Making Decisions via Simulation

[Law, Ch. 10], [Handbook of Sim. Opt.], [Haas, Sec. 6.3.6]

Peter J. Haas

CS 590M: Simulation
Spring Semester 2020

Overview

Goal: Select best system design or parameter setting

- Performance under each alternative estimated via simulation

$$\min_{\theta \in \Theta} f(\theta)$$

  where Θ = feasible set

- f is often of the form $f(\theta) = E_\theta[c(X,\theta)]$
  - X is estimated from the simulation
  - E_θ indicates that dist'n of X depends on θ

Overview, Continued

Three cases:

1. Θ is uncountably infinite (continuous optimization)
   - Robbins-Monro Algorithm
   - Metamodel-based optimization
   - Sample average approximation
2. Θ is small and finite (ranking and selection of best system)
   - E.g., Dudewicz and Dalal (HW #7)
3. Θ is a large discrete set (discrete optimization)

Not covered here: Markov decision processes

- Choose best policy: I.e., choose best *function* π, where $\pi(s)$ = action to take when new state equals s [Chang et al., 2007]

Factor Screening

Goal: Identify the most important drivers of model response

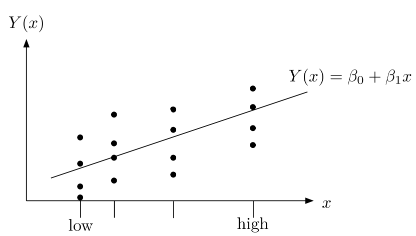
- Needed for understanding
- Needed to focus modeling resources (e.g., input distributions)
- Needed to select decision variables for optimization

Factor Screening, Continued

Based on a simulation metamodel, for example:

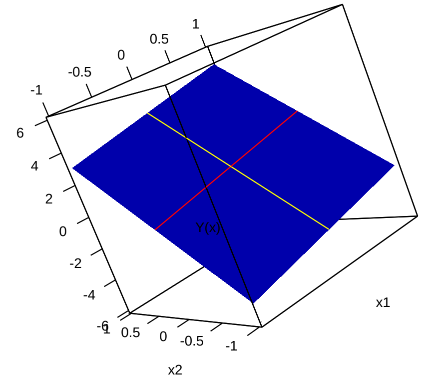
$$Y(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$$

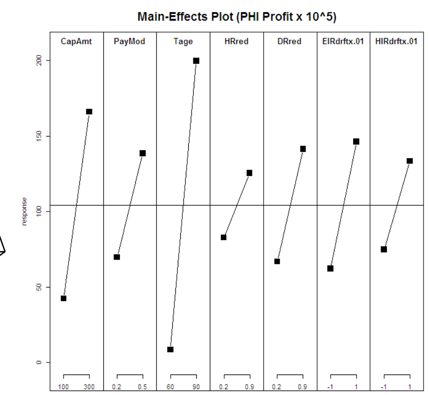
- Y = simulation model output
- Parameters $x = (x_1, \ldots, x_k)$
- ϵ = noise term (often Gaussian)
- Estimate the β_i's using "low" and "high" x_i values
- Test if each $|\beta_i|$ is significantly different from 0
- Will talk more about metamodels later on...



Factor Screening, Continued

β_i coefficients indicate parameter importance





Factor Screening, Continued

Challenge: Many Features

- Example with $k = 3$:

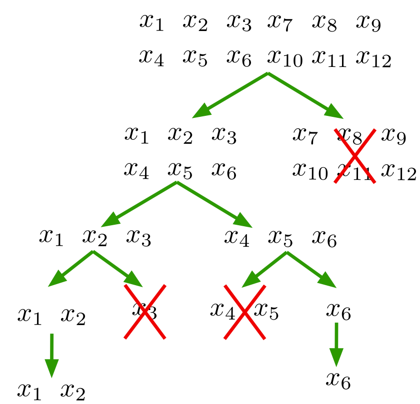
$$\hat{\beta}_1 = \frac{Y(h,l,l) + Y(h,l,h) + Y(h,h,l) + Y(h,h,h)}{4} - \frac{Y(l,l,l) + Y(l,l,h) + Y(l,h,l) + Y(l,h,h)}{4}$$

- In general, need 2^k simulations ("full factorial" design)
- Can be smarter, e.g., "fractional factorial" designs (will talk about this soon)
- In general: interplay between metamodel complexity (e.g., β_{ij} terms) and computational cost

Factor Screening, Continued

Sequential bifurcation

- For huge number of factors
- Assumes Gaussian noise, nonnegative β's
- Test groups (sums of β_i's)



Continuous Stochastic Optimization

Robbins-Monro Algorithm

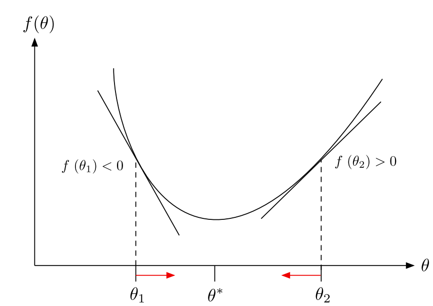
- Goal: $\min_{\theta \in [\underline{\theta}, \overline{\theta}]} f(\theta)$
- Estimate $f'(\theta)$ and use stochastic approximation (also called stochastic gradient descent)

$$\theta_{n+1} = \Pi\left(\theta_n - \left(\frac{a}{n}\right) Z_n\right)$$

where

- $a > 0$ (the gain)
- $E[Z_n] = f'(\theta_n)$
- $\Pi(\theta) = \begin{cases} \underline{\theta} & \text{if } \theta < \underline{\theta} \\ \theta & \text{if } \underline{\theta} \le \theta \le \overline{\theta} \\ \overline{\theta} & \text{if } \theta > \overline{\theta} \end{cases}$

(projection function)



Continuous Stochastic Optimization, Continued

Convergence

- Suppose that θ^* is true minimizer and the only local minimizer
- Under mild conditions, $\lim_{n\to\infty} \theta_n = \theta^*$ a.s.
- Q: If θ^* is not the only local minimizer, what can go wrong?
- For large n, θ_n has approximately a normal dist'n

Estimation Algorithm for $100(1-\delta)$% Confidence Interval

1. Fix $n \geq 1$ and $m \in [5, 10]$
2. Run the Robbins-Monro iteration for n steps to obtain θ_n
3. Repeat Step 2 a total of m times to obtain $\theta_{n,1}, \ldots, \theta_{n,m}$
4. Compute point estimator $\bar{\theta}_m = (1/m)\sum_{j=1}^m \theta_{n,j}$
5. Compute $100(1-\delta\%)$ CI as $[\bar{\theta}_m - \frac{s_m t_{m-1,\delta}}{\sqrt{m}}, \bar{\theta}_m + \frac{s_m t_{m-1,\delta}}{\sqrt{m}}]$

where $s_m^2 = \frac{1}{m-1}\sum_{j=1}^m (\theta_{n,j} - \bar{\theta})^2$ and $t_{m-1,\delta}$ = Student-t quantile

Continuous Stochastic Optimization, Continued

Remarks

- Variants available for multi-parameter problems
- Drawbacks to basic algorithm are slow convergence and high sensitivity to the gain a; current research focuses on more sophisticated methods
- Simple improvement: return best value seen so far

Kiefer-Wolfowitz algorithm

- Replaces derivative $f'(\theta_n)$ by finite difference $\frac{f(\theta_n+\Delta)-f(\theta_n-\Delta)}{2\Delta}$
- Spalls' simultaneous perturbation stochastic approximation (SPSA) method handles high dimensions
  - At the kth iteration of a d-dimensional problem, run simulation at $\theta_k \pm c\Delta_k$, where $c > 0$ and Δ_k is a vector of i.i.d. random variables $I_1, \ldots, I_d$ with $P(I_j = 1) = P(I_j = -1) = 0.5$

Estimating the Derivative $f'(\theta_n)$

Suppose that $f(\theta) = E_\theta[c(X, \theta)]$

- Ex: M/M/1 queue with interarrival rate λ and service rate θ
- X = average waiting time for first 100 customers
- $c(x, \theta) = a\theta + bx$ (trades off operating costs and delay costs)

Use likelihood ratios

- We have $f(\theta + h) = E_{\theta+h}[c(X, \theta + h)] = E_\theta[c(X, \theta + h)L(h)]$ for appropriate likelihood $L(h)$

$$\begin{aligned}
f'(\theta) &= \lim_{h\to 0}\frac{f(\theta+h)-f(\theta)}{h} \\
&= \lim_{h\to 0} E_\theta\left[\frac{c(X,\theta+h)L(h) - c(X,\theta)L(0)}{h}\right] \\
&= E_\theta\left[\lim_{h\to 0}\frac{c(X,\theta+h)L(h) - c(X,\theta)L(0)}{h}\right] \quad \text{under regularity cond.} \\
&= E_\theta\left[\frac{d}{dh}\big(c(X,\theta+h)L(h)\big)\Big|_{h=0}\right] \\
&= E_\theta\left[c'(X,\theta) + c(X,\theta)L'(0)\right] \qquad c' = \partial c/\partial\theta \quad L' = \partial L/\partial h
\end{aligned}$$

Derivative Estimation, Continued

To estimate $g(\theta) \triangleq f'(\theta) = E_\theta\left[c'(X,\theta) + c(X,\theta)L'(0)\right]$

- Simulate system to generate i.i.d. replicates $X_1, \ldots, X_m$
- At the same time, compute $L'_1(0), \ldots, L'_m(0)$
- Compute the estimate $g_m(\theta) = \frac{1}{m}\sum_{i=1}^m [c'(X_i,\theta) + c(X_i,\theta)L'_i(0)]$
- Robbins and Monro showed that taking $m = 1$ is optimal (many approximate steps vs few precise steps)

*n*th step of R-M algorithm

1. Generate a single sample X of the performance measure and compute $L'(0)$
2. Set $Z_n = g_1(\theta_n) = c'(X,\theta_n) + c(X,\theta_n)L'(0)$
3. Set $\theta_{n+1} = \Pi\left(\theta_n - \left(\frac{a}{n}\right)Z_n\right)$

Derivative Estimation, Continued

Ex: M/M/1 queue

- Let $V_1, \ldots, V_{100}$ be the 100 generated service times
- Let X = avg of the 100 waiting times (the perf. measure)

$$L(h) = \prod_{i=1}^{100} \frac{(\theta+h)e^{-(\theta+h)V_i}}{\theta e^{-\theta V_i}} = \prod_{i=1}^{100}\left(\frac{\theta+h}{\theta}\right) e^{-hV_i}$$

$$\Rightarrow L'(0) = \sum_{i=1}^{100}\left(\frac{1}{\theta} - V_i\right) \quad \text{(can be computed incrementally)}$$

$$c(x,\theta) = a\theta + bx \quad \Rightarrow \quad c'(x,\theta) = a$$

$$Z_n = c'(X_n,\theta_n) + c(X_n,\theta_n)L'_n(0) = a + (a\theta_n + bX_n)\sum_{i=1}^{100}\left(\frac{1}{\theta_n} - V_{n,i}\right)$$

Derivative Estimation, Continued

A trick for computing $L'(0)$

- Likelihood ratio often has form: $L(h) = r_1(h)r_2(h)\cdots r_k(h)$
- E.g., for a GSMP, $r_i(h) = \frac{f_{\theta+h}(X;s',e',s,e^*)}{f_\theta(X;s',e',s,e^*)}$ or $\frac{P_{\theta+h}(S_{j+1};S_j,e_j^*)}{P_\theta(S_{j+1};S_j,e_j^*)}$
- Using the product rule and the fact that $r_i(0) = 1$ for all i

$$\begin{aligned}\frac{d}{dh}L(h)\Big|_{h=0} &= \frac{d}{dh}\left(r_1(h)r_2(h)\cdots r_k(h)\right)\Big|_{h=0} \\ &= \left[r_1(h)\frac{d}{dh}\left(r_2(h)\cdots r_k(h)\right)\right]_{h=0} + \left[r'_1(h)r_2(h)\cdots r_k(h)\right]_{h=0} \\ &= \frac{d}{dh}\left[r_2(h)\cdots r_k(h)\right]_{h=0} + r'_1(0)\end{aligned}$$

- By induction: $L'(0) = r'_1(0) + \cdots + r'_k(0)$ (compute incrementally)
- For GSMP example (with $f'_\theta = \partial f_\theta / \partial\theta$):

$$r'_i(0) = \frac{\frac{d}{dh} f_{\theta+h}(X;s',e',s,e^*)\big|_{h=0}}{f_\theta(X;s',e',s,e^*)} = \frac{f'_\theta(X;s',e',s,e^*)}{f_\theta(X;s',e',s,e^*)}$$

Derivative Estimation, Continued

Trick continued: M/M/1 queue

$$L(h) = \prod_{i=1}^{100} r_i(h) = \prod_{i=1}^{100} \frac{f_{\theta+h}(V_i)}{f_\theta(V_i)}$$

$$f_\theta(v) = \theta e^{-\theta v} \quad \text{and} \quad f'_\theta(v) = (1-\theta v)e^{-\theta v}$$

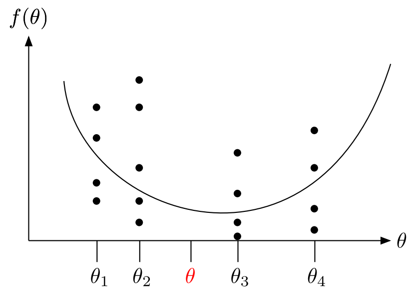
$$L'(0) = \sum_{i=1}^{100} \frac{f'_\theta(V_i)}{f_\theta(V_i)} = \sum_{i=1}^{100} \frac{(1-\theta V_i)e^{-\theta V_i}}{\theta e^{-\theta V_i}} = \sum_{i=1}^{100}\left(\frac{1}{\theta} - V_i\right)$$

Derivative Estimation, Continued

Remarks

- Derivative estimation is interesting outside of optimization for sensitivity analysis
- Drawback of likelihood-ratio derivative estimator: variance of likelihood ratio increases with length of simulation
- Alternative gradient estimation methods:
  - Infinitesimal perturbation analysis (IPA)
  - Smoothed perturbation analysis (SPA)
  - Measure-valued differentiation (MVD)
  - ...

Other Continuous Optimization Methods



Metamodel-based optimization

- Run simulation at selected "design points" and fit (fuzzy) response surface
- Then optimize over surface
- Surface can be fit locally or globally
- Surface models include:
  - Polynomials ("response surface methdology")
  - Gaussian field models (stochastic kriging, moving least squares)

Other Continuous Optimization Methods

Sample Average Approximation (discussed previously)

- Goal: $\min_{\theta \in \Theta} f(\theta)$, where $f(\theta) = E[c(X, \theta)]$
  - c is a deterministic function
  - X is a random variable whose dist'n is independent of θ
- Generate $X_1, \ldots, X_n$ i.i.d. and set $f_n(\theta) = (1/n) \sum_{i=1}^{n} c(X_i, \theta)$
- Use deterministic optimization methods to solve $\min_{\theta \in \Theta} f_n(\theta)$
- f_n and f need some structure (convexity, smoothness)
- Can use delta method to get confidence intervals
- Cn combine with likelihood ratios
  - Use LR to convert cost from $E_\theta[c(X, \theta)]$ to $E[c(X, \theta)L]$
  - Then use SAA as described above

Selection of the Best

Goal

- Systems 1 through k have expected perf. measures $\mu_1 \le \mu_2 \le \cdots \le \mu_k$
- Choose system with smallest expected value

Dudewicz and Dalal (HW #7)

- With probability $\ge p$, will return system i^* s.t. $\mu_{i^*} \le \mu_1 + \delta$
- δ is indifference zone: max. diff. that you care about
- 2-stage procedure tries to minimize overall simulation effort

Many variants

- Adaptive (multistage) R&S
- Confidence intervals (comparison with the best)
- Pre-screening, common random numbers, ...

Dudewicz and Dalal Procedure

Assumes normally distributed observations (e.g., by CLT)

D&D algorithm

1. Simulate n_0 replications for each of systems $1, 2, \ldots, k$
2. $\bar{X}_i^{(1)} = \text{avg}(X_{i,1}, \ldots, X_{i,n_0})$ and $S_i^2 = \text{svar}(X_{i,1}, \ldots, X_{i,n_0})$
3. $N_i = \max\big(n_0 + 1, \lceil h^2 S_i^2/\delta^2 \rceil\big)$ = final # of reps for sys. i
4. Simulate $N_i - n_0$ reps of system i for $i = 1, 2, \ldots, k$
5. $\bar{X}_i^{(2)} = \text{avg}(X_{i,n_0+1}, X_{i,n_0+2}, \ldots, X_{i,N_i})$
6. $W_i = \frac{n_0}{N_i}\left\{1 + \sqrt{1 - \frac{N_i}{n_0}\left[1 - \frac{(N_i - n_0)\delta^2}{h^2 S_i^2}\right]}\right\}$
7. $\tilde{X}_i = W_i \bar{X}_i^{(1)} + (1 - W_i)\bar{X}_i^{(2)}$
8. Return system with smallest value of $\tilde{X}_i$

h is a constant that depends on k, p, and n_0 (Law Table 10.11)

Dudewicz and Dalal: Proof Sketch

- Definition of W_i and N_i ensures that $T_i = \frac{\tilde{X}_i - \mu_i}{\delta/h}$ has t_{n_0-1} dist'n and T_i's are independent
- Assume that $\mu_2 - \mu_1 \ge \delta$ (hence $\mu_j - \mu_1 \ge \delta$ for $j \ge 2$)

$$
\begin{aligned}
P(CS) &= P\{\tilde{X}_1 < \tilde{X}_j \text{ for } j \ge 2\} \\
&= P\left\{\frac{\tilde{X}_1 - \mu_1}{\delta/h} + \frac{\mu_1}{\delta/h} \le \frac{\tilde{X}_j - \mu_j}{\delta/h} + \frac{\mu_j}{\delta/h} \text{ for } j \ge 2\right\} \\
&= P\left\{-T_j \le \frac{\mu_j - \mu_1}{\delta/h} - T_1 \text{ for } j \ge 2\right\} \\
&= \int_{-\infty}^{\infty} \prod_{j=2}^{k} F_{n_0}\left(\frac{\mu_j - \mu_1}{\delta/h} - t\right) f_{n_0}(t)\, dt \\
&\ge \int_{-\infty}^{\infty} [F_{n_0}(h - t)]^{k-1} f_{n_0}(t)\, dt \triangleq g_{n_0,k}(h)
\end{aligned}
$$

F_{n_0} is cdf of t_{n_0-1}
f_{n_0} is pdf of t_{n_0-1}

- Set $g_{n_0,k}(h) = p$ and solve for h

Subset Selection

$\mu_1 \le \mu_2 \le \cdots \le \mu_k$

Overview

- Goal: With probability $\ge p$, return a set I of size m that contains a system i^* s.t. $\mu_{i^*} \le \mu_1 + \delta$
- Usually requires many fewer rep's than selection of best (good for screening solution candidates)

Extended D&D Algorithm (next slide)

- Reduces to D&D algorithm when $m = 1$
- Proof is very similar to D&D

Many variants

- Ex: BNK algorithm where size of I is not specified
  - If size $= 1$ then you have the best
  - See Boesel et al. 2003 reference in Law bibliography
- Common random numbers, Bayesian procedures, ...

Subset Selection, Continued

Extended D&D algorithm

1. Simulate n_0 replications for each of systems $1, 2, \ldots, k$
2. $\bar{X}_i^{(1)} = \mathrm{avg}(X_{i,1}, \ldots, X_{i,n_0})$ and $S_i^2 = \mathrm{svar}(X_{i,1}, \ldots, X_{i,n_0})$
3. $N_i = \max\bigl(n_0 + 1, \lceil g^2 S_i^2/\delta^2 \rceil\bigr)$ = final # of reps for sys. i
4. Simulate $N_i - n_0$ reps of system i for $i = 1, 2, \ldots, k$
5. $\bar{X}_i^{(2)} = \mathrm{avg}(X_{i,n_0+1}, X_{i,n_0+2}, \ldots, X_{i,N_i})$
6. $W_i = \frac{n_0}{N_i}\left\{1 + \sqrt{1 - \frac{N_i}{n_0}\left[1 - \frac{(N_i - n_0)\delta^2}{g^2 S_i^2}\right]}\right\}$
7. $\tilde{X}_i = W_i \bar{X}_i^{(1)} + (1 - W_i)\bar{X}_i^{(2)}$
8. Return set I of systems with m smallest values of $\tilde{X}_i$

g is a constant that depends on k, p, n_0, and m
(Law Table 10.12)

Discrete Optimization

Setting: Large but finite set of alternatives Θ

- Global procedures: Simulate all $\theta \in \Theta$ to find global optimum
  - No finite stopping rule
  - Asymptotically simulates all $\theta \in \Theta$ infinitely many times
  - Asymptotic guarantee of finding the optimal solution wp1
  - Ex: stochastic ruler, stochastic branch and bound, R-BEESE, SMRAS
- Local procedures: Only finds local optimum
  - Only searches "promising" elements of Θ
  - Often searches in neighborhood of current optimal solution
  - Stopping rule, preferably followed by "cleanup" phase
    - Goal: Minimum additional simulations for statistical guarantee
    - Subset selection + R&S
  - Ex: COMPASS, AHA[1]

[1] J. Xu, B. L. Nelson, and L. J. Hong, "An Adaptive Hyperbox Algorithm for High-Dimensional Discrete Optimization via Simulation Problems", *INFORMS J. Comput.* 24(1), 2013, 133–146.

Discrete Optimization, Continued

Key ingredients

- Estimation set $E_n \subseteq \Theta$: Solutions to simulate at nth step
- Memory set M_n: Information about systems simulated so far
- Sampling distribution $F(\cdot|M_n)$: Used to choose E_n
- Sim. allocation rule $\mathrm{SAR}_n(E_n|M_n)$: # reps for each $\theta \in E_n$
- Stopping rule to decide if we are done

Generic Local Procedure

1. Initialization: $M_0 \leftarrow \emptyset$, $n = 0$, $\theta_0^* =$ initial feasible solution
2. Sampling: Sample from Θ using $F(\cdot|M_n)$ to form set E_n
3. Estimation: Apply $\mathrm{SAR}_n(E_n|M_n)$ to elements $\theta \in E_n$
4. Iteration: Update estimator $\hat{f}(\theta)$ for $\theta \in E_n$ and choose θ_{n+1}^* as solution wth best $\hat{f}$ value.
5. Stop at θ_{n+1}^*?
6. If yes, (run cleanup phase and) return answer, else set $n \leftarrow n+1$ and go to Step 2.

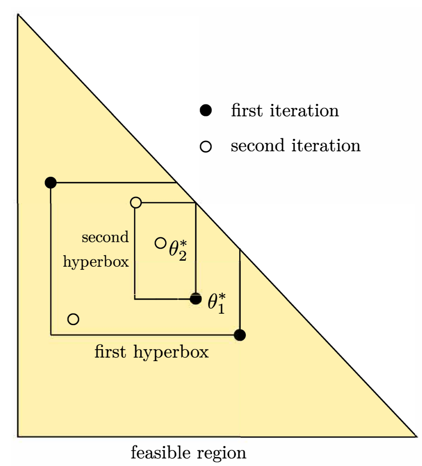
Example of Local Procedure: AHA

A particular instantiation of generic local algorithm

- Memory: $M_n =$ all $\left(\theta, \hat{f}(\theta)\right)$ pairs though nth iteration
- Sampling: $F(\cdot|M_n)$ samples m feasible points from hyperbox around current best solution θ_{n-1}^* (next slide)
- Estimation set: $E_n =$ best solution θ_{n-1}^* plus sampled points
- Allocation rule: Simulate at all points in E_n with cumulative replications given by $R_n(\theta) = \min\{5, 5(\log n)^{1.01}\}$
- Stopping rule: Test the hypothesis that $f(\theta_n^*)$ is minimum in neighborhood, if so, run cleanup[1]

[1] J. Boesel, B.L. Nelson, and S. Kim, "Using ranking and selection to "clean up" after simulation optimization", *Oper. Res.* 51(5), 2003, 814–825.

AHA Scenario Sampling



Commercial Solvers

Based on Robust metaheuristics

- Designed for deterministic problems
- Don't impose strong structural requirements
- Somewhat tolerant of some sampling variability
- No probabilistic guarantees provided
- Ex: Genetic algorithms, simulated annealing, tabu search

Example commercial solvers

- OptQuest (for Simul8, Arena, Simio, AnyLogic, etc.)
- Witness
- ExtendSim Evolutionary Optimizer
- RiskOptimizer
- AutoStat for AutoMod

Commercial Solvers, Continued

Increasing the effectiveness of commercial solvers

- Preliminary experiment to control sampling variability
  - Usually # of replications increases close to optimum
  - Some commercial packages used fixed # reps throughout
  - Always use "adaptive" option if available
  - Else simulate at a variety of θ values, estimate $n = \#$ reps needed to statistically distinguish between worst and best solutions, then use n as a minimal value
- Restart at a number of different initial θ values
- Run a cleanup phase