DIRECT SEARCH METHODS OVER LIPSCHITZ MANIFOLDS*

David W. Dreisigmeyer†
Los Alamos National Laboratory
Los Alamos, NM 87545

February 27, 2007

Abstract

We extend direct search methods to optimization problems that include equality constraints given by Lipschitz functions. The equality constraints are assumed to implicitly define a Lipschitz manifold. Numerically implementing the inverse (implicit) function theorem allows us to define a new problem on the tangent spaces of the manifold. We can then use a direct search method on the tangent spaces to solve the new optimization problem without any equality constraints. Solving this related problem implicitly solves the original optimization problem. Our main example utilizes the LTMADS algorithm for the direct search method. However, other direct search methods can be employed. Convergence results trivially carry over to our new procedure under mild assumptions.

Key words: Direct search methods, LTMADS, Lipschitz manifold, implicit function theorem, inverse function theorem

AMS subject classifications: 90C56, 53C21

1 Introduction

One difficulty with the lower-triangular mesh adaptive direct search method (LTMADS) [3] is its inability to handle equality constraints. This was relieved somewhat in [14, 15] by defining LTMADS over Riemannian manifolds. Here, one takes any equality constraints as implicitly defining a $\mathcal{C}^2$ Riemannian manifold $\mathcal{M}$. Then, LTMADS can be performed in the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ of $\mathcal{M}$ at a point $\mathbf{x} \in \mathcal{M}$. However, the amount of information one needs about the equality constraint $\mathbf{E}(\mathbf{x}) = \mathbf{0}$ can be prohibitive in actually running the algorithms in [14, 15]. Specifically, gradient and Hessian information about $\mathbf{E}(\mathbf{x})$ was required. Here we will relax these requirements.

*LA-UR-07-1073
†email: dreisigm@lanl.gov

This results in a more practical algorithm that is also aesthetically more pleasing since we will only have to assume that $\mathbf{E}(\mathbf{x})$ is Lipschitz continuous. So now we will arrive at a completely non-smooth version of LTMADS over a Lipschitz manifold $\mathcal{M}$.

The method employed in [14, 15] was to define a mapping $\mathrm{Exp}_{\mathbf{x}} : \mathcal{T}_{\mathbf{x}}\mathcal{M} \to \mathcal{M}$ that allowed one to pullback the objective function $O(\mathbf{x})$ and inequality constraints $\mathbf{I}(\mathbf{x})$ from $\mathcal{M}$ onto $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. That is, given a tangent vector $\omega \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$, a nonlinear system of ODEs was solved to find a unique point $\mathbf{y} \in \mathcal{M}$. One then implicitly defined new functions $\widehat{O}(\omega)$ and $\widehat{\mathbf{I}}(\omega)$ on $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ by letting $\widehat{O}(\omega) \doteq O(\mathbf{y})$ and $\widehat{\mathbf{I}}(\omega) \doteq \mathbf{I}(\mathbf{y})$. Since $\mathcal{T}_{\mathbf{x}}\mathcal{M} \simeq \mathbb{R}^m$ if $\mathcal{M}$ is an m-dimensional manifold, the standard LTMADS could be performed on $\mathcal{T}_{\mathbf{x}}\mathcal{M}$.

The idea that will be employed here is similar to that in [14, 15]. We will still use the pullback feature in order to perform LTMADS on the 'tangent spaces' of $\mathcal{M}$. However, instead of using the $\mathrm{Exp}_{\mathbf{x}}$ mapping that is defined over all of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, we will numerically implement the inverse (implicit) function theorem to perform the pullback. This makes our procedure similar to a non-smooth generalized reduced gradient method [5]. All of the convergence results for LTMADS can be carried over without modification to our method under some very mild restrictions.

For simplicity, we will initially assume that our manifold $\mathcal{M}$ and, hence, $\mathbf{E}(\mathbf{x})$, are $\mathcal{C}^1$. This allows us to develop the algorithm using the standard inverse (implicit) function theorem in its contraction mapping fixed point formulation. After the basic algorithm is developed, we can then drop the $\mathcal{C}^1$ assumption on $\mathcal{M}$ and develop the fully non-smooth algorithm. This will require only slight modifications of our initial method.

Our paper is organized as follows. In Section 2 we give the general solution method when $\mathcal{M}$ is assumed to be $\mathcal{C}^1$ and the gradient information $\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})$ is available. We also state the assumptions needed to (trivially) show our convergence results. In Section 3 we relax our assumptions on $\mathcal{M}$. Now we only take $\mathcal{M}$ to be a Lipschitz manifold. The required modifications to the procedure in Section 2 are given. We give an example of our algorithm over a Lipschitz manifold in Section 4. A discussion follows in Section 5.

2 The $\mathcal{C}^1$ manifold case

Here we will extend the LTMADS method in [3] to optimization problems with smooth equality constraints when gradient information is available for the equality constraints. The general problem we will look at is

$$\min_{\mathbf{x}\in\mathbb{R}^{n+m}} \quad O(\mathbf{x}) \tag{2.1a}$$

$$\text{subject to :} \quad \mathbf{E}(\mathbf{x}) = \mathbf{0} \tag{2.1b}$$

$$\mathbf{I}(\mathbf{x}) \leq \mathbf{0}, \tag{2.1c}$$

where $\mathbf{E} : \mathbb{R}^{n+m} \to \mathbb{R}^n$. The functions $O(\mathbf{x})$ and $\mathbf{I}(\mathbf{x})$ in (2.1) will all be taken to be Lipschitz continuous. Let $\Omega \doteq \{\mathbf{x} \mid \mathbf{I}(\mathbf{x}) \leq \mathbf{0}\}$. We will additionally assume that

$\mathcal{M} \doteq \{\mathbf{x} \mid \mathbf{E}(\mathbf{x}) = \mathbf{0}\}$ is a $\mathcal{C}^1$ manifold of dimension m, at least for $\mathcal{N} \doteq \mathcal{M} \cap \Omega$. Also, the gradient information $\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})$ is taken as accessible for this algorithm.

Assume we begin with an initial point $\mathbf{x} \in \mathcal{N}$ and we find $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. One can then use the contraction mapping fixed point proof of the implicit function theorem to find a unique point $\mathbf{y} \in \mathcal{M}$ for any $\omega \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ around a neighborhood of the origin of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ [18, 26, 27]. To see this let

$$G \quad \doteq \quad \nabla_{\mathbf{z}}\widehat{\mathbf{E}}(0, \mathbf{0}), \tag{2.2}$$

where we have translated $\mathbb{R}^{n+m}$ by $\mathbf{x}$ and rotated our space so the $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ corresponds to the first m directions to find $\widehat{\mathbf{E}}(\omega, \mathbf{z})$, where $\widehat{\mathbf{E}}(0, \mathbf{0}) = \mathbf{0}$. Here $\omega \in \mathbb{R}^m$ and $\mathbf{z} \in \mathbb{R}^n$. By assumption G is invertible. Then the mapping

$$L(\omega, \mathbf{z}) \quad \doteq \quad \mathbf{z} - G^{-1}\widehat{\mathbf{E}}(\omega, \mathbf{z}) \tag{2.3}$$

is a contraction mapping on a neighborhood of $\mathbf{0} \in \mathbb{R}^n$. Finding the fixed point $\hat{\mathbf{z}}$ of (2.3) for a fixed ω gives us the corresponding point $\mathbf{y} \in \mathcal{M}$, namely $\mathbf{y} = [\ \omega^T \ \ \hat{\mathbf{z}}^T\ ]^T$. A method for solving (2.3) is given in [26, 27].

The mapping in (2.3) places coordinates on $\mathcal{M}$ around $\mathbf{x}$ [26, 27]. The $\mathbf{y}$ found using (2.3) is assigned on $\mathcal{M}$ the coordinates ω. This can be done in some region around $\mathbf{x} \in \mathcal{M}$. Once we have found the point $\mathbf{y}$ that corresponds to $\omega \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$, we can pullback the function values $O(\mathbf{y})$ and $\mathbf{I}(\mathbf{y})$ to $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ by simply assigning their values to ω. This implicitly defines new functions $\widehat{O}(\omega)$ and $\widehat{\mathbf{I}}(\omega)$ on $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. If we leave the neighborhood of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ where this procedure is valid, we assign a value of $\widehat{O}(\omega) = +\infty$ to our pulled back objective function. Note that the Lipschitz conditions of $O(\mathbf{x})$ and $\mathbf{I}(\mathbf{x})$ are retained under this pullback procedure. It is this pullback procedure that allows us to extend LTMADS to (2.1). We will eventually, under our assumptions given below, be able to perform the standard LTMADS algorithm in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ for some $\mathbf{x} \in \mathcal{M}$.

The method given by Procedure 2.1 is very simple. We begin with an initial feasible point $\mathbf{x} \in \mathcal{N}$. We then find the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. This is used in (2.3) to find our coordinate system for a neighborhood $\mathcal{U}_{\mathbf{x}} \subset \mathcal{M}$ around $\mathbf{x}$. Implicitly, we can pullback $O(\mathbf{x})$ and $\mathbf{I}(\mathbf{x})$ from $\mathcal{M}$ to $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ to define the pulled-back functions $\widehat{\mathbf{I}}(\omega)$ and $\widehat{O}(\omega)$. Then we can do a standard LTMADS iteration in $\mathcal{T}_{\mathbf{x}}\mathcal{M} \simeq \mathbb{R}^m$ using the Lipschitz functions $\widehat{\mathbf{I}}(\omega)$ and $\widehat{O}(\omega)$. Now, after this iterate one of two things can happen. First, we may find an improving point $\widehat{\omega} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ that corresponds to a feasible point $\mathbf{y} \in \mathcal{N}$. If this happens, we then let our current incumbent solution become $\mathbf{y}$ and switch to $\mathcal{T}_{\mathbf{y}}\mathcal{M}$ to perform LTMADS in. Otherwise, $\mathbf{x}$ is a minimal frame center. In this case we will refine the mesh in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ and perform another LTMADS iterate in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. Note that we do not worry about the (optional) `SEARCH` step in Procedure 2.1, which the user can freely specify.

We can trivially prove the convergence of Procedure 2.1 under some mild assumptions. First, assume we remain in a compact region on $\mathcal{N}$, which is a standard assumption. Then we can cover this region with a finite number of coordinate systems constructed via (2.3). Additionally, assume the iterates of the LTMADS algorithm

Procedure 2.1 ***The General Method for Problem (2.1)***

1. *Let the equality constraints* $\mathbf{E}(\mathbf{x})$ *implicitly define a Lipschitz manifold* $\mathcal{M}$.
2. *Find an initial feasible point* $\mathbf{x} \in \mathcal{N}$.
3. *Use (2.3) to pullback the inequality constraints* $\mathbf{I}(\mathbf{x})$ *and the objective function* $O(\mathbf{x})$ *from* $\mathcal{M}$ *to* $\mathcal{T}_{\mathbf{x}}\mathcal{M}$.
4. *Do an LTMADS iteration in* $\mathcal{T}_{\mathbf{x}}\mathcal{M}$.
5. *If an improved feasible point* $\mathbf{y} \in \mathcal{N}$ *is found in Step 4, let* $\mathbf{x} \to \mathbf{y}$. *Goto Step 3.*
6. *If the stopping conditions are met, return* $\mathbf{x}$ *and stop. Else, refine the LTMADS mesh and go to Step 4.*

eventually enter and remain in one of these coordinate systems. This could be achieved, e.g., by only allowing a finite number of `SEARCH` steps in the LTMADS algorithm. In this case the `SEARCH` step does not have to evaluate points on some fixed mesh as in LTMADS. The reason is that if $\mathbf{y} \in \mathcal{N}$ is a successful search point, we will switch to $\mathcal{T}_{\mathbf{y}}\mathcal{M}$ and 'restart' the LTMADS algorithm. Since this can only happen a finite number of times, eventually we will be performing LTMADS without the optional `SEARCH` step. Now we need to slightly modify Step 5 in Procedure 2.1. We want to remain in a single tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ as the algorithm 'settles down' into the final coordinate patch $\mathcal{U}_{\mathbf{x}} \subset \mathcal{M}$. This could be achieved, e.g., by only switching to a new tangent space $\mathcal{T}_{\mathbf{y}}\mathcal{M}$ if our improving point $\widehat{\omega}$ satisfies the condition $\|\widehat{\omega}\| \geq \epsilon$ for some user specified constant $\epsilon > 0$.

So we have the following situation. Initially we move carefully but 'carefreely' over $\mathcal{N}$, trying to find a nice point $\mathbf{x} \in \mathcal{N}$ that can live up to our exacting requirements. Eventually, the algorithm will settle into some neighborhood $\mathcal{U}_{\mathbf{x}} \subset \mathcal{M}$ around a feasible point $\mathbf{x} \in \mathcal{N}$. Then we will just be doing the standard LTMADS in $\mathcal{T}_{\mathbf{x}}\mathcal{M} \simeq \mathbb{R}^m$ with the implicitly defined Lipschitz constraint functions $\widehat{\mathbf{I}}(\omega)$ and Lipschitz objective function $\widehat{O}(\omega)$. So all of the convergence results proven for LTMADS in [3] carry over *without modification* to our algorithm in Procedure 2.1, though now the convergence analysis is done in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$.

We note that Procedure 2.1 is not restricted to using LTMADS. Any MADS method in [3] can be used. In fact, any direct search method can be performed in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ after the algorithm settles down into the final region $\mathcal{U}_{\mathbf{x}} \subset \mathcal{M}$. This includes, e.g., the filter methods in [2, 4, 11] and the frame based methods in [24].

Procedure 2.1 is similar to the generalized reduced gradient (GRG) method [5]. However, we do not perform a line search to minimize $O(\mathbf{x})$ along some predetermined direction in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. Rather, we implicitly define a mesh in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ as in the standard LTMADS algorithm. Also, instead of minimizing $O(\mathbf{x})$ restricted to $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ and then

projecting down onto $\mathcal{N}$, we (eventually) minimize $\widehat{O}(\omega)$ by pulling-back $O(\mathbf{x})$ from $\mathcal{M}$ onto $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. It is this pullback procedure that makes it easy to prove convergence results for Procedure 2.1. This should be contrasted with the difficulty of proving convergence results for the GRG method, which does not utilize the pullback procedure [5]. It may be possible to define an intrinsic GRG method that minimizes $O(\mathbf{x})$ on $\mathcal{M}$ directly [16]. This would require one to have access to $\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})$ and $\nabla_{\mathbf{xx}}\mathbf{E}(\mathbf{x})$ in order to find the geodesics on $\mathcal{M}$. We will not concern ourselves with this here. Finally, our method relies on no derivative information about $O(\mathbf{x})$ or $\mathbf{I}(\mathbf{x})$, which GRG may use. The derivative information for $O(\mathbf{x})$ would be used to minimize $O(\mathbf{x})$ over $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ in GRG.

3 The Lipschitz manifold case

Now we consider the case where our manifold $\mathcal{M}$ is only Lipschitz. That is, $\mathbf{E}(\mathbf{x})$ in (2.1b) is only assumed to be Lipschitz. The modification of Procedure 2.1 in Section 2 is rather minor and, creates no theoretical problems for our method. The only difference is that we will need to employ some Lipschitz version of the implicit function theorem. Other than this modification, Procedure 2.1 is unchanged. Additionally, the convergence properties given in Section 2 still hold when $\mathcal{M}$ is a Lipschitz manifold.

As we will see, actually implementing a Lipschitz implicit function theorem is not as straightforward computationally as the contraction mapping in (2.3). This creates potential numerical difficulties when employing the pullback procedure on Lipschitz manifolds. However, since the implicit function theorem is called repeatedly in Procedure 2.1, it is crucial that any solution method be highly efficient.

Eventually we want some version of a Lipschitz implicit function theorem that can actually be implemented numerically. An obvious criterion would be to have the implicit function theorem proved constructively with an explicit contraction mapping. We will eventually get to this position but, first let us start easy.

3.1 A Lipschitz implicit function theorem

Instead of relying on the usual inverse or implicit function theorems, we now need to use some form of non-smooth version of these theorems to construct our coordinate patches. There are many forms of implicit and/or inverse function theorems for Lipschitz functions, e.g., [9, 10, 12, 17, 18, 19, 20, 23, 28]. Each of these overcomes the lack of smoothness by placing additional assumptions on $\mathbf{E}(\mathbf{x})$. Here we will only concern ourselves with the implicit function theorem in [9, 10, 23]. We'll need to develop some machinery in order to state the theorem. Assume that $\mathbf{E}(\mathbf{x})$ is Lipschitz. Then, Rademacher's theorem states that $\mathbf{E}(\mathbf{x})$ is differentiable almost everywhere. Let $\Omega_{\mathbf{E}(\mathbf{x})}$ be the set of points where $\mathbf{E}(\mathbf{x})$ fails to be differentiable. Then we can start with the following definition:

Definition 3.1 ***The Generalized Jacobian [9]***

The **generalized Jacobian** $\partial\mathbf{E}(\mathbf{x})$ *of* $\mathbf{E}(\mathbf{x})$ *at* $\mathbf{x}$ *is the closed convex hull of all the matrices* Z *obtained as the limit of a sequence of the form* $\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x}_i)$ *where* $\mathbf{x}_i \to \mathbf{x}$ *and* $\mathbf{x}_i \notin \Omega_{\mathbf{E}(\mathbf{x})}$. *Symbolically, we have:*

$$\partial\mathbf{E}(\mathbf{x}) \quad \doteq \quad \overline{\mathrm{co}}\left\{\lim_{\mathbf{x}_i\to\mathbf{x}} \nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x}_i) \mid \mathbf{x}_i \notin \Omega_{\mathbf{E}(\mathbf{x})}\right\}. \tag{3.1}$$

We'll need one more definition to state our Lipschitzian version of the implicit function theorem. Let $\partial_{\mathbf{z}}\mathbf{E}(\mathbf{x})$ denote the set of all n-by-n matrices M such that there exists an n-by-m matrix N and a unitary matrix P where $[\ N \quad M\ ]P \in \partial\mathbf{E}(\mathbf{x})$, with $P\mathbf{x} = [\ \mathbf{w}^T \ \ \mathbf{z}^T\ ]^T$.

Definition 3.2 ***Maximal and Uniformly Maximal Rank [9, 30]***

We say that $\partial\mathbf{E}(\mathbf{x})$ *is of* **maximal rank** *at the point* $\mathbf{x}$ *if every matrix in* $\partial\mathbf{E}(\mathbf{x})$ *is of maximal rank. The generalized Jacobian* $\partial\mathbf{E}(\mathbf{x})$ *is said to have* **uniformly maximal rank** *at* $\mathbf{x}$ *if there exists an* n*-dimensional vector* $\mathbf{z}$ *such that* $\partial_{\mathbf{z}}\mathbf{E}(\mathbf{x})$ *has maximal rank.*

We can now state the following implicit function theorem:

Theorem 3.3 ***The Implicit Function Theorem Version I [9, 10, 23]***

Let $\partial_{\mathbf{z}}\mathbf{E}(\hat{\mathbf{w}}, \hat{\mathbf{z}})$ *have maximal rank. Then there exists a neighborhood* W *of* $\hat{\mathbf{w}}$ *and a Lipschitz function* $\varphi : W \to \mathbb{R}^n$ *such that* $\varphi(\hat{\mathbf{w}}) = \hat{\mathbf{z}}$ *and, for every* $\mathbf{w} \in W$,

$$\mathbf{E}(\mathbf{w}, \varphi(\mathbf{w})) \quad = \quad \mathbf{E}(\hat{\mathbf{w}}, \hat{\mathbf{z}}). \tag{3.2}$$

Now we're going to specialize Theorem 3.3 to implicitly defined Lipschitz manifolds. First let us give one more definition.

Definition 3.4 ***Uniformly Regular Value [30]***

A vector $\mathbf{y} \in \mathbb{R}^n$ *is a* **uniformly regular value** *of* $\mathbf{E}(\mathbf{x})$ *if, for each* $\mathbf{x} \in \mathbf{E}^{-1}(\mathbf{y})$, $\partial\mathbf{E}(\mathbf{x})$ *has uniformly maximal rank at* $\mathbf{x}$.

This leads to the type of manifolds we will consider.

Theorem 3.5 ***Uniformly Regular Level Set [30]***

If $\mathbf{y}$ *is a uniformly regular value of* $\mathbf{E}(\mathbf{x})$, *then* $\mathbf{E}^{-1}(\mathbf{y})$ *is an* m*-dimensional Lipschitz manifold or is empty. We call* $\mathbf{E}(\mathbf{x}) = \mathbf{y}$ *a* **uniformly regular level set**.

For the level set $\mathbf{E}(\mathbf{x}) = \mathbf{0}$, we will take $\mathbf{0}$ as being a uniformly regular value of $\mathbf{E}(\mathbf{x})$, where we assume $\mathbf{E}^{-1}(\mathbf{0}) \neq \emptyset$. Then we will have an implicitly defined Lipschitz manifold $\mathcal{M}$.

We need to turn our implicit function theorem into a practical algorithm. That is, we want to derive an equation similar to (2.3). When deriving (2.3) we had that $\omega \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$. The important feature of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ is that $\mathcal{T}_{\mathbf{x}}\mathcal{M} \cap \mathcal{N}_{\mathbf{x}}\mathcal{M} = \{\mathbf{0}\}$, where $\mathcal{N}_{\mathbf{x}}\mathcal{M} = \mathrm{span}([\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})]^T)$ are the normal directions to $\mathcal{M}$ at $\mathbf{x}$. Any other m-dimensional linear subspace $\mathsf{T} \subset \mathbb{R}^{n+m}$ such that $\mathsf{T} \cap \mathcal{N}_{\mathbf{x}}\mathcal{M} = \{\mathbf{0}\}$ could also have been used. Now, with some abuse of notation, $\partial_{\mathbf{x}}\mathbf{E}(\mathsf{U}\mathbf{x}) = \partial_{\mathbf{y}}\mathbf{E}(\mathbf{y})\mathsf{U}$, $\mathbf{y} = \mathsf{U}\mathbf{x}$, for any unitary matrix U. If we let $\mathsf{U} = [\ \mathsf{W}\ \ \mathsf{Z}\ ]$ such that $\mathbf{x} = \mathsf{W}\mathbf{w} + \mathsf{Z}\mathbf{z}$ and, assume $\partial_{\mathbf{z}}\mathbf{E}(\mathsf{W}\hat{\mathbf{w}} + \mathsf{Z}\hat{\mathbf{z}}) = \partial_{\mathbf{x}}\mathbf{E}(\hat{\mathbf{x}})\mathsf{Z}$ has maximal rank, then, by Theorem 3.3, we can let $\mathsf{T} = \mathsf{W}$. $\partial_{\mathbf{x}}\mathbf{E}(\hat{\mathbf{x}})\mathsf{Z}$ has maximal rank if and only if $\mathsf{W} \cap \mathrm{span}(\mathsf{J}^T) = \{\mathbf{0}\}$ for every $\mathsf{J} \in \partial_{\mathbf{x}}\mathbf{E}(\mathbf{x})$. So, the requirement that $\mathsf{T} \cap \mathrm{span}([\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})]^T) = \{\mathbf{0}\}$ when $\mathcal{M}$ is $\mathcal{C}^1$ is replaced by the requirement that $\mathsf{T} \cap \mathrm{span}(\mathsf{J}^T) = \{\mathbf{0}\}$ for every $\mathsf{J} \in \partial_{\mathbf{x}}\mathbf{E}(\mathbf{x})$ when $\mathcal{M}$ is Lipschitz. Since, by assumption, $\mathbf{E}^{-1}(\mathbf{0})$ is a nonempty uniformly regular level set, we are guaranteed that an appropriate T exists. So, theoretically we are done.

3.2 Numerical issues

Practically, though, we need to be able to find an appropriate T and, numerically implement some Lipschitz implicit function theorem. Unfortunately, the proof of Theorem 3.3 does not provide a constructive way for finding the implicit function $\varphi(\mathbf{w})$. There's a more general Lipschitz implicit function theorem given by Kummer in [19] that will prove more valuable for us (see also [17, 18]). Define the set

$$\Delta\mathbf{E}(\mathbf{x};\mathbf{u}) \quad = \quad \left\{ \mathbf{v} \,\middle|\, \mathbf{v} = \lim_k \frac{\mathbf{E}(\mathbf{x}_k + \lambda_k\mathbf{u}) - \mathbf{E}(\mathbf{x}_k)}{\lambda_k}, \mathbf{x}_k \to \mathbf{x}, \lambda_k \downarrow \mathbf{0} \right\}. \quad (3.3)$$

Then we have the following implicit function theorem:

Theorem 3.6 ***The Implicit Function Theorem Version II [17, 18, 19]***

Let $\mathbf{E}(\hat{\mathbf{w}}, \hat{\mathbf{z}}) = \hat{\mathbf{y}}$. *Assume* $\mathbf{0} \notin \Delta\mathbf{E}((\hat{\mathbf{w}}, \hat{\mathbf{z}}); (\mathbf{0}, \zeta))$ *for any* $\zeta \in \mathbb{R}^n$, $\|\zeta\| = 1$. *Then, there exists neighborhoods* Z *of* $\hat{\mathbf{z}}$ *and* W *of* $(\hat{\mathbf{y}}, \hat{\mathbf{w}})$ *such that, for every* $(\mathbf{y}, \mathbf{w}) \in W$ *there is a unique* $\mathbf{z} = \varphi(\mathbf{y}, \mathbf{w}) \in Z$, *where* $\mathbf{E}(\mathbf{w}, \varphi(\mathbf{y}, \mathbf{w})) = \mathbf{y}$. *Further,* $\varphi(\mathbf{y}, \mathbf{w})$ *is Lipschitz on* W.

First we'll concern ourselves with finding the implicit function $\varphi(\mathbf{0}, \mathbf{w})$. What we're after is some numerically implementable contraction mapping similar to (2.3). Such a mapping is given in [19]. First, assume $\mathbf{E}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ and fix some $\mathbf{w} \in \mathsf{T}$, for an appropriately chosen T. Now let $\mathbf{y}_k = \mathbf{E}(\mathbf{0}, \mathbf{z}_k) - \mathbf{E}(\mathbf{w}, \mathbf{z}_k)$ and, let $\mathbf{z}_{k+1}$ be the solution to the equation $\mathbf{E}(\mathbf{0}, \mathbf{z}) = \mathbf{y}_k$. If $\mathbf{y}_{k+1} = \mathbf{y}_k$, then $\mathbf{z} \doteq \mathbf{z}_{k+1}$ is such that $\mathbf{E}(\mathbf{w}, \mathbf{z}) = \mathbf{0}$. An intelligent choice for $\mathbf{z}_0$ is obviously important for making this method practical.

There are many methods for solving $\mathbf{E}(\mathbf{0}, \mathbf{z}) = \mathbf{y}_k$, see, e.g., [21, 22, 25, 29, 32, 33] for some recent algorithms along with their references. Typically these solution methods require extra assumptions on $\mathbf{E}(\mathbf{x})$ beyond the Lipschitz condition in order to prove convergence results. An additional way of solving $\mathbf{E}(\mathbf{0}, \mathbf{z}) = \mathbf{y}_k$ is to employ some

of the techniques developed in [1] for constructing piecewise-linear approximations to $\mathcal{M}$. We'll employ the latter technique in Section 4 to illustrate its use.

We're really looking for the inverse function $\Phi(\mathbf{y})$ in a neighborhood of $\mathbf{y} = \mathbf{0}$ where $\mathbf{E}(\mathbf{0}, \Phi(\mathbf{y})) = \mathbf{y}$. Being able to efficiently construct an approximation $\widetilde{\Phi}(\mathbf{y})$ to $\Phi(\mathbf{y})$ is especially important as the direct search method converges and we remain in a fixed T around a point $\mathbf{x} \in \mathcal{M}$. Then $\widetilde{\Phi}(\mathbf{y})$ will, hopefully, provide us with a good approximation to $\mathbf{z}_{k+1}$. This ultimate goal should be kept in mind when trying to solve $\mathbf{E}(\mathbf{0}, \mathbf{z}) = \mathbf{y}_k$. For example, it may be worthwhile to store all previous evaluations of $\mathbf{E}(\mathbf{x})$ in order to reuse them at future iterations of Procedure 2.1. A procedure for approximating implicit and inverse functions is presented in [13].

Looking at (3.3), we see that Kummer's implicit function theorem also gives us a practical way of trying to find an appropriate T. Fix a small $\lambda > 0$ and, choose a random unitary matrix $\mathsf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_{n+m}]$, where $\mathbf{u}_i \in \mathbb{R}^{n+m}$. Now find the quantities $n_i = \|E(\lambda \mathbf{u}_i) - E(\mathbf{0})\|$ for $i = 1, \ldots, n+m$. Then, we will let $\mathsf{T} = [\mathbf{u}_{i_1}, \ldots, \mathbf{u}_{i_m}]$, where $0 \leq n_{i_1} \leq \ldots \leq n_{i_m} \leq \ldots \leq n_{i_{n+m}}$. Choosing T in this way helps us guarantee that $\mathbf{0} \notin \Delta\mathbf{E}(\mathbf{0}; \mathbf{y})$, where $y_j = 0$ for $j = i_1, \ldots, i_m$, $\|\mathbf{y}\| = 1$ and $\mathbf{E}(\mathbf{0}) = \mathbf{0}$.

It is somewhat important to choose a random U above. The reason is, we don't want to end up with a T where for some $\mathbf{w} \in \mathsf{T}$ we have an uncountably infinite number of points $\mathbf{z} \in \mathcal{M}$ such that $\mathbf{E}(\mathbf{w}, \mathbf{z}) = \mathbf{0}$. This situation can cause the direct search method to become 'stuck' at a nonlocal solution on $\mathcal{M}$. It also suggests that we try a restart procedure after finding a potential solution to ensure it is a local minimizer on $\mathcal{M}$. This restart procedure would involve choosing a new T at our current potential solution and rerunning the direct search algorithm. An example of this potential situation is given in Section 4. A restart procedure would also be warranted if the implicit function algorithm consistently fails. This would indicate a poor choice for T.

An alternate, and probably more intelligent, method for finding T would be to use the SVD based procedure in [6, 7] for determining an approximation to the tangent plane. Roughly, one takes $\hat{\jmath}$ points $\mathbf{y}_{\hat{\imath}} \in \mathcal{M}$ within an epsilon ball of a given point $\mathbf{x} \in \mathcal{M}$. Then the first m left singular vectors of the matrix $[\, \mathbf{y}_1 - \mathbf{x}, \ldots, \mathbf{y}_{\hat{\jmath}} - \mathbf{x} \,]$ will approximately span $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. This works quite well for Riemannian manifolds, especially when one knows *a priori* the dimensionality of the manifold, as we do. Additionally, it requires no derivative information. However, there are no theoretical guarantees for its performance when we are dealing with a Lipschitz manifold. Also, the method is somewhat sensitive to the curvature of the manifold. Finally, the method can breakdown if the manifold almost self-intersects at $\mathbf{x}$. That is, around $\mathbf{x}$ there are points on $\mathcal{M}$ that are 'close' to $\mathbf{x}$ when measured extrinsically in the ambient space but, are 'far' from $\mathbf{x}$ when measured intrinsically on $\mathcal{M}$. The last two limitations of the SVD procedure become less important, if $\mathcal{M}$ is Riemannian, as our optimization algorithm approaches a solution because we can shrink our epsilon ball arbitrarily small. Also, the SVD procedure in [6, 7] at least gives us an intelligent method for determining $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ when $\mathcal{M}$ is actually $\mathcal{C}^1$ but we do not have access to $\nabla_{\mathbf{x}}\mathbf{E}(\mathbf{x})$. A restart procedure is still a good idea even if the SVD method is used to find T, especially if we suspect that $\mathcal{M}$ is

Lipschitz. Again, we'll see an example of why this is so in Section 4.

4 Example

Now we'll look at an example similar to the one used in [3, 14] to demonstrate the LTMADS algorithm. There, a linear objective function was minimized in $\mathbb{R}^n$ over a closed n-dimensional ball [3] or, an $(n-1)$-dimensional hypersphere [14]. Here, we will minimize a linear objective function over the $(n-1)$-dimensional hypercube embedded in $\mathbb{R}^n$. So now we are dealing with a Lipschitz manifold rather than a Riemannian manifold. Our optimization problem is:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad O(\mathbf{x}) = \mathbf{1}^T\mathbf{x} \tag{4.1a}$$

$$\text{subject to :} \quad \mathbf{E}(\mathbf{x}) = \|\mathbf{x}\|_\infty - 3n = 0 \tag{4.1b}$$

where $n = 5, 10, 20$ or 50. The optimal solution is given by $\mathbf{x} = -3n\mathbf{1}$ with a corresponding optimal value of $-3n^2$.

Note that if we were to use the fixed matrix $\mathsf{U} = I$ to determine our T matrix, it is possible for the direct search algorithm to become 'stuck' in a nonlocal optimizer. To see this, consider the case where $n = 2$ and choose an initial starting point $[0, 6]^T \in \mathcal{M}$. Then $\mathsf{T} = \text{span}([6, 0]^T)$. As the algorithm proceeds, we will end up at the point $[-6, 6]^T$ which is not a local optimizer on $\mathcal{M}$. The reason is that the 'west' face of $\mathcal{M}$ is normal to T and, the implicit function theorem will not allow us to move away from $[-6, 6]^T$ with our current (fixed) T. This is the reason a random U should be used to determine T and/or a restart procedure should be performed with a different T after locating a potential solution.

In order to solve $\mathbf{E}(\mathbf{0}, \mathbf{z}) = \mathbf{y}_k$ when implementing Kummer's implicit function theorem, we will borrow some techniques developed in [1]. Assume we have chosen an appropriate T and want to implement the implicit function theorem for $\mathbf{w} \in \mathsf{T}$. Now, unless $\mathbf{w} \in \mathcal{M}$, we will have $\mathbf{E}(\mathbf{w}) > 0$ or $\mathbf{E}(\mathbf{w}) < 0$. For simplicity, assume we have $\mathbf{E}(\mathbf{w}) > 0$, so that $\mathbf{w}$ lies outside of the hypercube. In order to find the point $\mathbf{x} \in \mathcal{M}$ that corresponds to $\mathbf{w}$, we will move in the one-dimensional subspace $\mathsf{N} \doteq \mathsf{T}^\perp$. Assume we do this, and find a new point $\boldsymbol{w}$ such that $\mathbf{E}(\boldsymbol{w}) < 0$. Then we are assured that $\mathcal{M}$ must intersect the line segment $[\mathbf{w}, \boldsymbol{w}]$ that connects $\mathbf{w}$ to $\boldsymbol{w}$ in $\mathbb{R}^n$. We can then begin to subdivide this line segment and, can easily determine which piece $\mathcal{M}$ intersects because the endpoints must have different signs for $\mathbf{E}(\mathbf{x})$. In this way we can 'zoom in' on the point $\mathbf{x} \in \mathcal{M}$ that corresponds to $\mathbf{w} \in \mathsf{T}$. This technique can be extended to arbitrary uniformly regular level sets.

This problem was solved using the maximal positive basis LTMADS [3] in T_k at iteration k using $D_k = [\ \mathsf{T}_k \quad -\mathsf{T}_k\ ]$. The b_l were saved and reused to construct the B matrices, though the B matrices in LTMADS were multiplied by the T_k to construct the `POLL` points on the current mesh. This is similar to what was done in [14] for doing LTMADS over $\mathcal{C}^2$ Riemannian manifolds using geodesics. The T_k were given by the

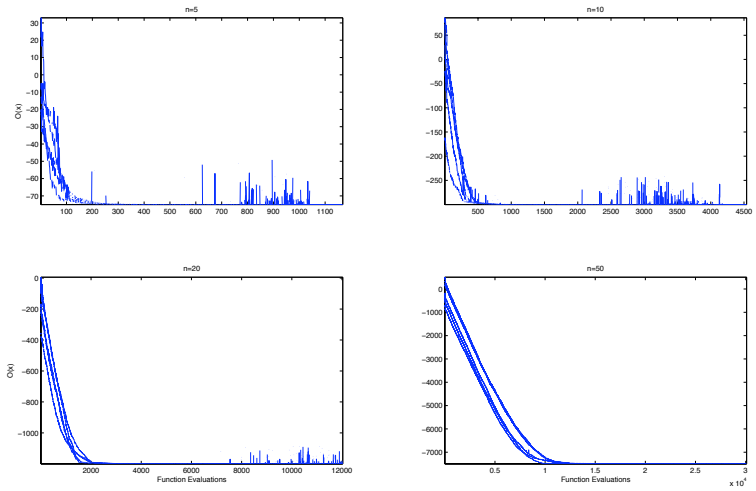

Figure 1: *The objective function value versus number of function evaluations for problem (4.1) using the LTMADS algorithm in* T_k.

best $(n-1)$ columns of the Q matrix from a QR decomposition of a perturbation of the identity matrix, using $\lambda = 10^{-8}$ to determine the best columns for the procedure outlined in Section 3.2. We started with the initial values $\Delta_0^m = \Delta_{max}^m = 1$ and $\Delta_0^p = \Delta_{max}^p = 1$. If we found an improving point, a SEARCH step was done in the same 'tangent' direction by moving 4 times the step length, in T_k, that gave the improving point. We terminated the algorithm whenever $\Delta_k^p \leq 10^{-12}$ or, when $k > 600n$. The algorithm was run five times for each n, starting with a randomly chosen point on the unit hypercube. For $n = 20$ and 50 we always exceeded the maximum allowed function evaluations. The results are shown in Figure 1. The algorithm always converged to (nearly) the correct solution, even when $n = 20$ or 50.

5 Discussion

In [14, 15] it was assumed that the manifolds were $\mathcal{C}^2$ and, that Jacobian and Hessian information was available for the equality constraints. Then we could use the $\mathrm{Exp}_{\mathbf{x}}$ mapping of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ into $\mathcal{M}$ to pullback the objective and inequality constraints from $\mathcal{M}$ to $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. This method is somewhat unsatisfactory for two reasons. First, assuming that Hessian information is available for the equality constraints restricts the practicality of the algorithm. Secondly, the resulting method is not completely a derivative-free technique.

We've seen how to extend the techniques developed in [14, 15] to $\mathcal{C}^1$ and Lipschitz

manifolds that are implicitly defined as uniformly regular level sets. Now we employ an implicit function theorem in order to do our pullback procedure instead of the $\mathrm{Exp}_{\mathbf{x}}$ mapping. This has the disadvantage of typically requiring us to move from one tangent space to another because the implicit function theorem only works locally around the origin of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, while the mapping $\mathrm{Exp}_{\mathbf{x}}$ works globally in $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. This is really not too great of a disadvantage since one will usually switch tangent spaces when employing the methods in [14, 15] in order to make the $\mathrm{Exp}_{\mathbf{x}}$ mapping cheaper to calculate. The main advantage of our new technique is that less derivative information is required in order to employ it. Also, as in [14, 15], proving convergence results is trivial under rather mild restrictions.

Theoretically the $\mathcal{C}^1$ and Lischitz cases are entirely equivalent. We must admit, however, that the $\mathcal{C}^1$ case is an easier to implement and, currently, a more practical algorithm than the Lipschitz case. This is because the $\mathcal{C}^1$ version of the implicit function theorem gives us a readily implementable solution method in (2.3). In contrast, Kummer's Lipschitz implicit function theorem requires us to implement a Lipschitz version of the inverse function theorem. Solution methods for this problem are not currently as well developed as they are for the $\mathcal{C}^1$ case. On a more aesthetic note, the Lipschitz case does provide us with a completely derivative-free method for handling both inequality and equality constraints. Also, as the solution methods for non-smooth equations improve, dealing with Lipschitz manifolds will become a more manageable task.

Potentially the most fruitful avenue for extending all of these methods is to examine how one can deal with a breakdown in the manifold structure. For smooth manifolds this breakdown would be indicated by a non-maximal rank Jacobian of our equality constraints. This falls under the heading of singularity theory and the more specialized, but also more impressively named, catastrophe theory. Another area that may prove interesting is the applications of Morse theory to our optimization problems. Roughly, Morse theory will inform us about the topology of our manifold and, when we may need to split an optimization problem into 'sub-pieces' in order to try and find a global solution to our problem. We refer the reader to [8, 31] for some research in this area.

6 Acknowledgments

The author would like to thank John E. Dennis, JR. (Rice University) for his very helpful discussions. Additional appreciations go to Los Alamos National Laboratory for their postdoctoral program and, the Director's of Central Intelligence Postdoctoral Fellowship program for providing prior research funding, out of which this paper grew.

References

[1] E. L. Allgower and P. H. Schmidt. An algorithm for piecewise-linear approximation of an implicitly defined manifold. *SIAM Journal on Numerical Analysis*, 22:322–346, 1985.

[2] C. Audet and J. E. Dennis, JR. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, 14:980–1010, 2004.

[3] C. Audet and J. E. Dennis, JR. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, 17:188–217, 2006.

[4] C. Audet and J. E. Dennis, JR. Derivative-free nonlinear programming by filters and mesh adaptive direct searches. Unpublished, 2007.

[5] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley, 2nd edition, 1993.

[6] D. S. Broomhead, R. Indik, A. C. Newell, and D. A. Rand. Local adaptive Galerkin bases for large-dimensional dynamical systems. *Nonlinearity*, 4:159–197, 1991.

[7] D. S. Broomhead, R. Jones, and G. P. King. Topological dimension and local coordinates from time series data. *Journal of Physics A*, 20:L563–L569, 1987.

[8] J. Casti. Singularity theory for nonlinear optimization problems. *Applied Mathematics and Computation*, 23:137–161, 1987.

[9] F. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.

[10] F. H. Clarke. On the inverse function theorem. *Pacific Journal of Mathematics*, 64:97–102, 1976.

[11] J. E. Dennis, JR., C. J. Price, and I. D. Coope. Direct search methods for nonlinearly constrained optimization using filters and frames. *Optimization and Engineering*, 5:123–144, 2004.

[12] A. L. Dontchev. Implicit function theorems for generalized equations. *Mathematical Programming*, 70:91–106, 1995.

[13] D. W. Dreisigmeyer. Approximate inverse and implicit functions using an M/NSET variation. *In preparation*.

[14] D. W. Dreisigmeyer. Direct search algorithms over Riemannian manifolds. *Submitted to SIOPT*.

[15] D. W. Dreisigmeyer. Equality constraints, Riemannian manifolds and direct search methods. *Submitted to SIOPT*.

[16] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20:303–353, 1998.

[17] M. Fečkan. An inverse function theorem for continuous mappings. *Journal of Mathematical Analysis and Applications*, 185:118–128, 1994.

[18] S. G. Krantz and H. R. Parks. *The Implicit Function Theorem: History, Theory and Applications*. Birkhäuser, 2002.

[19] B. Kummer. An implicit-function theorem for $C^{0,1}$-equations and parametric $C^{1,1}$-equations. *Journal of Optimization Theory and Applications*, 158:35–46, 1991.

[20] L. Kuntz. An implicit function theorem for directionally differentiable functions. *Journal of Optimization Theory and Applications*, 86:263–270, 1995.

[21] Z. Meng and J. Zhang. Nonlinear Krylov subspace methods for solving nonsmooth equations. *Applied Mathematics and Mechanics (English Edition)*, 26:1172–1180, 2005.

[22] J. S. Pang and L. Qi. Nonsmooth equations: motivation and algorithms. *SIAM Journal on Optimization*, 3:443–465, 1993.

[23] B. H. Pourciau. Analysis and optimization of Lipschitz continuous mappings. *Journal of Optimization Theory and Applications*, 22:311–351, 1977.

[24] C. J. Price and I. D. Coope. Frames and grids in unconstrained and linearly constrained optimization: a nonsmooth approach. *SIAM Journal on Optimization*, 14:415–438, 2003.

[25] L. Qi and X. Chen. A globally convergent successive approximation method for severely nonsmooth equations. *SIAM Journal on Control and Optimization*, 33:402–418, 1995.

[26] W. C. Rheinboldt. On the computations of multi-dimensional solution manifolds of parametrized equations. *Numerische Mathematik*, 53:165–181, 1988.

[27] W. C. Rheinboldt. MANPAK: a set of algorithms for computations on implicitly defined manifolds. *Computers and Mathematics with Applications*, 32:15–28, 1996.

[28] S. M. Robinson. An implicit function theorem for a class of nonsmooth functions. *Mathematics of Operations Research*, 16:292–309, 1991.

[29] S. M. Robinson. Newton's method for a class of nonsmooth functions. *Set-Valued Analysis*, 12:291–305, 1994.

[30] C. Shannon. A prevalent transversality theorem for Lipschitz functions. *Proceedings of the American Mathematical Society*, 134:2755–2765, 2006.

[31] D. E.. Stewart. Constrained optimization and Morse theory. Computational Mathematics Technical Report 98-107, Department of Mathematics, University of Iowa, 1998.

[32] H. Xu and X. W. Chang. Approximate Newton methods for nonsmooth equations. *Journal of Optimization Theory and Applications*, 93:373–394, 1997.

[33] Y. F. Yang. A new trust region method for nonsmooth equations. *The Australian and New Zealand Industrial and Applied Mathematics Journal*, 44:595–607, 2003.