

Black-Box Optimization Algorithms for Problems with Convex Regularizers

joint work with Lindon Roberts (University of Sydney)

Yanjun Liu, Australian National University / Princeton University
yanjun.liu@princeton.edu

ISMP, 26 July 2024

- 1 Problem Setup
- 2 Algorithm Design
- 3 Implementation
- 4 Full Algorithm and Results Summary

Problem Setup

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x})$$

- 1 $f(\mathbf{x}) := \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2 = \frac{1}{2} \sum_{i=1}^m r_i(\mathbf{x})^2$,
where $\mathbf{r}(\mathbf{x}) := [r_1(\mathbf{x}) \dots r_m(\mathbf{x})]^T$ mapping from \mathbb{R}^n to \mathbb{R}^m .
Assume $\mathbf{r}(\mathbf{x}) \in C^1$ with Jacobian $[J(\mathbf{x})]_{i,j} = \frac{\partial r_i(\mathbf{x})}{\partial x_j}$.
However, these derivatives might not be accessible!
- 2 $h : \mathbb{R}^n \rightarrow \mathbb{R}$ is a convex but possibly nonsmooth regularization term.
Assume the proximal operator of h is cheap to evaluate.

Learning MRI Sampling Patterns (Ehrhardt and Roberts 2021):

$$\min_{\boldsymbol{\theta}} \frac{1}{2} \|\mathbf{x}^*(\boldsymbol{\theta}) - \mathbf{x}_{true}\|^2 + \|\boldsymbol{\theta}\|_1$$

- $\boldsymbol{\theta} \in \mathbb{R}^d$: weights determining importance of Fourier coefficients of the image
- $\mathbf{x}^*(\boldsymbol{\theta})$: reconstruct process is complicated!
- 1-norm: keep sparsity to save time for MRI scan.

Derivative-free optimization (DFO):

- black-box, noisy or expensive to evaluate.
- several approaches: direct search, Nelder-Mead, [model-based](#), ...

Classical Trust Region Framework

$$\min_{\mathbf{x}} \Phi(\mathbf{x}), \quad \Phi \text{ is smooth}$$

At k -th iteration:

- 1 Construct a model function m_k approximating Φ within trust region $B(\mathbf{x}_k, \Delta_k)$
- 2 Find a minimizer of m_k within the trust region

$$\mathbf{s}_k \in \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s})$$

- 3 Calculate the ratio

$$R_k = \frac{\text{objective decrease}}{\text{model decrease}} = \frac{\Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}$$

- 4 Update iterate \mathbf{x}_{k+1} , trust region radius Δ_{k+1} based on R_k .
(R_k close to 1: step \mathbf{s}_k successful)

Convergence:

Under reasonable assumption, stationary measure $\|\nabla \Phi(\mathbf{x}_k)\| \rightarrow 0$.

Classical Trust Region Framework

$$\min_{\mathbf{x}} \Phi(\mathbf{x}), \quad \Phi \text{ is smooth}$$

At k -th iteration:

- 1 Construct a model function m_k approximating Φ within trust region $B(\mathbf{x}_k, \Delta_k)$
- 2 Find a minimizer of m_k within the trust region

$$\mathbf{s}_k \in \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s})$$

- 3 Calculate the ratio

$$R_k = \frac{\text{objective decrease}}{\text{model decrease}} = \frac{\Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}$$

- 4 Update iterate \mathbf{x}_{k+1} , trust region radius Δ_{k+1} based on R_k .
(R_k close to 1: step \mathbf{s}_k successful)

Convergence:

Under reasonable assumption, stationary measure $\|\nabla \Phi(\mathbf{x}_k)\| \rightarrow 0$.

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- 1 Model construction?
- 2 Finding minimizer of model function?
- 3 Update rule?
- 4 Stationary measure?

Model Construction

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

[Cartis and Roberts 2019]

- ① Derivative-based (Jacobian $J(\mathbf{x})$ available): Taylor expand $\mathbf{r}(\mathbf{x})$:

$$\mathbf{r}(\mathbf{x}_k + \mathbf{s}) \approx \mathbf{r}(\mathbf{x}_k) + \mathbf{J}(\mathbf{x}_k) \mathbf{s}$$

$$\Phi(\mathbf{x}_k + \mathbf{s}) \approx f(\mathbf{x}_k) + \mathbf{r}(\mathbf{x}_k)^T \mathbf{J}(\mathbf{x}_k) \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{J}(\mathbf{x}_k)^T \mathbf{J}(\mathbf{x}_k) \mathbf{s} + h(\mathbf{x}_k + \mathbf{s})$$

- ② Derivative-free: Approximate Jacobian $J(\mathbf{x}_k)$ at iterate \mathbf{x}_k by J_k :

$$\mathbf{r}(\mathbf{x}_k + \mathbf{s}) \approx \mathbf{m}_k(\mathbf{x}_k + \mathbf{s}) := \mathbf{r}(\mathbf{x}_k) + \mathbf{J}_k \mathbf{s}$$

$$\Phi(\mathbf{x}_k + \mathbf{s}) \approx m_k(\mathbf{x}_k + \mathbf{s}) := \underbrace{f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{H}_k \mathbf{s}}_{p_k(\mathbf{x}_k + \mathbf{s})} + h(\mathbf{x}_k + \mathbf{s})$$

where $\mathbf{g}_k := J_k^T \mathbf{r}(\mathbf{x}_k)$ and $\mathbf{H}_k := J_k^T J_k$ (**symmetric + p.s.d.**).

Calculation of \mathbf{g}_k and \mathbf{H}_k : For each iteration, maintain an interpolation set

$\mathbf{Y}_k := \{\mathbf{y}_0 := \mathbf{x}_k, \mathbf{y}_1, \dots, \mathbf{y}_n\}$. Interpolation condition:

$$\mathbf{m}_k(\mathbf{y}_t) = \mathbf{r}(\mathbf{y}_t), \forall t = 0, 1, \dots, n$$

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- 1 Model construction? Include h in m_k
- 2 Finding minimizer of model function?
- 3 Update rule? Interpolation
- 4 Stationary measure?

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x}_k)\|^2$$

Derivative-free:

- Deal with nonsmooth Ψ without exploiting structure:
 - ① Model-based: [Audet and Hare 2020]
 - ② Direct search: [Audet and Dennis 2006]
- Deal with $f(\mathbf{x}) + h(\mathbf{c}(\mathbf{x}))$
for f and c black-box smooth, h convex nonsmooth:
 - ① [Grapiglia, J. Yuan, and Y.-x. Yuan 2016]:
 - DFO version of [Cartis, Gould, and Toint 2011]
 - ② [Garmanjani, Júdice, and Vicente 2016]:
 - convergence, worse-case complexity
 - smooth vs composite approach
 - ③ [Larson and Menickelly 2024]:
 - model-based trust region

Stationary Measure

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x}_k)\|^2$$

① If the objective function Φ is smooth: $\|\nabla \Phi(\mathbf{x}^*)\| = 0$.

② If ∇f accessible:

$$\begin{aligned} l(\mathbf{x}, \mathbf{s}) &:= f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{s} + h(\mathbf{x} + \mathbf{s}) \\ \zeta(\mathbf{x}) &:= l(\mathbf{x}, 0) - \min_{\|\mathbf{s}\| \leq 1} l(\mathbf{x}, \mathbf{s}) \end{aligned}$$

We say that \mathbf{x}^* is a *critical point* of Φ if $\zeta(\mathbf{x}^*) = 0$.

[Cartis, Gould, and Toint 2011]

③ If ∇f inaccessible: At k -th iteration, after calculating a local approximation p_k of f :

$$\begin{aligned} \tilde{l}(\mathbf{x}, \mathbf{s}) &:= f(\mathbf{x}) + \nabla p_k(\mathbf{x})^T \mathbf{s} + h(\mathbf{x} + \mathbf{s}), \mathbf{s} \in \mathbb{R}^n \\ \eta(\mathbf{x}) &:= \tilde{l}(\mathbf{x}, 0) - \min_{\|\mathbf{s}\| \leq 1} \tilde{l}(\mathbf{x}, \mathbf{s}). \end{aligned}$$

Note: If $h \equiv 0$, $\eta(\mathbf{x}_k) = \|\mathbf{g}_k\|$. [Grapiglia, J. Yuan, and Y.-x. Yuan 2016]

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- 1 Model construction? Include h in m_k
- 2 Finding minimizer of model function?
- 3 Update rule? Interpolation
- 4 Stationary measure? Introduce $\eta(\mathbf{x}_k)$ (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- 1 Model construction? Include h in m_k
 - 2 Finding minimizer of model function?
 - 3 Update rule? Interpolation
 - 4 Stationary measure? Introduce $\eta(\mathbf{x}_k)$ (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)
- New Problem:
- 5 For convergence, we need the criticality phase to ensure that Δ_k is comparable to $\eta(\mathbf{x}_k)$.
How to compute the stationary measure $\eta(\mathbf{x}_k)$?

- 1 Problem Setup
- 2 Algorithm Design
 - Model Construction
 - Stationary Measure
- 3 Implementation
- 4 Full Algorithm and Results Summary

1 Problem Setup

2 Algorithm Design

3 Implementation

- Implementation: Inexact Solver for Two Subproblems
- Choosing Accuracy Level

4 Full Algorithm and Results Summary

Implementation: Inexact Solver for Two subproblems

- ① Calculating approximate stationary measure:

$$\begin{aligned}\tilde{l}(\mathbf{x}_k, \mathbf{s}) &:= f(\mathbf{x}_k) + \nabla p_k(\mathbf{x}_k)^T \mathbf{s} + h(\mathbf{x}_k + \mathbf{s}) \\ &= f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + h(\mathbf{x}_k + \mathbf{s}) \\ \eta_1(\mathbf{x}_k) &:= \tilde{l}(\mathbf{x}_k, 0) - \min_{\|\mathbf{s}\| \leq 1} \tilde{l}(\mathbf{x}_k, \mathbf{s})\end{aligned}$$

- ② Calculating step size \mathbf{s}_k :

$$\mathbf{s}_k \in \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s}) = f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{H}_k \mathbf{s} + h(\mathbf{x}_k + \mathbf{s})$$

Implementation: Inexact Solver for Two subproblems

- ① Calculating approximate stationary measure:

$$\begin{aligned}\tilde{l}(\mathbf{x}_k, \mathbf{s}) &:= f(\mathbf{x}_k) + \nabla p_k(\mathbf{x}_k)^T \mathbf{s} + h(\mathbf{x}_k + \mathbf{s}) \\ &= f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + h(\mathbf{x}_k + \mathbf{s}) \\ \eta_1(\mathbf{x}_k) &:= \tilde{l}(\mathbf{x}_k, 0) - \min_{\|\mathbf{s}\| \leq 1} \tilde{l}(\mathbf{x}_k, \mathbf{s})\end{aligned}$$

- ② Calculating step size \mathbf{s}_k :

$$\mathbf{s}_k \in \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s}) = f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{H}_k \mathbf{s} + h(\mathbf{x}_k + \mathbf{s})$$

\Rightarrow Both are of the form: convex smooth + convex nonsmooth s.t. ball constraint. Specifically, given \mathbf{g} , \mathbf{H} , h , \mathbf{x} , r , and $C := B(0, r)$,

$$\min_{\mathbf{d}} G(\mathbf{d}) := \underbrace{\mathbf{g}^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T \mathbf{H} \mathbf{d}}_{\text{smooth}} + \underbrace{h(\mathbf{x} + \mathbf{d})}_{\text{nonsmooth}} + \underbrace{l_C(\mathbf{d})}_{\text{nonsmooth}}.$$

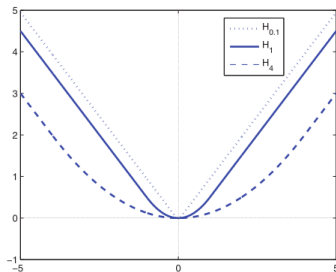
Implementation: Inexact Solver for Two subproblems

At k -th iteration, given \mathbf{g} , H , h , \mathbf{x} , r , and $C := B(0, r)$ and

$$\min_{\mathbf{d}} G(\mathbf{d}) := \underbrace{\mathbf{g}^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T H \mathbf{d}}_{\text{smooth}} + \underbrace{h(\mathbf{x} + \mathbf{d})}_{\text{nonsmooth}} + \underbrace{l_C(\mathbf{d})}_{\text{nonsmooth}}.$$

IDEA: Replacing the nonsmooth h by its **smooth approximation**. Given a smoothing parameter $\mu > 0$, smoothing h by its *Moreau envelope*:

$$M_h^\mu(\mathbf{y}) := \min_{\mathbf{z}} \left\{ h(\mathbf{z}) + \frac{1}{2\mu} \|\mathbf{y} - \mathbf{z}\|^2 \right\}$$



Implementation: Inexact Solver for Two Subproblems

Smoothed version:

$$\Rightarrow \min_{\mathbf{d}} G_{\mu}(\mathbf{d}) := \underbrace{\mathbf{g}^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T H \mathbf{d} + M_h^{\mu}(\mathbf{x} + \mathbf{d})}_{\text{smooth convex} =: F_{\mu}(\mathbf{d})} + \underbrace{I_C(\mathbf{d})}_{\text{nonsmooth convex}}.$$

Now applying accelerated proximal gradient method (FISTA):

- $\nabla F_{\mu}(\mathbf{d}) = \mathbf{g} + H\mathbf{d} + \nabla M_h^{\mu}(\mathbf{x} + \mathbf{d})$
- proximal operator of I_C is the projection operator P_C onto C .

Algorithm (Solving two subproblems: Smooth-FISTA (Beck 2017))

Given smoothing parameter $\mu > 0$.

- 1 Set $\mathbf{d}^0 = \mathbf{y}^0 = 0$, $t_0 = 1$, and step size $L = \|H\| + \frac{1}{\mu}$.
- 2 For $j = 0, 1, 2, \dots$
- 3 set $\mathbf{d}^{j+1} = P_C(\mathbf{y}^j - \frac{1}{L} \nabla F_{\mu}(\mathbf{y}^j))$;
- 4 compute $\mathbf{y}^{j+1} = \mathbf{d}^{j+1} + \left(\frac{t_j - 1}{t_{j+1}}\right)(\mathbf{d}^{j+1} - \mathbf{d}^j)$.

Implementation: Inexact Solver for Two Subproblems

At k -th iteration, given \mathbf{g} , H , h , \mathbf{x} , r , and $C := B(0, r)$,

$$\min_{\mathbf{d}} G(\mathbf{d}) := \underbrace{\mathbf{g}^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T H \mathbf{d}}_{\text{smooth}} + \underbrace{h(\mathbf{x} + \mathbf{d})}_{\text{nonsmooth}} + \underbrace{l_C(\mathbf{d})}_{\text{nonsmooth}}.$$

Theorem (S-FISTA, (Beck 2017))

Suppose that h is convex and L_h -Lipschitz continuous. Let $\{\mathbf{d}^j\}_{j \geq 0}$ be the sequence generated by S-FISTA. For an accuracy level $\epsilon > 0$, if the smoothing parameter μ and the number of iterations J are set as

$$\mu = \frac{2\epsilon}{L_h(L_h + \sqrt{L_h^2 + 2\|H\|\epsilon})} \quad \text{and} \quad J = \frac{r(2L_h + \sqrt{2\|H\|\epsilon})}{\epsilon}, \quad (1)$$

then for any $j \geq J$, it holds that $G(\mathbf{d}^j) - G(\mathbf{d}^*) \leq \epsilon$.

Questions

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- ① Model construction? Include h in m_k
- ② Finding minimizer of model function? **Using S-FISTA!**
- ③ Update rule? Interpolation
- ④ Stationary measure? Introduce $\eta(\mathbf{x}_k)$ (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)
New Problem:
- ⑤ For convergence, we need the criticality phase to ensure that Δ_k is comparable to $\eta(\mathbf{x}_k)$.
How to compute the stationary measure $\eta(\mathbf{x}_k)$? **Using S-FISTA!**

Questions

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

- 1 Model construction? Include h in m_k
- 2 Finding minimizer of model function? **Using S-FISTA!**
- 3 Update rule? Interpolation
- 4 Stationary measure? Introduce $\eta(\mathbf{x}_k)$ (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)
New Problem:
- 5 For convergence, we need the criticality phase to ensure that Δ_k is comparable to $\eta(\mathbf{x}_k)$.
How to compute the stationary measure $\eta(\mathbf{x}_k)$? **Using S-FISTA!**

But S-FISTA is **inexact**! New issues:

- To get our algorithm work, What is the accuracy we need the stationary measure computed to?
- What is the sufficient decrease condition for computing trust region steps?

⇒ **How to pick ϵ in both cases?**

Implementation: Choosing Accuracy Level

Theoretically, we discussed:

- Model construction: include h in m_k
- Stationary measure: introduce η (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)

Practically, how to implement the algorithm?

- How to find a minimizer of m_k within the trust region?
- For convergence, we have the criticality phase to ensure Δ_k is comparable to $\eta(\mathbf{x}_k)$.

How to compute the stationary measure $\eta(\mathbf{x}_k)$?

Solution: Using S-FISTA!

Implementation: Choosing Accuracy Level

Theoretically, we discussed:

- Model construction: include h in m_k
- Stationary measure: introduce η (if $h \equiv 0$, equal to the $\|\mathbf{g}_k\|$)

Practically, how to implement the algorithm?

- How to find a minimizer of m_k within the trust region?
- For convergence, we have the criticality phase to ensure Δ_k is comparable to $\eta(\mathbf{x}_k)$.

How to compute the stationary measure $\eta(\mathbf{x}_k)$?

Solution: Using S-FISTA!

But S-FISTA is **inexact**! New issues:

- To get our algorithm work, What is the accuracy we need the stationary measure computed to?
- What is the sufficient decrease condition for computing the step size?

⇒ **How to pick ϵ in both cases?**

Implementation: Choosing Accuracy Level

Inaccurate estimation:

- 1 Stationary measure $\eta(\mathbf{x}_k) := \tilde{l}(\mathbf{x}_k, 0) - \min_{\|\mathbf{s}\| \leq 1} \tilde{l}(\mathbf{x}_k, \mathbf{s})$: Applying S-FISTA until

$$\eta(\mathbf{x}_k) - \bar{\eta}(\mathbf{x}_k) \leq \epsilon_1 \Delta_k$$

- 2 Step size $\mathbf{s}_k \in \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s})$: Applying S-FISTA until

$$m_k(\mathbf{x}_k + \mathbf{s}_k) - \arg \min_{\|\mathbf{s}\| \leq \Delta_k} m_k(\mathbf{x}_k + \mathbf{s}) \leq (1 - \epsilon_2) \bar{\eta}(\mathbf{x}_k) \min \left\{ \Delta_k, \frac{\bar{\eta}(\mathbf{x}_k)}{\max\{1, \|H_k\|\}} \right\}.$$

Remark: This implies a generalized *Cauchy decrease condition*:

$$m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k) \geq \epsilon_2 \bar{\eta}(\mathbf{x}_k) \min \left\{ \Delta_k, \frac{\bar{\eta}(\mathbf{x}_k)}{\max\{1, \|H_k\|\}} \right\}.$$

- 1 Problem Setup
- 2 Algorithm Design
- 3 Implementation
- 4 Full Algorithm and Results Summary

Full Algorithm

Calculate approximate stationary measure $\eta(\mathbf{x}_k)$ *inaccurately*: applying using S-FISTA until

$$\eta(\mathbf{x}_k) - \bar{\eta}(\mathbf{x}_k) \leq \epsilon_1 \Delta_k$$

- 1 If $\bar{\eta}(\mathbf{x}_k) < \epsilon$, go to the criticality phase.
- 2 Construct a model function m_k within trust region $B(\mathbf{x}_k, \Delta_k)$:

$$m_k(\mathbf{x}_k + \mathbf{s}) = f(\mathbf{x}_k) + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T H_k \mathbf{s} + h(\mathbf{x}_k + \mathbf{s})$$

- 3 Find a minimizer of m_k within the trust region: using *inexact* solver S-FISTA to get a step \mathbf{s}_k satisfying $\|\mathbf{s}_k\| \leq \Delta_k$, $m_k(\mathbf{x}_k + \mathbf{s}_k) \leq m_k(\mathbf{x}_k)$ and

$$m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k) \geq \epsilon_2 \bar{\eta}(\mathbf{x}_k) \min \left\{ \Delta_k, \frac{\bar{\eta}(\mathbf{x}_k)}{\max\{1, \|H_k\|\}} \right\}.$$

- 4 Calculate the decrease ratio R_k
- 5 Update iterate \mathbf{x}_{k+1} , trust region radius Δ_{k+1} based on R_k and *interpolation set*.
(R_k close to 1 & good geometry of *interpolation set*: step \mathbf{s}_k successful)

Convergence & Complexity

$$\min_{\mathbf{x} \in \mathbb{R}^n} \Phi(\mathbf{x}) = f(\mathbf{x}) + h(\mathbf{x}), \quad f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x}_k)\|^2$$

Convergence and worst-case complexity match for the case $h = 0$.

Assumptions:

- f is continuously differentiable; ∇f is Lipschitz continuous.
- h is convex (possibly nonsmooth) and Lipschitz continuous.
- (standard) the model Hessians $\|H_k\|$ are uniformly bounded.

Theorem (Convergence - *true* stationary measure)

$$\lim_{k \rightarrow \infty} \zeta(\mathbf{x}_k) = 0.$$

Theorem (Complexity)

For $\epsilon \in (0, 1]$, the number of iterations until $\Psi_1(\mathbf{x}_k) < \epsilon$ for the first time is at most $k = \mathcal{O}(\epsilon^{-2})$, same as the unregularized DFO.

Improve the state-of-the-art solver DFO-LS: [Cartis, Fiala, et al. 2019]

- Use S-FISTA to calculate the generalized stationary measure and trust region subproblem with regularization *inaccurately*.
- Extend the safety phase from DFO-LS to the case with regularization: detect insufficient decrease generated by the step size $\|\mathbf{s}_k\|$ before evaluating $f(x_k)$.
- Require the proximal operator of h easy-to-compute

Tested on a collection of 53 low-dimensional, unconstrained nonlinear least squares (from [Moré and Wild 2009]) with 1-norm regularization.

Numerical Experiments

We compare DFO-LSR to:

NOMAD - direct search DFO solver (not exploit the least-squares structure).

[Le Digabel 2011]

Measuring the proportion of problems solved vs. the number of evaluations

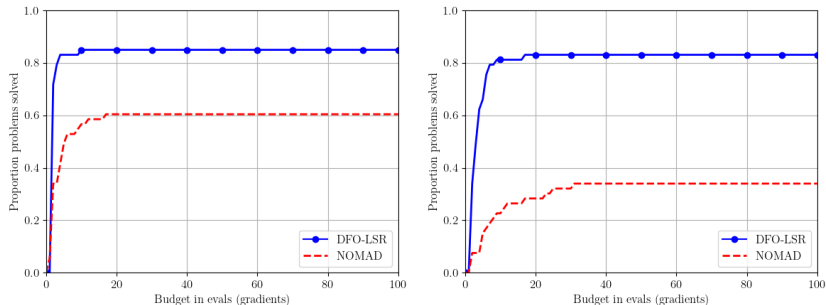


Figure: Left: accuracy level $\tau = 10^{-1}$; Right: accuracy level $\tau = 10^{-3}$

Summary and Future Work

Summary:

- Generalize model-based DFO method for minimizing nonconvex smooth function with convex regularizers
- Applying S-FISTA to compute stationary measure and step size inaccurately, with practical implementation and theoretical analysis (results matching with unregularized DFO)
- New software for least-squares problems with convex regularizers

Future work:






- Adapt to model functions as the sum of derivative-free but *possibly nonconvex quadratic* approximation and convex regularizer.

🔗: <https://github.com/yanjunliu-regina/dfols>





🎓: <https://arxiv.org/abs/2407.14915>

📄: https://yanjunliu-regina.github.io/files/yanjun_liu_ismp_2024.pdf




References I

-  Audet, Charles and J. E. Dennis (2006). “Mesh Adaptive Direct Search Algorithms for Constrained Optimization”. In: *SIAM Journal on Optimization* 17.1, pp. 188–217.
-  Audet, Charles and Warren Hare (2020). “Model-Based Methods in Derivative-Free Nonsmooth Optimization”. In: *Numerical Nonsmooth Optimization: State of the Art Algorithms*. Springer International Publishing, pp. 655–691.
-  Beck, Amir (2017). *First-order methods in optimization*. SIAM.
-  Cartis, Coralia, Jan Fiala, et al. (2019). “Improving the flexibility and robustness of model-based derivative-free optimization solvers”. In: *ACM Transactions on Mathematical Software (TOMS)* 45.3, pp. 1–41.
-  Cartis, Coralia, Nicholas IM Gould, and Philippe L Toint (2011). “On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming”. In: *SIAM Journal on Optimization* 21.4, pp. 1721–1739.

References II

-  Cartis, Coralia and Lindon Roberts (2019). “A derivative-free Gauss–Newton method”. In: *Mathematical Programming Computation* 11, pp. 631–674.
-  Ehrhardt, Matthias J and Lindon Roberts (2021). “Inexact derivative-free optimization for bilevel learning”. In: *Journal of mathematical imaging and vision* 63.5, pp. 580–600.
-  Garmanjani, Rohollah, Diogo Júdice, and Luís Nunes Vicente (2016). “Trust-region methods without using derivatives: worst case complexity and the nonsmooth case”. In: *SIAM Journal on Optimization* 26.4, pp. 1987–2011.
-  Grapiglia, Geovani Nunes, Jinyun Yuan, and Ya-xiang Yuan (2016). “A derivative-free trust-region algorithm for composite nonsmooth optimization”. In: *Computational and Applied Mathematics* 35.2, pp. 475–499.

References III

-  Larson, Jeffrey and Matt Menickelly (2024). “Structure-aware methods for expensive derivative-free nonsmooth composite optimization”. In: *Mathematical Programming Computation* 16.1, pp. 1–36.
-  Le Digabel, Sébastien (2011). “Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm”. In: *ACM Transactions on Mathematical Software (TOMS)* 37.4, pp. 1–15.
-  Moré, Jorge J and Stefan M Wild (2009). “Benchmarking derivative-free optimization algorithms”. In: *SIAM Journal on Optimization* 20.1, pp. 172–191.