NJ, 3rd edition, 2008.
- N. I. Gould, D. Orban, and P. L. Toint. CUTEst: A Constrained and Unconstrained Testing Environment with safe threads for mathematical optimization. *Comput. Optim. Appl.*, 60(3):545–557, 2015. DOI: [10.1007/s10589-014-9687-3](https://doi.org/10.1007/s10589-014-9687-3).
- Y. Goussard, M. Golkar, A. Wagner, and M. Voorons. Cylindrical coordinate representation for statistical 3D CT reconstruction. In *Proc. Int. Meeting on Fully 3D Image Reconstr. in Rad. and Nucl. Med.*, pages 138–141, Lake Tahoe, CA, Jun. 2013.
- B. Hamelin. Accélération d’une Approche Régularisée de Reconstruction en Tomographie à Rayons X avec Réduction des Artéfacts Métalliques. PhD thesis, École Polytechnique de Montréal, 2009.
- B. Hamelin, Y. Goussard, J.-P. Dussault, G. Cloutier, G. Beaudoin, and G. Soulez. Design of iterative ROI transmission tomography reconstruction procedures and image quality analysis. *Med. Phys.*, 37(9):4577–4589, Sep. 2010. DOI: [10.1118/1.3447722](https://doi.org/10.1118/1.3447722).
- G. T. Herman. *Fundamentals of computerized tomography: image reconstruction from projections*. Springer Science & Business Media, 2009.
- M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. of Stand.*, 49(6), 1952. DOI: [10.6028/jres.049.044](https://doi.org/10.6028/jres.049.044).
- H. M. Hudson and R. S. Larkin. Accelerated image reconstruction using ordered subsets of projection data. *IEEE Trans. Medical Imaging*, 13(4):601–609, Dec. 1994. DOI: [10.1109/42.363108](https://doi.org/10.1109/42.363108).

- T. L. Jensen, J. H. Jørgensen, P. C. Hansen, and S. H. Jensen. Implementation of an optimal first-order method for strongly convex total variation regularization. *BIT Num. Math.*, 52(2):329–356, Jun 2012. DOI: [10.1007/s10543-011-0359-8](https://doi.org/10.1007/s10543-011-0359-8).
- D. Kim, S. Ramani, and J. A. Fessler. Combining ordered subsets and momentum for accelerated X-ray CT image reconstruction. *IEEE transactions on medical imaging*, 34(1):167–178, 2014. DOI: [10.1109/TMI.2014.2350962](https://doi.org/10.1109/TMI.2014.2350962).
- G. Landi and E. Loli Piccolomini. A projected Newton-CG method for nonnegative astronomical image deblurring. *Numerical Algorithms*, 48(4):279–300, Aug 2008. DOI: [10.1007/s11075-008-9198-3](https://doi.org/10.1007/s11075-008-9198-3).
- K. Lange and J. A. Fessler. Globally convergent algorithms for maximum a posteriori transmission tomography. *IEEE Trans. Image Process.*, 4(10):1430–1438, Oct. 1995. DOI: [10.1109/83.465107](https://doi.org/10.1109/83.465107).
- C.-J. Lin and J. J. Moré. Newton’s method for large bound-constrained optimization problems. *SIAM J. Optim.*, 9(4):1100–1127, 1999. DOI: [10.1137/S1052623498345075](https://doi.org/10.1137/S1052623498345075).
- M. McLaughlin. Méthodes sans factorisation pour la tomographie à rayons X en coordonnées cylindriques. Master’s thesis, École Polytechnique de Montréal, 2017. URL <https://publications.polymtl.ca/2742>.
- J. L. Morales and J. Nocedal. Remark on “Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization”. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):7, 2011. DOI: [10.1145/2049662.2049669](https://doi.org/10.1145/2049662.2049669).
- Y. E. Nesterov. A method for solving the convex programming problem with convergence rate  $o(1/k^2)$ . In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547, 1983.
- H. Nien and J. A. Fessler. Fast X-ray CT image reconstruction using a linearized augmented Lagrangian method with ordered subsets. *IEEE Transactions on Medical Imaging*, 34(2):388–399, Feb 2015. DOI: [10.1109/TMI.2014.2358499](https://doi.org/10.1109/TMI.2014.2358499).
- J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comp.*, 35(151):773–782, 1980. DOI: [10.1090/S0025-5718-1980-0572855-7](https://doi.org/10.1090/S0025-5718-1980-0572855-7).
- F. Noo, K. Hahn, H. Schöndube, and K. Stierstorfer. Iterative CT reconstruction using coordinate descent with ordered subsets of data. In *Medical Imaging 2016: Physics of Medical Imaging*, volume 9783, page 97834A. International Society for Optics and Photonics, 2016. DOI: [10.1117/12.2217558](https://doi.org/10.1117/12.2217558).
- T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *Int. Conf. on Comp. Vision*, pages 1762–1769. IEEE, 2011. DOI: [10.1109/ICCV.2011.6126441](https://doi.org/10.1109/ICCV.2011.6126441).
- K. D. Sauer and C. A. Bouman. A local update strategy for iterative reconstruction from projections. *IEEE Trans. Signal Process.*, SP-41(2):534–548, Feb. 1993. DOI: [10.1109/78.193196](https://doi.org/10.1109/78.193196).
- W. P. Segars, M. Mahesh, T. J. Beck, E. C. Frey, and B. M. W. Tsui. Realistic CT simulation using the 4D XCAT phantom. *Med. Phys.*, 35(8):3800–8, Sep. 2008. URL [http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2809711/pdf/MPHYA6-000035-003800\\_1.pdf](http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2809711/pdf/MPHYA6-000035-003800_1.pdf).
- E. Y. Sidky, J. S. Jørgensen, and X. Pan. First-order convex feasibility algorithms for X-ray CT. *Medical physics*, 40(3), 2013. DOI: [10.1118/1.4790698](https://doi.org/10.1118/1.4790698).
- C. Thibaudau, J.-D. Leroux, R. Fontaine, and R. Lecomte. Fully 3D iterative CT reconstruction using polar coordinates. *Med. Phys.*, 40(11):111904, 2013. DOI: [10.1118/1.4822514](https://doi.org/10.1118/1.4822514).
- Q. Xu, D. Yang, J. Tan, A. Sawatzky, and M. A. Anastasio. Accelerated fast iterative shrinkage thresholding algorithms for sparsity-regularized cone-beam CT image reconstruction. *Medical physics*, 43(4):1849–1872, 2016. DOI: [10.1118/1.4942812](https://doi.org/10.1118/1.4942812).
- C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778. L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization. *ACM Trans. Math. Soft.*, 23(4):550–560, 1997. DOI: [10.1145/279232.279236](https://doi.org/10.1145/279232.279236).