Beyond the Black Box in Derivative-Free and Simulation-Based Optimization

Stefan Wild

Argonne National Laboratory
Mathematics and Computer Science Division

Joint work with Prasanna Balaprakash (Argonne), Aswin Kannan (IBM), Kamil Khan (Argonne→McMaster), Slava Kungurtsev (Czech TU Prague), Jeff Larson (Argonne), Matt Menickelly (Lehigh), Jorge Moré (Argonne)

July 13, 2016

Optimizing (Almost) Everything!

0. Simulation-based and derivative-free optimization?

I. Optimization of black boxes

- Empirical performance tuning of HPC codes
- Model-based algorithms

II. Exploiting structure in *functions of black boxes*

- Least squares - calibrating DFT sims
- Nonsmoothness - bioremediation
- Some partials - multilevel energy functionals
- Constraints

Optimizing (Almost) Everything!

0. Simulation-based and derivative-free optimization?

I. Optimization of black boxes

- Empirical performance tuning of HPC codes
- Model-based algorithms

II. Exploiting structure in *functions of black boxes*

- Least squares - calibrating DFT sims
- Nonsmoothness - bioremediation
- Some partials - multilevel energy functionals
- Constraints

Doing something with little

Doing better with little more

0. SBO and DFO

Simulation-Based Optimization

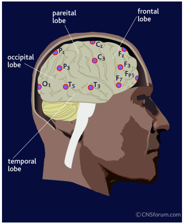
$$\min_{x \in \mathbb{R}^n} \{ f(x) = F[x, S(x)] : c_I[x, S(x)] \le 0,\ c_E[x, S(x)] = 0 \}$$

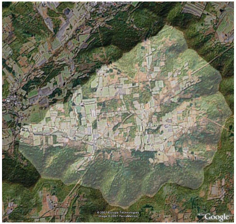
$\approx$ *"parameter estimation"* $\approx$ *"model calibration"* $\approx$ *"design optimization"* $\approx \ldots$

- $S : \mathbb{R}^n \to \mathbb{C}^p$ simulation output, often "noisy" (even when deterministic)
- Derivatives $\nabla_x S$ often unavailable or prohibitively expensive to obtain/approximate directly
- S can contribute to objective and/or constraints
- Single evaluation of S could take seconds/minutes/hours/...

$\Rightarrow$ Evaluation is a bottleneck for optimization

Functions of complex (numerical) simulations arise everywhere

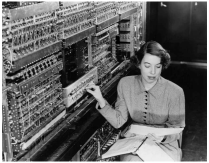




Blame Computing!

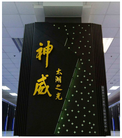
. . . for pervasiveness of simulations in sci&eng

- Parallel/multi-core environments common
- Simulations ("forward problem") faster, more realistic/complex



Argonne's AVIDAC
(1953 vacuum tubes)

Argonne's BlueGene/Q
(2012 0.8M cores)

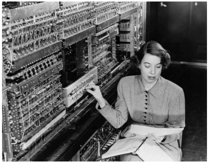


Sunway TaihuLight
(2016 11M cores)

Blame Computing!

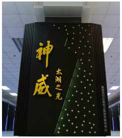
. . . for pervasiveness of simulations in sci&eng

- ◇ Parallel/multi-core environments common
- ◇ Simulations ("forward problem") faster, more realistic/complex

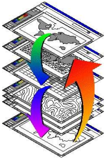


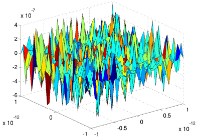
Argonne's AVIDAC
(1953 vacuum tubes)

Argonne's BlueGene/Q
(2012 0.8M cores)



Sunway TaihuLight
(2016 11M cores)





. . . for the challenges in SBO

- ◇ Optimization, UQ often an afterthought
- ◇ Obstacles for Algorithmic Differentiation (coupled legacy/proprietary codes, memory)
  → [Coleman & Xu; SIAM 2016], [Griewank & Walther; SIAM 2008]
  → MS76 today!
- ◇ Computational noise can complicate everything
  →[Moré & W.; SISC 2011]
- ◇ Finite differences noisy, possibly expensive
  → [Moré & W.; TOMS 2012]
- ◇ Computational budget limits # f evals

Derivative-Free Optimization

"Some derivatives ($\nabla_x S(x)$) unavailable for optimization purposes"

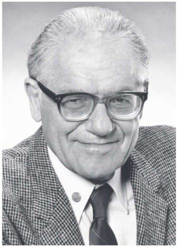
Derivative-Free Optimization

"Some derivatives ($\nabla_x S(x)$) unavailable for optimization purposes"

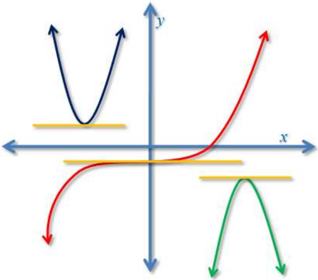
The Challenge:
Optimization is tightly coupled with derivatives

Typical optimality (no noise, smooth functions)

$$\nabla_x f(x_*) + \lambda^\top \nabla_x c_E(x_*) = 0, c_E(x_*) = 0$$



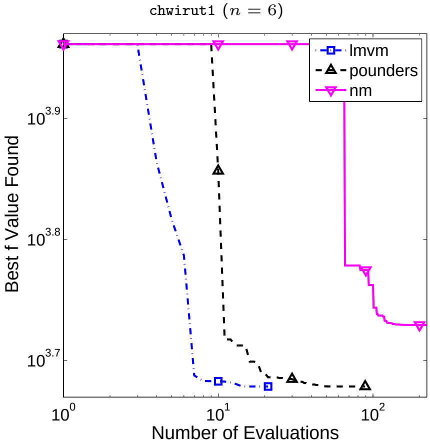
William Karush *[Optimization Stories, 2012]*



(sub)gradients $\nabla_x f$, $\nabla_x c$ enable:

- Faster feasibility
- Faster convergence
  - Guaranteed descent
  - Approximation of nonlinearities
- Better termination
  - Measure of criticality $\|\nabla_x f\|$, $\|\mathcal{P}_\Omega(\nabla_x f)\|$
- Sensitivity analysis
  - Correlations, standard errors, UQ, ...

The Price of Algorithm Choice: Solvers in PETSc/TAO



Toolkit for Advanced Optimization
[Munson et al.; mcs.anl.gov/tao]

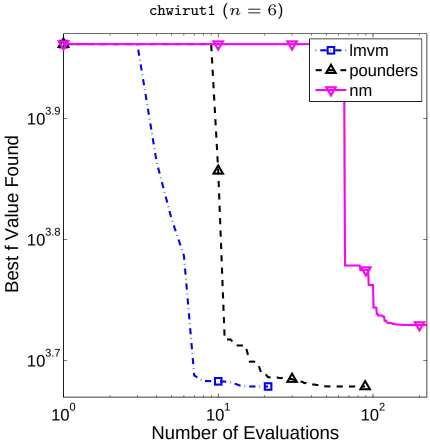
Increasing level of user input:

nm Assumes $\nabla_x f$ unavailable, black box

pounders Assumes $\nabla_x f$ unavailable, exploits problem structure

lmvm Uses available $\nabla_x f$

The Price of Algorithm Choice: Solvers in PETSc/TAO



Toolkit for Advanced Optimization
[Munson et al.; mcs.anl.gov/tao]

Increasing level of user input:

nm Assumes $\nabla_x f$ unavailable, black box

pounders Assumes $\nabla_x f$ unavailable, exploits problem structure

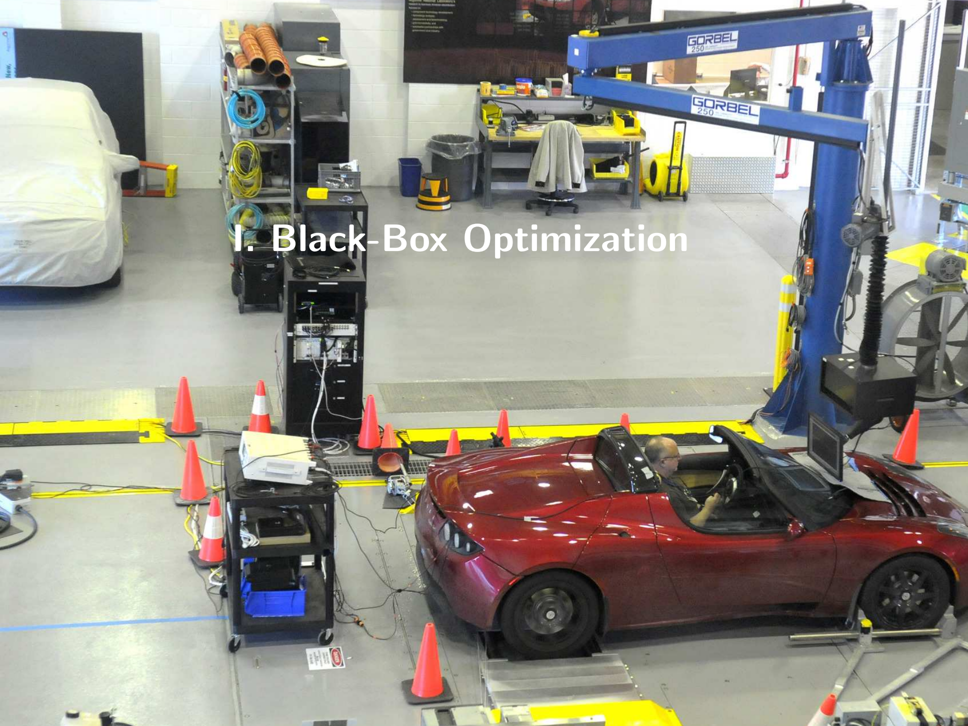
THIS TALK

lmvm Uses available $\nabla_x f$

DFO methods should be designed to beat finite-difference-based methods

Observe: Constrained by budget on #evals, method limits solution accuracy/problem size

I. Black-Box Optimization

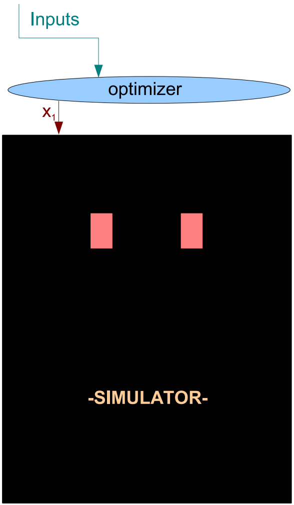


Black-Box Optimization

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

Only access to $f = S$ is through sampling

- (Scalar) Output of an experiment
- Proprietary libraries/closed codes
- Often discrete/compact domains

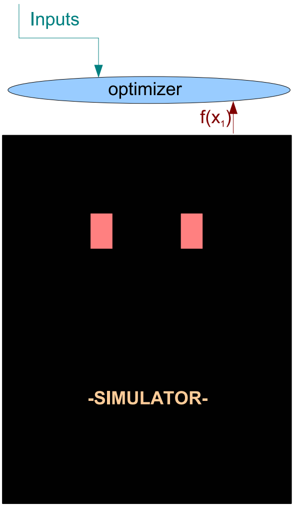


Black-Box Optimization

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

Only access to $f = S$ is through sampling

- (Scalar) Output of an experiment
- Proprietary libraries/closed codes
- Often discrete/compact domains



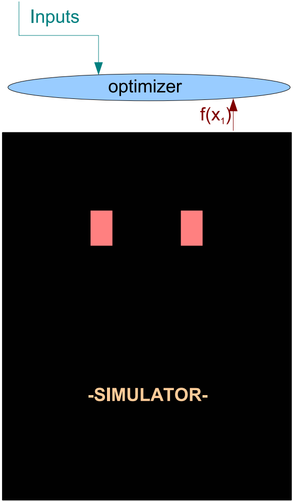
Black-Box Optimization

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

Only access to $f = S$ is through sampling

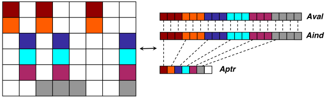
- (Scalar) Output of an experiment
- Proprietary libraries/closed codes
- Often discrete/compact domains

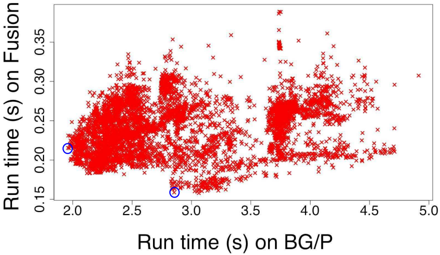
Throughout this talk:
"Black box" is both good and evil



A Black Box: Automating Empirical Performance Tuning

Given semantically equivalent codes $x_1, x_2, \ldots$, minimize run time subject to energy consumption





$$\min\left\{f(x) : (x_{\mathcal{C}}, x_{\mathcal{I}}, x_{\mathcal{B}}) \in \Omega_{\mathcal{C}} \times \Omega_{\mathcal{I}} \times \Omega_{\mathcal{B}}\right\}$$

- x multidimensional parameterization (compiler type, compiler flags, unroll/tiling factors, internal tolerances, . . .)
- Ω search domain (feasible transformation, no errors)
- f quantifiable performance objective (requires a run)

→ [Audet & Orban; SIOPT 2006], [Balaprakash, W., Hovland; ICCS 2011], [Porcelli & Toint; 2016]

Numerical Linear Algebra → [N. Higham; SIMAX 1993], . . .

Black-Box Algorithms: Stochastic Methods

Random search

Repeat:

1. Randomly generate direction $d_k \in \mathbb{R}^n$
2. Evaluate "gradient-free oracle" $g(x_k; h_k) = \frac{f(x_k + h_k\, d_k) - f(x_k)}{h_k} d_k$ ($\approx$ directional derivative)
3. Compute $x_{k+1} = x_k - \delta_k g(x_k; h_k)$, evaluate $f(x_{k+1})$

Convergence (for different types of f) tends to be probabilistic

[Kiefer & Wolfowitz; AnnMS 1952], [Polyak; 1987], [Ghadimi & Lan; SIOPT 2013], [Nesterov & Spokoiny; FoCM 2015], . . .

Black-Box Algorithms: Stochastic Methods

Random search

Repeat:

1. Randomly generate direction $d_k \in \mathbb{R}^n$
2. Evaluate "gradient-free oracle" $g(x_k; h_k) = \frac{f(x_k + h_k d_k) - f(x_k)}{h_k} d_k$
   ($\approx$ directional derivative)
3. Compute $x_{k+1} = x_k - \delta_k g(x_k; h_k)$, evaluate $f(x_{k+1})$

Convergence (for different types of f) tends to be probabilistic

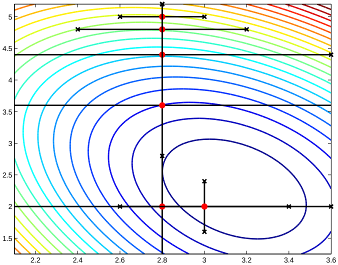
[Kiefer & Wolfowitz; AnnMS 1952], [Polyak; 1987], [Ghadimi & Lan; SIOPT 2013], [Nesterov & Spokoiny; FoCM 2015], . . .

Stochastic heuristics (nature-inspired methods, etc.)

- Popular in practice, especially in engineering
- Typically global in nature
- Require many f evaluations

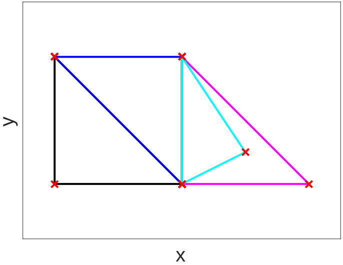
Black-Box Algorithms: Direct Search Methods

Pattern Search + Variants



Easy to parallelize f evaluations

Nelder-Mead + Variants



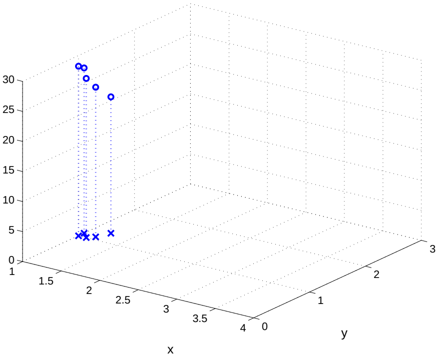
Popularized by Numerical Recipes

- Rely on indicator functions: $[f(x_k + s) <^? f(x_k)]$
- Work with **black-box** $f(x)$, do not exploit structure $F[x, S(x)]$
- Convergence results for variety of settings

Survey → [Kolda, Lewis, Torczon; SIREV 2003]
Newer NM → [Lagarias, Poonen, Wright; SIOPT 2012]
Tools → `DFL` [Liuzzi et al.], `NOMAD` [Audet et al.], . . .

Making the Most of Little Information About Smooth f

- Overhead of the optimization routine is minimal (negligible?) relative to cost of evaluating simulation



Bank of data, $\{x_i, f(x_i)\}_{i=1}^{k}$:

= Points (& function values) evaluated so far

= Everything known about f

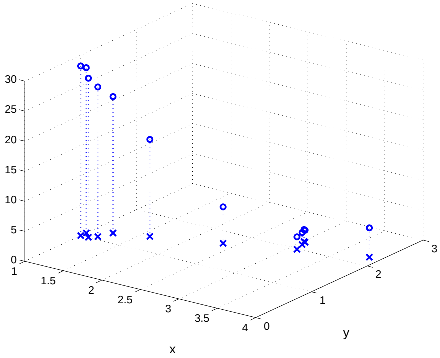
Goal:

- Make use of growing Bank as optimization progresses
- Limit unnecessary evaluations

(geometry/approximation)

Making the Most of Little Information About Smooth f

- Overhead of the optimization routine is minimal (negligible?) relative to cost of evaluating simulation



Bank of data, $\{x_i, f(x_i)\}_{i=1}^{k}$:

= Points (& function values) evaluated so far

= Everything known about f

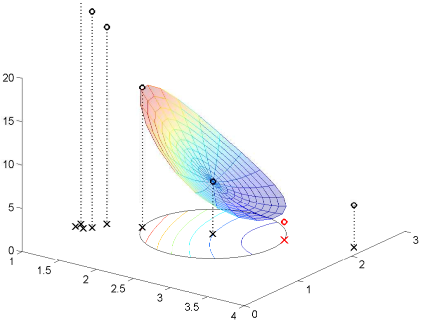
Goal:

- Make use of growing Bank as optimization progresses
- Limit unnecessary evaluations

(geometry/approximation)

Making the Most of Little Information About Smooth f

- Overhead of the optimization routine is minimal (negligible?) relative to cost of evaluating simulation



Bank of data, $\{x_i, f(x_i)\}_{i=1}^{k}$:

= Points (& function values) evaluated so far

= Everything known about f

Goal:

- Make use of growing Bank as optimization progresses
- Limit unnecessary evaluations

(geometry/approximation)

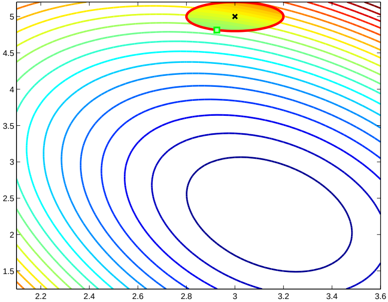
Model-Based Trust-Region Algorithms

Substitute $\min\{q_k(x) : x \in \mathcal{B}_k\}$ for $\min f(x)$

f expensive, no ∇f

q_k cheap, analytic derivatives

$$q_k(x) = f(x_k) + g_k^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H_k (x - x_k)$$



Trust region:
$\mathcal{B}_k = \{x \in \Omega : \|x - x_k\| \leq \Delta_k\}$

- Trust $q_k \approx f$ in $\mathcal{B}_k$
- Update based on $\rho_k = \dfrac{f(x_k) - f(x_+)}{q_k(x_k) - q_k(x_+)}$

Typical models

- Only need (occasional) local approximation
- Taylor-based: $g_k = \nabla f(x_k)$, $H_k \approx \nabla^2 f(x_k)$

$\longrightarrow$ [Conn, Gould, Toint; SIAM 2000]

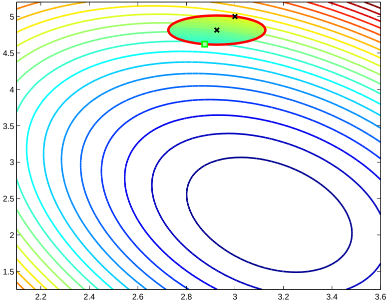
Model-Based Trust-Region Algorithms

Substitute $\min\{q_k(x) : x \in \mathcal{B}_k\}$ for $\min f(x)$

$$q_k(x) = f(x_k) + g_k^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H_k (x - x_k)$$

f expensive, no ∇f

q_k cheap, analytic derivatives



Trust region: $\mathcal{B}_k = \{x \in \Omega : \|x - x_k\| \leq \Delta_k\}$

- Trust $q_k \approx f$ in $\mathcal{B}_k$
- Update based on $\rho_k = \frac{f(x_k) - f(x_+)}{q_k(x_k) - q_k(x_+)}$

Typical models

- Only need (occasional) local approximation
- Taylor-based: $g_k = \nabla f(x_k)$, $H_k \approx \nabla^2 f(x_k)$

$\longrightarrow$ [Conn, Gould, Toint; SIAM 2000]

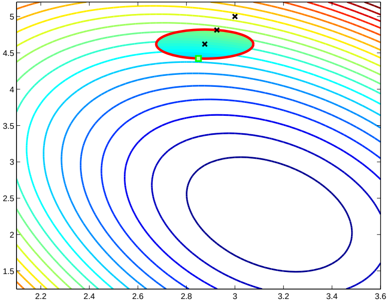
Model-Based Trust-Region Algorithms

Substitute $\min\{q_k(x) : x \in \mathcal{B}_k\}$ for $\min f(x)$

$$q_k(x) = f(x_k) + g_k^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H_k (x - x_k)$$

f expensive, no ∇f

q_k cheap, analytic derivatives



Trust region: $\mathcal{B}_k = \{x \in \Omega : \|x - x_k\| \le \Delta_k\}$

- Trust $q_k \approx f$ in $\mathcal{B}_k$
- Update based on $\rho_k = \frac{f(x_k) - f(x_+)}{q_k(x_k) - q_k(x_+)}$

Typical models

- Only need (occasional) local approximation
- Taylor-based: $g_k = \nabla f(x_k)$, $H_k \approx \nabla^2 f(x_k)$

$\longrightarrow$ [Conn, Gould, Toint; SIAM 2000]

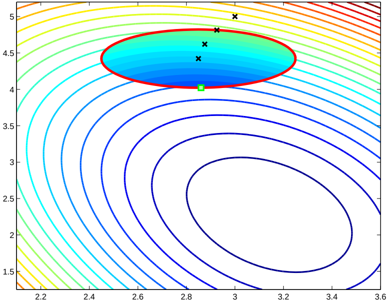
Model-Based Trust-Region Algorithms

Substitute $\min\{q_k(x) : x \in \mathcal{B}_k\}$ for $\min f(x)$

$$q_k(x) = f(x_k) + g_k^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H_k (x - x_k)$$

f expensive, no ∇f

q_k cheap, analytic derivatives



Trust region:
$\mathcal{B}_k = \{x \in \Omega : \|x - x_k\| \le \Delta_k\}$

- Trust $q_k \approx f$ in $\mathcal{B}_k$
- Update based on $\rho_k = \frac{f(x_k) - f(x_+)}{q_k(x_k) - q_k(x_+)}$

Typical models

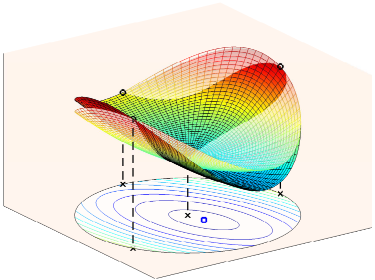
- Only need (occasional) local approximation
- Taylor-based: $g_k = \nabla f(x_k)$, $H_k \approx \nabla^2 f(x_k)$

$\longrightarrow$ [Conn, Gould, Toint; SIAM 2000]

Black-Box Algorithms: Building Models Without Derivatives

Given data $(\mathcal{X}_k, f(\mathcal{X}_k))$ and basis Φ, "solve"

$$\Phi(\mathcal{X}_k)z = \begin{bmatrix} \Phi_c & \Phi_g & \Phi_H \end{bmatrix} \begin{bmatrix} z_c \\ z_g \\ z_H \end{bmatrix} = \underline{\mathrm{f}} = f(\mathcal{X}_k)$$



$n = 2, \, |\mathcal{X}_k| = 4$

Full quadratics, $|\mathcal{X}_k| = \frac{(n+1)(n+2)}{2}$

- Interpolation: $q_k(y_i) = f(y_i), \quad \forall y_i \in \mathcal{X}_k$
- Geometric conditions on points in $\mathcal{X}_k$

Undetermined interp., $|\mathcal{X}_k| < \frac{(n+1)(n+2)}{2}$

- Use (Powell) Hessian updates

$$\min_{g_k, H_k} \quad \|H_k - H_{k-1}\|_F^2 \qquad \text{s.t.} \quad q_k = \underline{\mathrm{f}} \text{ on } \mathcal{X}_k$$

Regression, $|\mathcal{X}_k| > \frac{(n+1)(n+2)}{2}$

- Solve $\min_z \|\Phi z - \underline{\mathrm{f}}\|$

Multivariate (Scattered Data) Interpolation is a Different Kind of Animal

$$m(y_i) = f(y_i) \qquad \forall y_i \in \mathcal{X}$$

$n = 1$ Given distinct points, can find a unique degree $|\mathcal{X}| - 1$ polynomial m

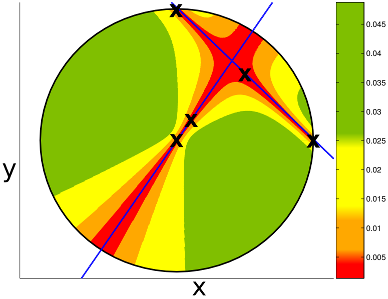
$n > 1$ Not true! (see Mairhuber-Curtis Theorem)

Multivariate (Scattered Data) Interpolation is a Different Kind of Animal

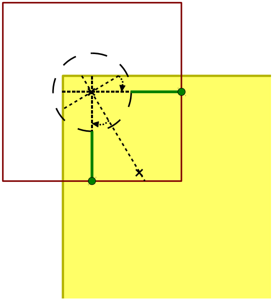
$$m(y_i) = f(y_i) \qquad \forall y_i \in \mathcal{X}$$

$n = 1$ Given distinct points, can find a unique degree $|\mathcal{X}| - 1$ polynomial m

$n > 1$ Not true! (see Mairhuber-Curtis Theorem)



6th point for a quadratic in $\mathbb{R}^2$



Nearby constraints affect geometry

→ [Wendland; CUP 2010]

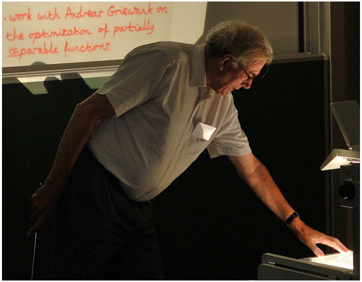
Convergence to Stationary Points & Software

$\lim_{k\to\infty} \nabla f(x_k) = 0$ provided:

0. f is sufficiently smooth and regular (e.g., bounded level sets)
1. Control $\mathcal{B}_k$ based on model quality
2. (Occasional) approximation within $\mathcal{B}_k$

   Our quadratics satisfy

   - $|q_k(x) - f(x)| \le \kappa_1(\gamma_f + \|H_k\|)\Delta_k^2, \qquad \forall x \in \mathcal{B}_k$
   - $\|g_k + H_k(x - x_k) - \nabla f(x)\| \le \kappa_2(\gamma_f + \|H_k\|)\Delta_k, \qquad \forall x \in \mathcal{B}_k$
3. Sufficient decrease



Michael J.D. Powell, 1936-2015

Survey →[Conn, Scheinberg, Vicente; SIAM 2009]

Methods →[Powell: COBYLA, UOBYQA, NEWUOA, BOBYQA, LINCOA],

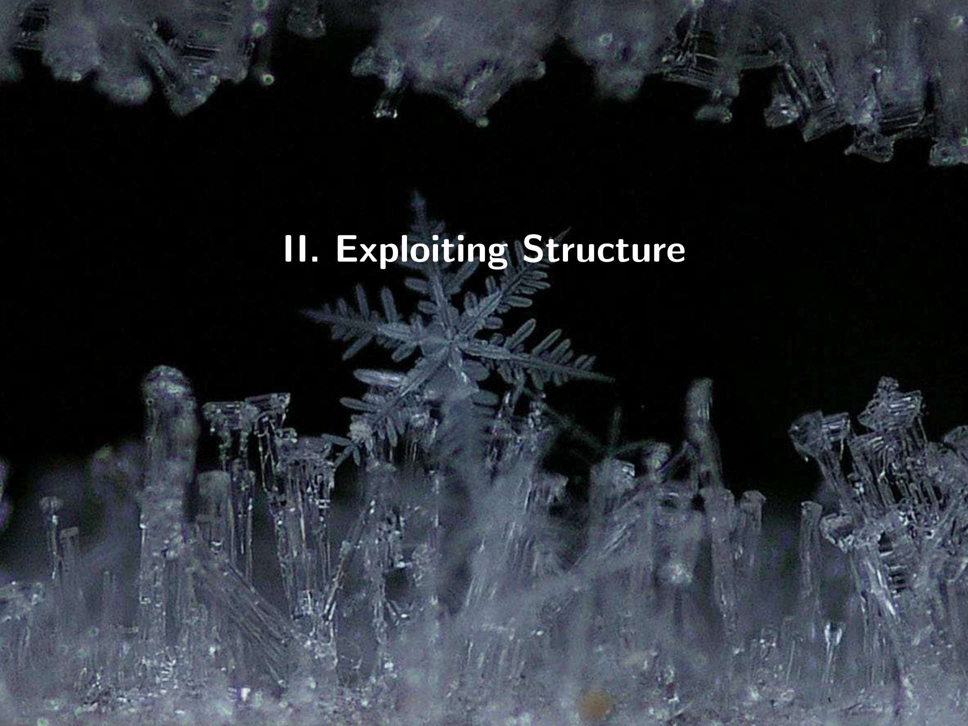
...

Line search methods also work →[Kelley et al; IFFCO]

RBF models also work →[W. & Shoemaker; SIREV 2013]

Probabilistic models→[Bandeira, Scheinberg, Vicente; SIOPT 2014]

II. Exploiting Structure



Structure in Simulation-Based Optimization, $\min f(x) = F[x, S(x)]$

f is often not a black box S

NLS Nonlinear least squares

$$f(x) = \sum_i (S_i(x) - d_i)^2$$

CNO Composite (nonsmooth) optimization

$$f(x) = h(S(x))$$

SKP Not all variables enter simulation

$$f(x) = g(x_I, x_J) + h(S(x_J))$$

SCO Only some constraints depend on simulation

$$\min\{f(x) : c_1(x) = 0, c_S(x) = 0\}$$

$+$ Slack variables

$$\Omega_S = \{(x_I, x_J) : S(x_J) + x_I = 0, x_I \geq 0\}$$

...

Model-based methods offer one way to exploit such structure

General Setting – Modeling Smooth $S_1(x), S_2(x), \ldots, S_p(x)$

Assume:

- each S_i is continuously differentiable, available
- each ∇S_i is Lipschitz continuous, unavailable

General Setting – Modeling Smooth $S_1(x), S_2(x), \ldots, S_p(x)$

Assume:

- each S_i is continuously differentiable, available
- each ∇S_i is Lipschitz continuous, unavailable

$m^{S_i} : \mathbb{R}^n \to \mathbb{R}$ approximates S_i on $\mathcal{B}(x, \Delta)$ $\qquad i = 1, \ldots, p$

Fully Linear Models

m^{S_i} fully linear on $\mathcal{B}(x, \Delta)$ if there exist constants $\kappa_{i,\mathrm{ef}}$ and $\kappa_{i,\mathrm{eg}}$ independent of x and Δ so that

$$
\begin{aligned}
|S_i(x+s) - m^{S_i}(x+s)| &\le \kappa_{i,\mathrm{ef}} \Delta^2 \quad && \forall s \in \mathcal{B}(0, \Delta) \\
\|\nabla S_i(x+s) - \nabla m^{S_i}(x+s)\| &\le \kappa_{i,\mathrm{eg}} \Delta \quad && \forall s \in \mathcal{B}(0, \Delta)
\end{aligned}
$$

NLS– Nonlinear Least Squares $f(x) = \frac{1}{2}\sum_i R_i(x)^2$

Obtain a vector of output $R_1(x), \ldots, R_p(x)$

- Model each R_i

$$R_i(x) \approx m_k^{R_i}(x) = R_i(x_k) + (x - x_k)^\top g_k^{(i)} + \frac{1}{2}(x - x_k)^\top H_k^{(i)}(x - x_k)$$

- Approximate:

$$\begin{aligned}
\nabla f(x) &= \sum_i \nabla \mathbf{R_i(x)} R_i(x) \qquad \longrightarrow \sum_i \nabla m_k^{R_i}(x) R_i(x) \\
\nabla^2 f(x) &= \sum_i \nabla \mathbf{R_i(x)} \nabla \mathbf{R_i(x)}^\top + \sum_i R_i(x) \nabla^2 \mathbf{R_i(x)} \\
&\longrightarrow \sum_i \nabla m_k^{R_i}(x) \nabla m_k^{R_i}(x)^\top + \sum_i R_i(x) \nabla^2 m_k^{R_i}(x)
\end{aligned}$$

- Model f via Gauss-Newton or similar

regularized Hessians $\longrightarrow$ DFLS [Zhang, Conn, Scheinberg]

full Newton $\longrightarrow$ POUNDERS [W., Moré]

NLS– Consequences for $f(x) = \frac{1}{2}\sum_i R_i(x)^2$

Pay a (negligible for expensive S) price in terms of p models

- Save linear algebra using interpolation set $\mathcal{X}_k$ common to all models
  - Single system solve, multiple right hand sides

$$\Phi(\mathcal{X}_k)\left[\begin{array}{ccc} z^{(1)} & \cdots & z^{(p)} \end{array}\right] = \left[\begin{array}{ccc} \underline{\mathsf{R}}_1 & \cdots & \underline{\mathsf{R}}_p \end{array}\right]$$

  - m^{R_1} quality $\Rightarrow$ quality of all m^{R_i}

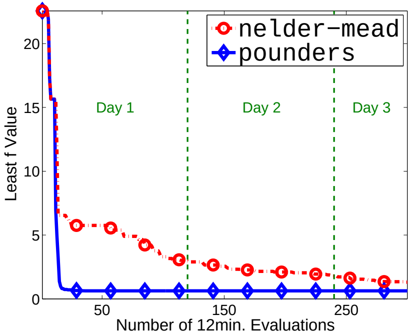
+ (nearly) exact gradients for R_i (nearly) linear
- No longer interpolate function at data points

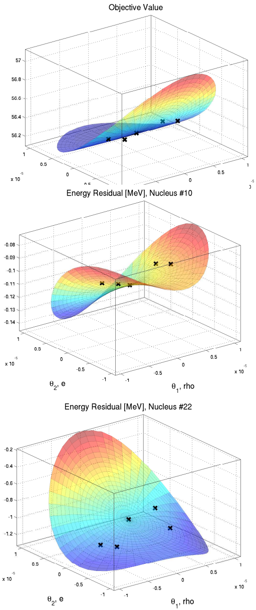
$$\begin{aligned} m(x_k+\delta) = \quad & f(x_k) \\ & +\delta^\top \textstyle\sum_i g_k^{(i)} R_i(x_k) \\ & +\tfrac{1}{2}\delta^\top \textstyle\sum_i \left(g_k^{(i)} (g_k^{(i)})^\top + R_i(x_k) H_k^{(i)} \right) \delta \\ & + \text{missing h.o. terms} \end{aligned}$$

NLS– POUNDERS in Practice: DFT Calibration/MLE

$$\min_x \sum_{i=1}^{p} w_i \left(S_i(x) - d_i\right)^2$$

- $S_i(x)$ Simulated (DFT) nucleus property
- d_i Experimental data i
- w_i Weight for data type i
- p Parallel simulations (12 wallclock mins)





→[Kortelainen et al., PhysRevC 2010]

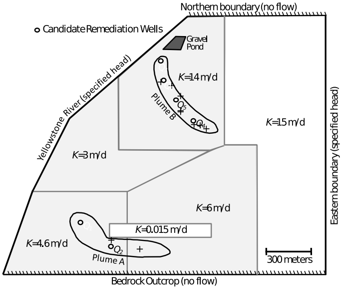
CNO– Composite Nonsmooth Optimization Examples

Ex.- Groundwater remediation

Determine rates x for extraction/injection wells

- Regulator's simulator returns flow $S_i(x)$ in/out of cell i
- Minimize plume fluxes (e.g., regulatory \$ penalties) $f(x) = \sum_i |S_i(x)|$

→ See MS90 later today



Lockwood Solvent Ground Water Plume Site (LSGPS)

Ex.- Particle accelerator design

Minimize particle losses: $f(x) = \max_{t_i \in \mathcal{T}_1} S(x; t_i) - \min_{t_i \in \mathcal{T}_2(x)} S(x; t_i)$

CNO– Composite Nonsmooth Optimization $f(x) = h(S(x); x)$

nonsmooth (algebraically available) function $h : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$
of a smooth (blackbox) mapping $S : \mathbb{R}^n \to \mathbb{R}^p$

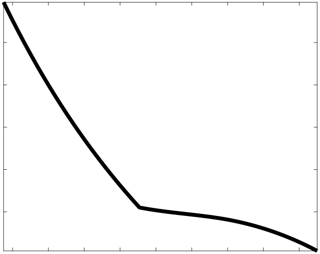
CNO– Composite Nonsmooth Optimization $f(x) = h(S(x); x)$

nonsmooth (algebraically available) function $h : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$
of a smooth (blackbox) mapping $S : \mathbb{R}^n \to \mathbb{R}^p$

Basic Idea: Knowledge of vector $S(x^k)$ & potential nondifferentiability at $S(x^k)$ should enhance (theoretical and practical) progress to a stationary point

Ex.- $f^1(x) = \|S(x)\|_1 = \sum_{i=1}^p |S_i(x)|$

$$\partial f^1(x) = \sum_{i:S_i(x)\neq 0} \operatorname{sgn}(S_i(x))\nabla S_i(x) + \sum_{i:S_i(x)=0} \mathbf{co}\left\{-\nabla S_i(x), \nabla S_i(x)\right\}$$



◇ $\mathcal{D}^c = \{x : \exists i$ with $S_i(x) = 0, \nabla S_i(x) \neq 0\}$

+ Compact $\partial f(x)$
- $\mathcal{D}^c$ depends on $\nabla S_i(x)$

CNO– The Nuisance Set, $\mathcal{N}$

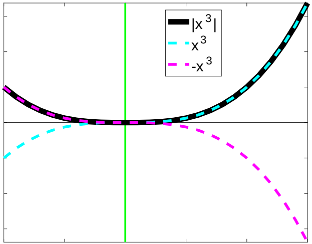
Relaxation $\mathcal{N} \subseteq \mathcal{D}^c$ using only zero-order information

f^1:

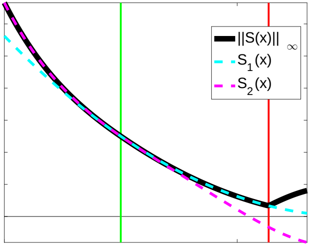
$$\mathcal{N} = \{x : \exists i \text{ with } S_i(x) = 0\}$$

f^∞:

$$\mathcal{N} = \left\{x : f^\infty(x) = 0 \text{ or } \left|\arg\max_i |S_i(x)|\right| > 1\right\}$$



$f^1(x) = |x^3|$



$f^\infty(x) = \max\{|S_1(x)|, |S_2(x)|\}$

CNO– The Nuisance Set, $\mathcal{N}$

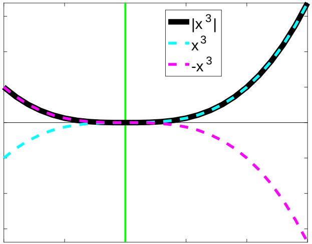
Relaxation $\mathcal{N} \subseteq \mathcal{D}^c$ using only zero-order information

f^1:

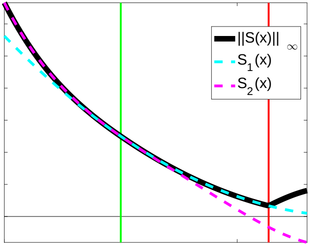
$$\mathcal{N} = \{x : \exists i \text{ with } S_i(x) = 0\}$$

f^∞:

$$\mathcal{N} = \left\{x : f^\infty(x) = 0 \text{ or } \left|\arg\max_i |S_i(x)|\right| > 1\right\}$$



$f^1(x) = |x^3|$



$f^\infty(x) = \max\{|S_1(x)|, |S_2(x)|\}$

Observe

When $x^k \notin \mathcal{N}$,

$$\begin{aligned}\partial f(x^k) &= \nabla f(x^k)\\ &= \nabla_x S(x^k)^\top \nabla_S h(S(x^k))\\ &\approx \nabla_x M(x^k)^\top \nabla_S h(S(x^k))\end{aligned}$$

and smooth approximation is justified

CNO– Subdifferential Approximation

- $x^k \in \mathcal{N}$, we build a set of generators $\mathcal{G}(x^k)$ based on $\partial_S h(S(x^k))$.
  - $\mathbf{co}\left\{\mathcal{G}(x^k)\right\}$ approximates $\partial f(x^k)$

Ex.- $f^1(x) = \|S(x)\|_1$

$$\mathcal{G}(x^k) = \nabla M(x^k)^\top \left\{ \operatorname{sgn}(S(x^k)) + \bigcup_{i:S_i(x^k)=0} \{-e_i, 0, e_i\} \right\}$$

CNO– Subdifferential Approximation

- $x^k \in \mathcal{N}$, we build a set of generators $\mathcal{G}(x^k)$ based on $\partial_S h(S(x^k))$.
  - $\mathbf{co}\left\{\mathcal{G}(x^k)\right\}$ approximates $\partial f(x^k)$

Ex.- $f^1(x) = \|S(x)\|_1$

$$\mathcal{G}(x^k) = \nabla M(x^k)^\top \left\{ \operatorname{sgn}(S(x^k)) + \bigcup_{i:S_i(x^k)=0} \{-e_i, 0, e_i\} \right\}$$

Nearby data $\mathcal{X} \subset \mathcal{B}(x^k, \Delta_k)$ informs models $M = m^S$ and generator set

- Manifold sampling method uses manifold(s) of $\mathcal{X}$

$$\nabla M(x^k)^\top \bigcup_{y^i \in \mathcal{X}} \mathbf{mani}\left(S(y^i)\right)$$

- Traditional gradient sampling $\longrightarrow$ [Burke, Lewis, Overton; SIOPT 2005]

$$\bigcup_{y^i \in \mathcal{X}} \nabla M(y^i)^\top \mathbf{mani}\left(S(y^i)\right)$$

CNO– Smooth Trust-Region Subproblem

Smooth master model from minimum-norm element

$$m^f(x^k+s) = f(x^k) + \left\langle s, \mathbf{proj}\left(0, \mathbf{co}\left\{\mathcal{G}(x^k)\right\}\right)\right\rangle + \cdots$$

CNO– Smooth Trust-Region Subproblem

Smooth master model from minimum-norm element

$$m^f\left(x^k+s\right)=f(x^k)+\left\langle s,\mathbf{proj}\left(0,\mathbf{co}\left\{\mathcal{G}(x^k)\right\}\right)\right\rangle+\cdots$$

$\Rightarrow$ smooth subproblems

$$\min\left\{m^f\left(x^k+s\right):s\in\mathcal{B}(0,\Delta_k)\right\}$$

- Convex h (e.g., $\|S(x)\|_1$) and ∇S_i is Lipschitz
  $\Rightarrow$ every cluster point of $\{x^k\}_k$ is Clarke stationary
  $\longrightarrow$ [Larson, Menickelly, W.; Preprint 2016]
- OK to sample at $x^k\in\mathcal{D}^C$
- More general (piecewise differentiable f) results:
  $\longrightarrow$ [Larson, Khan, W.; in prog. 2016]
  (yesterday in MS155!)

vs.

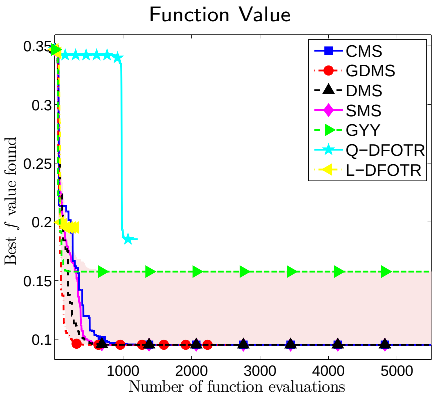
Nonsmooth subproblems

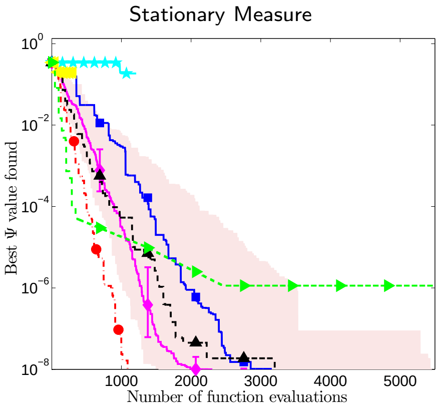
$$\min\left\{h\left(M\left(x^k+s\right)\right):s\in\mathcal{B}(0,\Delta_k)\right\}$$

- Requires convex h
  $\longrightarrow$ [Fletcher; MathProgStudy 1982]
  $\longrightarrow$ [Grapiglia, Yuan, Yuan; C&A Math. 2016]
  Complexity results
  $\longrightarrow$ [Garmanjani, Júdice, Vicente; SIOPT 2016]

Roger Fletcher

CNO– Example Performance on L_1 Test Problems





Smooth black-box methods can fail in practice, even when $\mathcal{D}^C$ has measure zero

Numerical tests: $\rightarrow$ [Larson, Menickelly, W.; Preprint 2016]

SKP– Some Known Partials Example

Ex.- Bi-level model calibration structure

$$\min_x \left\{ f(x) = \sum_{i=1}^{p} (S_i(x) - d_i)^2 \right\}$$

$S_i(x)$ solution to lower-level problem depending only on x_J

$$\begin{aligned} S_i(x) &= g_i(x) + \min_y \{h_i(x_J; y) : y \in \mathcal{D}_i\} \\ &= g_i(x) + h_i(x_J; y_{i,*}[x_J]) \end{aligned}$$

For $x = (x_I, x_J)$

- $\nabla_{x_I} S_i(x_I, x_J)$ available
- $\nabla_{x_J} S_i(x) \approx \nabla_{x_J} g_i(x) + \nabla_{x_J} m^{\tilde{S}_i}(x_J)$
- $S_i(x)$ continuous and smooth in x_I
- $g_i(x)$ cheap to compute!
- No noise/errors introduced in $g_i(x)$

General bi-level $\longrightarrow$[Conn & Vicente, OMS 2012]

SKP– Some Known Partials

$x = (x_I, x_J)$; have $\frac{\partial f}{\partial x_I}$ but not $\frac{\partial f}{\partial x_J}$

"Solve"

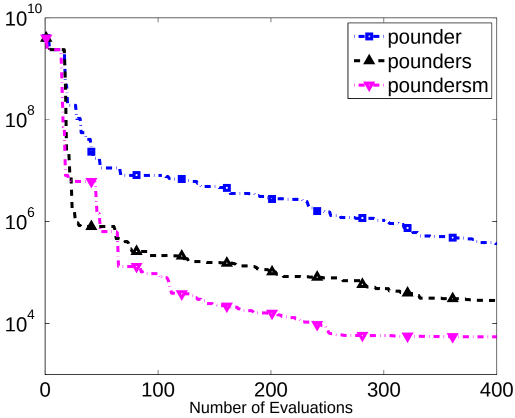
$$\Phi z = \underline{\mathsf{f}}$$

with known $z_{g,I}, z_{H,I}$

$$\begin{bmatrix} \Phi_c & \Phi_{g,J} & \Phi_{H,J} \end{bmatrix} \begin{bmatrix} z_c \\ z_{g,J} \\ z_{H,J} \end{bmatrix} = \underline{\mathsf{f}} - \Phi_{g,I} z_{g,I} - \Phi_{H,I} z_{H,I}$$

- Still have interpolation where required
- Effectively lowers dimension to $|J| = n - |I|$ for
  - approximation
  - model-improving evaluations
  - linear algebra
- $\lim_{k\to\infty} \nabla f(x_k) = 0$ as before:
  - Guaranteed descent in some directions

SKP– Numerical Results With Some Partials



Three approaches:

- \- black box
- s exploit least squares
- m use ∇_{x_I} derivatives

◇ $n = 16$, $|I| = 3$

◇ 5-10 secs/evaluation

Same algorithmic framework, performance advantages from exploiting structure

→[Bertolli, Papenbrock, W., PRC 2012]

SCO- General Constraints

$$\min\{f(x) : c_1(x) = 0, c_S(x) = 0\}$$

- Lagrangian (key to optimality conditions):

$$\begin{aligned} \nabla L &= \nabla f + \lambda_1^\top \nabla c_1 + \lambda_2^\top \nabla \mathbf{c_S} \\ &\to \nabla f + \lambda_1^\top \nabla c_1 + \lambda_2^\top \nabla m \end{aligned}$$

- Use favorite method: filters, augmented Lagrangian, ...
- Slack variables
  - Do not increase effective dimension
  - Subproblems can treat separately
  - Know derivatives

→[Lewis & Torczon; 2010]

Modified AL methods →[Diniz-Ehrhardt, Martínez, Pedroso; C&A Math. 2011]

SBO constraints have unique properties →[Le Digabel & W.; ANL/MCS-P5350-0515 2016]

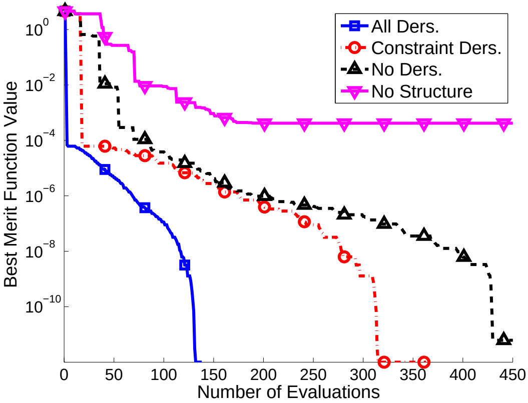
SCO– What Constraint Derivatives Buy You

Ex.- Augmented Lagrangian methods, $L_A(x, \lambda; \mu) = f(x) - \lambda^\top c(x) + \frac{1}{\mu}\|c(x)\|^2$

$\min_x \{f(x) : c(x) = 0\}$

Four approaches:

1. Penalize constraints
2. Treat c and f both as (separate) black boxes
3. Work with f and $\nabla_x c$
4. Have both $\nabla_x f$ and $\nabla_x c$



$n = 15$, 11 constraints

OPTIMIZE EVERYTHING

Mathematically unwrap problems to expose (the deepest) black boxes

- Structure is everywhere, even in legacy-code-driven optimization problems
- Exploiting structure is one way to expand range of optimization to solve grand-challenge problems
- Sacrifice little in convenience
  - Output & model residuals $\{r_i(x)\}_i$, not $\|r(x)\|$
  - Output & model constraints $\{c_i(x)\}_i$, not a penalty $P(c(x))$
  - Explicitly handle nonsmoothness (and noise, . . .)

- Papers and links at `www.mcs.anl.gov/~wild`
- Collaborators in this work:

Awesome opportunities for students

Aswin Kannan (UIUC), Slava Kungurtsev (UCSD), Jeff Larson (UC-Denver), Matt Menickelly (Lehigh)

. . . and postdocs!

Prasanna Balaprakash (UL Bruxelles), Kamil Khan (MIT)

OPTIMIZE EVERYTHING

Mathematically unwrap problems to expose (the deepest) black boxes

- Structure is everywhere, even in legacy-code-driven optimization problems
- Exploiting structure is one way to expand range of optimization to solve grand-challenge problems
- Sacrifice little in convenience
  - Output & model residuals $\{r_i(x)\}_i$, not $\|r(x)\|$
  - Output & model constraints $\{c_i(x)\}_i$, not a penalty $P(c(x))$
  - Explicitly handle nonsmoothness (and noise, . . .)

- Papers and links at `www.mcs.anl.gov/~wild`
- Collaborators in this work:

Awesome opportunities for students

Aswin Kannan (UIUC), Slava Kungurtsev (UCSD),
Jeff Larson (UC-Denver), Matt Menickelly (Lehigh)

. . . and postdocs!

Prasanna Balaprakash (UL Bruxelles), Kamil Khan (MIT)

Thank YOU!