When Life Hands You Lemons: Optimize Away!

Stefan Wild

Joint work with Christine Shoemaker (Cornell) and Jorge Moré (ANL)

Cornell School of Operations Research and Information Engineering
September '08: Argonne National Laboratory Director's Postdoctoral Fellow

June 17, 2008
Things I hope to convince you of (with very little mathematics and notation):

a. Many CS&E applications can be cast as simulation-based optimization problems

b. These problems are often computationally expensive and are *lemons* for traditional optimization techniques

c. By building tractable models of the objective, our algorithms efficiently find good solutions
Advances in Computing Hardware

Mean that simulation-based problems are evolving

ANL’s new 445-teraflops Blue Gene/P (photo: George Joch)
Advances in Computing Hardware

Mean that simulation-based problems are evolving

- Some problems become faster to solve
- Others become more realistic

ANL’s new 445-teraflops Blue Gene/P (photo: George Joch)
Optimization of Computationally Expensive Functions

Optimization is the "science of better"

Find decision parameters \( x = (x_1, \ldots, x_n) \) to improve objective \( f(x) \)
Motivating Problems

Optimization of Computationally Expensive Functions

Optimization is the “science of better”

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

$f(x)$ is expensive to evaluate at point $x$

- Evaluating $f(x)$ means running deterministic simulation $S$ which depends on $x$: $f(x) = g(S(x))$
  - Ex- $S = $ solving PDEs via finite elements
- $S$ (could/must be parallelized) takes secs/mins/hrs for 1 $x$
- Need to evaluate at many $x$ to find a good $\hat{x}_*$
Ex. 1: Town Brook Subwatershed Calibration Problem

Need accurate model to assess changes in management practices

Goal: Calibrate Soil and Water Assessment Tool (SWAT) model for flow/sediment/phosphorous (f/s/p) against 1096 days of measured data

\[ f(x) = \sum_{t=1}^{1096} \| M_t - S_t(x) \|^2 \]

- \( x \): 14 model parameters (e.g., Snow fall temp, Snowmelt temp threshold, Melt factor, Surface runoff lag, Groundwater delay)
- \( M_t \): Measured f/s/p at time \( t \)
- \( S_t(x) \): SWAT f/s/p at time \( t \)

Model requires 7mins./evaluation (EPA’s Chesapeake model > 120mins/eval)
Ex. 2: Cleanup of the Hastings Naval Ammunition Depot

48,800 acres, east of Hastings, NE

Goal: Minimize “cost” of clean up

\[ f(x) = C(x) + P(S(x)) \]

- \( x \): Pumping rates at a set of existing wells
- \( C(x) \): Cost of pumping strategy \( x \)
- \( P(\cdot) \): Penalty associated with limits on TCE, TNT
- \( S(x) \): (Simulated) Concentration given strategy \( x \)

TCE/TNT found in irrigation wells
Ex. 2: Evaluating the Hastings Function

Discretized model of the site:

- Grid covers 134 miles$^2$
- Six vertical layers: various aquifer layers and thicknesses

Evaluation requires 20 year simulation:

1. Groundwater flow [MODFLOW]
2. Contaminant transport/reaction [MT3DMS] (models TCE, PCE, TCA, DCE, TNT, RDX)

Evaluating $f$ takes several to many minutes
Ex. 3: Parameters for the Universal Nuclear Energy Density Functional (UNEDF)

SciDAC nuclear energy project

Goal: Determine parameters in the functional to fit experimental data

\[
f(x) = \sum_k w_k \| D_k - S_k(x) \|^2
\]

- \( x \): 10-20 model parameters
- \( D_k \): Data vector for \( k \)th nucleus
- \( S_k(x) \): Set of observables from the HFODD code for \( k \)th nucleus
- \( w_k \): Weight for the \( k \)th nucleus

HFODD for \( U_{236} \) requires 90 mins
(\( \approx 2000 \) nuclei \( \Rightarrow \approx 125 \) days/eval!)
Motivating Problems

These Problems are Lemons for Optimizers

Optimization takes advantage of known structure, but:

- $f$ is often a blackbox (executable only or proprietary/legacy codes)
- Only give a single output (no derivatives $\nabla S(x), \nabla^2 S(x)$)
These Problems are Lemons for Optimizers

Optimization takes advantage of known structure, but:

- $f$ is often a blackbox (executable only or proprietary/legacy codes)
- Only give a single output (no derivatives $\nabla S(x), \nabla^2 S(x)$)

Good solutions guaranteed in the limit, but:

- Usually have computational budget (due to scheduling, finances, deadlines)
- Limited number of evaluations
Our Goal

Solve general problems \( \min \{ f(x) : x \in D \subseteq \mathbb{R}^n \} \):

- Only require function values (no \( \nabla f(x) \))
- Don’t rely on finite-difference approximations
  - Can be misleading due to noise
  - Can be inefficient (each set of \( n + 1 \) evaluations useful for a single step only)
- Seek greedy and rapid decrease of function value
- Take advantage of the expense of the function
Make Use of Bank of Previously Evaluated Points

$f$ is expensive $\Rightarrow$ can afford to make better use of points

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
- Use points in a neighborhood of the best point
Motivating Problems

Our Approach

DFO Algorithms

Make Use of Bank of Previously Evaluated Points

\( f \) is expensive \( \Rightarrow \) can afford to make better use of points

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

- Points (& function values) evaluated so far
- Everything known about \( f \)

Goal:

- Make use of growing Bank as optimization progresses
- Use points in a neighborhood of the best point

Stefan Wild June 17, 2008
Motivating Problems

Our Approach

DFO Algorithms

Replace Expensive Function with Tractable Surrogate

To reduce # of expensive evaluations

Interpolation Surrogate Model:

\[ m(y^i) = f(y^i) \text{ for all } y^i \in \mathcal{Y} \]

- Conditions give model parameters
  
  Quadratic \( m(x) \):
  \[
  a + b^T x + \frac{1}{2} x^T C x
  \]
  
  RBF \( m(x) \):
  \[
  \sum_i a_i \phi(\|x - y^i\|) + p(x)
  \]

- Require geometric conditions on \( \mathcal{Y} \) to ensure interpolation is well-posed

- Need to bound \( f(x) - m(x) \)
Nonlinear Programing Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation
- Update $m_k$ and $B_k$
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation
- Update $m_k$ and $B_k$
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation
- Update $m_k$ and $B_k$
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation
- Update $m_k$ and $B_k$
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $\mathcal{B}_k$
- Minimize $m_k$ within $\mathcal{B}_k$ to obtain next point for evaluation
- Update $m_k$ and $\mathcal{B}_k$
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$.
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation.
- Update $m_k$ and $B_k$. 
Nonlinear Programming Technique: Trust-regions

Iteration $k$:

- Trust model $m_k$ within region $B_k$
- Minimize $m_k$ within $B_k$ to obtain next point for evaluation
- Update $m_k$ and $B_k$
Main Theoretical Results

Introduced framework for interpolating evaluated points while keeping model stable

Approximation bounds

- \( |f(x) - m(x)| = O(\Delta^2) \)
- \( |\nabla f(x) - \nabla m(x)| = O(\Delta) \)
- \( \nabla^2 m(x) \leq \kappa \)

for all \( x \in \{ x : \|x - x^k\| \leq \Delta \} \)

Convergence

- \( \lim_{k \to \infty} \|\nabla f(x^k)\| = 0 \)

Figure shows regions where additional points cannot be added (varying precision levels)
Our Algorithms and Software

Local & Global, Unconstrained & Bound Constrained Solvers:

**ORBIT Algorithm (RBFs)**

- Matlab code available, Open Source C code soon

**MNH Algorithm (Quadratics)**

- Matlab code available, other versions under development

**GORBIT Algorithm (RBFs)**

- Matlab code available, Open Source C code soon
Ex. 1: Town Brook Subwatershed Calibration Problem

Need accurate model to assess changes in management practices

Goal: Calibrate Soil and Water Assessment Tool (SWAT) model for flow/sediment/phosphorous (f/s/p) against 1096 days of measured data

\[
f(x) = \sum_{t=1}^{1096} \| M_t - S_t(x) \|^2
\]

\(x\) 14 model parameters (e.g., Snowfall temp, Snowmelt temp threshold, Melt factor, Surface runoff lag, Groundwater delay)

\(M_t\) Measured f/s/p at time \(t\)

\(S_t(x)\) SWAT f/s/p at time \(t\)

Model requires 7mins./evaluation (EPA’s Chesapeake model > 120mins/eval)
Town Brook Calibration Problem \( (n = 14) \)

Goal: Rapid Function Value Decrease

Best function value in \( k \) evals

Solvers in MATLAB:

- Opportunistic Pattern Search best in initial stages
- ORBIT best for budgets between 20 and 140 evals
- For larger numbers, ORBIT and PS roughly the same
- ORBIT’s 95% bands narrowest

Note: Genetic algorithms do much worse on this problem

(Mean in 30 Trials, lower is better)
Future Work and Conclusions

- Despite/because of HPC, abundance of computationally expensive blackbox functions
- Our algorithms find good solutions with fewer evaluations
- Done at the cost of additional work at optimization level (negligible CPU time relative to evaluation)

Future Work

- Address even more types of problems (general constraints, noise, parallel function evaluations)
Acknowledgments

Christine Shoemaker & Rommel Regis

Jorge Moré & MCS

DOE Computational Science Graduate Fellowship, grant DE-FG02-97ER25308

The Krell Institute

Papers available at www.orie.cornell.edu/~wild

Acknowledgments

Christine Shoemaker & Rommel Regis

Jorge Moré & MCS

DOE Computational Science Graduate Fellowship, grant DE-FG02-97ER25308

The Krell Institute

Papers available at www.orie.cornell.edu/~wild


...Thanks and problems always welcome!

Stefan Wild
June 17, 2008