NORTHWESTERN UNIVERSITY

Search and Selection for Large-Scale Stochastic Optimization

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the Degree

DOCTOR OF PHILOSOPHY

Field of Industrial Engineering and Management Sciences

By

Justin Boesel

EVANSTON, ILLINOIS

June 1999

To Molly, for her support.

Acknowledgements

First and foremost, I would like to thank my advisor, Barry L. Nelson, for his clear mind, hard work, and friendship. I would also like to thank the following people for their contributions:

> Dr. Nobuaki Ishii of the JGC Corporation (Japan) for his initiative and support, and for his contributions to the software chapter,
>
> Seong-Hee Kim of Northwestern University for her work on the ranking-and-selection experiments, and
>
> Professor Y. Tong of Georgia Tech for proving a key result related to our proof of the validity of Rinott's procedure under unequal initial sample sizes.

Financial support for this research was provided by National Science Foundation Grant numbers DMI-9622065 and DMI-9622269, and grants from Systems Modeling Corporation, Symix Corporation/Pritsker Division, and JGC Corporation (Japan).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation studies optimization of unstructured stochastic simulation problems. An unstructured optimization problem is one in which there are no known properties, such as convexity or continuity, that can be used to find an optimal solution. Since our problems are stochastic, the optimal solution is defined to be the one that delivers either maximum or minimum expected (long-run average) performance.

Computer simulation is widely used to model proposed designs, or improve existing designs, of industrial, retail, computer, telecommunications and service systems. In most of these complex systems, the output performance measures, such as work in progress, time in queue, or average throughput, are stochastic in nature. This stochastic variation makes it more difficult to compare alternative designs. Simulation is an *evaluative* tool; it does not naturally suggest or generate new solutions.

Optimization, on the other hand, is used to seek out the best design from a possibly large number of different combinations of controllable decision variables. Most traditional optimization methods, such as linear or integer programming, assume deterministic (as opposed to stochastic) outputs, as well as an underlying mathematical model that can be exploited. In a computer simulation of a complex system, however, neither of these assumptions typically hold. Researchers have succeeding in adapting a few traditional optimization techniques, such as gradient search, to a simulation setting, but in most cases the simulation model must adhere to a restrictive set of assumptions; in other words, the problem must have a particular structure, such as convexity or continuity.

We have developed theory that allows a heuristic search procedure—a deterministic optimization procedure that does not rely heavily on problem structure—to function in a stochastic environment. We also developed theory that allows us to select the *true* best system from among a large number of stochastic systems. This theory is used *after* the search procedure has run its course. We implemented both the search and selection theory into software.

This dissertation is motivated in large part by a request from JGC, a Japanese construction management firm that uses simulation to evaluate the designs of a diverse set of projects such as automobile manufacturing plants, pharmaceutical plants, and port facilities. JGC approached Northwestern University looking for a "scenario-generation" scheme that would automatically change the values of decision variables in simulation models designed by JGC engineers. For example, in a simulation of an automobile manufacturing plant, the design variables might be the number of drills at a particular workstation, the speed of those drills, and the speed of a conveyor belt. While JGC wanted to automate the generation of scenarios, their primary concern was error control. They wanted to make statistically valid statements about the scenarios that they selected. Although commercial simulation-optimization products are available, they make no statistical guarantees, either on the search or on the final results. The methods we developed provide statistical guarantees, and will make no, or only very mild, *a priori* assumptions about the structure of the problem's response surface. Thus, they are applicable to a broad class of problems, including problems with continuous, integer, and categorical decision variables. This generality is of great practical value, because even though an analyst can often exploit strong problem structure, the difficulty typically lies in finding and recognizing the structure. In fact, there is a relatively large amount of idle cpu time available in most workplaces, while skilled analyst time is in relatively short supply.

This dissertation has three parts. Chapter 2 explores the issues involved in developing simulation-optimization software; this software incorporates the theory derived in Chapters 3 and 4. Chapter 3 derives the theory that allows a heuristic search, specifically a genetic algorithm, to function effectively in a stochastic environment, while Chapter 4 derives procedures for efficiently selecting the best of a large number of stochastic systems. The next three sections of this chapter summarize the contributions of Chapters 2–4, respectively.

1.1 Software

Most applied simulation-optimization approaches use heuristic optimization procedures—such as genetic algorithms, tabu search, or neural nets—that were designed for use in a deterministic setting. Typically, the number of replications taken at each solution is pre-set by the user. While these approaches often find good solutions quickly, they may also devolve into a random search if the level of output variability is high or the user has set the number of replications per solution too low. On the other hand, these procedures may be overly conservative and slow if the user sets the number of replications per solution too high or the level of output variability is low. Furthermore, these approaches do not provide statistical guarantees about the "goodness" of the observed best solution.

Our software works in three phases:

1. **Problem Definition:** In this phase, the user defines the problem by providing the simulation model to evaluate, as well as the output measure to be optimized and the objective (maximization or minimization). Furthermore, the user must define the design variables within the model, and specify the allowable range and increment of each variable. Finally, the user must choose the amount of time available, the required confidence level, $1-\alpha$, and the smallest practically significant difference worth detecting, δ.

2. **Solution Generation and Search:** In this phase, new solutions are generated by a search procedure or other means. Our software has several "non-search" solution generators, including one that allows the user to input solutions manually, so that if the user has an idea as to which solutions might be good, then those solutions can be evaluated early in

the process. Like some commercially available packages, our software employs a heuristic search procedure (genetic algorithm) to seek out better solutions. Unlike any commercially available package, our algorithm uses variance information to adjust the number of replications taken at each solution during the search. This provides adequate (but not excessive) error control during the search, keeping it from blindly devolving into a random search. This control is obtained using results from Chapter 3.

3. **Selection of the Best:** After the solution-generation phase has finished, the solutions are passed to a procedure that provides a statistical guarantee as to which of the generated solutions is the best. Our software uses one of the procedures developed in Chapter 4.

Because of the difficulties mentioned above (lack of known response properties, stochastic response and limited time), our algorithm does not guarantee that it returns the best solution over the entire solution space, just over the solutions visited by the search procedure. In other words, we are not willing to make statements about unvisited solutions. Of course, if the solution generators exhaust the solution space and visit all feasible solutions, then the statistical guarantee applies to the entire solution space. Because exhaustion is possible in smaller problems, we designed the software to explicitly exhaust the solution space if the user has provided adequate time.

The key contribution of Chapter 2 is to propose the decomposition of error control between search and selection, and to provide a proof-of-concept for the theory derived for controlling these errors in the chapters that follow.

1.2 Search

While optimizing an unstructured problem in a deterministic environment is difficult, working in a stochastic setting compounds the difficulty. Genetic algorithms (GAs)—probabilistic search and optimization schemes that apply the ideas of Darwinian evolution (survival of the fittest, reproduction, and mutation) to find good solutions to difficult deterministic problems—have shown some promise in tackling the problem of unstructured simulation optimization. Each iteration of a genetic algorithm maintains a "population" of solutions. Superior or "fitter" solutions are assigned a higher probability of "selection," which gives them an opportunity to survive and recombine or "mate" with other survivors, passing their characteristics onto subsequent generations. These offspring, which combine their parents' characteristics, have some probability of undergoing a random “mutation.”

Several aspects of GAs make them attractive candidates for optimizing stochastic systems evaluated via discrete-event simulation:

- A GA is a direct or “black box” optimization method, requiring only evaluation information, rather than indirect information such as a gradient (which is often impossible to obtain in a simulation setting);
- A GA can handle integer, continuous, and even categorical variables;
- A GA can work well on many different “landscapes” or response surfaces, even when the structure of those response surfaces is unknown; and
- A GA is relatively robust to low levels of stochastic variation.

Unfortunately, when stochastic variation becomes high, a GA can deteriorate into an aimless, almost random, search. We present theory that helps to explain why GAs perform well under low levels of stochastic variation and we describe how to enable GAs to perform well under higher levels of variation. We focus our attention upon GA selection mechanisms because these are the only GA operators that perform differently in deterministic and stochastic environments. Other GA operators, such as mutation and recombination, will be unaffected by stochastic variation.

Our key insight is that in a stochastic environment we want to avoid mimicking the *mechanics* of GA selection schemes, which were designed for use in a deterministic environment, while still achieving the primary *goal* of these schemes, which is to assign higher selection probabilities to superior solutions. To accomplish this we attain a type of stochastic equivalence between a GA applied to a stochastic problem and the same GA applied to a corresponding deterministic problem. This equivalence can be obtained with far fewer replications per solution than would be required to fully rank the solutions with a high degree of confidence, by instead ranking at least a minimum number of *groups* of solutions correctly, and adjusting the selection probabilities.

1.3 Selection

Because of stochastic variation, the system with the best sample mean at the end of the search procedure may not coincide with the true best system encountered during the search. We develop statistical procedures that return the best system encountered by the search (or one near the best) with a pre-specified probability. We approach this problem using combinations of statistical subset and indifference-zone selection procedures. The subset procedures use only the data already collected by the search procedure to screen out the obviously inferior systems, while the indifference-zone selection procedures require additional simulation effort to distinguish the best from the less obviously inferior systems.

The extensions we obtain to existing subset and indifference-zone procedures focus on the problems and opportunities that may arise at the end of stochastic search: unequal numbers of replications across solutions; a very large number of potential solutions with widely differing performance; and the possibility to reallocate sampling effort based on the results already obtained.

Chapter 2

A Framework for Simulation-Optimization Software

2.1 The Problem

Like many organizations, JGC, a Japanese construction management company, uses simulation to evaluate and compare proposed designs for facilities such as pharmaceutical plants, chemical refineries and automobile manufacturing plants. Simulation is especially important to firms like JGC who must propose designs that satisfy client requirements within a limited time and budget. In an effort to optimize, or at least improve, the design of facilities, these firms use simulation to evaluate and compare alternative designs.

At JGC, many different simulation studies are conducted simultaneously. Some examples of such studies are described below:

Design of materials handling system (MHS) in a pharmaceuticals plant: The MHS consists of a large Automated Storage and Retrieval System, Automated Guided Vehicles (AGVs), AGV stations, lifters and conveyors. The design variables include the number of AGVs, load-per-AGV and the AGV routings. The performance criterion is based upon AGV utilization, waiting time of each transportation order, and the overall investment cost.

Design of liquefied natural gas (LNG) tanker transportation system: An LNG transportation system consists of LNG tankers, LNG loading and unloading facilities and LNG storage facilities. Design variables include LNG tanker size, number of LNG tankers required, capacity of loading and unloading facilities (including the number of jetties) and LNG tank capacity in the shipping and receiving sites. Simulation is used to evaluate a cost-based performance criterion for each alternative design.

Buffer allocation in an automobile engine assembly line: In an auto assembly line, a larger buffer (queue) between work stations can increase workstation utilization, but may also drive up space requirements. Simulation evaluates each alternative design using a cost function that weighs these competing factors.

Despite the availability of simulation modeling software—which allows engineers to build models quickly—and powerful computers—which allow even large complicated models to run relatively quickly—optimization via simulation is still a time- and labor-consuming process. For complicated facilities, like the ones described above, an analyst may encounter a large number of design alternatives. Furthermore, an analyst may spend a great deal of time on the analysis of simulation outputs, such as ensuring that simulations have reached a steady state and determining whether observed differences between systems are statistically significant. As a result, it usually takes several months to complete a simulation study. This time is critical for engineering firms because they must propose a good design to their clients at an early phase of each project.

JGC wanted to reduce the amount of time required to complete a simulation study and wanted its simulation analysts to spend more time on model development, rather than trial-and-error search. JGC approached Northwestern University asking for research and development of simulation-optimization software that could provide good results on a broad range of problems in a reasonable amount of computer time, and could provide statistical guarantees on those results.

From an optimization viewpoint, projects like the ones described above present several difficulties. First, the optimization approach needs to handle simulation models that combine integer decision variables (such as the number of AGVs), continuous decision variables (such as LNG tank capacity) and categorical decision variables (such as AGV routes or scheduling rules). This means that some simulation-optimization techniques designed for problems with just one type of variable, such as gradient-search methods, can not be applied.

Second, the response properties of the problems are unknown. That is, no exploitable structure, such as convexity or even continuity, is known to exist. Not surprisingly, this makes the task of finding the best system much more difficult, because it prevents one from saying or inferring anything about solutions that are not explicitly evaluated.

Third, the responses are stochastic, so one needs multiple (and perhaps very many) replications to get reliable information on a single solution.[1]

Broadly speaking, two general approaches have been developed for simulation-optimization problems that include discrete variables. The first, which exists largely in the academic research community, guarantees convergence to a global optimum as the number of iterations approaches infinity. These procedures typically do not seek rapid improvement in the early stages of the algorithm, and provide no statistical guarantees for a finite number of replications. See, for instance, Andradóttir (1998) for an overview of these techniques. This approach is not widely applied in industrial settings.

Most applied simulation-optimization approaches use heuristic optimization procedures—such as genetic algorithms, tabu search, or neural nets—that were designed for use in a deterministic setting. Typically, the number of replications taken at each solution is pre-set by the user. While these approaches often find good solutions quickly, they may also devolve into a random search if the level of output variability is high or the user has set the number of replications too low. On the other hand, these procedures may be overly conservative and slow if the

[1]Throughout the paper we use the terms "system" and "solution" interchangeably since, in our context, the solution to the simulation-optimization problem represents a distinct system design.

user sets the number of replications too high or the level of output variability is low. Furthermore, these approaches do not provide statistical guarantees about the "goodness" of the observed best solution.

Our approach works in three phases:

1. **Problem Definition:** In this phase, the user defines the problem by providing the simulation model to evaluate, as well as the output measure to be optimized and the objective (maximization or minimization). Furthermore, the user must define the design variables within the model, and specify the allowable range and increment of each variable. Finally, the user must choose the amount of clock time available, the required confidence level, $1-\alpha$, and the smallest practically significant difference worth detecting, δ.

2. **Solution Generation and Search:** In this phase, new solutions are generated by a search procedure or other means. Our software has several "non-search" solution generators, including one that allows the user to input solutions manually, so that if the user has an idea as to which solutions might be good, then those solutions can be evaluated early in the process. Like some commercially available packages, our software em-

ploys a heuristic search procedure (genetic algorithm) to seek out better solutions. Unlike any commercially available package, our algorithm uses variance information to adjust the number of replications taken at each solution during the search. This provides adequate (but not excessive) error control during the search, keeping it from blindly devolving into a random search. These techniques are developed in Boesel and Nelson (1999).

3. **Selection of the Best:** After the solution-generation phase has finished, the solutions are passed to a procedure that provides a statistical guarantee as to which of the generated solutions is the best. Some additional simulation may be required to achieve the guarantee. Our software uses one of the procedures developed by Boesel, Nelson and Kim (1999) for this purpose.

Because of the difficulties mentioned above (lack of known response properties, stochastic response and limited time), our algorithm does not guarantee that it returns the best solution over the entire solution space, just over the solutions visited by the search procedure. In other words, we are not willing to make statements about unvisited solutions. Of course, if the solution generators exhaust the solution space and visit all feasible solutions, then the statistical guarantee applies to the entire solution space. Because exhaustion is possible in smaller problems, we designed the software to explicitly exhaust the solution space if the user has provided adequate time.

2.2 Software

The software Northwestern delivered to JGC has five interrelated components that are described below. The flow of information among these five components in each of the three phases (Problem Definition, Solution Generation and Selection of the Best) is diagrammed in Figure 2.1.

1. **Interface:** The interface allows the user to define the simulation-optimization problem (amount of time available, ranges of the decision variables, etc.) and lets the user add promising solutions directly into the search. As Figure 2.1 shows, some of this information goes exclusively to the solution generators, some information goes exclusively to the Selection Procedure, and some goes to both.

2. **Solution Generators:** The software currently has four methods for generating new solutions, which are described below. All of the solutions are evaluated by the simulator and passed on to the database, as shown in

Figure 2.1.

User-Defined Solutions: Because the engineer developing the simulation model usually has good ideas about what solutions might be promising, the software interface allows the user to input these solutions at the beginning of the algorithm.

Extreme Point Finder: Because good solutions often lie at the extreme points of the solution space, our software generates all of the extreme (vertex) point solutions at the beginning of the algorithm. These extreme points may later be fed into the genetic algorithm, ensuring that it provides an adequately broad search of the solution space.

Exhauster: On smaller problems, it often makes sense to simply evaluate all possible solutions. Our software explicitly exhausts the solution space if there is adequate time. The software decides whether there is adequate time by observing the average time to evaluate a solution. Exhaustion is desirable because the statistical guarantee returned under exhaustion covers the entire solution space.

Genetic Algorithm (GA): A genetic algorithm is a "population-based" search algorithm, as opposed to a "point-based" algorithm, meaning that in each iteration it simultaneously keeps a number of solutions active rather than just one. A GA uses the Darwinian concepts of "natural selection" and "survival of the fittest" to produce improved solutions. Essentially, a GA assigns better solutions higher probabilities of survival, where survival means the ability to pass on characteristics to future populations. These characteristics are passed on by combining or "mating" parent solutions to form a new population of child solutions. Our genetic algorithm is initialized by filling the first population with the extreme points and the best of the user-input solutions.

3. **Simulator:** Central to the software is a simulation package which evaluates each solution produced by the solution generators. Our software employs the AweSim! simulation package (Symix Corporation/Pritsker Division), which is used by JGC. The user develops a simulation model as usual, independent of our software, and performs a few modifications to make the decision parameters into variables. The user then defines the

objective function, which can be any function of any combination of simulation outputs, in a C++ user insert, which also provides the "hooks" that allow our software to control AweSim!

4. **Selection Procedure:** After the solution-generation phase has concluded, all solutions are sent from the database (described below) to the selection procedure, which provides the statistical guarantee that makes our software unique among simulation-optimization packages. Although several different selection procedures are available, our software employs a sort-screen-and-select procedure that sorts systems by sample mean and requests additional replications on the observed best system. Each subsequent system is then either screened out by an earlier system, or it also receives additional replications. Under certain conditions, the procedure guarantees that the solution with the best sample mean is the best or within δ of the true best solution visited by the search with probability $\geq 1 - \alpha$.

5. **Database:** Because the solution generators may produce a large number of alternatives, each of which has unique parameter settings and output data, we maintain a database to record this information. Each unique

solution generated has a record in the database. Furthermore, because a GA tends to generate the same solution more than once, and because we want to avoid wasting simulation effort on repeat evaluations, the GA first checks the database to see if a solution has been evaluated previously. If not, the GA requests the information from the simulator. The simulator writes all output information to the database, while the selection procedure writes status information (such as "screened" or "unscreened") to the database. These information flows are diagrammed in Figure 2.1. The database also enables the user to analyze solution output after the simulation-optimization run has concluded.

Our software, dubbed `Scenario Seeker`, runs under Windows 95, 98 and NT. We developed the interface in Visual Basic, while the solution generators and the statistical procedures were written in C++. We used GALib—a C++ library of genetic algorithms written by Matthew Wall at M.I.T.—to develop our GA.

`Scenario Seeker` employs an Access database to maintain all simulation results.

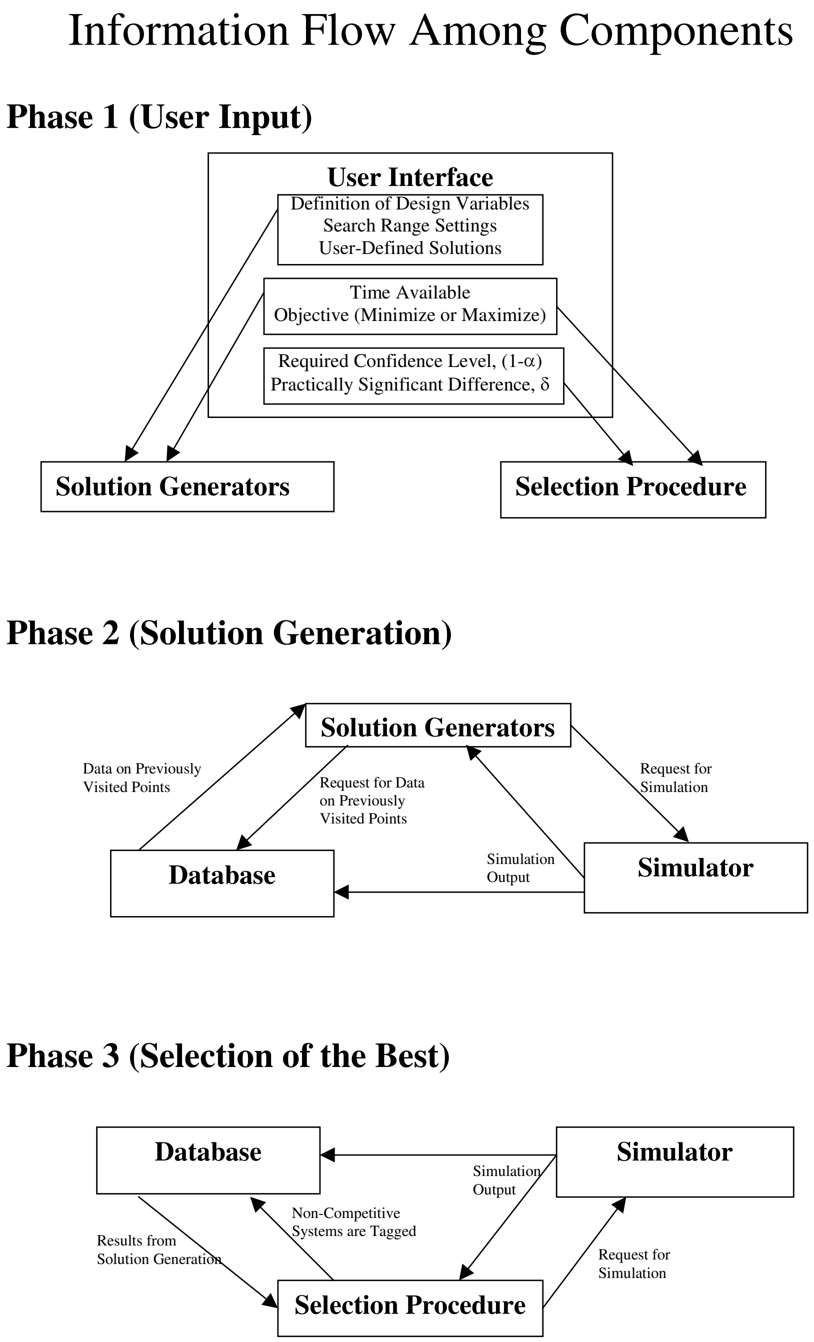

Figure 2.1: The three phases of simulation optimization.

2.3 Error Control in the Search

A GA is a probabilistic search and optimization scheme that applies the ideas of Darwinian evolution (survival of the fittest, reproduction, and mutation) to difficult optimization problems. Essentially, a genetic algorithm starts with an initial population of m distinct solutions, selects the better ones for "breeding," creates children by "mating" parent solutions, "mutates" a few of the children, and starts over again by evaluating the new generation of m solutions. In a deterministic setting, a genetic algorithm determines the "fitness" of each individual solution by evaluating it via an objective function. Loosely speaking, the better/fitter solutions are assigned a higher probability of being selected for mating.

We employed a real-valued GA in which child solutions were created via two-parent recombination by taking a uniformly distributed, convex combination of each parent's decision-variable value, for continuous and integer variables, or making a random choice between the parent solutions for categorical variables. Independent Gaussian mutation was used on all decision variables. The selection probability mechanism is described more fully below. The population size was held constant at $m = 30$ throughout the run.

To adapt a deterministic GA for use in a stochastic setting, we focused our attention upon the selection probability assignment mechanism, which is the algorithm's "guidance system." Because this mechanism depends upon objective function evaluations, this is the only GA component that could be badly misled in a stochastic setting where we have an uncertain and expensive evaluation method (simulation). The other operators, such as mating (also called crossover) and mutation do not depend directly upon fitness evaluations.

In a deterministic setting, there are several ways to translate a solution's objective function value into a selection probability. In a *ranking* scheme, a solution's selection probability is assigned strictly by rank in the current generation. In other words, it does not matter if the best system is 100 or 0.01 units better than the next best, the assigned probability will be the same. By contrast, in schemes such as proportional selection, if the best dominated the rest, it would receive a higher selection probability than if it barely beat the rest (Bäck, 1996).

In a genetic algorithm, the selection probabilities *favor* the better solutions while still giving the poorer solutions some chance of survival; this keeps the search broad and robust. The mapping of function evaluations to the selection probabilities, however, can take several different forms and still provide good search performance. These mappings, then, are not set in stone.

In a deterministic setting, we can easily and accurately evaluate the objective function, so we use these exact evaluations to determine selection probabilities. In a stochastic setting, however, it is not possible to conclusively rank any population of solutions without expending excessive simulation effort (number of replications). Because the success of an GA does not depend upon a strict mapping of solutions to probabilities, it is not worthwhile to expend a great deal of effort trying to get highly accurate estimates if less accurate estimates will suffice. In other words, we have an uncertain evaluation method (simulation), so we want to take advantage of the fact that the GA seems relatively robust to a range of selection probability mappings.

Therefore, our approach is to expend enough simulation effort to achieve *stochastic equivalence* for some important characteristic of our GA applied to a stochastic problem and a GA applied to a corresponding deterministic problem. For instance, we can guarantee the same selective pressure—measured in terms of the expected number of copies of the best solution in the current population that goes on to the mating pool for the next generation—in a stochastic setting as would have been attained in a deterministic setting. In many instances, such a guarantee can be delivered at a reasonable sampling cost, as we show below.

In a stochastic setting, it typically requires less effort (fewer simulation replications) to form and rank *groups* of solutions than to perform a comprehensive ranking of all solutions individually. For instance, we may be reasonably confident that a group of $g < m$ solutions contains the best solution, and that each member of that group is superior to each non-member, even though we may be uncertain about the within-group ordering of the g solutions. Below, we show how to ensure that the expected number of copies of the best solution placed in the mating pool is the same under a group-ranking procedure (which typically requires very little sampling effort) as under a comprehensive ranking procedure (which may require much more effort).

To simplify the analysis, we use anti-ranks, instead of ranks, so the best solution has an index m rather than 1. Using anti-ranks and q-tournament selection (Bäck, 1996), solution i in the current generation is assigned a selection probability of

$$p_i = \frac{i^q - (i-1)^q}{m^q} \tag{2.1}$$

where $2 \leq q \leq m$ is a parameter that controls the "pressure" on good solutions (larger q implies more pressure which means highly ranked solutions are assigned relatively larger values of p_i). Under this scheme, the sum of the selection probabilities for a group of size g starting with anti-rank j can be expressed as

$$\begin{aligned}\sum_{i=j}^{j+g-1} p_i &= \sum_{i=j}^{j+g-1} \frac{1}{m^q}\left(i^q - (i-1)^q\right) \\ &= \frac{(j+g-1)^q - (j-1)^q}{m^q}.\end{aligned}$$

In the best group, j begins with $m-g+1$. As a result, the sum of the selection probabilities of the best group is

$$\sum_{i=m-g+1}^{m} p_i = \frac{m^q - (m-g)^q}{m^q}. \tag{2.2}$$

In a deterministic q-tournament, the selection probability of the single best solution in the current generation is

$$p_m = \frac{m^q - (m-1)^q}{m^q}. \tag{2.3}$$

Because the mating pool is formed by taking m independent random draws with replacement from the current population, the expected number of copies of the best solution in the mating pool is mp_m.

Suppose that we want to assign each member of the best group a selection probability so that the expected number of copies of the true best solution in the mating pool is *still* mp_m. Because we are uncertain as to which of the g group members is the true best, we assign all group members the same selection probability, $p_{Best\ g}$. Let q' be the pressure parameter used to determine the selection probabilities under grouping. As a result, each member of the best group is assigned its group's average selection probability

$$p_{Best\ g} = \frac{m^{q'} - (m-g)^{q'}}{g m^{q'}}$$

which is simply the right-hand side of (2.2) divided by g, the number of group members.

To get the same expected number of copies of the true best in the mating pool in a stochastic (and grouped) environment as in a deterministic (and ungrouped) environment, we give each member of the best group the same selection probability as the best solution would receive in a deterministic environment, that is $p_{Best\ g} = p_m$. To do this, we set q' so that

$$\frac{m^{q'} - (m-g)^{q'}}{g m^{q'}} = \frac{m^q - (m-1)^q}{m^q}$$

implying a revised pressure parameter

$$q' = \frac{\ln\left(1 - g\left(1 - \left(\frac{m-1}{m}\right)^q\right)\right)}{\ln\left(\frac{m-g}{m}\right)}. \tag{2.4}$$

To implement (2.4), we use a statistical grouping procedure based on Calinski and Corsten (1985) to divide the solutions into c groups where group h has g_h different solutions and $g_1 + \cdots + g_c = m$, the population size. If we let G_h be the set of the indices of the solutions in group h, then

$$
\begin{aligned}
G_1 &= \{1, 2, \ldots, g_1\} \qquad \text{(worst group)} \\
G_2 &= \{(g_1 + 1), \ldots, (g_1 + g_2)\} \\
&\vdots \\
G_c &= \{(g_1 + g_2 + \cdots + g_{c-1} + 1), \ldots, (g_1 + g_2 + \cdots + g_{c-1} + g_c)\} \qquad \text{(best group).}
\end{aligned}
$$

If g_c, the size of the group containing the best system, is greater than $1/p_m$, then it is not possible to match the selection pressures, because $p_m g_c > 1$, and the sum of all selection probabilities must be equal to one. In this case, one would have to go back for more data (that is, perform more simulation replications) to reduce g_c.

Once g_c is small enough, q' can be determined according to equation (2.4), letting $g = g_c$. To find the individual selection probabilities, first find each group's total selection probability, then divide by the group's size to assign each solution its group probability. If we let $j = g_1 + \cdots + g_{h-1} + 1$ be the index of the worst system in group h, the total selection probability for group h is given by the expression

$$
\sum_{i=j}^{j+g_h-1} p_i = \frac{(j + g_h - 1)^{q'} - (j-1)^{q'}}{m^{q'}}
$$

$$= \frac{(g_1 + \cdots + g_h)^{q'} - (g_1 + \cdots + g_{h-1})^{q'}}{m^{q'}}$$

$$= \frac{\left(\sum_{\ell=1}^{h} g_\ell\right)^{q'} - \left(\sum_{\ell=1}^{h-1} g_\ell\right)^{q'}}{m^{q'}}.$$

As a result, each individual solution i in group h is assigned the average of the group's probability,

$$p_{ih} = \frac{\left(\sum_{\ell=1}^{h} g_\ell\right)^{q'} - \left(\sum_{\ell=1}^{h-1} g_\ell\right)^{q'}}{g_h m^{q'}} \tag{2.5}$$

where $h = 1, 2, \ldots, c$ and $\sum_{\ell=1}^{0} \equiv 0$. This concept of stochastic equivalence based on a statistical grouping procedure was implemented in the search portion of `Scenario Seeker`.

Although stochastic equivalence, in terms of the expected number of copies of the best solution in the mating pool, is an effective way to insure progress of the GA search, it has a few drawbacks. For instance, solutions not in the group containing the best may receive selection probabilities much lower than those they would have received in a deterministic environment, and the non-best solutions grouped with the true best solution may receive much higher selection probabilities. A more comprehensive measure of "stochastic equivalence," which ameliorates some of these drawbacks, is described in Boesel and Nelson (1999)

2.4 Error Control in Selection after Search

At the conclusion of the search phase, the GA has explored some portion of the decision space, uncovering good solutions and (quite likely) many poor solutions as well. We therefore turn our attention to separating those solutions into the best, near best and inferior solutions. Since we apply only enough error control in the search phase to ensure that the search makes progress, it is quite likely that there is too much sampling error in the performance estimates to make these finer distinctions.

Several different procedures for finding the best among a large number of simulated systems are developed in Nelson et al. (1998) and Boesel, Nelson and Kim (1999). These procedures guarantee, with probability $\geq 1 - \alpha$, that the selected solution is the true best *of all the solutions visited by the search* or is within some user-defined distance, δ, of the true best. First, these procedures perform a *subset selection*, using the sample data gathered during the search phase to eliminate or *screen out* clearly inferior systems, then they perform a *selection-of-the-best* procedure, which usually requires additional replications, to determine which of the remaining systems is the true best. The primary assumption behind these procedures is that the simulation output data are normally distributed; therefore, they are most appropriate when the performance measure is estimated by the sample average of a large number of more basic outputs.

A classical selection-of-the-best procedure, such as Rinott's procedure (Rinott, 1978), assumes that system means are arrayed in the *least favorable configuration* (LFC); that is, the configuration of means that is most likely to cause the procedure to fail (for further discussion, see Bechhofer, Santner, and Goldsman, 1995). Because Nature is rarely so malevolent, this LFC assumption can be grossly conservative. Therefore, combining a subset-selection (screening) procedure with a selection-of-the-best procedure is often more efficient than a selection-of-the-best procedure alone because clearly superior solutions will screen out inferior solutions. As a result, no additional sampling is required from the inferior solutions, greatly reducing the total simulation effort needed. This is especially important in our setting, where we encounter a large number of solutions, many of which are clearly inferior.

Further refinements to the combined screen-and-select procedure, described in Nelson et al. (1998) and Boesel, Nelson and Kim (1999), can boost efficiency dramatically. For instance, one such refinement sorts systems by their first-stage sample means and takes additional replications of the best observed system. The next-best observed system then faces screening by the best system; if it survives, additional replications are taken from it as well. Similarly, each subsequent system faces screening by all those which have preceded it.

To set up the procedure, let n_{0i} be the number of replications that system i received during the search, and let

$$W_{ij} = \left(\frac{t_i^2 S_{0i}^2}{\tilde{N}_i} + \frac{t_j^2 S_{0j}^2}{\tilde{N}_j} \right)^{1/2}$$

where

$$\tilde{N}_i = \begin{cases} n_{0i}, \text{if system } i \text{ has only received first-stage (search) sampling} \\ N_i, \text{if system } i \text{ has received second-stage (selection) sampling} \end{cases}$$

the constant t is a quantile from the t distribution and S_{0i}^2 is a sample variance (both defined more carefully below). We next present a step-by-step description of the sort-screen-and-select procedure. Assume that maximization is the goal, and that the search has uncovered k potential solutions. In the procedure a superscript (1) indicates a quantity computed from first-stage (search) data, while a superscript (2) indicates a quantity computed from all available data after first- and second-stage (selection) sampling.

1. Select overall confidence level $1-\alpha$ and indifference zone parameter δ. Set $t_i = t_{(1-\alpha/2)^{\frac{1}{k-1}}, n_{0i}-1}$ and $h = h((1-\alpha/2)^{1/(k-1)}, n_{min}, 2)$, where $n_{min} =$

$\min_i\{n_{0i}\}$, h is Rinott's constant and $t_{\beta,\nu}$ is the β quantile of the t distribution with ν degrees of freedom.

2. Let $I_0 = \emptyset$ and $J_0 = \emptyset$ (I is the set of solutions still in contention to be the best, while J is the set of solutions that have been eliminated from consideration).

3. Based on the data generated during the search phase, compute the first-stage sample means and variances, $\bar{X}_i^{(1)}$ and S_{0i}^2 and set $\tilde{N}_i = n_{0i}$ for all i.

4. Sort by sample mean, $\bar{X}_i^{(1)}$. Reset the indices to reflect the sorted order; that is, let $\bar{X}_i^{(1)} \geq \bar{X}_{i+1}^{(1)}$.

5. For each system, $i = 1, 2, \ldots, k$, do the following:

   If $\bar{X}_i^{(1)} \geq \bar{X}_j^{(1)} - W_{ij}, \forall j \in J_{i-1}$ and $\bar{X}_i^{(1)} \geq \bar{X}_j^{(2)} - W_{ij}, \forall j \in I_{i-1}$, then

   (a) compute the total sample size N_i based on Rinott's procedure; that is:

   $$N_i = \max\left\{n_{0i}, \left\lceil \left(\frac{hS_{0i}}{\delta}\right)^2 \right\rceil\right\};$$

   (b) Sample $N_i - n_{0i}$ additional replications from system i, and compute the overall sample mean $\bar{X}_i^{(2)}$;

(c) place this system into group I, so $I_i = I_{i-1} \cup i$. Let $J_i = J_{i-1}$.

Otherwise, place the system into J, so $J_i = J_{i-1} \cup i$. Let $I_i = I_{i-1}$.

6. Select as best the system $i \in I_k$ with the largest sample mean $\bar{X}_i^{(2)}$.

This provably valid sort-screen-and-select procedure uses the second-stage replications taken on the better systems to help in the screening process (see Boesel, Nelson and Kim 1999 for the proofs). These additional replications increase $\tilde{N}_i$, thus reducing W_{ij}, making it easier to eliminate inferior systems. Sorting by first-stage sample means ensures that the better systems—the ones most likely to screen out inferior systems—receive second-stage sampling early, increasing their ability to eliminate inferior systems. Eliminating inferior systems from contention before they can receive second-stage sampling saves simulation effort. A version of this procedure is implemented in `Scenario Seeker`.

2.5 Example

To test the simulation-optimization software, JGC prepared a simulation model of the automobile assembly line problem mentioned earlier. In brief, the problem is to determine the size of four buffers in an automobile engine assembly line, while accounting for the additional production that larger buffer sizes could allow over a two-week period. Our objective is to minimize the expected value of:

$$\$\frac{K + 1000 \times \text{Total Buffer Capacity}}{\text{Number of Engines Produced}}$$

where K is a constant representing costs not allowed to vary within our model. The capacities of the first and third buffers were allowed to range from 9 to 36, while the capacities of the second and fourth buffers were allowed to range from 15 to 60. As a result, $28^2 \times 46^2 = 1,658,944$ combinations were possible. The main user interface screen for `Scenario Seeker` (with settings for the example problem) is presented in Figure 2.2.

We allowed the program to run for 8 hours. Of the 68 combinations evaluated by the software, the best combination encountered was (9,15,9,15). This combination yielded an expected per-unit cost of \$164.69. We are guaranteed that this is the best solution visited by the search, or within $\delta = \$10$ of the best, with 90% confidence. The next best solution encountered, (9,24,9,24), had an estimated per-unit cost of \$175.94.

The best combination was the first solution encountered, so the extreme-point generator, rather than the search procedure, found the best solution.

To give a sense of the range of costs across solutions, the worst combination encountered was (36,60,36,60), which had an estimated cost of \$256.00.

2.6 Conclusions

Our approach to simulation-optimization has three distinct phases: the first phase allows the user to define the problem and input promising solutions; the second phase generates new solutions; and the third phase uses a statistical procedure to determine which solution is best. In the search segment of the solution-generation phase, we have incorporated adaptive error control so that our approach expends adequate—but not excessive—simulation effort to deal with sampling variability.

One of the strengths of our approach is its separation of the search from the selection procedure, which enables it to incorporate improved "component" procedures. For instance, if a problem-specific search procedure is known to work better than our general search procedure, it can be incorporated into our framework. Similarly, when better post-search "clean-up" procedures are developed, they too, can be incorporated. In fact, we believe that a better search procedure may make the post-search statistical selection procedure more

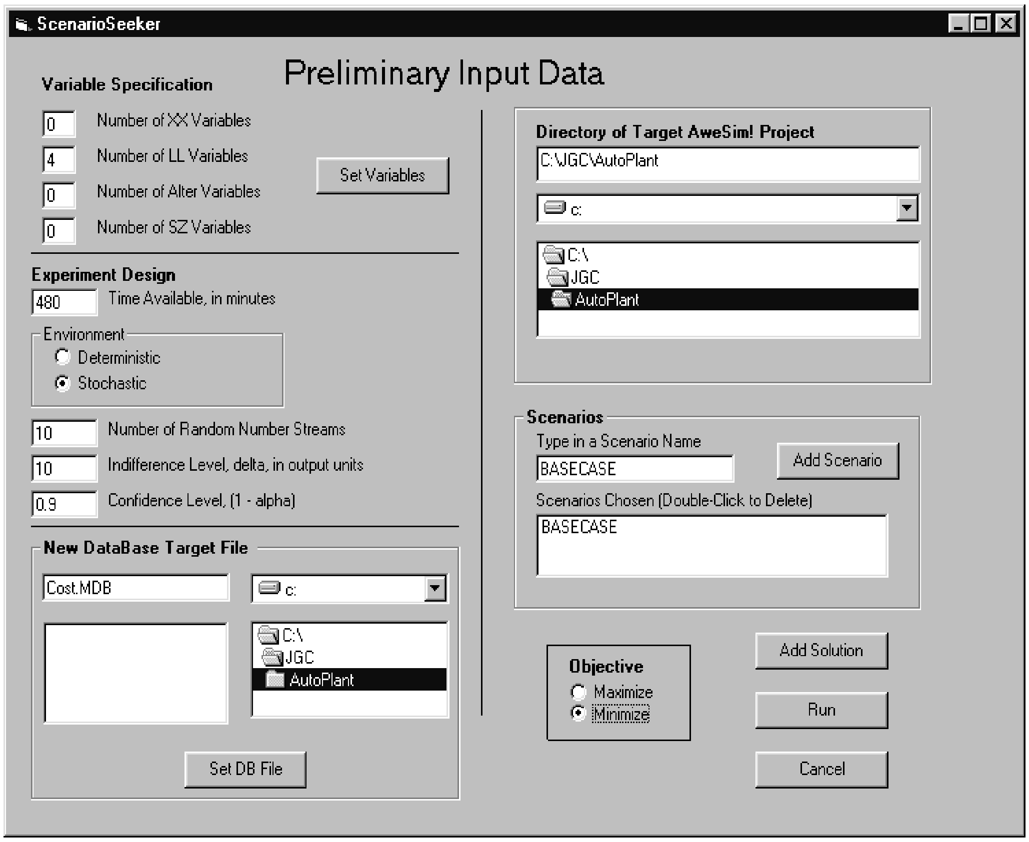

Figure 2.2: `Scenario Seeker`'s main user interface screen.

efficient by generating better solutions that screen out inferior solutions with less simulation effort.

Chapter 3

Designing Evolutionary Algorithms for Stochastic Optimization

3.1 Introduction

Discrete-event simulation is widely used to model and evaluate stochastic systems, especially when system performance is too complex to evaluate analytically. Unfortunately, optimizing systems via simulation is usually more difficult than optimizing systems whose performance is evaluated via simple functions because:

- Evaluation via discrete-event simulation typically requires more computational effort than does a simple function evaluation,
- Simulation of a stochastic system produces random output rather than a fixed, deterministic quantity,
- Indirect information, such as a gradient, is often impossible to obtain in a simulation setting, and
- Very little may be known about a system's response surface because of the complexity of systems being modeled via simulation.

Evolutionary algorithms (EAs)— probabilistic search and optimization schemes that apply the ideas of Darwinian evolution (survival of the fittest, reproduction, and mutation) to find good solutions to difficult deterministic problems — have shown some promise in tackling the problem of simulation optimization. Each iteration of an evolutionary algorithm maintains a "population" of solutions. Superior or "fitter" solutions are assigned a higher probability of "selection," which gives them an opportunity to survive and recombine or "mate" with other survivors, passing their characteristics onto subsequent generations. These offspring, which combine their parents' characteristics, have some probability of undergoing a random "mutation."

Several aspects of EAs make them attractive candidates for optimizing stochastic systems evaluated via discrete-event simulation (Hammel, 1997):

- An EA is a direct or "black-box" optimization method, requiring only evaluation information, rather than indirect information such as a gradient;
- An EA can handle integer, continuous, and even categorical variables;
- An EA can work well on many different "landscapes" or response surfaces, even when the structure of those response surfaces is unknown; and
- EAs are relatively robust to low levels of stochastic variation.

Unfortunately, when stochastic variation becomes high, an EA can deteriorate into an aimless, almost random, search. In this paper, we present theory that helps to explain why EAs perform well under low levels of stochastic variation and we describe how to enable EAs to perform well under higher levels of stochastic variation. We focus our attention upon EA selection mechanisms because these are the only EA operators that perform differently in deterministic and stochastic environments. Other EA operators, such as mutation and recombination, will be unaffected by stochastic variation.

The key idea of this paper is that in a stochastic environment we want to avoid mimicking the *mechanics* of EA selection schemes, which were designed for use in a deterministic environment, while still achieving the primary *goal* of these schemes, which is to assign higher selection probabilities to superior solutions.

The following section of this paper reviews other literature on the use of EAs in a stochastic setting, and relates these approaches to the approach presented here. Section 3 describes how different EA selection mechanisms perform in a deterministic setting. Section 4 then proposes several methods for adapting these selection mechanisms for use in a stochastic setting; the key ideas are to measure the difference between a selection scheme in a deterministic and stochastic setting and to devise ways to reduce that difference. Section 5 describes how statistical procedures can be chosen and integrated with some of the methods proposed in Section 4. The Appendix contains proofs of theorems and lemmas presented in the body of the text.

3.2 Related Approaches

There are many articles on the combination of simulation with Evolutionary Algorithms, and experimental results from these articles suggest that EAs are relatively robust to stochastic variation, especially low-level variation (Hammel, 1997). Most of these articles, however, tend to be of a practical, or applied nature; in fact, not much theoretical work has been done on the problem of using EAs in a stochastic setting. Some of this theoretical work seeks to explain how stochastic variation affects the progress or convergence properties of EAs over many generations. In these papers, however, little is done to control the variation; the EAs are run as though the sample mean (calculated from a fixed sample size, sometimes of size one) were a deterministic evaluation.

Fitzpatrick and Grefenstette (1988) recognize the trade-off between performing more replications at each solution and allowing more solutions to be explored under a fixed computer budget. Their experimental work suggests that while taking many samples at a single solution increases evaluation accuracy, it hampers overall genetic algorithm performance by reducing the number of solutions that can be visited.

In his doctoral dissertation, Miller (1997) develops a convergence model of genetic algorithms (GAs) operating in noisy conditions. This model assumes that one replication per solution is being performed, and that the GA has not been adapted for a stochastic environment. The model suggests that under these conditions noise slows convergence speed. Miller extends this convergence model and uses it to determine the best trade-off between performing additional replications at each solution (reducing the variance of each solution's sample mean) and allowing the GA to produce additional generations. Miller assumes that the number of replications per solution will remain constant within each population and over all generations.

Much of the work in the papers of Miller and of Fitzpatrick and Grefenstette relies upon the unproven "Building Block" hypothesis, first presented by Holland (1975). The hypothesis states that GAs find better solutions by implicitly evaluating "hyperplanes" or "building blocks," common components of different solutions. In the Miller and Fitzpatrick and Grefenstette papers, the number of different solutions evaluated is related to the number of replications performed at each solution through the idea that one replication taken at two different solutions is equivalent to performing two replications of the "building blocks" shared by those solutions. Fitzpatrick and Grefenstette present experiments in which increasing the population size produces better results than increasing the number of replications per solution in a noisy environment.

Hammel and Bäck (1994) present experiments confirming the observation that Evolution Strategies are robust to low levels of variation. As the noise increases, however, the searches produce poorer results. In these experiments, the authors find that increasing the number of replications per solutions is more effective than increasing the population size when running an Evolution Strategy in the presence of noise. Furthermore, the authors describe additional experiments in which increasing the population size actually impairs the search. As the article points out, these results contradict those of Fitzpatrick and Grefenstette.

Aizawa and Wah (1994) recognize that if the difference between solutions is large, less accuracy (and therefore fewer replications) are required to allow the search to make progress. Based on this idea, they devise a procedure that allocates more replications to generations whose members have closely spaced sample means. They also develop an adaptive procedure that allocates replications one at a time, favoring higher-variance, superior solutions that have received relatively few replications. This rule is based on the idea that better candidates have a better chance of being selected for reproduction in the next generation, so overall performance of the genetic algorithm will be enhanced by performing more replications on superior rather than inferior solutions. The authors run a number of experiments comparing their adaptive procedures to "static" or non-adaptive procedures, where the number of replications allocated to each solution is fixed in advance. The adaptive procedures outperform most, though not all, of the static procedures. Importantly, performance of the dynamic procedure is superior to the average performance of the static procedures, indicating that if one is unsure of how to set parameters in a static procedure (which is typically the case), then the dynamic procedure is a wiser choice.

Marrison and Stengel (1997) incorporate statistical procedures within an EA to validate the selection process and allocate replications according to the within-solution variance of each solution. Essentially, the authors employ tournament selection (apparently with a group size of two) on the principle that if the error of the difference due to noise is smaller than the true difference, tournament selection will be unaffected. To control the error, the authors allocate more replications to high-variance solutions than to their low-variance counterparts. To help ensure that the error is adequately small, the authors base the number of replications required for each solution on the ratio between the observed cross-solution variance of the best 25% of the solutions in the population and the average within-solution variance of those solutions.

Our paper differs from earlier papers in that it presents theory that helps to explain *why* EAs are robust to low levels of stochastic variation; this theory does not rely upon the "Building Block" hypothesis. We then use this theory to develop adaptive methods that can take advantage of a low-variance environment, but can still handle a high-variance environment.

3.3 Selection in a Deterministic Environment

In a deterministic setting, an evolutionary algorithm determines the "fitness" of each solution (which is essentially a set of parameters) by evaluating it via an objective function or other deterministic "black box". Loosely speaking, the better/fitter solutions are assigned a higher probability of "surviving" and being put into the "mating pool" of size M, which is formed by performing M independent draws (with replacement) from the current population. EA researchers have developed a number of methods for assigning these survival probabilities in a deterministic setting, including the following:

Proportional Selection

Under proportional selection, solution i's probability of survival, p_i, is proportional to its fitness. For example, in a maximization problem, we may see

$$p_i = \frac{f_i}{\sum_{j=1}^{k} f_j}$$

where f_i is the performance or objective-function value of solution i and k is the population size.

While this method works well for some problems, it has fallen into disfavor because so-called "super individuals" may dominate the population within a few generations, thwarting the algorithm's ability to explore the solution space. A "super individual" is a solution whose objective-function value is much better than those of all others in the current population, but may be inferior to many other solutions in the broader search space.

Ranking Selection

In a ranking scheme, a solution's survival probability depends only upon the rank, and not the magnitude, of the solution's fitness within the current population. This type of scheme allows the user to explicitly cap the selection probability of the best solution, ameliorating the problem of early convergence due to "super individuals." One such scheme is known as *linear ranking*, where the i^{th}-best solution is assigned a selection probability

$$p_i = \frac{1}{k}\left(\eta - 2(\eta - 1)\left(\frac{i-1}{k-1}\right)\right)$$

where k is the population size and η is a constant between 1 and 2.

Tournament Selection

In the previous schemes, M (mating pool size) independent draws (with replacement) from the population are performed, with the i^{th}-ranked solution having probability p_i of being selected in any single draw. In a tournament scheme, q solutions are selected from the population (with uniform probability) to participate in a tournament. Typically q equals 2, but q must be an integer such that $2 \leq q \leq k$ (k is the population size). The fittest solution in the tournament survives and goes on to the mating pool. Within each generation, M such tournaments are held. This method is primarily designed for ease of implementation; no large-scale sorting is required, and not every solution need be evaluated. In effect, however, tournament selection with tournament size q is a form of ranking selection where the i^{th}-best solution has the selection probability

$$p_i = \frac{1}{k^q}((k-i+1)^q - (k-i)^q)$$

(Bäck, 1996). A characteristic of any *rank-based* scheme, such as linear or tournament selection, is that the selection probabilities can be represented by a fixed $k \times 1$ vector, $\mathbf{p} = \{p_1, p_2, \ldots, p_k\}$ such that

$$\sum_{i=1}^{k} p_i = 1.$$

3.4 Selection in a Stochastic Environment

In a stochastic environment it is more difficult to decide which solutions are truly better, because our ability to conclusively declare one solution superior to another depends both upon the true difference between them and the sampling variance of each. Clearly, this will make the implementation of rank-based selection schemes more difficult.

Before tackling this problem, we need to define some notation. Let X_{ij} be the output of the j^{th} replication for solution i, and let $\bar{X}_i$ be the *sample* mean of all output obtained from solution i. We assume that the output of each replication is a random variable drawn from a distribution with a *true* (but unknown) mean value of μ_i. In a stochastic setting, $E(X_{ij}) = \mu_i$, but due to variation, each observation X_{ij} is not (necessarily) equal to μ_i. In a deterministic setting, any observation at solution i will equal μ_i because there is no variation.

One approach to assigning selection probabilities in this environment is to take a fixed number of sample replications (perhaps just one) at each solution, calculate the sample mean, $\bar{X}_i$, and simply use the *sample* mean as if it were the *true* mean. This procedure may provide accurate rankings if variances are low, but if variances are relatively large, the seemingly "best" solution may be one that simply got a lucky random sample.

Another approach is to take as many replications as necessary to conclusively rank all solutions within a population (to a pre-specified confidence level), and then use those ranks to determine the selection probabilities. Unfortunately, this approach may require so many replications that it could render the algorithm ineffective. In other words, the algorithm would spend so much time trying to rank the solutions in one population that it would not have enough time to move on to new populations that explore other parts of the solution space. This tradeoff between the need for accurate evaluation of individuals and the need to expore other parts of the solution space was recognized by Fitzpatrick and Grefenstette (1988).

The problem with both of the extreme approaches mentioned above is that they try to fit stochastic results into a deterministic selection scheme: the first approach does this by simply ignoring stochastic variation, while the second approach beats the stochastic variation out of the evaluations by taking a large number of replications. The schemes presented in this paper will neither ignore stochastic variation nor eliminate it from the problem. Instead, we will create new selection probability assignment schemes that maintain the spirit of the deterministic schemes — assigning higher selection probabilities to better solutions — without slavishly following the mechanics of the deterministic schemes.

To do this, we will take advantage of the fact that in a deterministic environment evolutionary algorithms are somewhat robust to changes in the specific selection probabilities. For example, one can use linear ranking with a pressure parameter $\eta = 1.4$ rather than $\eta = 1.5$ and still get good results from an EA. Judging from EA robustness to changes in selection probabilities, it seems reasonable to assume that rankings and selection probabilities assigned in a stochastic setting need not be *identical* to those in a deterministic setting, just close enough. One of the primary contributions of this paper is to propose ways to measure what is "close enough."

This robustness to changes in selection probabilities helps to explain why

EAs perform well under low levels of stochastic variation. If stochastic variation is low, incorrect rankings will be infrequent and the magnitude of such errors is not likely to be high. For example, while the true best solution may occasionally be wrongly ranked second- or third-best, it will very rarely be ranked last. This means that the selection probabilities are not changed too dramatically as a result of incorrect ranking.

Below, we provide an example of how this idea of “closeness” between selection probabilities assigned in deterministic and stochastic environments might be more formally defined.

3.4.1 Sum of Total Squared Deviations

Even in a deterministic environment, where we can *assign* selection probabilities, p_i, perfectly, we do not *realize* them exactly in the mating pool. In other words, the number of times solution i is actually selected to go into the mating pool, M_i, is a random variable. Consequently, $\hat{p}_i = M_i/M$, the proportion of the mating pool occupied by solution i, is also a random variable. This is the natural variability inherent to a probabilistic selection scheme. Of course, this variation is a fundamental (and desirable) characteristic of an EA, as it helps to keep the search from getting trapped in local optima.

As a measure of this variability, we define SST to be the total sum of squared deviations between p_i and $\hat{p}_i$. That is

$$SST = \sum_{i=1}^{k} \left(\hat{p}_i - p_i\right)^2. \tag{3.1}$$

In a stochastic setting, we also have variability in the *assignment* of selection probabilities. That is, when stochastic variation causes us to misrank solutions, the selection probability actually assigned to the i^{th}-best solution, S_i, may differ from p_i, the desired selection probability for the i^{th}-best solution. To measure the "closeness" between the stochastic and deterministic environments, it makes sense to look at the ratio of $E(SST_s)$, the expected value of SST in a stochastic environment, to $E(SST_d)$, its deterministic counterpart.

In this section, we will show that the ratio $F = E(SST_s)/E(SST_d)$ can be controlled (lowered) by increasing our ability to distinguish solutions (accomplished by performing additional simulation replications) or by decreasing the selective pressure of the underlying selection scheme. Furthermore, we will show that under certain conditions, $E(SST_s) \geq E(SST_d)$, so $F \geq 1$.

Define the expected sum of squared deviations due to misranking as

$$E(SSD) = E\left\{\sum_{i=1}^{k}(p_i - S_i)^2\right\}.$$

In the Appendix, we show that

$$E(SST) = \sum_{i=1}^{k} E\left\{\frac{S_i(1-S_i)}{M}\right\} + E(SSD).$$

If we define

$$E(SSA) \equiv \sum_{i=1}^{k} E\left\{\frac{S_i(1-S_i)}{M}\right\}$$

then $E(SST) = E(SSA) + E(SSD)$. In a deterministic setting, where $S_i = p_i$, $\forall i$, we have $E(SSD) = 0$. As a result $E(SST_d) = E(SSA_d)$. Therefore,

$$F = \frac{E(SSA_s) + E(SSD_s)}{E(SSA_d)}. \tag{3.2}$$

From the definition of $E(SSA)$, and from the fact that $S_i = p_i\ \forall i$, in a deterministic setting, we can see that

$$E(SSA_d) = \frac{1}{M}\sum_{i=1}^{k} p_i(1-p_i).$$

This expression, which normalizes $E(SST_s)$ in F, depends only upon the chosen selection scheme.

To illustrate the effect of a stochastic setting on $E(SST)$, suppose that we are able to form g equally sized groups of size n (where n is an integer) and suppose that all solutions in group j are known to be superior to all solutions in group $j+1$. Furthermore, suppose that there is no information on how to rank the n solutions within group j; that is, all $n!$ within-group orderings are equally likely to be observed. This is the worst case in the sense that if a statistical grouping procedure is able to form groups of solutions, *at least* this much information about the ranking of the solutions will be available.

In Section 3.4.2 we prove that we can minimize $E(SSD)$ by assigning each group member its group-average selection probability. That is, each member of group j is assigned a selection probability, m_j, where

$$m_j = \frac{1}{n} \sum_{i=(j-1)n+1}^{jn} p_i \quad \text{for } j = 1, 2, \ldots, g.$$

Suppose that we pursue this $E(SSD)$-minimizing course. Define the notation $B(j) = \{(j-1)n+1, (j-1)n+2, \ldots, jn\}$. Under these conditions,

$$\begin{aligned}
E(SSA_s) &= \sum_{j=1}^{g} \sum_{i \in B(j)} E\left\{\frac{S_i(1-S_i)}{M}\right\} \\
&= \frac{1}{M} \sum_{j=1}^{g} n m_j (1-m_j) \\
&= \frac{n}{M} \sum_{j=1}^{g} \left(\frac{1}{n} \sum_{i \in B(j)} p_i\right) \left(1 - \frac{1}{n} \sum_{i \in B(j)} p_i\right) \\
&= \frac{1}{nM} \sum_{j=1}^{g} \left(\sum_{i \in B(j)} p_i\right) \left(\sum_{i \in B(j)} (1-p_i)\right). \qquad (3.3)
\end{aligned}$$

Consequently,

$$F = \frac{\dfrac{1}{nM}\displaystyle\sum_{j=1}^{g}\left(\sum_{i\in B(j)} p_i\right)\left(\sum_{i\in B(j)}(1-p_i)\right)}{\dfrac{1}{M}\displaystyle\sum_{i=1}^{k} p_i(1-p_i)}. \tag{3.4}$$

In the Appendix, we show that when we follow this $E(SSD)$-minimizing course, $F \geq 1$. We also show that

$$\begin{aligned} F &= \frac{\frac{1}{M}(1-U_g)+V-U_g}{\frac{1}{M}(1-V)} \\ &= \frac{VM+1-U_g(M+1)}{(1-V)} \end{aligned}$$

where $V = \sum_{i=1}^{k} p_i^2$ and $U_g = k/g\sum_{j=1}^{g} m_j^2$. This decomposes F into a selective-pressure component, V, and a group-formation or uncertainty component, U_g. From the expressions above, one can see that $U_1 = 1/k$ and $U_k = V$. This means that when we are able to form k groups, $F = (1-V)/(1-V) = 1$.

In the Appendix we derive expressions for V and U_g under Tournament Selection. Using these expressions, we were able to calculate F for various values of g and q, when $M = k = 60$. These values are shown in Table 3.1. The table shows that forming just a few distinct groups is enough to get the ratio F close to one. This is encouraging from an implementation standpoint, because it means that fewer replications will be required to reduce F to a tolerable level, making the stochastic and deterministic EA equivalent in this measure. Furthermore, the table shows that there are a number of different $\{g, q\}$ combinations for which one can attain a fixed value of F. For instance, $F = 1.02$ at $\{g = 4, q = 2\}$, $\{g = 12, q = 4\}$ and $\{g = 15, q = 5\}$. In other words, there is a direct relationship between the maximum selective pressure and the number of groups required to attain a fixed F.

For linear ranking, we show in the Appendix that

$$U_g = \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{k(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3} + \frac{1}{3}\left(1 - \frac{k^2}{g^2}\right)\right) \tag{3.5}$$

and

$$V = \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{k(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3}\right). \tag{3.6}$$

For linear ranking, it is clear that U_g rises with g, so F falls as g increases. Less apparent is the fact that V increases in η; to prove this, it is enough to show that $dV/d\eta$ is positive. From (3.6), one can see that

$$\frac{dV}{d\eta} = \frac{2\eta - 2}{k}\left(\frac{1}{(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3}\right) - 1\right).$$

To get our desired result, it is enough to show that

$$\frac{1}{(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3}\right) > 1.$$

This is shown below for $k > 1$:

$$\begin{aligned}
\frac{1}{(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3}\right) &= \frac{2}{3(k-1)^2}\left(1 - 3k + 2k^2\right) \\
&= \frac{2}{3(k-1)^2}(2k-1)(k-1) \\
&= \frac{2}{3(k-1)}(2k-1) \\
&> 1.
\end{aligned}$$

As with tournament selection, there is a direct relationship between the maximum selective pressure and the number of groups required for a fixed F. In a sense, schemes with higher selective pressure are "riskier" or more aggressive because they probabilistically allocate more search effort to a seemingly superior area of the search space. In a stochastic setting, this risk is increased by the possibility of misranking solutions; as a result, more groups (that is, more ranking certainty) are required to control F.

3.4.2 Minimizing the expected sum of squared deviations

While F—the ratio of $E(SST)$ in stochastic and deterministic environments—allows us to gauge an EA's relative variability at the mating pool level, $E(SSD)$—the expected value of SSD—directly measures the sum of squared deviations between the desired selection probabilities and the selection probabilities actually assigned. As a result, minimizing $E(SSD)$ increases the closeness between the S_i, the assigned selection probabilities, and the p_i, the desired selection probabilities. Not surprisingly, $E(SSD) = 0$ when $S_i = p_i, \ \forall i$; that is, when ranking is completely accurate. Below, we will show how we can minimize $E(SSD)$ under a rank-based scheme by assigning a new vector $\mathbf{m} = \{m_1, m_2, \ldots, m_k\}$, where m_i is defined as the selection probability assigned to the i^{th}-best *observed* solution in a stochastic environment. In other words, we will not even *try* to assign p_i, but rather will assign m_i to the i^{th} *observed* best.

In a deterministic environment, evaluations are certain, so ranking the solutions can result in only one ordering. In a stochastic environment, evaluations are not certain, so if one ranks solutions based upon the *observed* values (as opposed to the unknown true values), many orderings are possible. In fact, because there are k solutions in a population, there are a total of $k!$ possible orderings.

To make the selection probabilities assigned in a stochastic environment closer to those assigned in a deterministic environment, we seek to minimize $E(SSD)$. In theory, one could drive $E(SSD)$ to zero by taking a very large number of replications of each solution. If enough replications were performed, one could conclusively rank all solutions in the population and simply assign each solution the selection probability it would have received in a deterministic setting. Of course, this approach would require an extremely high number of simulation replications.

Rather than waste simulation replications to force solutions to fit into a deterministic scheme, it will be more efficient if we devise a scheme that takes advantage of the data at hand. To this end, we show how to minimize $E(SSD)$ under a ranked-based scheme if distribution $\mathbf{d}$, the probability of observing each of the possible $k!$ orderings of solutions, is known. This will help us to assign selection probabilities when we are uncertain about the true ordering of solutions within a population.

For each of the $k!$ orderings, we can define a $k \times 1$ vector, $\mathbf{a}_h$ that represents the *correct* selection probabilities to be assigned to observed ordering h. This vector $\mathbf{a}_h$ is simply a re-ordering of our original vector of selection probabilities, $\mathbf{p}$. For instance, suppose ordering h was correct except that the *true* best solution was *observed* to be the second-best and vice versa. In this case, $a_{h1} = p_2,\ a_{h2} = p_1$, and $a_{hi} = p_i \quad \text{for } i = 3, 4, \ldots, k$.

Define a $k \times k!$ matrix, $\mathbf{A}$, in which each column is a distinct vector $\mathbf{a}_h$. Because the likelihood of observing each ordering may be different, we define d_h to be the probability of observing a particular ordering h. Let $\mathbf{d} = \{d_1, d_2, \ldots, d_{k!}\}$ be a $k! \times 1$ vector of these probabilities. The vector $\mathbf{d}$ forms a discrete probability distribution.

Now we express the expected sum of squared deviations as

$$E(SSD) = \sum_{h=1}^{k!} \sum_{j=1}^{k} (A_{jh} - m_j)^2 d_h$$

where A_{jh} is the h^{th} element in the j^{th} row of $\mathbf{A}$ and m_j is the selection probability assigned to the *observed* j^{th}-best solution in a stochastic setting.

Call $\mathbf{p}$ (the $k \times 1$ vector that contains the selection probabilities assigned to solutions in a deterministic setting) our "target vector," and call $\mathbf{m} = \{m_1, m_2, \ldots, m_k\}$ (where m_i is the selection probability assigned to the i^{th}-best *observed* solution in a stochastic setting) the "muted vector" because, as we will show below, the selection probabilities in vector $\mathbf{m}$ tend to be less extreme than those in the target vector, $\mathbf{p}$. Intuitively, the less certain we are about the ordering of solutions, the more we "hedge" by averaging selection probabilities. Recall that $\mathbf{A}$ is a $k \times k!$ matrix where each column is a unique permutation of the $k \times 1$ vector $\mathbf{p}$. Recall that $\mathbf{d} = \{d_1, d_2, \ldots, d_{k!}\}$ is a $k! \times 1$ vector denoting the likelihood of each possible ordering (each column of $\mathbf{A}$).

Theorem 1 *To minimize $E(SSD)$, one should set each element, m_j, of the* *muted vector so that,*

$$m_j = \sum_{h=1}^{k!} A_{jh} d_h.$$

Theorem 1, which is proved in the Appendix, can guide us in assigning selection probabilities when we are not completely certain on how to order the solutions within a population.

Theorem 2 *Suppose that the k solutions can be divided into g groups, and that all solutions in group i are known to be superior to all solutions in group $i+1$. Furthermore, suppose that there is no information on how to rank the k_i solutions within group i; that is, all $k_i!$ within-group orderings are equally likely to be observed. Under these assumptions, $E(SSD)$ is minimized when all members of each group are assigned their group's average selection probability. That is*

$$m_j^{(i)} = \frac{1}{k_i} \sum_{j=1}^{k_i} p_j^{(i)} \tag{3.7}$$

where $p_j^{(i)}$ is the selection probability that would have been assigned to the j^{th}-best solution in the i^{th}-best group in a deterministic setting.

The proof of Theorem 2, which is in the Appendix, is similar to the proof of Theorem 1.

Theorem 2 shows how to minimize $E(SSD)$ if we know how to group solutions and we have a target selection probability assignment scheme. Below, we apply this result to several different schemes, and derive expressions for $E(SSD)$. We assume that the population size is k and each group is of equal size. Furthermore, following Theorem 2, each group member is assigned its group average as a selection probability.

Linear Ranking under Grouping

Recall that under linear ranking, the i^{th}-best solution is assigned a selection probability of

$$p_i = \frac{1}{k}\left(\eta - 2(\eta - 1)\left(\frac{i-1}{k-1}\right)\right)$$

where η is a constant between 1 and 2 (Bäck, 1996). Under linear ranking, when g groups, each of size $n = k/g$, can be formed

$$\begin{aligned} E(SSD) &= \frac{(\eta-1)^2(n-1)(n+1)}{3k(k-1)^2} \\ &= \frac{(\eta-1)^2(k^2-g^2)}{3k(k-1)^2 g^2}. \end{aligned} \tag{3.8}$$

Expression (3.8) is derived in the Appendix. Note that $E(SSD)$ increases quadratically with η, so higher selective pressure implies greater $E(SSD)$, and if $\eta = 1$ (random walk), $E(SSD) = 0$.

In the worst case, where it is impossible to distinguish any solution from any other, we are able to form only one group, so $g = 1$. Under muting,

$$E(SSD) = \frac{(\eta - 1)^2(k^2 - 1)}{3k(k-1)^2}.$$

We will call this $E(SSD_{max})$. On the other hand, in the best case, where we can distinguish each solution from every other, we can form k groups so $g = k$ and $E(SSD) = 0$. For intermediate cases, where the number of groups g varies so that $1 < g < k$, we can represent $E(SSD)$, which we will call $E(SSD_g)$, as a *proportion* of $E(SSD_{max})$:

$$\begin{aligned}\frac{E(SSD_g)}{E(SSD_{max})} &= \frac{\frac{(\eta - 1)^2(k^2 - g^2)}{3k(k-1)^2 g^2}}{\frac{(\eta - 1)^2(k^2 - 1)}{3k(k-1)^2}} \\ &= \frac{k^2 - g^2}{g^2(k^2 - 1)} \\ &= \frac{\frac{k^2}{g^2} - 1}{k^2 - 1}.\end{aligned}$$

Because k, the population size, is typically greater than 30 in our setting, we have

$$\begin{aligned}\frac{\frac{k^2}{g^2} - 1}{k^2 - 1} &\approx \frac{\frac{k^2}{g^2} - 1}{(k^2)} \\ &= \frac{1}{g^2} - \frac{1}{k^2}. \qquad (3.9)\end{aligned}$$

As (3.9) shows, the ratio $E(SSD_g)/E(SSD_{max})$ decreases quadratically in g. In other words, we get a diminishing reduction $E(SSD)$ for each additional group, so the biggest decreases in $E(SSD)$ will come from improving from $g = 1$ to $g = 2$, and each additional group formed provides less and less benefit. This is a useful insight because in our setting, forming each additional group comes at the expense of additional simulation replications.

Tournament Selection under Grouping

If we group solutions and assign each its group-average selection probability, the basic expression for $E(SSD)$ is

$$\begin{aligned} E(SSD) &= \sum_{i=1}^{k}(p_i - m_i)^2 \\ &= \sum_{i=1}^{k} p_i^2 - 2\sum_{i=1}^{k} p_i m_i + \sum_{i=1}^{k} m_i^2. \end{aligned}$$

In the Appendix, we show that under tournament selection, when the solutions can be divided correctly into groups of equal size, and we assign each solution its group-average selection probability, as in Theorem 2, then

$$\sum_{i=1}^{k} m_i^2 = \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2 .$$

We also show that

$$\sum_{i=1}^{k} p_i m_i = \sum_{i=1}^{k} m_i^2.$$

As a result,

$$\begin{aligned} E(SSD_g) &= \sum_{i=1}^{k} p_i^2 - \sum_{i=1}^{k} m_i^2 \\ &= \sum_{i=1}^{k} p_i^2 - \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2 . \end{aligned}$$

Using this expression, we calculated the ratio $E(SSD_g)/E(SSD_{max})$ for selected tournament sizes q and number of groups g in a tournament selection scheme with population size, k, equal to 60. These calculations are presented in Table 3.4.2, which shows that the greatest reductions in $E(SSD_g)/E(SSD_{max})$ come from moving from one group to two, and that the reductions diminish as g increases. It also shows that $E(SSD_g)/E(SSD_{max})$ drops more quickly when the tournament size, q (and therefore the selective pressure) is lower.

Targeting Simulation Replications to Reduce SSD

As the table in Section 3.4.2 shows, there is a decreasing marginal benefit in forming additional equal-sized groups. Below, we will show that allocating simulation replications to better solutions reduces $E(SSD)$ at least as quickly as allocating them to inferior solutions. To illustrate this point, suppose that we can expend only enough simulation replications to form a group of size z, where $z < k/2$. Furthermore, suppose that we can choose whether to form a group of the z *best* solutions *or* a group of the z *worst* solutions, but not both. Below, we will show that choosing the first option (forming a group of the z best solutions) is as least as good—in terms of reducing $E(SSD)$—as the second option (forming a group of the z worst solutions).

Define SSD_{Wz} as the SSD resulting from forming two groups: one containing the z *worst* solutions, the other containing the $k-z$ best solutions. Similarly, define SSD_{Bz} as the SSD resulting from forming one group containing the z *best* solutions, and another group containing the $k-z$ worst solutions. Below we derive an expression for $E(SSD_{Wz}) - E(SSD_{Bz})$. If this expression is greater than zero, then we can conclude that it is better (in term of $E(SSD)$ reduction) to form a single group of the z best rather than a single group of the z worst. On the other hand, if this expression is less than zero, we can conclude that the opposite is true.

First, we will define some notation. Let the sum of the selection probabilities of the z worst solutions be defined as

$$R = \sum_{h=k-z+1}^{k} p_h \tag{3.10}$$

and let the sum of the selection probabilities of the z best solutions be defined as

$$Q = \sum_{h=1}^{z} p_h. \tag{3.11}$$

Now, if we group the solutions and assign each solution its group-average selection probability, we can write

$$E(SSD_{Wz}) = \sum_{i=1}^{k-z} \left(p_i - \left(\frac{1-R}{k-z}\right)\right)^2 + \sum_{i=k-z+1}^{k} \left(p_i - \frac{R}{z}\right)^2 \tag{3.12}$$

where the left-hand summation represents the sum of the $k-z$ best probabilities and the right-hand summation represents the sum of the z worst probabilities. Similarly, we can write

$$E(SSD_{Bz}) = \sum_{i=z+1}^{k} \left(p_i - \left(\frac{1-Q}{k-z}\right)\right)^2 + \sum_{i=1}^{z} \left(p_i - \frac{Q}{z}\right)^2. \tag{3.13}$$

In the Appendix we show that

$$E(SSD_{Wz}) - E(SSD_{Bz}) = \frac{k(Q^2 - R^2)}{z(k-z)} - \frac{2(Q-R)}{k-z}.$$

As a result, if $E(SSD_{Wz}) - E(SSD_{Bz}) > 0$, then

$$\frac{k(Q^2 - R^2)}{z(k-z)} - \frac{2(Q-R)}{k-z} > 0 \tag{3.14}$$

which implies that

$$\frac{Q+R}{2z} > \frac{1}{k} \tag{3.15}$$

as long as $Q > R$, which is always the case if the selection scheme assigns higher selection probabilities to better solutions. In other words, if the average of the z worst and the z best selection probabilities is greater than $1/k$, $E(SSD)$ is reduced *more* by grouping the z best solutions rather than by grouping the z worst solutions.

This tells us that additional simulation replications, which may be necessary to separate a group from the rest, are better spent simulating the best (rather than the worst) solutions, (assuming that the number of additional simulation replications required to form a group of the best z solutions is no greater than that required to form a group of the worst z solutions.)

Notice that the result above holds for *any* selection probability assignment scheme, such as linear ranking or tournament selection. Below, we will examine conditions under which the average of the z best and the z worst selection probabilities is greater than $1/k$ for specific schemes.

Average of the z Best and z Worst under Linear Ranking

Recall that under linear ranking,

$$p_i = \frac{1}{k}\left(\eta - 2(\eta - 1)\left(\frac{i-1}{k-1}\right)\right). \tag{3.16}$$

Therefore, the *sum* of the best z and worst z solutions can be expressed as

$$\begin{aligned}
\sum_{i=1}^{z} p_i + \sum_{i=k-z+1}^{k} p_i &= \sum_{i=1}^{z} \left(\frac{1}{k} \left(\eta - 2(\eta - 1) \left(\frac{i-1}{k-1} \right) \right) \right) + \\
&\quad \sum_{i=k-z+1}^{k} \left(\frac{1}{k} \left(\eta - 2(\eta - 1) \left(\frac{i-1}{k-1} \right) \right) \right) \\
&= \frac{2z\eta}{k} - \frac{2(\eta-1)}{k(k-1)} \left(\sum_{i=1}^{z} (i-1) + \sum_{i=k-z+1}^{k} (i-1) \right) \\
&= \frac{2z\eta}{k} - \frac{2(\eta-1)}{k(k-1)} \left(\sum_{i=0}^{z-1} i + \sum_{i=k-z}^{k-1} i \right) \\
&= \frac{2z\eta}{k} - \frac{2z(\eta-1)}{k} \\
&= \frac{2z}{k}.
\end{aligned}$$

Thus, the *average* of the z best and z worst solutions is equal to $1/k$ regardless of the selective pressure, η. This means that for linear ranking, forming a group of the z best solutions has the same impact on $E(SSD)$ as does forming a group of the z worst solutions.

Average of the z Best and z Worst under Tournament Selection

Because tournament selection with tournament size $q = 2$ is equivalent to linear ranking with pressure parameter $\eta = 2$, and tournament selection with $q = 1$ is equivalent to linear ranking with $\eta = 1$ (Bäck 1996), we can expect to draw the same conclusions under these schemes as well.

More generally, for tournament selection with tournament size q, the sum of the selection probabilities of the z best and the z worst solutions can be expressed as

$$\begin{aligned}
\sum_{i=1}^{z} p_i + \sum_{i=k-z+1}^{k} p_i &= \sum_{i=1}^{z} \frac{1}{k^q}\left(i^q - (i-1)^q\right) + \sum_{i=k-z+1}^{k} \frac{1}{k^q}\left(i^q - (i-1)^q\right) \\
&= \frac{1}{k^q}\left(z^q - 0^q\right) + \frac{1}{k^q}\left(k^q - (k-z)^q\right) \\
&= \frac{1}{k^q}\left(k^q + z^q - (k-z)^q\right) \\
&= 1 + \frac{z^q}{k^q} - \frac{(k-z)^q}{k^q} \\
&= 1 + \left(\frac{z}{k}\right)^q - \left(1 - \frac{z}{k}\right)^q .
\end{aligned}$$

If we let $c = z/k$, then the expression above is equal to $1 + c^q - (1-c)^q$. We can use this expression to examine how the sum (and therefore the average) of the z best and worst selection probabilities changes with q. Note that we require $z \leq k/2$, implying that $c \leq 1/2$.

Define Δq as the amount by which the sum of the q best and q worst selection probabilities changes as q increases by one.

$$\begin{aligned}
\Delta q &= \left(1 + c^{q+1} - (1-c)^{q+1}\right) - \left(1 + c^q - (1-c)^q\right) \\
&= c^{q+1} - c^q + (1-c)^q - (1-c)^{q+1}
\end{aligned}$$

$$
\begin{aligned}
&= c^q(c-1) + (1-c)^q\,(1-(1-c)) \\
&= -c^q(1-c) + (1-c)^q c \\
&= c(1-c)\left((1-c)^{q-1} - c^{q-1}\right).
\end{aligned}
$$

Because $c \leq 1/2$, we see that Δq is non-negative. This means that as tournament size q increases, the sum (and therefore the average) of the z best and z worst selection probabilities is non-decreasing. Because we know that tournament selection with $q = 1$ and $q = 2$ is equivalent to linear ranking with $\eta = 1$ and $\eta = 2$, respectively, we know that the average of the z best and z worst selection probabilities is $1/k$ when tournament size $q = 1$ or $q = 2$. This means that under tournament selection for *all* values of q, $E(SSD)$ is reduced at least as much by grouping the z best rather than the z worst solutions.

3.5 Choosing and Integrating a Statistical Procedure

In the preceding section, we simply assumed that we knew how to separate all solutions into groups. In a realistic setting, we would need to employ a statistical procedure to gain that sort of knowledge.

3.5.1 Choosing the Procedure

Ideally, we would like a statistical procedure that uses a minimal number of replications to group solutions, allowing us to reduce $E(SSD)$ to an acceptable level. More specifically, such a procedure would:

Allow sequential data collection

Because we have no information as to the relative means and variances of solutions prior to sampling, we would like a procedure that starts by taking a very small number of replications from each solution. If we are fortunate, and a small initial sample allows us to form groups and lower $E(SSD)$ and therefore $E(SST)$ to an acceptable level, we want to be able to take advantage of that fact. If we are not so fortunate, we would like the statistical procedure to carefully allocate additional replications based on the information collected in the initial sample.

Allow unequal and unknown solution variances

Many statistical procedure are designed to work under the assumption that solutions have equal variances. Unfortunately, this assumption does not hold in a stochastic simulation environment. Furthermore, we would like a procedure that can handle an unbalanced allocation of experiments;

specifically, we would like to allocate more replications to solutions with higher variances.

Produce non-overlapping sets

Several existing procedures may place the same solution into two different but overlapping groups. While there may be good reasons for doing this, it does not serve our purposes. We would like a procedure that returns the statement "All solutions in group i are better than all solutions in group $i-1$." In other words, it is not so important that solutions in the same group be the same, but that solutions in different groups be different. Perhaps the best approach would be to start with the null hypothesis that "All solutions are in the same group" and have the procedure prove otherwise.

Estimate required number of replications for different groupings

Typically, it will take more simulation replications to produce many small groups than to produce a few large groups. If our procedure could estimate the number of replications required to form different groupings, we could calculate the potential reduction in $E(SSD)$ caused by a specified group split, an easy task on a computer, *before* actually expending simulation replications to make the split. Then, with estimates of the number of replications required to make different splits, and the respective potential reductions of $E(SSD)$, one could choose among different possible group splits.

Although we are unaware of any statistical procedure that does *all* of the above, there are a number of grouping procedures that produce non-overlapping subsets (e.g., Calinski and Corsten, 1985). The authors of this paper used such a procedure in `Scenario Seeker`, a simulation-optimization software package described in Boesel, Nelson and Ishii (1999).

3.5.2 Allocating Replications to Form More Groups

To give an example of how a targeted allocation scheme might work within such a statistical grouping procedure, suppose that a grouping procedure simply considers each adjacent pair of solutions, and groups solutions together if a statistical test deems the difference between them not statistically significant. If one thinks of each group as a chain of solutions, the weakest links of that chain—that is, the areas in which the chain is most easily broken with minimal additional simulation replications—are the pairs of solutions where each solution has low variance, few replications already performed, and where there is a relatively large distance between the solutions' sample means. Of course, in trying to choose which weak link to exploit, one must keep in mind the benefits (measured in terms of $E(SSD)$ reduction) as well as the costs (measured in terms of replications allocated).

Under such a procedure, a "back-of-the-envelope" estimate of the number of replications required to split a group at a particular location, say between solutions A and B, might require that

$$\frac{\bar{X}_A - \bar{X}_B}{\sqrt{\frac{S^2_{A-B}}{n_{AB}}}} \geq c_{req}$$

where c_{req} is a statistical constant, $S^2_{A-B} = S^2_A + S^2_B$ where S^2_A and S^2_B are the sample variances of solutions A and B, respectively, and n_{AB} is the total number of replications that must be allocated to each solution. In such a setting, one can estimate that

$$n_{AB} \geq \frac{c^2_{req} S^2_{A-B}}{\left(\bar{X}_A - \bar{X}_B\right)^2}.$$

Now let ΔSSD_{AB} be the reduction in $E(SSD)$ resulting from a split between solutions A and B. That is,

$$\Delta SSD_{AB} = E(SSD_{before}) - E(SSD_{after})$$

where SSD_{before} is the total SSD before the split, and SSD_{after} is the total SSD after the split.

When allocating additional replications, one could choose the pair (A, B) with the highest value of $\Delta SSD_{AB}/n_{AB}$, which is no more than:

$$\frac{\Delta SSD_{AB}(\bar{X}_A - \bar{X}_B)^2}{c_{req}^2 S_{A-B}^2}.$$

Because we are comparing ratios, we can drop the constant c_{req}, and simply choose the pair where the expression

$$\frac{\Delta SSD_{AB}(\bar{X}_A - \bar{X}_B)^2}{S_{A-B}^2}$$

is highest.

The allocation scheme presented above has several shortcomings: it does not account for the number of replications *already* run on each solution, and allocating additional replications using sample mean and variance information might invalidate the statistical guarantee of a particular grouping procedure. Furthermore, such a scheme would be myopic or "greedy," and may not be the most effective way to reduce $E(SSD)$ with minimal cost. Even so, it gives us an idea of the type of things that we would like to see in an allocation procedure, such as considering the benefit of $E(SSD)$ reduction along with the cost of performing additional replications.

3.6 Conclusion

This paper has presented methods to measure and control the effect of stochastic evaluation in evolutionary algorithms. Our framework provides a way to determine if more data (replications) need to be collected in order to keep the EA's "guidance system" (assignment of selection probabilities) working adequately. We can take a small number of replications initially and go back for more only when necessary. Under a fixed-sample-size procedure, one must *hope* that stochastic variation is low enough (or that sample size is large enough) so that ranking is accurate enough to allow the EA to function properly. In a fixed-sample-size procedure, if the sample size is set low, one runs the risk of taking too little data at each solution and allowing the EA to wander randomly, or worse, assigning higher selection probabilities to inferior solutions. On the other hand, if the sample size is set large, one risks taking too much data at each solution, thereby unnecessarily reducing the number of solutions that the EA can evaluate with a given computing budget.

3.7 Appendix

This appendix is divided into nine subsections. The first subsection derives $E(SST)$ and the second proves some properties of F. The third and fourth subsections derive U_g and V under tournament selection and linear ranking, respectively. The fifth and sixth subsections prove Theorems 1 and 2, respectively. The seventh subsection derives SSD under linear ranking and grouping. The eighth subsection derives some expressions for tournament selection under grouping, and the ninth subsection shows some results related to targeting simulation replications.

3.7.1 Derivation of $E(SST)$

Theorem 3

$$E(SST) = \sum_{i=1}^{k} E\left\{\frac{S_i(1-S_i)}{M}\right\} + E\left\{\sum_{i=1}^{k}(p_i - S_i)^2\right\}.$$

Proof:

$$\begin{aligned} SST &= \sum_{i=1}^{k} (\hat{p}_i - S_i + S_i - p_i)^2 \\ &= \sum_{i=1}^{k} \left\{(\hat{p}_i - S_i)^2 - 2(\hat{p}_i - S_i)(p_i - S_i) + (S_i - p_i)^2\right\}. \end{aligned}$$

Let $\mathbf{S} = \{\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_k}\}$. Then

$$\begin{aligned} E[SST] &= E\left\{E[SST|\mathbf{S}]\right\} \\ &= E\Bigg\{\sum_{i=1}^{k} E\left[(\hat{p}_i - S_i)^2|\mathbf{S}\right] - 2\sum_{i=1}^{k} E\left[(\hat{p}_i - S_i)(p_i - S_i)|\mathbf{S}\right] + \\ &\qquad E\left[\sum_{i=1}^{k}(S_i - p_i)^2 \middle| \mathbf{S}\right].\Bigg\}. \end{aligned} \tag{3.17}$$

But notice that, conditional on $\mathbf{S}$, $E\left[(\hat{p}_i - S_i)^2|\mathbf{S}\right]$ is the variance of a binomial proportion, so

$$E[(\hat{p}_i - S_i)^2|\mathbf{S}] = \frac{S_i(1 - S_i)}{M}.$$

Furthermore,

$$\begin{aligned} E\left[(\hat{p}_i - S_i)(p_i - S_i)|\mathbf{S}\right] &= (p_i - S_i)E\left[(\hat{p}_i - S_i)|\mathbf{S}\right] \\ &= (p_i - S_i) \cdot 0 = 0. \end{aligned}$$

As a result

$$E(SST) = \sum_{i=1}^{k} E\left\{\frac{S_i(1 - S_i)}{M}\right\} + E\left\{\sum_{i=1}^{k}(p_i - S_i)^2\right\}.$$

□

3.7.2 Properties of F

Theorem 4 *Suppose that one is able to place solutions into non-overlapping groups. If one then assigns each solution its group-average selection probability,* *then $E(SST)$ in a stochastic environment will be greater than or equal to its deterministic counterpart. That is, $F \geq 1$.*

Proof: It is enough to show that the numerator in equation (3.4) in the text is no less than the denominator. To show this, it is enough to show that

$$\left(\sum_{i=1}^{k} p_i\right)\left(\sum_{i=1}^{k}(1-p_i)\right) - n\sum_{i=1}^{k} p_i(1-p_i) \geq 0$$

Clearly, the equation above is true if:

$$\left(\sum_{i\in B(j)} p_i\right)\left(\sum_{i\in B(j)}(1-p_i)\right) - n\sum_{i\in B(j)} p_i(1-p_i) \geq 0 \quad \forall j.$$

Define the notation $\ell \neq i \in B(j)$ to mean all elements in $B(j)$ that are not equal to i. Below, we show that the equation above is true:

$$\begin{aligned}
&\left(\sum_{i\in B(j)} p_i\right)\left(\sum_{i\in B(j)}(1-p_i)\right) - n\sum_{i\in B(j)} p_i(1-p_i)\\
&= \sum_{i\in B(j)} p_i(1-p_i) + \sum_{i\in B(j)}\sum_{\ell\neq i\in B(j)} p_i(1-p_\ell) - n\sum_{i\in B(j)} p_i(1-p_i)\\
&= -(n-1)\sum_{i\in B(j)} p_i(1-p_i) + \sum_{i\in B(j)}\sum_{\ell\neq i\in B(j)} p_i(1-p_\ell)\\
&= -(n-1)\sum_{i\in B(j)} p_i + (n-1)\sum_{i\in B(j)} p_i^2 + (n-1)\sum_{i\in B(j)} p_i - \sum_{i\in B(j)}\sum_{\ell\neq i\in B(j)} p_i p_\ell\\
&= (n-1)\sum_{i\in B(j)} p_i^2 - \sum_{i\in B(j)}\sum_{\ell\neq i\in B(j)} p_i p_\ell\\
&= n\sum_{i\in B(j)} p_i^2 - \left(\sum_{i\in B(j)} p_i^2 + \sum_{i\in B(j)}\sum_{\ell\neq i\in B(j)} p_i p_\ell\right)\\
&= n\sum_{i\in B(j)} p_i^2 - \left(\sum_{i\in B(j)} p_i\right)^2 \geq 0
\end{aligned}$$

□

Theorem 5 *Suppose that one is able to place solutions into equal-sized, non-overlapping groups. If one then assigns each solution its group-average selection probability, then*

$$F = \frac{VM + 1 - U_g(M+1)}{(1-V)}.$$

Proof: From (3.2), we know that

$$F \quad = \quad \frac{E(SSA_s) + E(SSD_s)}{E(SSA_d)}$$

and from (3.3), we know that if we have equal-sized groups and we assign each solution its group-average selection probability:

$$\begin{aligned}
E(SSA_s) &= \frac{1}{nM}\sum_{j=1}^{g}\left(\sum_{i\in B(j)} p_i\right)\left(\sum_{i\in B(j)}(1-p_i)\right) \\
&= \frac{1}{nM}\sum_{j=1}^{g}(nm_j)(n - nm_j) \\
&= \frac{n}{M}\sum_{j=1}^{g}(m_j)(1-m_j) \\
&= \frac{n}{M}\sum_{j=1}^{g}m_j - \frac{n}{M}\sum_{j=1}^{g}m_j^2 \\
&= \frac{1}{M} - \frac{n}{M}\sum_{j=1}^{g}m_j^2 \\
&= \frac{1}{M}\left(1 - n\sum_{j=1}^{g}m_j^2\right) \\
&= \frac{1}{M}\left(1-U_g\right).
\end{aligned}$$

Furthermore,

$$\begin{aligned}
E(SSA_d) &= \frac{1}{M}\sum_{i=1}^{k} p_i(1-p_i) \\
&= \frac{1}{M}\sum_{i=1}^{k} p_i - p_i^2 \\
&= \frac{1}{M}\left(1-\sum_{i=1}^{k} p_i^2\right) \\
&= \frac{1}{M}\left(1-V\right).
\end{aligned}$$

Finally, if we have equal-sized groups and we assign each solution its group-average selection probability,

$$\begin{aligned}
E(SSD_s) &= \sum_{j=1}^{g}\left(\sum_{i\in B(j)} (p_i-m_j)^2\right) \\
&= \sum_{j=1}^{g}\left(\sum_{i\in B(j)} (p_i^2-2p_i m_j+m_j^2)\right) \\
&= \sum_{i=1}^{k} p_i^2 + n\sum_{j=1}^{g} m_j^2 - 2\sum_{j=1}^{g} m_j \sum_{i\in B(j)} p_i \\
&= \sum_{i=1}^{k} p_i^2 + n\sum_{j=1}^{g} m_j^2 - 2\sum_{j=1}^{g} m_j n m_j \\
&= \sum_{i=1}^{k} p_i^2 - n\sum_{j=1}^{g} m_j^2 \\
&= V - U_g.
\end{aligned}$$

Thus,

$$F = \frac{E(SSA_s)+E(SSD_s)}{E(SSA_d)}$$

$$= \frac{\frac{1}{M}(1-U_g)+V-U_g}{\frac{1}{M}(1-V)}$$

$$= \frac{VM+1-U_g(M+1)}{1-V}.$$

□

3.7.3 Derivation of U_g and V under Tournament Selection

By definition:

$$\begin{aligned}
U_g &= \frac{k}{g}\sum_{j=1}^{g} m_j^2 \\
&= \frac{k}{g}\sum_{j=1}^{g}\left(\frac{1}{n}\sum_{i\in B(j)} p_i\right)^2 \\
&= \frac{g}{k}\sum_{j=1}^{g}\left(\sum_{i\in B(j)} p_i\right)^2 \\
&= \frac{g}{k}\sum_{j=1}^{g}\left(\sum_{i=(j-1)n+1}^{jn}\frac{1}{k^q}\left(i^q-(i-1)^q\right)\right)^2 \\
&= \frac{g}{k}\sum_{j=1}^{g}\left(\frac{1}{k^q}\left\{(jn)^q-((j-1)n)^q\right\}\right)^2 \\
&= \frac{g}{k}\frac{n^{2q}}{k^{2q}}\sum_{j=1}^{g}\left((j^q-(j-1)^q)\right)^2 \\
&= \frac{g}{k}\frac{1}{g^{2q}}\sum_{j=1}^{g}\left(j^q-(j-1)^q\right)^2
\end{aligned}$$

$$= \frac{1}{kg^{2q-1}} \sum_{j=1}^{g} \left(j^q - (j-1)^q\right)^2 .$$

By definition:

$$\begin{aligned}
V &= \sum_{i=1}^{k} p_i^2 \\
&= \sum_{i=1}^{k} \left(\frac{1}{k^q} \left(i^q - (i-1)^q\right) \right)^2 \\
&= \frac{1}{k^{2q}} \sum_{i=1}^{k} \left(\left(i^q - (i-1)^q\right)\right)^2 .
\end{aligned}$$

3.7.4 Derivation of U_g and V under Linear Ranking

By definition:

$$\begin{aligned}
U_g &= \frac{k}{g} \sum_{j=1}^{g} m_j^2 \\
&= \frac{k}{g} \sum_{j=1}^{g} \left(\frac{1}{n} \sum_{i \in B(j)} p_i \right)^2 \\
&= \frac{g}{k} \sum_{j=1}^{g} \left(\sum_{i \in B(j)} p_i \right)^2 \\
&= \frac{g}{k} \sum_{j=1}^{g} \left(\sum_{i \in B(j)} \frac{1}{k} \left(\eta - 2(\eta - 1) \left(\frac{i-1}{k-1} \right) \right) \right)^2 \\
&= \frac{g}{k^3} \sum_{j=1}^{g} \left(n\eta - 2 \left(\frac{\eta - 1}{k-1} \right) \sum_{i \in B(j)} (i-1) \right)^2 \\
&= \frac{g}{k^3} \sum_{j=1}^{g} \left(n^2\eta^2 - 4n\eta \left(\frac{\eta - 1}{k-1} \right) \sum_{i \in B(j)} (i-1) + 4 \left(\frac{\eta - 1}{k-1} \right)^2 \left(\sum_{i \in B(j)} (i-1) \right)^2 \right)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{g}{k^3}\left(gn^2\eta^2 - 4n\eta\left(\frac{\eta-1}{k-1}\right)\sum_{i=1}^{k}(i-1) + 4\left(\frac{\eta-1}{k-1}\right)\sum_{j=1}^{g}\left(\sum_{i\in B(j)}(i-1)\right)^2\right)\\
&= \frac{\eta^2}{k} - \frac{4\eta}{k^2}\left(\frac{\eta-1}{k-1}\right)\frac{k(k-1)}{2} + 4\frac{g}{k^3}\left(\frac{\eta-1}{k-1}\right)^2\sum_{j=1}^{g}\left(\frac{n}{2}((2j-1)n-1)\right)^2\\
&= \frac{\eta^2}{k} - \frac{2\eta(\eta-1)}{k} + \frac{g(\eta-1)^2n^2}{k^3(k-1)^2}\sum_{j=1}^{g}((2j-1)n-1)^2\\
&= \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{gk(k-1)^2}\sum_{j=1}^{g}((2j-1)n-1)^2. \qquad (3.18)
\end{aligned}
$$

For simplicity, we will isolate

$$
\begin{aligned}
\sum_{j=1}^{g}((2j-1)n-1)^2 &= \sum_{j=1}^{g}\Big(((2j-1)n)^2 - 2(2j-1)n + 1\Big)\\
&= g + \sum_{j=1}^{g}\Big(n^2(4j^2-4j+1) - n(4j-2)\Big)\\
&= g + \left(n^2\left(4\sum_{j=1}^{g}j^2 - 4\sum_{j=1}^{g}j + g\right) - n\left(4\sum_{j=1}^{g}j - 2g\right)\right)\\
&= g + gn^2 + 2gn + n^2\left(4\sum_{j=1}^{g}j^2 - 2(g^2+g) - 2g^2 + 2g^2\right)\\
&\quad -n\Big(2(g^2+g)\Big)\\
&= g + gn^2 - 2g^2n + n^2\left(4\left(\frac{g^3}{3} - \frac{g^2}{2} + \frac{g}{6}\right) + 2g^2 - 2g\right)\\
&= g + gn^2 - 2g^2n + n^2\left(\frac{4g^3}{3} + \frac{2g}{3} - 2g\right)\\
&= g - 2g^2n + \frac{4n^2g^3}{3} - \frac{gn^2}{3}\\
&= g - 2gk + \frac{4gk^2}{3} - \frac{k^2}{3g}. \qquad (3.19)
\end{aligned}
$$

Combining (3.18) and (3.19) we see that

$$\begin{aligned}
U_g &= \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{gk(k-1)^2}\left(g - 2gk + \frac{4gk^2}{3} - \frac{k^2}{3g}\right) \\
&= \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{k(k-1)^2}\left(1 - 2k + \frac{4k^2}{3} - \frac{k^2}{3g^2}\right) \\
&= \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{k(k-1)^2}\left(\frac{2}{3} - 2k + \frac{4k^2}{3} + \frac{1}{3}\left(1 - \frac{k^2}{g^2}\right)\right) \qquad (3.20)
\end{aligned}$$

which is (3.5). To show (3.6), begin with the definition:

$$\begin{aligned}
V &= \sum_{i=1}^{k} p_i^2 \\
&= \sum_{i=1}^{k}\left(\frac{1}{k}\left(\eta - 2(\eta-1)\left(\frac{i-1}{k-1}\right)\right)\right)^2 \\
&= \frac{1}{k^2}\sum_{i=1}^{k}\left(\eta^2 - 4\eta(\eta-1)\frac{i-1}{k-1} + 4(\eta-1)^2\left(\frac{i-1}{k-1}\right)^2\right) \\
&= \frac{1}{k^2}\left(k\eta^2 - \frac{4\eta(\eta-1)}{k-1}\sum_{i=1}^{k}(i-1) + \frac{4(\eta-1)^2}{(k-1)^2}\sum_{i=1}^{k}(i-1)^2\right) \\
&= \frac{1}{k^2}\left(k\eta^2 - 2\eta(\eta-1)k + \frac{4(\eta-1)^2}{(k-1)^2}\left(-k^2 + \sum_{i=1}^{k} i^2\right)\right) \\
&= \frac{1}{k^2}\left(-k\eta^2 + 2\eta k + \frac{4(\eta-1)^2}{(k-1)^2}\left(\sum_{i=1}^{k-1} i^2\right)\right) \\
&= \frac{1}{k^2}\left(\eta k(2-\eta) + \frac{4(\eta-1)^2}{(k-1)^2}\left(\frac{k^3}{3} - \frac{k^2}{2} + \frac{k}{6}\right)\right) \\
&= \frac{\eta(2-\eta)}{k} + \frac{4(\eta-1)^2}{(k-1)^2}\left(\frac{k}{3} - \frac{1}{2} + \frac{1}{6k}\right) \\
&= \frac{\eta(2-\eta)}{k} + \frac{(\eta-1)^2}{k(k-1)^2}\left(\frac{4k^2}{3} - 2k + \frac{2}{3}\right)
\end{aligned}$$

which is (3.6).

3.7.5 Proof of Theorem 1

Recall that

$$\sum_{j=1}^{k} p_j = \sum_{j=1}^{k} m_j = \sum_{h=1}^{k!} d_h = 1 \tag{3.21}$$

$$0 \leq p_j, \quad 0 \leq m_j, \ \forall j \ \text{ and } 0 \leq d_h, \ \forall h,$$

and

$$E(SSD) = \sum_{h=1}^{k!} \sum_{j=1}^{k} (A_{jh} - m_j)^2 d_h.$$

For a given j, taking the first partial derivative yields

$$\frac{\partial E(SSD)}{\partial m_j} = \sum_{h=1}^{k!} -2(A_{jh} - m_j) d_h$$

and the second partial derivative is

$$\frac{\partial^2 E(SSD)}{\partial m_j^2} = \sum_{h=1}^{k!} 2d_h = 2.$$

Because the second partial derivative is always positive, we can minimize by setting the first partial to zero:

$$\sum_{h=1}^{k!} -2(A_{jh} - m_j) d_h = 0.$$

Let m_j^* be the solution. Therefore $\sum_{h=1}^{k!} A_{jh} d_h = m_j^* \sum_{h=1}^{k!} d_h = m_j^*, \ \forall j$ by equation (3.21). Now we need to show that $\mathbf{m}^*$ is a probability vector, that is

$0 \leq m_j^* \leq 1, \quad \forall\, j$ and $\sum_{j=1}^{k} m_j^* = 1$. Notice that

$$m_j^* = \sum_{h=1}^{k!} A_{jh} d_h \leq \sum_{h=1}^{k!} 1 d_h = 1 \quad \forall j \tag{3.22}$$

and

$$m_j^* \geq \sum_{h=1}^{k!} 0 d_h = 0 \quad \forall j.$$

Therefore $0 \leq m_j^* \leq 1$. Also notice that

$$\begin{aligned}\sum_{j=1}^{k} m_j^* &= \sum_{j=1}^{k} \sum_{h=1}^{k!} A_{jh} d_h \\ &= \sum_{h=1}^{k!} \sum_{j=1}^{k} A_{jh} d_h \\ &= \sum_{h=1}^{k!} d_h \sum_{j=1}^{k} A_{jh}. \end{aligned} \tag{3.23}$$

Because each column of $\mathbf{A}$ is a permutation of $\mathbf{p}$, $\sum_{j=1}^{k} A_{jh} = 1, \quad \forall h$. Consequently, (3.23) is equivalent to $\sum_{h=1}^{k!} d_h = 1$. $\square$

3.7.6 Proof of Theorem 2

Notation:

k_i: Number of solutions within group i.

$\mathbf{p_i}$: A $(k_i \times 1)$ sub-vector of $\mathbf{p}$, representing the "target" probabilities of group i.

$\mathbf{A_i}$: A $(k_i \times k_i!)$ matrix, each of whose columns is a unique permutation of $\mathbf{p_i}$.

$A^{(i)}_{jh}$: The component in the j^{th} row and h^{th} column of $\mathbf{A_i}$.

$\mathbf{d_i}$: A $(k_i! \times 1)$ vector representing the probability of each outcome represented by $\mathbf{A_i}$.

$d^{(i)}_h$: The h^{th} component of $\mathbf{d_i}$.

$\mathbf{m_i}$: A $(k_i \times 1)$ vector representing the "muted" probabilities of group i.

C: Number of outcomes with non-zero probability.

$\mathbf{A_R}$: A $(k \times C)$ matrix, containing all columns of $\mathbf{A}$ that have non-zero probability.

A^R_{jh}: The component in the j^{th} row and h^{th} column of $\mathbf{A_R}$.

Because each group can be ordered in $k_i!$ ways and because orderings in one group do not preclude orderings in different groups, the total number of possible orderings is $C = \prod_{i=1}^{g}(k_i!)$. Furthermore, because we have assumed that each within-group ordering is equally likely to be observed, $d^{(i)}_h = 1/k_i!, \ \ \forall h$. Recall that $E(SSD) = \sum_{h=1}^{k!}\sum_{j=1}^{k}(A_{jh} - m_j)^2 d_h$.

As a first step, we would like to rewrite the above in terms of the matrix $\mathbf{A_R}$, whose columns represent all orderings that are possible under our grouping assumption. In other words, these are the columns that correspond to the non-zero components of $\mathbf{d}$.

Because each non-zero element of $\mathbf{d}$ is equally likely, we know that

$$E(SSD) = \sum_{h=1}^{C}\sum_{j=1}^{k}(A_{jh}^{(R)} - m_j)^2\frac{1}{C}. \tag{3.24}$$

Since the g groups are consecutive, non-overlapping, and exhaustive over the k solutions, we can consider the contribution of each group i to the $E(SSD)$ independently, rather than considering all C outcomes at once, as is done above. Each group has $k_i!$ possible outcomes, and each column of $\mathbf{A_i}$ represents a possible outcome within group i. In $\mathbf{A_R}$, each of these group outcomes is repeated $C/k_i!$ times. As a result, we know that

$$(3.24) = \sum_{i=1}^{g}\sum_{h=1}^{k_i!}\frac{C}{k_i!}\sum_{j=1}^{k_i}(A_{jh}^{(i)} - m_j^{(i)})^2\frac{1}{C} \tag{3.25}$$

$$= \sum_{i=1}^{g}\sum_{h=1}^{k_i!}\sum_{j=1}^{k_i}(A_{jh}^{(i)} - m_j^{(i)})^2\ \frac{1}{k_i!}. \tag{3.26}$$

Taking the first partial derivative with respect to $m_j^{(i)}$ yields

$$\frac{\partial E(SSD)}{\partial m_j^{(i)}} = \sum_{h=1}^{k_i!} -2\left(A_{jh}^{(i)} - m_j^{(i)}\right)\frac{1}{k_i!}$$

and the second partial derivative is

$$\frac{\partial^2 E(SSD)}{\partial m_j^{(i)2}} = \sum_{h=1}^{k_i!} \frac{2}{k_i!} = 2.$$

Because the second partial derivative is always positive, we can minimize by setting the first partial derivative to zero:

$$\sum_{h=1}^{k_i!} -2\left(A_{jh}^{(i)} - m_j^{(i)}\right)\frac{1}{k_i!} = 0.$$

This implies that

$$\sum_{h=1}^{k_i!} A_{jh}^{(i)} \frac{1}{k_i!} = m_j^{i*} \sum_{h=1}^{k_i!} \frac{1}{k_i!} \quad \forall\{i,j\}$$

where m_j^{i*} is the solution, so

$$m_j^{i*} = \frac{1}{k_i!} \sum_{h=1}^{k_i!} A_{jh}^{(i)} \quad \forall\{i,j\}. \tag{3.27}$$

Because $\mathbf{A_i}$ is a matrix of all possible orderings of $\mathbf{p_i}$, the sum of the elements in each row is equal, so we can sum up all the elements of the matrix and divide by k_i to get the sum of any row. As a result,

$$\begin{aligned} (3.27) &= \frac{1}{k_i}\frac{1}{k_i!}\sum_{j=1}^{k_i}\sum_{h=1}^{k_i!} A_{jh}^{(i)} \\ &= \frac{1}{k_i}\frac{1}{k_i!}k_i!\sum_{j=1}^{k_i} p_j^{(i)} \\ &= \frac{1}{k_i}\sum_{j=1}^{k_i} p_j^{(i)} \end{aligned}$$

which is simply the average of the group i's selection probabilities. Now, we need to show that $\sum_{i=1}^{g}\sum_{j=1}^{k_i} m_j^{i*} = 1$ and $0 \le m_j^{i*}, \quad \forall\{i,j\}$. Recall from (3.27)

$$\begin{aligned}
\sum_{j=1}^{k_i} m_j^{i*} &= \sum_{j=1}^{k_i} \frac{1}{k_i!} \sum_{h=1}^{k_i!} A_{jh}^{(i)} \quad \forall i \\
&= \frac{1}{k_i!} \sum_{h=1}^{k_i!} \sum_{j=1}^{k_i} A_{jh}^{(i)} \quad \forall i \qquad (3.28)
\end{aligned}$$

Because each column of $\mathbf{A_i}$ is a permutation of $\mathbf{p_i}$,

$$\begin{aligned}
\sum_{j=1}^{k_i} m_j^{i*} &= \frac{1}{k_i!} \sum_{h=1}^{k_i!} \sum_{j=1}^{k_i} p_j^{(i)} \\
&= \sum_{j=1}^{k_i} p_j^{(i)}.
\end{aligned}$$

So

$$\sum_{i=1}^{g}\sum_{j=1}^{k_i} m_j^{i*} = \sum_{i=1}^{g}\sum_{j=1}^{k_i} p_j^{(i)} = 1.$$

Also notice that $m_j^{i*} = (1/k_i!)\sum_{h=1}^{k_i!} A_{jh}^{(i)}$ and $0 \le A_{jh}^{(i)}$, so $m_j^{i*} \ge 0 \sum_{h=1}^{k_i!} d_h^{(i)} = 0, \quad \forall i.$ □

3.7.7 Derivation of SSD under linear ranking and grouping

Let m_i be the average target probability of group i. To simplify the analysis, we assume that $g = k/n$ is an integer. First, we will derive an expression for

m_i.

$$\begin{aligned}
m_i &= \frac{1}{n}\sum_{j=(i-1)n+1}^{in} p_j \\
&= \frac{1}{n}\sum_{j=(i-1)n+1}^{in} \frac{1}{k}(\eta - 2(\eta-1)\frac{j-1}{k-1}) \\
&= \frac{\eta}{k} - \frac{2(\eta-1)}{nk(k-1)}\sum_{j=(i-1)n+1}^{in} j-1 \\
&= \frac{\eta}{k} - \frac{2(\eta-1)}{nk(k-1)}\sum_{j=(i-1)n}^{in-1} j \\
&= \frac{\eta}{k} - \frac{2(\eta-1)}{nk(k-1)}\frac{n(2ni-n-1)}{2} \\
&= \frac{\eta}{k} - \frac{(\eta-1)(2ni-n-1)}{k(k-1)}.
\end{aligned}$$

Now, we will derive the $E(SSD_i)$ for a single group.

$$\begin{aligned}
E(SSD_i) &= \sum_{j=(i-1)n+1}^{in} (p_j - m_i)^2 \\
&= \sum_{j=(i-1)n+1}^{in} (p_j^2 - 2p_j m_i + m_i^2) \\
&= nm_i^2 - 2m_i\left(\sum_{j=(i-1)n+1}^{in} p_j\right) + \left(\sum_{j=(i-1)n+1}^{in} p_j^2\right) \\
&= nm_i^2 - 2m_i n m_i + \left(\sum_{j=(i-1)n+1}^{in} p_j^2\right) \\
&= -nm_i^2 + \left(\sum_{j=(i-1)n+1}^{in} p_j^2\right) \\
&= -nm_i^2 + \sum_{j=(i-1)n+1}^{in} \left(\frac{1}{k}\left(\eta - 2(\eta-1)\frac{j-1}{k-1}\right)\right)^2
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{k^2}\sum_{j=(i-1)n+1}^{in}\left(\eta^2 - 4\eta(\eta-1)\frac{j-1}{k-1} + \left(2(\eta-1)\frac{j-1}{k-1}\right)^2\right) \\
&\quad -n\left(\frac{\eta^2}{k^2} - 2\frac{\eta(\eta-1)(2ni-n-1)}{k^2(k-1)} + \left(\frac{(\eta-1)(2ni-n-1)}{k(k-1)}\right)^2\right) \\
&= \frac{1}{k^2}\left[\frac{-4\eta(\eta-1)}{k-1}\left(\sum_{j=(i-1)n+1}^{in}(j-1)\right) + \frac{4(\eta-1)^2}{(k-1)^2}\left(\sum_{j=(i-1)n+1}^{in}(j-1)^2\right)\right] \\
&\quad -n\left[\frac{-2\eta(\eta-1)}{k^2(k-1)}(2ni-n-1) + \frac{(\eta-1)^2}{k^2(k-1)^2}(2ni-n-1)^2\right] \\
&= \frac{1}{k^2}\left[\frac{4(\eta-1)^2}{(k-1)^2}\left(\sum_{j=(i-1)n+1}^{in}(j^2-2j+1)\right)\right] - n\left[\frac{(\eta-1)^2}{k^2(k-1)^2}(2ni-n-1)^2\right] \\
&= \frac{1}{k^2}\left[\frac{4(\eta-1)^2}{(k-1)^2}\left[\left(\sum_{j=(i-1)n+1}^{in}j^2\right) - 2\left(\sum_{j=(i-1)n+1}^{in}j\right) + n\right]\right] \\
&\quad -n\left[\frac{(\eta-1)^2}{k^2(k-1)^2}(4n^2i^2 - 4n^2i - 4ni + n^2 + 2n + 1)\right] \\
&= \frac{1}{k^2}\left[\frac{4(\eta-1)^2}{(k-1)^2}\left[\left(\sum_{j=(i-1)n+1}^{in}j^2\right) - n(2in-n+1) + n)\right]\right] \\
&\quad -n\left[\frac{(\eta-1)^2}{k^2(k-1)^2}(4n^2i^2 - 4n^2i - 4ni + n^2 + 2n + 1)\right] \\
&= \frac{1}{k^2}\left[\frac{4(\eta-1)^2}{(k-1)^2}\left[(n^3i^2 - n^3i + n^3/3 + n^2i - n^2/2 + n/6) - (2n^2i - n^2)\right]\right] \\
&\quad -n\left[\frac{(\eta-1)^2}{k^2(k-1)^2}(4n^2i^2 - 4n^2i - 4ni + n^2 + 2n + 1)\right] \\
&= \frac{1}{k^2}\left[\frac{4(\eta-1)^2}{(k-1)^2}[n^3/3 - n^2i + n^2/2 + n/6]\right] \\
&\quad -\frac{1}{k^2}\left[\frac{(\eta-1)^2}{(k-1)^2}[n^3 - 4n^2i + 2n^2 + n]\right] \\
&= \frac{1}{k^2}\left[\frac{(\eta-1)^2}{(k-1)^2}[n^3/3 - n/3]\right] \\
&= \frac{(\eta-1)^2 n(n^2-1)}{3k^2(k-1)^2}.
\end{aligned}
$$

Interestingly, $E(SSD_i)$ is the same for each group (independent of i), so

$$\begin{aligned} E(SSD) &= g \times E(SSD_i) \\ &= \frac{k}{n}\frac{(\eta-1)^2 n(n^2-1)}{3k^2(k-1)^2} \\ &= \frac{(\eta-1)^2(n^2-1)}{3k(k-1)^2}. \end{aligned}$$

□

3.7.8 Tournament Selection Under Grouping

Derivation of Expression for $\sum_{i=1}^{k} m_i^2$

Let m_i be the selection probability assigned to *solution* i and let $m^{(h)}$ be the selection probability assigned to each solution in *group* h. We will show that

$$\sum_{i=1}^{k} m_i^2 = \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2.$$

Because all members of the same group have the same muted probability,

$$\sum_{i=1}^{k} m_i^2 = \sum_{h=1}^{g} n(m^{(h)})^2.$$

To make calculations easier, we will use anti-ranks, so $\tilde{p}_i$ is the selection probability of the i^{th}-*worst* solution. Now we will derive an expression for $\tilde{m}^{(h)}$ the muted selection probability of the h^{th}-worst group. By definition

$$\tilde{m}^{(h)} = \frac{1}{n} \sum_{i=(h-1)n+1}^{hn} \tilde{p}_i \tag{3.29}$$

where $n = k/g$ is the number of solutions in each (equally sized) group. Using an expression for $\tilde{p}_i$ under tournament selection, we see that

$$\begin{aligned}
\tilde{m}^{(h)} &= \frac{1}{n} \sum_{i=(h-1)n+1}^{hn} \frac{1}{k^q} \left(i^q - (i-1)^q\right) \\
&= \frac{1}{nk^q} \left((hn)^q - ((h-1)n)^q\right) \\
&= \frac{n^q}{nk^q} \left(h^q - (h-1)^q\right) \\
&= \frac{1}{kg^{q-1}} \left(h^q - (h-1)^q\right).
\end{aligned}$$

Now with the result above, we can see that

$$\begin{aligned}
\sum_{i=1}^{k} m_i^2 &= n \sum_{h=1}^{g} \left(\frac{1}{kg^{q-1}} \left(h^q - (h-1)^q\right) \right)^2 \\
&= \frac{n}{k^2 g^{2q-2}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2 \\
&= \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2 \qquad (3.30)
\end{aligned}$$

which is the desired result.

Proof that $\sum_{i=1}^{k} p_i m_i = \sum_{i=1}^{k} m_i^2$

We will rewrite the left-hand side of the expression above to take advantage of the fact that $m^{(h)}$ is the same for all solutions within the same group, so

$$\sum_{i=1}^{k} p_i m_i = \sum_{h=1}^{g} m^{(h)} \left(\sum_{i=(h-1)n+1}^{hn} p_i \right). \qquad (3.31)$$

We also know that $\sum_{i=1}^{k} p_i m_i = \sum_{i=1}^{k} \tilde{p}_i \tilde{m}_i$. Using the definition of $\tilde{p}_i$ and our expression for $\tilde{m}_h$, we can rewrite the above as

$$
\begin{aligned}
\sum_{i=1}^{k} \tilde{p}_i \tilde{m}_i &= \sum_{h=1}^{g} \frac{1}{kg^{q-1}} \left(h^q - (h-1)^q\right) \left(\sum_{i=(h-1)n+1}^{hn} \frac{1}{k^q} \left(i^q - (i-1)^q\right) \right) \\
&= \frac{1}{kg^{q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right) \frac{1}{k^q} \left((hn)^q - ((h-1)n)^q\right) \\
&= \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right) \left(h^q - (h-1)^q\right) \\
&= \frac{1}{kg^{2q-1}} \sum_{h=1}^{g} \left(h^q - (h-1)^q\right)^2 .
\end{aligned}
$$

The expression above is (3.30), which is equal to $\sum_{i=1}^{k} m_i^2$, so we have our result.

3.7.9 Targeting Simulation Replications to Reduce SSD

We will show that

$$SSD_{Wz} - SSD_{Bz} = \frac{k(Q^2 - R^2)}{z(k-z)} - \frac{2(Q-R)}{k-z}.$$

Expanding the expressions used in equations (3.12) and (3.13) in the text, we get

$$
\begin{aligned}
SSD_{Wz} &= \sum_{i=1}^{k-z} \left(p_i^2 - 2p_i \left(\frac{1-R}{k-z} \right) + \left(\frac{1-R}{k-z} \right)^2 \right) \\
&\quad + \sum_{i=k-z+1}^{k} \left(p_i^2 - 2p_i \left(\frac{R}{z} \right) + \left(\frac{R}{z} \right)^2 \right)
\end{aligned}
$$

and

$$\begin{aligned}
SSD_{Bz} &= \sum_{i=z+1}^{k} \left(p_i^2 - 2p_i \left(\frac{1-Q}{k-z} \right) + \left(\frac{1-Q}{k-z} \right)^2 \right) \\
&\quad + \sum_{i=1}^{z} \left(p_i^2 - 2p_i \left(\frac{Q}{z} \right) + \left(\frac{Q}{z} \right)^2 \right).
\end{aligned}$$

As a result,

$$\begin{aligned}
SSD_{Wz} - SSD_{Bz} &= \sum_{i=z+1}^{k-z} \left(-2p_i \left(\frac{(1-R)-(1-Q)}{k-z} \right) + \frac{(1-R)^2-(1-Q)^2}{(k-z)^2} \right) \\
&\quad + \sum_{i=1}^{z} \left(p_i^2 - 2p_i \left(\frac{1-R}{k-z} \right) + \left(\frac{1-R}{k-z} \right)^2 \right) \\
&\quad - \sum_{i=k-z+1}^{k} \left(p_i^2 - 2p_i \left(\frac{1-Q}{k-z} \right) + \left(\frac{1-Q}{k-z} \right)^2 \right) \\
&\quad + \sum_{i=k-z+1}^{k} \left(p_i^2 - 2p_i \left(\frac{R}{z} \right) + \left(\frac{R}{z} \right)^2 \right) \\
&\quad - \sum_{i=1}^{z} \left(p_i^2 - 2p_i \left(\frac{Q}{z} \right) + \left(\frac{Q}{z} \right)^2 \right)
\end{aligned}$$

$$\begin{aligned}
&= -2 \left(\frac{Q-R}{k-z} (1-(Q+R)) \right) + (k-2z) \left(\frac{(1-R)^2-(1-Q)^2}{(k-z)^2} \right) \\
&\quad + \sum_{i=1}^{z} \left(-2p_i \left(\frac{1-R}{k-z} - \frac{Q}{z} \right) + \left(\frac{1-R}{k-z} \right)^2 - \left(\frac{Q}{z} \right)^2 \right) \\
&\quad + \sum_{i=k-z+1}^{k} \left(-2p_i \left(\frac{R}{z} - \frac{1-Q}{k-z} \right) + \left(\frac{R}{z} \right)^2 - \left(\frac{1-Q}{k-z} \right)^2 \right)
\end{aligned}$$

$$= -2 \left(\frac{Q-R}{k-z} (1-(Q+R)) \right) + (k-2z) \left(\frac{(1-R)^2-(1-Q)^2}{(k-z)^2} \right)$$

$$-2Q\left(\frac{1-R}{k-z}-\frac{Q}{z}\right)+z\left(\left(\frac{1-R}{k-z}\right)^2-\left(\frac{Q}{z}\right)^2\right)$$
$$-2R\left(\frac{R}{z}-\frac{1-Q}{k-z}\right)+z\left(\left(\frac{R}{z}\right)^2-\left(\frac{1-Q}{k-z}\right)^2\right)$$

$$\begin{aligned}
&= -2\left(\frac{Q-R}{k-z}\left(1-(Q+R)\right)\right)+(k-2z)\left(\frac{(1-R)^2-(1-Q)^2}{(k-z)^2}\right)\\
&\quad -2\left(\frac{Q-R}{k-z}+\frac{R^2-Q^2}{z}\right)+z\left(\frac{(1-R)^2-(1-Q)^2}{(k-z)^2}+\frac{R^2-Q^2}{z^2}\right)\\
&= -2\left(\frac{Q-R}{k-z}\left(2-(Q+R)\right)+\frac{R^2-Q^2}{z}\right)\\
&\quad +(k-z)\left(\frac{(1-R)^2-(1-Q)^2}{(k-z)^2}\right)+\frac{R^2-Q^2}{z}\\
&= \frac{Q^2-R^2}{z}+\frac{Q^2-R^2-2Q+2R}{k-z}\\
&= \frac{k(Q^2-R^2)}{z(k-z)}-\frac{2(Q-R)}{k-z}.
\end{aligned}$$

Table 3.1: F as a function of tournament size, q, and the number of equally sized groups formed, g.

	Tournament Size, q			
g	2	3	4	5
1	1.35	1.84	2.36	2.89
2	1.09	1.25	1.55	1.96
3	1.04	1.11	1.26	1.50
4	1.02	1.06	1.15	1.29
5	1.01	1.04	1.10	1.19
6	1.01	1.03	1.07	1.14
10	1.00	1.01	1.02	1.05
12	1.00	1.01	1.02	1.03
15	1.00	1.00	1.01	1.02
20	1.00	1.00	1.01	1.01
30	1.00	1.00	1.00	1.00
60	1.00	1.00	1.00	1.00

Table 3.2: $E(SSD_g)/E(SSD_{max})$ for selected tournament sizes q in a tournament selection scheme, population size $k = 60$

g	$q = 2$	$q = 6$	$q = 10$
1	1	1	1
2	0.297	0.587	0.766
3	0.136	0.331	0.554
4	0.077	0.203	0.395
5	0.049	0.135	0.286
6	0.034	0.096	0.213
10	0.012	0.035	0.085
12	0.008	0.024	0.059
15	0.005	0.015	0.038
20	0.003	0.008	0.020
30	0.001	0.003	0.008
60	0.000	0.000	0.000

Chapter 4

Using Ranking and Selection to 'Clean Up' After Simulation Optimization

4.1 Introduction

Many of the current approaches to simulation optimization can be placed into one of two categories:

1. **Asymptotically convergent search algorithms:** These algorithms, encountered frequently in the academic literature, guarantee that the best system is returned as search effort goes to infinity. Restrictive conditions, however, can make them difficult to apply to a broad range of problems, and the amount of simulation effort required to converge can be prohibitive.

2. **Deterministic search algorithms applied to stochastic problems:** These algorithms, which are widely implemented in commercial software packages, may provide good solutions, but may also provide misleading results because they frequently ignore stochastic variation and they provide no statistical guarantees.

While the first approach guarantees to return the true best solution, the second approach does not. The aim of this paper is to "clean up" results generated by the second approach. Specifically, we will develop methods that return the best system of all those systems visited by the search—or one within a user-specified distance of the best—with a pre-specified probability. Our goal is to deliver this statistical guarantee with as little additional simulation effort as possible beyond that already expended in the search. This paper will not discuss methods for improving the search or generating new and better solutions. Rather, it will deal only with the problem of finding the best solution *among those already visited by the search algorithm.* The procedures developed here can also be used independently of a search when there are a fixed (and perhaps large) number of alternatives to compare.

The remainder of the paper is organized into four sections. Section 4.2 gives a brief overview of screening and selection procedures, and shows how they can work together. The next two sections present distinct strategies for cleaning up after a search. Section 4.3.1 describes a simple "restart" procedure that combines screening and selection, and proposes an algorithm to find the best sample size when re-running first-stage samples. Section 4.4 describes a procedure that sorts the systems by first-stage sample mean and does not re-run the first-stage samples. An empirical evaluation and comparison of these approaches is provided in Section 4.5. The Appendix contains proofs of the validity of procedures developed in the paper, as well as supporting procedures and results.

4.2 Background

We assume that a preliminary or *first-stage* set of simulation output data generated by a search procedure is "dropped into our laps." Let k be the number of different systems in the data set, and let n_{0i} be the number of replications already performed on system i. Notice that we do not require equal first-stage sample sizes, since a search procedure may revisit solutions or take differing numbers of observations from them. Further, let X_{ij} be the output from replication j of system i, which we assume are i.i.d. $\mathrm{N}(\mu_i, \sigma_i^2)$ random variables. Assuming normality is often reasonable when each replication output variable X_{ij} is the average of a large number of more basic output variables. Systems are to be compared based on their true means, μ_i, and we assume that larger μ_i is better throughout this paper.

The first-stage sample mean of system i is

$$\bar{X}_i^{(1)} = \frac{1}{n_{0i}} \sum_{j=1}^{n_{0i}} X_{ij}$$

while S_{0i}^2 is the first-stage sample variance of system i; that is

$$S_{0i}^2 = \frac{1}{n_{0i}-1} \sum_{j=1}^{n_{0i}} \left(X_{ij} - \bar{X}_i^{(1)} \right)^2.$$

Finally, let S_{ri}^2 be the sample variance of system i based on a second, independent sample of size n_{ri}.

4.2.1 Overview of Basic Methods

Screening

In many cases there will be systems visited by the search that are much worse than others visited by the search. We will use a *subset-selection* procedure to screen out these clearly inferior systems. A subset-selection procedure returns a subset (whose size can be random or predetermined) that contains the best of the k systems with probability $\geq 1-\alpha$. We propose the following procedure when the best system is the one with the largest true mean:

Screen-to-the-best Procedure

1. Let

$$W_{i\ell} = \left(\frac{t_i^2 S_{0i}^2}{n_{0i}} + \frac{t_\ell^2 S_{0\ell}^2}{n_{0\ell}} \right)^{\frac{1}{2}}, \forall i \neq \ell$$

   where $t_i = t_{(1-\alpha)^{\frac{1}{k-1}}, n_{0i}-1}$ and $t_{\beta,\nu}$ is the β quantile of the t distribution with ν degrees of freedom.

2. Set $I = \left\{ i : 1 \leq i \leq k \text{ and } \bar{X}_i^{(1)} \geq \bar{X}_\ell^{(1)} - W_{i\ell}, \forall \ell \neq i \right\}$.

3. Return I as the subset of retained systems.

In a single-stage subset selection procedure, such as this one, the number of systems included in the subset is random. If one is fortunate, the subset includes only a single system, the best. If one is unfortunate, the subset includes all k systems, and the procedure has not reduced the field. Nelson et al. (1998) developed a single-stage subset selection procedure that permits unequal and unknown variances. Our Screen-to-the-best Procedure extends their's to allow the unequal sample sizes that may be the result of a search. A complete description of a slightly more general version of this procedure, and a proof of its validity, are included in the Appendix.

Selection

To choose the best system among those systems that are *not* obviously inferior, we will employ a two-stage *indifference-zone selection* (IZ) procedure, which requires additional sampling of the competitive systems. Two-stage IZ procedures guarantee to select the best system with probability $\geq 1-\alpha$ whenever the best is at least a user-specified amount, δ, better than the others. If there are some near-best solutions within δ of the best, two-stage IZ procedures will return the best or one of these near-best solutions. The user-specified quantity, δ, is called the indifference zone, and it represents the smallest difference worth detecting. In a typical IZ procedure, such as Rinott's (1978) procedure, the total sample size required of each system i is:

$$N_i = \max\left\{n_{0i}, \left\lceil\left(\frac{hS_{0i}}{\delta}\right)^2\right\rceil\right\} \tag{4.1}$$

where $h = h(k, 1-\alpha, n_{0i}-1)$ is a constant determined by k, the number of systems being compared; $1-\alpha$, the desired confidence level; and $n_{0i}-1$, the degrees of freedom in S_{0i}^2. The constant h increases in k, and decreases in α and n_{0i}. Rinott's original paper assumed that the initial sample sizes were equal, but we have extended Rinott's procedure to allow unequal initial sample sizes.

Extended Rinott Procedure

1. Sample $X_{ij}, i = 1, 2, \ldots, k,\ j = 1, 2, \ldots, n_{0i}$, where the X_{ij} are i.i.d. $\mathrm{N}(\mu_i, \sigma_i^2)$ random variables.

2. Set $n_{\min} = \min_i \{n_{0i}\}$.

3. Set $h = h(2, (1-\alpha)^{1/(k-1)}, n_{\min})$.

4. Compute the sample means and variances $\bar{X}_i^{(1)}$ and S_{0i}^2 for $i = 1, 2, \ldots, k$.

5. Determine the total required sample size for each system

$$N_i = \max\left\{n_{0i}, \left\lceil\left(\frac{hS_{0i}}{\delta}\right)^2\right\rceil\right\}.$$

6. Take $N_i - n_{0i}$ additional observations from each system i.

7. Select as best the system i with the largest overall sample mean $\bar{X}_i^{(2)} = \sum_{j=1}^{N_i} X_{ij}/N_i$.

The validity of this extension is proved in the Appendix.

Remark: Notice that the constant $h = h(2, (1-\alpha)^{1/(k-1)}, n_{\text{min}})$ used to determine the second-stage sample size is based on the smallest of the first-stage sample sizes. Further, it is not $h' = h(k, 1-\alpha, n_{\text{min}})$, the standard Rinott constant for comparing k systems. We conjecture that the procedure is still valid if h is replaced by h', but have been unable to prove this (unless all the n_{0i} are equal, in which case the proof is trivial). The constant h is in fact an upper bound on h'; in the Appendix we show that it is a very tight bound, so little is lost in using h rather than h'.

Indifference-zone selection satisfies our overall goal of finding the best or near-best system, but it may be statistically inefficient because it assumes that the systems' true means are arrayed in the "Least Favorable Configuration" (LFC). The LFC is the configuration of system means that would give a procedure the smallest probability of returning the statement "system i is the best" when system i is indeed the true best and is at least δ better than anything else. For IZ procedures, the LFC is typically the "slippage" configuration, which places the true mean of the best system exactly δ larger than the means of all the other systems. Of course, because Nature rarely (if ever) places systems in the LFC, $1-\alpha$ is a lower bound on the probability that the IZ procedure returns a true statement. This makes IZ procedures conservative. We use screening to reduce this problem.

Combining Screening and Selection

Both indifference-zone and subset-selection procedures have shortcomings that hamper their usefulness in a simulation-optimization setting. As mentioned above, a single-stage subset-selection procedure requires no additional simulation effort after the search has finished, but it may not eliminate many (or any) systems. On the other hand, an IZ procedure guarantees to return a single system within δ of the best with a pre-specified probability, but it may require an enormous amount of additional simulation effort to do so. As Equation (4.1) shows, N_i, the total number of replications required for system i, can become quite large, especially if the σ_i^2 are large and δ is small. For this reason, IZ procedures fare best when k, the number of systems in contention, is small. In our environment, however, we may have hundreds or thousands of systems to consider. The simulation effort required to use an IZ procedure alone in such a setting quickly becomes prohibitive.

Fortunately, the two approaches (subset and IZ) can work together to deliver a single system that meets our indifference and probability requirements with less simulation effort than would be required by the IZ procedure alone (Nelson et al., 1998). As mentioned earlier, the IZ procedures assume that the means of the competing systems are arrayed in the LFC. Even after the first-stage data have been collected, the IZ procedures retain the LFC assumption, making no use of the information provided by the first-stage sample means. This strict adherence to the LFC assumption makes the IZ procedures wastefully conservative. Combining a subset-selection procedure with an IZ procedure can reduce this effort by using the first-stage sample data to "screen out" clearly inferior systems. This makes the indifference-zone procedure more efficient by reducing the number of systems on which additional sampling is required.

Restarting

Unfortunately, even when we employ a screening procedure, the fact that we use the initial samples from our search in our IZ procedure means that the constant

h used to determine N_i, the total number of replications required of system i, still depends on the original value of k, the number of systems in contention *before* screening. This means that h remains large. Furthermore, if n_{0i} is quite small, h will also remain large.

If, however, we re-run the first-stage samples of the M systems that survive screening, we can eliminate some of these problems. Restart allows us to use M, rather than the original k, in our determination of h. This could reduce h, perhaps dramatically. Furthermore, restarting gives us an opportunity to increase n_{0i}, which further reduces h. In many cases, the savings gained through these reductions in h more than offset the losses involved in re-running the first-stage samples.

In the next two sections we describe procedures that employ various combinations of screening and selection. The first section describes a simple screen-and-restart procedure, then proposes a method that takes advantage of the variance estimates from the discarded first-stage samples to better choose the size of the restarted first-stage samples. The second section describes a procedure that does not re-run any first-stage samples, but uses second-stage sample information from better systems to increase the chance that they will screen out inferior systems.

4.3 Simple Combination of Screen, Restart, and Select

The combined procedure presented below is simple, yet statistically valid; it employs a subset-selection procedure to screen out inferior systems, then employs an independent IZ procedure on the survivors by taking a new first-stage sample on each.

4.3.1 A Restart Procedure

Consider the following procedure:

Screen, Restart and Select Procedure

1. Select the desired confidence level, $1-\alpha$, and the indifference level, δ. We assume that the first-stage sample sizes and data are given.

2. Run the subset procedure (described below). (To obtain an overall confidence guarantee of $1-\alpha$, we set $1-\alpha_0 = \sqrt{1-\alpha}$ for the screening procedure and $1-\alpha_1 = \sqrt{1-\alpha}$ for the selection procedure.)

   (a) Compute the first-stage sample means and variances, $\bar{X}_i^{(1)}$ and S_{0i}^2, for $i = 1, 2, \ldots, k$. Let

$$W_{i\ell} = \left(\frac{t_i^2 S_{0i}^2}{n_{0i}} + \frac{t_\ell^2 S_{0\ell}^2}{n_{0\ell}} \right)^{\frac{1}{2}} \tag{4.2}$$

where $t_i = t_{(1-\alpha_0)^{\frac{1}{k-1}}, n_{0i}-1}$.

(b) Set $I = \left\{ i : 1 \leq i \leq k \text{ and } \bar{X}_i^{(1)} \geq \bar{X}_\ell^{(1)} - W_{i\ell}, \forall \ell \neq i \right\}$.

(c) Return I, the group of systems that survive the screen, and let $M = |I|$.

3. Take independent samples of size n_{ri} from each system $i \in I$ (discarding the initial first-stage sample), and calculate a new sample variance estimate, S_{ri}^2.

4. Calculate the total required sample size from system i, N_i, as

$$N_i = \max \left\{ n_{ri}, \left\lceil \left(\frac{h S_{ri}}{\delta} \right)^2 \right\rceil \right\} \tag{4.3}$$

where $h = h(2, (1-\alpha_1)^{1/(M-1)}, n_{\min})$ is Rinott's (1978) constant with $k = 2$, confidence level $(1-\alpha_1)^{1/(M-1)}$, and first-stage sample size $n_{min} = \min_{i \in I}\{n_{ri}\}$.

5. Take $N_i - n_{ri}$ additional observations from each system $i \in I$.

6. Of the M surviving systems, select as best the system i with the largest overall sample mean $\bar{X}_i^{(2)} = \sum_{j=1}^{N_i} X_{ij}/N_i$.

Proof of Validity: The Screen, Restart and Select procedure will attain a *correct selection* (CS) if the true best system survives the screen and has the largest second-stage sample mean of the survivors. Let $\bar{X}^{(n)}_{[j]}$ be the n^{th}-stage sample mean of the system whose true mean has antirank j, where larger index indicates a larger value (thus, $[k]$ is the index of the best system). Based on the properties of the screening procedure, we know that the true best system will survive the screen with a probability greater than or equal to $\sqrt{1-\alpha}$. That is,

$$\Pr\left\{\bar{X}^{(1)}_{[k]} \geq \bar{X}^{(1)}_{[i]} - W_{[i][k]}, \forall i \neq k\right\} \geq \sqrt{1-\alpha}.$$

Based on the properties of the selection procedure, we know that if the true best system survives the screen, it will have the largest second-stage sample mean with a probability greater than or equal to $\sqrt{1-\alpha}$ (assuming $\mu_{[k]} - \mu_{[k-1]} \geq \delta$). That is,

$$\Pr\left\{\bar{X}^{(2)}_{[k]} \geq \bar{X}^{(2)}_{[j]}, \forall j \neq k \,\middle|\, k \text{ survives screening}\right\} \geq \sqrt{1-\alpha}.$$

Furthermore, because the procedure takes new first-stage samples between the screening and selection stages,

$$\begin{aligned}
\Pr\{CS\} &= \Pr\left\{\bar{X}^{(1)}_{[k]} \geq \bar{X}^{(1)}_{[i]} - W_{[i][k]}, \forall i \neq k\right\} \times \\
&\quad \Pr\left\{\bar{X}^{(2)}_{[k]} \geq \bar{X}^{(2)}_{[j]}, \forall j \neq k \,\middle|\, k \text{ survives screening}\right\}
\end{aligned}$$

$$\geq \sqrt{1-\alpha}\sqrt{1-\alpha}$$

$$= 1-\alpha.$$

Although the restart procedure has easily provable statistical properties, it is unfortunate that it discards data. If the initial sample size is large or if the screen fails to eliminate many systems, re-running the initial samples becomes wasteful. Nelson et al. (1998) developed a provably valid screen-and-select procedure, which we will call the Screen-and-Continue procedure, that does not require restart. Unfortunately, the validity guarantee for the Screen-and-Continue procedure requires that the critical value h be determined as though all k systems remain in contention, rather than just the M that survive screening.

Remark: We also considered a procedure that postpones the choice between restart and Screen-and-Continue procedures until after we have observed the first-stage data. Under this *choice* procedure, restart is chosen only if it results in a smaller (estimated) total expected number of replications than Screen-and-Continue. Essentially, the choice boils down to the following: Does the reduction in h (due to screening out of systems) under restart make up for the cost of re-running the initial sample of the survivors? Although this procedure sounds questionable, in the Appendix we prove that $\Pr\{CS\} \geq 1-3\alpha/2$. Unfortunately, our experimental results suggest that $\Pr\{CS\} < 1-\alpha$ (but $\geq 1-3\alpha/2$) when systems are arrayed in the least favorable configuration (LFC).

4.3.2 Setting the Initial Sample Size Under Restart

Restart provides an opportunity to reduce a system's *total* required sample size by increasing its *initial* sample size. Recall that n_{0i} is system i's initial sample size as a result of the search, and n_{ri} is i's initial sample size when the sample is re-run. Under our procedure, the total required sample size for system i (after discarding the first n_{0i} replications) is

$$N_i = \max\left\{n_{ri}, \left\lceil\left(\frac{hS_{ri}}{\delta}\right)^2\right\rceil\right\}.$$

The critical value, h, which helps to determine the total sample size, N_i, decreases as the initial sample size increases. The impact on N_i is illustrated in Figure 4.1, which plots N_i as a function of n_{ri} for fixed values of k, α, and $(S_{ri}/\delta)^2$. This argues for increasing n_{ri}, at least up to a point, to decrease N_i. Of course, if n_{ri} is increased too much, it will exceed $\lceil(hS_{ri}/\delta)^2\rceil$, which defeats the purpose. Given that we have seen the results of the first-stage sample, how might we use that information to better set n_{ri}? Suppose we take a simplistic view, by assuming that the effects of rounding are negligible and that $S_{ri}^2 = S_{0i}^2$, where S_{0i}^2 is the sample variance of the first stage for the initial sample and S_{ri}^2 is the sample variance of the first stage for the restart sample.

Let $h_{(n_{ri})}$ represent h as a function of n_{ri} for fixed values of k and α. Notice that N_i is minimized at the point where

$$n_{ri} = \left(\frac{h_{(n_{ri})} S_{ri}}{\delta}\right)^2. \tag{4.4}$$

The graphical analysis of Figure 4.1 applies only to setting n_{ri} for a single system; in our environment, the decision on how to set n_{ri} under restart will be complicated by the fact that we have many systems with unequal variances. Although initial sample sizes need not be equal for an IZ procedure to be *valid*, h is set according to the smallest n_{0i}. Clearly, then, if one system is given a small initial sample size, the benefits of giving other systems greater initial sample sizes are diminished. For this reason, we set a single n_{ri} for use by all systems under restart. To find the single best n_{ri}, we will take advantage of the fact that the line $N_i = n_{ri}$ is convex, and our conjecture that $h_{(n_{ri})}$ is convex.[1] If our conjecture is correct, then $(h_{(n_{ri})} S_{ri}/\delta)^2$ would also be convex. Therefore,

[1] Although we were unable to prove the convexity of $h_{(n_{ri})}$, we believe that it is true based on graphical analysis. See Figure 4.2.

the upper envelope

$$\max\left\{n_{ri}, \left(\frac{h_{n_{ri}} S_{ri}}{\delta}\right)^2\right\}$$

is also convex. Because this is true for every system, the sum

$$\sum_{i=1}^{m}\left\{\max\left\{n_{ri}, \left(\frac{h_{(n_{ri})} S_{ri}}{\delta}\right)^2\right\}\right\}$$

is convex, too. Thus, we can use a search procedure such as the Golden-Section method (Bazaraa, Sherali, and Shetty, 1993) to find the minimum.

The Golden-Section method is an iterative search procedure that narrows a minimum-containing interval with each iteration. We chose this method because it requires only function evaluations (not hard-to-calculate derivatives) and because we only need to narrow the minimum-containing interval to a width of one (since we only consider integer values of n_{ir}). We will postpone discussion of a method for selecting $[n_{low}, n_{high}]$, the minimum-containing interval needed to start the Golden-Section method, to the Appendix.

Suppose we drop the assumption that $S_{ri}^2 = S_{0i}^2$, and assume instead that we estimate the true variance σ_i^2 perfectly in the initial sample (that is, $S_{0i}^2 = \sigma_i^2$). Under this assumption we seek to minimize $E(N_i)$. Figure 4.3 is a plot of $\mathrm{E}(N_i)$ and $\sqrt{\mathrm{Var}(N_i)}$ as functions of n_{ri} for fixed values of k, α, and $(S_{ri}/\delta)^2$. The plots reveal that $\mathrm{E}(N_i)$ decreases rapidly as n_{ri} rises from very low values, and remains relatively flat until n_{ri} begins to overtake $\mathrm{E}(N_i)$. This flatness is encouraging from an implementation standpoint because its suggests that the penalty for being wrong—that is, setting n_{ri} too high or too low—is not great. Furthermore, the plot shows that $\mathrm{Var}(N_i)$ decreases in n_{ri}.

Because we are restarting, the validity of carefully selecting n_{ri} is not in question. The amount of simulation effort saved, however, is less clear. We conjecture that the biggest gains will come when there is a large number of widely spaced systems and initial samples are small. In a search setting, we are very likely to encounter problems of this kind. Section 4.5 provides experimental results that illustrate the performance of the Screen, Restart and Select Procedure.

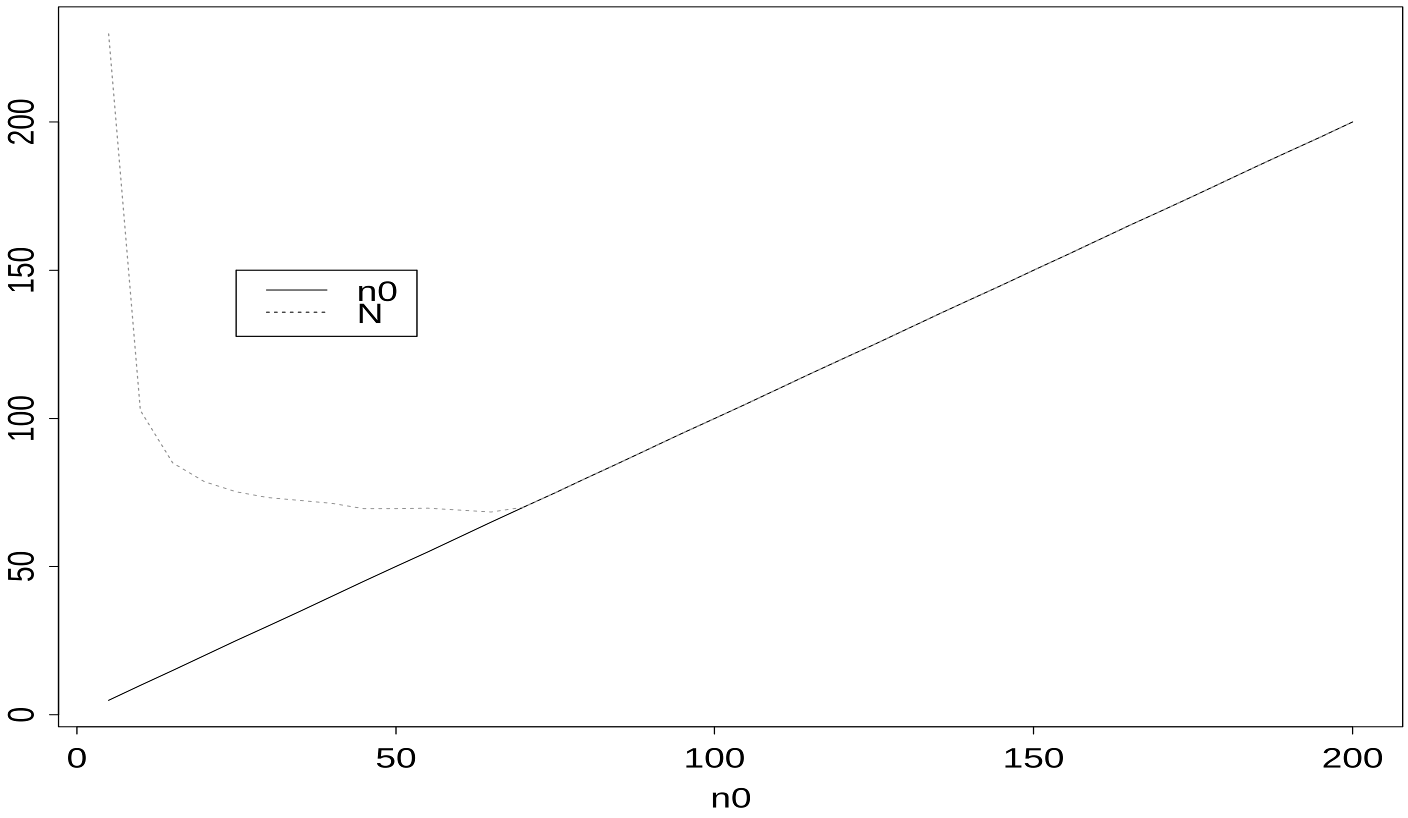

Figure 4.1: Total sample size as a function of initial sample size.

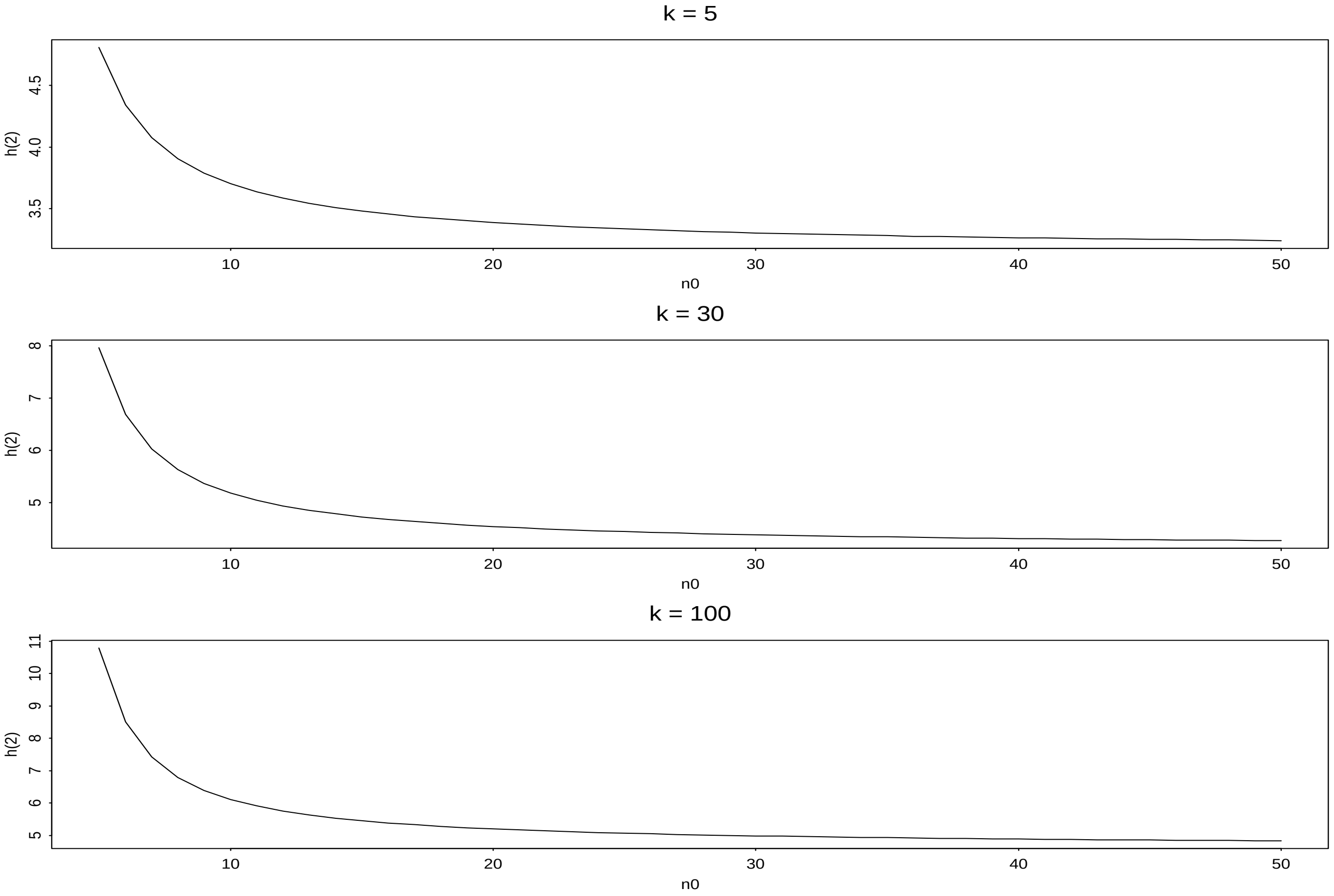

Figure 4.2: h vs. n_0 for selected values of k

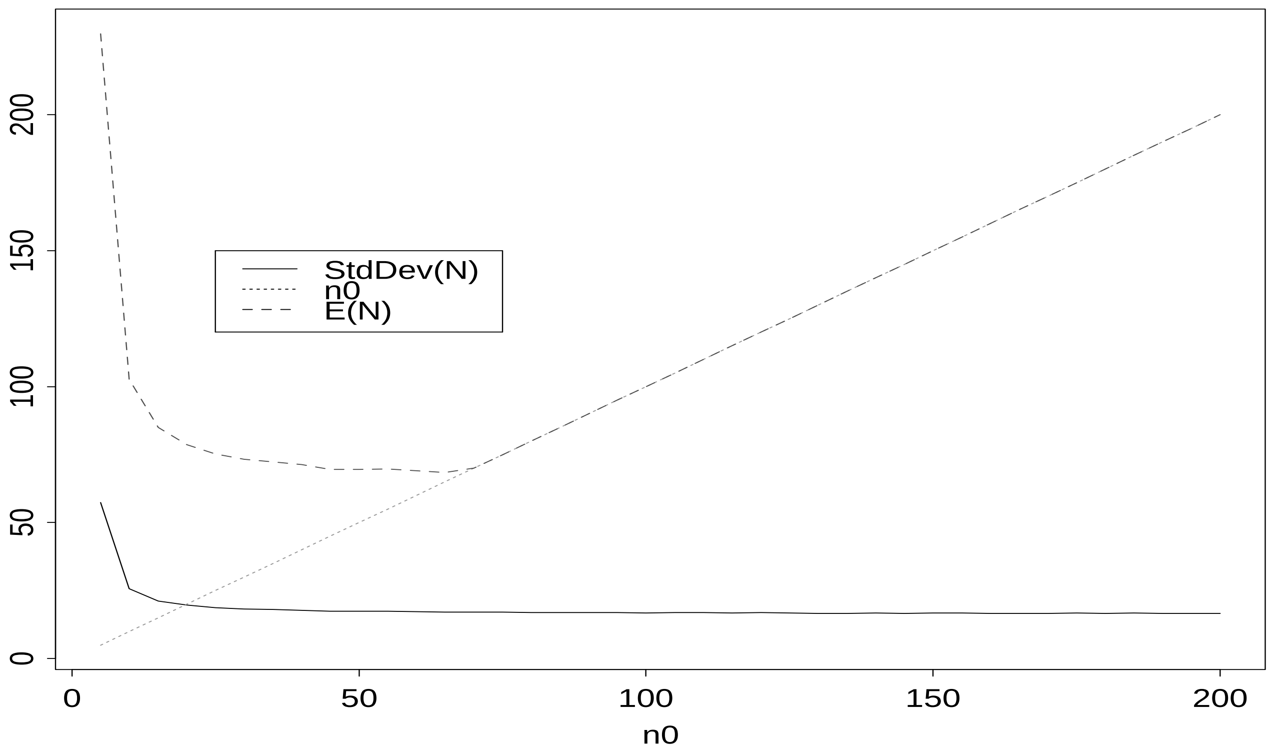

Figure 4.3: Expected total sample size and standard deviation of total sample size as a function of initial sample size.

4.4 Sorting and Group Screening

Nelson et al. (1998) describe and prove the validity of a Group-Screening procedure that uses *second-stage* sample information to screen out inferior systems. Using this information provides a tighter screening procedure than if only first-stage information were used, which means that more systems are eliminated, potentially reducing the total simulation effort required. A modified version of this procedure is presented below. This procedure differs from the one presented in Nelson et al. in the following ways:

- Initial sample sizes, n_{0i}, can vary from system to system;
- Systems that have been eliminated by screening nevertheless continue to act as screeners against subsequent systems;
- The indifference zone, δ, is incorporated *only* in the IZ procedure, not the subset selection procedure. Including δ in both procedures hampers our ability to be sure that the system with the best *sample* mean is within δ of the system with the best *true* mean.

4.4.1 A Modified Group-Screening Procedure

To set up the procedure, let $G_1, G_2, \ldots, G_p$ be groups of systems such that $G_1 \cup G_2 \cup \cdots \cup G_p = \{1, 2, \ldots, k\}$, $G_i \cap G_j = \emptyset$ for $i \neq j$, and $|G_i| \geq 1$ for all i. At the ℓth step in the experiment, the systems in G_ℓ will be screened with respect to each other and all systems from previous groups. However, some of the systems retained from previous steps have already received second-stage sampling, so screening is based on

$$W_{ij} = \left(\frac{t_i^2 S_{0i}^2}{\tilde{N}_i} + \frac{t_j^2 S_{0j}^2}{\tilde{N}_j} \right)^{1/2}$$

where

$$\tilde{N}_i = \begin{cases} n_{0i}, & \text{if system } i \text{ has only received first-stage sampling} \\ N_i, & \text{if system } i \text{ has received second-stage sampling.} \end{cases}$$

The potential savings occur because typically $N_i \gg n_{0i}$, which shortens W_{ij}, providing a tighter screening procedure.

Modified Group-screening Procedure

1. Select overall confidence level $1 - \alpha$ and indifference zone δ. Set $t_i = t_{(1-\alpha/2)^{\frac{1}{k-1}}, n_{0i}-1}$ and $h = h(2, (1-\alpha/2)^{1/(k-1)}, n_{min})$, where $n_{min} = \min_i\{n_{0i}\}$ and h is Rinott's constant.

2. Let $I_0 = \emptyset$ and $J_0 = \emptyset$.

3. Do the following for $\ell = 1, 2, \ldots, p$:

   (a) Compute $\bar{X}_i^{(1)}$ and S_{0i}^2 and set $\tilde{N}_i = n_{0i}$ for all $i \in G_\ell$.

   Comment: G_ℓ is the current group of systems to be screened.

   (b) Let $I_\ell = I^{\text{new}} \cup I_{\ell-1}$, where

$$
\begin{aligned}
I^{\text{new}} = \{ i : i \in G_\ell, \\
& \bar{X}_i^{(1)} \geq \bar{X}_j^{(1)} - W_{ij}, \forall j \in G_\ell \cup J_{\ell-1} \text{ and} \\
& \bar{X}_i^{(1)} \geq \bar{X}_j^{(2)} - W_{ij}, \forall j \in I_{\ell-1} \}
\end{aligned}
$$

   Let $J_\ell = J^{\text{new}} \cup J_{\ell-1}$, where $J^{\text{new}} = \{ i : i \in G_\ell \setminus I^{\text{new}} \}$.

   Comment: I^{new} is the set of newly screened systems that survive the screen, and J^{new} is the set of newly screened systems that are screened out of contention.

   (c) For all $i \in I^{\text{new}}$ compute the second-stage sample size N_i based on Rinott's procedure, that is:

$$
N_i = \max \left\{ n_{0i}, \left\lceil \left(\frac{h S_{0i}}{\delta} \right)^2 \right\rceil \right\}.
$$

(d) Sample $N_i - n_{0i}$ additional observations, and compute the second-stage sample mean $\bar{X}_i^{(2)}$ for all $i \in I^{\text{new}}$. Set $\tilde{N}_i = N_i$.

4. Select as best the system $i \in I_p$ with the largest sample mean $\bar{X}_i^{(2)}$.

In the Appendix, we prove that $\Pr\{CS\} \geq 1 - \alpha$ whenever $\mu_{[k]} - \mu_{[k-1]} \geq \delta$ under this modified procedure.

In Nelson et al., the authors point out that if the procedure happens to encounter a system with a good sample mean early in the process, that system will likely receive second-stage sampling and act as a very tough screen, eliminating many inferior systems. Eliminating these additional systems will reduce the total simulation effort required.

4.4.2 Sorting

Our setting differs slightly from that envisioned in Nelson et al. in that we assume that the first-stage samples from all of our systems are available at the same time. This allows us to *sort* the systems from best to worst based on first-stage sample means. So, rather than *hope* that the procedure happens to encounter a good system early, we can *cause* the procedure to encounter a good system early by sorting. We need to make only two changes to the modified Nelson et al. Group-screening procedure presented above: set group size to one, and sort all the systems by first-stage sample mean. We call this the Sort-and-Screen procedure. In the Appendix, we prove the validity of sorting when group size is one. We conjecture that sorting is also valid for group sizes other than one.

Our experimental results show that sorting can greatly reduce the simulation effort required, while maintaining $\Pr\{CS\} \geq 1 - \alpha$.

4.5 Empirical Evaluation

We conducted an extensive empirical evaluation to compare the procedures introduced in this paper to existing procedures. The systems are represented as various configurations of k normal distributions. In all cases, system 1 was the best (had the largest mean). We evaluated the screening procedures on different variations of the systems, examining factors including the indifference zone, δ; the number of systems, k; and the configuration of the means, μ_i, and the variances, σ_i^2, for $i = 1, 2, \ldots, k$.

In the first set of experiments, the Sort-and-Screen procedure (described in Subsection 4.4.2) is compared to the Modified Group-screening Procedure *without* sorting (described in Section 4.4.1).

In the second set of experiments, the following three procedures are compared:

1. The Screen-and-Continue Procedure, as described in Section 4.2.1.

2. The Screen, Restart and Select Procedure, with the initial sample size under restart held constant, as described in Section 4.3.1.

3. The Screen, Restart and Select Procedure, described in Section 4.3.2, with the initial sample size under restart chosen using the variance information gained during the screening stage.

4.5.1 Configurations

In all cases, the true mean of the best was set at 1. To examine a difficult scenario for the screening procedures, the slippage configuration (SC) of the means was used. In the SC, the mean of the best system was set exactly one indifference zone, δ, above the other systems, and all of the inferior systems had the same mean.

To investigate the effectiveness of the screening procedure in removing non-competitive systems, monotone decreasing means (MDM) were also used. In the MDM configuration, the means of all systems were spaced evenly apart. The size of the spaces between systems were set at δ/τ where $\tau = 1, 2, 3$.

To gauge the effects of sorting by first-stage sample mean, we performed experiments under two different orderings of the means. In one set of experiments we ordered the systems from best to worst, to see how the non-sorting procedure (Modified Group-screening) would perform under the most fortunate circumstances. In another set of experiments we ordered the systems from worst to best, to gauge the performance of the non-sorting procedure under the most unfortunate circumstances.

In some cases the variances of all systems were equal ($\sigma_i^2 = 1$), while in others they differed. For each configuration, we examined the effects of equal and unequal variances on the procedures. In the unequal variance case, the variance of the best system was set both higher and lower than other systems. In the SC, $\sigma_1^2 = \rho\sigma^2$, with $\rho = 0.5, 2$ where σ^2 is the common variance of the inferior systems. In the MDM configurations, experiments were run with the variance directly proportional to the mean of each system, and inversely proportional to the mean of each system. Specifically, $\sigma_i^2 =\mid \mu_i - \delta \mid +1$ to examine the effects of increasing variance as the mean decreases, and $\sigma_i^2 = 1/(\mid \mu_i - \delta \mid +1)$ to examine the effect of decreasing variances as the mean decreases. (Because $\mu_1 = 1$, the lowest means are negative, but have large absolute values.)

4.5.2 Experiment Design

In the Restart experiments, 1000 macroreplications (complete repetitions of the entire experiment) were performed for each configuration. In the Sorting experiments, 500 macroreplications were performed.

In all experiments, the nominal probability of correct selection (PCS) was $1-\alpha = 0.95$. If the procedure's true PCS is close to the nominal level, then the standard error of the estimated PCS, based on 1000 macroreplications, is near $\sqrt{0.95(0.05)/1000}$, which is approximately 0.0069. The standard error of the estimated PCS based on 500 macroreplications is near $\sqrt{0.95(0.05)/500}$, which is approximately 0.0097. Since we are guaranteed that PCS $\geq 1-\alpha$ for normally distributed data, we want to examine how close to $1-\alpha$ we get. If PCS $\gg 1-\alpha$ for all configurations of the means, then the procedure is overly conservative.

The first-stage sample size was set at $n_0 = 10$. The number of systems in each experiment varied over $k = 2, 5, 10, 25, 100, 500$.

4.5.3 Summary of Results

Before presenting specific results, we briefly summarize what was observed from the entire empirical study.

In the Restart experiments, we compared three procedures: Screen-and-Continue with No Restart (NR), Restart with n_0 control (RW), and Restart without n_0 control (RW/O). In all but two instances, RW was more efficient than RW/O, often substantially better. When RW/O was better, it was only slightly better than RW. This suggests that our procedure for finding the best n_0 under restart is useful, but could be improved somewhat.

The NR procedure was often more efficient than the RW/O procedure when the number of systems, k, was 2, 5, or 10. The RW/O procedure was better than the NR procedure when $k \geq 25$. The RW procedure was almost always more efficient than NR for $k \geq 5$; even when $k = 2$, RW was more efficient than NR in about half of the trials.

Despite the improvements gained by adjusting n_0 under restart, these procedures are still conservative; while the nominal PCS was 0.95, the actual PCS was rarely less than 0.99 unless the systems were in the slippage configuration. Under the slippage configuration, the actual PCS was usually between 0.97 and

0.99. The only exception was the RW procedure for $k = 500$ in the common variance case (PCS $= 0.963$) and the case where the inferior systems had lower variance than the best (PCS $= 0.951$).

In the Sorting experiments, we compared three procedures: one was the Sort-and-Screen procedure (S), another was Modified Group-screening with group size of one and no sorting (NS), and the third was Rinott's procedure with no screening. The differences were dramatic; S was vastly superior to NS when the means were encountered in an unfavorable sequence. When the means were encountered in a favorable sequence (best first), the efficiency of S and NS was about equal, although sometimes (for $k = 500$) NS was somewhat more efficient than S. We conjecture that the reason is that S sorts by first-stage sample mean, and this may sometimes place poorer screeners first, while NS always used the true best as the first screener. Rinott's procedure was more efficient than S only when the number of systems was small $k = 2, 5$; that is, when the number of systems eliminated by screening could not make up for splitting α into α_0 for screening and α_1 for selection.

Comparing RW to S and NS yields interesting results. In our experiments, RW was typically better than both S and NS. One could imagine that if the initial sample size, n_0, were much larger than its current value of 10, that this would not be the case. On the other hand, if n_0 were smaller, then RW may look stronger yet.

We will not present comprehensive results from such a large simulation study. Instead, we present details of some typical examples. The performance measures that we estimated in each experiment include the probability of correct selection (PCS), the average number of samples per system (ANS), and the percentage of systems that received second-stage sampling (PSS). Notice that PSS is a measure of the effectiveness of the screening procedure in eliminating inferior systems.

4.5.4 Sorting Experiments

Tables 4.1 and 4.2 show that NS can be disastrous if variances are high and the procedure happens upon a poorly sequenced group of systems. It should be noted, however, that the NS procedure can perform group screening with group size greater than 1, instead of individual screening, which might make it perform better. Empirical results in Nelson et al. (1998) point out that larger groups reduce the penalty for discovering the best system last. On the same problem, Rinott's procedure (without any screening) had an ANS of 44,909 and a PCS of 1.000.

Table 4.1: The effect of screening *with* sorting relative to screening *without* sorting in the MDM configuration with $k = 500$ and $\tau = 1$. In all cases, variance *increases* as sample mean decreases, that is; $\sigma_i^2 = |\mu_i - \delta| + 1$ for all i.

	Favorable Sequencing		Unfavorable Sequencing	
	No Sorting	Sorting	No Sorting	Sorting
ANS	41	54	52,455	55
PSS	2%	2%	100%	2%
PCS	1.000	1.000	0.998	1.000

As Table 4.1 shows, sorting screened out most of the systems with poor means in the Unfavorable Sequencing (PSS = 2%). This was doubly helpful because under this configuration, the solutions with poor sample means also had high variance. In Table 4.2, the inferior systems had much smaller variance, so their elimination had less of an impact. On the same problem, Rinott's procedure (without any screening) had an ANS of 22 and a PCS of 1.000.

Table 4.2: The effect of screening *with* sorting relative to screening *without* sorting in the MDM configuration with $k = 500$ and $\tau = 1$. In all cases, variance *decreases* as sample mean decreases, that is; $\sigma_i^2 = 1/(|\mu_i - \delta| + 1)$ for all i.

	Favorable Sequencing		Unfavorable Sequencing	
	No Sorting	Sorting	No Sorting	Sorting
ANS	14	14	25	14
PSS	1%	1%	100%	1%
PCS	1.000	1.000	0.998	1.000

4.5.5 Restart Experiments

In these experiments we compared NR to RW and RW/O. Our findings can be divided into four categories:

1. SC, many systems (large k): In this case, RW/O was about as efficient as NR, but RW was about twice as efficient.

2. SC, few systems (small k): The effects of restart and adaptive sample sizing were mixed.

3. MDM, many systems (large k): Both RW and RW/O dominated NR, and RW tended to be better than RW/O.

4. MDM, few systems (small k): RW was somewhat better than NR, but NR was better than RW/O.

Tables 4.3 and 4.4 show some of these results. When considering these results, one should keep in mind that the adaptive procedure for finding a good initial sample size under restart takes time, as the constant h must be obtained for a number of different values of n_0.

Table 4.3: Screen-and-Continue vs. Screen-and-Restart, with and without adaptive sample sizing. The systems are in the slippage configuration, and all systems have equal variance.

	Number of Systems, $k = 2$			Number of Systems, $k = 500$		
	Continue	RW	RW/O	Continue	RW	RW/O
ANS	98	91	106	649	273	642
PSS	96%	96%	97%	100%	100%	100%
PCS	0.98	0.97	0.97	0.99	0.96	0.99

Table 4.4: Screen-and-Continue vs. Screen-and-Restart, with and without adaptive sample sizing. The systems are in the MDM configuration, $\tau = 1$ and all systems have equal variance.

	Number of Systems, $k = 2$			Number of Systems, $k = 500$		
	Continue	RW	RW/O	Continue	RW	RW/O
ANS	98	91	106	22	13	14
PSS	96%	96%	97%	2%	2%	2%
PCS	0.98	0.97	0.97	1.00	1.00	1.00

Table 4.5: Sort-and-Screen vs. Screen-and-Restart with adaptive sample sizing. The systems are in the MDM configuration, $\tau = 1$ and all systems have equal variance, as in Table 4.4.

	$k = 2$		$k = 25$		$k = 500$	
	Sort	Restart	Sort	Restart	Sort	Restart
ANS	95	91	66	44	19	13
PSS	92%	96%	18%	24%	1%	2%
PCS	0.970	0.974	0.998	0.995	1.000	0.996

4.5.6 A Comparison of Sort-and-Screen to Restart

So far, we have compared the Sort-and-Screen (S) to the Modified Group-screening procedure without sorting (NS). By and large, S is the better of the two procedures. We also compared the Screen-and-Continue procedure without restart (NR) to both the restart procedure without adaptive n_0 control (RW/O) and the restart procedure with adaptive n_0 control (RW). Of these three procedures, RW seems to be the best. Now, we will compare procedure S to procedure RW.

As one would expect, S, which uses second-stage sample information in

Table 4.6: Sort-and-Screen vs. Screen-and-Restart with adaptive sample sizing. The systems are in the MDM configuration, $\tau = 3$ and variance increases as system mean decreases ($\sigma_i^2 = |\mu_i - \delta| + 1$ for all i).

	$k = 2$		$k = 25$		$k = 500$	
	Sort	Restart	Sort	Restart	Sort	Restart
ANS	104	120	407	254	102	128
PSS	97%	98%	66%	80%	5%	10%
PCS	1.000	1.000	0.998	1.000	1.000	1.000

screening, has a lower PSS than RW. This extra screening power, however, did not make up for the savings the restart procedure gained by lowering h, so RW had the lower ANS.

Of course, for the configuration used for Table 4.5, screening out inferior systems is not such a tough job, so the extra screening power of S is not as critical. If we look at a similar, but more tightly spaced ($\tau = 3$ rather than $\tau = 1$) and highly variable configuration, the situation is less clear cut (see Table 4.6).

There are several factors at play here, and we could devise a configuration in which one or the other procedure would be better. For instance, if first-stage screening is easy, then the additional screening power of S is not so useful. On the other hand, if regular first-stage screening is much less effective than second-stage screening, then restarting will be a waste.

One should note that RW is a bit slow, so if two procedures have equal ANS, S will be faster. Of course, RW could be improved with a better, and faster, way to determine initial sample size under restart.

4.6 Conclusions

In our empirical study, the Restart procedure with adaptive n_r (RW) typically outperformed the Restart procedure without adaptive n_r (RW/O) and the Modified Group-screening procedures (with and without sorting). In those studies the first-stage sample size was $n_0 = 10$, but in a simulation-optimization setting, when a search procedure visits a large number of different solutions, n_0 may be much lower. This strengthens Restart's advantage. Although restart still has the unappealing feature of discarding the initial first-stage data, using the initial first-stage variance information (and the Golden-Section method) to adaptively set the restarted first-stage sample size, n_r, ameliorates this draw-back. Furthermore, the slowness of Restart's current method for finding a good n_r could be improved through use of approximations for h.

Of course, while RW is typically better, if one faces high-variance, closely-spaced systems, and a large number of initial replications, then the Sort-and-Screen procedure may prove superior.

The *choice* procedure, mentioned in Section 4.3.1, delays the choice between the Restart and the Screen-and-Continue procedures until after the first-stage data have been examined. Unfortunately, the price of keeping this option open is a slightly degraded $P\{CS\}$. Still, if one has no inkling about the environment before examining the first-stage data, the *choice* procedure may be worth considering.

4.7 Appendix

This appendix is divided into six sub-sections. In the first two sections we prove the validity of the Screen-to-the-Best Procedure and Extended Rinott's Procedure; in both cases the extensions are to allow unequal initial sample sizes, n_{0i}. The third section shows how we select $[n_{low}, n_{high}]$ to start the Golden-Section search described in Section 4.3.2. The fourth and fifth sections prove the validity of the modified Group-Screening Procedure and sorting variation, respectively. The sixth section provides a lower bound on the probability of correct selection if we choose between Restart and Screen-and-Continue after seeing the first-stage data.

The following lemmas are used in proving the validity of the Screen-to-the-best Procedure and the Group-screening Procedure.

Lemma 1 (Banerjee 1961) *Let Z be a $N(0,1)$ random variable that is independent of $Y_1, Y_2, \ldots, Y_k$, which are independent chi-squared random variables with Y_i having degrees of freedom ν_i. Let $\gamma_1, \gamma_2, \ldots, \gamma_k$ be arbitrary weights such that $\sum_{i=1}^{k} \gamma_i = 1$ and all $\gamma_i \geq 0$. Then*

$$\Pr\left\{Z^2 \leq \sum_{i=1}^{k} t_i^2 \gamma_i \frac{Y_i}{\nu_i}\right\} \geq 1 - \alpha$$

when $t_i = t_{1-\alpha/2,\nu_i}$.

Lemma 2 (Tamhane 1977) *Let $V_1, V_2, \ldots, V_k$ be independent random variables, and let $g_j(v_1, v_2, \ldots, v_k), j = 1, 2, \ldots, p$, be nonnegative, real-valued functions, each one nondecreasing in each of its arguments. Then*

$$E\left[\prod_{j=1}^{p} g_j(V_1, V_2, \ldots, V_k)\right] \geq \prod_{j=1}^{p} E\left[g_j(V_1, V_2, \ldots, V_k)\right].$$

4.7.1 Screen-to-the-best Procedure

We first present a generalization of the Screen-to-the-best Procedure that permits an indifference zone, δ, to be included in the selection criteria, then prove the validity of the general procedure.

1. Sample $X_{ij}, i = 1, 2, \ldots, k; j = 1, 2, \ldots, n_{0i}$, where the X_{ij} are i.i.d. $\mathrm{N}(\mu_i, \sigma_i^2)$ random variables.

2. Compute the sample means and variances $\bar{X}_i^{(1)}$ and S_{0i}^2, for $i = 1, 2, \ldots, k$. Let
$$W_{i\ell} = \left(\frac{t_i^2 S_{0i}^2}{n_{0i}} + \frac{t_\ell^2 S_{0\ell}^2}{n_{0\ell}} \right)^{\frac{1}{2}} \quad \forall\, i \neq \ell$$
where $t_i = t_{(1-\alpha_0)^{\frac{1}{k-1}}, n_{0i}-1}$.

3. Set $I = \left\{ i : 1 \leq i \leq k \text{ and } \bar{X}_i \geq \bar{X}_\ell - (W_{i\ell} - \delta)^+, \forall \ell \neq i \right\}$, where $y^+ = \max\{0, y\}$.

4. Return I.

We now prove that $\Pr\{[k] \in I\} \geq 1 - \alpha$ whenever $\mu_{[k]} - \mu_{[k-1]} \geq \delta$. As a special case, the procedure is valid when $\delta = 0$.

Let $n_{[i]}$ be the number of replications obtained from the system with mean $\mu_{[i]}$. Also let $\delta_{[k][j]} = \mu_{[k]} - \mu_{[j]}$ and

$$v_j = \left(\frac{\sigma^2_{[k]}}{n_{[k]}} + \frac{\sigma^2_{[j]}}{n_{[j]}} \right)^{1/2}.$$

Notice that

$$\begin{aligned}
\Pr\{CS\} &= \Pr\{\bar{X}_{[k]} \geq \bar{X}_{[j]} - (W_{[k][j]} - \delta)^+, \forall j \neq k\} \\
&= \Pr\left\{ \frac{\bar{X}_{[k]} - \bar{X}_{[j]} - \delta_{[k][j]}}{v_j} \geq \frac{-(W_{[k][j]} - \delta)^+}{v_j} - \frac{\delta_{[k][j]}}{v_j}, \forall j \neq k \right\} \qquad (4.5)
\end{aligned}$$

To simplify notation, let

$$Z_j = \frac{\bar{X}_{[k]} - \bar{X}_{[j]} - \delta_{[k][j]}}{v_j}.$$

Using the symmetry of the normal distribution, we can rewrite the right-hand side of (4.5) as

$$\begin{aligned}
& \Pr\left\{ Z_j \leq \frac{(W_{[k][j]} - \delta)^+}{v_j} + \frac{\delta_{[k][j]}}{v_j}, \forall j \neq k \right\} \\
\geq \;& \Pr\left\{ Z_j \leq \frac{W_{[k][j]}}{v_j}, \forall j \neq k \right\}. \qquad (4.6)
\end{aligned}$$

The inequality leading to (4.6) arises because $\delta_{[k][j]} \geq \delta$ under the assumed condition, and $(W_{[k][j]} - \delta)^+ + \delta \geq W_{[k][j]}$.

To further simplify notation, let $Q_j = W_{[k][j]}/v_j$. We now condition on $S^2_{01}, S^2_{02}, \ldots, S^2_{0k}$ and rewrite (4.6) as

$$\begin{aligned}
&\mathrm{E}\left[\Pr\left\{Z_j \le Q_j,\ \forall j \ne k \middle| S^2_{01}, S^2_{02}, \ldots, S^2_{0k}\right\}\right] \\
&\quad \ge \mathrm{E}\left[\prod_{j=1}^{k-1} \Pr\left\{Z_j \le Q_j \middle| S^2_{01}, S^2_{02}, \ldots, S^2_{0k}\right\}\right] \\
&\quad \ge \prod_{j=1}^{k-1} \mathrm{E}\left[\Pr\left\{Z_j \le Q_j\right\}\right] \qquad (4.7)
\end{aligned}$$

where the first inequality follows from Slepian's inequality (Tong, 1980), since $\mathrm{Cov}[Z_i, Z_j] \ge 0$, and the second inequality follows from Lemma 2, since $\Pr\{Z_j \le Q_j\}$ is increasing in Q_j, and Q_j is increasing in $S^2_{01}, S^2_{02}, \ldots, S^2_{0k}$.

To complete the proof, we attack the individual product terms in (4.7). Because $Q_j > 0$ and $Z_j \sim N(0,1)$,

$$\begin{aligned}
\Pr\left\{Z_j \le Q_j\right\} &= \frac{1}{2} + \Pr\left\{0 \le Z_j \le Q_j\right\} \\
&= \frac{1}{2} + \frac{1}{2}\Pr\left\{Z_j^2 \le Q_j^2\right\} \\
&= \frac{1}{2} + \frac{1}{2}\Pr\left\{Z_j^2 \le t^2_{[k]} \frac{S^2_{[k]}}{\sigma^2_{[k]}} \left(\frac{n_{[j]}\sigma^2_{[k]}}{n_{[j]}\sigma^2_{[k]} + n_{[k]}\sigma^2_{[j]}}\right)\right. \\
&\qquad \left. + t^2_{[j]} \frac{S^2_{[j]}}{\sigma^2_{[j]}} \left(\frac{n_{[k]}\sigma^2_{[j]}}{n_{[j]}\sigma^2_{[k]} + n_{[k]}\sigma^2_{[j]}}\right)\right\} \\
&\ge \frac{1}{2} + \frac{1}{2}\left(1 - 2\left(1 - (1-\alpha_0)^{1/(k-1)}\right)\right) \qquad (4.8) \\
&= (1-\alpha_0)^{1/(k-1)}
\end{aligned}$$

where the inequality in (4.8) follows from Lemma 1 with

$$\gamma_1 = 1 - \gamma_2 = \left(\frac{n_{[j]}\sigma^2_{[k]}}{n_{[j]}\sigma^2_{[k]} + n_{[k]}\sigma^2_{[j]}} \right).$$

Substituting this result into (4.7) shows that

$$\Pr\{CS\} \geq \prod_{j-1}^{k-1} (1-\alpha_0)^{1/(k-1)} = 1 - \alpha_0.$$

4.7.2 Extended Rinott Procedure

In his 1978 paper, Rinott showed that if first-stage samples are all of size n_0, then

$$\Pr\{CS\} > \prod_{i=1}^{k-1} \mathrm{E}\left\{ \Phi\left(\frac{h}{\left(\frac{n_0 - 1}{Y_i(n_0-1)} + \frac{n_0 - 1}{Y_k(n_0-1)} \right)^{1/2}} \right) \right\} \geq 1 - \alpha \tag{4.9}$$

where $Y_i(df)$ are independent χ^2_{df} random variables for $i = 1, 2, 3, \ldots, k$, and $h = h(2, (1-\alpha)^{1/(k-1)}, n_0)$.

Following steps analogous to Rinott, we can show that under our modified procedure, where each system i may have initial sample size n_{0i},

$$\Pr\{CS\} > \prod_{i=1}^{k-1} \mathrm{E}\left\{ \Phi\left(\frac{h}{\left(\frac{n_{0i} - 1}{Y_i(n_{0i}-1)} + \frac{n_{0k} - 1}{Y_k(n_{0k}-1)} \right)^{1/2}} \right) \right\} \tag{4.10}$$

where n_{0k} is the initial number of replications performed on the best system.

To prove the validity of our procedure, we need to show that (4.10) $\geq 1-\alpha$. Before we can show this, however, we need to prove a lemma that depends upon the following definitions and theorems from Shaked and Shanthikumar (1994).

Definition: Let X and Y be two random variables such that $\mathrm{E}[\phi(X)] \leq \mathrm{E}[\phi(Y)]$ for *all* convex [concave] functions $\phi : \Re \to \Re$, provided the expectations exist. Then X is said to be smaller than Y in the convex [concave] order, denoted by $X \leq_{cx} Y$ $[X \leq_{cv} Y]$.

Similarly, if $\mathrm{E}[\phi(X)] \leq \mathrm{E}[\phi(Y)]$ for all *increasing* convex [concave] functions $\phi : \Re \to \Re$, then X is said to be smaller than Y in the increasing convex [concave] order, denoted by $X \leq_{icx} Y$ $[X \leq_{icv} Y]$).

Based on the definitions above, it is clear that $X \leq_{cx} Y \Rightarrow X \leq_{icx} Y$. Also notice that $X \leq_{cv} Y \Leftrightarrow Y \leq_{cx} X$.

Definition: Let $a(x)$ be defined on I, where I is a subset of the real line. The number of sign changes of $a(\cdot)$ in I is defined by

$$S^{-}(a) = \sup S^{-}[a(x_1), a(x_2), \ldots, a(x_m)], \tag{4.11}$$

where $S^{-}[y_1, y_2, \ldots, y_m]$ is the number of sign changes of the indicated sequence, zero terms being discarded, and the supremum in (4.11) is extended over all sets

$x_1 < x_2 < \cdots < x_m$ such that $x_i \in I$ and $m < \infty$.

Theorem 6 (Shaked and Shanthikumar, 1994) *If $X_1, X_2, \ldots$ are independent and identically distributed random variables, then for each $n \geq 2$,*

$$\frac{1}{n}\sum_{i=1}^{n} X_i \leq_{cx} \frac{1}{n-1}\sum_{i=1}^{n-1} X_i. \tag{4.12}$$

Theorem 7 (Shaked and Shanthikumar, 1994) *Let X and Y be two random variables with cumulative distribution functions F and G, respectively, and with finite means such that $E[X] \leq E[Y]$. If $S^-(G-F) \leq 1$ and the sign sequence is $+, -$ $[-, +]$ when equality holds, then $X \leq_{icx} Y$ $[X \leq_{icv} Y]$.*

With these theorems and definitions in hand, we will prove the following lemma:

Lemma 3

$$E\left(\Phi\left(\frac{h}{\left(a+\dfrac{r}{Y}\right)^{1/2}}\right)\right) \geq E\left(\Phi\left(\frac{h}{\left(a+\dfrac{r-1}{X}\right)^{1/2}}\right)\right) \tag{4.13}$$

where h and a are positive constants, r is a positive integer, and $Y \sim \chi^2_r$ and $X \sim \chi^2_{r-1}$.

Proof: A random variable that is chi-squared with r degrees of freedom has the same distribution as the sum of the squares of r independent standard normal random variables. In other words, if $Y \sim \chi^2_r$, $X \sim \chi^2_{r-1}$ and $W_1, W_2, \ldots, W_r$ are i.i.d. $\mathrm{N}(0,1)$, then Y/r has the same distribution as $\sum_{i=1}^r W_i^2/r$, and $X/(r-1)$ has the same distribution as $\sum_{i=1}^{r-1} W_i^2/(r-1)$. Therefore by Theorem 6,

$$\begin{aligned} \frac{Y}{r} &\leq_{cx} \frac{X}{r-1} \\ \frac{Y}{r} &\geq_{cv} \frac{X}{r-1} \\ \frac{Y}{r} &\geq_{icv} \frac{X}{r-1}. \end{aligned} \tag{4.14}$$

Therefore, because the function $-1/b$ is concave increasing for all $b > 0$,

$$E\left(-\frac{r}{Y}\right) \geq E\left(-\frac{(r-1)}{X}\right) \tag{4.15}$$

by (4.14) and the definition of increasing concave order. It follows that

$$E\left(\frac{r}{Y}\right) \leq E\left(\frac{r-1}{X}\right) \tag{4.16}$$

and for any constant a

$$E\left(a + \frac{r}{Y}\right) \leq E\left(a + \frac{r-1}{X}\right). \tag{4.17}$$

This result will be used below.

Now let F_1 be the c.d.f. of $X/(r-1)$ and G_1 be the c.d.f. of Y/r. Below we show that the expression $F_1 - G_1$ has just one crossover point, and that its direction is $[+,-]$. Notice that

$$
\begin{aligned}
F_1(y) - G_1(y) &= \int_0^y \left\{ \frac{e^{-x/2} x^{\frac{r-1}{2}-1}}{2^{\frac{r-1}{2}} \Gamma\left(\frac{r-1}{2}\right)} - \frac{e^{-x/2} x^{\frac{r}{2}-1}}{2^{\frac{r}{2}} \Gamma\left(\frac{r}{2}\right)} \right\} dx \\
&= \int_0^y \left\{ 1 - \frac{x^{1/2}}{\sqrt{2} \frac{\Gamma\left(\frac{r}{2}\right)}{\Gamma\left(\frac{r-1}{2}\right)}} \right\} \frac{e^{-x/2} x^{\frac{r-1}{2}-1}}{2^{\frac{r-1}{2}} \Gamma\left(\frac{r-1}{2}\right)} dx
\end{aligned}
$$

Since the function *outside* of the brackets is a chi-squared density, it is always positive. The term *inside* the brackets is positive at $x = 0$, is decreasing for $x > 0$, and must eventually fall below zero. As a result, $F_1 - G_1$ has just one crossover point and the direction is $[+,-]$.

Let F_2 be the c.d.f. of $(r-1)/X$ and G_2 be the c.d.f. of r/Y. The expression $F_2 - G_2$ has the same number of crossover points as $F_1 - G_1$ (one), but has opposite direction, $[-,+]$.

Finally, let F_3 be the c.d.f. of $a + (r-1)/X$ and G_3 be the c.d.f. of $a + r/Y$. The expression $F_3 - G_3$ has the same number of crossover points as $F_2 - G_2$ (one), and has the same direction, $[-,+]$. Combining this fact with (4.17) satisfies the conditions of Theorem 7, so we get

$$
a + \frac{r}{Y} \leq_{icv} a + \frac{r-1}{X}. \tag{4.18}
$$

Thus, since the function $-(b)^{-1/2}$ is concave increasing for all $b > 0$,

$$E\left(-\left(a+\frac{r}{Y}\right)^{-1/2}\right) \leq E\left(-\left(a+\frac{r-1}{X}\right)^{-1/2}\right) \tag{4.19}$$

by (4.18) and the definition of increasing concave order. Therefore

$$E\left(\left(a+\frac{r}{Y}\right)^{-1/2}\right) \geq E\left(\left(a+\frac{r-1}{X}\right)^{-1/2}\right) \tag{4.20}$$

and for any constant $h > 0$

$$hE\left(\left(a+\frac{r}{Y}\right)^{-1/2}\right) \geq hE\left(\left(a+\frac{r-1}{X}\right)^{-1/2}\right) \tag{4.21}$$

which can be rewritten as

$$E\left(\frac{h}{\left(a+\frac{r}{Y}\right)^{1/2}}\right) \geq E\left(\frac{h}{\left(a+\frac{r-1}{X}\right)^{1/2}}\right). \tag{4.22}$$

Recall that if F_2 is the c.d.f. of $(r-1)/X$ and G_2 is the c.d.f. of r/Y, then the expression $F_2 - G_2$ has one crossover point with direction, $[-, +]$. Therefore, the expression $G_2 - F_2$ has one crossover point with direction, $[+, -]$. Let F_4 be the c.d.f. of $(a + (r-1)/X)^{1/2}$, and G_4 be the c.d.f. of $(a + r/Y)^{1/2}$. The expression $G_4 - F_4$ has the same number of crossover points as $G_2 - F_2$ (one), and has the same direction, $[+, -]$.

Next let F_5 be the c.d.f. of $(a + (r-1)/X)^{-1/2}$, and G_5 be the c.d.f. of $(a + r/Y)^{-1/2}$. The expression $G_5 - F_5$ has the same number of crossover points as $G_4 - F_4$ (one), but has opposite direction, $[-, +]$.

Finally, let F_6 be the c.d.f. of $h/\left(a+(r-1)/X\right)^{1/2}$, and G_6 be the c.d.f. of $h/\left(a+r/Y\right)^{1/2}$. The expression $G_6 - F_6$ has the same number of crossover points as $G_5 - F_5$ (one), and has the same direction, $[-,+]$. Thus, because of the number and direction of the crossover points and because of (4.22), Theorem 7 yields

$$\frac{h}{\left(a+\dfrac{r}{Y}\right)^{1/2}} \geq_{icv} \frac{h}{\left(a+\dfrac{r-1}{X}\right)^{1/2}}. \tag{4.23}$$

Because Φ, the c.d.f. of the unit normal distribution, is concave increasing on the interval $(0,\infty)$, the definition of increasing concave ordering gives the desired result

$$E\left(\Phi\left(\frac{h}{\left(a+\dfrac{r}{Y}\right)^{1/2}}\right)\right) \geq E\left(\Phi\left(\frac{h}{\left(a+\dfrac{r-1}{X}\right)^{1/2}}\right)\right). \tag{4.24}$$

□

Now we are in a position to show (4.10) $\geq 1-\alpha$. Let

$$R_i = \Phi\left(\frac{h}{\left(\dfrac{n_{0i}-1}{Y_i(n_{0i}-1)}+\dfrac{n_{0k}-1}{Y_k(n_{0k}-1)}\right)^{1/2}}\right).$$

Notice that each R_i is a non-negative, real-valued function of $Y_1, Y_2, \ldots, Y_k$, and is nondecreasing in each of its arguments. Consider an individual E$[R_i]$ term in the product (4.10) and condition on Y_k. Then

$$E[R_i] = E_{Y_k}\left\{E_{Y_i}\left\{\Phi\left(\frac{h}{\left(\frac{n_i-1}{Y_i(n_i-1)}+\frac{n_k-1}{Y_k(n_k-1)}\right)^{1/2}}\right)\right\}\right\} \tag{4.25}$$

Lemma 3 tells us that (4.25) is

$$\geq E_{Y_k}\left\{E_{Y_i}\left\{\Phi\left(\frac{h}{\left(\frac{n_i-2}{Y_i(n_i-2)}+\frac{n_k-1}{Y_k(n_k-1)}\right)^{1/2}}\right)\right\}\right\}. \tag{4.26}$$

We can continue this process of reducing the degrees of freedom of Y_i to show that (4.26) is

$$\geq E_{Y_k}\left\{E_{Y_i}\left\{\Phi\left(\frac{h}{\left(\frac{n_{min}-1}{Y_i(n_{min}-1)}+\frac{n_k-1}{Y_k(n_k-1)}\right)^{1/2}}\right)\right\}\right\}. \tag{4.27}$$

Applying this process for each $i \neq k$ we obtain

$$\prod_{i\neq k}\mathrm{E}[R_i] \geq \prod_{i\neq k} E_{Y_k}\left\{ E_{Y_i}\left\{\Phi\left(\frac{h}{\left(\frac{n_{min}-1}{Y_i(n_{min}-1)}+\frac{n_k-1}{Y_k(n_k-1)}\right)^{1/2}}\right)\right\}\right\}. \tag{4.28}$$

To complete the proof, we condition instead on Y_i and iteratively reduce the degrees of freedom of Y_k down to $n_{min}-1$ in exactly the same fashion, giving

$$\begin{aligned}\prod_{i\neq k}\mathrm{E}[R_i] &\geq \prod_{i\neq k}\mathrm{E}\left\{\Phi\left(\frac{h}{\left(\frac{n_{min}-1}{Y_i(n_{min}-1)}+\frac{n_{min}-1}{Y_k(n_{min}-1)}\right)^{1/2}}\right)\right\}\\ &\geq \prod_{i\neq k}(1-\alpha)^{1/(k-1)} = 1-\alpha\end{aligned}$$

where the last inequality follows because of the way we choose h. $\square$

Remark: Rinott also proved (4.9) for $h' = h(k, (1-\alpha), n_0)$, which is slightly smaller than our value, $h = h(2, (1-\alpha)^{1/(k-1)}, n_0)$. While we conjecture that (4.10) is true for this smaller h', we were unable to prove it. As Figure 4.4 illustrates, however, the difference between these values is small, about 2%–3% for $k \leq 100$.

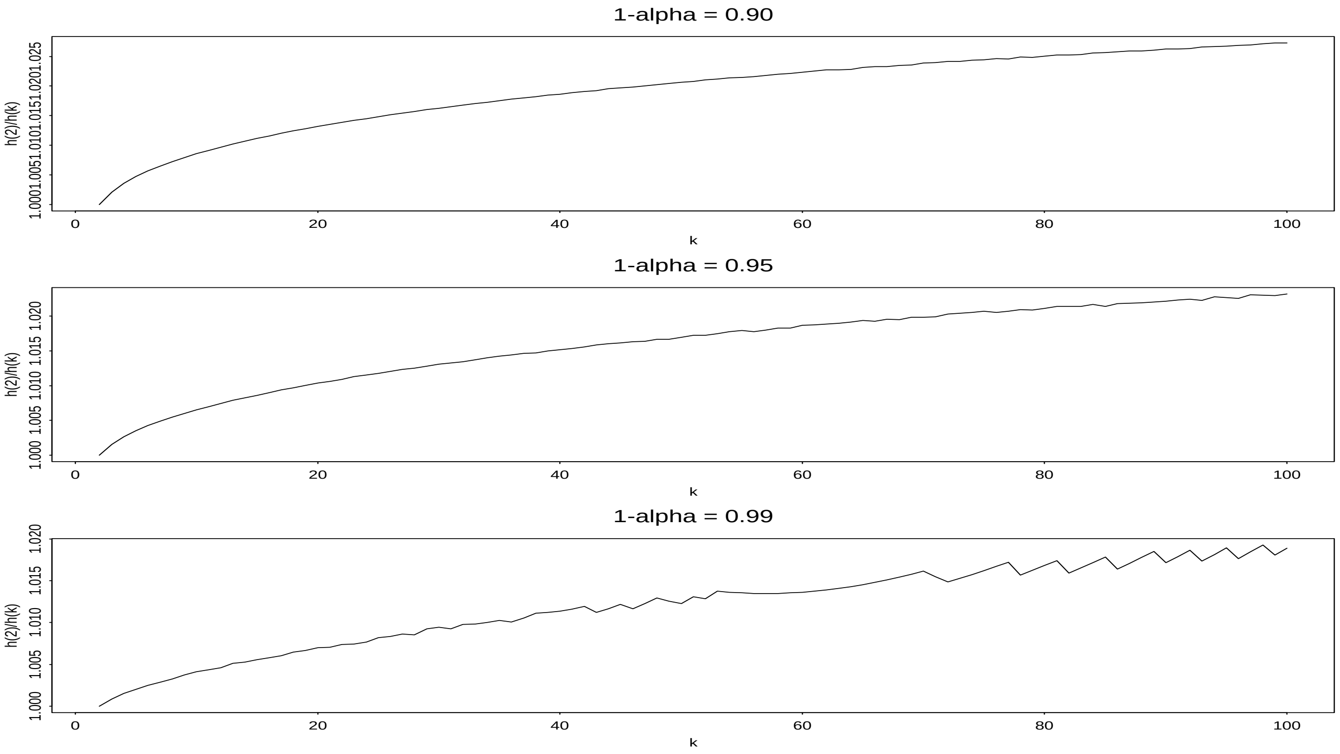

Figure 4.4: $h(2,(1-\alpha)^{1/(k-1)},n_0)/h(k,(1-\alpha),n_0)$ vs. k for $n_0 = 10$ and selected values of $1-\alpha$.

4.7.3 Selecting $[n_{low},\ n_{high}]$ to start the Golden-Section method

Recall that in the single-system case, N_i is minimized at the point where

$$n_{ri} = \left(\frac{h_{(n_{ri})} S_{ri}}{\delta} \right)^2 . \tag{4.29}$$

Let $S^2_{low} = \min_i S^2_{0i}$ and $S^2_{high} = \max_i S^2_{0i}$. If we assume that $S^2_{ri} = S^2_{0i}$, meaning that the sample variance of the initial sample and the restart sample will be the same, then no system will have a minimizing n_{ri} larger than $(h_{(n_{ri})} S_{high}/\delta)^2$, and no system will have a minimizing n_{ri} lower than $(h_{(n_{ri})} S_{low}/\delta)^2$. To set the endpoints of the starting interval, we find the n_{ir}'s that satisfy

$$n_{low} = \left(\frac{h_{(n_{low})} S_{low}}{\delta} \right)^2 \quad \text{and} \quad n_{high} = \left(\frac{h_{(n_{high})} S_{high}}{\delta} \right)^2$$

The minimum will fall within this interval because for any value of n_{ri} outside of the interval, moving a small distance ϵ toward the interval will reduce each N_i, thus reducing the total.

4.7.4 Screening by Non-Competitive ('Dead') Systems

In the original Group-Screening procedure of Nelson et al. (1998), the authors argue that if the true best system, $[k]$, survives screening with probability $\geq$

$1-\alpha_0$, then the entire procedure has $\Pr\{CS\} \geq 1-(\alpha_0+\alpha_1)$. Our Modified Group-screening procedure differs from the original in that each system that is screened must face screening by *all* systems in all previous groups, not just the surviving systems from previous groups; this provides a slightly tighter screen. If we can prove that the true best system survives screening with probability $\geq 1-\alpha_0$ despite this change, then we can claim that the entire procedure has $\Pr\{CS\} \geq 1-(\alpha_0+\alpha_1)$ using the same reasoning as Nelson et al. (1998).

Proof: Let $G_1, G_2, \ldots, G_p$ be groups of systems such that $G_1 \cup G_2 \cup \cdots \cup G_p = \{1, 2, \ldots, k\}$, $G_i \cap G_j = \emptyset$ for $i \neq j$, and $|G_i| \geq 1$ for all i. At the ℓth step in the experiment, the systems in G_ℓ will be screened with respect to each other and all systems from previous groups. Let ℓ^* be the index such that $[k] \in G_{\ell^*}$, and define $F = G_1 \cup G_2 \cup \cdots \cup G_{\ell^*-1}$ (with $F = \emptyset$ if $\ell^* = 1$). Furthermore, define I as all systems from F that survived screening, and $J = F \setminus I$, the systems in F that did not survive screening

Let $\mathcal{E}$ denote the event "system $[k]$ survives screening." Formally,

$$\mathcal{E} = \left\{ \bar{X}_{[k]}^{(1)} \geq \bar{X}_{[j]}^{(1)} - W_{[k]j}, \forall j \in J \cup G_{\ell^*} \text{ and } \bar{X}_{[k]}^{(1)} \geq \bar{X}_{[j]}^{(2)} - W_{[k]j}, \forall j \in I \right\}$$

where each $j \neq [k]$ that is encountered before $[k]$ is either in $J \cup G_{\ell^*}$ or in I, but not both.

Under our procedure, $[k]$ has a better chance of surviving if its group, G_{ℓ^*}, is encountered earlier, rather than later, in the search procedure. That is,

$$\Pr\{\mathcal{E}\big|\ell^* < p\} \geq \Pr\{\mathcal{E}\big|\ell^* = p\}.$$

The reason for this is that when $\ell^* = p$, $[k]$ faces *exactly* the same screens it would face if $\ell^* < p$, plus some more, as long as the allocation of systems to groups and the relative ordering of non-G_{ℓ^*} groups remains unchanged. Thus, if we can prove that $\Pr\{\mathcal{E}\big|\ell^* = p\} \geq 1 - \alpha_0$ over all allocations of systems to groups, then $\Pr\{\mathcal{E}\} \geq 1 - \alpha_0$.

Proof:

$$\begin{aligned}
&\Pr\{\mathcal{E}\big|\ell^* = p\} \\
&\geq \Pr\left\{\frac{\bar{X}^{(1)}_{[k]} - \bar{X}^{(1)}_{j}}{v^{(1,1)}_{j}} \geq \frac{-W_{[k]j}}{v^{(1,1)}_{j}}, \forall j \in J \cup G_{\ell^*} \text{ and} \right. \\
&\qquad \left. \frac{\bar{X}^{(1)}_{[k]} - \bar{X}^{(2)}_{j}}{v^{(1,2)}_{j}} \geq \frac{-W_{[k]j}}{v^{(1,2)}_{j}}, \forall j \in I \right\} \\
&= \Pr\left\{Z^{(1,1)}_{j} \geq Q^{(1,1)}_{j} - \frac{\delta_{[k]j}}{v^{(1,1)}_{j}}, \forall j \in J \cup G_{\ell^*} \text{ and} \right. \\
&\qquad \left. Z^{(1,2)}_{j} \geq Q^{(1,1)}_{j} - \frac{\delta_{[k]j}}{v^{(1,2)}_{j}}, \forall j \in I \right\}
\end{aligned} \tag{4.30}$$

where

$$v^{(1,1)}_{j} = \left(\frac{\sigma^2_{[k]}}{n_{0[k]}} + \frac{\sigma^2_{j}}{n_{0j}}\right)^{\frac{1}{2}} \quad \text{and} \quad v^{(1,2)}_{j} = \left(\frac{\sigma^2_{[k]}}{n_{0[k]}} + \frac{\sigma^2_{j}}{N_{j}}\right)^{\frac{1}{2}}$$

and

$$Z_j^{(a,b)} = \frac{\bar{X}_{[k]}^{(a)} - \bar{X}_j^{(b)} - \delta_{[k]j}}{v_j^{(a,b)}} \quad \text{and} \quad Q_j^{(a,b)} = \frac{W_{[k]j}}{v_j^{(a,b)}}.$$

Now, by symmetry of the normal distribution, and the fact that $\delta_{[k]j} \geq 0$, we can rewrite (4.30) as

$$\Pr\{\mathcal{E}|\ell^* = p\} \geq \Pr\left\{Z_j^{(1,1)} \leq Q_j^{(1,1)}, \forall j \in J \cup G_{\ell^*} \text{ and } Z_j^{(1,2)} \leq Q_j^{(1,2)}, \forall j \in I\right\}. \tag{4.31}$$

The proof proceeds by doing the following steps:

Step 1. Condition on $S_1^2, \ldots, S_k^2$ and use Slepian's inequality to break (4.31) into the product of two probabilities.

Step 2. Use Slepian's inequality and lemma 2 on each piece to break any joint probabilities into products of marginal probabilities for the $Z_j^{(a,b)}$. Altogether there will be $k-1$ terms in the product.

Step 3. Show that each marginal probability is $\geq (1-\alpha_0)^{\frac{1}{k-1}}$ using Banerjee's lemma.

Step 1. In order to apply Slepian's inequality, we must show that, conditional on $S_1^2, \ldots, S_k^2$ and $j \neq \ell \neq [k]$:

$$\mathrm{Cov}[\bar{X}_{[k]}^{(1)} - \bar{X}_j^{(1)}, \bar{X}_{[k]}^{(1)} - \bar{X}_{[}\ell]^{(2)}] \geq 0.$$

This is trivially true since Cov $=$ Var$[\bar{X}^{(1)}_{[k]}] \geq 0$. Therefore, by Slepian's inequality, still conditional on $S_1^2, \ldots, S_k^2$, (4.31) is greater than or equal to

$$\mathrm{E}\left[\Pr\left\{Z_j^{(1,1)} \leq Q_j^{(1,1)},\ \forall j \in J \cup G_{\ell^*}\right\} \Pr\left\{Z_j^{(1,2)} \leq Q_j^{(1,2)},\ \forall j \in I\right\}\right]. \qquad (4.32)$$

Step 2. Within each of the two probability statements above, it is easy to show that the Cov$[Z_j^{(a,b)}, Z_\ell^{(a,b)}] =$ Var$[\bar{X}^{(a)}_{[k]}]/(v_j^{(a,b)} v_\ell^{(a,b)}) \geq 0$, so we can do a further application of Slepian's inequality to show that (4.32) is greater than or equal to

$$E\left[\prod_{j \in J \cup G_{\ell^*}} \Pr\left\{Z_j^{(1,1)} \leq Q_j^{(1,1)}\right\} \prod_{j \in I} \Pr\left\{Z_j^{(1,2)} \leq Q_j^{(1,2)}\right\}\right]. \qquad (4.33)$$

Since each of these probabilities is an increasing function of $Q_j^{(a,b)}$, which is in turn an increasing function of the S_j^2, lemma 2 shows that (4.33) is greater than or equal to

$$\prod_{j \in J} E\left[\Pr\left\{Z_j^{(1,1)} \leq Q_j^{(1,1)}\right\}\right] \prod_{j \in I} E\left[\Pr\left\{Z_j^{(1,2)} \leq Q_j^{(1,2)}\right\}\right]. \qquad (4.34)$$

Step 3. All that remains is to show that

$$E\left[\Pr\left\{Z_j^{(a,b)} \leq Q_j^{(a,b)}\right\}\right] \geq (1 - \alpha_0)^{\frac{1}{k-1}}.$$

This was done for the case $(a, b) = (1, 1)$ in the proof of the Screen-to-the-best Procedure with unequal sample sizes. The proof for the $(1, 2)$ case follows exactly the same steps. Critical to the proof is that W_{ij} is based only on the first-stage sample variances, S_{i0}^2, because S_{i0}^2 is independent of both the first- and second-stage sample means.

4.7.5 Modified Group-Screening with Sorting

Below we will show that under the Modified Group-screening procedure, sorting by first-stage sample mean prior to screening does not alter the $\Pr\{CS\}$ guarantee, provided the group size is one.

Proof: Let set J include only those systems encountered before $[k]$ that fail screening, and let set I include only those systems encountered before $[k]$ that survive screening. Let $\mathcal{E}$ denote the event "system $[k]$ survives screening when no sorting is done." More formally,

$$\mathcal{E} = \left\{ \bar{X}_{[k]}^{(1)} \geq \bar{X}_j^{(1)} - W_{[k]j}, \forall j \in J \text{ and } \bar{X}_{[k]}^{(1)} \geq \bar{X}_j^{(2)} - W_{[k]j}, \forall j \in I \right\}.$$

For a fixed value of $\bar{X}_{[k]}^{(1)}$, system $[k]$ has a better chance of survival if it is encountered earlier, rather than later, in the screening process. That is,

$$\begin{aligned} &\Pr\left\{ \mathcal{E} \middle| \bar{X}_{[k]}^{(1)} = x \text{ and } [k] \text{ is } \textit{not} \text{ encountered last} \right\} \\ &\geq \quad \Pr\left\{ \mathcal{E} \middle| \bar{X}_{[k]}^{(1)} = x \text{ and } [k] \text{ is encountered last} \right\} \end{aligned} \tag{4.35}$$

The assertion above is obvious when group size is equal to one, because if $[k]$ is encountered last, it faces *exactly* the same screens it would face if it were not last, plus some more.

We know that without sorting $\Pr\left\{\mathcal{E}\middle|\ [k] \text{ is encountered last}\right\} \geq 1-\alpha$. However, it is unlikely that this statement is true when sorting is employed, because $[k]$ being encountered last under sorting implies that $\bar{X}_{[k]}^{(1)}$ has a relatively low value. Of course, under sorting, $[k]$ will not always be encountered last. To prove that sorting is valid, we will take an expectation over all possible values of $\bar{X}_{[k]}^{(1)}$, combining the monotonicity of position showed in (4.35) with the fact that $\Pr\left\{\mathcal{E}\middle|[k] \text{ is encountered last}\right\} \geq 1-\alpha$.

Let $f(\cdot)$ be the p.d.f. of the first-stage sample mean from system $[k]$. For any outcome of $\left\{\bar{X}_{[1]}^{(1)}, \bar{X}_{[2]}^{(1)}, \ldots, \bar{X}_{[k-1]}^{(1)}\right\}$, $\bar{X}_{[k]}^{(1)} = x$ implies a position (rank) for $[k]$ under sorting. Let $\mathcal{E}'$ be the same as $\mathcal{E}$ when sorting is employed. Then

$$\begin{aligned}
\Pr\{\mathcal{E}'\} &= \int_{-\infty}^{\infty} \sum_{i=1}^{k} \Bigg(\Pr\left\{\mathcal{E}'\middle|\bar{X}_{[k]}^{(1)} = x \text{ and } [k] \text{ has rank i } \right\} \\
&\quad \times \Pr\left\{[k] \text{ has rank i } \middle|\bar{X}_{[k]}^{(1)} = x\right\} \Bigg) f(x)dx \\
&\geq \int_{-\infty}^{\infty} \Pr\left\{\mathcal{E}\middle|\bar{X}_{[k]}^{(1)} = x \text{ and } [k] \text{ is encountered last } \right\} f(x)dx \\
&= \Pr\left\{\mathcal{E}\middle|\ [k] \text{ is encountered last when sorting is not employed}\right\} \\
&\geq 1-\alpha.
\end{aligned}$$

The inequality holds because for any rank of $[k]$, the probability of surviving screening is not increased by forcing $[k]$ to be encountered last, and if $[k]$ is always encountered last then this is equivalent to the event $\mathcal{E}$ without sorting.

$\square$

4.7.6 Lower Bound on PCS Under Choice Procedure

Suppose that, after screening, we use the first-stage data in conjunction with some decision rule to determine which sub-procedure, Restart or Screen-and-Continue, to pursue. Below, we will prove that under this *choice* procedure $\Pr\{CS\} \geq (1-\alpha_0) \times (1-2\alpha_1)$, where $1-\alpha_0$ is the confidence level used in the screening phase, and $1-\alpha_1$ is the confidence level used in the selection phase.

For notation, let B be the event that the best system survives screening, while D_R is the event that the decision rule—whatever it is—favors restart. A "bar" over an event indicates its complement. Let the subscript C indicate probabilities computed under the assumption that we always continue, while R indicates probabilities computed under the assumption that we always restart.

Using this notation, we can write the probability of a correct selection under the *choice* procedure, given that the best system survives screening, as

$$\Pr\{CS|B\} = \Pr_C\{CS|B, \bar{D}_R\}\Pr\{\bar{D}_R\} + \Pr_R\{CS|B, D_R\}\Pr\{D_R\}. \tag{4.36}$$

First, we will find a lower bound on $\Pr_C\{CS|B, \bar{D}_R\}\Pr\{\bar{D}_R\}$. We know from Nelson et al. (1998) that the probability of selecting the best in the Screen-and-Continue procedure, given that the true best survives screening, is greater than $1-\alpha_1$. In our notation,

$$\Pr_C\{CS|B\} \geq 1-\alpha_1.$$

Conditioning on the outcome of the decision rule yields

$$\Pr_C\{CS|B\} = \Pr_C\{CS|B, D_R\}\Pr\{D_R\} + \Pr_C\{CS|B, \bar{D}_R\}\Pr\{\bar{D}_R\} \geq 1-\alpha_1. \tag{4.37}$$

Therefore,

$$\begin{aligned}\Pr_C\{CS|B, \bar{D}_R\}\Pr\{\bar{D}_R\} &\geq 1-\alpha_1-\Pr_C\{CS|B, D_R\}\Pr\{D_R\} \\ &\geq 1-\alpha_1-\Pr\{D_R\}. \end{aligned} \tag{4.38}$$

We know that under restart, if the best survives screening, the probability of success is greater than or equal to $1-\alpha_1$, regardless of the outcome of the decision rule. As a result, $\Pr_R\{CS|B, D_R\} \geq 1-\alpha_1$. Combining this result with (4.38) yields,

$$\begin{aligned}
\Pr\{CS|B\} &= \Pr_C\{CS|B,\bar{D}_R\}\Pr\{\bar{D}_R\} + \Pr_R\{CS|B,D_R\}\Pr\{D_R\} \\
&\geq 1-\alpha_1-\Pr\{D_R\}+(1-\alpha_1)\Pr\{D_R\} \\
&= 1-\alpha_1-\alpha_1\Pr\{D_R\} \\
&\geq 1-2\alpha_1.
\end{aligned}$$

Consequently, the overall *choice* procedure (screening and selection phases) yields

$$\Pr\{CS\} \geq (1-\alpha_0)\times(1-2\alpha_1) \tag{4.39}$$

which is equal to $(1-\alpha/2)\times(1-\alpha)$ if $\alpha_0=\alpha_1=\alpha/2$. As a result,

$$\Pr\{CS\} \geq 1-3\alpha/2+\alpha^2/2 \geq 1-3\alpha/2. \tag{4.40}$$

Chapter 5

Conclusions and Future Research

The overall goal of this dissertation has been to return good, statistically valid results for unstructured stochastic simulation-optimization problems in a reasonable amount of time. To this end, we have: developed a framework (and written software) that separates the problem into a search phase and a selection phase; adapted heuristic search procedures for use in stochastic environments, and; improved statistical methodology that allows one to select the best of the visited solutions.

Our framework differs from academic approaches to stochastic optimization in that we do not guarantee an optimal solution as the number of replications approaches infinity, but we do provide a statistical guarantee (in the short-term) that the solution returned by our procedure is the best one encountered by the search process. At the other extreme, our framework differs from the approach taken by commercial simulation-optimization packages, which simply ignore stochastic variation. The framework that we have established will be able to incorporate future improvements in search and selection methodology. Some of the more promising avenues of research are described below.

Despite the improvements presented in this disseration, our statistical selection procedures tend to be overly conservative, which means that simulation effort is being wasted. Several improvements might be made to the two-stage procedures we employ. We will apply existing theory that reduces Rinott's constant h for equal initial sample sizes. We may try to prove that unequal n_{0i} can be used to reduce Rinott's constant h; currently, we can use the sample means and variances from unequal n_{0i}, but we must calculate h using $\min_i\{n_{0i}\}$. We will investigate the use of two-stage procedures other than Rinott's,

In addition to making the two-stage procedure less conservative, we plan to develop faster methods for calculating h to make the Golden-Section method more efficient in the RW procedure. We also plan to perform experiments comparing the *choice* procedure to the RW and S procedures.

A number of different search approaches (besides GAs) could be employed in our framework. Most promising are the Evolution Strategies, EAs that allow only the best subset of each population to survive (Bäck, 1996). Evolution Strategies may be easier to adapt to a stochastic environment because their selection procedure meshes more easily with existing statistical grouping procedures than do GA selection procedures.

Other probabilistically driven heuristics, such as simulated annealing, may also be adapted for use in stochastic environments using methods similar to those we used to adapt GAs.

In the disseration, we used heuristics because we did not want to depend upon structure to return a good solution. But just because we do not want to *depend* upon structure to return a good solution does not mean that we do not want to exploit structure if it exists. One could make better use of the database by fitting traditional linear and quadratic models to the data early in the search phase. If the models fit, they could be used to uncover promising areas of the solution space. Such models are likely to be more efficient than the heuristics, if they fit.

Finally, research on the interaction between the search and the selection procedures has the potential to greatly enhance overall performance. For instance, the effects of allowing the search phase to run longer are unclear: it might shorten the selection procedure by uncovering a few superior solutions that can screen out inferior solutions, or it might make the selection procedure longer by generating a large number of nearly equivalent solutions that require second-stage sampling.

References

Aizawa, A. N., and B. W. Wah. 1994. Scheduling of genetic algorithms in a noisy environment. *Evolutionary Computation*, 2(2): 97-122.

Andradóttir, S. 1998. Simulation optimization, Chapter 9 in *Handbook of Simulation* (J. Banks, ed.). New York: John Wiley and Sons.

Bäck, T. 1996. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* New York: Oxford University Press.

Banerjee, S. 1961. On confidence interval for two-means problem based on separate estimates of variances and tabulated values of t-table. *Sankhyā*, A23:359–378.

Bazaraa, M. S., H. D. Sherali, and C. M. Shetty. 1993. *Nonlinear programming: theory and algorithms.* New York: John Wiley & Sons.

Bechhofer, R. E., T. J. Santner and D. Goldsman. 1995. *Design and analysis for statistical selection, screening and multiple comparisons.* New York: John Wiley and Sons.

Boesel, J., and B. L. Nelson. 1999. Designing evolutionary algorithms for stochastic optimization. Technical Report, Dept. of Industrial Engineering and Management Sciences, Northwestern University.

Boesel, J., B. L. Nelson, and N. Ishii. 1999. A Framework for simulation-optimization software. Technical Report, Dept. of Industrial Engineering and Management Sciences, Northwestern University.

Boesel, J., B. L. Nelson and S-H. Kim. 1999. Using ranking and selection to 'clean up' after simulation optimization. Technical Report, Dept. of Industrial Engineering and Management Sciences, Northwestern University.

Calinski, T., and L. C. A. Corsten. 1985. Clustering means in ANOVA by simultaneous testing. *Biometrics*, 41:39–48.

Fitzpatrick, J. M., and J. J. Grefenstette. 1988. Genetic algorithms in noisy environments. *Machine Learning*, 3:101–120.

Hammel, U. 1997. Simulation models. In Bäck, T., Fogel, D. B., Michalewicz, Z., editors, *Handbook of Evolutionary Computation.* F1.8. New York: Oxford University Press.

Hammel, U. and T. Bäck. 1994. Evolution strategies on noisy functions: how to improve convergence properties. In Y. Davidor, H.-P. Schwefel and R. Männer, editors, *Parallel Problem solving from Nature - PPSN III, International Conference on on Evolutionary Computation.* 418-427. Berlin: Springer.

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

Marrison, C. I. , and R. F. Stengel. 1997. Robust control system design using random search and genetic algorithms. *IEEE Transactions on Automatic Control*, 42:835–839.

Miller, B. L. 1997. *Noise, Sampling, and Efficient Genetic Algorithms* (IlliGAL Report No. 97001). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Nelson, B. L., J. Swann, D. Goldsman, and W. Song. 1998. Simple procedures for selecting the best simulated system when the number of alternatives is large. Technical Report, Dept. of Industrial Engineering and Management Sciences, Northwestern University.

Rinott, Y. 1978. On two-stage selection procedures and related probability-inequalities. *Communications in Statistics—Theory and Methods*, A7:799–811.

Shaked, M. and J. G. Shanthikumar. 1994. *Stochastic orders and their applications.* San Diego: Academic Press.

Tamhane, A. C. 1977. Multiple comparisons in model I one-way anova with unequal variances. *Communications in Statistics—Theory and Methods*, A6:15–32.

Tong, Y. L. 1980. *Probability inequalities in multivariate distributions.* New York: Academic Press.