trlib: A vector-free implementation of the GLTR method for iterative solution of the trust region problem

F. Lenders* and C. Kirches* and A. Potschka*

*Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University, Germany.

(Received 00 Month 20XX; final version received 00 Month 20XX)

We describe trlib, a library that implements a variant of Gould’s Generalized Lanczos method (Gould et al. in SIAM J. Opt. 9(2), 504–525, 1999) for solving the trust region problem.

Our implementation has several distinct features that set it apart from preexisting ones. We implement both conjugate gradient (CG) and Lanczos iterations for assembly of Krylov subspaces. A vector- and matrix-free reverse communication interface allows the use of most general data structures, such as those arising after discretization of function space problems. The hard case of the trust region problem frequently arises in sequential methods for nonlinear optimization. In this implementation, we made an effort to fully address the hard case in an exact way by considering all invariant Krylov subspaces.

We investigate the numerical performance of trlib on the full subset of unconstrained problems of the CUTEst benchmark set. In addition to this, interfacing the PDE discretization toolkit FEniCS with trlib using the vector-free reverse communication interface is demonstrated for a family of PDE-constrained control trust region problems adapted from the OPTPDE collection.

Keywords: trust-region subproblem, iterative method, Krylov subspace method, PDE constrained optimization

AMS Subject Classification: 35Q90, 65K05, 90C20, 90C30, 97N90

1. Introduction

In this article, we are concerned with solving the trust region problem, as it frequently arises as a subproblem in sequential algorithms for nonlinear optimization.

For this, let \( \mathcal{H} \) denote a Hilbert space with inner product \( \langle \cdot, \cdot \rangle \) and norm \( \| \cdot \| \). Then, \( H : \mathcal{H} \to \mathcal{H} \) denotes a self-adjoint, bounded operator on \( \mathcal{H} \) and with compact negative part (this assumption is needed to guarantee sequential weak lower semicontinuity of \( x \mapsto \langle x, Hx \rangle \), for details compare [24]). The operator \( M : \mathcal{H} \to \mathcal{H} \) is self-adjoint, bounded and coercive such that it induces an inner product \( \langle \cdot, \cdot \rangle_M \) with corresponding norm \( \| \cdot \|_M \) via \( \langle x, y \rangle_M := \langle x, My \rangle \) and \( \| x \|_M := \sqrt{\langle x, x \rangle_M} \). Furthermore, let \( X \subseteq \mathcal{H} \) be a closed subspace.

* Corresponding author. Email: felix.lenders@iwr.uni-heidelberg.de.
The trust region subproblem we are interested in reads

\[
\left\{ \begin{array}{l}
\min_{x \in \mathcal{H}} \; \frac{1}{2} \langle x, Hx \rangle + \langle x, g \rangle \\
\text{s.t.} \; \|x\|_M \leq \Delta, \quad x \in \mathcal{X},
\end{array} \right.
\]

with gradient \( g \in \mathcal{H} \), objective function \( q(x) := \frac{1}{2} \langle x, Hx \rangle + \langle x, g \rangle \), and trust region radius \( \Delta > 0 \). Usually we take \( \mathcal{X} = \mathcal{H} \) but will also consider truncated versions where \( \mathcal{X} \) is a finite dimensional subspace of \( \mathcal{H} \).

Readers who are less comfortable with the function space setting may think of \( H \) as a symmetric positive definite matrix, of \( \mathcal{H} \) as \( \mathbb{R}^n \), and of \( M \) as the identity on \( \mathbb{R}^n \) inducing the standard scalar product and the euclidean norm \( \| \cdot \|_2 \). We follow the convention to indicate coordinate vectors \( x \in \mathbb{R}^n \) with boldface letters.

Related Work

Trust Region Subproblems are an important ingredient in modern optimization algorithms as globalization mechanism. The monography [9] provides an exhaustive overview on Trust Region Methods for nonlinear programming, mainly for problems formulated in finite-dimensional spaces. For trust region algorithms in Hilbert spaces, we refer to [23, 25, 50, 51]. In [1] applications of trust region subproblems formulated on Riemannian manifolds are considered. Recently, trust region-like algorithms with guaranteed complexity estimates in relation to the KKT tolerance have been proposed [5, 6, 10]. The necessary ingredients in the subproblem solver for the algorithm investigated by Curtis and Samadi [10] have been implemented in trlib as well.

Solution algorithms for trust region problems can be classified into direct algorithms that make use of matrix factorizations and iterative methods that access the operators \( H \) and \( M \) only via evaluations \( x \mapsto Hx \) and \( x \mapsto Mx \) or \( x \mapsto M^{-1}x \). For the Hilbert space context, we are interested in the latter class of algorithms. We refer to [9] and the references therein for a survey of direct algorithms, but point out the algorithm of Moré and Sorensen [35] that will be used on a specific tridiagonal subproblem, as well as the work of Gould et al. [21], who use higher order Taylor models to obtain high order convergence results. The first iterative method was based on the conjugate gradient process, and was proposed independently by Toint [49] and Steihaug [48]. Gould et al. [19] proposed an extension of the Steihaug-Toint algorithm. There, the Lanczos algorithm is used to build up a nested sequence of Krylov spaces, and tri-diagonal trust region subproblems are solved with a direct method. This idea also forms the basis for our implementation. Hager [22] considers an approach that builds on solving the problem restricted to a sequence of subspaces that use SQP iterates to accelerate and ensure quadratic convergence. Erway et al. [13, 14] investigate a method that also builds on a sequence of subspaces built from accelerator directions satisfying optimality conditions of a primal-dual interior point method. They allow to decouple the preconditioner from the operator defining the trust region norm. Eigenvalue based approaches are studied by Sorensen [47], Rendl and Wolkowicz [43], and Rojas et al. [45, 46].

Contributions

We introduce trlib which is a new vector-free implementation of the GLTR method for solving the trust region subproblem. We assess the performance of this implementation
on trust region problems obtained from the set of unconstrained nonlinear minimization problems of the CUTEst benchmark library, as well as on a number of examples formulated in Hilbert space that arise from PDE-constrained optimal control.

Structure of the Article

The remainder of this article is structured as follows. In §2, we briefly review conditions for existence and uniqueness of minimizers. The GLTR methods for iteratively solving the trust region problem is presented in §3 in detail. Our implementation, trlib is introduced in §4. Numerical results for trust-region problems arising in nonlinear programming and in PDE-constrained control are presented in §5. Finally, we offer a summary and conclusions in §6.

2. Existence and Uniqueness of Minimizers

In this section, we briefly summarize the main results about existence and uniqueness of solutions of the trust region subproblem. We first note that our introductory setting implies the following fundamental properties:

Lemma 1 (Properties of \((\text{TR}(H,g,M,\Delta,X))\))

1. The mapping \(x \mapsto \langle x, Hx \rangle\) is sequentially weakly lower semicontinuous, and Fréchet differentiable for every \(x \in H\).
2. The feasible set \(F := \{x \in H \mid \|x\|_M \leq \Delta\}\) is bounded and weakly closed.
3. The operator \(M\) is surjective.

Proof. \(H\) has compact negative part, so (1) follows from [24]. Fréchet differentiability follows from boundedness of \(H\). Boundedness of \(F\) follows from coercivity of \(M\). Furthermore, \(F\) is obviously convex and strongly closed, hence weakly closed. Finally, (3) follows by the Lax-Milgram theorem [8, ex. 6.19].

Lemma 2 (Existence of a solution)

Problem \((\text{TR}(H,g,M,\Delta,X))\) has a solution.

Proof. By Lemma 1, the objective functional \(q\) is sequentially weakly lower semicontinuous and the feasible set \(F\) is weakly closed and bounded, the claim follows then from a generalized Weierstrass Theorem [26, Ch. 7].

To present optimality conditions for the trust region subproblem, we first present a helpful lemma on the change of the objective function between two points on the trust region boundary.

Lemma 3 (Objective Change on Trust Region Boundary)

Let \(x^0, x^1 \in H\) with \(\|x^i\|_M = \Delta\) for \(i = 0, 1\) be boundary points of \((\text{TR}(H,g,M,\Delta,X))\), and let \(\lambda \geq 0\) satisfy \((H + \lambda M)x^0 + g = 0\). Then \(d = x^1 - x^0\) satisfies \(q(x^1) - q(x^0) = \frac{1}{2}\langle d, (H + \lambda M)d \rangle\).

Proof. Using \(0 = \|x^1\|_M^2 - \|x^0\|_M^2 = \langle x^1 + d, M(x^0 + d) \rangle - \langle x^0 + Mx^0, x^0 \rangle = \langle d, Md \rangle + 2\langle x^0, Md \rangle\)
and \( g = -(H + \lambda M)x^0 \) we find

\[
q(x^1) - q(x^0) = \frac{1}{2} \langle d, Hd \rangle + \langle d, Hx^0 \rangle + \langle g, d \rangle = \frac{1}{2} \langle d, Hd \rangle - \lambda \langle x^0, Md \rangle
\]

\[
= \frac{1}{2} \langle d, (H + \lambda M)d \rangle.
\]

\( \Box \)

**Theorem 4 (Necessary Optimality Conditions)**

Let \( x^* \in \mathcal{H} \) be a global solution of \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\). Then there is \( \lambda^* \geq 0 \) such that

(a) \( (H + \lambda^* M)x^* + g = 0 \),

(b) \( \|x^*\|_M - \Delta \leq 0 \),

(c) \( \lambda^* (\|x^*\|_M - \Delta) = 0 \),

(d) \( \langle d, (H + \lambda^* M)d \rangle \geq 0 \) for all \( d \in \mathcal{H} \).

**Proof.** Let \( \sigma : \mathcal{H} \to \mathbb{R}, \sigma(x) := \langle x, Mx \rangle - \Delta^2 \), so that the trust region constraint becomes \( \sigma(x) \leq 0 \). The function \( \sigma \) is Fréchet-differentiable for all \( x \in \mathcal{H} \) with surjective differential provided \( x \neq 0 \) and satisfies constraint qualifications in that case. We may assume \( x^* \neq 0 \) as the theorem holds for \( x^* = 0 \) (then \( g = 0 \)) for elementary reasons.

Now if \( x^* \) is a solution of \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\), conditions (a)–(c) are necessary optimality conditions, cf. [9]. If \( \|x^*\|_M < \Delta \), condition (d) follows from second order necessary conditions for unconstrained optimization, cf. [37]. If \( \|x^*\|_M = \Delta \), then (d) has to hold for all \( d \in \mathcal{H} \) with \( \langle d, Mx^* \rangle = 0 \) by necessary conditions. It remains to prove (d) for \( d \in \mathcal{H} \) with \( \langle d, Mx^* \rangle \neq 0 \). Given such a \( d \), there is \( \alpha \in \mathbb{R} \setminus \{0\} \) with \( \|x^0 + \alpha d\|_M = \Delta \).

Using Lemma 3 yields \( \langle d, (H + \lambda^* M)d \rangle = \frac{2}{\alpha^2} (q(x^* + \alpha d) - q(x^*)) \geq 0 \) since \( x^* \) is global minimizer. \( \Box \)

**Corollary 5 (Sufficient Optimality Condition)**

Let \( x^* \in \mathcal{H} \) and \( \lambda^* \geq 0 \) such that (a)–(c) of Thm. 4 hold and \( \langle d, (H + \lambda^* M)d \rangle > 0 \) holds for all \( d \in \mathcal{H} \). Then \( x^* \) is the unique global solution of \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\).

**Proof.** This is an immediate consequence of Lemma 3. \( \Box \)

### 3. The GLTR Method

The GLTR (Generalized Lanczos Trust Region) method is an iterative method to approximatively solve \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\) and has first been described in Gould et al. [19].

Our presentation follows the presentation there and only deviates in minor details.

In every iteration of the GLTR process, problem \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\) is restricted to the Krylov subspace \( \mathcal{K}_i := \text{span}\{(M^{-1}H)^jM^{-1}g | 0 \leq j \leq i\} \),

\[
\min_{x \in \mathcal{K}_i} \begin{cases} \frac{1}{2} \langle x, Hx \rangle + \langle x, g \rangle \\ \|x\|_M \leq \Delta, \end{cases}
\]

\( (\text{TR}(H, g, M, \Delta, \mathcal{K}_i)) \)

The following Lemma relates solutions of \((\text{TR}(H, g, M, \Delta, \mathcal{K}_i))\) to those
of \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\).

**Lemma 6 (Solution of the Krylov subspace trust region problem)**

Let \(x^i\) be a global minimizer of \((\text{TR}(H, g, M, \Delta, \mathcal{K}_i))\) and \(\lambda^i\) the corresponding Lagrange multiplier. Then \((x^i, \lambda^i)\) satisfies the global optimality conditions of \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\) (Thm. 4) in the following sense:

(a) \(\left( H + \lambda^i M \right)x^i + g \perp M \mathcal{K}_i \),
(b) \(\| x^i \|_M - \Delta \leq 0 \),
(c) \(\lambda^i (\| x^i \|_M - \Delta) = 0 \),
(d) \(\langle d, (H + \lambda^i M) d \rangle \geq 0 \) for all \(d \in \mathcal{K}_i \).

**Proof.** (b)–(d) are immediately obtained from Thm. 4 as \(\mathcal{K}_i \subseteq \mathcal{H}\) is a Hilbert space. Assertion (a) follows from \(x^* = x^i + x^\perp\) with \(x^i \in \mathcal{K}_i\), \(x^\perp \in \mathcal{K}^\perp\) and Thm. 4 for \(x^i\). ■

Solving problem \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\) may thus be achieved by iterating the following Krylov subspace process. Each iteration requires the solution of an instance of the truncated trust region subproblem \((\text{TR}(H, g, M, \Delta, \mathcal{K}_i))\).

**input:** \(H, M, g, \Delta\), \(\text{tol}\)

**output:** \(i, x^i, \lambda^i\)

**for** \(i \geq 0\) **do**

- Construct a basis for the \(i\)-th Krylov subspace \(\mathcal{K}_i\)
- Compute a representation of \(q(x)\) restricted to \(\mathcal{K}_i\)
- Solve the subproblem \((\text{TR}(H, g, M, \Delta, \mathcal{K}_i))\) to obtain \((x^i, \lambda^i)\)

**if** \(\| (H + \lambda^i M)x^i + g \|_{M^{-1}} \leq \text{tol} \) **then return**

**end**

Algorithm 1: Krylov subspace process for solving \((\text{TR}(H, g, M, \Delta, \mathcal{H}))\).

Algorithm 1 stops the subspace iteration as soon as \(\| (H + \lambda^i M)x^i + g \|_{M^{-1}}\) is small enough. The norm \(\| \cdot \|_{M^{-1}}\) is used in the termination criterion since it is the norm belonging to the dual of \((\mathcal{H}, \| \cdot \|_M)\) and the Lagrange gradient \((H + \lambda^i M)x^i + g\) should be regarded as element of the dual.

### 3.1 Krylov Subspace Buildup

In this section, we present the preconditioned conjugate gradient (pCG) process and the preconditioned Lanczos process (pL) for construction of Krylov subspace bases. We discuss the transition from pCG to pL upon breakdown of the pCG process.

#### 3.1.1 Preconditioned Conjugate Gradient Process

An \(H\)-conjugate basis \((\tilde{p}_j)_{0 \leq j \leq N}\) of \(\mathcal{K}_i\) may be obtained using preconditioned conjugate gradient (pCG) iterations, Algorithm 2.

The stationary point \(s^i\) of \(q(x)\) restricted to the Krylov subspace \(\mathcal{K}_i\) is given by \(s^i = \sum_{j=0}^{N-1} \alpha^j \tilde{p}^j\) and can thus be computed using the recurrence

\[
 s^0 \leftarrow \alpha^0 \tilde{p}^0, \quad s^{j+1} \leftarrow s^j + \alpha^{j+1} \tilde{p}^{j+1}, \quad 0 \leq j \leq N - 1
\]
November 14, 2016 Optimization Methods & Software

Algorithm 2: Preconditioned conjugate gradient (pCG) process.

3.1.2 Preconditioned Lanczos Process

An $M$-orthogonal basis $(p_j)_{0 \leq j \leq i}$ of $K_i$ may be obtained using the preconditioned Lanczos (pL) process, Algorithm 3, and permits to continue subspace iterations even after pCG breakdown.

Algorithm 3: Preconditioned Lanczos (pL) process.

The following simple relationship holds between the Lanczos iteration data and the pCG iteration data, and may be used to initialize the pL process from the final pCG iterate before breakdown:

$$
g^i = \frac{1}{\|v^i \|_M} \prod_{j=0}^{i-1} (-\text{sign} \alpha^j) \hat{v}_i, \quad g^i = \gamma^i / \|v^i \|_M \prod_{j=0}^{i-1} (-\text{sign} \alpha^j) \hat{g}_i.
$$
In turn, breakdown of the pL process occurs if an invariant subspace of $H$ is exhausted, in which case $\gamma^i = 0$. If this subspace does not span $H$, the pL process may be restarted if provided with a vector $g^0$ that is $M$-orthogonal to the exhausted subspace.

The pL process may also be expressed in compact matrix form as

$$HP_i - MP_iT_i = g^{i+1}e_{i+1}^T, \quad \langle P_i, MP_i \rangle = I,$$

with $P_i$ being the matrix composed from columns $p_0, \ldots, p_i$, and $T_i$ the symmetric tridiagonal matrix with diagonal elements $\delta^0, \ldots, \delta^i$ and off-diagonal elements $\gamma^1, \ldots, \gamma^i$.

As $P_i$ is a basis for $K_i$, every $x \in K_i$ can be written as $x = P_i h$ with a coordinate vector $h \in \mathbb{R}^{i+1}$. Using the compact form of the Lanczos iteration, one can immediately express the quadratic form in this basis as $q(x) = \frac{1}{2} \langle h, T_i h \rangle + \gamma^0(e_1, h)$. Similarly, $\|x\|_M = \|h\|_2$. Solving $(\text{TR}(H, g, M, \Delta, K_i))$ thus reduces to solving $\text{TR}(T_i, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$ on $\mathbb{R}^{i+1}$ and recovering $x = P_i h$.

### 3.2 Easy and Hard case of the Tridiagonal Subproblem

As just described, using the tridiagonal representation $T_i$ of $H$ on the basis $P_i$ of the $i$-th iteration of the pL process, the trust-region subproblem $\text{TR}(T_i, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$ needs to be solved. For notational convenience, we drop the iteration index $i$ from $T_i$ in the following. Considering the necessary optimality conditions of Thm. 4, it is natural to define the mapping

$$\lambda \mapsto x(\lambda) := (T + \lambda I)^n(-\gamma^0e_1) \text{ for } \lambda \in I \equiv [\max\{0, -\theta_{\min}\}, \infty),$$

where $\theta_{\min}$ denotes the smallest eigenvalue of $T$, and the superscript $+$ denotes the Moore-Penrose pseudo-inverse. On $I$, $T + \lambda I$ is positive semidefinite. The following definition relates $x(\lambda^*)$ to a global minimizer $(x^*, \lambda^*)$ of $\text{TR}(T, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$.

**Definition 7** (Easy Case and Hard Case)

Let $(x^*, \lambda^*)$ satisfy the necessary optimality conditions of Thm. 4.

If $\langle \gamma^0e_1, \text{Eig}(\theta_{\min}) \rangle \neq 0$, we say that $T$ satisfies the easy case. Then, $x^* = x(\lambda^*)$.

If $\langle \gamma^0e_1, \text{Eig}(\theta_{\min}) \rangle = 0$, we say that $T$ satisfies the hard case. Then, $x^* = x(\lambda^*) + v$ with suitable $v \in \text{Eig}(\theta_{\min})$. Here $\text{Eig}(\theta) = \{v \in \mathbb{R}^{i+1} \mid Tv = \theta v\}$ denotes the eigenspace of $T$ associated to $\theta$.

With the following theorem, Gould et al. in [19] use the the irreducible components of $T$ to give a full description of the solution $x(\lambda^*) + v$ in the hard case.

**Theorem 8** (Global Minimizer in the Hard Case)

Let $T = \text{diag}(R_1, \ldots, R_k)$ with irreducible tridiagonal matrices $R_j$ and let $1 \leq \ell \leq k$ be the smallest index for which $\theta_{\min}(R_\ell) = \theta_{\min}(T)$ holds. Further, let $x_1(\theta) = (R_1 + \theta I)^+(-\gamma^0e_1)$ and let $(x_1, \lambda_1^*)$ be a KKT-tuple corresponding to a global minimum of $\text{TR}(R_1, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$, $x_1 = x_1(\lambda_1^*)$.

If $\lambda_1^* \geq -\theta_{\min}$, then $x^* = (x_1(-\theta_{\min})^T, 0, \ldots, 0)^T$ satisfies Thm. 4 for $\text{TR}(T, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$.

If $\lambda_1^* < -\theta_{\min}$, then $x^* = (x_1(-\theta_{\min})^T, 0, \ldots, 0, v^T, 0, \ldots, 0)^T$, with $v \in \text{Eig}(R_\ell, \theta_{\min})$ such that $\|x^*\|_2^2 = \|x_1(-\theta_{\min})\|_2^2 + \|v\|_2^2 = \Delta^2$ satisfies Thm. 4 for $\text{TR}(T, \gamma^0e_1, I, \Delta, \mathbb{R}^{i+1})$.  

7
In particular, as long as $T$ is irreducible, the hard case does not occur. For the tridiagonal matrices arising from Krylov subspace iterations, this is the case as long as the pL process does not break down.

3.3 Solving the Tridiagonal Subproblem in the Easy Case

Assume that $T$ is irreducible, and thus satisfies the easy case. Solving the tridiagonal subproblem amounts to checking whether the problem admits an interior solution and, if not, to finding a value $\lambda^* \geq \max\{0, -\theta_{\min}\}$ with $\|x(\lambda^*)\| = \Delta$.

We follow Moré and Sorensen [35], who define $\sigma_p(\lambda) := \|x(\lambda)\|^p - \Delta^p$ and propose the Newton iteration

$$
\lambda^{i+1} \leftarrow \lambda^i - \sigma_p(\lambda^i)/\sigma'_p(\lambda^i) = \lambda^i - \frac{\|x(\lambda^i)\|^p - \Delta^p}{p\|x(\lambda^i)\|^{p-2}\langle x(\lambda^i), x'(\lambda^i) \rangle}, \ i \geq 0,
$$

with $x'(\lambda) = -(T + \lambda I)^+x(\lambda)$, to find a root of $\sigma_{-1}(\lambda)$. Provided that the initial value $\lambda^0$ lies in the interval $[\max\{0, -\theta_{\min}\}, \lambda^*]$, such that $(T + \lambda^0 I)$ is positive semidefinite, $\|x(\lambda^0)\| \geq \Delta$, and no safeguarding of the Newton iteration is necessary, it can be shown that this leads to a sequence of iterates in the same interval that converges to $\lambda^*$ at globally linear and locally quadratic rate, cf. [19].

Note that $\lambda^* > -\theta_{\min}$ as $\sigma_{-1}(\lambda)$ has a singularity in $-\theta_{\min}$ but $\sigma_{-1}(\lambda^*) = 1/\Delta$ and it thus suffices to consider $\lambda > \max\{0, -\theta_{\min}\}$.

Both the function value and derivative require the solution of a linear system of the form $(T + \lambda I)w = b$. As $T + \lambda I$ is tridiagonal, symmetric positive definite, and of reasonably small dimension, it is computationally feasible to use a tridiagonal Cholesky decomposition for this.

Gould et al. in [21] improve upon the convergence result by considering higher order Taylor expansions of $\sigma_p(\lambda)$ and values $p \neq -1$ to obtain a method with locally quartic convergence.

3.4 The Newton initializer

Cheap oracles for a suitable initial value $\lambda^0$ may be available, including, for example, zero or the value $\lambda^*$ of the previous iteration of the pL process. If these fail, it becomes necessary to compute $\theta_{\min}$. To this end, we follow Gould et al. [19] and Parlett and Reid [40], who define the Parlett-Reid Last-Pivot function $d(\theta)$:

**DEFINITION 9** (Parlett-Reid Last-Pivot Function)

$$
d(\theta) := \begin{cases} 
d_i, & \text{if there exists } (d_0, \ldots, d_i) \in (0, \infty)^i \times \mathbb{R}, \text{ and } L \text{ unit lower triangular such that } T - \theta I = L \text{diag}(d_0, \ldots, d_i) L^T \\
-\infty, & \text{otherwise.}
\end{cases}
$$

Since $T$ is irreducible, its eigenvalues are simple [18, Thm 8.5.1] and $\theta_{\min}$ is given by the unique value $\theta \in \mathbb{R}$ with $T - \theta I$ singular and positive semidefinite, or, equivalently, $d(\theta) = 0$.

A safeguarded root-finding method is used to determine $\theta_{\min}$ by finding the root of $d(\theta)$. An interval of safety $[\theta_l, \theta_u]$ is used in each iteration and a guess $\theta^k \in [\theta_l^k, \theta_u^k]$ is chosen. Gershgorin bounds may be used to provide an initial interval [18, Thm 7.2.1]. Depending
on the sign of \( d(\theta) \) the interval of safety is then contracted to \([\theta^k_l, \theta^k_u]\) if \( d(\theta^k) < 0 \) and to \([\theta^k_u, \theta^k_l]\) if \( d(\theta^k) \geq 0 \) as the interval of safety for the next iteration. One choice for \( \theta^k \) is bisection. Newton steps as previously described may be taken advantage of if they remain inside the interval of safety.

![Figure 1](image_url)

**Figure 1.** The Parlett-Reid last-pivot function \( d(\theta) \) and the lifted function \( \hat{d}(\theta) \) have the common zero \( \theta_{\min} \). Dashed lines show the analytic continuation of the right hand side of \( d(\theta) = \prod_j (\theta - \theta_j) / \prod_j (\theta - \hat{\theta}_j) \) into the region where \( d(\theta) = -\infty \).

For successive pL iterations, the fact that the tridiagonal matrices grow by one column and row in each iteration may be exploited to save most of the computational effort involved. As noted by Parlett and Reid [40], the recurrence to compute the \( d_i \) in Def. 9 is strongly related to the one that results from applying a Laplace expansion for tridiagonal matrices [18, §2.1.4], and yields the explicit formula

\[
d(\theta) = \frac{\text{det}(T - \theta I)}{\text{det}(\hat{T} - \theta I)} = \frac{\prod_j (\theta - \theta_j)}{\prod_j (\theta - \hat{\theta}_j)},
\]

where \( \hat{T} \) denotes the principal submatrix of \( T \) obtained by erasing the last column and row, and \( \theta_j \) and \( \hat{\theta}_j \) enumerate the eigenvalues of \( T \) and \( \hat{T} \), respectively.

It becomes apparent that \( d(\theta) \) has a pole of first order in \( \hat{\theta}_{\min} \). After lifting this pole, the function \( \hat{d}(\theta) := (\theta - \hat{\theta}_{\min})d(\theta) \) is smooth on a larger interval. When iteratively constructing the tridiagonal matrices in successive pL iterations, the value \( \hat{\theta}_{\min} \) is readily available and it becomes preferable to use \( \hat{d}(\theta) \) instead of \( d(\theta) \) for root finding.

### 3.5 Solving the Tridiagonal Subproblem in the Hard Case

If the hard case is present, the decomposition of \( T \) into irreducible components has to be determined. This is given in a natural way by Lanczos breakdown. Every time the Lanczos process breaks down and is restarted with a vector \( M \)-orthogonal to the previously considered Krylov subspaces, a new tridiagonal block is obtained. Solving the problem in the hard case then amounts to applying Theorem 8: First all smallest eigenvalue \( \theta_i \) of the irreducible blocks \( R_i \) have to be determined as well as the KKT tuple \((x^*_1, \lambda^*_1)\) by solving the easy case for \( \text{TR}(R_1, \gamma^0 e_1, I, \Delta, R^+) \). Again, let \( \ell \) be the
smallest index \(i\) with minimal \(\theta_i\). In the case \(\lambda^*_1 \geq -\theta_\ell\), the global solution is given by \(x^* = (x_1^T, 0, \ldots, 0)^T\). On the other hand if \(\lambda^*_1 < -\theta_\ell\) the eigenspace of \(R_\ell\) corresponding to \(\theta_\ell\) has to be obtained. As \(R_\ell\) is irreducible, all eigenvalues of \(R_\ell\) are simple and an eigenvector \(\tilde{v}\) spanning the desired eigenspace can be obtained for example by inverse iteration [18, §8.2.2]. The solution is now given by \(x^* = (x_1(-\theta_\ell)^T, 0, v^T, 0)^T\) with \(x_1(-\theta_{\text{min}}) = (R_1 - \theta(I) - \gamma_0 e_1)^{-1}\) and \(v := \alpha \tilde{v}\) where \(\alpha\) has been chosen as the root of the scalar quadratic equation \(\Delta^2 = \|x_1(-\theta_{\text{min}})\|^2 + \alpha^2 \|\tilde{v}\|^2\) that leads to the smaller objective value.

4. Implementation trlib

In this section, we present details of our implementation trlib of the GLTR method.

4.1 Existing Implementation

The GLTR reference implementation is the software package GLTR in the optimization library GALAHAD [17]. This Fortran 90 implementation uses conjugate gradient iterations exclusively to build up the Krylov subspace, and provides a reverse communication interface that requires exchange vector data to be stored as contiguous arrays in memory.

4.2 trlib Implementation

Our implementation is called trlib, short for trust region library. It is written in plain ANSI C99 code, and has been made available as open source [31]. We provide a reverse communication interface in which only scalar data and requests for vector operations are exchanged, allowing for great flexibility in applications.

Beside the stable and efficient conjugate gradient iteration we also implemented the Lanczos iteration and a crossover mechanism to expand the Krylov subspace, as we frequently found applications in the context of constrained optimization with an SLEQP algorithm [4, 29] where conjugate gradient iterations broke down whenever directions of tiny curvature have been encountered.

4.3 Vector Free Reverse Communication Interface

The implementation is built around a reverse communication calling paradigm. To solve a trust region subproblem, the according library function has to be repeatedly called by the user and after each call the user has to perform a specific action indicated by the value of an output variable. Only scalar data representing dot products and coefficients in axpy operations as well as integer and floating point workspace to hold data for the tridiagonal subproblems is passed between the user and the library. In particular, all vector data has to be managed by the user, who must be able to compute dot products \(\langle x, y \rangle\), perform \(\text{axpy } y := \alpha x + y\) on them and implement operator vector products \(x \mapsto Hx, x \mapsto M^{-1}x\) with the Hessian and the preconditioner.

Thus no assumption about representation and storage of vectorial data is made, as well as no assumption on the discretization of \(\mathcal{H}\) if \(\mathcal{H}\) is not finite-dimensional. This is beneficial in problems arising from optimization problems stated in function space that may not be stored naturally as contiguous vectors in memory or where adaptivity regarding the discretization may be used along the solution of the trust region subproblem. It also
gives a trivial mechanism for exploiting parallelism in vector operations as vector data may be stored and operations may be performed on GPU without any changes in the trust region library.

In particular, this interface allows for easy interfacing with the PDE-constrained optimization software DOLFIN-adjoint \cite{15, 16} within the finite element framework FEniCS \cite{2, 3, 32} without having to rely on assumptions how the finite element discretization is stored, see §5.2.

4.4 Conjugate Gradient Breakdown

Per default, conjugate gradient iterations are used to build the Krylov subspace. The algorithm switches to Lanczos iterations if the magnitude of the curvature $|\langle \hat{p}, H\hat{p} \rangle| \leq \text{tol}_\text{curvature}$ with a user defined tolerance $\text{tol}_\text{curvature} \geq 0$.

4.5 Easy Case

In the easy case after the Krylov space has been assembled in a particular iteration it remains to solve $\text{TR}(T_i, \gamma^{i0}e_1, I, \Delta, \mathbb{R}^{i+1})$ which we do as outlined in §3.3. As mentioned there, an improved convergence order can be obtained by higher order Taylor expansions of $\sigma_p(\lambda)$ and values $p \neq -1$, see \cite{21}. However in our cases the computational cost for solving the tridiagonal subproblem — often warmstarted in a suitable way — is negligible in comparison the the cost of computing matrix vector products $x \mapsto Hx$ and thus we decided to stick to the simpler Newton rootfinding on $\sigma_{-1}(\lambda)$.

To obtain a suitable initial value $\lambda^0$ for the Newton iteration, we first try $\lambda^*$ obtained in the previous Krylov iteration if available and otherwise $\lambda^0 = 0$. If these fail, we use $\lambda^0 = -\theta_{\min}$ computed as outlined in §3.4 by zero-finding on $d(\theta)$ or $\hat{d}(\theta)$. This requires suitable models for $\hat{d}(\theta)$. Gould et al. \cite{19} propose to use a quadratic model $\theta^2 + a\theta + b$ for $\hat{d}(\theta)$ that captures the asymptotics $t \to -\infty$ obtained by fitting function value and derivative in a point in the root finding process. We have also had good success with the linear Newton model $a\theta + b$, and with using a second order quadratic model $a\theta^2 + b\theta + c$, that makes use of an additional second derivative, as well. Derivatives of $d(\theta)$ or $\hat{d}(\theta)$ are easily obtained by differentiating the recurrence for the Cholesky decomposition. In our implementation a heuristic is used to select the option that is inside the interval of safety and promises good progress. The heuristic is given by using $\theta^2 + a\theta + b$ in case that the bracket width $\theta^k_u - \theta^k_l$ satisfies $\theta^k_u - \theta^k_l \geq 0.1 \max\{|\theta^k|\}$ and $a\theta^2 + b\theta + c$ otherwise.

The motivation behind this is that in the former case it is not guaranteed, that $\theta^k$ has been determined to high accuracy and thus the model that captures the global behaviour might be better suited while in the latter case $\theta^k$ to a certain accuracy and it is safe to the model representing local behaviour.

4.6 Hard Case

4.6.1 Exact Hard Case

The function for the solution of the tridiagonal subproblem implements the algorithm as given by Theorem 8 if provided with a decomposition in irreducible blocks. However, from local information it is not possible to distinguish between convergence to a global solution of the original problem and the case in which an invariant Krylov subspace is exhausted that may not contain the global minimizer as in both cases the
gradient vanishes.

The handling of the hard case is thus left to the user who has to decide in the reverse calling scheme if once arrived at a point where the gradient norm is sufficiently small the solution in the Krylov subspaces investigated so far or further Krylov subspaces should be investigated. In that case it is left to the user to determine a new nonzero initial vector for the Lanczos iteration that is $M$-orthogonal to the previous Krylov subspaces.

### 4.6.2 Near Hard Case

The near hard case arises if $\langle \gamma^0 e_1, \tilde{v} \rangle$ is tiny, where $\tilde{v}$ spans the eigenspace $\text{Eig}(\theta_{\text{min}}) = \text{span}\{\tilde{v}\}$.

Numerically this is detected if there is no $\lambda \geq \max\{0, -\theta_{\text{min}}\}$ such that $\|x(\lambda)\| \geq \Delta$ holds in floating point arithmetic. In that case we use the heuristic $\lambda^* = -\theta_{\text{min}}$ and $x^* = x(-\theta_{\text{min}}) + \alpha v$ with $v \in \text{Eig}(\theta_{\text{min}})$ where $\alpha$ is determined such that $\|x^*\| = \Delta$.

Another possibility would be to modify the tridiagonal matrix $T$ by dropping offdiagonal elements below a specified threshold and work on the obtained decomposition into irreducible blocks. However we have not investigated this possibility as the heuristic seems to deliver satisfactory results in practice.

### 4.7 Reentry with New Trust Region Radius

In nonlinear programming applications it is common that after a rejected step another closely related trust region subproblem has to be solved with the only changed data being the trust region radius. As this has no influence on the Krylov subspace but only on the solution of the tridiagonal subproblem, efficient hotstarting has been implemented. Here the tridiagonal subproblem is solved again with exchanged radius and termination tested. If this point does not satisfy the termination criterion, conjugate gradient or Lanczos iterations are resumed until convergence. However, we rarely observed the need to resume the Krylov iterations in practice.

### 4.8 Termination criterion

Convergence is reported as soon as the Lagrangian gradient satisfies

$$
\|\nabla L\|_{M^{-1}} \leq \begin{cases} 
\max\{\text{tol}_\text{abs}_i, \text{tol}_\text{rel}_i \|g\|_{M^{-1}}\}, & \text{if } \lambda = 0 \\
\max\{\text{tol}_\text{abs}_b, \text{tol}_\text{rel}_b \|g\|_{M^{-1}}\}, & \text{if } \lambda > 0 
\end{cases}
$$

The rationale for using possibly different tolerances in the interior and boundary case is motivated from applications in nonlinear optimization where trust region subproblems are used as globalization mechanism. There a local minimizer of the nonlinear problem will be an interior solution to the trust region subproblem and it is thus not necessary to solve the trust region subproblem in the boundary case to highest accuracy.

### 4.9 TRACE

In the recently proposed TRACE algorithm [10], trust region problems are also used. In addition to solving trust region problems, the following operations have to be performed:

- $\min_x \frac{1}{2} \langle x, (H + \lambda M)x \rangle + \langle g, x \rangle$, 

• Given constants $\sigma_l, \sigma_u$ compute $\lambda$ such that the solution point of $\min_x \frac{1}{2} \langle x, (H + \lambda M)x \rangle + \langle g, x \rangle$ satisfies $\sigma_l \leq \lambda \| x \| M \leq \sigma_u$.

These operations have to be performed after a trust region problem has been solved and can be efficiently implemented using the Krylov subspaces already built up.

We have implemented these as suggested in [10], where the first operation requires one backsolve with tridiagonal data and the second one is implemented as root finding on $\lambda \mapsto \frac{\lambda}{\| x(\lambda) \|} - \sigma$ with a certain $\sigma \in [\sigma_l, \sigma_u]$ that is terminated as soon as $\frac{\lambda}{\| x(\lambda) \|} \in [\sigma_l, \sigma_u]$.

4.10 C11 Interface

The algorithm has been implemented in C11. The user is responsible for holding vector-data and invokes the algorithm by repeated calls to the function `trlib_krylov_min` with integer and floating point workspace and dot products $\langle v, g \rangle$, $\langle p, Hp \rangle$ as arguments and in return receives status informations and instructions to be performed on the vectorial data. A detailed reference is provided in the Doxygen documentation to the code.

4.11 Python Interface

A low-level python interface to the C library has been created using Cython that closely resembles the C API and allows for easy integration into more user-friendly, high-level interfaces.

As a particular example, a trust region solver for PDE-constrained optimization problems has been developed to be used from DOLFIN-adjoint [15, 16] within FEniCS [2, 3, 32]. Here vectorial data is only considered as FEniCS-objects and no numerical data except of dot products is used of these objects.

5. Numerical Results

In this section, we present an assessment of the computational performance of our implementation `trlib` of the GLTR method, and compare it to the reference implementation GLTR as well as several competing methods for solving the trust region problem and their respective implementations.

5.1 Generation of Trust-Region Subproblems

For want of a reference benchmark set of non-convex trust region subproblems, we resorted to the subset of unconstrained nonlinear programming problems of the CUTEst benchmark library, and use a standard trust region algorithm, e.g. Gould et al. [19], for solving $\min_{x \in \mathbb{R}^n} f(x)$, as a generator of trust-region subproblems. The algorithm starts from a given initial point $x^0 \in \mathbb{R}^n$ and trust region radius $\Delta^0 > 0$, and iterates for $k \geq 0$:

In a first study, we compared our implementation `trlib` of the GLTR method to the reference implementation GLTR as well as several competing methods for solving the trust region problem, and their respective implementations, as follows:

• GLTR [19] in the GALAHAD library implements the GLTR method.
• LSTRS [46] uses an eigenvalue based approach. The implementation uses MATLAB and makes use of the direct ARPACK [28] reverse communication interface, which is deprecated in recent versions of MATLAB and lead to crashes within MATLAB 2013b used by
**Algorithm 4:** Standard trust region algorithm for unconstrained nonlinear programming, used to generate trust region subproblems from CUTEst.

**Input:** \( f, x^0, \Delta^0, \rho_{\text{acc}}, \rho_{\text{inc}}, \gamma^+, \gamma^- \), \( \text{tol}_{\text{abs}} \)

**Output:** \( k, x^k \)

**Algorithm:**

For \( k \geq 0 \) do

1. Evaluate \( g^k := \nabla f(x^k) \)
2. Test for termination: Stop if \( \|g^k\| \leq \text{tol}_{\text{abs}} \)
3. Evaluate \( H^k := \nabla^2 f(x^k) \)
4. Compute (approximate) minimizer \( d^k \) to \( \text{TR}(H^k, g^k, I, \Delta^k) \)
5. Assess the performance \( \rho^k := (f(x^k + d^k) - f(x^k))/q(d^k) \) of the step
6. Update step: \( x^{k+1} := \begin{cases} x^k + d^k, & \rho^k \geq \rho_{\text{acc}} \\ x^k, & \rho^k < \rho_{\text{acc}} \end{cases} \)
7. Update trust region radius: \( \Delta^{k+1} := \begin{cases} \gamma^+ \Delta^k, & \rho^k \geq \rho_{\text{inc}} \\ \Delta^k, & \rho_{\text{acc}} \leq \rho^k < \rho_{\text{inc}} \\ \gamma^- \Delta^k, & \rho^k < \rho_{\text{acc}} \end{cases} \)

end

We thus resorted to the standard `eigs` eigenvalue solver provided by MATLAB which might severely impact the behaviour of the algorithm.

- **SSM** [22] implements a sequential subspace method that may use an SQP accelerated step.
- **ST** is an implementation of the truncated conjugate gradient method proposed independently by Steihaug [48] and Toint [49].
- **trlib** is our implementation of the GLTR method.

All codes, with the exception of **LSTRS**, have been implemented in a compiled language, Fortran 90 in case of GLTR and C in for all other codes, by their respective authors. **LSTRS** has been implemented in interpreted MATLAB code. The benchmark code used to run this comparison has also been made open source and is available as **trbench** [30].

In our test case the parameters \( \Delta^0 = 1/\sqrt{n} \), \( \text{tol}_{\text{abs}} = 10^{-7} \), \( \rho_{\text{acc}} = 10^{-2} \), \( \rho_{\text{inc}} = 0.95 \), \( \gamma^+ = 2 \) and \( \gamma^- = \frac{1}{2} \) have been used. We used the standard termination criteria of the subproblem solver where we chose \( \text{tol}_{\text{abs}, i} = \text{tol}_{\text{abs}, b} = 0 \) and \( \text{tol}_{\text{rel}, i} = 10^{-8} \) and \( \text{tol}_{\text{rel}, b} = 10^{-5} \) in the **trlib** implementation of GLTR.

The performance of the different algorithms is assessed using extended performance profiles as introduced by [12, 33], for a given set \( S \) of solvers and \( P \) of problems the performance profile for solver \( s \in S \) on problem \( p \in P \) is defined by

\[
\rho_s(\tau) := \frac{1}{|P|} |\{ p \in P | r_{s,p} \leq \tau \}|,
\]

where \( r_{s,p} = \frac{t_{s,p}}{\min_{\sigma \in S, \sigma \neq s} t_{\sigma,p}} \) performance of \( s \in S \) on \( p \in P \).

It can be seen that **GLTR** and **trlib** are the most robust solvers on the subset of unconstrained problems from CUTEst in the sense that they eventually solve the largest fraction of problems among all solvers and that they are also among the fastest solvers. That **GLTR** and **trlib** show similar performance is to be expected as they implement the identical GLTR algorithm, where **trlib** is slightly more robust and faster. We attribute this to the implementation of efficient hotstart capabilities and also the Lanczos process to build up the Krylov subspaces once directions of zero curvature are encountered.
5.2 Function Space Problem

We solved a modified variant of \( \text{SCDIST1} \) [7, 34] of the \( \text{OPTPDE} \) benchmark library [38, 39] for PDE constrained optimal control problems. The state constraint has been dropped and a trust region constraint added in order to obtain the following function space trust region problem:

\[
\min_{y \in H^1(\Omega), u \in L^2(\Omega)} \frac{1}{2} \| y - y_d \|_{L^2(\Omega)}^2 + \frac{\beta}{2} \| u - u_d \|_{L^2(\Omega)}^2 \\
\text{s.t.} \quad -\Delta y + y = u, \quad x \in \Omega \\
\quad \quad \quad \partial_n y = 0, \quad x \in \partial \Omega \\
\quad \quad \quad \| y \|_{L^2(\Omega)}^2 + \| u \|_{L^2(\Omega)}^2 \leq \Delta^2
\]

Here \( \Omega \subseteq \mathbb{R}^n \), \( L^2(\Omega) \) denotes the Lebesgue space of square integrable functions \( f : \Omega \to \mathbb{R} \), \( H^1(\Omega) \) the sobolev space of square integrable functions that admit a square integrable weak derivative and \( \Delta : H \) is the Laplace operator \( \Delta = \sum_{i=1}^n \partial_{x_i}^2 \).

Tracking data \( y_d, u_d \) has been used as specified in \( \text{OPTPDE} \) where typical regularization parameters have been considered in the range \( 10^{-8} \leq \beta \leq 10^{-3} \). Different geometries \( \Omega \in \{ (0,1)^2, (0,1)^3, \{ x \in \mathbb{R}^2 \mid \| x \| \leq 1 \}, \{ x \in \mathbb{R}^3 \mid \| x \| \leq 1 \} \} \) have been studied.

The finite element software \( \text{FEnICS} \) has been used to obtain a finite element discretiza-
tion of the problem:

\[
\begin{align*}
\min_{y \in \mathbb{R}^{n_y}, u \in \mathbb{R}^{n_u}} & \quad \frac{1}{2} \|y - y_d\|_M^2 + \frac{\beta}{2} \|u - u_d\|_M^2 \\
\text{s.t.} & \quad Ay - Mu = 0, \\
& \quad \|y\|_M^2 + \|u\|_M^2 \leq \Delta^2,
\end{align*}
\]

where \(M\) denotes the mass matrix and \(A = K + M\) with \(K\) being the stiffness matrix.

We used the approach suggested by Gould et al. [20] to solve this equality constrained trust region problem:

1. A null-space projection in the preconditioning step of the Krylov subspace iteration is used to satisfy the discretized PDE constraint. The required preconditioner is given by

\[
\begin{pmatrix}
    y \\
    u
\end{pmatrix}
\rightarrow
\begin{pmatrix}
    I & 0 & 0 \\
    0 & I & 0
\end{pmatrix}
\begin{pmatrix}
    M & 0 & A \\
    0 & M & -M \\
    A & -M & 0
\end{pmatrix}
^{-1}
\begin{pmatrix}
    I & 0 \\
    0 & I \\
    0 & 0
\end{pmatrix}
\begin{pmatrix}
    y \\
    u
\end{pmatrix}.
\]

2. We used MINRES [41] for solving with the linear system arising in this preconditioner to high accuracy. MINRES iterations themselves are preconditioned using the approximate Schur-complement preconditioner

\[
\begin{pmatrix}
    \tilde{M} & \\
    \tilde{M} & \tilde{A}M^{-1}\tilde{A}
\end{pmatrix}
^{-1},
\]

as proposed by [42]. This preconditioner is an approximation to the optimal preconditioner

\[
\begin{pmatrix}
    M & \\
    M & AM^{-1}A + M
\end{pmatrix}
^{-1}
\]

that would lead to mesh-independent MINRES convergence in three iterations, provided exact arithmetic [27, 36] would be used.

3. In the MINRES preconditioner of step (2), products with \(\tilde{M}^{-1}\) and \(\tilde{A}^{-1}\) are computed using truncated conjugate gradients (CG) to high accuracy, again preconditioned using an algebraic multigrid as preconditioner.

In Fig. 3, it can be seen that using the GLTR method for these function space problems yields a solver with mesh-independent convergence behavior. The number of outer iterations is virtually constant on a wide range of different meshes and varies at most by one iteration. The number of inner (MINRES) iterations varies only slightly, as is to be expected due to the use of an approximately optimal preconditioner in step (2).

6. Conclusion

We presented trlib which implements Gould’s Generalized Lanczos Method for trust region problems. Distinct features of the implementation are by the choice of a reverse
Figure 3. Results for distributed control trust region problem for different mesh sizes. Results are shown for four different geometries. Regularization parameters $\beta \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$ have been considered, however computational results for a fixed geometry hardly change with $\beta$ leading to near-identical plots.

communication interface that does not need access to vector data but only to dot products between vectors and by the implementation of preconditioned Lanczos iterations to build up the Krylov subspace. The package \texttt{trbench}, which relies on \texttt{CUTEst}, has been introduced as a test bench for trust region problem solvers. Our implementation \texttt{trlib} shows similar and favorable performance in comparison to the \texttt{GLTR} implementation of the Generalized Lanczos Method and also in comparison to other iterative methods for solving the trust region problem.

Moreover, we solved an example from PDE constrained optimization to show that the implementation can be used for problems stated in Hilbert space as a function space solver with almost discretization independent behaviour in that example.

Funding

References


