An Overview of Optimization Formulations and Methods for Nuclear Theory

Stefan Wild

Mathematics and Computer Science Division
Argonne National Laboratory

Joint work with Jared O'Neal
and many physicist collaborators:
A. Ekström, C. Forssén, G. Hagen, M. Hjorth-Jensen, G.R. Jansen, M. Kortelainen, T. Lesinski, A. Lovell, R. Machleidt, J. McDonnell, H. Nam, N. Michel, W. Nazarewicz, F.M. Nunes, E. Olsen, T. Papenbrock, P.-G. Reinhardt, N. Schunck, M. Stoitsov, J. Vary, K. Wendt, **and others**

The Plan

1. Optimization background
   - Local and global
   - Derivatives and no derivatives
2. Typical optimization-based formulations
   - Nonlinear least squares
   - POUNDERS
   - Early Fayan's functional experiments + importance of scaling
3. Optimization under uncertainty
   - Stochastic optimization
   - Robust optimization
   - Trimmed optimization

Mathematical/Numerical Nonlinear Optimization

Find parameters $x = (x_1, \ldots, x_n)$ in domain Ω to improve objective f

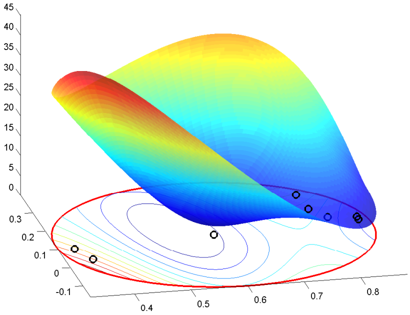
$$\min \{f(x) : x \in \Omega \subseteq \mathbb{R}^n\}$$

- (Unless Ω is very special) Need to evaluate f at many x to find a good $\hat{x}_*$

Here:

- Assume f is deterministic (and smooth except where noted)
- Assume that uncertainty modeled through constraints and objective(s)

$\longrightarrow$ *part 3*



Parameter Estimation is NOT a Generic/Blackbox Optimization Problem

Generic:

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

x n decision variables

f $: \mathbb{R}^n \to \mathbb{R}$ objective function

Ω feasible region,
$\{x : c_E(x) = 0, c_I(x) \leq 0\}$

- c_E (vector of) equality constraints
- c_I (vector of) inequality constraints

Parameter Estimation is NOT a Generic/Blackbox Optimization Problem

Generic:

$$\min_x \{f(x) : x \in \Omega \subseteq \mathbb{R}^n\}$$

- x n decision variables
- f $: \mathbb{R}^n \to \mathbb{R}$ objective function
- Ω feasible region, $\{x : c_E(x) = 0, c_I(x) \leq 0\}$
  - c_E (vector of) equality constraints
  - c_I (vector of) inequality constraints

Typical calibration problem:

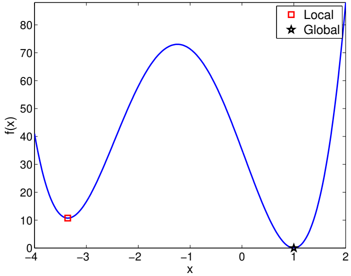
$$f(x) = \|\mathbf{R}(x)\|_2^2 = \sum_{i=1}^{p} R_i(x)^2$$

- x n coupling constants
- R_i $: \mathbb{R}^n \to \mathbb{R}$ residual function
  - Ex.- $\frac{1}{w_i}(S(x;\theta_i) - d_i)$
    - $S(x;\theta_i)$: numerical simulation
  - Ex.- Obtain $\chi^2(x)$ by $\frac{1}{p-n} f(x)$
- Ω $= \{x : \mathbf{l} \leq x \leq \mathbf{u}\}$
  - Finite bounds (for some x_i)
  - Often dictated by dom(S)

[Ekström et al, PRL 2013] [Kortelainen et al, PRC 2014]

Taking advantage of structure should reduce expense/improve accuracy

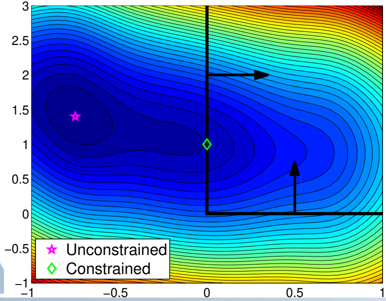
Local and Global Solutions

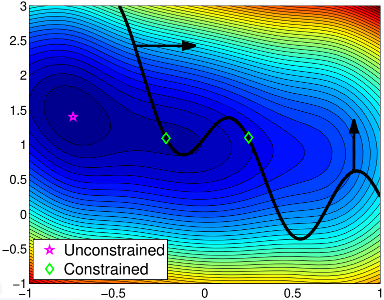


- **Local minimizer** $\hat{x}_*$:

$$f(\hat{x}_*) \leq f(x) \; \forall x \in \mathcal{N}(\hat{x}_*) \cap \Omega$$

- **Global convergence**: Convergence (to a local solution/stationary point) from anywhere in Ω
- **Convergence to a global minimizer**: Obtain x_* with $f(x_*) \leq f(x) \, \forall x \in \Omega$





Why Not Global Optimization, $\min_{x\in\Omega} f(x)$?

Anyone selling you global solutions when derivatives are unavailable:

either assumes more about your problem (e.g., convex f)

or expects you to wait forever

Törn and Žilinskas: An algorithm converges to the global minimum for any continuous f if and only if the sequence of points visited by the algorithm is dense in Ω.

Why Not Global Optimization, $\min_{x \in \Omega} f(x)$?

Anyone selling you global solutions when derivatives are unavailable:

either assumes more about your problem (e.g., convex f)

or expects you to wait forever

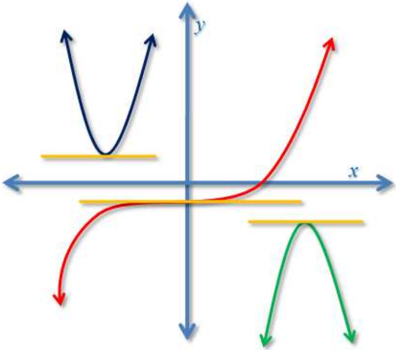
Törn and Žilinskas: An algorithm converges to the global minimum for any continuous f if and only if the sequence of points visited by the algorithm is dense in Ω.

Instead:

- Rapidly find good local solutions and/or be robust to poor solutions
- Find several good local solutions concurrently (`APOSMM`/`LibEnsemble`)
- Exploit parallelism afforded by statistical/Bayesian/space-filling designs

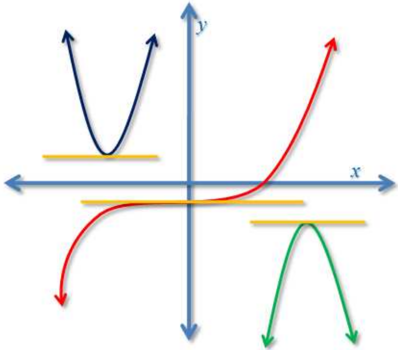
Optimization Tightly Coupled With Derivatives (WRT Parameters)

Typically necessary for optimality: $\nabla_x f(x_*) + \lambda^T \nabla_x c_E(x_*) = 0, c_E(x_*) = 0$



Optimization Tightly Coupled With Derivatives (WRT Parameters)

Typically necessary for optimality: $\nabla_x f(x_*) + \lambda^T \nabla_x c_E(x_*) = 0, c_E(x_*) = 0$



Algorithmic/Automatic Differentiation (AD)

"Exact* derivatives!"

? No black boxes allowed

? Not always automatic/"cheap"

Finite Differences (FD)

"Nonintrusive", "Numerical Differentiation"

? Expense grows with n

? Sensitive to stepsize choice/noise
→[Moré & W.; SISC 2011], [Moré & W.; TOMS 2012]

But some derivatives are not always available/do not always exist

(Computationally Expensive) Simulation-Based Optimization

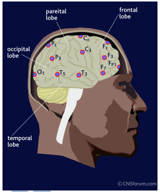
$$\min_{x \in \mathbb{R}^n} \{ f(x) = F[\mathbf{S}(\mathbf{x})] : c(\mathbf{S}(\mathbf{x})) \leq 0, x \in \mathcal{B} \}$$

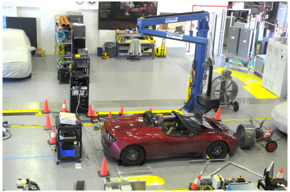
"parameter estimation", "model calibration", "design optimization", . . .

- Evaluating S means running a simulation modeling some (smooth) process
- S can contribute to objective and/or constraints, possibly noisy
- Derivatives $\nabla_x S$ often unavailable or prohibitively expensive to obtain
- S (even when parallelized) takes secs/mins/days

Evaluation is a bottleneck for optimization

$\mathcal{B}$ compact, known region (e.g., finite bound constraints)





Typical Optimization-Based Formulations

Standard "χ^2"-based objective

$$f(x) = \frac{1}{p-n}\sum_{i=1}^{p} R_i(x)^2 = \frac{1}{p-n}\sum_{i=1}^{p}\left(\frac{S(x;\theta_i) - d_i}{\sigma_i}\right)^2$$

$d_1, \ldots, d_p$: the data

$S(x;\theta_i)$: the ith simulation (modeled/theory) output given parameters x

$\sigma_1, \ldots, \sigma_p$: the (inverse) weights

Typical Optimization-Based Formulations

Standard "χ^2"-based objective

$$f(x) = \frac{1}{p-n}\sum_{i=1}^{p} R_i(x)^2 = \frac{1}{p-n}\sum_{i=1}^{p}\left(\frac{S(x;\theta_i)-d_i}{\sigma_i}\right)^2$$

$d_1,\ldots,d_p$: the data

$S(x;\theta_i)$: the ith simulation (modeled/theory) output given parameters x

$\sigma_1,\ldots,\sigma_p$: the (inverse) weights

NB-

- Multiplying f by positive constant does not affect the solution of $\min_x f(x)$
- $\Rightarrow$ all σ could be multiplied by a common constant
- $\Rightarrow$ interpretation of $f(x)$ values comes from something other than the optimization

Exploiting Nonlinear Least Squares Structure

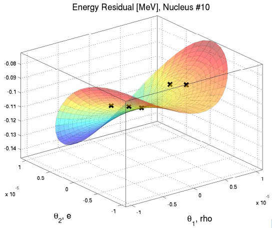
Obtain a *vector* of output $R_1(x), \ldots, R_p(x)$

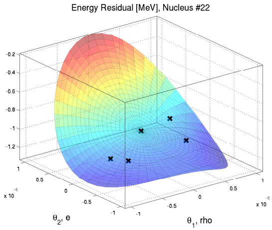
- (Locally) Model each R_i by a surrogate $q_k^{(i)}$

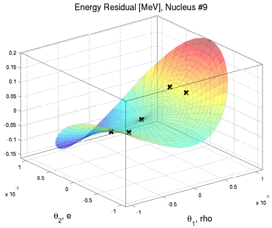
$$R_i(x) \approx q_k^{(i)}(x) = R_i(x_k) + (x - x_k)^\top \mathbf{g}_k^{(i)} + \frac{1}{2}(x - x_k)^\top \mathbf{H}_k^{(i)}(x - x_k)$$

- Employ models in the approximation

$$\begin{aligned} \nabla f(x) &= \sum_i \nabla \mathbf{R_i}(\mathbf{x}) R_i(x) && \to \sum_i g_k^{(i)}(x) R_i(x) \\ \nabla^2 f(x) &= \sum_i \nabla \mathbf{R_i}(\mathbf{x}) \nabla \mathbf{R_i}(\mathbf{x})^T + R_i(x) \nabla^2 \mathbf{R_i}(\mathbf{x}) && \to \sum_i \mathbf{g}_k^{(i)}(x) \mathbf{g}_k^{(i)}(x)^T + R_i(x) \mathbf{H}_k^{(i)}(x) \end{aligned}$$







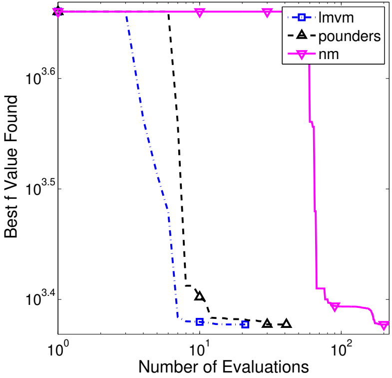
The POUNDERS Method & Open-Source Software

Practical Optimization Using No DERivatives for sums of Squares

- a local, model-based, full Newton-like, trust-region algorithm
- for unconstrained and bound-constrained
- nonlinear-least squares problems
- in the absence of some derivatives (derivative-free)

that

- is a misnomer (uses some derivatives)
- is robust to noise/poor local minima
- has a simple interface (provide routine for $\mathbf{S}$)
- allows for parallel evaluation of $\mathbf{S}$
- has asymptotic convergence guarantees
- performs well in practice
- is available in PETSc/TAO [http://mcs.anl.gov/tao]



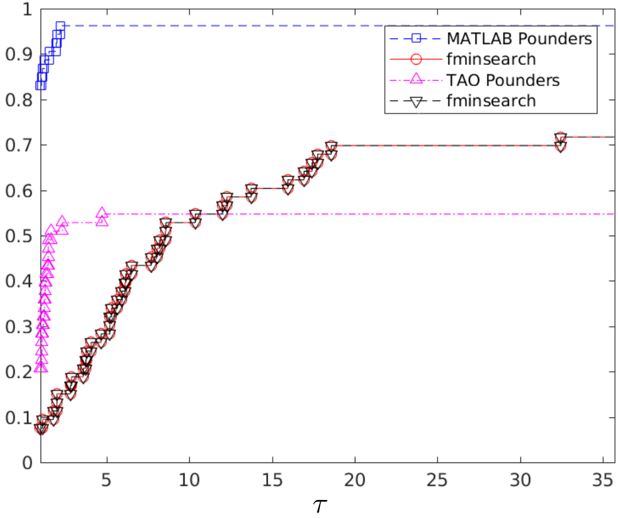
TAO solvers

- **nm** $\nabla_x f$ unavailable, black box
- **pounders** $\nabla_x f$ unavailable, exploits problem structure
- **lmvm** Uses available $\nabla_x f$

POUNDERS Variants

Performance profiles: Proportion of problems solved within a factor τ of the fastest solver

- fminsearch is N-M
- Feedback from nuclear physics users helps us solve (your) problems faster/better
- Behavior shown now fixed in PETSc/TAO!



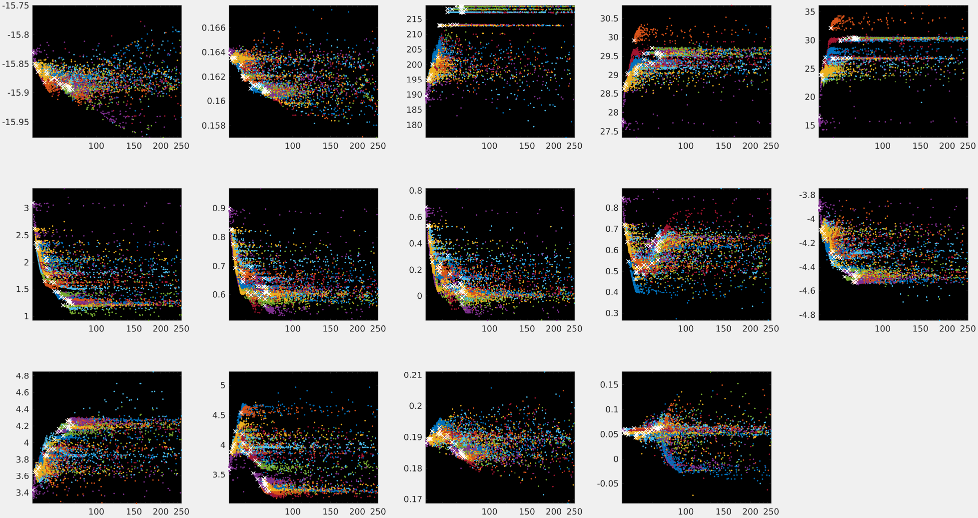
A Quick Example: Fayan's Functional

- Work with functional form proposed in [S.A. Fayans, JETP Lett. 68, 169 (1998)]
- Analyze FaNDF0 functional from [Reinhard & Nazarewicz, PRC 95, 064328 (2017)]
- 201+ observables
  - binding energies, radii, surface thickness, pairing gaps, ...
- 14+ model parameters
  - including 7 nuclear matter parameters

Computing

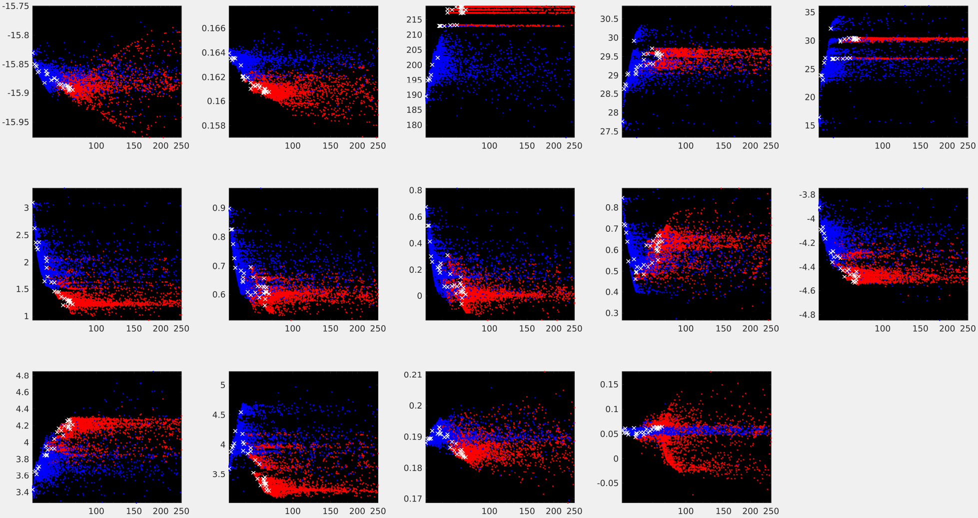
- We compute all observables in a couple of seconds using 192 cores
- Inexpensive: can benchmark optimization and UQ algorithms
- Currently: Run optimization for a few hours, change initial configuration/hyperparameters, repeat

Fayan’s Initial Experiments: Importance of Scaling



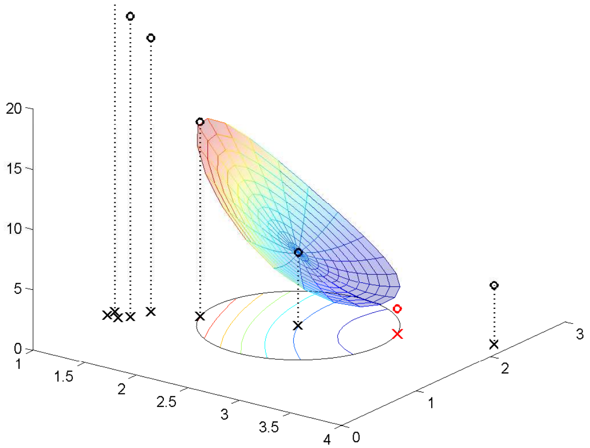
ISNET-6

Fayan's Initial Experiments: Importance of Scaling



ISNET-6

Why Scaling? Interpolation-Based Trust-Region Methods

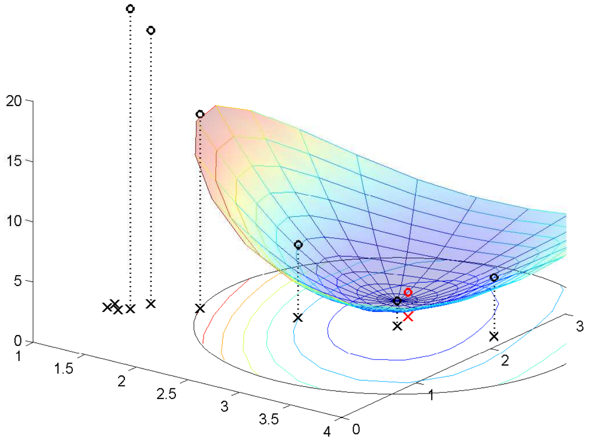


Basic trust region iteration:

- Build surrogate model m (POUNDERS: for each residual R_i)
- Trust approximation of m within region $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$ **NB**- This norm (could but) is not changing
- Use m to obtain next point within $\mathcal{B}$ for evaluation

Incorporate prior knowledge through scaling, norm selection, initial Δ_0

Why Scaling? Interpolation-Based Trust-Region Methods

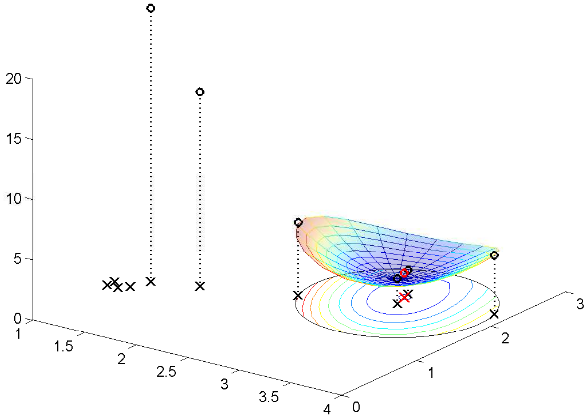


Basic trust region iteration:

- Build surrogate model m (POUNDERS: for each residual R_i)
- Trust approximation of m within region $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$ **NB**- This norm (could but) is not changing
- Use m to obtain next point within $\mathcal{B}$ for evaluation

Incorporate prior knowledge through scaling, norm selection, initial Δ_0

Why Scaling? Interpolation-Based Trust-Region Methods

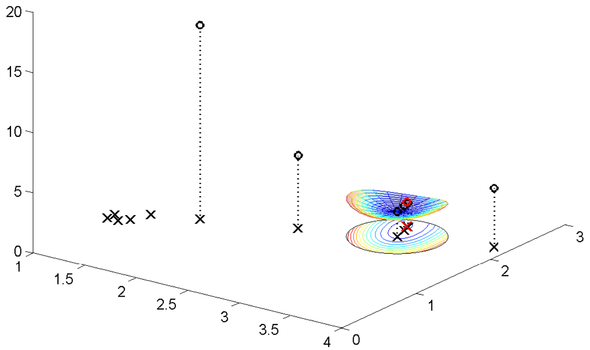


Basic trust region iteration:

- Build surrogate model m (POUNDERS: for each residual R_i)
- Trust approximation of m within region $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$
  NB- This norm (could but) is not changing
- Use m to obtain next point within $\mathcal{B}$ for evaluation

Incorporate prior knowledge through scaling, norm selection, initial Δ_0

Why Scaling? Interpolation-Based Trust-Region Methods



Basic trust region iteration:

- Build surrogate model m (POUNDERS: for each residual R_i)
- Trust approximation of m within region $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$ **NB**- This norm (could but) is not changing
- Use m to obtain next point within $\mathcal{B}$ for evaluation

Incorporate prior knowledge through scaling, norm selection, initial Δ_0

Other Deterministic Objective/Loss/Training Function Forms

Standard "χ^2": Assumes independence

$$f(x) = \frac{1}{p-n}\sum_{i=1}^{p} R_i(x)^2 = \frac{1}{p-n}\sum_{i=1}^{p}\left(\frac{S(x;\theta_i)-d_i}{\sigma_i}\right)^2$$

Correlated: For $\mathbf{W}$ symmetric positive definite:

$$f(x) = \sum_i \sum_j W_{i,j} R_i(x) R_j(x) = \|\mathbf{R}(x)\|_{\mathbf{W}}^2$$

Gaussian priors: $f(x) = \|\mathbf{R}(x)\|_{\mathbf{W}}^2 + \|x - \hat{x}\|_{\mathbf{C}}^2$

(Censored) L1 loss: (LAD)

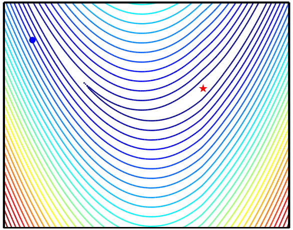
$$f(x) = \sum_i w_i \left|d_i - S_i(x)\right| \quad \text{or} \quad f(x) = \sum_i w_i \left|d_i - \max\left\{S_i(x), c_i\right\}\right|$$

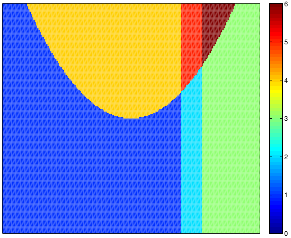
Solvers exist for many forms of objective; objective form matters!

Nonsmooth Compositions Require Additional Care

L1 Loss:

$$\sum_{i=1}^{p} |S_i(x)|$$





Nonsmooth Compositions Require Additional Care

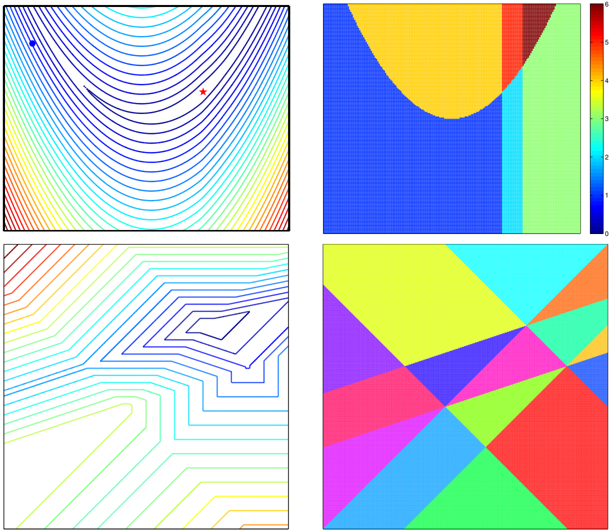
L1 Loss:

$$\sum_{i=1}^{p} |S_i(x)|$$

Censored L1 loss:

$$\sum_{i=1}^{p} |d_i - \max\{S_i(x), c_i\}|$$

NB- Can truncate some multimodality



→ Manifold sampling: [Larson, Menickelly, W.; SIOPT 2016], [Khan, Larson, W.; Preprint 2018]

Optimization Under Uncertainty

$\rightarrow u$ denotes vector of uncertain variables

Examples

- Stochastic optimization

  $$\min_x \mathbb{E}_u \left[F(x, u) \right]$$

- Robust(/"worst-case") optimization

  $$\min_x \max_{u \in \mathcal{U}} f(x, u) \qquad \text{or} \qquad \min_x \left\{ f(x) : \; |R_i(x; u)| \leq \kappa \, \forall u \in \mathcal{U}, \forall i \right\}$$

- Trimmed/quantile loss

  $$f(x) = \sum_{i=1}^{q} \left| R_{(i)}(x) \right|$$

Stochastic Optimization

General problem

$$\min \{ f(x) = \mathbb{E}_u [F(x,u)] : \; x \in X \} \tag{1}$$

- $x \in \mathbb{R}^n$ decision variables
- u vector of random variables
  - independent of x
  - $P(u)$ distribution function for u
  - u has support $\mathcal{U}$
- $F(x, \cdot)$ functional form of uncertainty for decision x
- $X \subseteq \mathbb{R}^n$ set defined by deterministic constraints

Approach of Sampling Methods for $f(x) = \mathbb{E}_u[F(x,u)]$

- Let $u^1, u^2, \cdots, u^N \sim P$
- For $x \in X$, define:

$$f_N(x) = \frac{1}{N}\sum_{i=1}^{N} F(x, u^i)$$

  - f_N is a random variable (really, a stochastic process) (depends on $(u^1, u^2, \cdots, u^N)$)
  - Motivated by $\mathbb{E}_u[f_N(x)] = f(x)$

Bias of Sampling Methods

- Let $f^* = f(x^*)$ for $x^* \in X^* \subseteq X$

Bias of Sampling Methods

- Let $f^* = f(x^*)$ for $x^* \in X^* \subseteq X$
- For any $N \geq 1$:

$$\mathbb{E}_u \left[f_N^* \right] \leq f^* = \mathbb{E}_u \left[F(x^*, u) \right]$$

because

$$\mathbb{E}_u \left[f_1^* \right] = \mathbb{E}_u \left[\min \left\{ F(x, u) : x \in X \right\} \right] \leq \min \left\{ \mathbb{E}_u \left[F(x, u) \right] : x \in X \right\} = f^*$$

Bias of Sampling Methods

- Let $f^* = f(x^*)$ for $x^* \in X^* \subseteq X$
- For any $N \geq 1$:
$$\mathbb{E}_u\left[f_N^*\right] \leq f^* = \mathbb{E}_u\left[F(x^*, u)\right]$$
because
$$\mathbb{E}_u\left[f_1^*\right] = \mathbb{E}_u\left[\min\left\{F(x,u) : x \in X\right\}\right] \leq \min\left\{\mathbb{E}_u\left[F(x,u)\right] : x \in X\right\} = f^*$$
- Sampling problems result in optimal values below f^*
- f_N^* is biased estimator of f^*

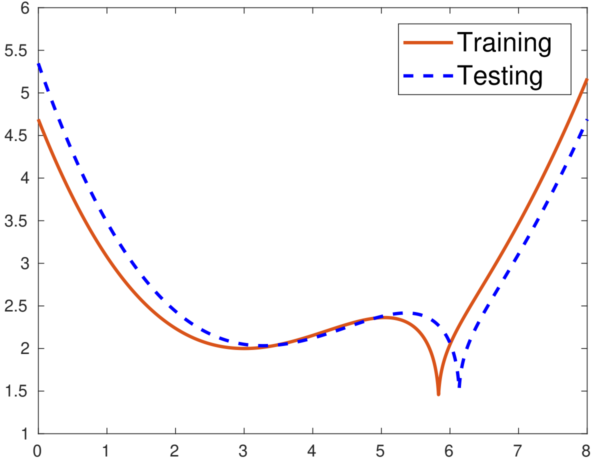
Sample Average Approximation

- Draw realizations $\hat{u}^1, \hat{u}^2, \cdots, \hat{u}^N \sim P$ of $\left(u^1, u^2, \cdots, u^N\right)$
- Replace (1) with

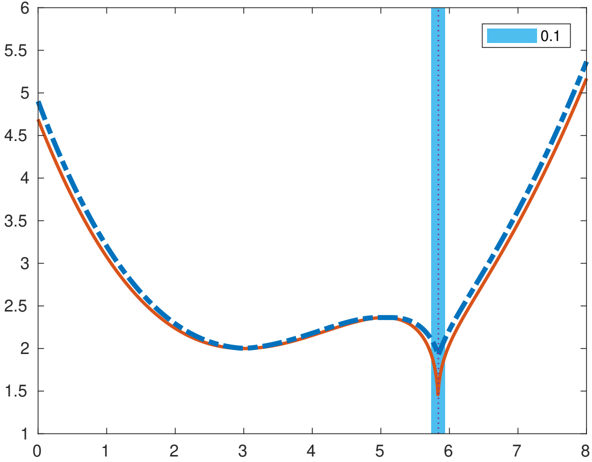
$$\min \left\{ \frac{1}{N} \sum_{i=1}^{N} F(x, \hat{u}^i) : \; x \in X \right\} \tag{2}$$

  - $\hat{f}_N(x) = \frac{1}{N} \sum_{i=1}^{N} F(x, \hat{u}^i)$ deterministic
  - Follows mean of the N sample paths defined by the (fixed) $\hat{u}^i$

Robust Optimization: Deterministic Incorporation of Robustness Desires



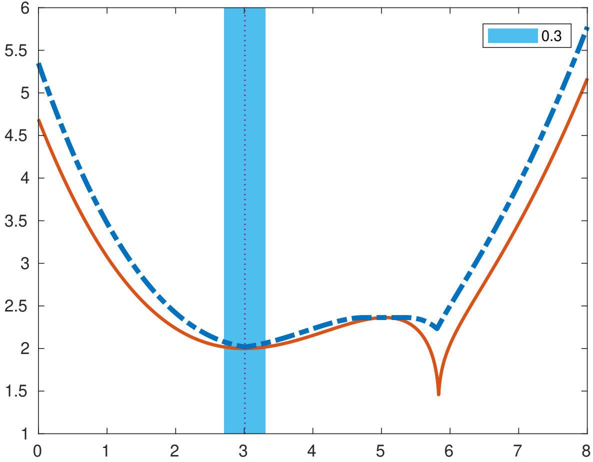
Robust Optimization: Deterministic Incorporation of Robustness Desires



$$\Psi(x) = \max_u \{f(x+u) : \|u\| \le \alpha\}$$

Game: You choose x, opponent chooses u

Robust Optimization: Deterministic Incorporation of Robustness Desires

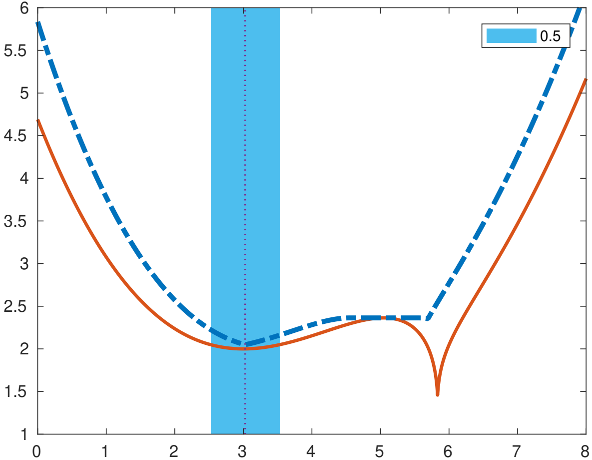


$$\Psi(x) = \max_u \{f(x+u) : \|u\| \leq \alpha\}$$

Game: You choose x, opponent chooses u

Ex.- Deep learning generalization [Keskar, Mudigere, Nocedal, Smelyanskiy, Tang; ICLR 2017]

Robust Optimization: Deterministic Incorporation of Robustness Desires

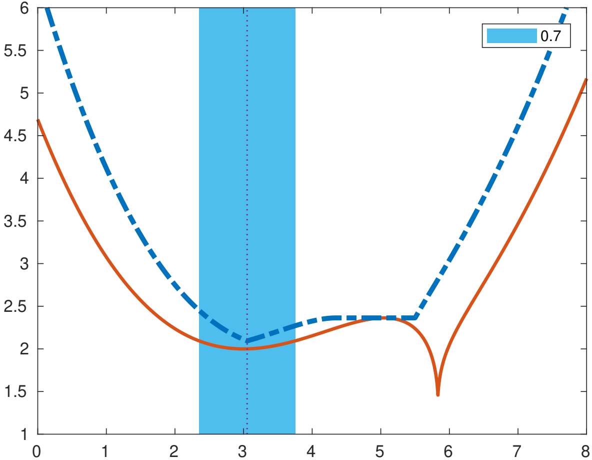


$$\Psi(x) = \max_u \{f(x+u) : \|u\| \leq \alpha\}$$

Game: You choose x, opponent chooses u

Ex.- Deep learning generalization [Keskar, Mudigere, Nocedal, Smelyanskiy, Tang; ICLR 2017]

Robust Optimization: Deterministic Incorporation of Robustness Desires

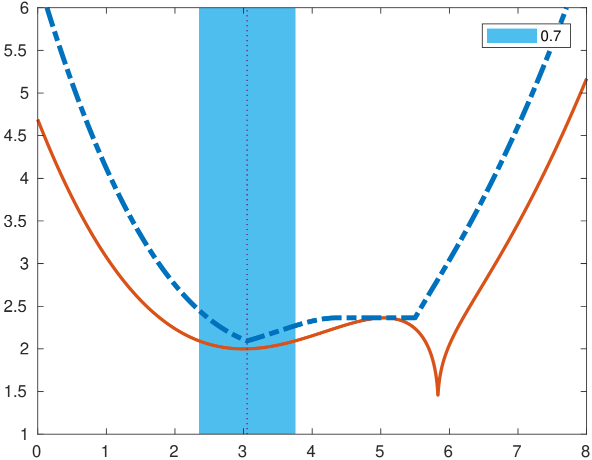


$$\Psi(x) = \max_u \{f(x+u) : \|u\| \le \alpha\}$$

Game: You choose x, opponent chooses u

Ex.- Deep learning generalization [Keskar, Mudigere, Nocedal, Smelyanskiy, Tang; ICLR 2017]

Robust Optimization: Deterministic Incorporation of Robustness Desires



$$\Psi(x) = \max_u \{f(x+u) : \|u\| \leq \alpha\}$$

Game: You choose x, opponent chooses u

Possible challenges

? Ability to compute $\Psi(x)$

$\ldots \partial\Psi(x)$

? Determination of $\alpha > 0$

$\ldots$ uncertainty set

Ex.- $\mathcal{U} = \{u : \|u\| \leq \alpha\}$

Ex.- Deep learning generalization [Keskar, Mudigere, Nocedal, Smelyanskiy, Tang; ICLR 2017]

Nonlinear Robust Optimization

Guard against worst-case uncertainty in the problem data

$$\min_{x \in \mathbb{R}^n} \left\{ f(x) \ : \ c(x, u) \leq 0 \qquad \forall u \in \mathcal{U} \right\}$$

f certain objective

$c : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ uncertain constraints

u uncertain variables/data

$\mathcal{U} \subset \mathbb{R}^m$ uncertainty set (compact, convex, $|\mathcal{U}| = \infty$)

Nonlinear Robust Optimization

Guard against worst-case uncertainty in the problem data

$$\min_{x\in\mathbb{R}^n} \left\{ f(x) \;:\; c(x,u) \le 0 \qquad \forall u \in \mathcal{U} \right\}$$

f certain objective

$c : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ uncertain constraints

u uncertain variables/data

$\mathcal{U} \subset \mathbb{R}^m$ uncertainty set (compact, convex, $|\mathcal{U}| = \infty$)

Special cases:

Minimax

$$\min_{x\in\mathbb{R}^n} \max_{u\in\mathcal{U}} f(x,u)$$

Implementation errors

$$\min_{x\in\mathbb{R}^n} \max_{u\in\mathcal{U}} f(x+u)$$

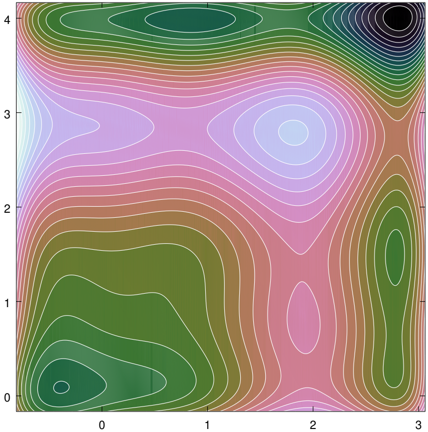
Bounded errors

$$\min_{x} \{ f(x) : \; |R_i(x;u)| \le \kappa \, \forall u \in \mathcal{U}, \forall i \}$$

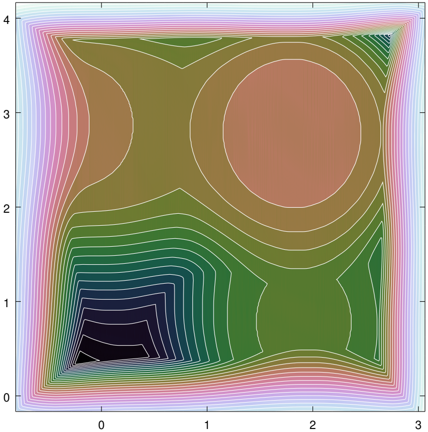
2D Example: Bertsimas-Nohadani-Teo Implementation Error Problem

$$\Psi_{\mathcal{U}_\alpha}(x) := \max_{u:\|u\|_2 \le \alpha} f(x,u) := \max_{u:\|u\|_2 \le \alpha} g(x+u)$$

parameter $\alpha \ge 0$ ($\alpha = 0.5$ typical)



$g(x) = f(x, 0)$



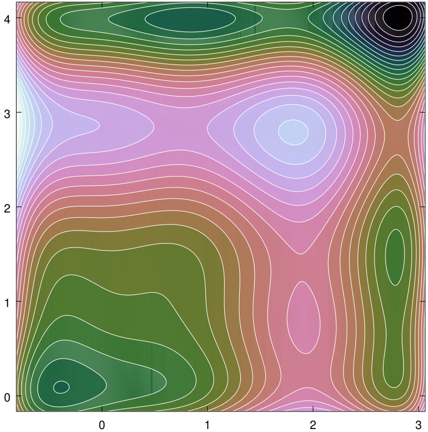
$\Psi_{\mathcal{U}_{0.5}}(x)$

$$\begin{aligned} g(x) \quad = \quad & 2x_1^6 - 12.2x_1^5 + 21.2x_1^4 - 6.4x_1^3 - 4.7x_1^2 + 6.2x_1 + x_2^6 - 11x_2^5 + 43.3x_2^4 \\ & -74.8x_2^3 + 56.9x_2^2 - 10x_2 - 0.1x_1^2 + x_2^2 + 0.4x_1^2x_2 + 0.4x_2^2x_1 - 4.1x_1x_2 \end{aligned}$$

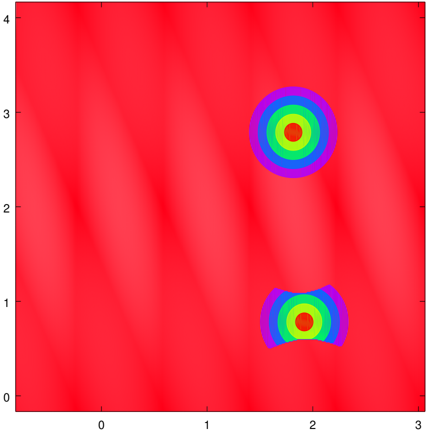
2D Example: Bertsimas-Nohadani-Teo Implementation Error Problem

$$\Psi_{\mathcal{U}_\alpha}(x) := \max_{u:\|u\|_2\le\alpha} f(x,u) := \max_{u:\|u\|_2\le\alpha} g(x+u)$$

parameter $\alpha \ge 0$ ($\alpha = 0.5$ typical)



$g(x) = f(x,0)$



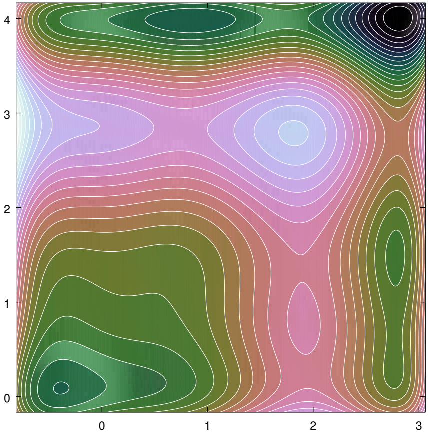
$\|u^*\|$

$$\begin{aligned} g(x) &= 2x_1^6 - 12.2x_1^5 + 21.2x_1^4 - 6.4x_1^3 - 4.7x_1^2 + 6.2x_1 + x_2^6 - 11x_2^5 + 43.3x_2^4 \\ &\quad -74.8x_2^3 + 56.9x_2^2 - 10x_2 - 0.1x_1^2x_2^2 + 0.4x_1^2x_2 + 0.4x_2^2x_1 - 4.1x_1x_2 \end{aligned}$$

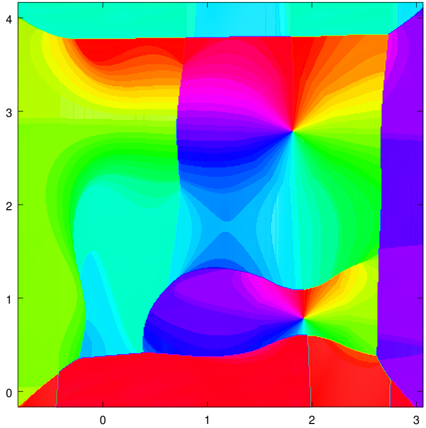
2D Example: Bertsimas-Nohadani-Teo Implementation Error Problem

$$\Psi_{\mathcal{U}_\alpha}(x) := \max_{u:\|u\|_2 \le \alpha} f(x,u) := \max_{u:\|u\|_2 \le \alpha} g(x+u)$$

parameter $\alpha \geq 0$ ($\alpha = 0.5$ typical)



$g(x) = f(x,0)$



$\angle u^*$

$$g(x) = 2x_1^6 - 12.2x_1^5 + 21.2x_1^4 - 6.4x_1^3 - 4.7x_1^2 + 6.2x_1 + x_2^6 - 11x_2^5 + 43.3x_2^4$$
$$-74.8x_2^3 + 56.9x_2^2 - 10x_2 - 0.1x_1^2 + x_2^2 + 0.4x_1^2x_2 + 0.4x_2^2x_1 - 4.1x_1x_2$$

Trimmed Optimization Motivation: Supervised Learning

Obtain model prediction $F(\cdot, x)$ by solving

$$\min_x \sum_{i=1}^{N} l\left(F(\theta^i, x), y^i\right)$$

- $\mathbb{T} = \{(\theta^i, y^i)\}_{i=1}^{N} \subset \mathbb{R}^d \times \mathbb{R}$ — Training data
- $y^i \in \mathbb{R}$ — label associated with input θ^i
- $x \in \mathbb{R}^n$ — weights
- $F : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ — trained model
- $l : \mathbb{R}^2 \to \mathbb{R}$ — loss function

e.g., $l(a, b) = (a - b)^2$

Trimmed Optimization

Perform training robust to outliers in a training dataset

$$\min_x f(x) \triangleq \min_w \frac{1}{q} \sum_{j=1}^{q} l_{(j)} \left(F(\theta^{(j)}, x), y^{(j)} \right),$$

where (j) denotes the index associated with the jth-order statistic, i.e.,

$$l_{(j-1)} \left(F(\theta^{(j-1)}, x), y^{(j-1)} \right) \leq l_{(j)} \left(F(\theta^{(j)}, x), y^{(j)} \right) \text{ for } j = 2, \ldots, N.$$

- Removes outliers (defined by the quantile q) *dynamically* (i.e., based on x)
- objective f is nonsmooth but in a special way

Summary

General

- Move beyond “blackbox” optimization
- Exploiting structure yields better solutions, in fewer simulations
- Promote optimization/modeling considerations during code development
- Optimization problem formulation matters
- Optimization can play a role in function-evaluation-limited, goal-oriented studies
- Expanded opportunity for scalable parallelism through optimization, sensitivity analysis, UQ

`www.mcs.anl.gov/~wild` (Get in touch!)