Two-Level Trust-Region Method with Random Subspaces

Andrea Angino[0009–0000–8525–375X], Alena Kopaničáková[0000–0001–8388–5518], and Rolf Krause[0000–0001–5408–5271]

1 Introduction

We consider minimization problems of the following type:

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1}$$

where f is the twice differentiable objective function (bounded from below) and n possibly very large. In practice, optimization problems of this type are often solved using trust-region (TR) methods [2], which provide global convergence also in the presence of non-convexity. TR methods generate search directions by building a local quadratic model of the objective function and minimizing this model while adhering to a step-size constraint at each iteration. The acceleration of TR methods through multilevel approaches (Recursive Multilevel Trust Region or RMTR) was considered in [7, 8, 11, 10, 12, 5, 6]. These methods aim to reduce the computational cost of TR methods while ensuring their algorithmic scalability. This is achieved by exploring computationally cheaper, albeit less accurate, models of f on "coarser" levels, in addition to the standard quadratic models used on the original "fine" level.

In this work, we introduce a new approach to construct the search directions in the TR algorithm, giving rise to a novel two-level TR (TLTR) method. Methodologically, our TLTR method is built upon the "magic" TR framework [2, Section 10.4.1.], which assumes the existence of an *oracle/magic* that is used to improve the search directions obtained by minimizing the quadratic models. This framework has been successfully explored in the literature in the context of constrained [3] and non-smooth optimization problems [4, 18]. Here, we extend it to unconstrained optimization problems by identifying the *oracle/magic* steps with the *coarse-level/subspace*

Andrea Angino
Università della Svizzera italiana, UniDistance Suisse; e-mail: andrea.angino@unidistance.ch

Alena Kopaničáková
Toulouse INP-Enseeiht, IRIT, ANITI; e-mail: alena.kopanicakova@toulouse-inp.fr

Rolf Krause
Università della Svizzera italiana, UniDistance Suisse, King Abdullah University of Science and Technology; e-mail: rolf.krause@kaust.edu.sa

steps, commonly utilized in multilevel/subspace optimization. Thus, in each iteration, our TLTR method utilizes a composite search direction consisting of a step obtained by minimizing the quadratic model and a step obtained by minimizing a coarse-level/subspace model. This approach conceptually differs from RMTR methods [7], which alternate between these two search directions instead of composing them.

From the application point of view, we aim to apply our TLTR method to the training of the machine learning models. To this aim, we construct coarse-level/subspace models using randomly generated subspaces obtained through sketching [9]. It is worth noting that while sketching was initially proposed to reduce data dimensionality while controlling information loss, it has recently been also employed to reduce the computational cost of optimization methods. This has led to the development of several sketched first/second-order optimization methods, e.g., sketched gradient descent [14], Newton methods [15, 1] or subspace methods [17]. To the best of our knowledge, a majority of the existing sketched optimization algorithms rely solely on sketched models throughout every iteration. This imposes restrictions on the minimal size of the sketched subspace, as it directly affects the quality of the resulting search directions. Unlike traditional sketched methods that rely solely on reduced models, our approach embodies the *divide–conquer–combine* paradigm. We divide the full space into subspaces using sketching, conquer the problem within each subspace, and then combine both coarse (subspace) and fine (full-space) search directions to update the solution. Thus, our TLTR method uses sketching solely to enhance full-space computations; consequently, the size of the sketched subspace affects only the algorithm's speed, not its convergence.

This manuscript is organized as follows. In Section 2, we introduce the proposed TLTR method. In Section 3, we demonstrate the effectiveness of the TLTR method using numerical examples from the field of machine learning.

2 Two-level TR (TLTR) with random subspaces

In this section, we present the novel TLTR algorithm, which can be seen as the TR method that utilizes the composite search direction to find a solution of (1). Motivated by the multigrid methods [16], the composite direction is constructed by combining a step performed on the full space, called a smoothing step, and the step performed on the subspace called a coarse-level/subspace step.

2.1 The TLTR algorithm

Utilizing the magical TR framework [2], the TLTR method starts with an initial guess $x_0 \in \mathbb{R}^n$. At each k^{th} iteration, the algorithm first performs a smoothing, achieved by approximately solving the following quadratic minimization problem (QP):

$$\begin{aligned} \min_{p_k^F \in \mathbb{R}^n} \; & m_k^F\left(p_k^F\right) := f(x_k) + \left\langle \nabla f\left(x_k\right), p_k^F \right\rangle + \frac{1}{2}\left\langle p_k^F, \nabla^2 f(x_k) p_k^F \right\rangle, \\ \text{subject to} \quad & \left\| p_k^F \right\| \leq \Delta_k, \end{aligned} \tag{2}$$

where the model m_k^F is constructed by utilizing a second order Taylor approximation of f around current iterate x_k and the symbol $\Delta_k \in \mathbb{R}^+$ denotes a trust-region radius.

After the smoothing step is performed, the algorithm proceeds to obtain the subspace step. To this aim, we consider an iteration-dependent restriction operator $S_k \in \mathbb{R}^{\ell \times n}$ and the prolongation operator $S_k^T \in \mathbb{R}^{n \times \ell}$, where $\ell \ll n$. These transfer operators are assembled using the sketching techniques [9]. The algorithm then obtains the subspace step $p_k^S \in \mathbb{R}^\ell$ by taking the advantage of a model m_k^S of f, constructed around the iterate $x_{k+1/2} := x_k + p_k^F$, i.e., the iterate obtained after the smoothing step. Notably, the model m_k^S can be constructed using various techniques, e.g., following $1^{\text{st}}/2^{\text{nd}}$-order consistency approaches outlined in [13]. For the purpose of this work, we utilize a simple restriction of the quadratic approximation of f to a subspace induced by S_k. Thus, $p_k^S \in \mathbb{R}^\ell$ is obtained by solving the following sketched QP problem:

$$\min_{p_k^S \in \mathbb{R}^\ell} m_k^S\left(p_k^S\right) := f(x_{k+1/2}) + \left\langle S_k \nabla f(x_{k+1/2}), p_k^S \right\rangle + \frac{1}{2}\left\langle p_k^S, S_k \nabla^2 f(x_{k+1/2}) S_k^T p_k^S \right\rangle,$$
$$\text{subject to} \quad \left\| p_k^S \right\| \leq \Delta_k, \tag{3}$$

where the TR radius Δ_k again controls the size of p_k^S. This algorithmic design follows closely the RMTR algorithm [13], where the size of the coarse-level correction is also controlled by means of Δ_k.

Subsequently, the subspace search direction p_k^S is transferred back to the full space. However, the prolongated search direction $S_k^T p_k^S$ is not automatically considered by the TLTR algorithm, but only if it provides a decrease in the objective function, thus only if

$$f\left(x_k + p_k^F + \alpha_k S_k^T p_k^S\right) \leq f\left(x_k + p_k^F\right). \tag{4}$$

Otherwise, the subspace search direction is discarded, i.e., we set p_k^S to be the zero vector. Note, in order to improve quality of $S_k^T p_k^S$, one can utilize $\alpha_k \in \mathbb{R}^+$, obtained using a line-search strategy, such that $f(x_k + p_k^F + \alpha_k S_k^T p_k^S) < f(x_k + p_k^F + S_k^T p_k^S)$.

Finally, the TLTR algorithm assesses the quality of the composite search direction $p_k := p_k^F + \alpha_k S_k^T p_k^S$ by means of the TR ratio ϱ_k, defined as

$$\varrho_k := \frac{f(x_k) - f\left(x_k + p_k^F + \alpha_k S_k^T p_k^S\right)}{m_k^F(x_k) - m_k^F\left(x_k + p_k^F\right) + f\left(x_k + p_k^F\right) - f\left(x_k + p_k^F + \alpha_k S_k^T p_k^S\right)}. \tag{5}$$

Here, the numerator measures the objective function change before and after taking the composite step p_k. The denominator measures the change in the model m_k^F after the smoothing step p_k^F as well as the change in the objective function f after performing the subspace step. If $\varrho_k > \eta_1$, where $\eta_1 > 0$, then the composite step p_k decreases the objective function sufficiently and it is safe to utilize it. Otherwise the

Algorithm 1 Two-Level Trust Region (TLTR) Method

Require: $f : \mathbb{R}^n \to \mathbb{R}$, $x_0 \in \mathbb{R}^n$, $\Delta_0 \in \mathbb{R}^+$, $\ell < n \in \mathbb{N}$, $k = 0$

Constants: $0 < \eta_1 \leq \eta_2 < 1,\ 0 < \gamma_1 \leq \gamma_2 < 1$

while not converged **do**

$p_k^F := \underset{\|p_k^F\| \leq \Delta_k}{\operatorname{argmin}}\ m_k^F\left(p_k^F\right)$ ▹ Obtain full-space search direction

$x_{k+1/2} := x_k + p_k^F$

Construct S_k via sketching

$p_k^S := \underset{\|p_k^S\| \leq \Delta_k}{\operatorname{argmin}}\ m_k^S\left(p_k^S\right)$ ▹ Obtain subspace search-direction

▹ Assess the quality of the subspace step

$$p_k^S = \begin{cases} p_k^S, & \text{if } f\left(x_k + p_k^F + \alpha_k S_k p_k^S\right) < f\left(x_k + p_k^F\right) \\ 0, & \text{otherwise} \end{cases}$$

Evaluate ϱ_k as in (5) ▹ Assess the quality of the composite trial step

$$x_{k+1} := \begin{cases} x_k + p_k^F + \alpha_k S_k p_k^S, & \text{if } \varrho_k > \eta_1 \\ x_k, & \text{otherwise} \end{cases} \qquad \Delta_{k+1} := \begin{cases} [\Delta_k, \infty), & \text{if } \varrho_k \geq \eta_2 \\ [\gamma_2 \Delta_k, \Delta_k], & \text{if } \varrho_k \in [\eta_1, \eta_2] \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k], & \text{if } \varrho_k < \eta_2 \end{cases}$$

$k = k + 1$

end while

return x_{k+1}

composite search direction p_k has to be discarded. In addition, the TR radius Δ_k has to be adjusted accordingly, i.e., enlarged, or shrunk depending on the value of ϱ_k.

We remark that if p_k^S is a zero vector, the TLTR algorithm reduces to the standard, single-level, TR algorithm with $\varrho_k^{TR} := \frac{f(x_k) - f\left(x_k + p_k^F\right)}{m_k^F(x_k) - m_k^F\left(x_k + p_k^F\right)}$. Moreover, if $S_k p_k^S$ is a non-zero vector and the condition (4) holds, then $\varrho_k > \varrho_k^{TR}$, whenever $\varrho_k^{TR} < 1$. As a consequence, the composite step p_k might be accepted by the TLTR algorithm even if the standard TR algorithm would have rejected the full-space smoothing step p_k^F. This algorithmic feature allows for the enhanced convergence of the TLTR algorithm compared to the standard TR method. Algorithm 1 summarizes the described procedure.

2.2 The computational cost of the TLTR method

Compared to the standard TR algorithm, one iteration of the TLTR method is computationally more expensive. The increased computational cost is mostly associated with the random subspace minimization step. In particular, we sketch the subspaces by generating $S_k \in \mathbb{R}^{\ell \times n}$ using one of the following two sketching approaches:

- **Gaussian:** All entries of S_k are independently distributed as $\mathcal{N}(0, \ell^{-1})$.
- **S-hashing (sparse):** For each column $j \in \{1, \ldots, n\}$, independently, we uniformly sample $s \ll \ell$ distinct row indices $i_1, \ldots, i_s \in \{1, \ldots, \ell\}$ without replacement. We then assign $(S_k)_{i_k, j} = \pm \frac{1}{\sqrt{s}}$ for $k = 1, \ldots, s$.

The Gaussian approach generates a dense S_k, while s-hashing creates a sparse S_k with s non-zero entries per column. Moreover, s-hashing maintains the sparsity of the matrices it acts upon, allowing for efficient construction of m_k^S and the solution of (3) with a lower computational cost compared to the Gaussian approach.

Notably, obtaining p_k^S is significantly cheaper than p_k^F, as the sketched derivatives can be computed at reduced costs since only their subspace projections are required. Additionally, since $\ell \ll n$, the cost of solving the subspace QP problem (3) is significantly lower than (2). Here, we point out that, for global convergence of the TR/TLTR algorithm, it is sufficient to solve (2) approximately, as long as resulting p_k^F fulfills the sufficient decrease condition, see [2]. Consequently, in our numerical experiments, we solve (2) using computationally inexpensive QP solvers. Specifically, we utilize the Cauchy Point (CP) method, or a few iterations of the Steihaug-Toint conjugate gradient (ST-CG) method [2]. In contrast, there are no requirements on the necessary accuracy of solving (3), as long as the resulting $S_k p_k^S$ satisfies (4). However, due to the small dimensionality of the sketched problem, we opt to solve (3) as accurately as possible by the ST-CG method. This algorithmic design closely mirrors the well-established multigrid approaches [16], where smoothing is typically achieved using computationally inexpensive solution strategies that are particularly effective at reducing high-frequency components of the error. Moreover, it is standard practice to solve the subspace problems accurately due to their small size, which in turn allows the effective elimination of low-frequency components of the error.

3 Numerical examples

We illustrate the numerical performance of the proposed TLTR method using classification problems. In particular, given a data set $D = (z_i, y_i)_{i=1}^N$, we consider

- **logistic loss**, defined as $f_{LL}(x) := \sum_{i=1}^N \log\left(1 + \mathrm{e}^{-y_i \langle x, z_i \rangle}\right) + \frac{\lambda}{2}\|x\|^2$,
- **least-square loss**, defined as $f_{LS}(x) := \frac{1}{N}\sum_{i=1}^N \left(y_i - \frac{\mathrm{e}^{\langle x, z_i \rangle}}{1+\mathrm{e}^{\langle x, z_i \rangle}}\right)^2 + \frac{\lambda}{2}\|x\|^2$,

where $\lambda = \frac{1}{N}$ represents the regularization term. For both problems, we utilize Australian, Mushrooms, and Gisette datasets, from the LIBSVM database[1]. Please, refer to Table 1 for the details regarding the number of samples (N), the parameter dimension (n), and the approximate condition number (κ) of the arising Hessians.

Table 1: Properties of the datasets

Dataset	N	n	κ_{LL}	κ_{LS}
Australian	621	14	10^6	10^4
Mushroom	6,499	112	10^2	10^6
Gisette	6,000	5,000	10^4	10^8

We first investigate the performance of the proposed TLTR method with respect to different sketching strategies. For reference, the TLTR's performance is always compared to the standard TR method (without subspace step). For both methods, the full QP problems are solved using either ST-CG or CP method. Moreover, we also include a comparison with the sketched Newton (SN) method (Newton performed only on sketched subspace) with backtracking line-search, as proposed in [1, Algorithm 1]. In this case, the linear systems arising in each Newton iteration are solved

[1] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

exactly. For all experiments, we utilize random initial guesses and terminate the solution process as soon as $\|\nabla f\| < 10^{-7}$.

Figure 1 illustrates the behavior of the TLTR method for different subspace dimensions ℓ for the logistic loss f_{LL} (first row) and the least square loss f_{LS} (second row), with the Australian and Mushroom datasets. As expected, as the size of ℓ increases, the number of TLTR iterations decreases, but at the same time, the computational cost of solving the sketched QP subproblem increases. As a consequence, sketching subspace such that ℓ equals approximately to $20 - 30\%$ of n strikes an optimal balance between the convergence speedup and the required computational cost. We also see that for least squares problem with the Mushroom dataset significantly small subspaces facilitate the fast convergence. Moreover, we can observe that by utilizing the CP method as the full-space QP solver, the performance of the TR solver rapidly deteriorates. In this case, the speedup obtained by the TLTR method is more prominent due to the use of the second-order information, albeit only within the sketched subspace.

We further investigate the impact of the parameter s, used within the s-hashing strategy, on the performance of the TLTR method. Figure 2 (left) illustrates the obtained results for f_{LL} loss with the Mushroom dataset. As we can see, larger values of s, which give rise to a denser transfer operator, do not give rise to significantly faster convergence. Therefore, for the rest of the numerical experiments, the parameter s is set to $\approx 10\%$ of the subspace parameter ℓ. Figure 2 (right) depicts the results for f_{LS} loss for the TR and TLTR methods with the Gisette dataset over the fixed budget of $5,000$ iterations. As we can observe, the TLTR method enables convergence to a more accurate solution of an order of magnitude.

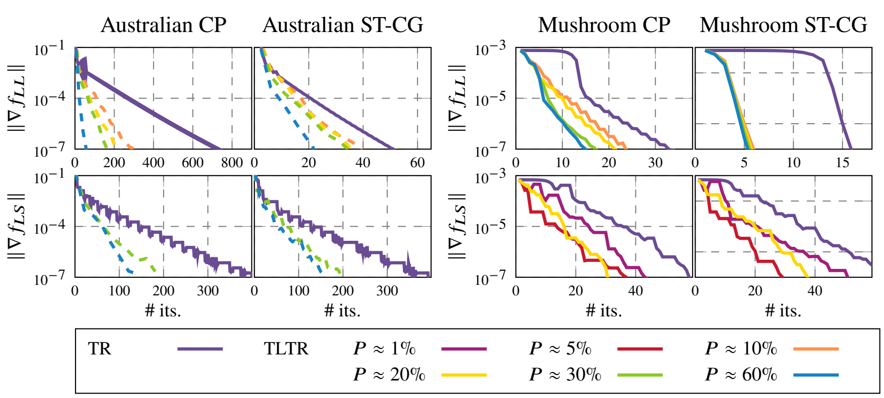

Fig. 1: Convergence history of TR and TLTR for solving (1) with f_{LL} (first row) and f_{LS} (second row) with Gaussian (dashed lines, Australian dataset) and s-hashing (solid lines, Mushroom, $s = \lceil \ell/4 \rceil$) for subspaces of varying size $\ell = \lceil nP \rceil$, where P denotes the portion of the full-space parameters. To solve QP problems on full space, we use 2 iterations of ST-CG, or CP methods.

Figure 3 illustrates the comparison of the TLTR method with respect to the TR and SN methods for logistic regression examples. Following the analysis presented

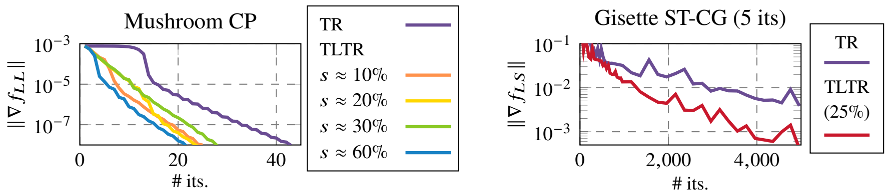

Fig. 2: Left: Convergence of the TLTR method with s-hashing strategy ($\ell = \lceil n/5 \rceil$) for different values of sketching parameter s for logistic loss with Mushroom dataset. Right: Convergence history of TR and TLTR for the least-square loss minimization problem with Gisette dataset.

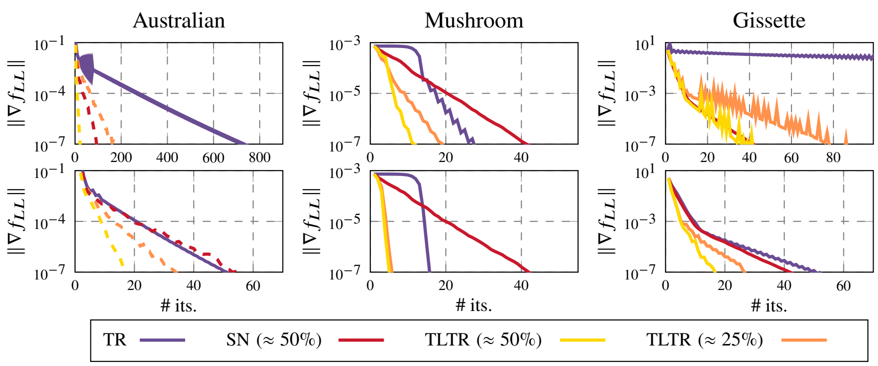

Fig. 3: Convergence history of TR, SN and TLTR with Gaussian (dashed lines, Australian dataset) and s-hashing (solid lines, Gissette/Mushroom) subspaces. The subspace sizes are chosen using a portion of full space n stated in brackets. The QP problems on the full space are solved using CP/ST-CG (2/5 its) methods (Top/Bottom row).

in [1], the size of the sketched subspace for the SN method is set to be fairly large (50% of n). For the TLTR method, we consider the same settings, as well as significantly smaller subspaces (25% of n), guided by our previous experiments. As we can observe from the obtained results, the TLTR method outperforms the TR and SN methods. We attribute this behavior to taking advantage of both full-space and sketched search directions. In contrast, TR takes advantage of only full-space information, while SN utilizes only sketched quantities to obtain a search direction. Moreover, we also note that the observed speedups grow as the problems become larger and more ill-conditioned, i.e., as n and κ grow. This highlights the potential of the proposed TLTR method for solving large-scale problems of practical importance, which we plan to explore in the future work.

Acknowledgements This work was partially supported by the Swiss Platform for Advanced Scientific Computing (PASC) project ExaTrain and by the Swiss National Science Foundation through the projects "ML^2 - Multilevel and Domain Decomposition Methods for Machine Learning" (197041). The work of A.K. benefited from the AI Interdisciplinary Institute ANITI, funded by the France 2030 program under Grant Agreement No. ANR-23-IACL-0002.

References

1. C. Cartis, J. Fiala, and Z. Shao. Randomised subspace methods for non-convex optimization, with applications to nonlinear least-squares, 2022. arXiv e-prints.
2. A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. Society for Industrial and Applied Mathematics, 2000.
3. A. R. Conn, L. N. Vicente, C. A. R. Visweswariah, N. I. M. Gould, and P. L. Toint. Two-step algorithms for nonlinear optimization with structured applications. *SIAM Journal on Optimization*, 9, 1999.
4. R. Fletcher. Second order corrections for non-differentiable optimization. In *Numerical Analysis*, pages 85–114. Pringer Berlin Heidelberg, 1982.
5. S. Gratton, A. Kopaničáková, and Ph. Toint. Multilevel objective-function-free optimization with an application to neural networks training. *SIAM Journal on Optimization*, 33:2772–2800, 2023.
6. S. Gratton, A. Kopaničáková, and Ph. L. Toint. Recursive bound-constrained adagrad with applications to multilevel and domain decomposition minimization, 2025. ArXiv preprint arXiv:2507.11513.
7. S. Gratton, A. Sartenaer, and Ph. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19:414–444, 2008.
8. C. Gross and R. Krause. On the convergence of recursive trust-region methods for multiscale nonlinear optimization and applications to nonlinear mechanics. *SIAM Journal on Numerical Analysis*, 47(4):3044–3069, 2009.
9. W. B. Johnson, J. Lindenstrauss, and G. Schechtman. Extensions of Lipschitz maps into Banach spaces. *Israel J. Math.*, 54:129–138, 1986.
10. A. Kopaničáková and R. Krause. Subdivision-based nonlinear multiscale cloth simulation. *SIAM Journal on Scientific Computing*, 41:S433–S46, 2019.
11. A. Kopaničáková and R. Krause. A recursive multilevel trust region method with application to fully monolithic phase-field models of brittle fracture. *Computer Methods in Applied Mechanics and Engineering*, 360:112720, 2020.
12. A. Kopaničáková and R. Krause. Globally convergent multilevel training of deep residual networks. *SIAM Journal on Scientific Computing*, 45:S254–S280, 2022.
13. A. Kopaničáková. On the use of hybrid coarse-level models in multilevel minimization methods. In *International Conference on Domain Decomposition Methods*, pages 303–310. Springer Nature Switzerland Cham, 2022.
14. D. Kozak, S. Becker, A. Doostan, and L. Tenorio. A stochastic subspace approach to gradient-free optimization in high dimensions. *Computational Optimization and Applications*, 79:339–368, 2021.
15. M. Pilanci and M. Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27:205–245, 2017.
16. U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Elsevier, 2000.
17. N. Tsipinakis, P. Tigkas, and P. Parpas. A Multilevel Low-Rank Newton Method with Superlinear Convergence Rate and its Application to Non-convex Problems. `https://arxiv.org/abs/2305.08742`, 2023.
18. Y. Yuan. On the superlinear convergence of a trust region algorithm for nonsmooth optimization. *Mathematical Programming*, 31:269–285, 1985.