Numerical Optimization for Physicists and Statisticians

Stefan Wild

Mathematics and Computer Science Division
Argonne National Laboratory

Grateful to 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**

November 7, 2017

Mathematical/Numerical Optimization for ISNET

Possible Topics Today

- Optimization Basics
- Optimization for Expensive Model Calibration

  fast, – limiting the number of expensive simulation evaluations

  local, – given enough resources, find you a point for which you cannot improve the objective in a local neighborhood

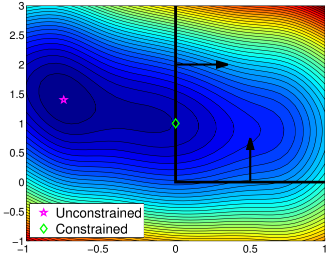
  derivative-free – useful in situations where derivatives unavailable
- Beyond χ^2 Minimization
- Stochastic Optimization
- Bayesian Optimization

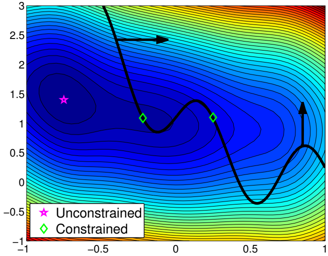
Optimization is the *"science of better"*

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

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

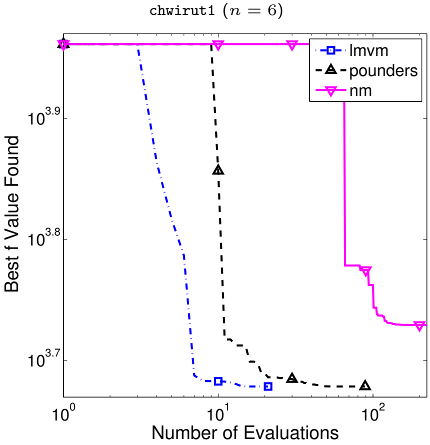
- (Unless Ω is very special) Need to evaluate f at many x to find a good $\hat{x}_*$
- Focus on local solutions: $f(\hat{x}_*) \leq f(x) \ \forall x \in \mathcal{N}(\hat{x}_*) \cap \Omega$





Implicitly assume that uncertainty modeled through constraints and objective(s)

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

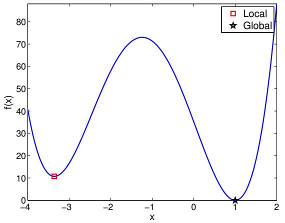
lmvm Uses available $\nabla_x f$

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

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

Careful:

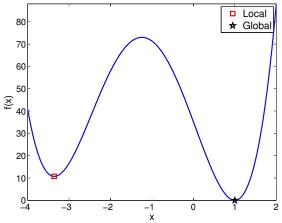
- **Global convergence**: Convergence (to a local solution/stationary point) from anywhere in Ω
- **Convergence to a global minimizer**: Obtain x_* with $f(x_*) \leq f(x)\, \forall x \in \Omega$



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

Careful:

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



Anyone selling you global solutions when derivatives are unavailable:

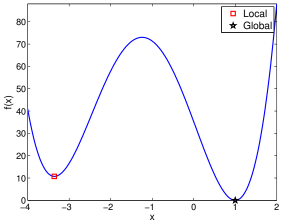
either assumes more about your problem (e.g., convex f)

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.

or cannot be trusted

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



Careful:

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

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.

or cannot be trusted

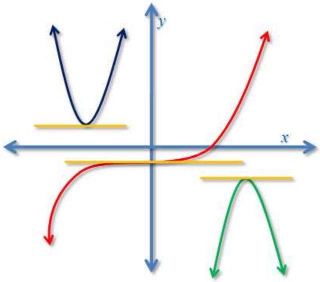
Instead:

- Rapidly find good local solutions and/or be robust to poor solutions
- Find several good local solutions concurrently (`APOSMM`/`LibEnsemble`)

Optimization Tightly Coupled With Derivatives (WRT Parameters)

Typical optimality (no noise, smooth functions)

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



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

- Faster feasibility
- Faster convergence
  - Guaranteed descent
  - Approximation of nonlinearities
- Better termination
  - Measure of criticality $\|\nabla_x f\|$ or $\|\mathcal{P}_\Omega(\nabla_x f)\|$

But derivatives $\nabla_x S(x)$ are not always available/do not always exist

Obtain Derivatives $\nabla_x S$ Whenever Possible

Handcoding (HC)

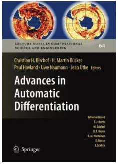
"Army of students/programmers"

- ? Prone to errors/conditioning
- ? Intractable as number of ops increases

Algorithmic/Automatic Differentiation (AD)

"Exact* derivatives!"

- ? No black boxes allowed
- ? Not always automatic/cheap/well-conditioned



Finite Differences (FD)

"Nonintrusive"

- ? Expense grows with n
- ? Sensitive to stepsize choice/noise

→[Moré & W.; SISC 2011], [Moré & W.; TOMS 2012]

...then apply derivative-based method (that handles inexact derivatives)

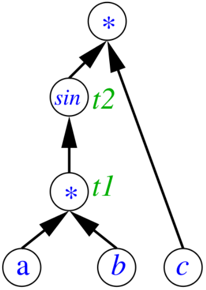
Algorithmic Differentiation

→ [Coleman & Xu; SIAM 2016], [Griewank & Walther; SIAM 2008]

Computational Graph

- $y = sin(a * b) * c$
- Forward and reverse modes
- AD tool provides code for your derivatives

Write codes and formulate problems with AD in mind!



Many tools (see `www.autodiff.org`):

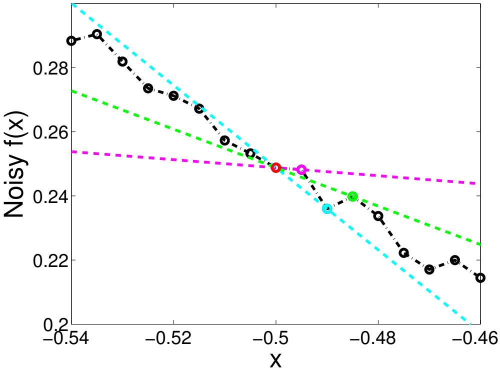
F	`OpenAD`
F/C	`Tapenade`, `Rapsodia`
C/C++	`ADOL-C`, `ADIC`
Matlab	`ADiMat`, `INTLAB`
Python/R	`ADOL-C`

Also done in `AMPL`, `GAMS`, `JULIA`!

Numerical Differentiation

The Problem: Finite differences sensitive to choice of h

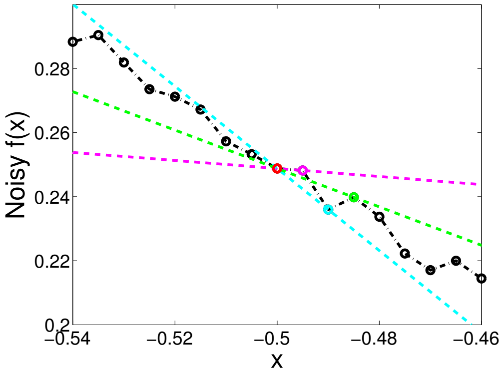
$$\frac{f(t_0+h)-f(t_0)}{h} \approx f_s'(t_0)$$



Numerical Differentiation

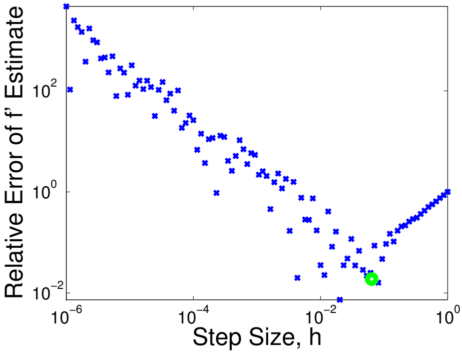
The Problem: Finite differences sensitive to choice of h

$$\frac{f(t_0 + h) - f(t_0)}{h} \approx f_s'(t_0)$$



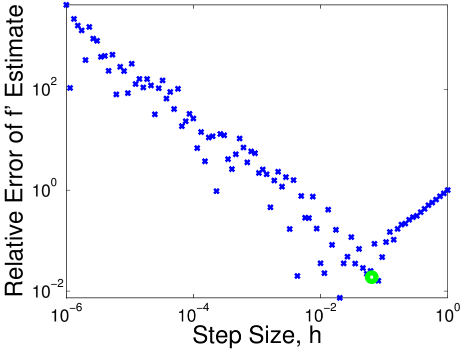
Minimize $$\mathrm{E}\left\{\mathcal{E}(h)\right\} = \mathrm{E}\left\{\left(\frac{f(t_0+h)-f(t_0)}{h} - f_s'(t_0)\right)^2\right\}$$

Optimal Forward Difference Parameter h



Optimal Forward Difference Parameter h

$$\frac{1}{4}\mu_L^2 h^2 + 2\frac{\varepsilon_f^2}{h^2} \le \mathrm{E}\left\{\mathcal{E}(h)\right\} \le \frac{1}{4}\mu_M^2 h^2 + 2\frac{\varepsilon_f^2}{h^2}$$



$h \downarrow$ Variance (noise) dominates

$h \uparrow$ Bias (f'') dominates

1. Upper bound minimized by
   $h_M = 8^{1/4}\left(\frac{\varepsilon_f}{\mu_M}\right)^{1/2}$
   - ε_f^2 =Var $f(t_0)$
   - $\mu_M \ge |f''|$
2. When $\mu_L > 0$, h_M is near-optimal:

$$\mathrm{E}\left\{\mathcal{E}(h_M)\right\} = \sqrt{2}\mu_M\varepsilon_f \le \left(\frac{\mu_M}{\mu_L}\right) \min_{0\le h\le h_0} \mathrm{E}\left\{\mathcal{E}(h)\right\}.$$

[Estimating Noisy Derivatives. Moré & W., TOMS 2012]]

Simulation-Based Optimization

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

Optimize expensive, nonlinear functions arising in science & engineering

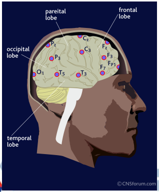
"parameter estimation", "model calibration", "design optimization", ...

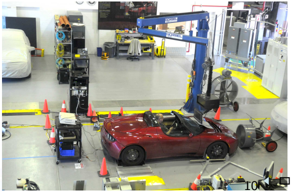
- $f : \mathbb{R}^n \to \mathbb{R}$ objective, $S : \mathbb{R}^n \to \mathbb{R}^p$ numerical simulation, Ω constraints
- Evaluating S means running a simulation modeling some (smooth) process

  Ex- $S =$ solving PDEs via finite elements
  - Here: assume f is from a deterministic computer simulation
- S can contribute to objective and/or constraints, possibly noisy
- Derivatives $\nabla_x S$ often unavailable or prohibitively expensive to obtain/approximate directly
- S (could/must be parallelized) takes secs/mins/hrs/days for 1 x

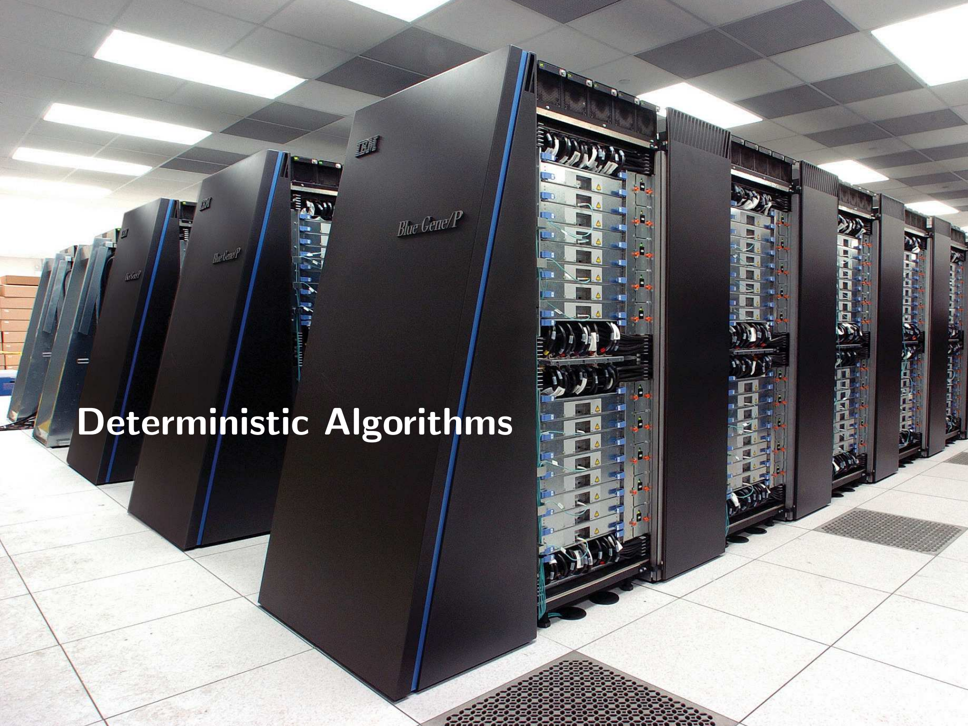
  Evaluation is a bottleneck for optimization

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

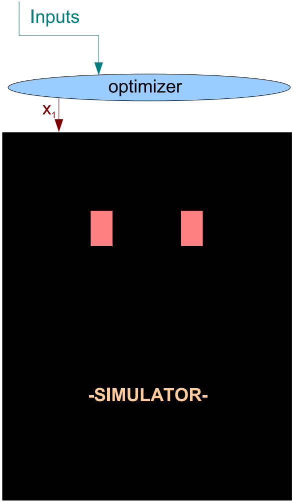


Deterministic Algorithms



"Simplest" (=Most Naive) Formulation: Blackbox f



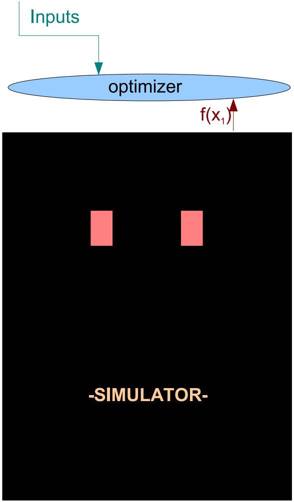
Optimizer gives x, physicist provides $f(x)$

- f can be a blackbox (executable only or proprietary/legacy codes)
- Only give a single output
  - no derivatives with respect to x: $\nabla_x S(x), \nabla^2_{x,x} S(x)$
  - no problem structure

Good solutions guaranteed in the limit, but:

- Computational budget limits number of evaluations

“Simplest” (=Most Naive) Formulation: Blackbox f



Optimizer gives x, physicist provides $f(x)$

- f can be a blackbox (executable only or proprietary/legacy codes)
- Only give a single output
  - no derivatives with respect to x: $\nabla_x S(x), \nabla^2_{x,x} S(x)$
  - no problem structure

Good solutions guaranteed in the limit, but:

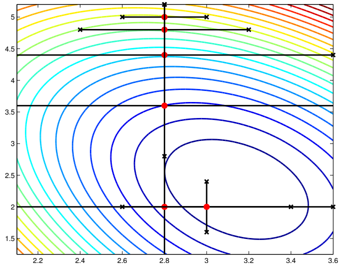
- <u>Computational budget</u> limits number of evaluations

Two main styles of local algorithms

- Direct search methods (pattern search, Nelder-Mead, . . .)
- Model- (“surrogate-”)based methods (quadratics, radial basis functions, . . .)

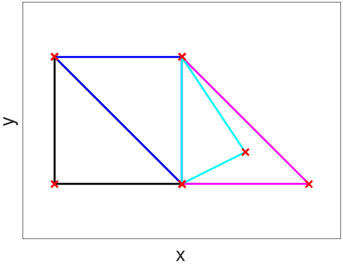
Algorithms: Direct Search Methods

Pattern Search



Easy to parallelize f evaluations

Nelder-Mead



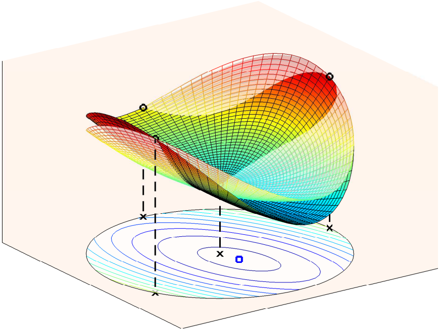
Popularized by *Numerical Recipes*

- Rely on indicator functions: $[f(x_k + \mathbf{s}) <^? f(x_k)]$
- Work with **black-box** $f(x)$, do not exploit structure $F[x, S(x)]$

$\longrightarrow$[Kolda, Lewis, Torczon, SIREV 2003]

Trust-Region Methods Use Models Instead of f

To reduce the number of expensive f evaluations
$\rightarrow$ Replace difficult optimization problem $\min f(x)$ with a much simpler one $\min\{m(x) : x \in \mathcal{B}\}$

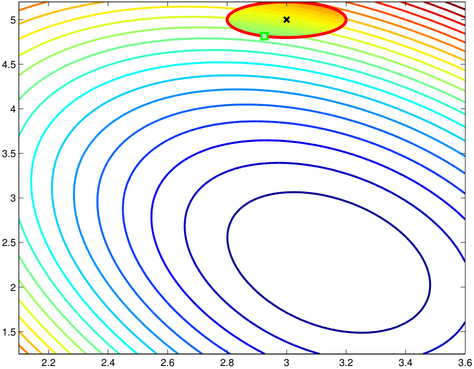


Classic NLP Technique:

- f Original function: computationally expensive, no derivatives
- m Surrogate model: computationally attractive, analytic derivatives

Basic Trust-Region Idea

Use a surrogate $m(x)$ in place of the unwieldy $f(x)$

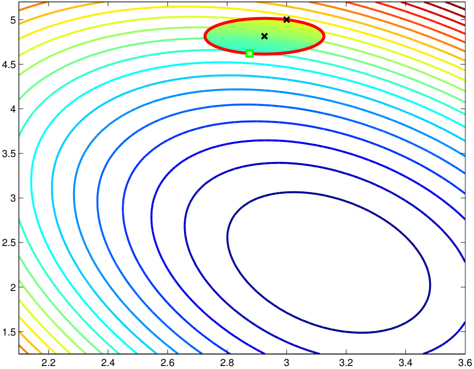


Optimize over m to avoid expense of f

- Trust m to approximate f within $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$,
- Obtain next point from $\min\{m(x) : x \in \mathcal{B}\}$,
- Evaluate function and update (x_k, Δ_k) based on how good the model's prediction was.

Basic Trust-Region Idea

Use a surrogate $m(x)$ in place of the unwieldy $f(x)$

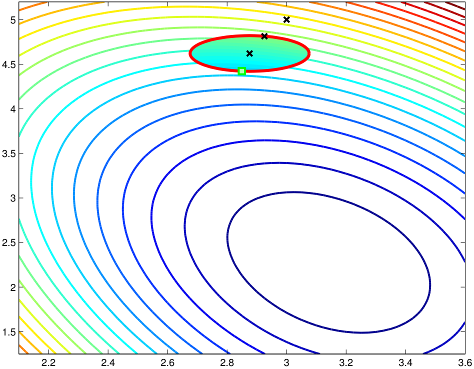


Optimize over m to avoid expense of f

- Trust m to approximate f within $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$,
- Obtain next point from $\min\{m(x) : x \in \mathcal{B}\}$,
- Evaluate function and update (x_k, Δ_k) based on how good the model's prediction was.

Basic Trust-Region Idea

Use a surrogate $m(x)$ in place of the unwieldy $f(x)$

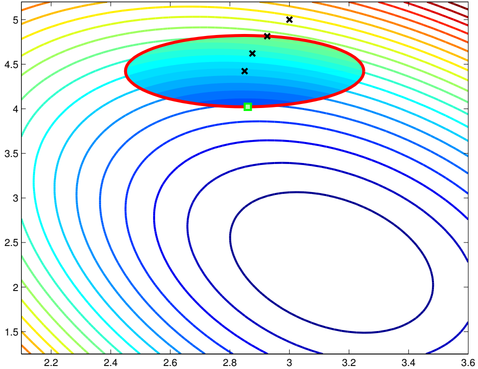


Optimize over m to avoid expense of f

- Trust m to approximate f within $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$,
- Obtain next point from $\min \{m(x) : x \in \mathcal{B}\}$,
- Evaluate function and update (x_k, Δ_k) based on how good the model's prediction was.

Basic Trust-Region Idea

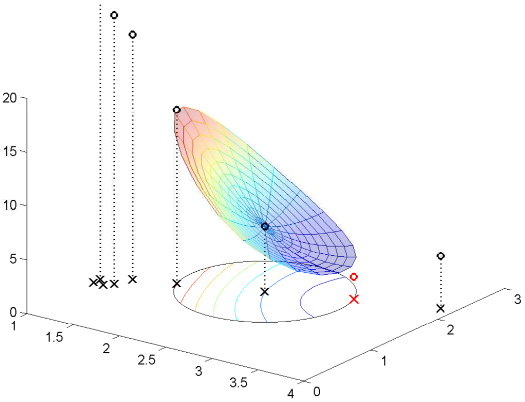
Use a surrogate $m(x)$ in place of the unwieldy $f(x)$



Optimize over m to avoid expense of f

- Trust m to approximate f within $\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$,
- Obtain next point from $\min\{m(x) : x \in \mathcal{B}\}$,
- Evaluate function and update (x_k, Δ_k) based on how good the model's prediction was.

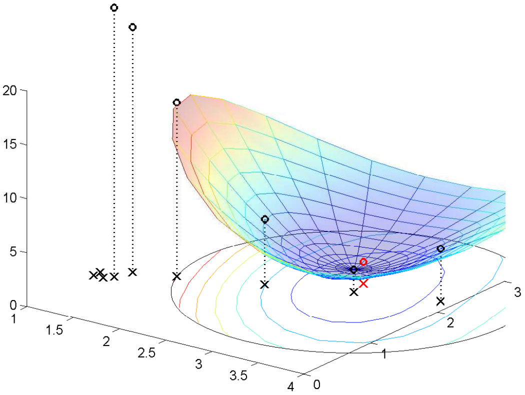
Interpolation-Based Trust-Region Methods



Iteration k:

- Build a model m_k interpolating f on $\mathcal{Y}_k$
- Trust m_k within region $\mathcal{B}_k$
- Minimize m_k within $\mathcal{B}_k$ to obtain next point for evaluation
- Do expensive evaluation
- Update m_k and $\mathcal{B}_k$ based on how good model prediction was

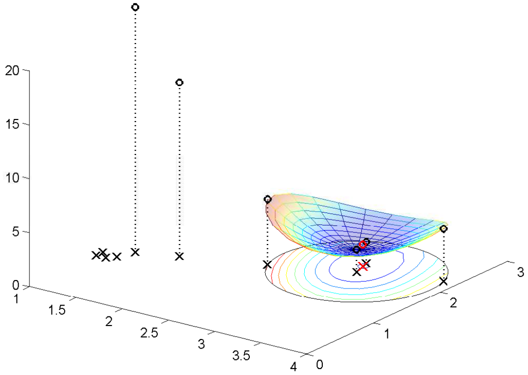
Interpolation-Based Trust-Region Methods



Iteration k:

- Build a model m_k interpolating f on $\mathcal{Y}_k$
- Trust m_k within region $\mathcal{B}_k$
- Minimize m_k within $\mathcal{B}_k$ to obtain next point for evaluation
- Do expensive evaluation
- Update m_k and $\mathcal{B}_k$ based on how good model prediction was

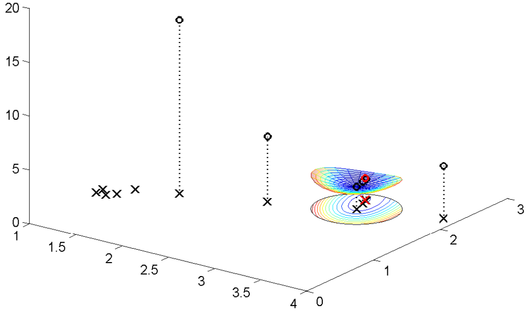
Interpolation-Based Trust-Region Methods



Iteration k:

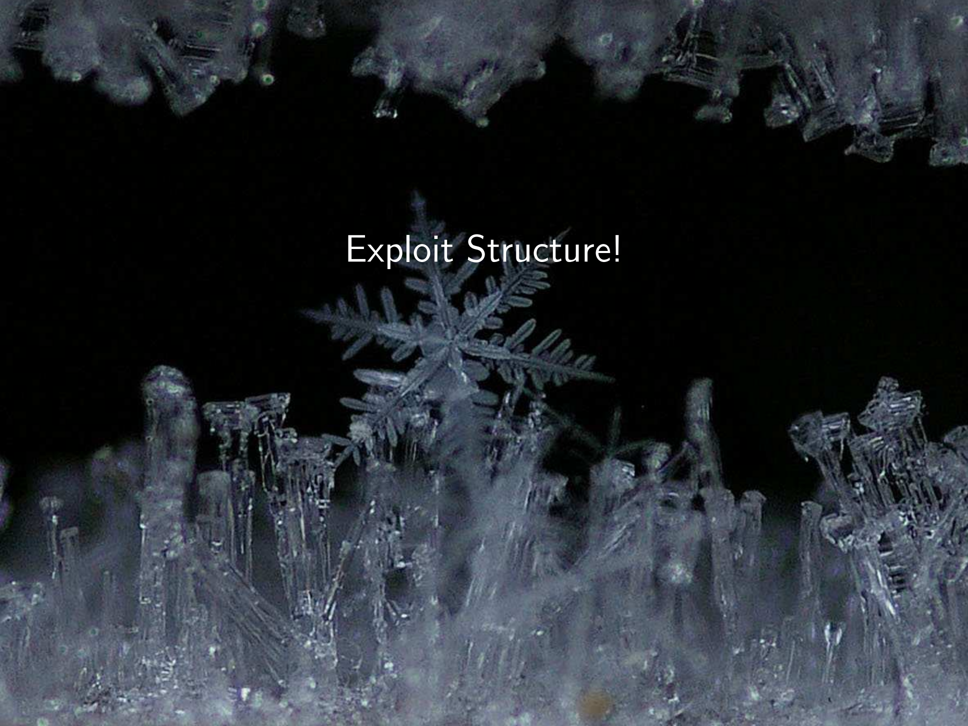
- Build a model m_k interpolating f on $\mathcal{Y}_k$
- Trust m_k within region $\mathcal{B}_k$
- Minimize m_k within $\mathcal{B}_k$ to obtain next point for evaluation
- Do expensive evaluation
- Update m_k and $\mathcal{B}_k$ based on how good model prediction was

Interpolation-Based Trust-Region Methods

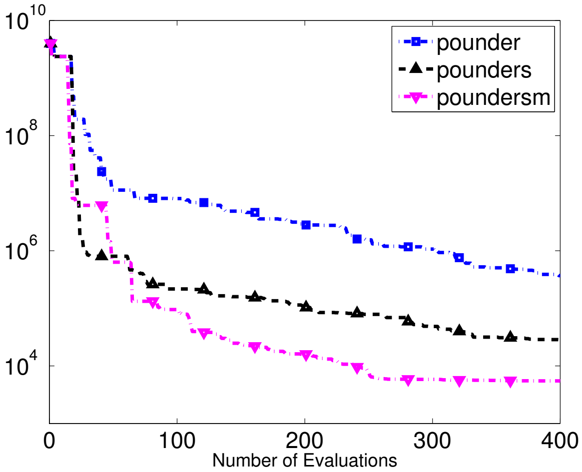


Iteration k:

- Build a model m_k interpolating f on $\mathcal{Y}_k$
- Trust m_k within region $\mathcal{B}_k$
- Minimize m_k within $\mathcal{B}_k$ to obtain next point for evaluation
- Do expensive evaluation
- Update m_k and $\mathcal{B}_k$ based on how good model prediction was


Exploit Structure!

Performance of Model-Based Methods



Optimizing EDF in [Bertolli et al., PRC 2012]

Parameter Estimation is NOT a Blackbox 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 Blackbox 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]

Parameter Estimation is NOT a Blackbox 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 $\mathrm{dom}(S)$

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

◇ Taking advantage of structure should further reduce # of expensive evaluations

Exploiting Nonlinear Least Squares Structure

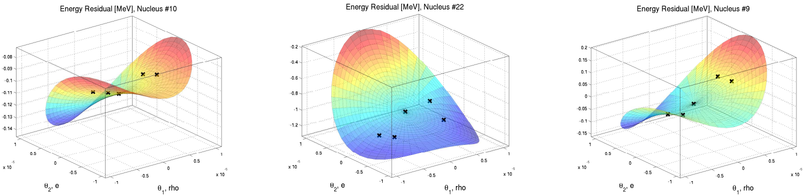
Obtain a *vector* of output $R_1(x), \ldots, R_p(x)$

- (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

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



General Nonlinear Least Squares

$$\min_{x} f(x) = \|\mathbf{R}(x)\|_{\mathbf{W}}^2$$

$\mathbf{R}$: $\mathbb{R}^n \to \mathbb{R}^p$ "residual vector"

$\to$ Think: $R_i(x) = S(x; \theta_i) - d_i$

$\mathbf{W}$ norm: $\|\mathbf{y}\|_{\mathbf{W}} = \left(\mathbf{y}^T \mathbf{W} \mathbf{y}\right)^{1/2}$

$\to$ $\mathbf{W} = I_p$ recovers $\|\cdot\|_2$

General Nonlinear Least Squares

$$\min_x f(x) = \|\mathbf{R}(x)\|_{\mathbf{W}}^2$$

$\mathbf{R}$: $\mathbb{R}^n \to \mathbb{R}^p$ "residual vector"

$\to$ Think: $R_i(x) = S(x;\theta_i) - d_i$

$\mathbf{W}$ norm: $\|\mathbf{y}\|_{\mathbf{W}} = \left(\mathbf{y}^T\mathbf{W}\mathbf{y}\right)^{1/2}$

$\to$ $\mathbf{W} = I_p$ recovers $\|\cdot\|_2$

$\mathbf{W}$ symmetric positive definite

- $\mathbf{W} = \mathbf{W}^T$
- $\mathbf{y}^T\mathbf{W}\mathbf{y} > 0$ for all $\mathbf{y} \neq \mathbf{0}$

$$f(x) = \sum_{i=1}^{p}\sum_{j=1}^{p} W_{i,j} R_i(x) R_j(x) \geq 0$$

General Nonlinear Least Squares

$$\min_x f(x) = \|\mathbf{R}(x)\|_{\mathbf{W}}^2$$

$\mathbf{R}$: $\mathbb{R}^n \to \mathbb{R}^p$ "residual vector"

$\to$ Think: $R_i(x) = S(x;\theta_i) - d_i$

$\mathbf{W}$ norm: $\|\mathbf{y}\|_{\mathbf{W}} = \left(\mathbf{y}^T\mathbf{W}\mathbf{y}\right)^{1/2}$

$\to$ $\mathbf{W} = I_p$ recovers $\|\cdot\|_2$

$\mathbf{W}$ symmetric positive definite

- $\mathbf{W} = \mathbf{W}^T$
- $\mathbf{y}^T\mathbf{W}\mathbf{y} > 0$ for all $\mathbf{y} \neq \mathbf{0}$

$$f(x) = \sum_{i=1}^{p}\sum_{j=1}^{p} W_{i,j} R_i(x) R_j(x) \geq 0$$

- $\mathbf{W} = (\text{diag}(\sigma))^{-1}$ yields familiar

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

A Warning

Can I pass this to my favorite $\min_x \chi^2(x) = \|\tilde{\mathbf{R}}(x)\|^2$ solver?

A Warning

Can I pass this to my favorite $\min_x \chi^2(x) = \|\tilde{\mathbf{R}}(x)\|^2$ solver?

$$\begin{array}{rl} & \sum_{i=1}^p \sum_{j=1}^p \left(\tilde{R}_{i,j}(x)\right)^2 \\ = & \sum_{i=1}^p \sum_{j=1}^p \left(\sqrt{|W_{i,j} R_i(x) R_j(x)|}\right)^2 \\ \neq & \sum_{i=1}^p \sum_{j=1}^p W_{i,j} R_i(x) R_j(x) \end{array}$$

A Warning

Can I pass this to my favorite $\min_x \chi^2(x) = \|\tilde{\mathbf{R}}(x)\|^2$ solver?

$$
\begin{aligned}
& \sum_{i=1}^p \sum_{j=1}^p \left(\tilde{R}_{i,j}(x)\right)^2 \\
= \quad & \sum_{i=1}^p \sum_{j=1}^p \left(\sqrt{|W_{i,j} R_i(x) R_j(x)|}\right)^2 \\
\neq \quad & \sum_{i=1}^p \sum_{j=1}^p W_{i,j} R_i(x) R_j(x)
\end{aligned}
$$

! Allow for complex-valued residuals

! Disallow $W_{i,j} R_i(x) R_j(x) < 0$

In any case, you will likely suffer algorithmically

Relationship to Covariance Matrices

Data $\{(\theta_1, d_1), \cdots, (\theta_p, d_p)\}$

- Errors independent and normally distributed: $d \sim N(\mu, \Sigma)$,

$$d_i = \mu(\theta_i; x_*) + \varepsilon_i, \qquad \varepsilon_i \sim N(0, \sigma_i^2) \qquad i = 1, \ldots, p.$$

  Σ is a $p \times p$ diagonal matrix, with ith diagonal entry σ_i^2

- Model, $S(\theta; x)$ with Gaussian errors:

$$[S(\theta_1; x), \cdots, S(\theta_p; x)]^T \sim N\left(\mu(\cdot; x), C\right),$$

- C a ($p \times p$ symmetric positive definite) covariance matrix accounting for correlation between model outputs (i.e., $\mathrm{Cov}(S(\theta_i; x), S(\theta_j; x)) = C_{i,j}$)
- Assuming model errors are independent of data errors,

$$[m(\hat{x}; \theta_1) - d_1, \cdots, m(\hat{x}; \theta_p) - d_p]^T \sim N(0, C + \Sigma),$$

- Joint likelihood $l(x; \theta; d) \propto \exp\left[-\frac{1}{2}\mathbf{R}(x; \theta)^T \left(\mathbf{C} + \mathbf{\Sigma}\right)^{-1} \mathbf{R}(x; \theta)\right]$

Relationship to Covariance Matrices

Data $\{(\theta_1, d_1), \cdots, (\theta_p, d_p)\}$

- Errors independent and normally distributed: $d \sim N(\mu, \Sigma)$,

$$d_i = \mu(\theta_i; x_*) + \varepsilon_i, \qquad \varepsilon_i \sim N(0, \sigma_i^2) \qquad i = 1, \ldots, p.$$

  Σ is a $p \times p$ diagonal matrix, with ith diagonal entry σ_i^2
- Model, $S(\theta; x)$ with Gaussian errors:

$$[S(\theta_1; x), \cdots, S(\theta_p; x)]^T \sim N(\mu(\cdot; x), C),$$

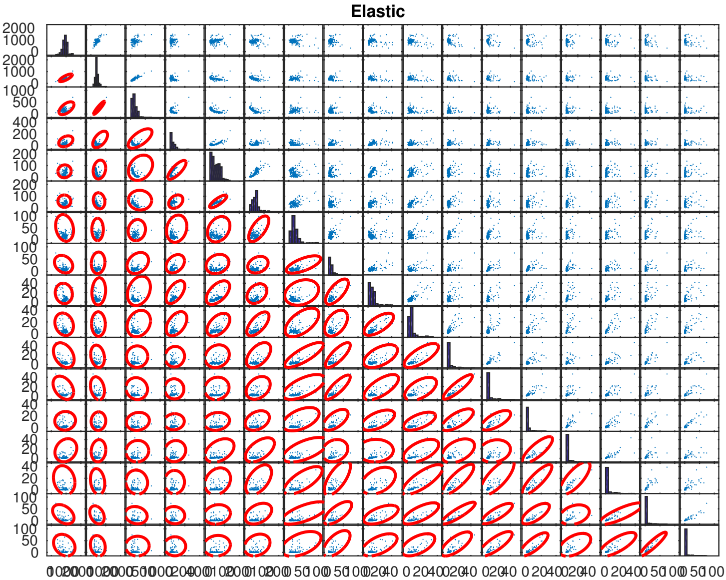
- C a ($p \times p$ symmetric positive definite) covariance matrix accounting for correlation between model outputs (i.e., $\mathrm{Cov}(S(\theta_i; x), S(\theta_j; x)) = C_{i,j}$)
- Assuming model errors are independent of data errors,

$$[m(\hat{x}; \theta_1) - d_1, \cdots, m(\hat{x}; \theta_p) - d_p]^T \sim N(0, C + \Sigma),$$

- Joint likelihood $l(x; \theta; d) \propto \exp\left[-\frac{1}{2}\mathbf{R}(x; \theta)^T (\mathbf{C} + \mathbf{\Sigma})^{-1} \mathbf{R}(x; \theta)\right]$

Warning: $\mathbf{C}, \mathbf{\Sigma}$ can no longer hide behind constants of proportionality

Optical Potentials: Incorporating Covariances in W



→ Monday talk of Lovell

Applications Using the Jacobian $[\hat{J}]_{i,j} = \frac{\partial R_i(\hat{x})}{\hat{\partial} x_j} = \frac{1}{w_i}\frac{\partial S(x;\theta_i)}{\partial x_j}$

Residual $\mathbf{R}(x) \in \mathbb{R}^p$ undergoes a change by $\epsilon \in \mathbb{R}^p$

- Ex.- normalized datum $\frac{d_i}{w_i}$ is changed to $\frac{d_i}{w_i} + \epsilon_i$

$$\hat{\mathbf{x}} \in \arg\min_{\hat{\mathbf{x}}\in\mathbb{R}^n} f^0(x) = \|\mathbf{R}(x)\|_2^2 \qquad \hat{\mathbf{x}}_\epsilon \in \arg\min_{\hat{\mathbf{x}}\in\mathbb{R}^n} f(x) = \|\mathbf{R}(x) + \epsilon\|_2^2$$

A second-order expansion of $f = \|\mathbf{R}(x) + \epsilon\|_2^2$ about $\hat{\mathbf{x}}$:

$$f(\hat{\mathbf{x}}) + 2\epsilon^T \hat{J}(x - \hat{\mathbf{x}}) + \frac{1}{2}(x - \hat{\mathbf{x}})^T \left(\nabla^2 f^0(\hat{\mathbf{x}}) + 2\sum_{i=1}^{p} \epsilon_i \nabla^2 R_i(\hat{\mathbf{x}})\right)(x - \hat{\mathbf{x}}),$$

When ϵ is small, this quadratic will be convex and hence minimized at

$$x_\epsilon - \hat{\mathbf{x}} = 2\left(\nabla^2 f^0(\hat{\mathbf{x}})\right)^{-1} \hat{J}^T \epsilon + \mathcal{O}(\|\epsilon\|^2).$$

When $\mathbf{R}(\hat{\mathbf{x}})$ is small, $\nabla^2 f^0(\hat{\mathbf{x}}) \approx 2\hat{J}^T\hat{J}$ and

$$\tilde{x}_\epsilon \approx \hat{\mathbf{x}} + \left(\hat{J}^T\hat{J}\right)^{-1} \hat{J}^T \epsilon$$

Stochastic Optimization

Stochastic Optimization

General problem

$$\min\left\{f(x) = \mathbb{E}_\xi\left[F(x,\xi)\right] :\ x \in X\right\} \tag{1}$$

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

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

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

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

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

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 $N \geq 1$:

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

because

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

Bias of Sampling Methods

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

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

  because

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

- Sampling problems result in optimal values below f^*
- f_N^* is biased estimator of f^*

Sample Average Approximation

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

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

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

Convergence with N

- A sufficient condition:
  - For any $\epsilon > 0$ there exists N_ϵ so that
    $$\left| \hat{f}_N(x) - f(x) \right| < \epsilon \qquad \forall\, N \geq N_\epsilon \quad \forall\, x \in X$$
    with probability 1 (*wp1*).
- Then $\hat{f}_N^* \to f^*$ *wp1*.
- (With additional assumptions on f and $X^* \subset X$):
  $$\text{dist}(x_N^*, X^*) \to 0$$
- (+ uniqueness, $X^* = x^*$):
  $$x_N^* \to x^*$$

Stochastic Approximation Method

Basically just:

Input x^0

1. $x^{k+1} \leftarrow \mathcal{P}_X \left\{x^k - \alpha_k s^k\right\}, \qquad k = 0, 1, \ldots$

- α_k a step size
- s^k a random direction

Stochastic Approximation Method

Basically just:

Input x^0

1. $x^{k+1} \leftarrow \mathcal{P}_X \left\{ x^k - \alpha_k s^k \right\}, \qquad k = 0, 1, \ldots$

- α_k a step size
- s^k a random direction

Generally assume:

α_k: $\sum_{k=0}^{\infty} \alpha_k = \infty$, $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$ (e.g., $\alpha_k = \frac{c}{k}$)

s^k: $\mathrm{E}\left\{ \nabla f(x^k)^T s^k \right\} > 0$

s^k is an ascent direction (in expectation) at x^k

Stochastic Approximation Method

Basically just:

Input x^0

1. $x^{k+1} \leftarrow \mathcal{P}_X\left\{x^k - \alpha_k s^k\right\}, \qquad k = 0, 1, \ldots$

- α_k a step size
- s^k a random direction

Generally assume:

α_k: $\sum_{k=0}^{\infty} \alpha_k = \infty$, $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$ (e.g., $\alpha_k = \frac{c}{k}$)

s^k: $\mathrm{E}\left\{\nabla f(x^k)^T s^k\right\} > 0$

s^k is an ascent direction (in expectation) at x^k

- "Exact" Stochastic Gradient Descent: $s^k = \nabla f(x^k)$

Classic SA Algorithms

- "Original" method is Robbins-Monro (1951)
- Without derivatives: Kiefer-Wolfowitz (1952)
  replaces gradient with finite-difference approximation, e.g.,

1. $x^{k+1} \leftarrow x^k - \alpha_k s^k, \qquad k = 0, 1, \ldots$

  - where

$$s^k = \frac{F(x^k + h_k I_n; \hat{\xi}^k) - F(x^k - h_k I_n; \hat{\xi}^{k+1/2})}{2h_k}$$

Classic SA Algorithms

- "Original" method is Robbins-Monro (1951)
- Without derivatives: Kiefer-Wolfowitz (1952)
  replaces gradient with finite-difference approximation, e.g.,

1. $x^{k+1} \leftarrow x^k - \alpha_k s^k, \qquad k = 0, 1, \ldots$

  - where
  $$s^k = \frac{F(x^k + h_k I_n; \hat{\xi}^k) - F(x^k - h_k I_n; \hat{\xi}^{k+1/2})}{2h_k}$$
  - Requires $2n$ evaluations every iteration
  - Can appeal to variance reduction techniques (e.g., common RNs)
  - Convergence $x^k \to x^*$ if f strongly convex (near x^*), usual conditions on α_k, $h_k \to 0$, $\sum_k \frac{\alpha_k^2}{h_k^2} < \infty$
  - K-W recommend: $\alpha_k = \frac{1}{k}$, $h_k = \frac{1}{k^{1/3}}$

Classic SA Algorithms

- "Original" method is Robbins-Monro (1951)
- Without derivatives: Kiefer-Wolfowitz (1952)
  replaces gradient with finite-difference approximation, e.g.,

  1. $x^{k+1} \leftarrow x^k - \alpha_k s^k, \qquad k = 0, 1, \ldots$

  - where
    $$s^k = \frac{F(x^k + h_k I_n; \hat{\xi}^k) - F(x^k - h_k I_n; \hat{\xi}^{k+1/2})}{2h_k}$$
  - Requires $2n$ evaluations every iteration
  - Can appeal to variance reduction techniques (e.g., common RNs)
  - Convergence $x^k \to x^*$ if f strongly convex (near x^*), usual conditions on α_k, $h_k \to 0$, $\sum_k \frac{\alpha_k^2}{h_k^2} < \infty$
  - K-W recommend: $\alpha_k = \frac{1}{k}$, $h_k = \frac{1}{k^{1/3}}$
- Extensions such as SPSA (Spall) reduce number of evaluations (see randomized methods slides. . .)

Derivative-Based Stochastic Gradient Descent

Input x^0; Repeat:

1. Draw realization $\hat{\xi}^k \sim P$ of ξ^k
2. Compute $s^k = \nabla_x F(x^k; \hat{\xi}^k)$
3. Update $x^{k+1} \leftarrow \mathcal{P}_X \left\{ x^k - \alpha_k s^k \right\}$

Derivative-Based Stochastic Gradient Descent

Input x^0; Repeat:

1. Draw realization $\hat{\xi}^k \sim P$ of ξ^k
2. Compute $s^k = \nabla_x F(x^k; \hat{\xi}^k)$
3. Update $x^{k+1} \leftarrow \mathcal{P}_X \left\{ x^k - \alpha_k s^k \right\}$

- $\nabla_x F(x^k; \hat{\xi}^k)$ is an unbiased estimator for $\nabla f(x^k)$

Derivative-Based Stochastic Gradient Descent

Input x^0; Repeat:

1. Draw realization $\hat{\xi}^k \sim P$ of ξ^k
2. Compute $s^k = \nabla_x F(x^k; \hat{\xi}^k)$
3. Update $x^{k+1} \leftarrow \mathcal{P}_X \left\{ x^k - \alpha_k s^k \right\}$

- $\nabla_x F(x^k; \hat{\xi}^k)$ is an unbiased estimator for $\nabla f(x^k)$
- Can incorporate curvature if desired

  e.g., $B^k s^k$ an unbiased estimator for $\left(\nabla^2 f(x^k)\right)^{-1} \nabla f(x^k)$
- Can work with subgradients
- Can even output $x^N = \frac{1}{N} \sum_{k=1}^{N} x^k$

Randomized Algorithms for Deterministic Problems

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

- f deterministic
- Random variables are now generated by the method, *not from the problem*
- Often assume properties of f

  e.g., ∇f is L'-Lipschitz:

$$\|\nabla f(x) - \nabla f(y)\| \leq L' \|x - y\| \qquad \forall x, y \in X$$

  e.g., f is strongly convex (with parameter τ):

$$f(x) \geq f(y) + (x-y)^T \nabla f(y) + \frac{\tau}{2}\|x-y\|^2 \qquad \forall x, y \in X$$

Basic Algorithms

Matyas (e.g., 1965):

- Input x^0; repeat:
  1. Generate Gaussian u^k (centered about 0)
  2. Evaluate $f(x^k + u^k)$
  3. $x^{k+1} = \begin{cases} x^k + u^k & \text{if } f(x^k + u^k) < f(x^k) \\ x^k & \text{otherwise.} \end{cases}$

Basic Algorithms

Matyas (e.g., 1965):

- Input x^0; repeat:
  1. Generate Gaussian u^k (centered about 0)
  2. Evaluate $f(x^k + u^k)$
  3. $x^{k+1} = \begin{cases} x^k + u^k & \text{if } f(x^k + u^k) < f(x^k) \\ x^k & \text{otherwise.} \end{cases}$

Poljak (e.g., 1987)

- Input x^0, $\{h_k, \mu_k\}_k$; repeat:
  1. Generate a random $u^k \in R^n$
  2. $x^{k+1} = x^k - h_k \dfrac{f(x^k + \mu_k u^k) - f(x^k)}{\mu_k} u^k$

  - $h_k > 0$ is the step size
  - $\mu_k > 0$ is called the smoothing parameter

Applying SA-Like Ideas to Special Cases

$$\min\left\{f(x)=\frac{1}{m}\sum_{i=1}^{m}F_i(x) : x\in X\right\}$$

m huge

Applying SA-Like Ideas to Special Cases

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

m huge

Warning: likely nonconvex!

Ex.- *Nonlinear Least Squares*
$F_i(x) = \|\phi(x; \theta^i) - d^i\|^2$
Evaluating $\phi(\cdot, \cdot)$ requires solving a large PDE

Ex.- *Sample Average Approximation*
$F_i(x) = R(x; \hat{\xi}^i)$
$\hat{\xi}^i \in \Omega$ a scenario/RV realization
(and R depends nontrivially on $\hat{\xi}^i$)

Applying SA-Like Ideas to Special Cases

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

m huge

Ex.- *Nonlinear Least Squares*
$F_i(x) = \|\phi(x; \theta^i) - d^i\|^2$
Evaluating $\phi(\cdot, \cdot)$ requires solving a large PDE

Warning: likely nonconvex!

Ex.- *Sample Average Approximation*
$F_i(x) = R(x; \hat{\xi}^i)$
$\hat{\xi}^i \in \Omega$ a scenario/RV realization
(and R depends nontrivially on $\hat{\xi}^i$)

The good:

- $\nabla f(x) = \sum_{i=1}^{m} \nabla F_i(x)$

The bad:

- m still huge

Residual Stochastic Averaging

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

"$F_i(x)$ is a member of a population of size m"

Residual Stochastic Averaging

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

"$F_i(x)$ is a member of a population of size m"

- Randomly sample $\mathcal{S}$, a subset of size $|\mathcal{S}|$, from $\{1, \cdots, m\}$

Residual Stochastic Averaging

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

"$F_i(x)$ is a member of a population of size m"

- Randomly sample $\mathcal{S}$, a subset of size $|\mathcal{S}|$, from $\{1, \cdots, m\}$
- Under minimal assumptions:

$$\mathrm{E}\left\{ \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} F_i(x) \right\} = f(x) \qquad \text{and} \qquad \mathrm{E}\left\{ \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla F_i(x) \right\} = \nabla f(x)$$

Residual Stochastic Averaging

$$\min \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} F_i(x) : x \in X \right\}$$

"$F_i(x)$ is a member of a population of size m"

- Randomly sample $\mathcal{S}$, a subset of size $|\mathcal{S}|$, from $\{1, \cdots, m\}$
- Under minimal assumptions:

$$\mathrm{E}\left\{ \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} F_i(x) \right\} = f(x) \qquad \text{and} \qquad \mathrm{E}\left\{ \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla F_i(x) \right\} = \nabla f(x)$$

- Use $-\nabla f_{\mathcal{S}} = -\frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla F_i(x)$ as direction s^k

Residual Stochastic Averaging

$$\min\left\{f(x)=\frac{1}{m}\sum_{i=1}^{m}F_i(x) : x\in X\right\}$$

"$F_i(x)$ is a member of a population of size m"

- Randomly sample $\mathcal{S}$, a subset of size $|\mathcal{S}|$, from $\{1,\cdots,m\}$
- Under minimal assumptions:

$$\mathrm{E}\left\{\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}F_i(x)\right\}=f(x) \quad \text{and} \quad \mathrm{E}\left\{\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\nabla F_i(x)\right\}=\nabla f(x)$$

- Use $-\nabla f_{\mathcal{S}}=-\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\nabla F_i(x)$ as direction s^k
- How to choose $\mathcal{S}$?

$$\mathrm{E}\left\{\|\nabla f_{\mathcal{S}_n}-\nabla f\|^2\right\}=\left(1-\frac{|\mathcal{S}|}{m}\right)\mathrm{E}\left\{\|\nabla f_{\mathcal{S}_r}-\nabla f\|^2\right\}$$

$\Rightarrow$ sampling *without replacement* ($\mathcal{S}_n$) gives lower variance than does sampling *with replacement* ($\mathcal{S}_r$)

Bayesian Optimization for Approximate Global Optimization

Statistical approaches (e.g., EGO [Jones et al., 1998])

- enjoy global exploration properties,
- excel when simulation is expensive, noisy, nonconvex

. . . but offer limited support for constraints

[Schonlau et al., 1998]; [Gramacy & Lee, 2011]; [Williams et al., 2010]

Bayesian Optimization for Approximate Global Optimization

Statistical approaches (e.g., EGO [Jones et al., 1998])

- enjoy global exploration properties,
- excel when simulation is expensive, noisy, nonconvex

...but offer limited support for constraints

[Schonlau et al., 1998]; [Gramacy & Lee, 2011]; [Williams et al., 2010]

Combine (global) statistical (objective-only) optimization tools

a) response surface modeling/emulation: training a flexible model f^k on $\{x^{(i)}, y^{(i)}\}_{i=1}^{k}$ to guide choosing $x^{(k+1)}$

e.g., [Mockus, et al., 1978], [Booker et al., 1999]

b) expected improvement (EI) via Gaussian process (GP) emulation [Jones, et al., 1998]

... with a tool from mathematical programming

c) augmented Lagrangian (AL): for handling nonlinear constraints [Powell, 1969], [Bertsekas, 1982], ...

Similar approach for combining other data terms

[Picheny, Gramacy, W., Le Digabel. *NIPS 2016*]; [Gramacy et al, *Technometrics* 2016]

Expected Improvement

Improvement: $I(x) = \max\{0, f_{\min}^k - Y(x)\}, \qquad f_{\min}^k \equiv \min_{i=1,\dots,k} f(x^i)$

Expectation of improvement (EI) has closed-form expression:

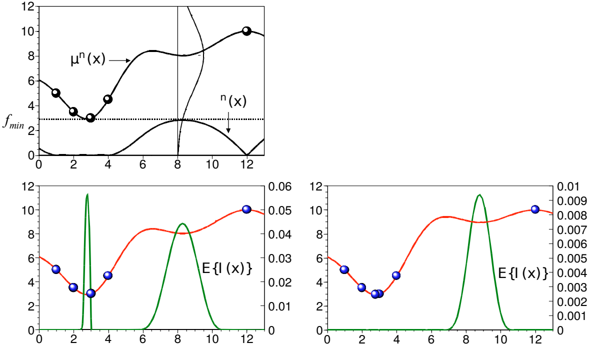
$$\mathbb{E}\{I(x)\} = (f_{\min}^k - \mu^k(x))\Phi\left(\frac{f_{\min}^k - \mu^k(x)}{\sigma^k(x)}\right) + \sigma_n(x)\phi\left(\frac{f_{\min}^k - \mu^k(x)}{\sigma^k(x)}\right)$$

Expected Improvement

Improvement: $I(x) = \max\{0, f^k_{\min} - Y(x)\}, \qquad f^k_{\min} \equiv \min_{i=1,\dots,k} f(x^i)$

Expectation of improvement (EI) has closed-form expression:

$$\mathbb{E}\{I(x)\} = (f^k_{\min} - \mu^k(x))\Phi\left(\frac{f^k_{\min} - \mu^k(x)}{\sigma^k(x)}\right) + \sigma_n(x)\phi\left(\frac{f^k_{\min} - \mu^k(x)}{\sigma^k(x)}\right)$$



- balance exploitation and exploration
- e.g., EGO: [Jones, et al., 1998]

Separate, Independent Component Modeling

- $f \longrightarrow Y_f(x)$
- $c = (c_1, \ldots, c_m) \longrightarrow Y_c(x) = (Y_{c_1}(x), \ldots, Y_{c_m}(x))$

Distribution of composite random variable serves as a surrogate for $L_A(x; \lambda, \rho)$:

$$Y(x) = Y_f(x) + \lambda^\top Y_c(x) + \frac{1}{2\rho} \sum_{j=1}^{m} \max(0, Y_{c_j}(x))^2$$

Separate, Independent Component Modeling

- $f \longrightarrow Y_f(x)$
- $c = (c_1, \ldots, c_m) \longrightarrow Y_c(x) = (Y_{c_1}(x), \ldots, Y_{c_m}(x))$

Distribution of composite random variable serves as a surrogate for $L_A(x; \lambda, \rho)$:

$$Y(x) = Y_f(x) + \lambda^\top Y_c(x) + \frac{1}{2\rho} \sum_{j=1}^{m} \max(0, Y_{c_j}(x))^2$$

Simplifications when f is known:

- Composite posterior mean available in closed form; e.g., under GP priors:

$$\mathbb{E}\{Y(x)\} = \mu_f^k(x) + \lambda^\top \mu_c^k(x) + \frac{1}{2\rho} \sum_{j=1}^{m} \mathbb{E}\{\max(0, Y_{c_j}(x))^2\}$$

- Generalized EI [Schonlau et al., 1998] gives

$$\mathbb{E}\{\max(0, Y_{c_j}(x))^2\} = \sigma_{c_j}^{2n}(x) \left[\left(1 + \left(\frac{\mu_{c_j}^k(x)}{\sigma_{c_j}^k(x)} \right)^2 \right) \Phi\left(\frac{\mu_{c_j}^k(x)}{\sigma_{c_j}^k(x)} \right) + \frac{\mu_{c_j}^k(x)}{\sigma_{c_j}^k(x)} \phi\left(\frac{\mu_{c_j}^k(x)}{\sigma_{c_j}^k(x)} \right) \right]$$

Summary

- Move beyond "blackbox" optimization
- Exploiting structure yields better solutions, in fewer simulations
- Promote optimization/modeling considerations during code development
- Correlated residuals a first step
- **Highlights attention that must be paid to model and data uncertainties**
- Can repeat for nonGaussian, MAPs, ...

[www.mcs.anl.gov/tao (Optimization toolkit) www.mcs.anl.gov/~wild (Get in touch!)]

Grateful to relevant coauthors

M. Bertolli, A. Ekström, C. Forssén, R. Gramacy, G. Hagen, M. Hjorth-Jensen, D. Higdon, G.R. Jansen, M. Kortelainen, E. Lawrence, T. Lesinski, A. Lovell, R. Machleidt, J. McDonnell, J. Moré, T. Munson, H. Nam, W. Nazarewicz, F.M. Nunes, E. Olsen, T. Papenbrock, A. Pastore, P.-G. Reinhardt, J. Sarich, N. Schunck, M. Stoitsov, J. Vary, K. Wendt, *and others*

Summary

- Move beyond "blackbox" optimization
- Exploiting structure yields better solutions, in fewer simulations
- Promote optimization/modeling considerations during code development
- Correlated residuals a first step
- **Highlights attention that must be paid to model and data uncertainties**
- Can repeat for nonGaussian, MAPs, . . .

[www.mcs.anl.gov/tao (Optimization toolkit) www.mcs.anl.gov/~wild (Get in touch!)]

Grateful to relevant coauthors

M. Bertolli, A. Ekström, C. Forssén, R. Gramacy, G. Hagen, M. Hjorth-Jensen, D. Higdon, G.R. Jansen, M. Kortelainen, E. Lawrence, T. Lesinski, A. Lovell, R. Machleidt, J. McDonnell, J. Moré, T. Munson, H. Nam, W. Nazarewicz, F.M. Nunes, E. Olsen, T. Papenbrock, A. Pastore, P.-G. Reinhardt, J. Sarich, N. Schunck, M. Stoitsov, J. Vary, K. Wendt, *and others*

Thank You!