Exploring the NOMAD Framework using Surrogate Problems in Optimization

Alexander Iannantuono & Zenab Kagdiwala

The University of British Columbia - Okanagan

December 8, 2020

Overview

In Chapter 1 of the course textbook [AH17], we actually dealt with a surrogate problem.

Exercise 1.4 asks to solve:

$$\min_{x} \left\{ \sum_{i=1}^{n} |x_i| : x \in \mathbb{R}^n, c(x) \leq 0 \right\} \tag{1}$$

where $c \in \mathcal{C}^1$. But this is non-smooth!

The trick was to reformulate the non-smooth problem into a smooth one by introducing new constraints. This is actually a surrogate!

Definition (Surrogates)

The problem

$$\min_{x \in X \subset \mathbb{R}^n} \{\tilde{f}(x) : \tilde{c}(x) \leq 0\},$$

is said to be a surrogate for an optimization problem

$$\min_{x \in X \subset \mathbb{R}^n} \{f(x) : c(x) \leq 0\},$$

if $\tilde{f} : X \to \mathbb{R} \cup \{\infty\}$ and $\tilde{c} : X \to (\mathbb{R} \cup \{\infty\})^m$ share some similarities with f and c but are faster to evaluate. The functions $\tilde{f}$ and $\tilde{c}$ are said to be surrogates of the true functions f and c.

- Similiarities between the smooth and non-smooth formulations of equation 1
- Other examples of surrogates

Our Investigation

- Explore surrogates using NOMAD
  - Developed by GERAD in Montréal, Québec
  - Specializes in blackbox optimization
  - Interfaced with using the command line, MATLAB, Python or other languages
- We took some sample problems from a paper by Müller et al. in [MSP13].

Investigated Sample Problems

- These problems are in the category of *reliability and redundancy*
- We have a system of $n \in \mathbb{N}$ components that work together to keep the system running as a whole
- We can make the system more reliable through number of components of type i using equation 2.

$$R_i(x_i, u_i) = 1 - (1 - x_i)^{u_i} \quad x_i \in \mathbb{R}, u_i \in \mathbb{Z}. \tag{2}$$

Observing for fixed x_i

$$\lim_{u_i \to \infty} (1 - x_i)^{u_i} = 0 \implies R_i(x_i, u_i) \to 1 \text{ as } u_i \to \infty$$

- Problem is attempting to find the best trade-off between reliability of the system and its redundancy (via number of components)
- Cost of the system is a constraint through a component reliability relation (shown in next slide), not the objective

Problem 19: A Bridge System

$$\max_{x,u} f(x,u) = \max_{x,u} R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 - R_1R_2R_3R_4$$
$$-R_1R_2R_3R_5 - R_1R_2R_4R_5 - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5$$

subject to

$$\sum_{i=1}^{5} p_i u_i^2 \leq P \qquad \sum_{i=1}^{5} w_i u_i \exp\left(\frac{u_i}{4}\right) \leq W$$

$$\sum_{i=1}^{5} c_i(x_i)\left(u_i + \exp\left(\frac{u_i}{4}\right)\right) \leq C \qquad \begin{array}{ll} 0 \leq x_i \leq 1 - 10^{-6} & i = 1, \ldots, 5 \\ u_i \in \{1, 2, \ldots, 10\} & i = 1, \ldots, 5 \end{array}$$

With $P = 250$, $W = 500$, $C = 400$, $t = 1000$. The cost reliability relation $c_i(x_i)$ function is defined as $c_i(x_i) := \alpha_i \left(\frac{-t}{\log x_i}\right)^{-\beta_i}$ for $i = 1, \ldots, 5$. The values $\alpha_i, \beta_i, p_i, w_i$ are all provided. The best known objective function value is 0.999659.

Problem 19: A Bridge System - Pictured

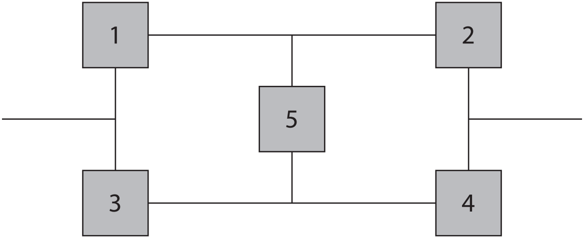

Figure: An abstract representation of the bridge system from Problem 19.

Problem 20: An Overload-Protection System

$$\max_{x,u} f(x,u) = \max_{x,u} \prod_{i=1}^{4} R_i$$

subject to

$$\sum_{i=1}^{4} p_i u_i^2 \leq P \qquad \sum_{i=1}^{4} w_i u_i \exp\left(\frac{u_i}{4}\right) \leq W$$

$$\sum_{i=1}^{4} c_i(x_i)\left(u_i + \exp\left(\frac{u_i}{4}\right)\right) \leq C \qquad \begin{array}{rl} 0.5 \leq x_i \leq 1 - 10^{-6} & i = 1, \dots, 4 \\ u_i \in \{1, 2, \dots, 10\} & i = 1, \dots, 4 \end{array}$$

With $P = 110$, $W = 200$, $C = 175$. The best known objective function value is 0.999889, and the cost reliability relation function is as before. The values $\alpha_i, \beta_i, p_i, w_i$ are different, but are all provided.

Problem 21: A Series-Parallel Bridge System

$$\max_{x,u} \; f(x,u) = \max_{x,u} \; 1 - (1 - R_1 R_2)(1 - (1 - R_3)(1 - R_4)R_5)$$

subject to

$$\sum_{i=1}^{5} p_i u_i^2 \leq P \qquad \sum_{i=1}^{5} w_i u_i \exp\left(\frac{u_i}{4}\right) \leq W$$

$$\sum_{i=1}^{5} c_i(x_i)\left(u_i + \exp\left(\frac{u_i}{4}\right)\right) \leq C \qquad \begin{array}{ll} 0 \leq x_i \leq 1 - 10^{-6} & i = 1, \ldots, 5 \\ u_i \in \{1, 2, \ldots, 10\} & i = 1, \ldots, 5 \end{array}$$

With $P = 180$, $W = 100$, $C = 175$. The best known objective function value for this problem is 0.999725. The values $\alpha_i, \beta_i, p_i, w_i$ are (again) different, but are all provided.

Problem 21: A Series-Parallel Bridge System - Pictured

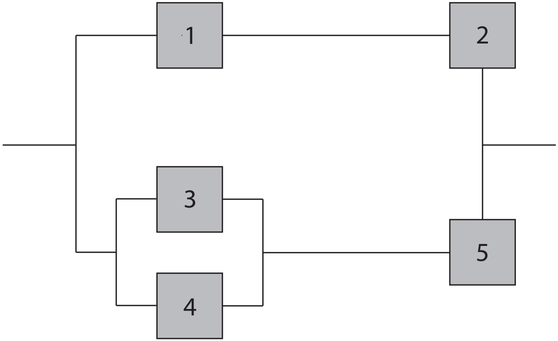

Figure: An abstract representation of the series-parallel bridge system from Problem 21.

Our Surrogate Problem

- Modified the constraints in Problems 19-21 that has $\exp\left(\frac{x}{4}\right)$
- Used an order-4 approximation via the Maclaurin series

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} \implies \exp\left(\frac{x}{4}\right) = \sum_{k=0}^{\infty} \frac{\left(\frac{x}{4}\right)^k}{k!} = \sum_{k=0}^{\infty} \frac{x^k}{4^k k!}$$

- An approximation (especially a truncation) is a surrogate!
- **Important:** We changed the lower bounds from $0 \leq x_i$ to $10^{-6} \leq x_i$ for $i = 1, \ldots, 5$
- Link to GitHub repository: github.com/armeehn/nomad

Results

- General visible patterns in the data
- Randomness has more of an effect for small n
- Important point: “feasible” solutions in a surrogate might not be feasible to the original problem!

	Problem 19				Problem 20				Problem 21			
	Python		MATLAB		Python		MATLAB		Python		MATLAB	
n	$\bar{x}$	s^2	$\bar{x}$	s^2	$\bar{x}$	s^2	$\bar{x}$	s^2	$\bar{x}$	s^2	$\bar{x}$	s^2
100	0.8948	4.46E-02	0.7267	3.65E-02	0.9788	3.27E-03	0.7719	7.41E-02	0.7621	9.80E-01	0.8246	1.80E-02
200	0.9854	1.09E-03	0.7610	3.65E-02	0.9992	5.99E-07	0.8371	8.63E-02	0.9796	1.36E-03	0.8907	4.23E-03
300	0.9987	2.08E-06	0.8143	7.64E-02	0.9997	4.24E-07	0.9000	2.17E-02	0.9969	4.46E-05	0.9096	2.19E-03

Table: Data obtained over $N = 30$ runs for Problems 19, 20, and 21

Summary

Surrogates are good in cases when:

- The true optimization problem is known but "expensive" to compute
- Can create a similar problem that gives a better idea of feasible solutions

But, keep in mind that:

- Not the exact same problem unless it is a reformulation that can use fast methods
- "Feasible" solutions might not be truly feasible w.r.t. original problem

Some Wisdom for NOMAD usage

In our project, we used NOMAD through the MATLAB interface, and invoked a Python script that was treated as a blackbox. These things matter! That being said, this dives into programming languages, compilers, etc.

References

Charles Audet and Warren Hare, *Derivative-free and blackbox optimization*, Springer International Publishing, 2017.

Juliane Müller, Christine A. Shoemaker, and Robert Piché, *So-mi: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems*, Computers & Operations Research **40** (2013), no. 5, 1383 – 1400.

Thanks!