Mesh Adaptive Direct Searches for Constrained Optimization

Charles Audet, École Polytechnique de Montréal

John Dennis, Rice University

LtCol Mark Abramson, AFIT

July 2007
1 Introduction
1 Introduction

2 Generalized pattern search (GPS)
1 Introduction
2 Generalized pattern search (GPS)
3 Mesh adaptive direct search (MADS)
1 Introduction

2 Generalized pattern search (GPS)

3 Mesh adaptive direct search (MADS)

4 A progressive barrier MADS algorithm
1. Introduction
2. Generalized pattern search (GPS)
3. Mesh adaptive direct search (MADS)
4. A progressive barrier MADS algorithm
5. Some numerical results
Presentation Outline

1. Introduction
2. Generalized pattern search (GPS)
3. Mesh adaptive direct search (MADS)
4. A progressive barrier MADS algorithm
5. Some numerical results
6. Discussion
Presentation outline

1. Introduction
   - Target optimization problems
   - Direct search methods
   - Nonsmooth optimality conditions

2. Generalized pattern search (GPS)

3. Mesh adaptive direct search (MADS)

4. A progressive barrier MADS algorithm

5. Some numerical results

6. Discussion
My main research interest is nonsmooth optimization:

\[
\begin{align*}
\text{(NLP)} & \quad \text{minimize} \quad f(x) \\
\text{subject to} & \quad x \in \Omega,
\end{align*}
\]

where \( f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \).
Target optimization problems

My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize} \quad f(x)\]

subject to \( x \in \Omega, \)

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \) and:

- evaluation of \( f \) and of the functions defining \( \Omega \) are usually the result of a computer code (a black box)
Target optimization problems

My main research interest is nonsmooth optimization:

\[(NLP)\]

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad x \in \Omega,
\end{align*}
\]

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \) and:

- evaluation of \( f \) and of the functions defining \( \Omega \) are usually the result of a computer code (a black box)
- the functions are nonsmooth, with some 'if's and 'goto's
Target optimization problems

My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize } f(x) \]
\[\text{subject to } x \in \Omega,\]

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \) and:

- evaluation of \( f \) and of the functions defining \( \Omega \) are usually the result of a computer code (a black box)
- the functions are nonsmooth, with some 'if's and 'goto's
- the functions are expensive black boxes - secs, mins, days
My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize} \quad f(x) \]

subject to \( x \in \Omega \),

where \( f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \) and:

- evaluation of \( f \) and of the functions defining \( \Omega \) are usually the result of a computer code (a black box);
- the functions are nonsmooth, with some 'if's and 'goto's;
- the functions are expensive black boxes - secs, mins, days;
- the functions may fail unexpectedly even for \( x \in \Omega \).
My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize} \quad f(x) \]
subject to \(x \in \Omega,\)

where \(f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}\) may be discontinuous, and \(\Omega\) is any subset of \(\mathbb{R}^n\) and:

- evaluation of \(f\) and of the functions defining \(\Omega\) are usually the result of a computer code (a black box)
- the functions are nonsmooth, with some 'if's and 'goto's
- the functions are expensive black boxes - secs, mins, days
- the functions may fail unexpectedly even for \(x \in \Omega\)
- only a few correct digits are ensured
My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize} \quad f(x) \]
subject to \( x \in \Omega, \)

where \( f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\} \) may be discontinuous, and \( \Omega \) is any subset of \( \mathbb{R}^n \) and:

- evaluation of \( f \) and of the functions defining \( \Omega \) are usually the result of a computer code (a black box)
- the functions are nonsmooth, with some 'if's and 'goto's
- the functions are expensive black boxes - secs, mins, days
- the functions may fail unexpectedly even for \( x \in \Omega \)
- only a few correct digits are ensured
- accurate approximation of derivatives is problematic
My main research interest is nonsmooth optimization:

\[(NLP) \quad \text{minimize} \quad f(x) \]

subject to \(x \in \Omega,\)

where \(f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}\) may be discontinuous, and \(\Omega\) is any subset of \(\mathbb{R}^n\) and:

- evaluation of \(f\) and of the functions defining \(\Omega\) are usually the result of a computer code (a black box)
- the functions are nonsmooth, with some 'if's and 'goto's
- the functions are expensive black boxes - secs, mins, days
- the functions may fail unexpectedly even for \(x \in \Omega\)
- only a few correct digits are ensured
- accurate approximation of derivatives is problematic
- the constraints defining \(\Omega\) may be nonlinear, nonconvex, nonsmooth and may simply return 'yes/no'.
Example: spent potliner treatment (aluminium industry)

Evaluation of $f$ and $\Omega$ requires running the ASPEN chemical engineering process simulation software.

7 variables, 4 black box constraints. Aspen fails on 43% of function calls (floor in graph).
Example: Maintenance planning

A noisy objective function $f$. 

![Graph of a noisy objective function](image)
Example: Underwater acoustic

Many local optimal solutions.
Direct search methods

- Use directly the function values.
- Do not require derivatives.
- Do not attempt to estimate derivatives.
- Ex: DIRECT, MADS, Nelder-Meade, Pattern Search.
Nonsmooth optimality conditions

Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

\[
\begin{align*}
(NLP) & \quad \rightarrow \quad \text{MADS} \\
\text{MADS} & \quad \rightarrow \quad \hat{x}
\end{align*}
\]
Nonsmooth optimality conditions

Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

\[
(NLP) \quad \Rightarrow \quad \text{MADS} \quad \Rightarrow \quad \hat{x}
\]

Unconstrained optimization hierarchy of results:

- if \(f\) is \(C^1\) then \(\nabla f(\hat{x}) = 0\)
Nonsmooth optimality conditions

Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

\[ (NLP) \quad \rightarrow \quad \text{MADS} \quad \rightarrow \quad \hat{x} \]

Unconstrained optimization hierarchy of results:

- If \(f\) is \(C^1\), then \(\nabla f(\hat{x}) = 0 \iff f'(\hat{x}; d) \geq 0 \quad \forall d \in \mathbb{R}^n\)
Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

Unconstrained optimization hierarchy of results:

- If \(f\) is \(C^1\) then \(\nabla f(\hat{x}) = 0 \iff f'(\hat{x}; d) \geq 0 \ \forall d \in \mathbb{R}^n\)
- If \(f\) is convex then \(0 \in \partial f(\hat{x})\)
Nonsmooth optimality conditions

Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

\[
(NLP) \rightarrow \text{MADS} \rightarrow \hat{x}
\]

Unconstrained optimization hierarchy of results:

If \(f\) is \(C^1\), then \(\nabla f(\hat{x}) = 0 \iff f'(\hat{x}; d) \geq 0 \ \forall d \in \mathbb{R}^n\)

If \(f\) is convex, then \(0 \in \partial f(\hat{x})\)

If \(f\) is Lipschitz near \(\hat{x}\), then \(0 \in \partial f(\hat{x})\)
Nonsmooth optimality conditions

Given any optimization problem \((NLP)\), our MADS algorithm produces a solution \(\hat{x}\)

\[ (NLP) \rightarrow \text{MADS} \rightarrow \hat{x} \]

Unconstrained optimization hierarchy of results:

- If \( f \) is \(C^1\)
  \[ \nabla f(\hat{x}) = 0 \iff f'(\hat{x}; d) \geq 0 \quad \forall d \in \mathbb{R}^n \]

- If \( f \) is convex
  \[ 0 \in \partial f(\hat{x}) \]

- If \( f \) is Lipschitz near \( \hat{x} \)
  \[ 0 \in \partial f(\hat{x}) \iff f^\circ(\hat{x}; d) \geq 0 \quad \forall d \in \mathbb{R}^n. \]
Constrained optimization optimality conditions

Necessary optimality condition

If \( \hat{x} \in \Omega \) is a local minimum of a differentiable function \( f \) over a convex set \( \Omega \subset \mathbb{R}^n \), then

\[
f'(\hat{x}; d) \geq 0 \quad \forall \ d \in T_{\Omega}(\hat{x}),
\]

where \( f'(\hat{x}; d) = \lim_{t \to 0} \frac{f(\hat{x} + td) - f(\hat{x})}{t} = d^T \nabla f(\hat{x}) \)

and \( T_{\Omega}(\hat{x}) \) is the tangent cone to \( \Omega \) at \( \hat{x} \).
Constrained optimization optimality conditions

Necessary optimality condition

*If* $\hat{x} \in \Omega$ *is a local minimum of a differentiable function* $f$ *over a convex set* $\Omega \subset \mathbb{R}^n$, *then*

$$f'(\hat{x}; d) \geq 0 \quad \forall \ d \in T_{\Omega}(\hat{x}),$$

where $f'(\hat{x}; d) = \lim_{t \to 0} \frac{f(\hat{x} + td) - f(\hat{x})}{t} = d^T \nabla f(\hat{x})$

and $T_{\Omega}(\hat{x})$ *is the tangent cone to* $\Omega$ *at* $\hat{x}$.

Necessary optimality condition

*If* $\hat{x} \in \Omega$ *is a local minimum of the function* $f$ *over the set* $\Omega \subset \mathbb{R}^n$, *then*

$$f^{\circ}(\hat{x}; d) \geq 0 \quad \forall \ d \in T_{\Omega}^H(\hat{x}),$$

where $f^{\circ}(\hat{x}; d)$ *is a generalization of the directional derivative*,

and $T_{\Omega}^H(\hat{x})$ *is a generalization of the tangent cone.*
1. Introduction

2. Generalized pattern search (GPS)
   - Coordinate search
   - Surrogate functions
   - Convergence analysis

3. Mesh adaptive direct search (MADS)

4. A progressive barrier MADS algorithm

5. Some numerical results

6. Discussion
Consider the unconstrained problem

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

where $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$. 
Consider the unconstrained problem

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

where $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$.

**Hypothesis**

*An initial $x_0 \in \mathbb{R}^n$ with $f(x_0) < \infty$ is provided.*
Coordinate search (ancestor of pattern search).

- **INITIALIZATION:**
  - $x_0$ : initial point in $\mathbb{R}^n$ such that $f(x_0) < \infty$
  - $\Delta_0 > 0$ : initial mesh size.
Coordinate search (ancestor of pattern search).

- **Initialization:**
  \[ x_0 : \text{initial point in } \mathbb{R}^n \text{ such that } f(x_0) < \infty \]
  \[ \Delta_0 > 0 : \text{initial mesh size.} \]

- **Poll step:** for \( k = 0, 1, \ldots \)
  If \( f(t) < f(x_k) \) for some \( t \in P_k := \{ x_k \pm \Delta_k e_i : i \in N \} \),
  then set \( x_{k+1} = t \)
  and \( \Delta_{k+1} = \Delta_k \).
Coordinate search (ancestor of pattern search).

- **Initialization:**
  \[ x_0 : \text{initial point in } \mathbb{R}^n \text{ such that } f(x_0) < \infty \]
  \[ \Delta_0 > 0 : \text{initial mesh size.} \]

- **Poll step:** for \( k = 0, 1, \ldots \)
  If \( f(t) < f(x_k) \) for some \( t \in P_k := \{ x_k \pm \Delta_k e_i : i \in \mathbb{N} \} \),
  then set \( x_{k+1} = t \)
  and \( \Delta_{k+1} = \Delta_k \);
  otherwise \( x_k \) is a local minimum with respect to \( P_k \),
  set \( x_{k+1} = x_k \)
  and \( \Delta_{k+1} = \frac{\Delta_k}{2} \).
$x_0 = (2, 2)^T, \Delta_0 = 1$

$f = 4401$
$x_0 = (2,2)^T, \Delta_0 = 1$

$\boldsymbol{f} = 4401$
$x_0 = (2, 2)^T, \Delta_0 = 1$

\[ f = 4401 \]
Coordinate search

\[ x_0 = (2, 2)^T, \Delta_0 = 1 \]

\[
\begin{array}{c}
\begin{array}{ccc}
& & \\
\end{array}
\end{array}
\]

\[ f = 4401 \quad f = 29286 \]
$x_0 = (2, 2)^T, \Delta_0 = 1$

\[
\begin{array}{c}
\begin{array}{c}
f = 4772 \\
f = 4401 \\
f = 29286
\end{array}
\end{array}
\]
Coordinate search

\[ x_0 = (2, 2)^T, \Delta_0 = 1 \]

\[ f = 4772 \]

\[ f = 4401 \]

\[ f = 29286 \]

\[ f = 166 \]
Coordinate search

\[ x_0 = (2, 2)^T, \Delta_0 = 1 \]

\[ f = 4772 \]

\[ f = 4401 \]

\[ f = 29286 \]

\[ f = 166 \]
$x_1 = (1, 2)^T, \Delta_1 = 1$

$\nabla f = 166$
$x_1 = (1,2)^T, \Delta_1 = 1$

\[ f = 166 \]
$x_1 = (1,2)^T, \Delta_1 = 1$

$f = 81, \quad f = 166$
$x_2 = (2,2)^T, \Delta_2 = 1$

$f = 81$
$x_2 = (2, 2)^T, \Delta_2 = 1$

$f = 2646$  
$f = 81$
Coordinate search

\[ x_2 = (2, 2)^T, \Delta_2 = 1 \]

\[ f = 2646 \quad f = 81 \quad f = 166 \]
$x_2=(2,2)^T, \Delta_2=1$

$f=2646$  $f=81$  $f=166$  $f=152$
Coordinate search

\[ x_2 = (2, 2)^T, \Delta_2 = 1 \]

\[ f = 2646 \quad f = 81 \quad f = 166 \quad f = 36 \]
$x_3 = (0, 1)^T, \Delta_3 = 1$

$f = 36$
$x_3 = (0, 1)^T, \Delta_3 = 1$
\[ x_4 = (0,0)^T, \Delta_4 = 1 \]

\[ f = 17 \]
Coordinate search

\[ x_4 = (0, 0)^T, \Delta_4 = 1 \]
$x_5 = (0,0)^T, \Delta_5 = \frac{1}{2}$
Coordinate search

\( x_5 = (0,0)^T, \Delta_5 = \frac{1}{2} \)

\[ f = 17 \]
$x_5 = (0,0)^T, \Delta_5 = \frac{1}{2}$

$T$, $\Delta_5 = \frac{1}{2}$

- $f = 40$
- $f = 17$
- $f = 2$
- $f = 18$
Initialization:

- $x_0$: initial point in $\mathbb{R}^n$ such that $f(x_0) < \infty$
- $\Delta_0 > 0$: initial mesh size.
- $D$: finite positive spanning set of directions.
Generalized pattern search

- **Initialization:**
  \[ x_0 : \text{initial point in } \mathbb{R}^n \text{ such that } f(x_0) < \infty \]
  \[ \Delta_0 > 0 : \text{initial mesh size.} \]
  \[ D : \text{finite positive spanning set of directions}. \]

- For \( k = 0, 1, \ldots \)
  
  - **Search step:** Evaluate \( f \) at a finite number of mesh points.
  
  - **Poll step:** If the search failed, evaluate \( f \) at the poll points
    \( \{x_k + \Delta_k d : d \in D_k\} \) where \( D_k \subset D \) is a positive spanning set.
  
  - **Parameter update:**
    Set \( \Delta_{k+1} < \Delta_k \) if no new incumbent was found,
    otherwise set \( \Delta_{k+1} \geq \Delta_k \), and call \( x_{k+1} \) the new incumbent.
Generalized pattern search

\[ x_0 = (2,2)^T, \Delta_0 = 1 \]

\[ f = 4401 \]
$x_0 = (2,2)^T, \Delta_0 = 1$

$f = 4401$
$x_0 = (2,2)^T, \Delta_0 = 1$
$x_0 = (2, 2)^T, \Delta_0 = 1$

\[ f = 4401 \]

\[ f = 4772 \]
Generalized pattern search

\[ x_0 = (2, 2)^T, \Delta_0 = 1 \]
$x_1 = (1,2)^T, \Delta_1 = 2$
Generalized pattern search

\[ x_1 = (1,2)^T, \Delta_1 = 2 \]
Generalized pattern search

\[ x_1 = (1, 2)^T, \Delta_1 = 2 \]

\[ f = 2341 \]

\[ f = 101 \]

\[ f = 967 \]

\[ f = 150 \]
Generalized pattern search

\[ x_2 = (2, 2)^T, \Delta_2 = 1 \]

\[ f = 101 \]
Generalized pattern search

\[ x_2 = (2, 2)^T, \Delta_2 = 1 \]

\[ f = 101 \]
Surrogate functions within the search step

\[ f(x_1, x_2) \]

\* is the incumbent solution
Surrogate functions within the search step

\[ f(x_1, x_2) \]

* is the incumbent solution

Charles Audet (Optimization 2007)
Surrogate functions within the search step

\[ f(x_1, x_2) \]

* is the incumbent solution
Surrogate functions within the search step

\[ f(x_1, x_2) \]

* is the incumbent solution
Surrogate functions within the search step

\[ f(x_1, x_2) \]

* is the incumbent solution

Trial point produced using the surrogate
Surrogate functions within the search step

\[ f(x_1, x_2) \]

\( \star \) is the incumbent solution

\( f \) is evaluated at the trial point
Surrogate functions within the search step

\[ f(x_1, x_2) \]

\(*\) is the incumbent solution

Poll around the incumbent
Surrogate functions within the search step

\[ f(x_1, x_2) \]

\* is the incumbent solution

Poll around the incumbent

Charles Audet (Optimization 2007)
If $f$ is Lipschitz$^1$ near $x \in \mathbb{R}^n$, then the Clarke generalized derivative of $f$ at $x$ in the direction $v \in \mathbb{R}^n$ is

$$f^\circ(x; v) = \limsup_{y \to x, \ t \downarrow 0} \frac{f(y + tv) - f(y)}{t}.$$ 

$^1$i.e., there exists a scalar $K$ such that

$$|f(x') - f(x'')| \leq K\|x' - x''\|$$

for every $x', x''$ in some neighborhood of $x$. 
Clarke derivatives and generalized gradient

If $f$ is Lipschitz\(^1\) near $x \in \mathbb{R}^n$, then the Clarke generalized derivative of $f$ at $x$ in the direction $v \in \mathbb{R}^n$ is

$$f^\circ(x; v) = \limsup_{y \to x, \ t \downarrow 0} \frac{f(y + tv) - f(y)}{t}.$$

The generalized gradient of $f$ at $x$ is defined to be

$$\partial f(x) = \{ \zeta \in \mathbb{R}^n : f^\circ(x; v) \geq v^T \zeta \text{ for every } v \in \mathbb{R}^n \}$$

\(^1\)i.e., there exists a scalar $K$ such that

$$|f(x') - f(x'')| \leq K \|x' - x''\|$$

for every $x', x''$ in some neighborhood of $x$. 
Clarke derivatives and generalized gradient

If \( f \) is Lipschitz\(^1 \) near \( x \in \mathbb{R}^n \), then the Clarke generalized derivative of \( f \) at \( x \) in the direction \( v \in \mathbb{R}^n \) is

\[
\left( f \circ (x; v) \right) = \limsup_{y \to x, \; t \downarrow 0} \frac{f(y + tv) - f(y)}{t}.
\]

The generalized gradient of \( f \) at \( x \) is defined to be

\[
\partial f(x) = \{ \zeta \in \mathbb{R}^n : f \circ (x; v) \geq v^T \zeta \text{ for every } v \in \mathbb{R}^n \}
\]

\[
= \text{co}\{ \lim \nabla f(x_i) : x_i \to x \text{ and } \nabla f(x_i) \text{ exists} \}.
\]

\(^1\)i.e., there exists a scalar \( K \) such that

\[
|f(x') - f(x'')| \leq K\|x' - x''\|
\]

for every \( x', x'' \) in some neighborhood of \( x \).
The generalized derivative may be obtained from the generalized gradient: \( f^\circ (x; v) = \max \{ v^T \zeta : \zeta \in \partial f(x) \} \).

If \( f \) is convex, \( \partial f(x) = \) sub-gradient.

\( f \) is regular at \( x \) if for any \( v \in \mathbb{R}^n \), the directional derivative \( f'(x; v) \) exists and equals \( f^\circ (x; v) \).

\( f \) is strictly differentiable at \( x \) if \( \partial f(x) \) contains a single element, and that element is \( \nabla f(x) \).
Clarke calculus

- The generalized derivative may be obtained from the generalized gradient: 
  \[ f^\circ(x; v) = \max \{ v^T \zeta : \zeta \in \partial f(x) \} \].

- If \( f \) is convex, \( \partial f(x) = \) sub-gradient.

- \( f \) is regular at \( x \) if for any \( v \in \mathbb{R}^n \), the directional derivative \( f'(x; v) \) exists and equals \( f^\circ(x; v) \).

- \( f \) is strictly differentiable at \( x \) if \( \partial f(x) \) contains a single element, and that element is \( \nabla f(x) \).

### Necessary optimality condition

If \( x \) is a local minimizer of \( f \), and \( f \) is Lipschitz near \( x \), then \( 0 \in \partial f(x) \).

This generalizes the first-order \( C^1 \) condition: \( 0 = \nabla f(x) \).
Differentiable, but not strictly differentiable

\[ y = x^2(2+\sin(\pi/x)) \]

\( f \) is Lipschitz and differentiable near 0:

\[ y'(0) = 0 \quad \text{and} \quad y' = 2x(2+\sin(\pi/x)) - \pi \cos(\pi/x) \]

The derivative is not continuous at \( x = 0 \):

\[ y'(\frac{1}{2k}) = \frac{2}{k} - \pi \]

\( \partial f(0) = [-\pi, \pi] \)

\( f \) is not strictly differentiable:

\[ f^\circ(0, \pm 1) = \pi \neq f'(0, \pm 1) = 0 \]

\( f \) is not regular:
Strictly, but not continuously differentiable

\[ f(x) = \int_0^x \varphi(u) du \quad \text{où} \quad \varphi(u) = \begin{cases} 
  u & \text{si} \ u \leq 0 \\
  \frac{1}{1+\kappa} & \text{si} \ \kappa + 1 > \frac{1}{u} \geq \kappa
\end{cases} \]

\( f \) is Lipschitz near \( \hat{x} = 0 \), and \( \partial f(\frac{1}{\kappa}) = [\frac{1}{\kappa+1}, \frac{1}{\kappa}] \)

\( f \) is not strictly differentiable, nor continuously differentiable, in any neighbourhood of \( \hat{x} = 0 \).

\( \partial f(0) = \{0\} \), and therefore \( f \) is strictly differentiable at \( \hat{x} = 0 \).
The series of iterates \( \{x_k\} \) all belong to some bounded set.

- there exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[ f(x_k + \Delta_k d) \geq f(x_k) \text{ for all } d \in D_k, \text{ with } x_k \to \hat{x}, \Delta_k \downarrow 0. \]
GPS convergence analysis

Hypothesis

The series of iterates \( \{x_k\} \) all belong to some bounded set.

- there exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[
f(x_k + \Delta_k d) \geq f(x_k) \quad \text{for all } d \in D_k, \quad \text{with } x_k \to \hat{x}, \Delta_k \downarrow 0.
\]

Theorem (Main result for our analysis)

If \( f \) is Lipschitz near \( \hat{x} \),
then \( f^\circ(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.
Hypothesis

The series of iterates \( \{x_k\} \) all belong to some bounded set.

- there exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[
  f(x_k + \Delta_k d) \geq f(x_k) \quad \text{for all } d \in D_k, \text{ with } x_k \to \hat{x}, \Delta_k \downarrow 0.
  \]

Theorem (Main result for our analysis)

If \( f \) is Lipschitz near \( \hat{x} \),

then \( f^\circ(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.

**Proof:** \( f^\circ(\hat{x}; d) := \limsup_{y \to \hat{x}, t \downarrow 0} \frac{f(y + td) - f(y)}{t} \)
Hypothesis

The series of iterates \( \{x_k\} \) all belong to some bounded set.

- there exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[
  f(x_k + \Delta_k d) \geq f(x_k) \quad \text{for all } d \in D_k, \text{ with } x_k \to \hat{x}, \Delta_k \downarrow 0.
  \]

Theorem (Main result for our analysis)

If \( f \) is Lipschitz near \( \hat{x} \),

then \( f^\circ(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.

Proof:

\[
 f^\circ(\hat{x}; d) := \limsup_{t \downarrow 0} \frac{f(y + td) - f(y)}{t} \geq \limsup_{k \in K} \frac{f(x_k + \Delta_k d) - f(x_k)}{\Delta_k}
\]
GPS convergence analysis

Hypothesis

The series of iterates \( \{x_k\} \) all belong to some bounded set.

- There exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[
  f(x_k + \Delta_k d) \geq f(x_k) \quad \text{for all} \quad d \in D_k, \quad \text{with} \quad x_k \to \hat{x}, \quad \Delta_k \downarrow 0.
  \]

Theorem (Main result for our analysis)

If \( f \) is Lipschitz near \( \hat{x} \),

then \( f^\circ(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.

Proof:

\[
\begin{align*}
  f^\circ(\hat{x}; d) & := \limsup_{y \to \hat{x}, \ t \downarrow 0} \frac{f(y + td) - f(y)}{t} \\
 & \geq \limsup_{k \in K} \frac{f(x_k + \Delta_k d) - f(x_k)}{\Delta_k} \geq 0.
\end{align*}
\]

Note: This set of direction is a positive spanning set.
Hypothesis

The series of iterates \( \{x_k\} \) all belong to some bounded set.

- there exists some \( \hat{x} \in \mathbb{R}^n \), the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine:
  \[
  f(x_k + \Delta_k d) \geq f(x_k) \quad \text{for all } d \in D_k, \text{ with } x_k \to \hat{x}, \Delta_k \downarrow 0.
  \]

Theorem (Hierarchy of convergence results)

If \( f \) is Lipschitz near \( \hat{x} \),
then \( f^\circ(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.

If \( f \) is regular near \( \hat{x} \),
then \( f'(\hat{x}; d) \geq 0 \) for each directions \( d \in D \) used infinitely often.

If \( f \) is strictly differentiable near \( \hat{x} \), then \( \nabla f(\hat{x}) = 0 \).
1 Introduction

2 Generalized pattern search (GPS)

3 Mesh adaptive direct search (MADS)
   - Limitations of GPS
   - Barrier approach for constraints
   - Convergence analysis - extreme barrier

4 A progressive barrier MADS algorithm

5 Some numerical results

6 Discussion
Limitations of GPS

\[ f(x) = \|x\|_\infty \text{ with } x_0 = (1, 1)^T. \]

Level set:
\[ \{ x \in \mathbb{R}^2 : f(x) = 1 \} \]

Mesh adaptive direct search (MADS)
Limitations of GPS

\[ f(x) = \|x\|_\infty \text{ with } x_0 = (1, 1)^T. \]

Level set:
\[ \{ x \in \mathbb{R}^2 : f(x) = 1 \} \]

\[ x_0 = (1, 1)^T \]

A directional algorithm
Limitations of GPS

\[ f(a, b) = a + b \text{ with } x_0 = (1, 1)^T. \]

Domain:
\[ \{x \in \mathbb{R}^2 : f(x) = 1\} \]

\[ x_0 = (1, 1)^T \]
Limitations of GPS

\[ f(a, b) = a + b \text{ with } x_0 = (1, 1)^T. \]

Domain:
\[ \{ x \in \mathbb{R}^2 : f(x) = 1 \} \]

All poll points are infeasible
Limitations of GPS

- In order for the poll to escape a non-optimal point, it has to generate some trial points inside the tangent cone to $\Omega$. 
In order for the poll to escape a non-optimal point, it has to generate some trial points inside the tangent cone to $\Omega$.

It is easy to identify the tangent cone generators when $\Omega$ is a polyhedron: $\{T_\Omega(x) : x \in \Omega\}$ contains a finite number of generators. The poll direction in $D$ can contain these directions.
In order for the poll to escape a non-optimal point, it has to generate some trial points inside the tangent cone to $\Omega$.

It is easy to identify the tangent cone generators when $\Omega$ is a polyhedron: $\{T_{\Omega}(x) : x \in \Omega\}$ contains a finite number of generators. The poll direction in $D$ can contain these directions.

In presence of general constraints, $\{T_{\Omega}(x) : x \in \Omega\}$ usually contain an infinite number of cones: It is therefore impossible to generate all generators.
MADS – Barrier approach for constraints

\[ x_k \]

\[ \Omega \]
MADS – Barrier approach for constraints

Search

\[ x_k \]

\[ \Omega \]
MADS – Barrier approach for constraints

Search
Poll

\[ \Omega \]

Mesh adaptive direct search (MADS)

Charles Audet (Optimization 2007)
MADS – Barrier approach for constraints

$$x_{k+1} = p^2$$
MADS – Barrier approach for constraints

Search

Poll

Success

\[ x_{k+1} = p^2 \]
Mesh local optimizer

\[ x_{k+1} = x_k \]
MADS – Barrier approach for constraints

Mesh local optimizer

\[ x_{k+1} = x_k \]

Charles Audet (Optimization 2007)
Convergence analysis - LTMAD barrier

**Theorem**

As \( k \to \infty \), the set of LTMADS normalized poll directions is dense in the unit sphere, with probability 1.
As $k \to \infty$, the set of LTMADS normalized poll directions is dense in the unit sphere, with probability 1.

Let $\hat{x}$ be the limit of a subsequence of mesh local optimizers on meshes that get infinitely fine. If $f$ is Lipschitz near $\hat{x}$, then $f^\circ(\hat{x}, v) \geq 0$ for all $v \in T^H_{\Omega}(\hat{x})$ with probability 1.
Presentation Outline

1. Introduction
   - Target optimization problems
   - Direct search methods
   - Nonsmooth optimality conditions

2. Generalized pattern search (GPS)
   - Coordinate search
   - Surrogate functions
   - Convergence analysis

3. Mesh adaptive direct search (MADS)
   - Limitations of GPS
   - Barrier approach for constraints
   - Convergence analysis - extreme barrier

4. A progressive barrier MADS algorithm
   - Open and closed constraints
   - A progressive barrier MADS algorithm
   - Convergence analysis - progressive barrier

5. Some numerical results
   - The disk problem
   - The crescent problem
MADS treats the constraints by the aggressive 'extreme barrier' approach.
All infeasible trial points are rejected from consideration.
MADS treats the constraints by the aggressive ‘extreme barrier’ approach. All infeasible trial points are rejected from consideration.

We now propose an alternate way to deal with open constraints.

\[
\min_{x \in \Omega} f(x)
\]
Optimization format for NOMAD

MADS treats the constraints by the aggressive ‘extreme barrier’ approach.
All infeasible trial points are rejected from consideration.

We now propose an alternate way to deal with open constraints.

\[
\min_{x \in \Omega} f(x)
\]

where \( \Omega \equiv \{x \in X : C(x) \leq 0\} \subset \mathbb{R}^n \).
The constraints are partitioned into two groups.

- \( X \) contains the closed constraints.
- \( C(x) \leq 0 \) are called the open constraints.
Consider the toy problem:

\[
\begin{align*}
\min_{x \in \mathbb{R}^2} & \quad x_1^2 - \sqrt{x_2} \\
\text{s.t.} & \quad -x_1^2 + x_2^2 \leq 1 \\
& \quad x_2 \geq 0
\end{align*}
\]
Consider the toy problem:

\[
\begin{align*}
\min_{x \in \mathbb{R}^2} & \quad x_1^2 - \sqrt{x_2} \\
\text{s.t.} & \quad -x_1^2 + x_2^2 \leq 1 \\
& \quad x_2 \geq 0
\end{align*}
\]

- Closed constraints *must* be satisfied at every trial vector of decision variables in order for the functions to evaluate.
Consider the toy problem:

\[
\begin{align*}
\min_{x \in \mathbb{R}^2} & \quad x_1^2 - \sqrt{x_2} \\
\text{s.t.} & \quad -x_1^2 + x_2^2 \leq 1 \\
& \quad x_2 \geq 0
\end{align*}
\]

- Closed constraints *must* be satisfied at every trial vector of decision variables in order for the functions to evaluate. Here \( x_2 \geq 0 \) is a closed constraint, because if it is violated, the objective function will fail.
Consider the toy problem:

\[
\begin{align*}
\min_{x \in \mathbb{R}^2} & \quad x_1^2 - \sqrt{x_2} \\
\text{s.t.} & \quad -x_1^2 + x_2^2 \leq 1 \\
& \quad x_2 \geq 0
\end{align*}
\]

- **Closed constraints** *must* be satisfied at every trial vector of decision variables in order for the functions to evaluate. Here \( x_2 \geq 0 \) is a closed constraint, because if it is violated, the objective function will fail.
- **Open constraints** must be satisfied at the solution, but an optimization algorithm may use some trial points that violate it. Here \(-x_1^2 + x_2^2 \leq 1\) is an open constraint.
Open, closed and hidden constraints

Consider the toy problem:

\[
\begin{align*}
\min_{x \in \mathbb{R}^2} & \quad x_1^2 - \ln(x_2) \\
\text{s.t.} & \quad -x_1^2 + x_2^2 \leq 1 \\
& \quad x_2 \geq 0
\end{align*}
\]

- Closed constraints must be satisfied at every trial vector of decision variables in order for the functions to evaluate. Here \(x_2 \geq 0\) is a closed constraint, because if it is violated, the objective function will fail.

- Open constraints must be satisfied at the solution, but an optimization algorithm may use some trial points that violate it. Here \(-x_1^2 + x_2^2 \leq 1\) is an open constraint.

- Lets change the objective. \(x_2 \neq 0\) is now an hidden constraint. \(f\) is set to \(\infty\) when \(x \in \Omega\) but \(x\) fails to satisfy an hidden constraint.
\[
\min_{x \in \Omega} f(x)
\]

The extreme barrier handles the closed and hidden constraints \( X \). A filter handles \( C(x) \leq 0 \).
Filter approach to constraints (Based on Fletcher - Leyffer)

\[
\min_{x \in \Omega} f(x)
\]

The extreme barrier handles the closed and hidden constraints \( X \). A filter handles \( C(x) \leq 0 \).

Define the nonnegative constraint violation function

\[
h(x) := \begin{cases} 
\sum_j \max(0, c_j(x))^2 & \text{if } x \in X \text{ and } f(x) < \infty, \\
+\infty & \text{otherwise.}
\end{cases} 
\]

\( h(x) = 0 \) if and only if \( x \in \Omega \).
Filter approach to constraints
Filter approach to constraints

\[ \Omega \subset X \]

Optimal solution

\[ h = 0 \]

\[ h = 3 \]

\[ h = 2 \]

Local min of \( h \)

Charles Audet (Optimization 2007)

A progressive barrier MADS algorithm
Filter approach to constraints

Optimal solution

$h = 0$

$h = 3$

$h = 2$

Local min of $h$

Charles Audet (Optimization 2007)
Filter approach to constraints

\[ \Omega \times X \]

Optimal solution

\[ h = 0 \]

\[ h = 3 \]

\[ h = 2 \]

Local min of \( h \)

\[ (h(x^I, f(x^I)) \]

A progressive barrier MADS algorithm
Filter approach to constraints

\[ f(x) = \begin{cases} 
0 & \text{if } h(x) < 0 \\
\infty & \text{otherwise}
\end{cases} \]

\[ h(x) = \min \{ h(x^I, f(x^I)) \} \]

Optimal solution:
- \( h = 0 \)
- \( h = 3 \)
- \( h = 2 \)
Filter approach to constraints

\[ f(x) = h(x) \]

Optimal solution

- \( h = 0 \)
- \( h = 3 \)
- \( h = 2 \)

Local min of \( h \)

Charles Audet (Optimization 2007)
Filter approach to constraints

\[ f(X), h(X) \]

Optimal solution

\[ h = 0 \]

\[ h = 3 \]

\[ h = 2 \]
Filter approach to constraints

\[ \Omega \]

\[ X \]

\[ s \]

\[ f - h \]

\[ s (h(X), f(X)) \]

\[ s (h(\Omega), f(\Omega)) \]

Optimal solution

\[ h = 0 \]

\[ h = 3 \]

\[ h = 2 \]

Local min of

Charles Audet (Optimization 2007)
Filter approach to constraints

Optimal solution
Filter approach to constraints

Optimal solution

$h = 0$

$h = 2$

$h = 3$

Optimal solution

$\Omega$

$X$

$f$

$h$

Optimal solution
Filter approach to constraints

Optimal solution

Local min of $h$

$h=0$

$h=2$

$h=3$

$\Omega$

$x^F$

$x_I$

$X$

A progressive barrier MADS algorithm

Charles Audet (Optimization 2007)
We present an algorithm that

- At iteration $k$, any trial point whose constraint violation value exceeds the value $h_{k}^{\text{max}}$ is discarded from consideration.
We present an algorithm that

- At iteration $k$, any trial point whose constraint violation value exceeds the value $h_{k}^{\text{max}}$ is discarded from consideration.
- As $k$ increases, the threshold $h_{k}^{\text{max}}$ is reduced.
We present an algorithm that

- At iteration $k$, any trial point whose constraint violation value exceeds the value $h_k^{\text{max}}$ is discarded from consideration.
- As $k$ increases, the threshold $h_k^{\text{max}}$ is reduced.
- The algorithm accepts some trial points that violate the open constraints.
- Two types of poll centers: the feasible and infeasible ones.
Progressive barrier algorithm: Iteration 0

\[ x_0 \bullet \]

\[ \Omega \]

\[ X \]
Progressive barrier algorithm: Iteration 0

\[ \Omega \]

\[ X \]

\[ x_0 \]

\[ f \]

\[ h^{\max} \]

\[ h \]
Progressive barrier algorithm: Iteration 0

\[ x_0 \in X \]

\[ \Omega \]

\[ h_0^{\text{max}} \]

\[ f \]

Charles Audet (Optimization 2007)
A progressive barrier MADS algorithm
Worst $h$ and worst $f$
Outside closed constraints $X$: reject point (barrier approach)
Better $h$ but worst $f$
Better $h$ and better $f$
The new infeasible incumbent is the one to the left (in \((h, f)\) plot) of the last one with the best \(f\) value.
Progressive barrier algorithm: Iteration 1

\[ x_1 \]

\[ \Omega \]

\[ X \]

\[ f \]

\[ h_{\text{max}} \]

\[ h \]
Progressive barrier algorithm: Iteration 1

A progressive barrier MADS algorithm
Progressive barrier algorithm: Iteration 1

New infeasible incumbent
Progressive barrier algorithm : Iteration 1

New feasible incumbent

Charles Audet (Optimization 2007)
Progressive barrier algorithm : Iteration 2

\[ f - h \]

\[ x_2^I \]

\[ x_2^F \]

\[ h_{2\text{max}} \]
Progressive barrier algorithm: Iteration 2

A progressive barrier MADS algorithm
Progressive barrier algorithm: Iteration 2

Primary poll center

The diagram shows a geometric representation of a progressive barrier algorithm. The plot features a region marked with an inequality constraint, illustrating the poll center and its implications on the feasible set $X$ and the barrier function $h$.
Secondary poll center

\[ x^I \]

\[ x^F \]

\[ h_{2_{\text{max}}} \]
Unsuccessful iteration: All \((h, f)\) trial values are in shaded region.
$x_k^I$ is the primary POLL center since $f(x_k^I) < f(x_k^F) - \rho$

$\rho > 0$ is chosen by the user
Primary and secondary POLL centers

\[ x_k^F \] would be the primary POLL center
if \[ f(x_k^I) \geq f(x_k^F) - \rho \]

\( \rho > 0 \) is chosen by the user
Iteration $k$ is initiated with a feasible incumbent $x_0^F \in \Omega$ and/or an infeasible incumbent $x_k^I$. 
Iteration $k$ is initiated with a feasible incumbent $x_0^F \in \Omega$ and/or an infeasible incumbent $x_k^I$.

**Search:** Employ some finite strategy to try to identify a new incumbent mesh point.

**Poll:** If the Search failed, then $f$ must be evaluated at neighbouring trial points.

**Update:** If the Search or Poll was successful, then increase (or preserve) the mesh size, decrease (if possible) the barrier threshold, and restart from improved point. Else refine the mesh and restart from same point. Trial points with $h(x) > h_{\text{max}}^k$ are discarded from consideration.
Iteration $k$ is initiated with a feasible incumbent $x_0^F \in \Omega$ and/or an infeasible incumbent $x_k^I$.

**Search:** Employ some finite strategy to try to identify a new incumbent mesh point.

**Poll:** If the **Search** failed, then $f$ must be evaluated at neighbouring trial points.
Iteration $k$ is initiated with a feasible incumbent $x^{F}_0 \in \Omega$ and/or an infeasible incumbent $x^{I}_k$.

**Search:** Employ some finite strategy to try to identify a new incumbent mesh point.

**Poll:** If the Search failed, then $f$ must be evaluated at neighbouring trial points.

**Update:** If the Search or Poll was successful, then increase (or preserve) the mesh size, decrease (if possible) the barrier threshold, and restart from improved point.
Iteration $k$ is initiated with a feasible incumbent $x^F_0 \in \Omega$ and/or an infeasible incumbent $x^I_k$.

**Search:** Employ some finite strategy to try to identify a new incumbent mesh point.

**Poll:** If the Search failed, then $f$ must be evaluated at neighbouring trial points.

**Update:** If the Search or Poll was successful,

- then increase (or preserve) the mesh size,
- decrease (if possible) the barrier threshold,
- and restart from improved point.

Else refine the mesh and restart from same point.
Iteration $k$ is initiated with a feasible incumbent $x^F_0 \in \Omega$ and/or an infeasible incumbent $x^I_k$.

**Search:** Employ some finite strategy to try to identify a new incumbent mesh point.

**Poll:** If the Search failed, then $f$ must be evaluated at neighbouring trial points.

**Update:** If the Search or Poll was successful,

- then increase (or preserve) the mesh size,
- decrease (if possible) the barrier threshold,
- and restart from improved point.

Else refine the mesh and restart from same point.

Trial points with $h(x) > h^\text{max}_k$ are discarded from consideration.
Parameter update rules

**Dominating iteration**

- Mesh size increases or is unaltered
- \( h_{k+1}^{\text{max}} = h_k^I \)

**Improving iteration**

- Mesh size is unaltered
- \( h_{k+1}^{\text{max}} < h_k^I \)

**Unsuccessful iteration**

- Mesh size decreases
- \( h_{k+1}^{\text{max}} = h_k^I \)
Convergence analysis

Assumptions

- At least one initial point in \( X \) is provided – but not required to be in \( \Omega \).
- All iterates belong to some compact set – it is sufficient to assume that level sets of \( f \) in \( X \) are bounded.
Convergence analysis

Assumptions

- At least one initial point in $X$ is provided – but not required to be in $\Omega$.
- All iterates belong to some compact set – it is sufficient to assume that level sets of $f$ in $X$ are bounded.

Theorem

$As \, k \to \infty$, MADS’s normalized polling directions form a dense set in the unit sphere (with probability 1).
Theorem

Let $\hat{x} \in \Omega$ be the limit of unsuccessful feasible poll centers $\{x^F_k\}$ on meshes that get infinitely fine. If $f$ is Lipschitz near $\hat{x}$, then $f^\circ(\hat{x}, v) \geq 0$ for all $v \in T^H_\Omega(\hat{x})$ (with probability 1).
Limit of feasible POLL centers

**Theorem**

Let \( \hat{x} \in \Omega \) be the limit of unsuccessful feasible poll centers \( \{x^F_k\} \) on meshes that get infinitely fine. If \( f \) is Lipschitz near \( \hat{x} \), then \( f^\circ(\hat{x}, v) \geq 0 \) for all \( v \in T_H^{\hat{x}}(x) \) (with probability 1).

**Corollary**

In addition, if \( f \) is strictly differentiable near \( \hat{x} \), and if \( \Omega \) is regular near \( \hat{x} \), then \( f'(\hat{x}, v) \geq 0 \) for all \( v \in T^{\hat{x}}(x) \) (with probability 1), i.e., \( \hat{x} \) is a KKT point for \( \min_{x \in \Omega} f(x) \).
Limit of infeasible POLL centers

Theorem

Let \( \hat{x} \in X \) be the limit of unsuccessful infeasible poll centers \( \{x^I_k\} \) on meshes that get infinitely fine. If \( h \) is Lipschitz near \( \hat{x} \), then \( h(\hat{x}, v) \geq 0 \) for all \( v \in T^H_X(\hat{x}) \) (with probability 1).
Limit of infeasible POLL centers

**Theorem**

Let $\hat{x} \in X$ be the limit of unsuccessful infeasible poll centers $\{x^I_k\}$ on meshes that get infinitely fine. If $h$ is Lipschitz near $\hat{x}$, then $h^\circ(\hat{x}, v) \geq 0$ for all $v \in T^H_X(\hat{x})$ (with probability 1).

**Corollary**

In addition, if $h$ is strictly differentiable near $\hat{x}$, and if $X$ is regular near $\hat{x}$, then $h'(\hat{x}, v) \geq 0$ for all $v \in T_X(\hat{x})$ (with probability 1), i.e., $\hat{x}$ is a KKT point for $\min_{x \in X} h(x)$.
A constraint qualification

**Theorem**

Let \( \hat{x} \in X \) be the limit of unsuccessful infeasible poll centers \( \{x^I_k\} \) on meshes that get infinitely fine. Consider the case where \( x \in \Omega \) and for every \( v \in T^H_\Omega(\hat{x}) \neq \emptyset \), there exists an \( \epsilon > 0 \) for which

\[
h^\circ(x; v) < 0 \text{ for all } x \in X \cap B_\epsilon(\hat{x}) \text{ such that } h(x) > 0.
\]

If \( h \) is Lipschitz near \( \hat{x} \), then \( f^\circ(\hat{x}, v) \geq 0 \) for all \( v \in T^H_\Omega(\hat{x}) \) (with probability 1).
A constraint qualification

**Theorem**

Let \( \hat{x} \in X \) be the limit of unsuccessful infeasible poll centers \( \{x^I_k\} \) on meshes that get infinitely fine. **Consider the case where** \( x \in \Omega \) and for every \( v \in T^H_{\Omega}(\hat{x}) \neq \emptyset \), there exists an \( \epsilon > 0 \) for which \( h^\circ(x;v) < 0 \) for all \( x \in X \cap B_\epsilon(\hat{x}) \) such that \( h(x) > 0 \).

If \( h \) is Lipschitz near \( \hat{x} \), then \( f^\circ(\hat{x},v) \geq 0 \) for all \( v \in T^H_{\Omega}(\hat{x}) \) (with probability 1).

**Corollary**

In addition, if \( f \) is strictly differentiable near \( \hat{x} \), and if \( \Omega \) is regular near \( \hat{x} \), then \( f'(\hat{x},v) \geq 0 \) for all \( v \in T_{\Omega}(\hat{x}) \) (with probability 1), i.e., \( \hat{x} \) is a KKT point for \( \min_{x \in \Omega} f(x) \).
Presentation outline

1 Introduction

2 Generalized pattern search (GPS)

3 Mesh adaptive direct search (MADS)

4 A progressive barrier MADS algorithm

5 Some numerical results
   - The disk problem
   - The crescent problem
   - Styrene production process

6 Discussion
This is a nice and fat convex problem.

\[
\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad \sum_{i=1}^{n} x_i \\
\text{s.t.} & \quad \sum_{i=1}^{n} x_i^2 \leq 3n,
\end{align*}
\]

Starting points:
- Feasible: \((0, 0, 0, \ldots, 0, 0)\)
- Infeasible: \((3, 3, 3, \ldots, 3, 3)\).
Linear optimization over a disk in $\mathbb{R}^n$

This is a nice and fat convex problem.

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{n} x_i \quad \text{s.t.} \quad \sum_{i=1}^{n} x_i^2 \leq 3n,$$

Starting points:

- Feasible $(0, 0, 0, \ldots, 0, 0)$
- Infeasible $(3, 3, 3, \ldots, 3, 3)$

This problem is difficult for GPS. GPS produces iterates that get stuck on the boundary of the domain. LTMADS is able to escape from these points.
Linear optimization over a disk in $\mathbb{R}^n$

This is a nice and fat convex problem.

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{n} x_i$$

s.t. $\sum_{i=1}^{n} x_i^2 \leq 3n$,

Starting points:

Feasible (0, 0, 0, ..., 0, 0)

Infeasible (3, 3, 3, ..., 3, 3).

This problem is difficult for GPS.

GPS produces iterates that get stuck on the boundary of the domain. LTMADS is able to escape from these points.

2-Phase approach for barrier approach from infeasible starting point:

- Solve $\min_{x \in X} h(x)$ and stop when $h(x_k) = 0$.
- Then solve $\min_{x \in \Omega} f(x)$ with starting point $x_k$. 
Results for the disk problem - feasible starting point

$n=5$

$n=10$

$n=20$

$n=50$

Some numerical results

Charles Audet (Optimization 2007)
Results for the disk problem - infeasible starting point

For different values of \(n\), the following graphs show the number of evaluations required to reach a certain function value as a function of the number of evaluations.

- \(n=5\)
- \(n=10\)
- \(n=20\)
- \(n=50\)

The graphs illustrate the convergence of the optimization methods for different numbers of evaluations.

- **O GPS–EB**
- **GPS filter**
- **LTMADS–EB**
- **LTMADS–PB 1**
- **LTMADS–PB 2**

The graphs show the function values decreasing with increasing number of evaluations, indicating the effectiveness of the optimization methods.
Linear optimization over a crescent in $\mathbb{R}^n$

\[
\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad x_n \\
\text{s.t.} & \quad \sum_{i=1}^{n} (x_i - 1)^2 \leq n^2 \leq \sum_{i=1}^{n} (x_i + 1)^2,
\end{align*}
\]

Starting points:
- Feasible: $(n, 0, 0, \ldots, 0, 0)$
- Infeasible: $(n, 0, 0, \ldots, 0, -n)$.

There is a single optimal solution to that problem:
\[x^* = (1, 1, 1, \ldots, 1, 1 - n)^T\] with \(f(x^*) = 1 - n\).
Results for the crescent problem - feasible starting point

Some numerical results
Results for the crescent problem - infeasible starting point

$n=5$

$n=10$

$n=20$

$n=50$
### Table of results for the crescent problem

<table>
<thead>
<tr>
<th>n</th>
<th>LTMADS-EB</th>
<th></th>
<th>LTMADS-PB</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First feasible sol</td>
<td>Best sol</td>
<td>First feasible sol</td>
<td>Best sol</td>
</tr>
<tr>
<td></td>
<td># evals</td>
<td>f-value</td>
<td># evals</td>
<td>f-value</td>
</tr>
<tr>
<td>5</td>
<td>18.8</td>
<td>-0.4</td>
<td>-3.766</td>
<td>29.0</td>
</tr>
<tr>
<td>10</td>
<td>49.0</td>
<td>-1.4</td>
<td>-8.896</td>
<td>76.2</td>
</tr>
<tr>
<td>20</td>
<td>181.2</td>
<td>-8.0</td>
<td>-18.680</td>
<td>374.2</td>
</tr>
<tr>
<td>50</td>
<td>1004.0</td>
<td>-22.2</td>
<td>-45.902</td>
<td>2401.9</td>
</tr>
</tbody>
</table>

- EB on $\min_x h(x)$ finds rapidly a feasible solution. But $f(x)$ at that solution is usually large.
- PB takes twice as many evaluations to reach feasibility. But $f(x)$ is usually very good (for $n = 50$, the first feasible PB solution is better than EB after 30000 evaluations).
EB converges to a local optimizer and PB to the global solution.
Optimization of a styrene production process

8 continuous variables    4 closed yes-no constraints
1-3 sec per evaluations    7 open constraints
may fail to evaluate
We have presented some classes of direct search algorithms. Used in practice by our collaborators (Boeing, Exxon, GM,...)

Constraints may be handled by the extreme or progressive barrier, or a mix of both.

With a feasible starting point, our new progressive barrier is not necessarily better than the extreme barrier.

Variant for infeasible starting point: keep violated constraints in open constraints until they are satisfied. Then switch them to the closed constraints under the extreme barrier.

MADS defines a class of derivative-free algorithms for general constrained optimization.

Hierarchical convergence analysis tied to local smoothness.

NOMAD is Gilles Couture's C++ industrial strength implementation.

NOMADm is Mark Abramson's MATLAB implementation.

MADS is in the GADS mathworks MATLAB toolbox.

Work in progress: Multiobjective; Large scale problems
We have presented some classes of direct search algorithms. Used in practice by our collaborators (Boeing, Exxon, GM,...)

Constraints may be handled by the extreme or progressive barrier, or a mix of both.

- With a feasible starting point, our new progressive barrier is not necessarily better than the extreme barrier.
- Variant for infeasible starting point: keep violated constraints in open constraints until they are satisfied. Then switch them to the closed constraints under the extreme barrier.
We have presented some classes of direct search algorithms.

Used in practice by our collaborators (Boeing, Exxon, GM,...)

Constraints may be handled by the extreme or progressive barrier, or a mix of both.

- With a feasible starting point, our new progressive barrier is not necessarily better than the extreme barrier.
- Variant for infeasible starting point: keep violated constraints in open constraints until they are satisfied. Then switch them to the closed constraints under the extreme barrier.

MADS defines a class of derivative-free algorithms for general constrained optimization.

- Hierarchical convergence analysis tied to local smoothness.
- NOMAD is Gilles Couture’s C++ industrial strength implementation.
- NOMADm is Mark Abramson’s MATLAB implementation.
- MADS is in the GADS MathWorks MATLAB toolbox.

Work in progress: Multiobjective; Large scale problems

Charles Audet (Optimization 2007)
We have presented some classes of direct search algorithms. Used in practice by our collaborators (Boeing, Exxon, GM,...) Constraints may be handled by the extreme or progressive barrier, or a mix of both.
- With a feasible starting point, our new progressive barrier is not necessarily better than the extreme barrier.
- Variant for infeasible starting point: keep violated constraints in open constraints until they are satisfied. Then switch them to the closed constraints under the extreme barrier.

MADS defines a class of derivative-free algorithms for general constrained optimization.
- Hierarchical convergence analysis tied to local smoothness.
- NOMAD is Gilles Couture’s C++ industrial strength implementation.
- NOMADm is Mark Abramson’s MATLAB implementation.
- MADS is in the GADS MathWorks MATLAB toolbox.

Work in progress: Multiobjective; Large scale problems
We have presented some classes of direct search algorithms.

- Used in practice by our collaborators (Boeing, Exxon, GM, ...)
- Constraints may be handled by the extreme or progressive barrier, or a mix of both.
  - With a feasible starting point, our new progressive barrier is not necessarily better than the extreme barrier.
  - Variant for infeasible starting point: keep violated constraints in open constraints until they are satisfied. Then switch them to the closed constraints under the extreme barrier.

- MADS defines a class of derivative-free algorithms for general constrained optimization.
  - Hierarchical convergence analysis tied to local smoothness.
  - NOMAD is Gilles Couture’s C++ industrial strength implementation.
  - NOMADm is Mark Abramson’s MATLAB implementation.
  - MADS is in the GADS MathWorks MATLAB toolbox.

- Work in progress: Multiobjective; Large scale problems