# Derivative-Free Optimization for Constrained Least-Squares

Joint work with Lindon Roberts (ANU)

Matthew Hough, University of Queensland ⇒ University of Waterloo
mhough@uwaterloo.ca

INFORMS 2021 Annual Meeting, Oct. 24-27

# Outline

1. Introduction to DFO trust-region methods
2. DFO for constrained least-squares
3. Numerical results

# The Problem

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

- $f : \mathbb{R}^n \to \mathbb{R}$ continuously differentiable and possibly nonconvex
- Assume we cannot evaluate $\nabla f(\boldsymbol{x})$
  - Black-box
  - Noisy
  - Computationally expensive
- Applications: climate modelling, experimental design, machine learning, etc
- Seeking a local minimizer (approx. stationary point: $\|\nabla f(\boldsymbol{x}^*)\| \leq \epsilon$)

# Model-Based DFO

- Classic approach:

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx m_k(\boldsymbol{s}) = f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \nabla^2 f(\boldsymbol{x}_k)\boldsymbol{s}$$

- Instead, approximate:

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

- Find $\boldsymbol{g}_k$ and $H_k$ by interpolating $f$ over a set of points

# Model-Based DFO: The Least-Squares Case

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|_2^2, \qquad \boldsymbol{r}(\boldsymbol{x}) \in \mathbb{R}^n$$

- Typically linearize $\boldsymbol{r}$ at $\boldsymbol{x}_k$ using the Jacobian:

$$\boldsymbol{r}(\boldsymbol{x}_k+\boldsymbol{s}) \approx M(\boldsymbol{s}) = \boldsymbol{r}(\boldsymbol{x}_k)+J(\boldsymbol{x}_k)\boldsymbol{s}$$

- But in DFO, Jacobian is not available:

$$M(\boldsymbol{s}) = \boldsymbol{r}(\boldsymbol{x}_k) + J_k\boldsymbol{s}$$

- Find $J_k$ by interpolation

End up with a local quadratic model

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx m_k(\boldsymbol{s}) := \frac{1}{2}\|M_k(\boldsymbol{s})\|_2^2$$

# Model-Based DFO: Algorithm

(assuming our interpolation model is a good approx.)

1. Build local interpolation model:

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx m_k(\boldsymbol{s}) = \frac{1}{2}\|M_k(\boldsymbol{s})\|_2^2$$

2. Minimize the model within the trust-region $\Delta_k$ to get the step

$$\boldsymbol{s}_k = \underset{s \in \mathbb{R}^d}{\arg\min}\, m_k(\boldsymbol{s}) \quad \text{s.t. } \|\boldsymbol{s}\|_2 \leq \Delta_k$$

3. Evaluate $f(\boldsymbol{x}_k + \boldsymbol{s}_k)$, check sufficient decrease, select $\boldsymbol{x}_{k+1}$ and $\Delta_{k+1}$
4. Update interpolation set with the new point $\boldsymbol{x}_k + \boldsymbol{s}_k$

# Model-Based DFO: Interpolation Geometry

We may not get sufficient decrease if...

1. $\Delta_k$ is too large
2. $m_k$ is not a good approximation to $f$ (bad geometry)

- Good geometry $\implies$ accurate model $\implies$ convergence
- Need interpolation set $\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_n\}$ to be "well-poised" in $B(\boldsymbol{y}_0, \Delta)$
- **$\Lambda$-poised** if all $\boldsymbol{y}_t \in B(\boldsymbol{y}_0, \Delta)$ and exists $\Lambda \geq 1$ s.t.

$$\max |\ell_t(\boldsymbol{y})| \leq \Lambda, \quad \forall \boldsymbol{y} \in B(\boldsymbol{y}_0, \Delta)$$

  - $\ell_t(\boldsymbol{y}_s) = \delta_{s,t}$ for all $s, t$
  - Points are "well-spaced"

[Conn, Scheinberg & Vicente, 2009]

# Model-based DFO: Interpolation Geometry

- $\Lambda$-poisedness $\implies$ fully linear model:
  - $|f(\boldsymbol{x}_k + \boldsymbol{s}) - m(\boldsymbol{s})| \leq \kappa_{ef}\Delta_k^2$
  - $\|\nabla f(\boldsymbol{x}_k + \boldsymbol{s}) - \nabla m(\boldsymbol{s})\| \leq \kappa_{eg}\Delta_k$

($\kappa_{ef}$, $\kappa_{eg}$ depend on $\Lambda$)

for all $\boldsymbol{y} \in B(\boldsymbol{y}_0, \Delta_k)$, $\|\boldsymbol{s}\| \leq \Delta_k$

- Fully linear model $\implies$ convergence
- Two important algorithms:
  1. Checks $\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_n\}$ is $\Lambda$-poised
  2. Makes $\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_n\}$ $\Lambda$-poised if it is not already

[Conn, Scheinberg & Vicente, 2009]

# The Constrained Problem

$$\min_{\boldsymbol{x}\in\mathcal{C}} f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|_2^2, \qquad \boldsymbol{r}(\boldsymbol{x}) \in \mathbb{R}^n$$

- $f : \mathbb{R}^d \to \mathbb{R}$ continuously differentiable and possibly nonconvex
- Assume we cannot evaluate $\nabla f(\boldsymbol{x})$
- $\mathcal{C} \subseteq \mathbb{R}^d$ has nonempty interior, closed, and convex
  - Cannot evaluate $f$ outside of $\mathcal{C}$
  - Only accessible via projection, $P_{\mathcal{C}} : \mathbb{R}^d \to \mathcal{C}$

# Constrained DFO: Algorithm

1. Build local interpolation model:

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx m_k(\boldsymbol{s}) = \frac{1}{2}\|M_k(\boldsymbol{s})\|_2^2$$

2. Minimize the model within $B(\boldsymbol{y}_0, \Delta_k) \cap \mathcal{C}$ to get the step

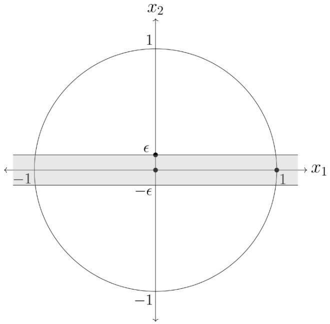
$$\boldsymbol{s}_k = \underset{\boldsymbol{s} \in B(\boldsymbol{y}_0, \Delta_k) \cap \mathcal{C}}{\arg\min} \; m_k(\boldsymbol{s})$$

3. Evaluate $f(\boldsymbol{x}_k + \boldsymbol{s}_k)$, check sufficient decrease, select $\boldsymbol{x}_{k+1}$ and $\Delta_{k+1}$
4. Update interpolation set with the new point $\boldsymbol{x}_k + \boldsymbol{s}_k$

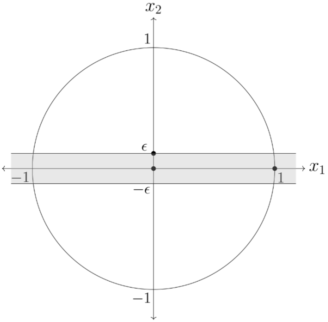
# Constrained DFO: Geometry

- $\mathcal{C} = \{(x_1, x_2) \ : \ |x_2| \leq \epsilon\} \subseteq \mathbb{R}^2$
- $Y = \{(0,0),(1,0),(0,\epsilon)\} \subseteq B(0,1)$
- In $B(0,1)$, points are $\Lambda$-poised with $\Lambda \sim \epsilon^{-1}$

$\implies$ large $\kappa_{ef}$, $\kappa_{eg}$



# Constrained DFO: Geometry




- $\mathcal{C} = \{(x_1, x_2) \ : \ |x_2| \leq \epsilon\} \subseteq \mathbb{R}^2$
- $Y = \{(0,0), (1,0), (0,\epsilon)\} \subseteq B(0,1)$
- In $B(0,1)$, points are $\Lambda$-poised with $\Lambda \sim \epsilon^{-1}$

  $\implies$ large $\kappa_{ef}$, $\kappa_{eg}$

- **$\Lambda$-poised** if all $\boldsymbol{y}_t \in B(\boldsymbol{y}_0, \Delta) \cap \mathcal{C}$ and exists $\Lambda \geq 1$ s.t.

$$\max |\ell_t(\boldsymbol{y})| \leq \Lambda, \quad \forall \boldsymbol{y} \in B(\boldsymbol{y}_0, \Delta) \cap \mathcal{C}$$

- Now we have $\Lambda \leq 3$ independent of $\epsilon \implies$ improved error bounds

# Constrained DFO: Geometry

- $\Lambda$-poisedness $\implies$ fully linear model in $B(\boldsymbol{x}_k, \Delta_k)$:

$$\max_{\substack{\boldsymbol{x}_k + \boldsymbol{s} \in \mathcal{C} \\ \|\boldsymbol{s}\| \leq \Delta_k}} |f(\boldsymbol{x}_k + \boldsymbol{s}) - m_k(\boldsymbol{s})| \leq \kappa_{ef} \Delta_k^2$$

$$\max_{\substack{\boldsymbol{x}_k + \boldsymbol{s} \in \mathcal{C} \\ \|\boldsymbol{s}\| \leq 1}} |(\nabla f(\boldsymbol{x}_k) - \boldsymbol{g}_k)^T \boldsymbol{s}| \leq \kappa_{eg} \Delta_k$$

- Slightly weaker:
  - $\nabla m(\boldsymbol{y}) \approx \nabla f(\boldsymbol{y})$ only at $\boldsymbol{y} = \boldsymbol{x}_k$
  - Only care about points in $\mathcal{C}$
- Still have important algorithms
  1. Check points are $\Lambda$-poised
  2. Make points $\Lambda$-poised if not

# Constrained DFO: Measuring Progress

$$\pi^f(\boldsymbol{x}) := \left| \min_{\substack{x+s\in\mathcal{C} \\ \|s\|\leq 1}} \nabla f(\boldsymbol{x})^T \boldsymbol{s} \right| \quad \Longrightarrow \quad \pi^m(\boldsymbol{x}) := \left| \min_{\substack{x+s\in\mathcal{C} \\ \|s\|\leq 1}} \boldsymbol{g}_k{}^T \boldsymbol{s} \right|$$

- For $\mathcal{C} = \mathbb{R}^d$, $\pi^f(\boldsymbol{x}_k) = \|\nabla f(\boldsymbol{x}_k)\|$, and $\pi^g(\boldsymbol{x}_k) = \|\boldsymbol{g}_k\|$
- fully linear $\Longrightarrow$ $|\pi^f(\boldsymbol{x}_k) - \pi^m(\boldsymbol{x}_k)| \leq \kappa_{eg}\Delta_k$

[Conn, Gould & Toint, 2000]

# Constrained DFO: Measuring Progress

$$\pi^f(\boldsymbol{x}) := \left| \min_{\substack{x+s\in\mathcal{C} \\ \|s\|\leq 1}} \nabla f(\boldsymbol{x})^T \boldsymbol{s} \right| \quad \Longrightarrow \quad \pi^m(\boldsymbol{x}) := \left| \min_{\substack{x+s\in\mathcal{C} \\ \|s\|\leq 1}} \boldsymbol{g}_k{}^T \boldsymbol{s} \right|$$

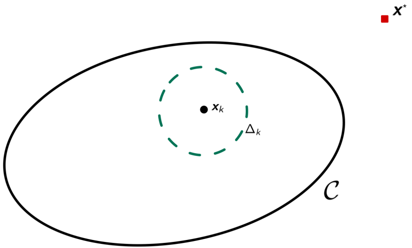
- For $\mathcal{C} = \mathbb{R}^d$, $\pi^f(\boldsymbol{x}_k) = \|\nabla f(\boldsymbol{x}_k)\|$, and $\pi^g(\boldsymbol{x}_k) = \|\boldsymbol{g}_k\|$
- fully linear $\implies |\pi^f(\boldsymbol{x}_k) - \pi^m(\boldsymbol{x}_k)| \leq \kappa_{eg}\Delta_k$

Solution to $\pi^f(\boldsymbol{x})$ is given by $\boldsymbol{s}^\star := p(t, \boldsymbol{x}) - \boldsymbol{x}$

- where $p(t, \boldsymbol{x}) = P_{\mathcal{C}}(\boldsymbol{x} - t\nabla f(\boldsymbol{x}))$, $t \geq 0$,
- and $\|p(t, \boldsymbol{x}) - \boldsymbol{x}\| = 1$

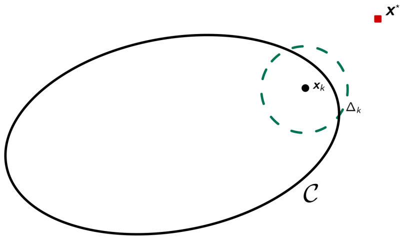
[Conn, Gould & Toint, 2000]

# Constrained DFO: Measuring Progress




- $s^\star = P_{\mathcal{C}}(x - t\nabla f(x)) - x = -t\nabla f(x)$
- $1 = \|s^\star\| = t\|\nabla f(x)\| \implies t = \frac{1}{\|\nabla f(x)\|} \implies s^\star = \frac{-\nabla f(x)}{\|\nabla f(x)\|}$
- $\implies \pi^f(x) = \|\nabla f(x)\|$

# Constrained DFO: Measuring Progress




- $P_C(\boldsymbol{x} - t\nabla f(\boldsymbol{x})) \neq \boldsymbol{x} - t\nabla f(\boldsymbol{x})$
- $\boldsymbol{s}^\star = p(t, \boldsymbol{x}) - \boldsymbol{x}$ gets smaller near the boundary
- $\implies \pi^f(\boldsymbol{x}) \to 0$ as approach constraints in direction of $x^\star$

# Constrained DFO: Convergence

1. Ensure we always have $m_k$ fully linear (by ensuring good geometry)
2. Ensure $\pi_k^m \sim \Delta_k$
3. When $\pi^m(\boldsymbol{x}_k) \to 0$, we are also getting $\pi^f(\boldsymbol{x}_k) \to 0$
4. Standard convergence results follow

- Worst-case complexity: at most $\mathcal{O}(\epsilon^{-2})$ iterations to have $\pi_k^m \leq \epsilon$

# Implementation

Open-source Python implementation: *DFO-LS*

- Github: `numericalalgorithmsgroup/dfols`

---

- Replace gradient-descent step with projected gradient-descent (PGD)
- Dykstra's algorithm for projecting onto $\mathcal{C}$
- New point becomes

$$\boldsymbol{x}_{k+1} = P_{\mathcal{Q}}(\boldsymbol{x}_k - t\boldsymbol{g}_k)$$

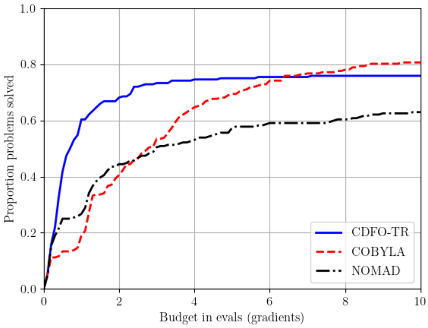
$\mathcal{Q} := \mathcal{C} \cap B(\boldsymbol{x}_k, \Delta_k)$

[Beck, 2017]

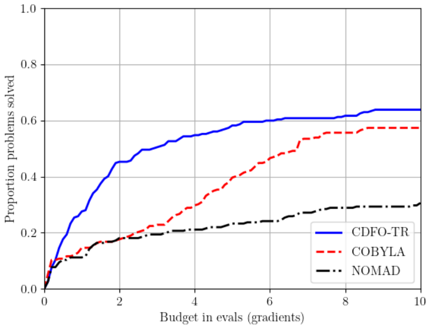
# Numerical results

58 test problems with ball, box, simplex, and no constraints

- [Moré & Wild, 2009], [Moré, Garbow, Hillstrom, 1981]




$\tau = 10^{-1}$




$\tau = 10^{-3}$

# Constrained DFO: Summary

- Can ensure good geometry

  $\implies$ fully linear model

  $\implies$ error bound on approx. criticality measure

  $\implies$ convergence

- Worst-case complexity same as in derivative-based case

# Constrained DFO: Future Work

- True quadratic models

  $\Downarrow$

- General objective function

- Extend to include regularization

# References

Amir Beck. *First-Order Methods in Optimization*. Oct. 2017.

Coralia Cartis and Lindon Roberts. "A derivative-free Gauss-Newton method". In: *Mathematical Programming Computation* 11.4 (2019), pp. 631–674.

Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. Vol. 1. MPS-SIAM Series on Optimization. Philadelphia: MPS/SIAM, 2000.

Andrew R. Conn, Katya Scheinberg, and Luís N. Vicente. *Introduction to Derivative-Free Optimization*. Vol. 8. MPS-SIAM Series on Optimization. Philadelphia: MPS/SIAM, 2009.

Jorge J. Moré, Burton S. Garbow, and Kenneth E. Hillstrom. "Testing Unconstrained Optimization Software". In: *ACM Transactions on Mathematical Software* 7.1 (Mar. 1981), pp. 17–41.

Jorge J. Moré and Stefan M. Wild. "Benchmarking Derivative-Free Optimization Algorithms". In: *SIAM Journal on Optimization* 20.1 (Jan. 2009), pp. 172–191.