Optimization of Noisy Functions: Application to Simulations

Geng Deng Michael C. Ferris

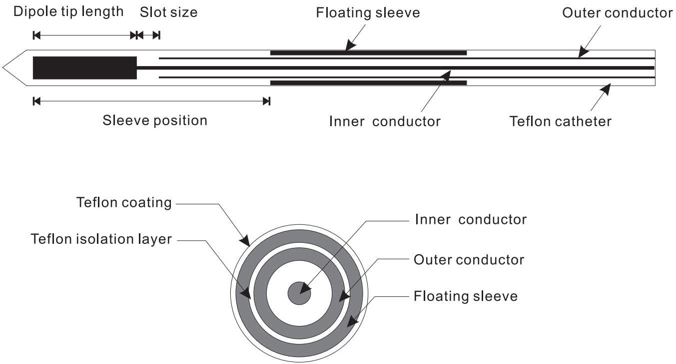
University of Wisconsin-Madison

Optimization and Engineering Applications
Banff International Research Station
November 13, 2006

Simulation-based optimization problems

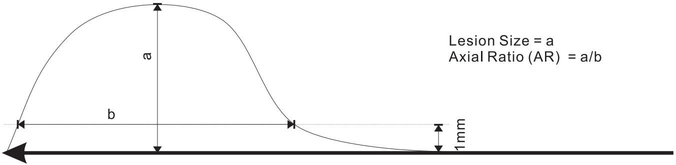
- Computer simulations are used as substitutes to evaluate complex real systems.
- Simulations are widely applied in epidemiology, engineering design, manufacturing, supply chain management, medical treatment and many other fields.
- The goal: Optimization finds the best values of the decision variables (design parameters or controls) that minimize some performance measure of the simulation.

Design a coaxial antenna for hepatic tumor ablation



Simulation of the electromagnetic radiation profile

Finite element models (MultiPhysics v3.2) are used to generate the electromagnetic (EM) radiation fields in liver given a particular design



Metric	Measure of	Goal
Lesion radius	Size of lesion in radial direction	Maximize
Axial ratio	Proximity of lesion shape to a sphere	Fit to 0.5
S_{11}	Tail reflection of antenna	Minimize

A general problem formulation

- We formulate the simulation-based optimization problem as

$$\min_{x \in \mathcal{S}} F(x) = \mathbb{E}_{\omega}[f(x, \omega(x))],$$

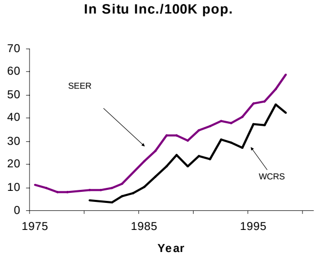
$\omega(x)$ is a random factor arising in the simulation process.
The sample response function $f(x, \omega)$

  - typically does not have a closed form, thus cannot provide gradient or Hessian information
  - is normally computationally expensive
  - is affected by uncertain factors in simulation

The underlying objective function $F(x)$ has to be estimated.

Simulation callibration

- Detailed individual-woman level discrete event simulation of Wisconsin Breast Cancer Incidence (using 4 processes):
  - Breast cancer natural history
  - Breast cancer detection
  - Breast cancer treatment
  - Non-breast cancer mortality among US women
- Replicate breast cancer surveillance data: 1975-2000



9 to 30 parameters related to distributions within simulations

Other Applications

- SVM parameter tuning
- Inverse Optimization, e.g. structural properties in existing buildings
- Stochastic Integer Programming
  - First stage (small scale) continuous decision
    - How many newspapers to send to different locations
    - How much “disaster relief” supplies to send to different locations
  - Second stage (large scale mixed integer) decision, after random demand known
    - What sales facilities to open and what to move where
    - Where to send the emergency teams and supplies

Two-stage stochastic program with recourse

$$\begin{array}{ll} \min_{x_i} & \sum_i C_i x_i + \mathbb{E}_\omega \left[f(\boldsymbol{x}, D(\omega)) \right] \\ \text{s.t.} & x_i \geq 0, \end{array}$$

Second stage recourse problem is a mixed-integer problem

$$\begin{array}{rll} f(\boldsymbol{x}, D) = & \min\limits_{l_j, s_j, z_j, t_{i,j}, u_j} & \sum_j P_j l_j + \sum_j H_j z_j + \sum_{i,j} S_{i,j} t_{i,j} + \sum_j O_j u_j \\ & \text{s.t.} & s_j + l_j = D_j, \quad \forall j, \\ & & s_j \leq D_j u_j, \quad \forall j, \\ & & z_j = -s_j + \sum_i t_{i,j}, \quad \forall j, \\ & & x_i = \sum_j t_{i,j}, \quad \forall i, \\ & & s_j, l_j, z_{i,j}, t_{i,j} \geq 0, \quad \forall i, j, \\ & & u_j \in \{0, 1\}, \quad \forall j. \end{array}$$

Basic framework and tools

- Small scale x controls/design variables
- Simulation is refinable (replications, more samples in DES, finer discretization)

$$F(x) \simeq \frac{1}{N} \sum_{j=1}^{N} f(x, \omega_j)$$

- Issues:
  - Comparisons
  - Termination
  - Model/solution volatility
  - Common random numbers

A simple discrete optimization case

- For example, test elasticity of a set of balls. Here $\mathcal{S} = \{1, 2, 3, 4, 5\}$ represents a set of 5 balls.

- Objective: Choose the ball with the largest expected bounce height $F(x_i)$. $f(x_i, \omega_j)$ corresponds to a single measurement in an experiment.

How to select the best system

- Choose the maximum sample mean

$$\arg\max_{i \in \mathcal{S}} \bar{\mu}_i := \frac{1}{N_i} \sum_{j=1}^{N_i} f(x_i, \omega_j),$$

  where N_i is the number of experiments.
- Select the best system with high accuracy, while controlling the total amount of simulation runs.
- Two approaches
  - Ranking and selection
    S.-H. Kim and B. L. Nelson, "Selecting the Best System: Theory and Methods."
  - Bayesian approach
    S. E. Chick, and K. Inoue, "New Two-stage and Sequential Procedures for Selecting the Best Simulated System."
    H.-C. Chen, C.-H. Chen, and E. Yucesan, "An Asymptotic Allocation for Simultaneous Simulation Experiments."

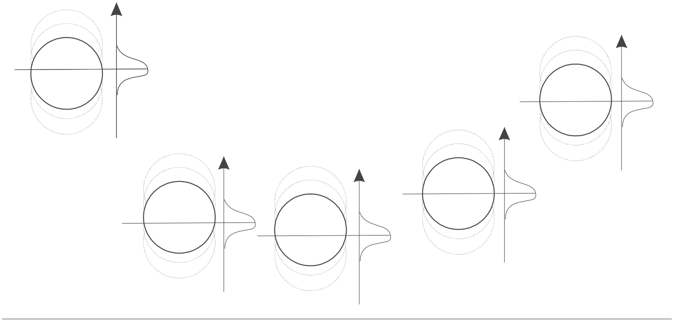
Bayesian approach

- Denote the mean of the simulation output for each system as $\mu_i = F(x_i) = \mathbb{E}_\omega[f(x_i, \omega)]$.
- In a Bayesian perspective, the means are considered as Gaussian random variables whose posterior distributions can be estimated as

$$\mu_i | X \sim N(\bar{\mu}_i, \hat{\sigma}_i^2 / N_i),$$

where $\bar{\mu}_i$ is sample mean and $\hat{\sigma}_i^2$ is sample variance. The above formulation is one type of posterior distribution.

Posterior distributions facilitate comparison



Now it is easy to compute the probability of correct selection (PCS).

Compute the PCS

- Pairwise comparison

$$PCS = Pr(\mu_1 \geq \mu_2) \sim Pr(\mu_1 \geq \mu_2 | X) = Pr(\mu_1 | X - \mu_2 | X \geq 0).$$

- Multiple comparisons (Bonferroni inequality):

$$\begin{aligned} PCS &= Pr(\mu_b - \mu_i \geq 0, i = \{1, 2, \cdots, K\} \setminus \{b\}) \\ &\sim 1 - \textstyle\sum_{i=1, i \neq b}^{K} Pr(\mu_b - \mu_i < 0) \end{aligned}$$

Summary of the Bayesian approach

- Once the PCS is determined, choose a suitable sample number of each system N_i such that the best system is selected with desired accuracy

$$PCS \geq 1 - \alpha.$$

- Bayesian approach
  - Utilizes both mean and variance information
  - Simple and direct to implement
  - Without using indifference-zone parameter δ
- Directly applicable to pattern search methods

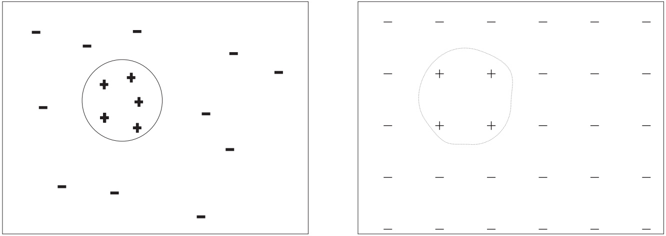
Two phase approach

- Linked two-phase approach
  - Phase I: global issues / exploration: rough
  - Phase II: local issues / exploitation: refined
- Phase I Classifier: surrogate for indicator function of the level set

$$L(c) = \{x \mid F(x) \leq c\} \simeq \left\{ x \;\middle|\; \frac{1}{N} \sum_{j=1}^{N} f(x, \omega_j) \leq c \right\}$$

- c is a quantile point of the responses
- Training set: space filling samples (points) from the whole domain (e.g. mesh grid; Latin Hypercube Sampling)

Classifiers predict new refined samples as promising



(a) Training samples in $L(c)$ are classified as positive and others are negative. The solid circle represents estimated $L(c)$.

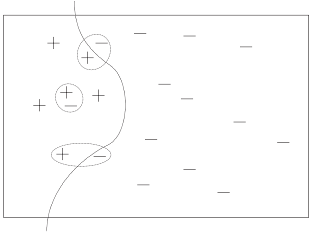
(b) Classify a set of more refined space-filling samples. Four points are predicted as positive and rest are negative. The classifier is refined.

Validate the subset of the identified promising points by performing additional simulations

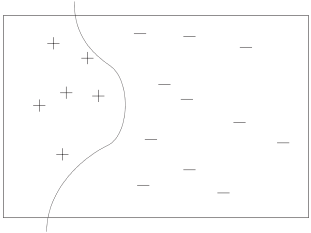
Imbalanced data

- Under-sampling of the negative class using one-sided selection:
  - Keep all the positive samples unchanged. To obtain a consistent subset C of the original training set T: Train 1-NN classifier with the positive samples plus one randomly chosen negative sample. Test the 1-NN rule on the rest of samples in the set T. The new subset C will consist of the misclassified samples plus the samples used for training. In doing this, we derive a consistent subset C of T such that all the samples in T can be correctly predicted using the 1-NN rule on C.
  - Detect the Tomek links in C and remove the associated negative samples.
- Over-sample of the positive class by duplicating all the positive samples once.

Cleaning the dataset with Tomek links

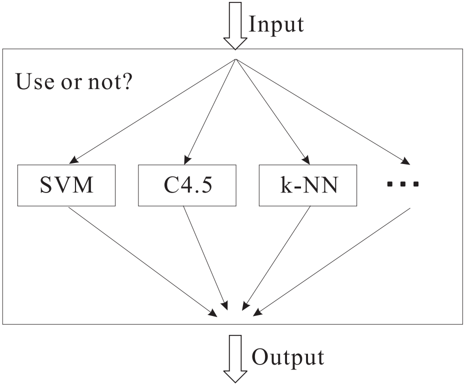


(c) Determine the pairs of Tomek links



(d) Remove the negative samples participating as Tomek links

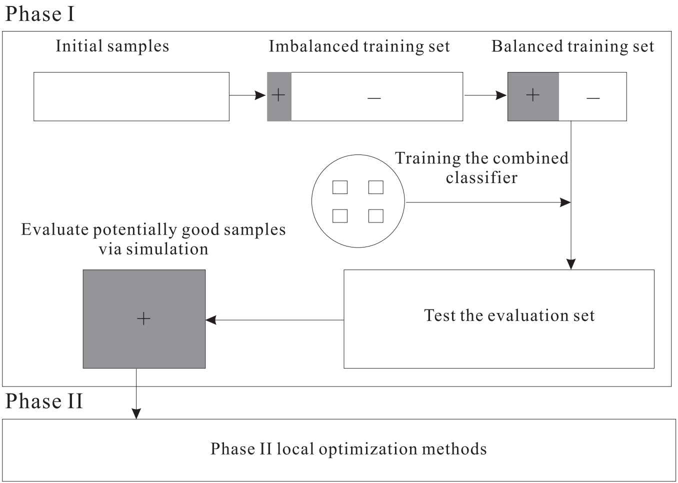
Assemble classifiers using a voting scheme



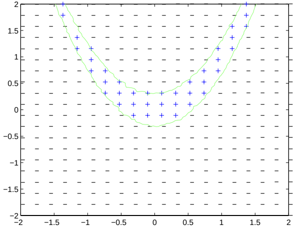
The voting scheme

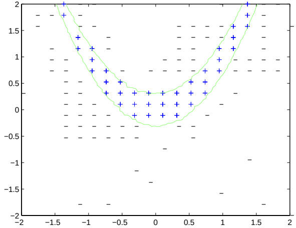
1. Split the input training set T into two subsets, denoted as training subset T_1 (randomly selected 75% of samples) and testing subset T_2 (the rest).
2. Perform a prior performance test: train each classifier on the training subset and evaluate it with the samples in the testing subset. If the classification accuracy is not assured, i.e., failing the criterion that g-mean $g \geq 0.5$, discard the classifier.
3. Classifiers that pass the performance test are trained on the original training set T. In the evaluation process, assign new samples to the class which is majorally voted.

Classifier Phase I approach

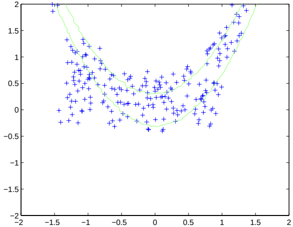


Banana example





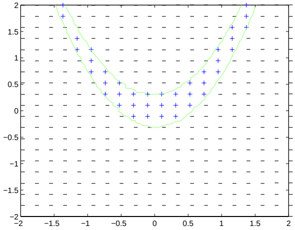
Original Predicted Training

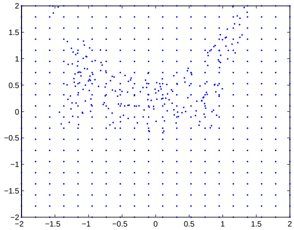


Application to WBCE

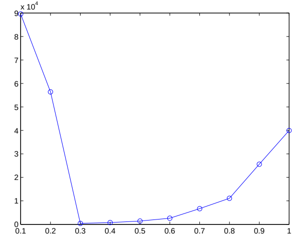
- 500,000 points x generated uniformly at random
- Using CONDOR (120 machines) can evaluate approximately 1000 per day $f(x,\omega)$ involves simulation of 3 million women
- 363 are in $L(10)$: "simulated points out of data envelope"
- Using Phase I: 10,000 points evaluated, 220 points suggested, 195 are in L(10)
- New dataset with 10 replications at points with scores ≤ 30
- Far fewer points in L(10)
- Phase I results in new points (all are good), but 2 of which seem better than the "experts" best solution

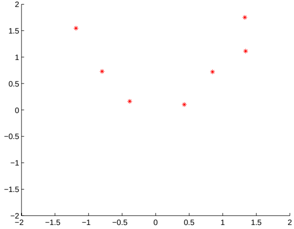
The non-parametric "linking" idea





Original / $sse(h)$ Data / Result





Determine TR radius Δ by non-parametric regression

The idea is to determine the best 'window size' for non-parametric local regression, and then use the 'window size' as the initial trust region radius Δ.

1. $\Delta \in \arg\min_h sse(h)$
2. $sse(h)$ is the sum of squares error of knock-one out prediction. Given a window-size h and a point x_0, the knock-one out predicted value is $Q(x_0)$, where $Q(x)$ is constructed using the data points within the ball $\{x| \, \|x - x_0\| \leq h\}$.

$$Q(x) = c + g^T(x - x_0) + \frac{1}{2}(x - x_0)^T H(x - x_0)$$

.

Steps to generate the initial point set $\mathcal{I}$

1. Use non-parametric regression method to determine the initial trust region radius Δ, and define the subregion radius

$$d := 2\Delta$$

2. Sort the available points by their objective values
3. Put the best point into the initial point set $\mathcal{I}$
4. For each x taken in ascending order from the candidate point set, compute the shortest distance from the point to the initial point set

$$dist = \min_{y_i \in \mathcal{I}} \|y_i - x\|$$

5. If $dist > d$, add the point to the initial point set $\mathcal{I} := \mathcal{I} \cup \{x\}$
6. Stop if $card(\mathcal{I}) > 10$ or all the points have been enumerated.

Issues for Phase I methods

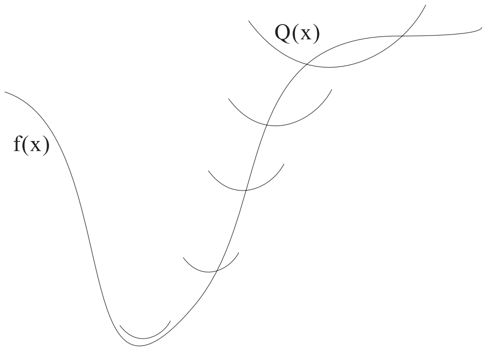
- Methods must provide a global view of function
- Should allow for varying region sizes
- Re-use of existing function evaluations
- Alternative approach: DIRECT (Jones, 1994)
- Pattern search, Nelder Mead do not routinely provide multi-start information

Phase II: refine solution

- Basic approach: reduce function uncertainty by averaging multiple samples per point.
- Potential difficulty:
  efficiency of algorithm vs number of simulation runs
- We apply Bayesian approach to determine appropriate number of samples per point, while simultaneously enhancing the algorithm efficiency
- Guarantee the global convergence of the algorithm

A noisy extension of the UOBYQA algorithm

The base derivative free optimization algorithm: The UOBYQA (Unconstrained Optimization BY Quadratic Approximation) algorithm is based on a trust region method. It constructs a series of local quadratic approximation models of the underlying function.



Quadratic model construction and trust region subproblem solution

For iteration $k = 1, 2, \ldots,$

- $\ldots$
- Construct a quadratic model via interpolation

$$Q(x,\omega) = f(x_k,\omega) + g_Q^T(\omega)(x - x_k) + \frac{1}{2}(x - x_k)^T G_Q(\omega)(x - x_k)$$

The model is unstable since interpolating noisy data

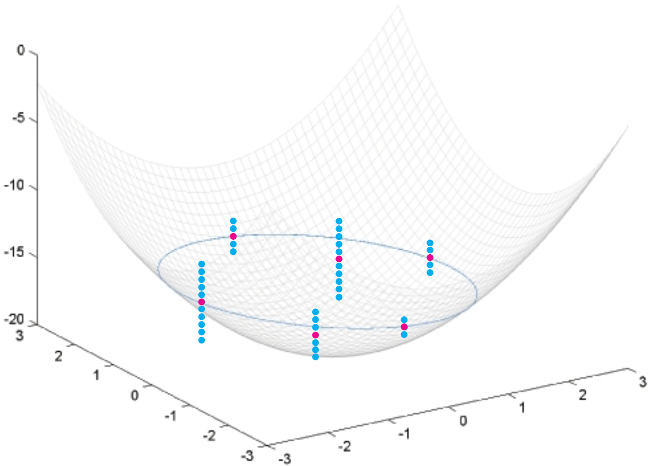
- Solve the trust region subproblem

$$\begin{aligned} s_k(\omega) = \arg\min_s \quad & Q(x_k + s, \omega) \\ s.t. \quad & \|s\|_2 \leq \Delta_k \end{aligned}$$

The solution is thus unstable

- $\ldots$

Why is the quadratic model unstable?



How to stabilize the quadratic model?

Let $\{y^1, y^2, \ldots, y^L\}$ be the interpolation set.

- Quadratic interpolation model is a linear combination of Lagrange functions:

$$Q(x, \omega) = \sum_{j=1}^{L} f(y^j, \omega) l_j(x).$$

- Each piece $l_j(x)$ is a quadratic polynomial, satisfying

$$l_j(y^i) = \delta_{ij}, i = 1, 2, \cdots, L.$$

- The coefficients of l_j are uniquely determined, independent of the random objective function.

Bayesian estimation of coefficients c_Q, g_Q, G_Q

In Bayesian approach, the mean of function output $\mu(y^j) := \mathbb{E}_\omega f(y^j, \omega)$ is considered as a random variable:
Normal posterior distributions:

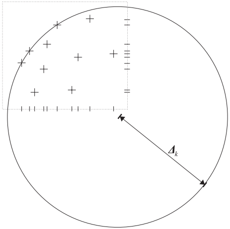
$$\mu(y^j)|X \sim N(\bar{\mu}(y^j), \hat{\sigma}^2(y^j)/N_j).$$

Thus the coefficients of the quadratic model are estimated as:

$$\begin{aligned} g_Q|X &= \textstyle\sum_{j=1}^{L}(\mu(y^j)|X)g_j, \\ G_Q|X &= \textstyle\sum_{j=1}^{L}(\mu(y^j)|X)G_j. \end{aligned}$$

- g_j, G_j are coefficients of Lagrange functions l_j.
- g_j, G_j are deterministic and determined by points y^j.

Constraining the variance of coefficients



- Generate samples of function values from these (estimated) distributions.
- Trial solutions are generated within a trust region. The standard deviation of the solutions is constrained.

$$\max_{i=1}^{n} std([s^{*(1)}(i), s^{*(2)}(i), \cdots, s^{*(M)}(i)]) \leq \beta\Delta_k.$$

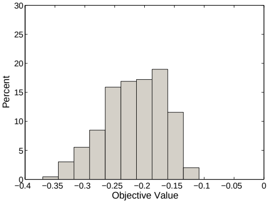
Noisy UOBYQA for Rosenbrock, $n = 2$ and $\sigma^2 = 0.01$

Iteration (k)	FN	$F(x_k)$	Δ_k
1	1	404	2
20	78	3.56	9.8×10^{-1}
40	140	0.75	1.2×10^{-1}
60	580	0.10	4.5×10^{-2}
80	786	0.0017	5.2×10^{-3}
✓ Stops with the new termination criterion			
100	1254	0.0019	2.8×10^{-4}
120	2003	0.0016	1.1×10^{-4}
✓ Stops with the termination criterion $\Delta_k \leq 10^{-4}$			

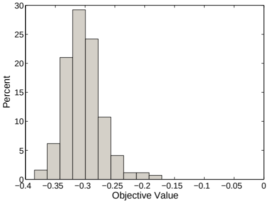
Two-phase approach to optimize antenna design metrics

- Uniform LHS to generate 2,000 design samples to evaluate with the FE simulation model (range [-0.3705, 3597])
- Histogram of objective values over interval [-0.3705, 0]
- $c = -0.2765$ the 10% quantile. $L(c)$ has 199 positive samples (1801 negative)
- Balancing procedure: 398 positive vs. 388 negative samples
- 5 (of 6 tested) classifiers in ensemble
- Refined data: 15,000 designs, 522 predicted by classifiers as positive, 74% correctly
- The best Phase I design has value -0.3850.

Coaxial antenna design



(e) First stage evaluations (training data)



(f) Our new antenna design

- (Modified) UOBYQA started from best point: (13.6 2.7 19.0 0.3 0.1) mm, value -0.3850.
- UOBYQA returned an optimal solution: (15.9 2.4 19.0 0.3 0.1) mm, value -0.4117.

Sample path extension: changing liver properties

- Common random numbers allow variance reduction, correlated noise.
- Extension of ideas to Variable-Number Sample-Path Optimization method.
- Application: Dielectric tissue properties varied within $\pm 10\%$ of average properties to simulate the individual variation.
- Bayesian VNSP algorithm yields an optimal design that is a 27.3% improvement over the original design and is more robust in terms of lesion shape and efficiency.

Other approaches to constrain the variance of coefficients

- Test the sufficient reduction criterion:

$$Pr\left(Q_k(x_k) - Q_k(x_k + s^*) \geq \kappa_{mdc}\|g_k^\infty\| \min\left[\frac{\|g_k^\infty\|}{\kappa_{Qh}}, \Delta_k\right]\right) \geq 1-\alpha$$

- Quantify variance of individual coefficient in Q:

$$\frac{std(g_Q(i'))}{E[g_Q(i')]} \leq \beta, i' = 1, \cdots, n$$

$$\frac{std(G_Q(i', j'))}{E[G_Q(i', j')]} \leq \beta, i', j' = 1, \cdots, n$$

Two-stage stochastic integer program

5 suppliers, 100 retailers, random demand $N(\mu, \sigma^2)$, $\mu \in [10, 30]$.
Phase I: classification-based search, Phase II: UOBYQA

- 2000 points for classification, sampled from box $\prod_{i=1}^{5}[200, 500]$ (range 5325-6467).
- Phase I as described, 10% of the points positive, all 6 classifiers applied, etc.
- 510 (from 5000) additional points were predicted as positive and evaluated via simulation (range 5313-5815).
- Non-parametric approach determined "window size" $\Delta = 90$
- Local optimization method (VNSP) started at 4 points from initial point set.
- Phase II objective values are close, (range 5262-5268). Each optimization problem used 5000-10000 MILP's (from GAMS).

Conclusions and future work

- Coupling statistical and optimization techniques can effectively process noisy function optimizations
- Significant gains in system performance and robustness are possible
- General framework proposed allows multiple methods to be "hooked" up
- How to reuse function evaluations from Phase I in Phase II?
- Application to more engineering problems
- Default parameters are being evaluated - maybe use the algorithm itself!