NORTHWESTERN UNIVERSITY

Efficient Simulation in Financial Risk Management

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

Ming Liu

EVANSTON, ILLINOIS

December 2010

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

ABSTRACT

Efficient Simulation in Financial Risk Management

Ming Liu

Assessing the risk of a portfolio is essential both for risk managers to conduct portfolio hedging and for regulators to construct rules, such as how much capital banks should put aside to guard against financial risks they may face. In the past decade, more and more derivative securities were invented corresponding to the increase of the over-the-counter market, and some of them are quite complex. Unfortunately, if a portfolio contains such complex securities, it could be very hard to analyze its risk. Monte-Carlo simulation is a very powerful tool for risk measure estimation of complex derivatives. However, an accurate simulation can take so long that the result is no longer useful when it is delivered because too much time has passed or the portfolio has changed too much. In this dissertation research, the computational efficiency of Monte-Carlo simulation is considered for portfolio risk assessment.

The first phase of the research focuses on efficient two-level simulation for point estimation of expected shortfall. Applying tools from ranking and selection and tools for simulation metamodeling, two different simulation procedures to deal with portfolios with different configurations are proposed. In the second phase of the research, a sequential experiment design

procedure is developed to construct multiple metamodels based on a single stochastic simulation model. This procedure is applied to approximate many securities' prices as functions of a financial scenario.

Acknowledgements

It is a pleasure to thank those who made this thesis possible and because of whom my graduate experience is one that I will cherish forever.

I owe my deepest gratitude to my advisors, Professor Barry L. Nelson and Professor Jeremy Staum for leading me to the wonderful world of financial engineering and computer simulation. Their critical insights and thoughtful guidance have immensely helped me during this thesis work. Their enthusiasm and inspiration have always stimulated me to take steps forward.

I am grateful to my dissertation proposal committee members Professor Bruce E. Ankenman and Professor Shane G. Henderson. The discussions with them were extremely helpful for my last year of research.

I would like to thank Dr. Lisa Goldberg for providing me the opportunity to work as an intern at MSCI Barra and for serving as one of my dissertation committee members. It was a memorable and enjoyable experience working with Lisa, Michael, and Vlad at MSCI Barra, and more importantly, it broadened my perspective on the practical aspects in the financial industry.

I would also like to acknowledge all professors who have taught me in Northwestern University and all graduate students and faculty members in IEMS Department who have given me helps during the last four years.

None of my achievements would have been possible without the love and support of my family. I am indebted to my parents for their understanding, encouragement and unflagging belief in me, especially when it was most required.

There are no words to describe my gratitude to my wife Zigeng Liu, to whom I would like to dedicate this dissertation. I truly thank Zigeng for her dedication, love and persistent confidence in me. Without her, all this would be meaningless.

Table of Contents

CHAPTER 1

Introduction

In financial risk management, the risk of holding a portfolio is always measured as a function of the distribution of this portfolio's value. When the portfolio contains derivative securities an analytical pricing formula may not exist, and so nested simulation may be required for risk measure estimation. In a two-level nested simulation framework, outer-level simulation generates possible future scenarios. These scenarios may arise from historical simulation or Monte Carlo sampling from the distribution of future changes in risk factors. Inner-level simulation of the more distant future, conditional on each scenario, yields an estimate of the portfolio's value, or profit and loss (P&L), in each scenario. The resulting computational burden can be quite large, with thousands of Monte Carlo replications performed in each of thousands of scenarios, for a total of millions of replications. Researchers have developed two approaches to making nested simulation more computationally efficient.

Frye (1998) and Shaw (1998) proposed to reduce computational cost by performing zero inner-level simulation replications in many of the scenarios. In this approach, inner-level simulation occurs only for a set of scenarios called *design points*. These authors estimate the P&L of other scenarios by interpolating among the simulation estimates of P&L at design points.

The other approach is more automated and generic. The earliest work is the thesis of Lee (1998), who studied point estimation of a quantile of the distribution of a conditional expectation. This is related to point estimation of value at risk (VaR): let the portfolio value V in

scenario Z be $V(Z) = \mathrm{E}[X|Z]$, where X is the discounted payoff of the securities in the portfolio and E represents risk-neutral expectation. Lee (1998) discusses how to reduce the mean squared error (MSE) of the point estimator by jackknifing to reduce its bias and by choosing the number of scenarios to sample in an asymptotically optimal way. Gordy and Juneja (2006, 2008) use similar ideas in proposing a simulation procedure for point estimation of a portfolio's VaR via two-level simulation. Expected shortfall (ES) is another widely used risk measure, closely related to conditional value at risk and tail conditional expectation, which is the conditional expectation of loss given that it exceeds VaR. Gordy and Juneja (2008) mention ES but do not provide a simulation procedure for estimating it. A two-level simulation procedure for interval estimation of ES is the topic of Lan et al. (2007a, 2008), who increase computational efficiency by dynamic allocation of the computational budget in multi-stage simulation.

The remainder of the dissertation is organized as follows. First, we use ranking and selection methods to help optimize inner-level computational budget allocation, and this procedure is described in Chapter 2. Second, we adopt the interpolation idea in Frye (1998) and Shaw (1998), and use stochastic kriging (Ankenman et al., 2010) to achieve better simulation design and interpolation taking the simulation uncertainty into account. The details of this second procedure is in Chapter 3. The topic of Chapter 4 is how to estimate many security's prices as functions of a financial scenario. A simulation procedure is proposed for providing an approximate picture of the way the prices of each security change as the market move. Such a picture can be used for assessing and hedging portfolio risks.

CHAPTER 2

Ranking and Selection Procedure for Point Estimation of Expected Shortfall

2.1. Introduction

In this and the next chapter, we focus on point estimation of ES and on the inner level of simulation. Our methods proposed in this chapter are related to those of Lan et al. (2008) and of Lesnevski et al. (2008), who considered another risk measure, but we apply them differently because our goal is efficient point estimation. To get an estimator with low MSE, we create a heuristic simulation procedure. Although we present some justifications for our heuristics based on the assumption that the simulated data are normally distributed, we do not prove anything about the performance of the procedure. We merely craft and explain a simulation procedure, then use experiments with normal and non-normal data to show that it performs well. Our procedure can attain a sufficiently low MSE even when the computational budget is so small that other methods for estimating ES yield answers that are not accurate enough to be useful. We compare our method to a standard two-level simulation of ES, without any efficiency techniques, and to the confidence interval procedure of Lan et al. (2008). We report experimental results in which our procedure delivers root mean squared error (RMSE) between 1% and 10% of the true ES while the RMSE of a standard two-level simulation and the confidence interval width of Lan et al. (2008) are about the same magnitude as ES, indicating that those procedures' answers are not useful.

2.2. Expected Shortfall

Let V be a random variable denoting the value of a portfolio at a future time T. Its cumulative distribution function is denoted by F_V. A risk measure, such as VaR or ES, is a functional $T(F_V)$ of this distribution. The expected shortfall at level $1-p$ is defined as

$$\text{ES}_{1-p} = -\frac{1}{p}\left(\text{E}[V\mathbf{1}_{\{V\le v_p\}}] + v_p(p - \Pr[V \le v_p])\right) \tag{2.1}$$

where v_p is the p-quantile of F_V; $-v_p$ is VaR at the $1-p$ level. In our analysis, we will assume that F_V is continuous at v_p, so that the second term on the right side of Equation (2.1) vanishes, but our procedure works even if this is not so.

Let us suppose we have k scenarios describing the state of the financial markets at time T. Each scenario specifies the levels of a vector Z of risk factors that determine the portfolio's value V. Examples of risk factors are underlying asset prices, volatilities, or interest rates. Define $V_i = \text{E}[X|Z = Z_i]$, the value of the portfolio in scenario i, expressed as a conditional risk-neutral expectation of the total discounted payoff X of the securities in the portfolio. To simplify notation, we let X_i represent a random variable whose distribution is the conditional distribution of X given $Z = Z_i$, so that $V_i = \text{E}[X_i]$, and we refer to X_i as a "payoff." The expectation is estimated by Monte Carlo simulation.

Let π_V be a permutation of $\{1, 2, \ldots, k\}$ such that $V_{\pi_V(1)} \le V_{\pi_V(2)} \le \ldots \le V_{\pi_V(k)}$, that is, scenario $\pi_V(i)$ is the one in which the portfolio value is the ith lowest. Also define γ to be the set of the $\lceil kp \rceil$ portfolios with the smallest values, i.e., $\gamma = \{\pi_V(1), \pi_V(2), \ldots, \pi_V(\lceil kp \rceil)\}$. We use the terms "tail" and "non-tail" to refer to γ and $\{1, 2, \ldots, k\} \setminus \gamma$, respectively. Then ES at

level $1-p$ of the empirical distribution of $V_1, V_2, \ldots, V_k$ is

$$\text{ES}_{1-p} = \sum_{i=1}^{\lceil kp \rceil} w_i V_{\pi_V(i)} \tag{2.2}$$

where

$$w_i = \begin{cases} -1/kp, & \text{for } i = 1, \ldots, \lfloor kp \rfloor, \\ -1 + \lfloor kp \rfloor / kp, & \text{for } i = \lfloor kp \rfloor + 1. \end{cases}$$

The efficient procedure we propose in this chapter focuses on estimating ES as specified by Equation (2.2) when the scenarios are given. The scenarios could be generated by historical data or sampled from a distribution F_V. If we sample them, this represents the outer level of a two-level simulation procedure. The procedure we propose in this chapter focuses on inner-level simulation, estimating the value of the portfolio in each scenario by simulating payoffs. It can be used either with a fixed set of scenarios or as part of a two-level simulation. We will give examples of historical simulation and two-level simulation in §2.5.

2.3. The Standard Procedure

In this section, we present the simplest possible simulation procedure for estimating ES as specified by Equation (2.2). There is a fixed computational budget expressed as a total number C of payoffs that can be simulated. The standard procedure divides the budget equally among the k scenarios and then treats the resulting sample average payoffs as though they were the true values of the scenarios. The procedure is:

(1) Simulate payoffs X_{ih} for $i = 1, 2, \ldots, k$, $h = 1, 2, \ldots, \lfloor C/k \rfloor$. Calculate sample averages $\bar{X}_1, \bar{X}_2, \ldots, \bar{X}_k$.

(2) Select the $\lceil kp \rceil$ smallest sample averages $\bar{X}_{(1)}, \bar{X}_{(2)}, \ldots, \bar{X}_{(\lceil kp \rceil)}$, where the subscript (i) denotes the scenario with the ith smallest sample average).

(3) Estimate ES by

$$\sum_{i=1}^{\lceil kp \rceil} w_i \bar{X}_{(i)}. \tag{2.3}$$

There are two main reasons that this standard procedure does not work well when the budget C is small. First, from Equation (2.3), we see that only $\lceil kp \rceil$ sample averages actually appear in the estimator, which means that only about pC payoffs appear. The other $(1-p)C$ payoffs are used solely to eliminate $k - \lceil kp \rceil$ scenarios. This way of selecting $\lceil kp \rceil$ scenarios to use in the estimator is inefficient. The second reason is that for all $i = 1, 2, \ldots, \lceil kp \rceil$, $\bar{X}_{(i)}$ is a biased estimator of $V_{\pi_V(i)}$, due to selection bias. Selection bias is defined as $\mathrm{E}[\bar{X}_i | i \in \hat{\gamma}] - \mathrm{E}[\bar{X}_i]$, where $\hat{\gamma}$ is the set of scenarios corresponding to the $\lceil kp \rceil$ smallest sample averages. When the budget C is small, the estimator (2.3) can be badly biased.

2.4. An Efficient Procedure

In this section we propose an efficient simulation procedure to estimate expected shortfall. This procedure overcomes the two disadvantages of the standard procedure mentioned above and can give an accurate point estimator of ES when the budget is small.

To avoid spending too much of the budget on scenarios which can be easily excluded from the tail, we follow Lan et al. (2008) and Lesnevski et al. (2008) in using screening. Screening is a method, based on the t-test, that eliminates ("screens out") some scenarios to concentrate computational resources on the scenarios that are most likely to be in the tail γ. We combine the goal of screening in Lan et al. (2008), to screen out all non-tail scenarios, with the highly

efficient screening tactics of Lesnevski et al. (2008), that use multiple stages of screening that terminate when a stopping rule judges that screening is no longer a good use of computational resources. That is, at each stage of the simulation procedure, we simulate more payoffs conditional on all surviving scenarios (the scenarios that we have not screened out yet) and screen out more scenarios that now seem unlikely to be in the tail. Thus, we overcome the first disadvantage of the standard procedure by allocating fewer payoffs to the non-tail scenarios.

We overcome the second disadvantage by avoiding selection bias altogether with a technique called "restarting" (Boesel et al., 2003): we throw out all the payoffs used in screening. After screening, we select a set $\hat{\gamma}$ of scenarios which we believe belong to the tail, and allocate the remaining computational budget to scenarios in $\hat{\gamma}$. We use only the sample averages of these new payoffs in our ES estimator. Those sample averages were not used in the decision about whether or not to include a scenario in $\hat{\gamma}$, which makes $\mathrm{E}[\bar{X}_i | i \in \hat{\gamma}] = \mathrm{E}[\bar{X}_i]$, and then they have no selection bias. This restarting technique is also used in Lan et al. (2008) and Lesnevski et al. (2008). The only source of bias in our procedure comes from the possibility that we may choose $\hat{\gamma}$ incorrectly, i.e., unequal to the true tail γ.

An important difference between our screening procedure and those of Lan et al. (2008) and Lesnevski et al. (2008) is that we dynamically select the error level of the t-tests at each stage. Because of their goal of providing a confidence interval with a minimum guaranteed coverage probability, Lan et al. (2008) and Lesnevski et al. (2008) were restricted to using a pre-specified, very low error level for the t-tests. Our procedure tends to choose higher error levels, thus screening more aggressively and concentrating more of the computational budget on the scenarios whose sample averages are used in the ES estimator.

2.4.1. Outline of the Procedure

We outline our procedure in this section, and elaborate on some steps in subsequent sections. For clarity, we split the procedure into two phases, Phase I and Phase II. Phase I includes multi-stage screening and selection of $\hat{\gamma}$. Phase II allocates the remaining computational budget to the selected scenarios, simulates more payoffs, and computes the ES estimator. Because Phase I contains multiple stages, we use $j = 0, 1, 2, \ldots$ to index the stages.

The user specifies the computational budget C, the sample size n_0 of the first stage, and the rate R at which the cumulative sample size grows from one stage to the next. The computational budget can be chosen based on the time available for the simulation experiment or on experience with the budget required to attain the desired precision. An experiment in §2.5.2 illustrates that it is not difficult to choose good values of n_0 and R, and leads to the recommendation of $n_0 = 30$ and $R = 1.2$ for most simulation problems.

Define I_j to be the set of scenarios that survive to the beginning of stage j and N_j to be the cumulative number of payoffs simulated for each scenario in I_j after stage j, so $N_0 = n_0$. Given the sample size growth factor R, $N_j = N_{j-1}R$ for $j \geq 1$. Let $\bar{X}_i(j)$ be the sample average of scenario i after stage j, i.e., $\bar{X}_i(j) = N_j^{-1} \sum_{h=1}^{N_j} X_{ih}$. Let $\pi_j(\cdot)$ be a mapping of $\{1, 2, \ldots, |I_j|\}$ to I_j such that $\bar{X}_{\pi_j(1)}(j) \leq \bar{X}_{\pi_j(2)}(j) \leq \cdots \leq \bar{X}_{\pi_j(|I_j|)}(j)$. That is, for any $i = 1, 2, \ldots, |I_j|$, $\pi_j(i)$ is the scenario with the sample average that is ith lowest after stage j among the scenarios in I_j. Let C_j be the remaining budget at the beginning of stage j, and J be the index of the last screening stage in Phase I, as determined by the stopping rule. Let α_j be the error level of each t-test at stage j, which we refer to as the error level for screening at stage j. An outline of our procedure follows; for the full details, see Appendix A.1. Figure 2.1 contains a flowchart illustrating the procedure.

Initialization.: Set $N_0 \leftarrow n_0$, $I_0 \leftarrow \{1, 2, \ldots, k\}$, $C_0 \leftarrow C$, and $j \leftarrow 0$.

Phase I.:

(1) If $j > 0$, set $n_j \leftarrow N_j - N_{j-1}$. Simulate n_j payoffs for each scenario in I_j using common random numbers (CRN; see Law and Kelton, 2000) to sharpen screening in Step 3. Calculate the remaining budget $C_{j+1} \leftarrow C_j - |I_j| n_j$.

(2) Choose the error level for screening, α_j (§2.4.3).

(3) **Screening:** Screen to compute I_{j+1}, the set of scenarios that survive screening after stage j (§2.4.2).

(4) If the stopping rule is not satisfied (§2.4.5), then set $j \leftarrow j + 1$ and go to Step 1.

(5) **Selection:** Set $J \leftarrow j$ and $\hat{\gamma} \leftarrow \{\pi_J(1), \pi_J(2) \ldots, \pi_J(\lceil kp \rceil)\}$.

Phase II.: Restart, allocate the remaining computational budget to scenarios in $\hat{\gamma}$, and compute the ES estimator (§2.4.4).

2.4.2. Screening

In this section we present the screening method given the target error level α_j at stage j; we will show how to choose α_j in §2.4.3. For all ordered pairs (i, r) in $I_j \times I_j$, we consider a t-test of the hypothesis that $V_i \leq V_r$ at significance level α_j. If this hypothesis is rejected, we say scenario i is "beaten" by scenario r, i.e., i is beaten by r if and only if

$$\bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha_j, N_j - 1} S_{ir}(j)}{\sqrt{N_j}}$$

where $t_{1-\alpha_j, N_j-1}$ is the $1 - \alpha_j$ quantile of the t-distribution with $N_j - 1$ degrees of freedom,

$$S_{ir}^2(j) = \frac{1}{N_j - 1} \sum_{h=1}^{N_j} \left(X_{ih} - X_{rh} - \left(\bar{X}_i(j) - \bar{X}_r(j) \right) \right)^2$$

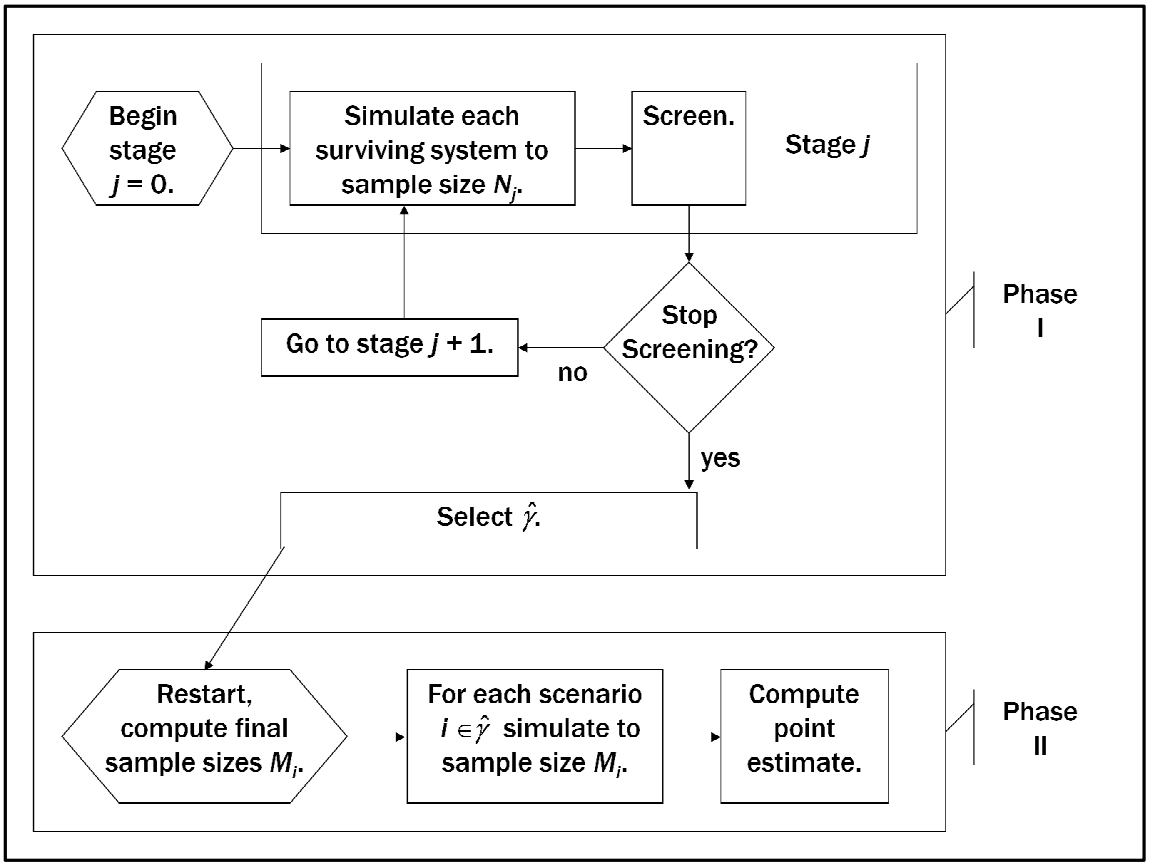

Figure 2.1. A flowchart representing our procedure.

is the sample variance of $X_i - X_r$, and $\bar{X}_i(j)$ is the sample average of $X_{i1}, X_{i2}, \ldots, X_{iN_i}$. Scenarios beaten at least $\lceil kp \rceil$ times are screened out, therefore

$$I_{j+1} = \left\{ i : \sum_{r \in I_j} \mathbf{1}\left\{ \bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha_j, N_j - 1} S_{ir}(j)}{\sqrt{N_j}} \right\} < kp, i \in I_j \right\}.$$

The use of the t-test is motivated by the observation that, if N_j is sufficiently large, $(\bar{X}_i(j) - \bar{X}_r(j) - (V_i - V_r))\sqrt{N_j}/S_{ir}$ is approximately Student t distributed (Henderson, 2006). For convenience in analysis, we will treat each payoff X_i as though it were normal. The adequacy of this assumption of normality in a closely related procedure was evaluated in Lesnevski et al.

(2008). Furthermore, our procedure neither provides a confidence interval nor guarantees a minimum probability of correctly identifying the tail, so the t-tests here do not need to be valid. We merely use them as a tool for decreasing the MSE of our point estimator given a fixed computational budget.

2.4.3. Error Level for Screening

The purpose of the stopping rule (§2.4.5) is to make sure that enough of the computational budget is left for Phase II to accurately estimate the values of the scenarios selected in Phase I, so in choosing the error level α_j for screening at stage j, we only consider how this affects the quality of the set $\hat{\gamma}$ of scenarios that we select in Phase I. In particular, define CS $:= \{\hat{\gamma} = \gamma\}$ to be the event of selecting γ at the end of Phase I. We would like to choose $\alpha_0, \alpha_1, \ldots, \alpha_J$ to maximize $\Pr\{\text{CS}\}$, the probability of correct selection. Unfortunately, this maximization problem is too hard to solve, primarily because we cannot express $\Pr\{\text{CS}\}$ in a useful form to allow it to be optimized over the error levels.

However, the principle behind the existence of an optimal choice of α_j is clear. A small α_j means screening cautiously at stage j, not screening out many scenarios, but having a low probability of mistakenly screening out a scenario that really belongs to the tail γ; a large α_j means screening aggressively, screening out many scenarios, but with a larger probability of mistakenly screening out tail scenarios. If $\alpha_0, \alpha_1, \ldots, \alpha_J$ are too big, we are very likely to make screening mistakes. If we do, some scenarios in γ will not be in I_{J+1}, i.e., will not survive screening, which will prevent us from making a correct selection at the end of Phase I. If $\alpha_0, \alpha_1, \ldots, \alpha_J$ are too small, we will probably not screen out many scenarios before we use up so much of the computational budget that we have to go to Phase II, when we must select

exactly $\lceil kp \rceil$ scenarios. If the number $|I_{J+1}|$ of scenarios that survive to this point is too large, each one has a small sample size N_J because the computational budget was depleted after a small number $J+1$ of stages. Then we will be forced to choose among many scenarios on the basis of sample averages that have high variance, because their variances are inversely proportional to N_J. This implies a large probability of making selection mistakes at the end of Phase I. In other words, by being too cautious during screening, we would waste much of the computational budget on scenarios that we should have been bold enough to eliminate. Then we would quickly find ourselves in a situation in which we would be forced to guess, on the basis of inadequate information, the identities of the tail scenarios from among a large set of scenarios. We will attempt to choose a moderate α_j that balances the risks of screening mistakes during Phase I and selection mistakes at the end of Phase I.

Our method chooses $\alpha_0, \alpha_1, \ldots, \alpha_J$ dynamically, on the basis of an approximation to $\Pr\{\text{CS}\}$ that is updated at every stage of screening. We choose the error level α_j at the end of stage j, just before screening. To simplify the problem, we assume while choosing α_j that this error level will be used in screening at the current stage j and all future stages. This is not how our procedure actually works: at stage $j+1$ we will choose α_{j+1} on the basis of new information, and α_{j+1} is generally not the same as α_j. However, the assumption relieves us of the need to consider $\alpha_{j+1}, \alpha_{j+2}, \ldots, \alpha_J$ while choosing α_j, which would be difficult to do.

To choose α_j, we would like to maximize the probability $\Pr\{\text{CS}_j\}$ of selecting all tail scenarios that have survived to stage j: $\text{CS}_j := \{\gamma \cap I_j \subseteq \hat{\gamma}\}$. Unfortunately, we can not write $\Pr\{\text{CS}_j\}$ as an explicit function of α_j; we have to replace it with some sort of approximation. Our approach is to use a forecast of the behavior of our procedure in later stages to construct

the following approximation to $\Pr\{\text{CS}_j\}$ when the error level is α:

$$\tilde{P}(j,\alpha) = \frac{(1-\lceil kp\rceil\,\alpha)^{\tilde{J}(j,\alpha)-j+1}}{\binom{|\tilde{I}(j,\alpha)|}{\lceil kp\rceil}} \tag{2.4}$$

where $\tilde{J}(j,\alpha)$ is the forecasted final stage of Phase I and $\tilde{I}(j,\alpha)$ is the forecasted set of scenarios that will survive screening after stage $\tilde{J}(j,\alpha)$. The procedure for making these forecasts is described in Appendix A.2.1.1. We choose α_j to maximize $\tilde{P}(j,\alpha)$ instead of $\Pr\{\text{CS}_j\}$, which we can not compute. The derivation of $\tilde{P}(j,\alpha)$ is in Appendix A.2.1. Briefly, the numerator is related to the probability that none of the tail scenarios are screened out in stages $j, j+1, \ldots, \tilde{J}(j,\alpha)$, while the reciprocal of the denominator is related to the probability of correctly choosing $\lceil kp\rceil$ scenarios out of the $|\tilde{I}(j,\alpha)|$ scenarios that are forecasted to survive screening.

Figure 2.2 illustrates how the scenarios' sample averages and the error level for screening α_j change during a single run of the procedure. At many stages, α_j is quite low, because the procedure judges that the number of surviving scenarios is small compared to the remaining computational budget. The same low level is chosen for α_j at many stages because we chose α_j using a search algorithm (see §A.1) that confines the search to a grid, and this level is the smallest in the grid. At other stages, such as 6, 9, and 21, the procedure judges that there are too many surviving scenarios compared to the remaining budget, so it increases the screening error level α_j and screens out many scenarios. In this run of the procedure, after stage 21, there are only 11 scenarios left, while we must select $kp = 10$. However, even though the 11th scenario is not screened out, the stopping rule takes until stage 32 to decide that screening is no longer worthwhile. This run is atypical; in replications of this example, screening usually stops when

only 10 scenarios remain. We chose to present an atypical run because its later stages show that the error level α_j selected by the procedure can vary greatly depending on the remaining budget and the current sample averages, even when the number of surviving scenarios does not change.

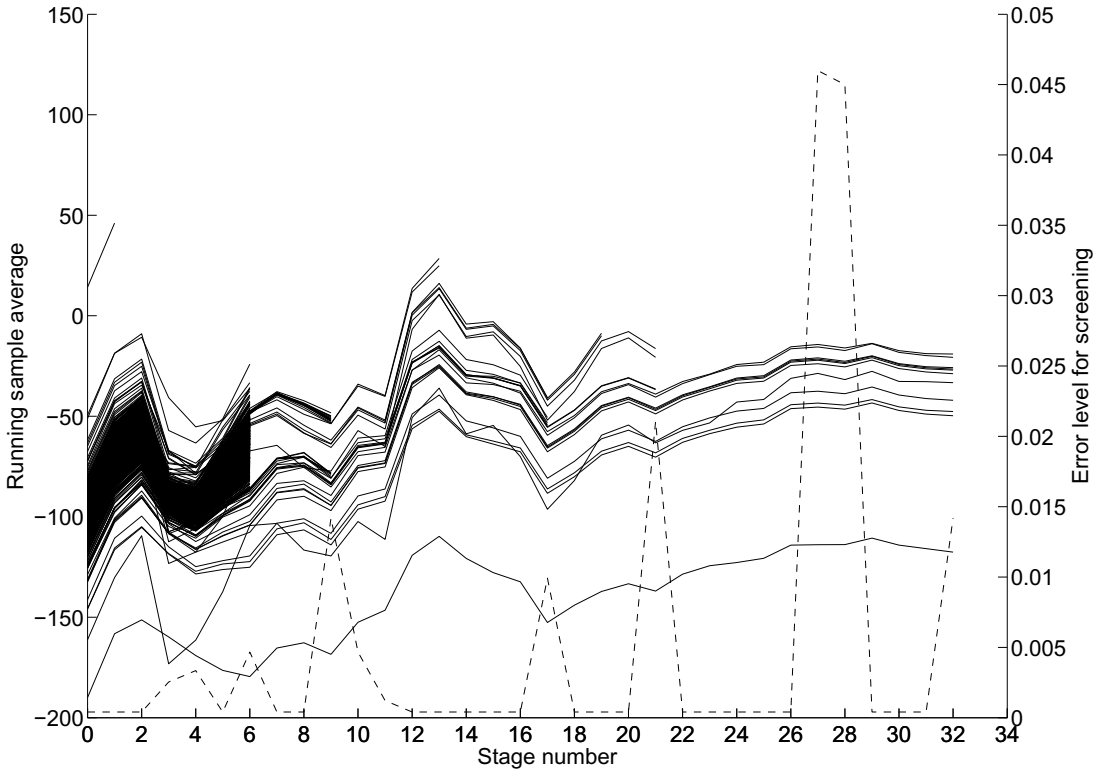

Figure 2.2. Multi-stage screening during one run of our procedure on the historical simulation example (§2.5.2). Solid lines represent sample averages of surviving scenarios. The dashed line is the error level for screening.

2.4.4. Allocating the Remaining Budget to Compute the Estimator

In this section, we describe Phase II of the procedure. After restarting, it is necessary to allocate the remaining computational budget to scenarios in $\hat{\gamma}$. We do this so as to minimize the variance

of the ES estimator. First we describe Phase II simulation and the ES estimator, then derive the optimal allocation.

Conditional on each scenario $i \in \hat{\gamma}$, we simulate M_i payoffs in Phase II and calculate the sample average $\bar{X}_i$. Because we do not do any comparisons between scenarios in Phase II, CRN is not used; typically, independent sampling leads to a lower variance for the ES estimator

$$\widehat{ES}_{1-p} = \sum_{i=1}^{\lceil kp \rceil} w_i \bar{X}_{\pi_J(i)}. \tag{2.5}$$

Now we consider how to choose the Phase II sample size M_i for $i \in \hat{\gamma}$. Because we use restarting, the bias of the ES estimator only comes from the possibility of a wrong selection $\hat{\gamma} \neq \gamma$ in Phase I. The bias does not depend on Phase II sample sizes, so if we want to minimize the MSE we only need to minimize the variance. The variance of $\widehat{ES}_{1-p}$ is

$$\text{Var}(\widehat{ES}_{1-p}) = \text{Var}\left(\sum_{i=1}^{\lceil kp \rceil} w_i \bar{X}_{\pi_J(i)}\right) = \sum_{i=1}^{\lceil kp \rceil} w_i^2 \frac{\sigma^2_{\pi_J(i)}}{M_{\pi_J(i)}}, \tag{2.6}$$

where $\sigma^2_{\pi_J(i)} = \text{Var}[X_{\pi_J(i)} | \pi_J(i)]$ is the conditional variance of the payoff given that the scenario is $\pi_J(i)$. Notice that, conditional on Phase I, $\text{Var}(\widehat{ES}_{1-p})$ is not a random variable. Since we do not know $\sigma^2_{\pi_J(i)}$, we use the sample variance $S^2_{\pi_J(i)}(J)$ of the N_J samples in Phase I instead. Then we consider the optimization problem

$$\min \sum_{i=1}^{\lceil kp \rceil} w_i^2 \frac{S^2_{\pi_J(i)}(J)}{M_{\pi_J(i)}} \quad \text{s.t.} \quad \sum_{i=1}^{\lceil kp \rceil} M_{\pi_J(i)} = C_{J+1}.$$

Using the Karush-Kuhn-Tucker (KKT) condition, the optimal M_i is

$$M_{\pi_J(i)} = C_{J+1} \frac{w_i S_{\pi_J(i)(J)}}{\sum_{r=1}^{\lceil kp \rceil} w_r S_{\pi_J(r)}(J)}. \tag{2.7}$$

2.4.5. Stopping Rule

At the end of each stage in Phase I, our procedure has to decide whether to go on screening, continuing Phase I, or to stop screening, select $\hat{\gamma}$, and end Phase I. Because of restarting, we do not want to continue Phase I too long, or we will throw out a lot of simulated payoffs, leaving too small a computational budget for Phase II, which will produce a high-variance estimator. On the other hand, if we end Phase I too soon, when it is not yet clear which scenarios belong to the tail, a large bias arises because we are likely to select $\hat{\gamma}$ badly. In this section we give a stopping rule for Phase I that balances these considerations.

We focus on the decision whether to stop Phase I after stage j, when I_{j+1} has just been computed. If $|I_{j+1}| = \lceil kp \rceil$, there is no need to do any more screening, so we stop. Otherwise, we approximate the MSE of the ES estimator if we stop now and if we continue, then make the decision that leads to the smallest MSE. In approximating the MSE if we stop now, our procedure is pessimistic about the bias of the ES estimator. In approximating the MSE if we continue, our procedure is optimistic in believing that only scenarios belonging to the tail will survive screening at stage $j+1$, and that these are the scenarios with the smallest conditional payoff variances σ^2 of all scenarios in I_{j+1}. Because we are optimistic about the next stage of screening and pessimistic about stopping, our procedure tends to continue screening when the remaining computational budget C_{j+1} is sufficiently large. As the remaining computational budget shrinks, the variance of the ES estimator grows, and this eventually forces the procedure to stop Phase I to save enough budget for ES estimation in Phase II. We adopt this idea of being pessimistic about stopping and optimistic about continuing because it performed well in Lesnevski et al. (2008).

Because of restarting, a Phase II sample average $\bar{X}_{\pi_J(i)}$ is an unbiased estimator of $V_{\pi_J(i)}$. Thus the bias of our ES estimator defined in Equation (2.5) is

$$\begin{aligned} \text{Bias}(\widehat{ES}_{1-p}) &= \mathrm{E}\left[\widehat{ES}_{1-p}\right] - ES_{1-p} \\ &= \sum_{i=1}^{\lceil kp \rceil} w_i \left(\mathrm{E}\left[\bar{X}_{\pi_J(i)}\right] - V_{\pi_V(i)}\right) \\ &= \sum_{i=1}^{\lceil kp \rceil} w_i \left(V_{\pi_J(i)} - V_{\pi_V(i)}\right). \end{aligned} \tag{2.8}$$

From the definition of w_i, $\pi_V(\cdot)$, and $\pi_j(\cdot)$, it follows that Bias$(\widehat{ES}_{1-p})$ is negative.

When we consider whether to stop screening after stage j, we can split the bias into two parts: the bias from screening mistakes up to stage j and the bias from any screening or selection mistakes after stage j. The bias due to screening up to stage j is the same whether we stop or continue after stage j, so we ignore it in formulating the stopping rule and only consider the bias due to screening or selection mistakes after stage j. To simplify matters, we suppose $\gamma \subseteq I_{j+1}$, that is, no screening mistakes have occurred so far. Given our optimistic view of continuing, we suppose there will be no screening or selection mistakes after stage j if we continue, producing zero bias. If we stop after stage j, the only bias comes from selection mistakes due to $|I_{j+1}| > \lceil kp \rceil$: if we select $\hat{\gamma}$ now on the basis of N_j samples from each surviving scenario, it may not be the same as γ. Consistent with our pessimistic approach to evaluating the decision to stop, we consider the following approximate lower bound for the bias (which is negative) due to stopping after stage j:

$$B(j) = \sum_{i=1}^{\min\{\lceil kp \rceil, |I_{j+1}| - \lceil kp \rceil\}} w_i \max_{\delta \geq 0} \delta \Phi\left(-\delta \sqrt{N_j}/\tau_j\right)$$

where $\Phi(\cdot)$ is the standard normal distribution function and $\tau_j = \max\{S_{ir}(j) : i, r \in I_{j+1}, i \neq r\}$. Details of the derivation are in Appendix A.2.2.

We estimate the variance of the ES estimator if we stop after stage j by

$$V_s(j) = \sum_{i=1}^{\lceil kp \rceil} \frac{w_i^2 S^2_{\pi_j(i)}(j)}{M_{\pi_j(i)}} = \frac{1}{C_{j+1}} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_j(i)}(j) \right)^2 .$$

The second equality follows from Equation (2.7) and $J = j$. Our pessimistic approximation to the MSE of the estimator if we stop after stage j is

$$MSE_s(j) = B^2(j) + V_s(j).$$

To analyze the variance if we continue, we optimistically suppose that the set of scenarios which will survive after one additional stage of screening is exactly γ, and that they have the smallest variances among the scenarios in I_{j+1}. According to this optimistic assumption, we will stop after stage $j+1$, our ES estimator will have zero bias, and its variance is estimated by

$$V_c(j) = \frac{1}{C_{j+1} - (N_{j+1} - N_j)|I_{j+1}|} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_{S(j)}(i)}(j) \right)^2$$

where $\pi_{S(j)}(\cdot)$ is a mapping of $\{1, 2, \ldots, |I_{j+1}|\}$ to I_{j+1} such that $S_{\pi_{S(j)}(1)}(j) \leq S_{\pi_{S(j)}(2)}(j) \leq \cdots \leq S_{\pi_{S(j)}(|I_{j+1}|)}(j)$; i.e., $S_{\pi_{S(j)}(i)}(j)$ is the ith smallest sample standard deviation among the scenarios surviving stage j. Our optimistic approximation to the MSE of the estimator if we continue after stage $j+1$ is

$$MSE_c(j) = V_c(j).$$

The stopping rule is: if $|I_{j+1}| = \lceil kp \rceil$ or $MSE_s(j) < MSE_c(j)$, select $\hat{\gamma}$ and go to Phase II, otherwise continue with stage $j+1$ of screening in Phase I. This rule determines when Phase I ends, but we also use it while choosing the screening error level α_j (§2.4.3) to forecast when Phase I will end. When we use the stopping rule for that purpose, we plug the forecasted sample averages, sample variances and sets of surviving scenarios into the MSE expressions given above.

Figure 2.3 shows how the stopping rule works on the same run of our procedure shown in Figure 2.2. The pessimistic approximation of MSE if we stop drops steeply at stages 13, 17, and 21, as the number of surviving scenarios gets close to $kp = 10$. As mentioned previously and illustrated in Figure 2.2, this run is atypical in that 11 scenarios survive from stages 21 to 32. On this run, optimism that the sole surviving non-tail scenario will be screened out is not borne out. Both estimates, MSE_c and MSE_s, of MSE rise after stage 21 as the computational budget is spent without achieving anything, but MSE_c rises faster because it includes the effects of continuing for one more extra, larger stage of screening. When it catches up to MSE_s, Phase I ends and the procedure selects the 10 scenarios with the lowest sample averages in Figure 2.2. On this run, 53% of the budget was spent in Phase I.

2.5. Experimental Results

We test the performance of our procedure on three examples. The first example features artificial configurations of $k = 1000$ scenarios in which the payoffs have heavy-tailed Pareto distributions. We vary a parameter that controls the difficulty of screening and selection and illustrate that our procedure attains lower MSE than the standard procedure (§2.3) for all values of the parameter that we considered. The second example is of a portfolio of eight call options,

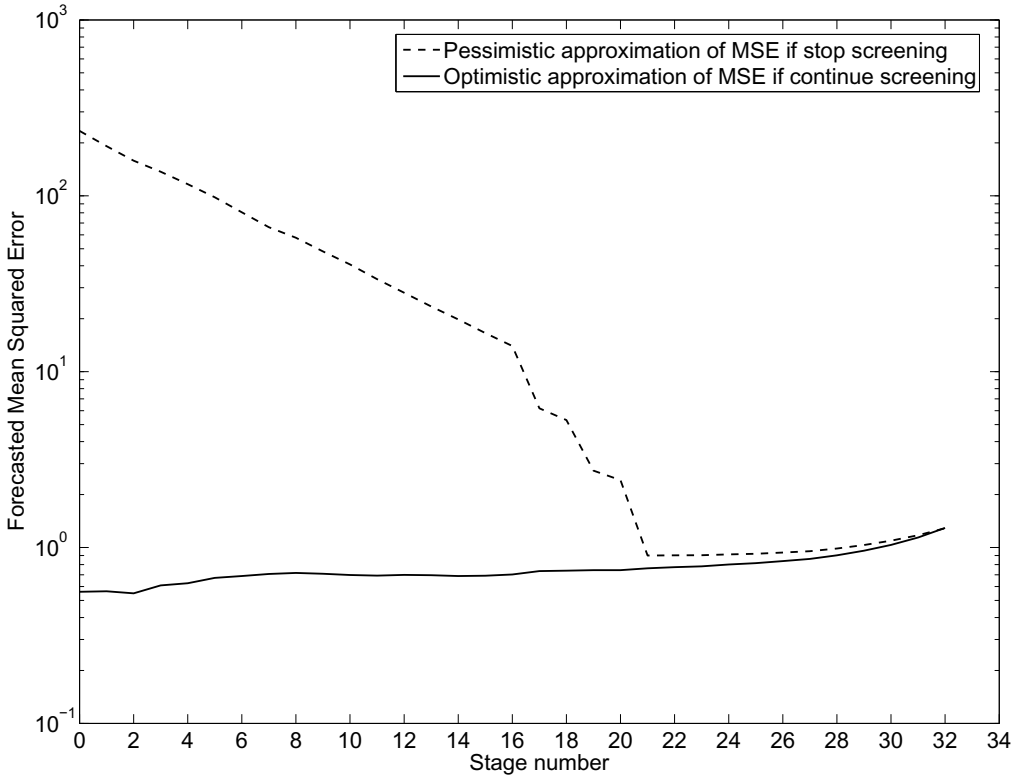

Figure 2.3. Operation of the stopping rule during one run of our procedure on the historical simulation example (§2.5.2).

with 1000 scenarios based on historical stock prices. The third example is similar to the second, but it is a two-level simulation, with scenarios sampled in an outer-level simulation, and our procedure governing the inner-level simulation. Using this example, we compare our procedure with the standard procedure and with the confidence interval procedure of Lan et al. (2008).

We compare the precision of the estimators these procedures produce given a computational budget expressed in total payoffs simulated. This comparison is not entirely fair because it excludes the overhead of screening, choosing error levels for screening, etc. Excluding the overhead of our procedure and that of Lan et al. (2008) is unfavorable to the standard procedure.

This issue is addressed experimentally by Lan et al. (2008), who also report comparisons in which there is a fixed budget of computing time.

2.5.1. Artificial Configuration Example

The artificial configuration of scenarios in this example is motivated by the "slippage configuration" used in the ranking and selection literature because it is difficult for screening and selection procedures (Kim and Nelson, 2006). In this configuration all tail scenarios have payoffs with a common distribution, while all non-tail scenarios' payoffs have a different common distribution. To make screening even more difficult, the payoffs of different scenarios are independent, so that common random numbers achieve nothing. In particular, if scenarios i and r are both in the tail γ, then $V_i = V_r$; and the $V_i = V_r$ also if neither i nor r are in the tail. If $i \in \gamma$ while $r \notin \gamma$, then $V_r = V_i + \delta$. The parameter δ governs the difficulty of screening and selection: when δ is small, it is difficult to distinguish tail from non-tail scenarios, so it will be hard to screen out scenarios and easy to make selection mistakes. On the other hand, the bias induced by selection mistakes will be small. By changing δ, we can compare our procedure to the standard procedure for a range of configurations with different characteristics.

Pareto distributions are often used to model heavy-tailed loss distributions. Using a heavy-tailed distribution challenges our procedure, which was designed with normally distributed data in mind. We use the Pareto distribution with cumulative distribution function $F(x) = 1 - (\lambda/(\lambda + x))^{2.5}$ for $x \geq 0$. The shape parameter is 2.5 and the scale parameter λ is 25 for tail scenarios, while for non-tail scenarios it is either 25.5, 25.875, 26.25, 26.625, 27, 27.75, or 28.5. The resulting values of δ, the difference between tail and non-tail scenarios' values, are 0.33, 0.58, 0.83, 1.08, 1.33, 1.83, or 2.33. There are $k = 1000$ scenarios and we estimate

$ES_{0.99}$, so there are $kp = 10$ tail scenarios. This example is simple enough that we can compute $ES_{0.99} = 16.67$, which makes it easier to determine the MSE of the simulation procedures.

Figure 2.4 shows the root mean squared error (RMSE) of estimating $ES_{0.99}$ for the standard procedure and our procedure. RMSE was estimated by running 1000 macro-replications of the simulation experiment, and the error bars represent the resulting 95% confidence interval for RMSE. In these experiments, the computational budget C is 4 million payoffs, the initial sample size $n_0 = 300$, and the sample size growth factor $R = 1.2$. From Figure 2.4 we see that as δ decreases, the RMSE of the standard procedure increases. The reason is that its selection bias increases: when the tail and non-tail scenarios are similar, it is very likely that some of the 990 non-tail scenarios will have sample averages that are less than the value of the tail scenarios and will be selected into $\hat{\gamma}$, the set of scenarios the procedure guesses are in the tail. Because our procedures eliminates selection bias by restarting, it gives a much more accurate point estimator when δ is small. When δ is big, our procedure outperforms the standard procedure because it allocates the computational budget more efficiently. In this experiment, our procedure always yields an RMSE below 0.44, which is small compared to the true $ES_{0.99} = 16.67$ and to the standard deviation of the tail scenarios' payoff distribution, which is 37.27.

2.5.2. Historical Simulation of an Options Portfolio

Next we consider a more realistic example, in which we estimate the ES of a portfolio of call options on Cisco (CSCO) and Sun Microsystems (JAVA), as shown in Table 3.1. The position given in Table 3.1 is the number of shares of stock the option contract is written on, and a negative value means a short position. Except for the position data, which we made up, all other data in this table comes from listed option prices on June 26, 2007. The risk-free rates come

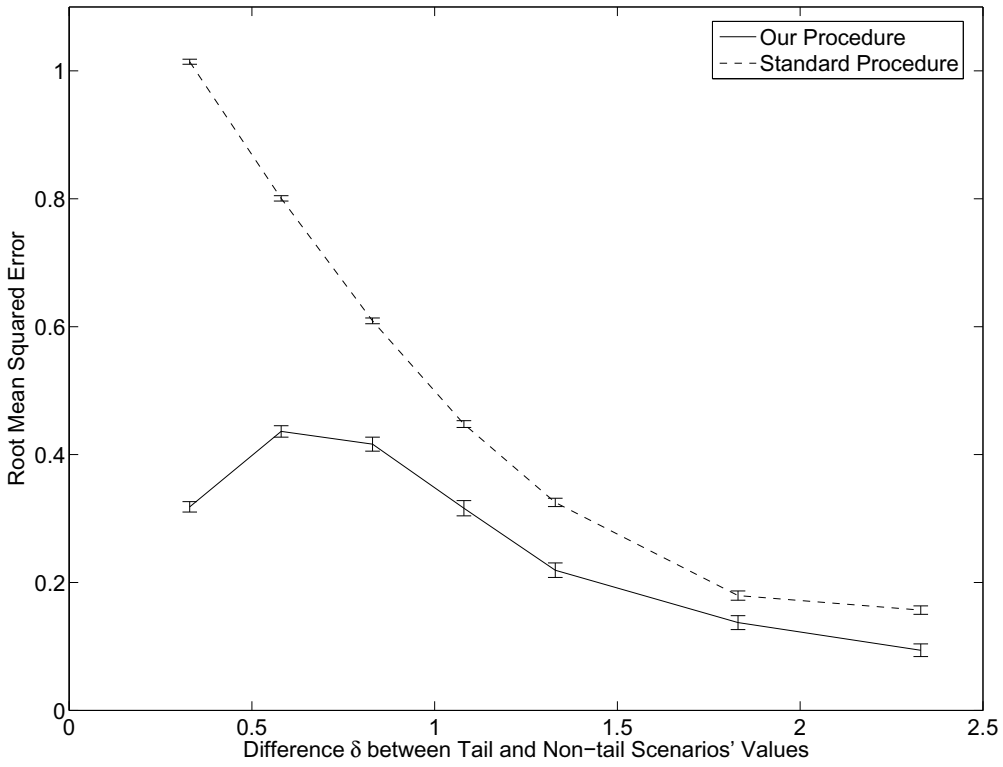

Figure 2.4. Root mean squared error of estimating expected shortfall at the 99% level, in artificial configurations of varying difficulties, with computational budget $C = 4$ million, first-stage sample size $n_0 = 300$, and sample size growth factor $R = 1.2$.

from the yields of US treasury bonds with the same maturities as the options. We estimate the ES of this portfolio's value on June 27, 2007 given information up to June 26, 2007: in this example, $T = 1$ day. We use historical simulation, getting $k = 1000$ scenarios from the daily returns on CSCO and JAVA stock, based on their closing prices from July 07, 2003 to June 26, 2007. A scenario consists of the stock prices of CSCO and JAVA on June 27, 2007, created by multiplying their prices on June 26, 2007 (respectively \$27.15 and \$5.01) by one plus their respective returns on a day in the historical data set.

Table 2.1. Portfolio of Call Options.

Underlying Stock	Position	Strike	Maturity (years)	Price	Risk Free Rate	Implied Volatility
CSCO	200	\$27.5	0.315	\$1.65	4.82%	26.66%
CSCO	-400	\$30	0.315	\$0.7	4.82%	25.64%
CSCO	200	\$27.5	0.564	\$2.5	5.01%	28.36%
CSCO	-200	\$30	0.564	\$1.4	5.01%	26.91%
JAVA	600	\$5	0.315	\$0.435	4.82%	35.19%
JAVA	1200	\$6	0.315	\$0.125	4.82%	35.67%
JAVA	-900	\$5	0.564	\$0.615	5.01%	36.42%
JAVA	-300	\$6	0.564	\$0.26	5.01%	35.94%

To determine how the portfolio value $V(\cdot)$ depends on the scenario, we need to specify how option values at time T depend on the scenario. We assume that the implied volatilities of these options obey the sticky-strike rule (Derman, 1999). That is, the implied volatilities of these options at time 0 are also the implied volatilities at time T. The simulation procedures estimate option i's value at time T by simulating the stock price S_i at maturity U_i as

$$S_i = \frac{S_T^{(j_i)}}{D_i} \exp\left(-\frac{1}{2}\sigma_i^2(U_i - T) + \sigma_i\sqrt{U_i - T}Z_i\right)$$

where the index j_i is 1 for the four options on CSCO and 2 for the four options on JAVA, σ_i is the implied volatility of option i, D_i is a discount factor from times T to U_i, and Z_i is a standard normal random variable independent of Z_j for $j \neq i$. In Monte Carlo simulation of the payoff, which is a weighted sum of payoffs for all options in the portfolio, we simulate the payoffs of each option independently. Common random numbers are used for simulating the payoff of the same option in different scenarios, but not for simulating the payoffs of different options.

Since neither stock pays dividends, early exercise is never optimal for these call options (Luenberger, 1998), so the Black-Scholes pricing formula can be used to evaluate the call option values at time T. Therefore, this example is also simple enough that the portfolio value in each scenario and thus ES can be calculated analytically, which helps in evaluating the MSE attained by simulation procedures: $\text{ES}_{0.99} = \$52.24$ and $\text{ES}_{0.95} = \$26.18$.

Table 2.2 shows the performance of the standard procedure and our procedure in estimating $\text{ES}_{0.99}$ and $\text{ES}_{0.95}$. As in the previous example, the computational budget C is 4 million payoffs, the initial sample size $n_0 = 300$, and the sample size growth factor $R = 1.2$. The table also provides the standard error (SE) of estimating each RMSE with 1000 macro-replications. The RMSE of our procedure is significantly smaller than the RMSE of the standard procedure, both in statistical and practical terms. If we define the relative RMSE as the ratio of RMSE to the ES being estimated, we find the relative RMSEs of our procedure for $\text{ES}_{0.99}$ and $\text{ES}_{0.95}$ are 1.9% and 5.7%, respectively, whereas the standard procedure yields RMSE that is about the same size as ES. Given this budget, our procedure provides moderate accuracy, while the standard procedure provides answers that are not useful and indeed misleading because they are extremely badly biased. It is surprising to see that our procedure delivers a lower RMSE when estimating $\text{ES}_{0.99}$ than for $\text{ES}_{0.95}$, because it is usually thought to be more difficult to estimate ES deeper in the tail. The primary reason for the surprising result here is that, given this set of 1000 scenarios, it is relatively easy to distinguish the 10 scenarios with the worst losses from the others, but it is not as easy to distinguish the 50 scenarios with the worst losses—for example, the 10th worst loss of \$31.72 is widely separated from the 11th worst loss of \$28.56, but the 50th and

Table 2.2. Comparison of our procedure with the standard procedure for historical simulation of a portfolio of stock options, with computational budget $C = 4$ million, first-stage sample size $n_0 = 300$, and sample size growth factor $R = 1.2$.

	Method	Variance	Bias	RMSE	SE of RMSE
$\text{ES}_{0.99}$	Standard Procedure	23.5	36.7	37.1	0.15
	Our Procedure	0.93	-0.01	0.97	0.02
$\text{ES}_{0.95}$	Standard Procedure	5.0	35.4	35.4	0.07
	Our Procedure	2.21	≈ 0	1.49	0.04

51st worst losses are separated by less than \$0.05, and there are 9 tail scenarios and 13 non-tail scenarios closely packed between the 42nd worst loss of \$16.03 and the 63rd worst loss of \$14.41.

We also tested the sensitivity of our procedure's performance with respect to the first-stage sample size n_0 and the sample size growth factor R, which the user must choose. For estimating $\text{ES}_{0.99}$ with computational budget $C = 4$ million, first we fixed $R = 1.2$, and varied n_0. As long as n_0 was between 30 and 1300, RMSE was below 1.11, not far from the best RMSE the procedure attains for any value of n_0. When n_0 was increased past 1300, RMSE increased: it is inefficient to spend a third or more of the computational budget in the first stage, before any scenarios can be screened out. These findings are similar to those of Lesnevski et al. (2007), and we likewise recommend choosing n_0 to be quite small, but large enough that the first-stage sample averages are approximately normal. Usually, $n_0 = 30$ is large enough (Lesnevski et al., 2008). Next we fixed $n_0 = 300$ and changed the growth factor R from 1.1 to 2.0. As in Lesnevski et al. (2007), this had little effect on the procedure's RMSE, which stayed between 0.92 and 1.11. In conclusion, we recommend $R = 1.2$ and $n_0 = 30$, unless the

payoff distributions are heavy-tailed (such as the current example), in which case n_0 should be increased until the first-stage sample averages are approximately normal.

2.5.3. Two-Level Simulation of an Options Portfolio

This example is the same as the example in §2.5.2, but scenarios are generated differently. Instead of using a fixed set of scenarios drawn from historical data, we generate them in an outer-level simulation. The outer-level simulation samples scenarios from a joint distribution of the two stocks' prices whose parameters are estimated from the historical data. Given the scenarios sampled, the rest of our simulation, i.e., the inner-level simulation, is the same as in §2.5.2. The purpose of considering this two-level simulation variant of the previous example is to compare our procedure with the two-level simulation procedure of Lan et al. (2008), which we refer to as the CI procedure because it generates a confidence interval for ES. We make comparisons by sampling scenarios with the CI procedure, then giving these scenarios to the standard procedure or our procedure, which perform inner-level simulation.

This example is also used in Lan et al. (2008), from which we draw the following description of how to generate scenarios from, and estimate the true value of, $ES_{0.99}$. The scenario Z is a bivariate normal random variable that determines the stock prices at time T:

$$S_T^{(j)} = S_0^{(j)} \exp\left(\left(\mu^{(j)} - \frac{1}{2}(\sigma^{(j)})^2\right)T + \sigma^{(j)}\sqrt{T}Z^{(j)}\right).$$

Based on sample moments of 1000 daily stock prices, the volatilities of CSCO and JAVA are respectively $\sigma^{(1)} = 32.85\%$ and $\sigma^{(2)} = 47.75\%$, while the correlation between the components of Z is 0.382. Because one day is such a short period of time that the effect of the drift μ is negligible, while mean returns are hard to estimate because of the high ratio of volatility to

mean, we take each $\mu^{(j)} = 0$. The value of option i at time T is the conditional expectation of the discounted payoff $Y_i := D_i(S_i - K_i)^+$ given $S_T^{(j_i)}$. The profit from holding the portfolio from 0 to T is $V(Z) = E[X|Z]$ where $X = \theta^\top(Y - P_0/D_0)$ and the discount factor $D_0 \approx 1$ because the time value of money over one day is negligible. According to a high-precision simulation, $\text{ES}_{0.99}$ is \$32.40. Notice that \$32.40 is much less than the $\text{ES}_{0.99}$ of \$52.24 produced by the historical simulation: it seems that, as usual, assuming that risk factors have a joint normal distribution leads to substantial underestimation of risk.

In Table 2.3, we report the average half-width of the 90% confidence interval for $\text{ES}_{0.99}$ generated by the CI procedure and compare it to the RMSEs of the standard procedure and our procedure, using the results of 100 macro-replications. Each procedure uses $k = 4000$ scenarios sampled from the bivariate normal distribution described above. The parameters $k = 4000$ and n_0 listed in the table were chosen by a pilot experiment described in Lan (2009) to make the CI procedure perform well. When using our procedure, we set sample size growth factor $R = 1.2$ and used the same n_0 as for the CI procedure, even though it is larger than the best n_0 for our procedure. A large n_0 is good for the CI procedure because it is a two-stage procedure, whereas our multi-stage procedure does well with small n_0. The choice of n_0 is intended to be favorable to the CI procedure and to show that the advantage of our procedure does not depend on picking the best values of the procedure's parameters. The RMSE of our procedure is not exactly comparable to the half-width of a confidence interval, but Table 2.3 shows our procedure's RMSE is so much smaller than the half-width of the CI procedure that we can conclude that our procedure is greatly preferable given a small computation budget. For these computational budgets, the CI procedure yields a confidence interval whose width is much greater than the ES we are trying to estimate, which is not useful; likewise, the RMSE of the

standard procedure is large compared to ES. Our procedure attains a relative RMSE of only a few percent when the budget is 8 or 16 million.

Table 2.3. Comparison of procedures for estimating expected shortfall at the 99% level in a two-level simulation of a portfolio of stock options, with $k = 4000$ scenarios.

Budget	n_0	CI Procedure		Standard Procedure		Our Procedure	
		Average CI half-width	Standard Error	RMSE	Standard Error	RMSE	Standard Error
4 million	612	164	1.0	109	0.40	6.7	1.6
8 million	1217	104	1.6	69	0.30	1.4	0.11
16 million	2557	49	2.3	41	0.23	0.9	0.07

CHAPTER 3

Stochastic Kriging Procedure for Point Estimation of Expected Shortfall

3.1. Introduction

In this chapter, we still focus on expected shortfall (ES) as the risk measure, but change the notation a little bit for convenience. Suppose there are K equally probable scenarios in which P&L is $\mathsf{Y}_1, \ldots, \mathsf{Y}_K$, and we are interested in a tail of probability p, where Kp is an integer. Then ES at the $1-p$ level is

$$\mathrm{ES}_{1-p} = -\frac{1}{Kp}\sum_{i=1}^{Kp}\mathsf{Y}_{(i)}, \tag{3.1}$$

where $\mathsf{Y}_{(i)}$ is the ith smallest P&L. As in Chapter 2, we refer to the scenarios whose P&L are among the Kp smallest as *tail scenarios*: they belong to the tail of the loss distribution and appear in Equation (3.1). We refer to the other scenarios as *non-tail scenarios*.

In this chapter, we improve upon the pioneering work on interpolation-based methods for risk management simulation in three ways.

(1) Instead of ordinary interpolation, we use *stochastic kriging* (Ankenman et al., 2010). This method is more powerful because it interpolates using simulation outputs from all the design points, not just those nearest to the scenario under consideration. Stochastic kriging can also be more accurate because it takes into account the inner-level sampling error.

(2) We create a two-stage experiment design suited for estimating ES. An *experiment design* is a way of choosing the design points. After the first stage of the simulation, our procedure learns which scenarios are most likely to entail the large losses that contribute to ES. It adds these scenarios to the set of design points used at the second stage. The related but different methods of Oakley (2004), who created a two-stage experiment design for a kriging procedure that estimates a quantile (VaR), inspired this aspect of our procedure.

(3) We allocate a fixed budget of inner-level replications to the design points unequally, in a way that is optimal according to the framework of stochastic kriging.

The result is a procedure that attained a root mean squared error (RMSE) dozens of times smaller than a standard simulation procedure in experiments that we ran. In these experiments, our procedure was also significantly more accurate in estimating ES than the advanced simulation procedure of Chapter 2. Our procedure's advantage over that of Liu et al. (2008) is particularly great when the number of scenarios is large or when the computational budget is small—in such examples our procedure's RMSE was three or four times smaller than that of Chapter 2.

The rest of this chapter is structured as follows. First we give a motivating example of a risk management simulation problem in Section 3.2. In Section 3.3, we review stochastic kriging and show how to use it to estimate ES. We present our new simulation procedure in Section 3.4. In Section 3.5, we provide the results of simulation experiments in which we applied our procedure to this example, and we demonstrate its advantages over other simulation procedures that estimate ES. We offer some conclusions Section 3.6.

3.2. Motivating Example

The example is almost identical to the one we considered in Chapter 2, to which we refer for details about the model and the data sources. We consider a portfolio of call options on the stocks of Cisco (CSCO) or of Sun Microsystems (JAVA), shown in Table 3.1. The example differs from that of Chapter 2 only in the portfolio's positions in the options; we explain the reason for considering a different portfolio in Section 3.4.3. In the table, the position is expressed as the number of shares of stock the option owner is entitled to buy, where a negative position means a short position in the call option.

Table 3.1. Portfolio of Call Options.

Underlying Stock	Position	Strike	Maturity (years)	Price	Risk Free Rate	Implied Volatility
CSCO	200	$27.5	0.315	$1.65	4.82%	26.66%
CSCO	-400	$30	0.315	$0.7	4.82%	25.64%
CSCO	200	$27.5	0.564	$2.5	5.01%	28.36%
CSCO	-200	$30	0.564	$1.4	5.01%	26.91%
JAVA	900	$5	0.315	$0.435	4.82%	35.19%
JAVA	1200	$6	0.315	$0.125	4.82%	35.67%
JAVA	-900	$5	0.564	$0.615	5.01%	36.42%
JAVA	-500	$6	0.564	$0.26	5.01%	35.94%

The simulation problem is to estimate the ES of this portfolio for a one-day time horizon. The scenario is the pair of tomorrow's stock prices. The model for P&L is that tomorrow, each option's value is given by the Black-Scholes pricing formula evaluated at the implied volatility given in Table 3.1. Figure 3.1 plots portfolio loss versus scenario; the vertical axis measures loss, the negative of P&L, so that the regions with the largest losses, which contribute to ES, are highest and most visually prominent.

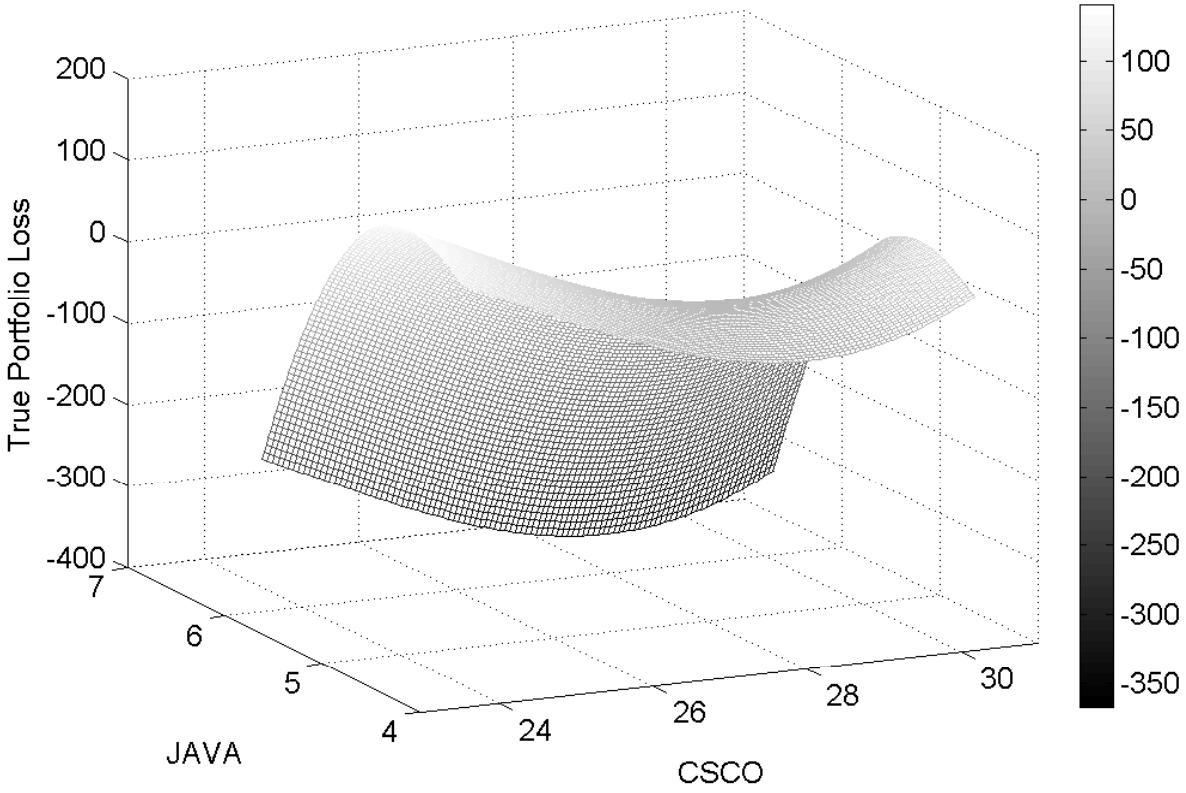

Figure 3.1. Portfolio Loss as a Function of Scenarios for Tomorrow's Stock Prices of Cisco (CSCO) and Sun Microsystems (JAVA).

When P&L is a known function of scenario, as in this example, there is no need for inner-level simulation. However, the purpose of our procedure is to handle problems in which inner-level simulation is necessary, so in applying our procedure to this example, we use inner-level simulation and not the Black-Scholes formula. An advantage of considering a simple example in which P&L is a known function of scenario is that it is easy to compute ES and thus to evaluate the accuracy of ES estimates.

We consider two versions of this example, with different kinds of outer-level simulation. In one version, the outer-level simulation is historical simulation, with a fixed set of one thousand

scenarios, portrayed in Figure 3.2. The other version uses Monte Carlo simulation, specifying a bivariate lognormal distribution for the pair of stock prices. For details, see Chapter 2.

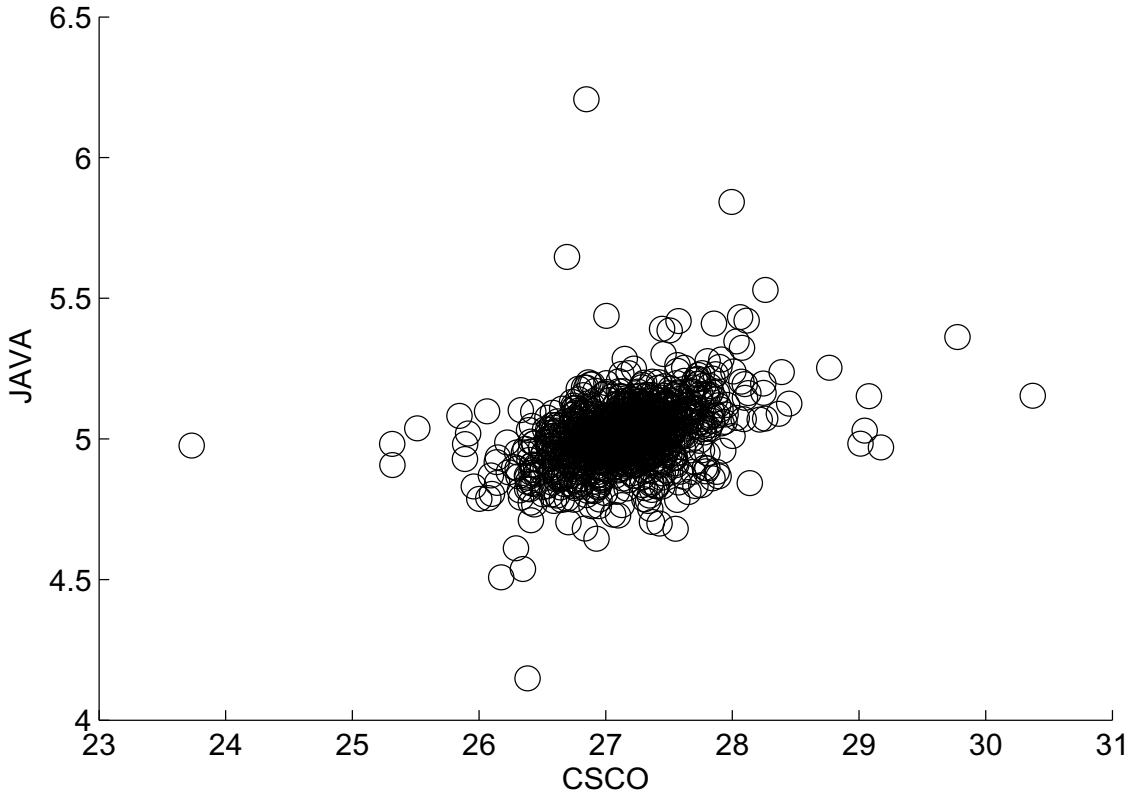

Figure 3.2. Scatter plot of 1000 scenarios from historical simulation.

3.3. Stochastic Kriging

Interpolation is one kind of *simulation metamodeling* (Barton and Meckesheimer, 2006; Kleijnen, 2008). The strategy of metamodeling is to run computationally expensive simulations only of certain scenarios, the design points, then use the simulation outputs to build a *metamodel* of the simulation model. In risk management simulation, the metamodel can be thought of as an

approximation to the unknown loss surface depicted in Figure 3.1. The metamodel can quickly provide an estimate of P&L in a scenario even if there has been no inner-level simulation of that scenario.

Stochastic kriging (Ankenman et al., 2010) is an interpolation-based metamodeling technique. It takes account of the variance that arises from inner-level simulation. Therefore, the metamodel, when evaluated at a scenario, may not equal the inner-level simulation estimate of that scenario's P&L: stochastic kriging knows that the inner-level simulation estimate may not be exactly correct. The significance of this property is that we can afford to use small sample sizes for inner-level simulation of some scenarios, because stochastic kriging smooths out the resulting noise. The following summary of stochastic kriging is based on Ankenman et al. (2010).

We model the P&L $\mathsf{Y}(\mathbf{x})$ in a scenario $\mathbf{x}$ as

$$\mathsf{Y}(\mathbf{x}) = \beta_0 + \mathsf{M}(\mathbf{x})$$

where the scenario $\mathbf{x} = [\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_d]^\top$ is a vector of risk factors, M is a stationary Gaussian random field with mean zero, and β_0 represents the overall mean. Treating M as a random field captures our uncertainty about P&L before running simulations. Ankenman et al. (2010) call this *extrinsic uncertainty*. We adopt a model frequently used in kriging, under which M is second-order stationary with a Gaussian correlation function. This means

$$\mathrm{Cov}[\mathsf{M}(\mathbf{x}), \mathsf{M}(\mathbf{x}')] = \tau^2 \exp\left(-\sum_{j=1}^{d} \theta_j (\mathrm{x}_j - \mathrm{x}'_j)^2 \right).$$

That is, τ^2 is the variance of $\mathsf{M}(\mathbf{x})$ for all $\mathbf{x}$, and the correlation between $\mathsf{M}(\mathbf{x})$ and $\mathsf{M}(\mathbf{x}')$ depends only on $\mathbf{x} - \mathbf{x}'$, with the parameter vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_d]^\top$ governing the importance of each dimension.

In addition to extrinsic uncertainty, there is also the *intrinsic uncertainty* that is inherent in Monte Carlo simulation: even after running an inner-level simulation for a scenario $\mathbf{x}$, we remain uncertain about the P&L $\mathsf{Y}(\mathbf{x})$ in that scenario. The model for simulation replication j at design point $\mathbf{x}$ is

$$Y_j(\mathbf{x}) = \beta_0 + \mathsf{M}(\mathbf{x}) + \varepsilon_j(\mathbf{x}),$$

where $\varepsilon_1(\mathbf{x}), \varepsilon_2(\mathbf{x}), \ldots$ are normal with mean zero and variance $\mathsf{V}(\mathbf{x})$, and independent of each other and of M. The simulation output at $\mathbf{x}_i$ after n_i replications is

$$\bar{Y}(\mathbf{x}_i) := \frac{1}{n_i} \sum_{j=1}^{n_i} Y_j(\mathbf{x}_i),$$

which is an estimator of the P&L $\mathsf{Y}(\mathbf{x}_i)$. Let $\boldsymbol{Y} := [\bar{Y}(\mathbf{x}_1), \ldots, \bar{Y}(\mathbf{x}_k)]^\top$ represent the vector of simulation outputs at all k design points, where n_i inner-level simulation replications are run for scenario $\mathbf{x}_i$.

We use the metamodel to estimate P&L at K scenarios $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$, referred to as *prediction points*. Before presenting the stochastic kriging predictor that provides these estimates, we define some notation. The vector of P&L at the design points is $\mathbf{Y}^k := [\mathsf{Y}(\mathbf{x}_1), \ldots, \mathsf{Y}(\mathbf{x}_k)]^\top$ and the vector of P&L at the prediction points is $\mathbf{Y}^K := [\mathsf{Y}(\boldsymbol{X}_1), \ldots, \mathsf{Y}(\boldsymbol{X}_K)]^\top$. Let $\boldsymbol{\Sigma}^{kk}$ denote the covariance matrix of $\mathbf{Y}^k$, $\boldsymbol{\Sigma}^{kK}$ denote the $k \times K$ covariance matrix of $\mathbf{Y}^k$ with $\mathbf{Y}^K$, and $\boldsymbol{\Sigma}^{Kk}$ be its transpose. Because simulations at different design points are independent, the covariance

matrix of the intrinsic noise $\boldsymbol{Y} - \mathbf{Y}^k$ is diagonal. It equals $\boldsymbol{C}^{kk}\boldsymbol{N}^{-1}$ where $\boldsymbol{C}^{kk}$ and $\boldsymbol{N}$ are diagonal matrices whose ith elements are respectively $\mathsf{V}(\mathbf{x}_i)$ and n_i. Define $\boldsymbol{\Sigma} := \boldsymbol{C}^{kk}\boldsymbol{N}^{-1} + \boldsymbol{\Sigma}^{kk}$, the sum of intrinsic and extrinsic covariance matrices for the design points. Let $\mathbf{1}^K$ and $\mathbf{1}^k$ be $K \times 1$ and $k \times 1$ vectors whose elements are all one. The stochastic kriging prediction is the Bayesian posterior mean of $\mathbf{Y}^K$ given observation $\boldsymbol{Y}$,

$$\hat{\mathbf{Y}}^K = \beta_0 \mathbf{1}^K + \boldsymbol{\Sigma}^{Kk}\boldsymbol{\Sigma}^{-1}(\boldsymbol{Y} - \beta_0 \mathbf{1}^k). \tag{3.2}$$

Ankenman et al. (2010) also give the covariance matrix of the Bayesian posterior distribution of $\mathbf{Y}^K$, which we use in Section 3.4.3.

Equation (3.2) involves parameters which are unknown in practice: β_0, τ^2, θ_1, …, θ_d, and $\mathsf{V}(\mathbf{x}_1)$, …, $\mathsf{V}(\mathbf{x}_k)$. As detailed by Ankenman et al. (2010), after running simulations, we compute maximum likelihood estimates of β_0, τ^2, and $\boldsymbol{\theta}$, and we estimate $\mathsf{V}(\mathbf{x}_1)$, …, $\mathsf{V}(\mathbf{x}_k)$ with sample variances. The output of the metamodel at $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$ is given by Equation (3.2) with these estimates plugged in. Let $\hat{\mathsf{Y}}_i$ represent the metamodel output at $\boldsymbol{X}_i$.

We use the metamodel as the basis for an estimator of ES. In the examples we consider here, we estimate ES at the $1-p$ level using a number K of scenarios such that Kp is an integer. Our methods are applicable when Kp is not an integer; for details on this case, see Chapter 2. Our estimator of ES based on the kriging metamodel is

$$\widehat{\mathrm{ES}}_{1-p} = -\frac{1}{Kp}\sum_{i=1}^{Kp} \hat{\mathsf{Y}}_{(i)} \tag{3.3}$$

where $\hat{\mathsf{Y}}_{(i)}$ is the ith lowest value among the stochastic kriging predictions $\hat{\mathsf{Y}}_1, \ldots, \hat{\mathsf{Y}}_K$ at the prediction points; cf. Equation (3.1).

We summarize the most important notation here for convenient reference:

- We want to learn about P&L in K scenarios $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$. We use stochastic kriging to compute $\hat{\mathbf{Y}}^K$ as a prediction of the P&L $\mathbf{Y}^K := [\mathsf{Y}(\boldsymbol{X}_1), \ldots, \mathsf{Y}(\boldsymbol{X}_K)]^\top$. Therefore we also call $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$ "prediction points."
- We run simulations at k design points $\mathbf{x}_1, \ldots, \mathbf{x}_k$.
- At first, we run n_0 simulation replications at each design point. In the end, there are n_i replications at design point x_i, and $\bar{Y}(\mathbf{x}_i)$ is the average of these n_i replications. The simulation output is $\boldsymbol{Y} := [\bar{Y}(\mathbf{x}_1), \ldots, \bar{Y}(\mathbf{x}_k)]^\top$.
- The variance of the simulation output for a single replication at design point $\mathbf{x}_i$ is $\mathsf{V}(\mathbf{x}_i)$, and $\boldsymbol{C}^{kk}$ is a diagonal matrix containing the variances $\mathsf{V}(\mathbf{x}_1), \ldots, \mathsf{V}(\mathbf{x}_k)$.
- The sum of the sample sizes $\sum_{i=1}^k n_i = C$, the computational budget.

3.4. Procedure

In this section, we present our simulation procedure for estimating ES using stochastic kriging. We provide an outline in Section 3.4.1 and supply the details in subsequent sections.

3.4.1. Outline of the Procedure

Our procedure uses stochastic kriging metamodels three times, so we split the description of the procedure into three stages. The estimator in Equation (3.3) uses only the third metamodel. The purpose of the first two metamodels is to guide the allocation of computational resources during the simulation procedure: deciding where to add design points and how many simulation replications to run at each design point.

The user must specify some parameters that govern the behavior of the procedure. The most important parameter is the computational budget C, which is the total number of inner-level simulation replications that the procedure can use. In the applications that we envision, inner-level simulation dominates the computational cost. Then, given the computing platform available, the computational budget roughly determines the time that the simulation procedure takes, so the user can set the computational budget to fill the time available before an answer is required. The other parameters are the target numbers k_1 of Stage I design points and k_2 of Stage II design points, the number n_0 of replications to use at each design point during Stages I and II, and the number M of times to sample from the posterior distribution of $\mathbf{Y}^K$ during Stage II. We provide some guidance about choosing these parameters after outlining the procedure.

In the outline, we refer to figures that illustrate the performance of our procedure. These figures are based on one run of the procedure on the historical simulation example of Section 3.2, using a computational budget C of 2 million replications, $K = 1000$ prediction points, a target of $k_1 = 50$ Stage I design points and $k_2 = 30$ Stage II design points, $n_0 = 5000$ replications per design point in Stages I and II, and sampling $M = 300$ times from the posterior distribution of P&L at the design points. Figure 3.3 lists the procedure's steps.

The performance of the procedure, that is, the accuracy of the ES estimator it produces, depends on the target numbers k_1 and k_2 of design points and the number n_0 of replications at each design point in Stages I and II. It is not easy to optimize the procedure's performance by choosing these parameters. Lan (2009) studies the problem of choosing such parameters for a related procedure, not based on stochastic kriging, for simulating ES. Ankenman et al. (2010, §3.3) discuss how to structure an experiment design for stochastic kriging, but not in the context

Stage I.:

(1) Generate K prediction points through outer-level simulation (historical or Monte Carlo). See Figure 3.2.

(2) Given these prediction points, generate Stage I design points. See Section 4.3.1 and Figure 3.4.

(3) Simulate n_0 replications for each of the Stage I design points. Based on the simulation outputs, create a stochastic kriging metamodel (Figure 3.5).

Stage II.:

(1) Sample a vector of P&L at each prediction point from its posterior distribution given the data generated in Stage I simulation. Based on M such samples, select the prediction points that seem likeliest to be tail scenarios, and add them to the set of design points. See Section 3.4.3 and Figure 3.6.

(2) Simulate n_0 replications for the new Stage II design points. Based on the simulation outputs, create a stochastic kriging metamodel (Figure 3.7).

Stage III.:

(1) Allocate the remaining computational budget to all design points. See Section 3.4.4 and Figure 3.8.

(2) Perform further simulation at the design points. Based on the simulation outputs, create a stochastic kriging metamodel (Figure 3.9).

(3) Compute the ES estimator in Equation (3.3) using the final metamodel.

Figure 3.3. Outline of the procedure.

of ES. We find that, with a little experience in applying the procedure to a class of problems, it is not too hard to choose parameters that result in good performance. Here we merely provide some guidelines based on our experience:

- There should be enough Stage I design points that, if P&L were known for all these scenarios, interpolation could provide a fairly accurate metamodel—sufficiently accurate to identify the region in which the tail scenarios lie. If there are too few Stage I design points to do this, the procedure's performance may be poor. The requisite number of design points is smaller in lower dimension d and when P&L is a smoother function of the scenario.

- It can be beneficial to add at least Kp design points in Stage II, which makes it possible for all Kp tail scenarios to become design points.
- To estimate the inner-level variance V well enough, the number n_0 of replications must be at least 10, or more if there is high kurtosis in inner-level sampling.
- We found that it worked well when $(k_1 + k_2)n_0$, the number of replications planned for simulation during Stages I and II, is a substantial fraction of the computational budget C, but less than half.
- In general, it is desirable to use a large number of design points, subject to two limitations. It may be counterproductive to use so many design points that n_0 needs to be too small. Also, if there are too many design points, the computer time required to perform stochastic kriging may become significant, or one may encounter difficulties with memory management because some matrices involved in stochastic kriging have size proportional to the square of the number of design points. This effect depends on the computing environment.
- As the number M of samples from the posterior distribution increases, the choice of Stage II design points converges to the set of scenarios that are likeliest to be tail scenarios, according to stochastic kriging. It is desirable to let M be large as long as this does not use up too much computer time, but M can also be much smaller than the values we use without causing major problems.

3.4.2. Choosing Stage I Design Points

As is standard in simulation metamodeling, we begin with a space-filling experiment design; the goal is to make sure that the prediction points are all near design points. In particular, we

use a maximin Latin hypercube design (Santner et al., 2003). The space that we want to fill with design points is the convex hull $\mathcal{X}$ of the prediction points $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K$. Kriging should not be used for extrapolation (Kleijnen and Beers, 2004), so we include among the design points all prediction points that fall on the boundary of the convex hull. Let k_c be the number of such points, and let $\mathcal{G}$ be the smallest d-dimensional box containing all the prediction points. In the absence of an algorithm for generating a space-filling design inside the convex set $\mathcal{X}$, we use a standard algorithm for generating a maximin Latin hypercube design in the box $\mathcal{G}$ (Santner et al., 2003). We only use the points in this design that fall inside $\mathcal{X}$, because the other points are too far away from the design points.

We want to have $k_1 - k_c$ such points. The fraction of the points in the maximin Latin hypercube design falling in $\mathcal{X}$ will be approximately the ratio of the volume of $\mathcal{X}$ to the volume of $\mathcal{G}$. The volume of a convex hull can be calculated efficiently (Barber et al., 1996), so we can calculate this ratio f. Therefore we choose the number of points in the maximin Latin hypercube design to be $\lceil (k_1 - k_c)/f \rceil$. However, the fraction of these points that actually falls in $\mathcal{X}$ may not be exactly f. Consequently, the number of Stage I design points may not be exactly k_1.

Figure 3.4 shows the Stage I design points chosen on one run of the procedure. The number of design points is 48, which is close to the planned number $k_1 = 50$. Compare Figure 3.4 to Figure 3.2, which shows the prediction points.

Figure 3.5 shows the absolute value of the error $\hat{\mathsf{Y}} - \mathsf{Y}$ of the stochastic kriging metamodel built in Stage I on this run of the procedure. At this stage, the error is substantial in many regions; compare the magnitude of the error in Figure 3.5 with the magnitude of P&L in Figure 3.1. We will see how the error shrinks after subsequent stages of the procedure.

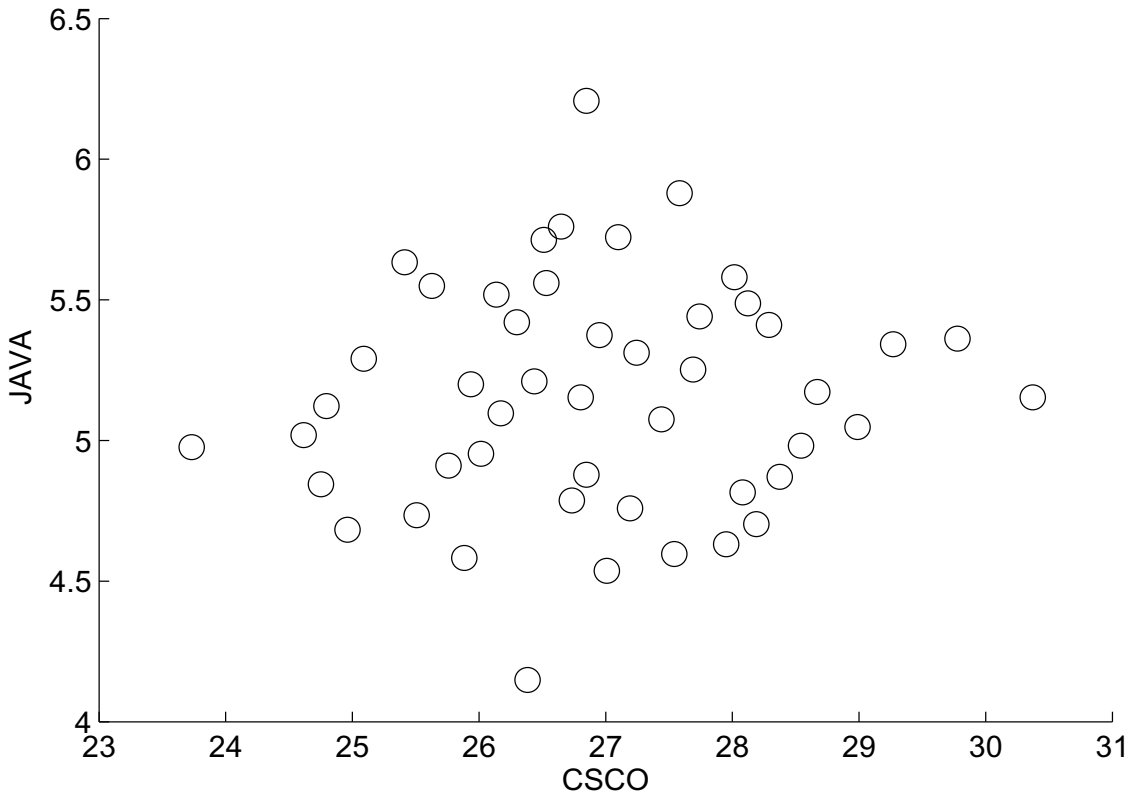

Figure 3.4. Design points chosen in Stage I on one run of the procedure.

3.4.3. Choosing Stage II Design Points

By comparing Equations (3.1) and (3.3), we see that our goal in experiment design for metamodeling should be to identify the tail scenarios and make the metamodel accurate in estimating their P&L. In Stage II, we attempt to identify the prediction points that are tail scenarios. We then add these points to the set of design points, and perform inner-level simulation of these scenarios, to learn more about their P&L.

After performing stochastic kriging in Stage I, we have the posterior distribution of $\mathbf{Y}^K$, the vector of P&L for all prediction points, which is multivariate normal (Ankenman et al., 2010).

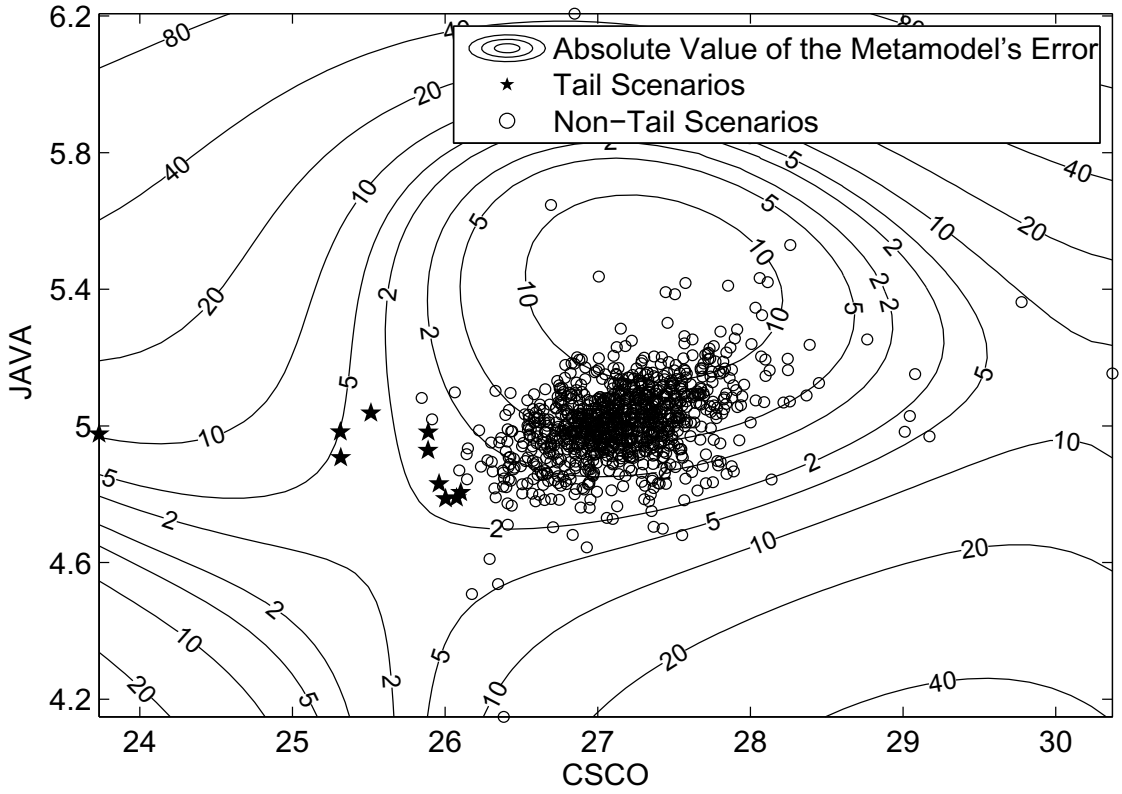

Figure 3.5. Absolute value of the error of the Stage I metamodel on one run of the procedure.

Because we are uncertain about $\mathbf{Y}^K$, we are uncertain about which prediction points are tail scenarios. Using a vector $\widetilde{\mathbf{Y}}$ sampled from the posterior distribution of $\mathbf{Y}^K$, we could try to guess which scenarios belong to the tail. We would guess that scenario i belongs to the tail if $\widetilde{\mathrm{Y}}_i$ is among the Kp lowest components of $\widetilde{\mathbf{Y}}$. However, for two reasons, this strategy of guessing would be likely to miss tail scenarios. One reason is that, if we select only Kp scenarios, we are unlikely to guess all the tail scenarios correctly. The other reason is that a single sample from the posterior distribution of $\mathbf{Y}^K$ may be unrepresentative of that distribution. Therefore, we proceed as follows in selecting up to k_2 additional design points; we envision that $k_2 > Kp$,

which improves the chances of selecting tail scenarios. We sample M vectors $\widetilde{\mathsf{Y}}^{(1)}, \ldots, \widetilde{\mathsf{Y}}^{(M)}$ independently from the posterior distribution of $\mathbf{Y}^K$. Let $T_i^{(j)}$ be an indicator function that equals one if $\widetilde{\mathsf{Y}}_i^{(j)}$ is among the Kp lowest components of $\widetilde{\mathsf{Y}}^{(j)}$, that is, scenario i is in the tail for the jth sample from the posterior distribution; otherwise, $T_i^{(j)} = 0$. Our estimated probability that scenario i is a tail scenario is $\hat{q}_i := \sum_{j=1}^M T_i^{(j)}/M$. We will use these estimated probabilities again in Stage III. In Stage II, we select the scenarios with the k_2 highest estimated probabilities, judging them likeliest to be among the tail scenarios, and make them design points. However, if fewer than k_2 scenarios have positive estimated probabilities, we only select these.

Figure 3.6 shows the design points chosen on one run of the procedure. Although $k_2 = 30$, only 17 design points were added in Stage II: the other scenarios' values were never among the $Kp = 10$ lowest in $M = 300$ samples from the posterior distribution of $\mathbf{Y}^K$. On this run of the procedure, all 10 tail scenarios were selected as design points, which is a success for the procedure.

Most of the additional design points are near each other and near the tail scenarios, but two are in a different region with a higher stock price for Cisco. Given the data available after Stage I, the procedure judges it possible that this other region might contain one of the tail scenarios, so it allocates computational resources to exploring this region. Indeed, in some risk management simulation problems, the tail scenarios may occupy multiple distant regions, and one tail scenario can be isolated from the others. The portfolio that we used as an example in Chapter 2 has this type of structure, which is more challenging for an interpolation-based procedure. Although our procedure works on that portfolio, we use a different portfolio here so as to show the procedure's performance on the type of problem for which it works best, which is a common type.

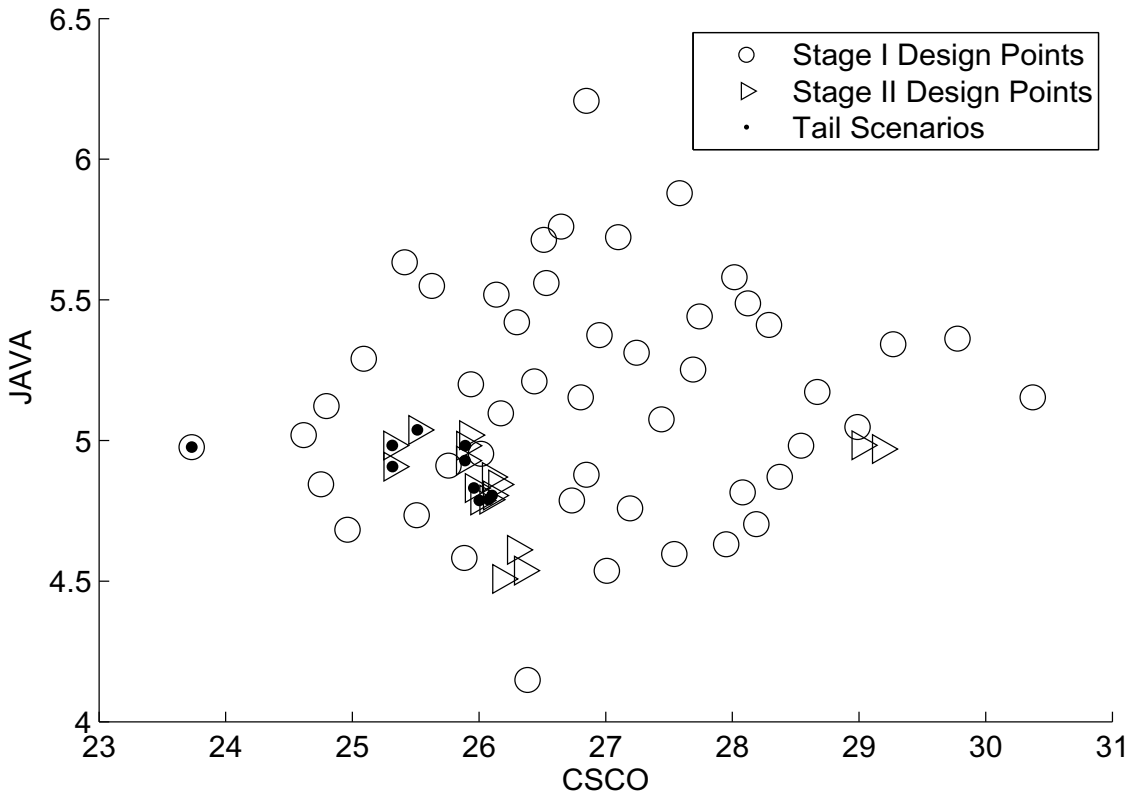

Figure 3.6. Design points chosen in Stages I and II on one run of the procedure.

Figure 3.7 shows the absolute value of the error $\hat{Y} - Y$ of the stochastic kriging metamodel built in Stage II on this run of the procedure.

3.4.4. Allocating the Remaining Computational Budget

In Stage III we allocate the remaining computational budget to inner-level simulation of the k design points chosen in Stages I and II. (The target number of design points is $k_1 + k_2$, but because of the way we choose design points, k may not exactly equal $k_1 + k_2$.) We choose an allocation with the aim of minimizing the posterior variance of the ES estimator in Equation (3.3).

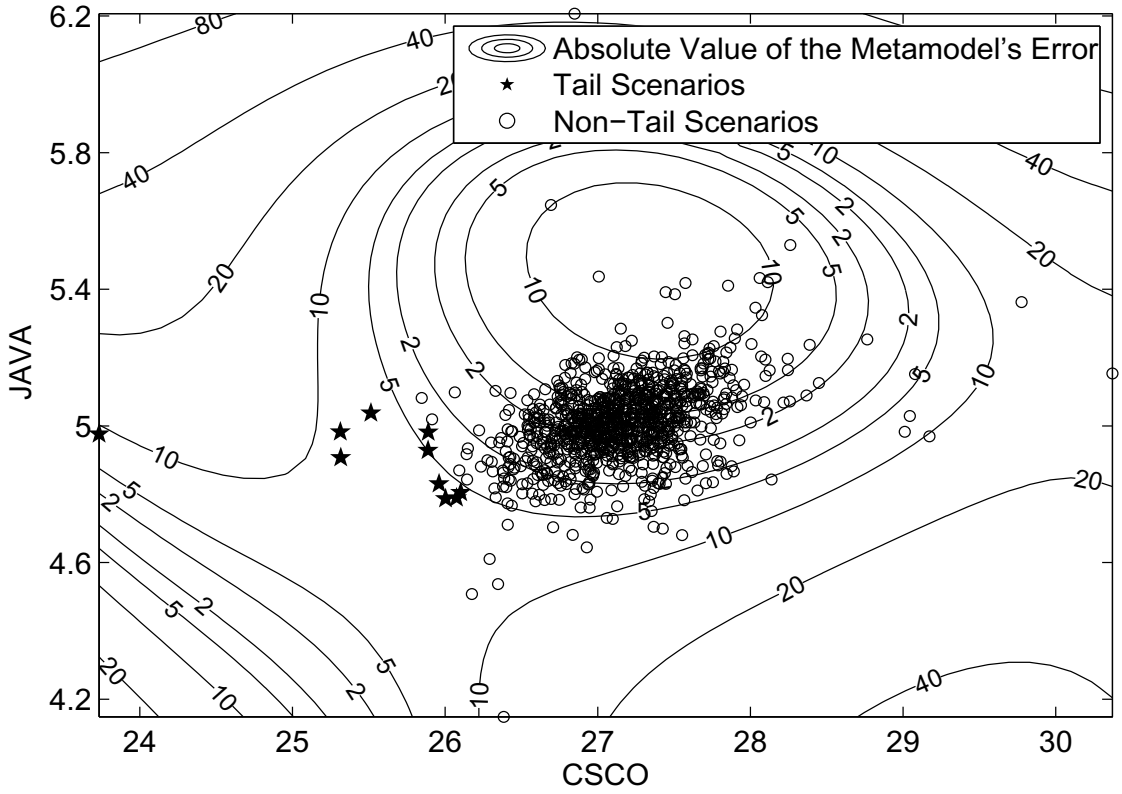

Figure 3.7. Absolute value of the error of the Stage II metamodel on one run of the procedure.

In Appendix B.1, we show how to solve a simplified version of that minimization problem by solving the optimization problem (3.4), in which the decision variable is the vector $\boldsymbol{n}$ specifying the number of replications at each design point. Because these numbers are large, we relax the integer constraint and allow them to be real numbers, without worrying about rounding. Recall from Section 3.3 that $\boldsymbol{C}^{kk}$ is a diagonal matrix with ith element $\mathsf{V}(\mathbf{x}_i)$, the intrinsic variance at the design point $\mathbf{x}_i$, $\boldsymbol{N}$ is a diagonal matrix with ith element n_i, and $\boldsymbol{\Sigma}^{kk}$ and $\boldsymbol{\Sigma}^{kK}$ are extrinsic covariance matrices. ES can be written as $\boldsymbol{w}^\top \mathbf{Y}^K$ where w_i is $-1/Kp$ if scenario i is a tail scenario, and 0 otherwise. Define $\boldsymbol{U} := (\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0)^{-1}\boldsymbol{\Sigma}^{kK}\boldsymbol{w}$. The optimization problem

is to

$$\text{(3.4)} \qquad \text{minimize} \quad \boldsymbol{U}^\top \boldsymbol{C}^{kk} \boldsymbol{N}^{-1} \boldsymbol{U} \quad \text{subject to} \quad \boldsymbol{n}^\top \mathbf{1}^k = C,\ \boldsymbol{n} \geq n_0.$$

In practice, we use maximum likelihood estimates of $\boldsymbol{\Sigma}^{kk}$ and $\boldsymbol{\Sigma}^{kK}$ and we use sample variances in estimating $\boldsymbol{C}^{kk}$, as discussed in Section 3.3. Likewise, we substitute $-\hat{q}_i/Kp$ for w_i, where $\hat{q}_i$ is the estimated probability that scenario i is a tail scenario, explained in Section 3.4.3. The optimization problem (3.4) can be solved by a variable pegging procedure (Bitran and Hax, 1981; Bretthauer et al., 1999):

Step 1.: Initialize the iteration counter $m = 1$, the index set $I(1) = \{1, \ldots, k\}$, and the unallocated budget $C(1) = C$.

Step 2.: For all $i \in I(m)$, compute $n_i(m) = C(m) U_i \sqrt{\mathsf{V}(\mathbf{x}_i)} / \sum_{j \in I(m)} U_j \sqrt{\mathsf{V}(\mathbf{x}_j)}$.

Step 3.: If $n_i(m) \geq n_0$ for all $i \in I(m)$, the solution is $\boldsymbol{n}(m)$ and we are done. Otherwise,

- the set of indices of design points that may yet receive more than n_0 replications is $I(m+1) = \{i : n_i(m) > n_0\}$,
- all other design points will receive n_0 replications: $n_i(m+1) = n_0$ for $i \notin I(m+1)$,
- and the unallocated budget is reduced to $C(m+1) = C - (k - |I(m+1)|)n_0$.

Let $m = m + 1$ and go to Step 2.

To get sample sizes from this procedure, we round the results to the nearest integers.

Figure 3.8 shows the allocation on one run of the procedure. The computational budget is spent primarily on design points that are tail scenarios or are near tail scenarios. Simulation

replications run at design points near the tail scenarios are not wasted: stochastic kriging uses them to improve the inference about the P&L in tail scenarios.

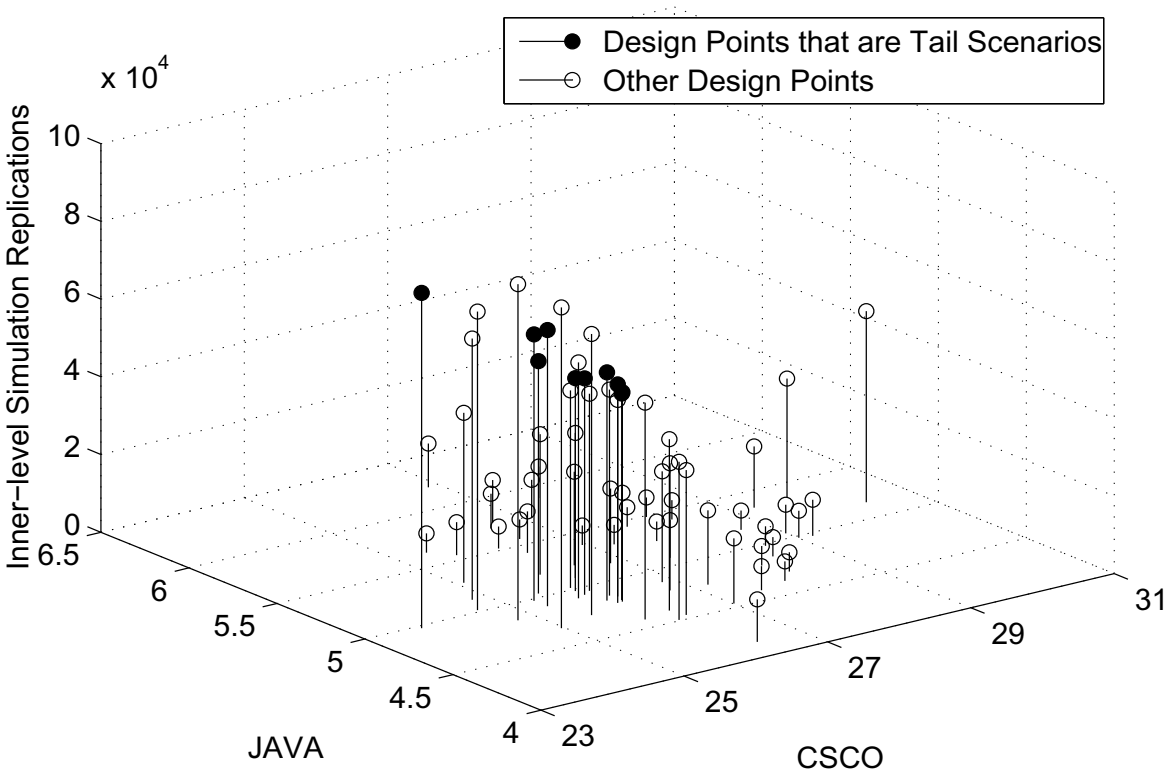

Figure 3.8. Number of simulation replications allocated to each design point on one run of the procedure.

Figure 3.9 shows the absolute value of the error $\hat{Y} - Y$ of the stochastic kriging metamodel built in Stage III on this run of the procedure. Comparing Figure 3.9 with Figure 3.7, we see that the error in estimating P&L of the tail scenarios has shrunk dramatically because of Stage III, and is now reasonably small. The error is still large in some regions, but this does not affect the quality of the ES estimation.

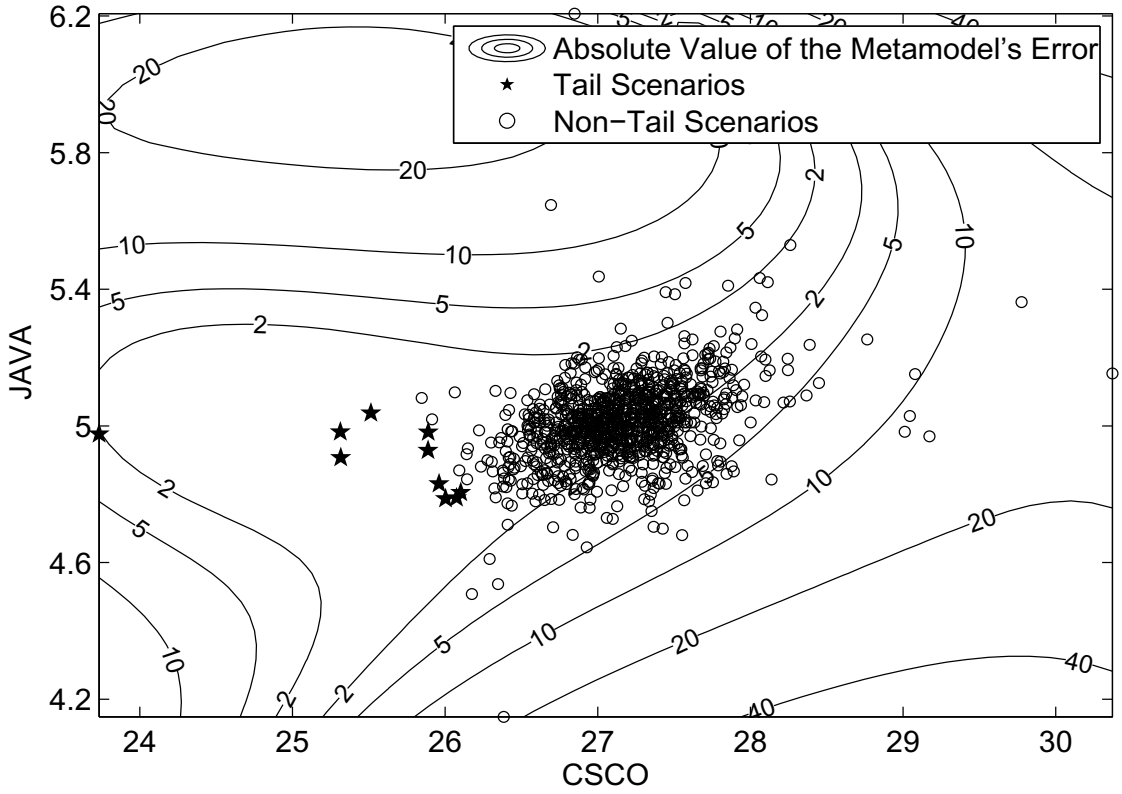

Figure 3.9. Absolute value of the error of the Stage III metamodel on one run of the procedure.

3.5. Numerical Study

To illustrate the performance of our procedure, we use the example described in Section 3.2. We present the results of simulation experiments to compare our procedure, which we call the "SK procedure," to two other procedures. One is the procedure, based on methods of statistical ranking and selection, that we proposed in Chapter 2, which we call the "RS procedure." The other is a standard procedure, involving an equal allocation of inner-level simulation replications to each scenario. It is described in detail in Chapter 2. We do not include the methods of Frye (1998), Shaw (1998), Oakley (2004), or Gordy and Juneja (2008) in the comparison.

Frye (1998) and Shaw (1998) provide strategies for simulation, not a detailed specification of a concrete procedure. Oakley (2004) and Gordy and Juneja (2008) specify simulation procedures that are tailored to estimation of VaR; although their approaches are relevant to estimating ES, construction of such procedures remains for the future.

3.5.1. Historical Simulation Example

In this section we consider the version of the example that uses historical simulation in the outer level. We first estimate ES at the $1 - p = 99\%$ level. For the SK procedure we target $k_1 = 50$ design points in Stage I and $k_2 = 30$ design points in Stage II, use $M = 300$ samples from the posterior distribution of P&L, and take sample sizes of $n_0 = 5000$ in Stages I and II. For the RS procedure, we use sample sizes that start at $n_0 = 30$ in the first stage and grow by $R = 1.1$ per stage; see Chapter 2. We run 1000 macro-replications of the simulation experiments. Figure 3.10 shows the resulting estimate of the relative root mean squared error (RRMSE) of the three procedures' ES estimators, with error bars representing 95% confidence intervals for RRMSE.

From Figure 3.10, we see that both the SK and RS procedures are far more accurate than the standard procedure for this example. For small computational budgets, the SK procedure is much more accurate than the RS procedure. It is possible to fit a straight line passing through the four error bars that describe the performance of the SK procedure, with slope roughly -0.5. The RMSE of ordinary Monte Carlo simulation procedures converges as $\mathcal{O}(C^{-0.5})$ as the computational budget grows, but the convergence rate can be less favorable for two-level simulation procedures (Lee, 1998; Lan et al., 2008). We have observed this behavior only over

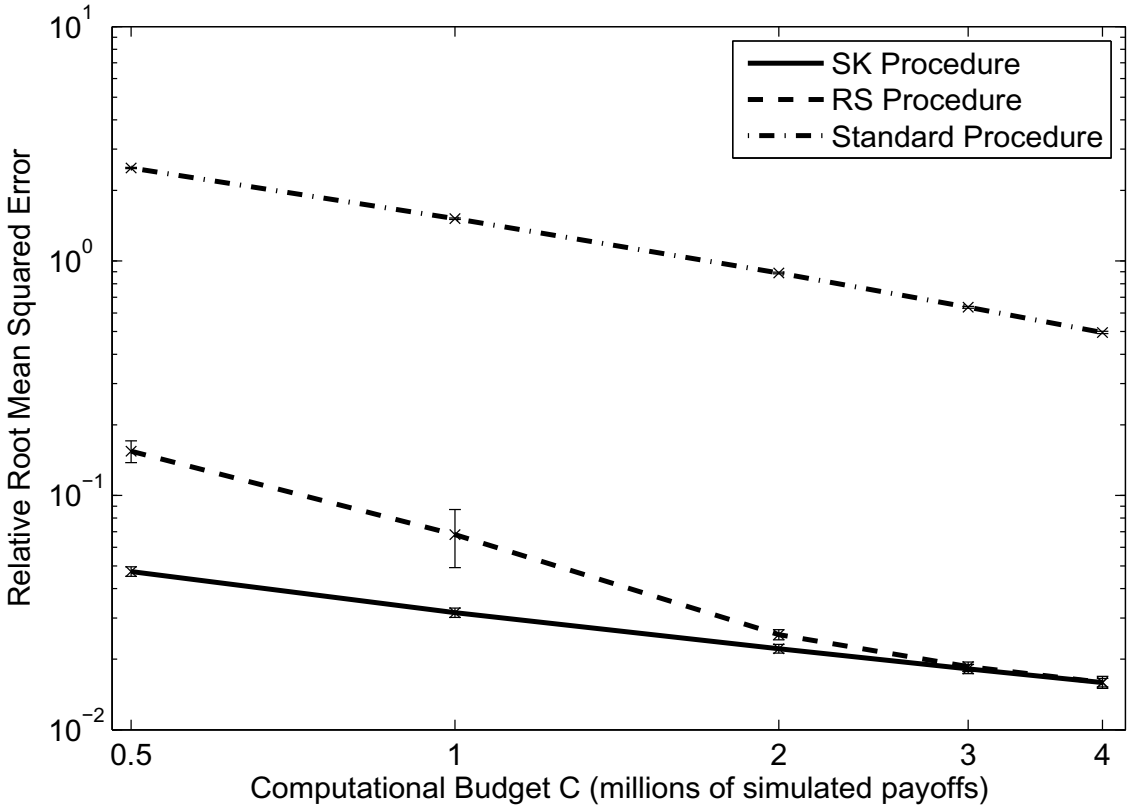

Figure 3.10. Accuracy in estimating expected shortfall at the 99% level for the historical simulation example.

a moderate range of budgets and do not know under what conditions, if any, the SK procedure has this behavior asymptotically.

Next we estimate the ES at the $1 - p = 95\%$ level. The parameters of RS procedure are the same as before. Because $Kp = 50$ is now much larger than in the previous experiment, in which it was 10, we adjust the parameters of the SK procedure. We still target $k_1 = 50$ design points in Stage I, but we allow for $k_2 = 60 > Kp$ additional design points in Stage II. We also increase the number M of samples from the posterior distribution of P&L to 600 because it is more difficult to identify the tail scenarios in this simulation problem. We still use sample

sizes of $n_0 = 5000$ in Stages I and II when the budget C is at least 1 million. However, $(k_1 + k_2)5000 > 0.5$ million, so when $C = 0.5$ million, we choose $n_0 = 2000$ instead. We run 1000 macro-replications of the simulation experiments, and show the resulting estimates of the procedures' RRMSE in Figure 3.11.

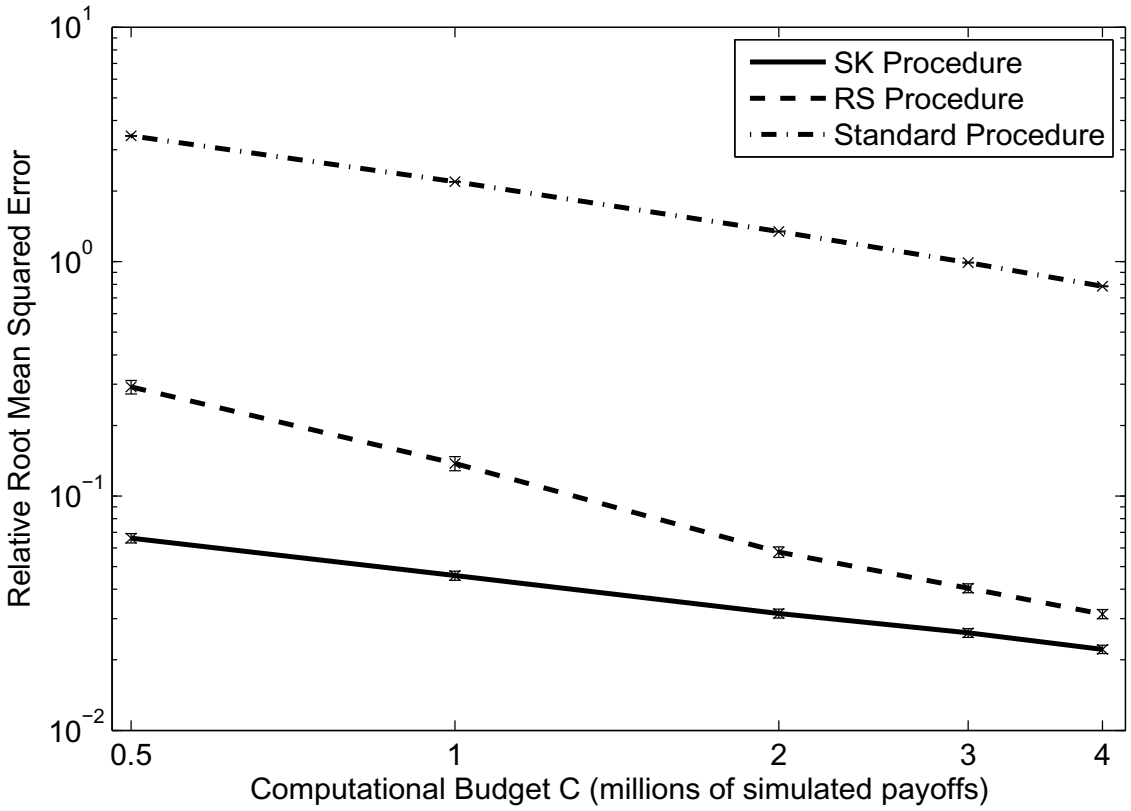

Figure 3.11. Accuracy in estimating expected shortfall at the 95% level for the historical simulation example.

Comparing Figure 3.10 and Figure 3.11, we see that the advantage of the SK procedure over the RS procedure is greater when estimating $\mathrm{ES}_{0.95}$ than $\mathrm{ES}_{0.99}$ in this example. This happens because there are more prediction points whose P&L is around the 5th percentile of P&L than around the 1st percentile. The RS procedure tries to "screen out" as many non-tail scenarios as

possible, so as to devote the remaining computational budget primarily to tail scenarios (Chapter 2). When there are many prediction points whose portfolio losses are around the pth percentile of P&L, it is hard to screen them out, so the RS procedure tends to use a lot of simulation replications in attempting to do so. Because it does not use that data in estimating ES, fewer simulation replications can be allocated to estimating ES, leading to larger error (Chapter 2). The SK procedure does not suffer from this shortcoming: all of the simulation replications contribute to the ES estimator. The curse of two-level risk management simulation is a bias that arises because, when we use simulation output to guess which scenarios entail large losses, we are likely to choose a scenario whose estimated loss is larger than its true loss (Lee, 1998; Lan et al., 2007b; Gordy and Juneja, 2008). Stochastic kriging mitigates this problem by smoothing the estimated P&L across neighboring scenarios.

3.5.2. Example with Outer-Level Monte Carlo Simulation

In this section we consider the version of the example that uses Monte Carlo simulation in the outer level. We investigate the effect of changing the number K of scenarios sampled at the outer level. In a two-level simulation with Monte Carlo at the outer level, K must grow for the simulation estimator to converge to the true value; however, if K is too large relative to the computational budget C, the estimator is poor due to excessive inner-level noise (Lee, 1998; Gordy and Juneja, 2008; Lan et al., 2008).

Figure 3.12 shows the results of 1000 macro-replications of a simulation experiment to estimate ES at the $1 - p = 99\%$ level. The computational budget C is 2 million in each of these experiments. The parameters of the RS procedure are the same as before. For the SK procedure, once again we target $k_1 = 50$ design points in Stage I and take sample sizes of $n_0 = 5000$ in

Stages I and II. We allow for $k_2 = 40$ design points in Stage II because 40 exceeds Kp even for the largest number K of scenarios we consider here, $K = 3000$. Compared to the version of this simulation with historical simulation in the outer level, it is more difficult to identify the tail scenarios, so we increase the number M of samples from the posterior distribution of P&L to 400.

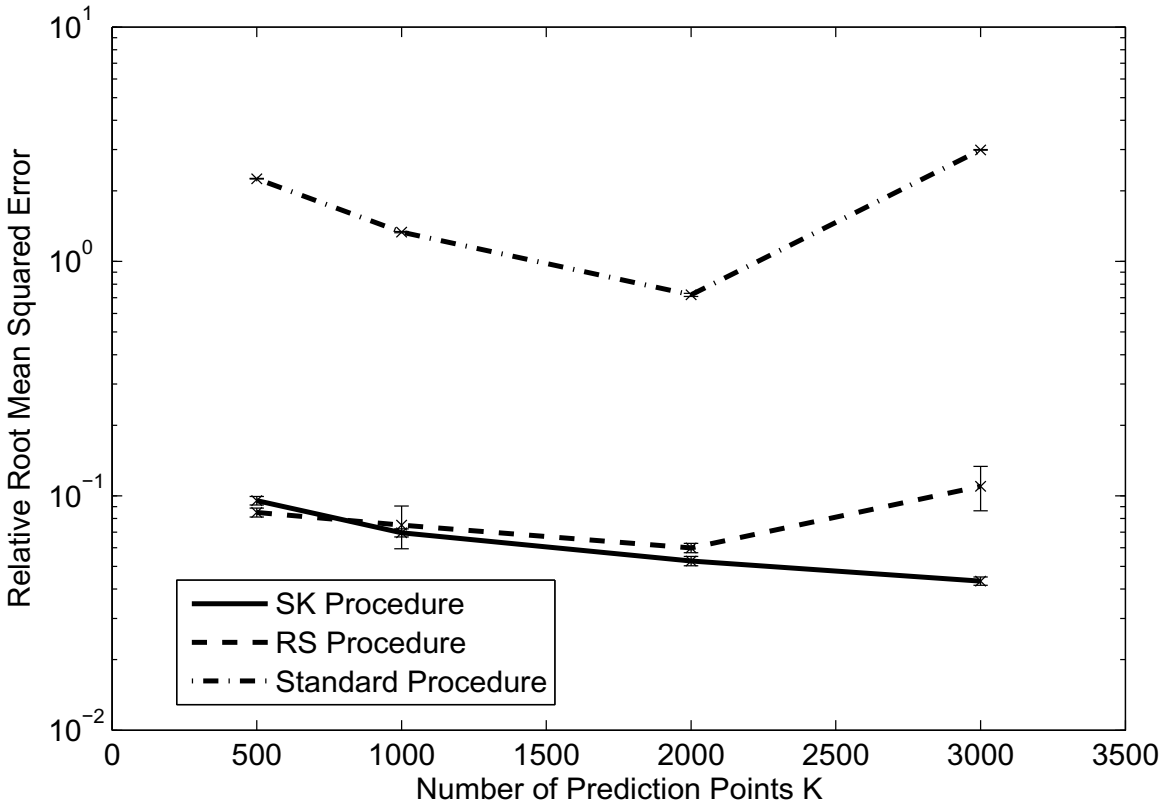

Figure 3.12. Accuracy in estimating expected shortfall at the 99% level for the two-level simulation example.

In Figure 3.12, we see that, given the budget $C = 2$ million, the best choice of K for the standard procedure and the RS procedure is around $K = 2000$, and they become much less accurate when the number of scenarios increases to $K = 3000$. When K is small, the

empirical distribution of the K scenarios is far from the true outer-level distribution; when K is large, there is a lot of inner-level noise in estimating each scenario's P&L, resulting in large bias in estimating ES (Lan et al., 2008; Lan, 2009). It is challenging to choose K well, and the procedure's performance depends greatly on this choice (Lan, 2009). By contrast, in the SK procedure, we can increase the number of K outer-level scenarios, i.e. prediction points, without increasing the number k of design points. Therefore the inner-level sample size for each design point can stay the same as we increase K. As Figure 3.12 illustrates, the RRMSE of the SK procedure's ES estimator decreases in K. Arguments in Oakley and O'Hagan (2002) suggest that the RRMSE converges to a positive value as K goes to infinity with computational budget C fixed.

We do not explore this effect in Figure 3.12 because, when K is very large, our `MATLAB` implementation of stochastic kriging encounters memory constraints on a PC with 3.4 GB of RAM. When K is very large, the RS and SK procedures have significant space and time requirements for operations other than inner-level simulation. These have to do, respectively, with comparing many scenarios to each other, and with operations involving large matrices. Because these effects depend greatly on the computing environment, we do not explore them here, instead treating inner-level simulation replications as the primary computational cost.

This experiment suggests two advantages of the SK procedure over the standard and RS procedures when using outer-level Monte Carlo simulation. The user need not worry about finding an optimal, moderate number K of outer-level scenarios, where the optimal K varies greatly from one simulation problem to another (Lan, 2009). Instead, one can always use the largest K such that stochastic kriging does not impose an excessive computational burden. Also,

we believe that, as in Figure 3.12, for many simulation problems, the SK procedure with large K performs better than the standard and RS procedures with optimal K.

3.6. Conclusions and Future Research

Stochastic kriging enables better estimation of expected shortfall. Our simulation procedure is well suited to dealing with small computational budgets. It works especially well compared to other procedures when the spatial structure of the simulation problem is such that most tail scenarios lie near other scenarios and P&L is a smooth function of the scenario, but it also works even when the problem does not have these properties. Another advantage of our procedure over its competitors is that it makes it far easier for the user to choose the number of outer-level Monte Carlo simulation replications. There are several opportunities for further investigation and improvement of risk management simulation procedures based on stochastic kriging.

We used two-dimensional examples to illustrate our method. It remains to be seen how well it performs for higher-dimensional examples. Higher-dimensional problems are more challenging for kriging methods: it is more difficult to find a good experiment design, and the error of the metamodel tends to increase. Dimension-reduction methods, such as those proposed by Frye (1998) and Shaw (1998), should help. However, kriging methods are capable of handling significantly higher-dimensional examples.

When the number of prediction points is very large, stochastic kriging may take up a great deal of memory and CPU time. This happens when stochastic kriging considers the influence of simulation at all design points on predictions at each prediction point, or the posterior covariance between P&L at every pair of prediction points. Using spatial correlation functions that imply

zero correlation between sufficiently distant points (Santner et al., 2003) reduces the number of pairs that must be considered and should help to make it feasible to use more prediction points.

In our study, we used the simplest version of stochastic kriging, which builds a metamodel purely by interpolation. However, stochastic kriging can incorporate regression methods in simulation metamodeling (Barton and Meckesheimer, 2006; Kleijnen, 2008). Many portfolios have structure that regression can capture (e.g., an increasing trend in P&L with the level of a global equity index), in which case regression will lead to lower error in metamodeling.

Our procedure uses a simple first-stage experiment design, which could be improved. In some simulation problems, there would be too many prediction points on the convex hull. A modification of the experiment design would find a larger convex polytope, with fewer vertices, still containing all the prediction points.

The second-stage experiment design worked well in the problems we studied, in which there were relatively few tail scenarios. This allowed us to aim to include all the tail scenarios among the design points and to ignore the spatial relationships among the scenarios that seemed likely to be tail scenarios. When there are many tail scenarios, it might be better to create a second-stage experiment design with a different goal: to aim to have some design point near every scenario that is likely to be a tail scenario.

CHAPTER 4

Simulation on Demand for Pricing Many Securities

4.1. Introduction

"Simulation on demand" is a computing paradigm that delivers real-time answers as accurate as those that would be generated by a time-consuming run of a simulation model. This is achieved by investing computational effort in advance, before decision-makers ask about a specific scenario. Simulation metamodeling is one method that supports simulation on demand. A metamodel is an approximation to the function y of interest, whose value $y(\boldsymbol{x})$ at scenario $\boldsymbol{x}$ is estimated by running the simulation model for scenario $\boldsymbol{x}$. Building a metamodel requires a simulation experiment in which the simulation model is run for several scenarios, but after this computational investment has been made, it can be very fast to evaluate the metamodel in any scenario. This chapter is devoted to experiment design for building multiple metamodels based on the same simulation model.

We consider a financial example: a firm deals in many securities whose prices are functions of the financial scenario. A single simulation model is used to determine the securities' prices. Simulation on demand provides an approximate picture of the way the prices change as the markets move. Our procedure is related to one due to Frye (1998), involving a grid-based interpolation technique, which requires a grid design for the simulation experiment. Because of the impracticality of a high-dimensional grid design, Frye used principal component analysis to reduce the dimension of the simulation model. One of our contributions is to extend Frye's

work by showing that the latest metamodeling techniques make it computationally feasible to construct highly accurate metamodels of each security's price, not merely one moderately accurate metamodel of the portfolio's value. In particular, by using an experiment design that is practical in higher dimension, we avoid the loss of accuracy entailed by dimension reduction. This requires a metamodeling technique other than grid-based interpolation. We use stochastic kriging (Ankenman et al., 2010), but our procedure works with many metamodeling techniques.

Our main contribution is a sequential experiment design procedure that adds design points and simulation effort at the design points to reduce all metamodels' prediction errors to an acceptable level relative to the true values. It is appropriate to focus on relative (not absolute) error in applications involving multiple metamodels which have very different magnitudes, including our financial example, in which some options have much larger prices than others. The key ingredient in our procedure is cross-validation, which is widely used for validating prediction schemes (Geisser, 1993). Kleijnen (2008) discusses cross-validation in stochastic simulation metamodeling. The novel aspect of our procedure for stochastic simulation is that it continues until the prediction errors are likely to be small, instead of continuing until the simulation output is consistent with the metamodeling technique's assumptions.

In some applications, the metamodels are functions of a scenario that can be regarded as random: in our financial example, the scenario that will occur tomorrow is random. In such applications, it is meaningful to consider the expected performance of the metamodels at a randomly selected scenario. Another contribution of this chapter is an initial experiment design that aims to make the scenario fall inside the convex hull of design points with high probability. This improves the metamodels' performance because many metamodeling techniques are much better at interpolation than at extrapolation.

4.2. Motivating Example

We consider a portfolio of 75 European-style options. The underlying vector stochastic process models six equity indices: S&P500, Nikkei225, Stoxx50, FTSE100, Hang Seng, and KOSPI Composite. To keep the example simple, we use a very basic model and approach to model calibration. The equity indices, denoted by $j = 1, 2, \ldots, 6$, follow geometric Brownian motion where the non-annualized daily volatilities σ_j and correlation matrix $\Sigma_{\boldsymbol{X}}$ are estimated from 1000 historical daily returns. A scenario $\boldsymbol{X}$ is a vector whose components are called risk factors, and it determines the values of the equity indices. The risk factors are six standard normal random variables with correlation matrix $\Sigma_{\boldsymbol{X}}$. If tomorrow's scenario is $\boldsymbol{X}$, then for each $j = 1, 2, \ldots, 6$, tomorrow's value of the jth index is $S_j(1) = S_j(0) \exp(\sigma_j X_j)$, where $S_j(0)$ is today's value of the jth index. Here we have set the drift to zero, because it is negligible over one day. The portfolio contains five classes of options: put options on the average return of all six indices, call options on the average return of the S&P500, Nikkei225, and Stoxx50, call options on the average return of the FTSE100, Hang Seng, and KOSPI Composite, call options on the minimum return of the S&P500, Nikkei225, and Stoxx50, and call options on the minimum return of the FTSE100, Hang Seng, and KOSPI Composite. Within each class, there are 15 options with one of five maturities and one of three strike prices. The maturities are 3, 4, 5, 6, and 7 years. The three strike prices are chosen to make the option in the money, (roughly) at the money, or out of the money. Given that tomorrow's scenario is $\boldsymbol{X}$, the value of the jth index after t days is

$$S_j(t) = S_j(1) \exp\left(\left(\mu_j - \frac{\sigma_j^2}{2}\right)(t-1) + \sigma_j \sqrt{t-1} B_j\right), \tag{4.1}$$

where $\boldsymbol{B}$ is a vector of six standard normal random variables with correlation matrix $\Sigma_{\boldsymbol{X}}$, and each risk-neutral drift μ_j equals the non-annualized daily yield of a government bond denominated in the relevant currency minus the dividend yield of the index. A sample path, simulated conditional on $\boldsymbol{X}$, includes the values of all six equity indices after 3, 4, 5, 6, and 7 years. The simulation model for option pricing computes discounted payoffs for all 75 options on a single sample path.

4.3. Simulation Procedure

Our procedure chooses scenarios, called design points, at which to run simulations. It also determines the number of sample paths to simulate at each design point. A sample path is a simulation of the underlying stochastic process conditional on the scenario given by the design point, as in Equation (4.1). The sample average $\bar{Y}_h(\mathbf{x})$ of the discounted payoffs of security h on every sample path simulated at design point $\mathbf{x}$ serves as an estimate of the price of security h in the scenario $\mathbf{x}$. The procedure produces one metamodel $\hat{\mathrm{Y}}_h(\boldsymbol{X})$ for the price of each security $h = 1, 2, \ldots, r$. The goal of the procedure's first phase, initial simulation, is to get accurate estimates of all security prices at all design points. This is not enough to ensure that the metamodels will give accurate estimates of all security prices at scenarios that are not design points. The goal of the second phase, metamodel validation, is to improve the accuracy of the metamodels away from the design points. This phase adds design points and sample paths until the metamodels pass a cross-validation test of their ability to estimate security prices at scenarios that are not used at design points. An outline of the procedure is:

Phase I.: Initial Simulation

(1) Generate k design points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$.

(2) For $i = 1, 2, \ldots, k$, simulate n_i sample paths at design point $\mathbf{x}_i$, where n_i is chosen to target the relative accuracy in estimating the value of each security at $\mathbf{x}_i$.

Phase II.: Metamodel Validation

(1) Create metamodels by stochastic kriging.

(2) Perform cross-validation on the metamodels. If they all pass the cross-validation test, the procedure terminates.

(3) If they do not all pass, simulate more sample paths at existing design points or generate additional design points and simulate sample paths at them. Then return to Step 1 of Phase II.

4.3.1. Initial Simulation Phase

The initial simulation phase consists of two parts: a method for choosing k design points and a two-stage simulation procedure. Information from the first stage, which simulates n_0 sample paths at each design point, is used to choose the total sample size n_i at each design point $\mathbf{x}_i, i = 1, 2, \ldots, k$. The second stage of simulation generates the sample paths required to reach those total sample sizes.

4.3.1.1. Design Points. In choosing design points, we have two goals. One goal is for tomorrow's scenario to fall inside the convex hull of the design points with high probability. The reason for this is that many metamodeling techniques, including stochastic kriging, are much better at interpolation than at extrapolation. The other goal is, as usual in simulation metamodeling, to fill the space of scenarios evenly with design points.

To address the probability that tomorrow's scenario $\boldsymbol{X}$ falls inside the convex hull of the design points, we need a joint distribution $F_{\boldsymbol{X}}$ for the risk factors $X_1, X_2, \ldots, X_d$. Our procedure requires that we be able to evaluate a function f such that, if $\boldsymbol{U}$ is uniformly distributed on $(0,1)^d$, then $f(\boldsymbol{U})$ has distribution $F_{\boldsymbol{X}}$. We choose design points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ in $\mathcal{X}$ by choosing design points $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k$ in $\mathcal{U}$ and transforming them to get $\mathbf{x}_i = f(\boldsymbol{u}_i)$ for $i = 1, 2, \ldots, k$. One way to get $f : \mathcal{U} \to \mathcal{X}$ is from a simulation algorithm: even if f is not known explicitly, a typical simulation algorithm takes independent uniform random variables as inputs and generates a random vector with distribution $F_{\boldsymbol{X}}$. However, sometimes the analyst may have only marginal distributions $F_{X_1}, F_{X_2}, \ldots, F_{X_d}$ for the risk factors and their correlation matrix $\Sigma_{\boldsymbol{X}}$. A further assumption about dependence among the risk factors is required to get a joint distribution $F_{\boldsymbol{X}}$. An assumption often made in financial engineering models (although it may give an unrealistic picture of extreme events), is that $\boldsymbol{X}$ has a Gaussian copula (McNeil et al., 2005, § 5.1). The same assumption is used in simulation input modeling in the normal-to-anything (NORTA) transformation (Cario and Nelson, 1997). Then the transformation $\boldsymbol{X} = f(\boldsymbol{U})$ is accomplished as follows, where Φ is the standard normal cumulative distribution function:

(1) For each $i = 1, 2, \ldots, k$, set $W_i = \Phi^{-1}(U_i)$ to get a standard normal vector $\boldsymbol{W}$.

(2) Where $\Sigma_{\boldsymbol{Z}}^{-1/2}$ satisfies $\Sigma_{\boldsymbol{Z}}^{-1/2} \left(\Sigma_{\boldsymbol{Z}}^{-1/2}\right)^\top = \Sigma_{\boldsymbol{Z}}$, set $\boldsymbol{Z} = \Sigma_{\boldsymbol{Z}}^{-1/2}\boldsymbol{W}$ to get a vector $\boldsymbol{Z}$ with standard normal marginal distributions and correlation matrix $\Sigma_{\boldsymbol{Z}}$.

(3) For each $i = 1, 2, \ldots, k$, set $\tilde{U}_i = \Phi(Z_i)$ to get a vector $\tilde{\boldsymbol{U}}$ whose components are dependent and have marginal distributions that are uniform on $(0,1)$.

(4) For each $i = 1, 2, \ldots, k$, set $X_i = F_{X_i}^{-1}(\tilde{U}_i)$ to get a vector $\boldsymbol{X}$.

Cario and Nelson (Cario and Nelson, 1997, 1998) show how to choose $\Sigma_{\boldsymbol{Z}}$ so that the correlation matrix of $\boldsymbol{X}$ is $\Sigma_{\boldsymbol{X}}$.

Our experiment design contains two kinds of points: corner points and points sampled via quasi-Monte Carlo. Let p be a target probability for tomorrow's scenario to fall into the convex hull of the design points. Each of the 2^d corner points is the image under f of a vertex of the hypercube $\mathcal{U}_p = \{\boldsymbol{u} : 0.5(1-p^{1/d}) \le u_j \le 0.5(1+p^{1/d}) \; \forall j = 1, 2, \ldots d\}$, which has volume p. Including 2^d corner points is feasible when the dimension d is moderate. The target probability p must be less than one if it is impossible to map the vertices of the unit hypercube $[0,1]^d$ to usable scenarios in $\mathcal{X}$: e.g., in the example of Section 4.2, if $U_1 = 1$, then the first risk factor X_1 and the value S_1 of the S&P500 index are infinite. The probability that tomorrow's scenario falls inside the convex hull of the design points is not guaranteed to be p, but in many cases, choosing a large target probability p makes tomorrow's scenario fall inside the convex hull of the design points with high probability. The remaining $k - 2^d$ design points are generated from a Sobol' sequence scaled to fit inside $\mathcal{U}_p$: if $\boldsymbol{u}'$ is a point in a Sobol' sequence in $[0,1)^d$, the corresponding design point is $f(0.5(1-p^{1/d}) + p^{1/d}\boldsymbol{u}')$. Figure 4.1, whose panels are in sequence left to right and then top to bottom, shows the process of creating $k = 50$ design points for a version of the example of Section 4.2 in which there are only two risk factors, corresponding to the S&P500 and Nikkei225 indices. The panels for $\boldsymbol{Z}$ and $\boldsymbol{X}$ are the same because the risk factors in this example are normally distributed, so the step of generating $\tilde{\boldsymbol{U}}$ is redundant.

4.3.1.2. Sample Sizes in a Two-Stage Simulation Procedure. In the first stage of Phase I, we simulate n_0 sample paths at each design point and compute the sample average $\bar{Y}_h(\mathbf{x}_i)$ and sample variance $s_h^2(\mathbf{x}_i)$ of the discounted payoffs for each design point $i = 1, 2, \ldots, k$ and each security $h = 1, 2, \ldots, r$. Then we choose a total sample size n_i to attain at each

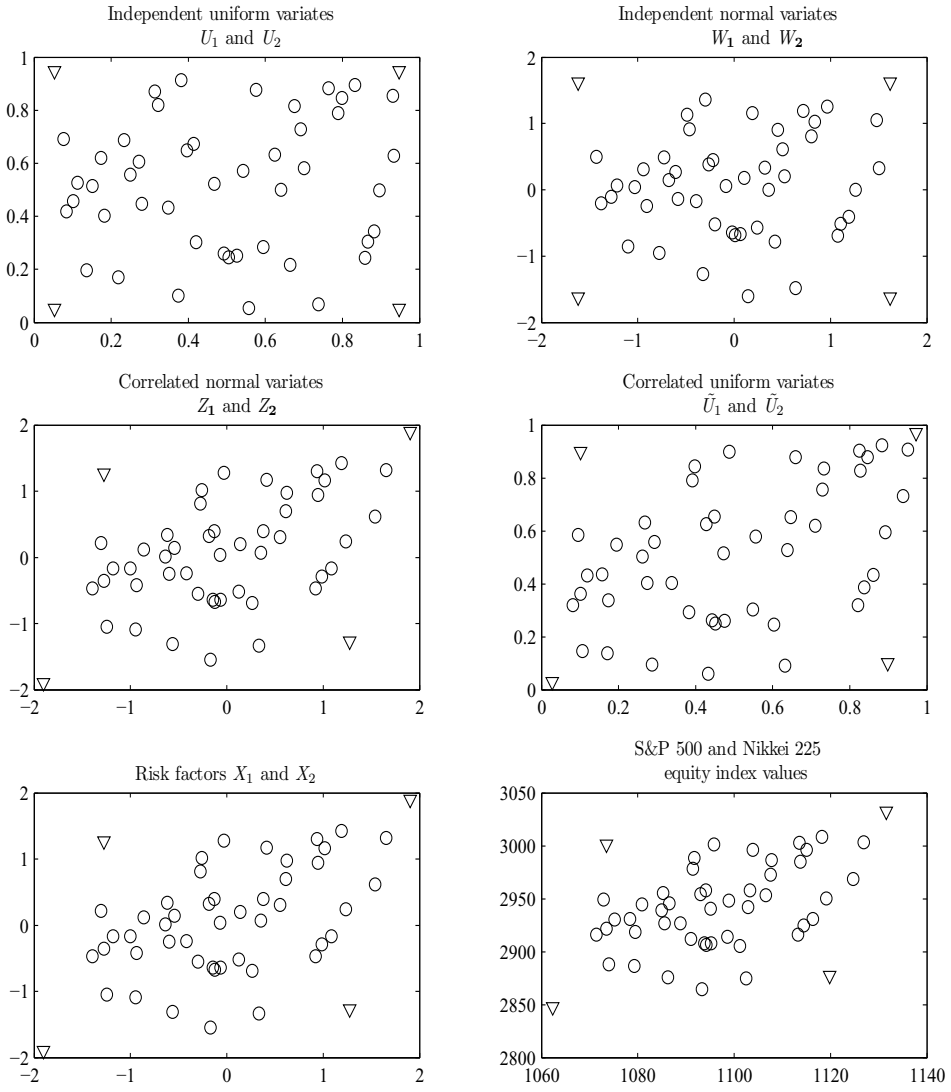

Figure 4.1. Construction of Phase I design points in a two-dimensional example: $\triangledown$ indicates corner points, and $\circ$ indicates points generated by quasi-Monte Carlo.

design point after a second stage of sampling, and simulate $n_i - n_0$ additional sample paths at design point $\mathbf{x}_i$ for $i = 1, 2, \ldots, k$. We choose the sample size n_i to target a relative precision

for the simulation output. Based on an assumption that the discounted payoffs are normally distributed with unknown variance, a fixed sample size of n yields a half-width of the $(1-\alpha)$ confidence interval for $y_h(\mathbf{x}_i)$ of $l_h(\mathbf{x}_i, n; \alpha) = t_{n-1,1-\alpha/2} s_h(\mathbf{x}_i)/\sqrt{n}$, where $t_{n-1,1-\alpha/2}$ is the $1-\alpha/2$ quantile of the t distribution with $n-1$ degrees of freedom. Let $y_h(\mathbf{x}_i)$ be the true price of security h in scenario $\mathbf{x}_i$. The relative precision of the average of n_i sample paths is $l_h(\mathbf{x}_i, n_i; \alpha)/\left|y_h(\mathbf{x}_i)\right|$. We target a relative precision of γ between 0 and 1. After the first stage, we choose the sample size

$$n_i = \max\left\{n_0, \left\lceil \max_{h=1,2,\ldots,r} \left(\frac{(1+\gamma) t_{n_0-1,1-\alpha/2} s_h(\mathbf{x}_i)}{\gamma \bar{Y}_h(\mathbf{x}_i)} \right)^2 \right\rceil \right\} \tag{4.2}$$

because it makes

$$\frac{l_h(\mathbf{x}_i, n_i; \alpha)}{\left|\bar{Y}_h(\mathbf{x}_i)\right|} \le \frac{\gamma}{1+\gamma} \tag{4.3}$$

for every security $h = 1, 2, \ldots, r$. The relative precision $l_h(\mathbf{x}_i, n_i; \alpha)/\left|y_h(\mathbf{x}_i)\right| \le \gamma$ with approximately $1-\alpha$ level confidence if Equation (4.3) holds (Law, 2007, p. 502). This is merely an approximation because the discounted payoffs are not normally distributed, and the sample size n_i is random, not fixed. In particular, it depends on the first-stage sample average $\bar{Y}_h(\mathbf{x}_i)$, which spoils the usual arguments for the validity of two-stage fixed-width confidence interval procedures such as Stein's (Stein, 1945). However, if the sample size is large and the true price is not too close to zero, we expect to attain the desired relative precision with high confidence if α is small.

4.3.2. Metamodel Validation Phase

After the initial simulation, we construct and validate metamodels. If they fail a test based on leave-one-out cross-validation, we add more design points or generate more sample paths at existing design points until the updated metamodels pass the test. The essential idea of leave-one-out cross-validation, when applied to a metamodel of a deterministic simulation model, is to look at the difference between the true value $y_h(\mathbf{x}_i)$, observed by running the simulation at the design point $\mathbf{x}_i$, and the leave-one-out prediction $\hat{\mathrm{Y}}_h^{(-i)}(\mathbf{x}_i)$ of a metamodel constructed using all the design points except $\mathbf{x}_i$. When the simulation is stochastic, the true value $y_h(\mathbf{x}_i)$ can not be observed. Our validation method considers the leave-one-out prediction $\hat{\mathrm{Y}}_h^{(-i)}(\mathbf{x}_i)$, the simulation output $\bar{Y}_h(\mathbf{x}_i)$ which serves as an estimate of the true value $y_h(\mathbf{x}_i)$, and the confidence interval half-width $l_h(\mathbf{x}_i, n_i; \alpha)$ as a measure of uncertainty in $\bar{Y}_h(\mathbf{x}_i)$.

We use cross-validation for a subset $\mathbb{I} \subseteq \{1, 2, \ldots, k\}$ of design points and a subset $\mathbb{H} \subseteq \{1, 2, \ldots, r\}$ of securities. The design points in $\mathbb{I}$ are those that are not on the convex hull of the set of design points. This ensures that for all $i \in \mathbb{I}$, $\hat{\mathrm{Y}}_h^{(-i)}(\mathbf{x}_i)$ is an interpolation, not an extrapolation. The subset $\mathbb{H}$ could contain all r securities, but if r is too large, cross-validation will take a very long time. One may choose a smaller subset $\mathbb{H}$ by including only one representative of each class of securities. For example, a class may consist of securities which differ from each other only in maturity and strike price. We suggest choosing the representative of a class to be the security which is most computationally expensive to price with good relative accuracy: for example, after Phase I, one may choose representatives with the highest value of $\max_{i=1,2,\ldots,k} s_h(\mathbf{x}_i)/\bar{Y}_h(\mathbf{x}_i)$.

Proposition 1 provides some justification for our method, which aims to control the relative leave-one-out prediction error $|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - y_h(\mathbf{x}_i)|/|y_h(\mathbf{x}_i)|$ at each design point. As in Section 4.3.1.2, non-normality of the simulation output and randomness of the sample sizes mean that the method is merely approximate: thus, in interpreting the proposition, we should remember that $\Pr\left\{\left|\bar{Y}_h(\mathbf{x}_i) - y_h(\mathbf{x}_i)\right| \le l_h(\mathbf{x}_i, n_i; \alpha)\right\}$ may not be exactly $1-\alpha$, the confidence sought in the construction of Equation (4.2). For reasons supplied by Proposition 1, our test of validity is based on

$$E_{hi} = \frac{l_h(\mathbf{x}_i, n_i; \alpha)}{\left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha)} + \frac{\left|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - \bar{Y}_h(\mathbf{x}_i)\right|}{\left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha)}, \tag{4.4}$$

where the first term measures the relative precision of simulation output and the second term measures the relative discrepancy between metamodel prediction and simulation output. These measurements are relative to $\left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha)$, a lower confidence limit for $y_h(\mathbf{x}_i)$.

Proposition 1. *For any* $h = 1, 2, \ldots, r$ *and* $i = 1, 2, \ldots, k$, $\left|\bar{Y}_h(\mathbf{x}_i) - y_h(\mathbf{x}_i)\right| \le l_h(\mathbf{x}_i, n_i; \alpha)$ *implies* $\left|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - y_h(\mathbf{x}_i)\right| / \left|y_h(\mathbf{x}_i)\right| \le E_{hi}$.

Proof If $\bar{Y}_h(\mathbf{x}_i) - l_h(\mathbf{x}_i, n_i; \alpha) \le y_h(\mathbf{x}_i) \le \bar{Y}_h(\mathbf{x}_i) + l_h(\mathbf{x}_i, n_i; \alpha)$, then

$$\left|y_h(\mathbf{x}_i)\right| \ge \min\left\{\left|\bar{Y}_h(\mathbf{x}_i) - l_h(\mathbf{x}_i, n_i; \alpha)\right|, \left|\bar{Y}_h(\mathbf{x}_i) + l_h(\mathbf{x}_i, n_i; \alpha)\right|\right\} = \left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha),$$

which is positive because Equation (4.2) implies that $l_h(\mathbf{x}_i, n_i; \alpha) < |\bar{Y}_h(\mathbf{x}_i)|$. Therefore

$$\begin{aligned}
\frac{\left|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - y_h(\mathbf{x}_i)\right|}{\left|y_h(\mathbf{x}_i)\right|} &\le \frac{\left|\bar{Y}_h(\mathbf{x}_i) - y_h(\mathbf{x}_i)\right| + \left|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - \bar{Y}_h(\mathbf{x}_i)\right|}{\left|y_h(\mathbf{x}_i)\right|} \\
&\le \frac{l_h(\mathbf{x}_i, n_i; \alpha) + \left|\hat{\mathsf{Y}}_h^{(-i)}(\mathbf{x}_i) - \bar{Y}_h(\mathbf{x}_i)\right|}{\left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha)} = E_{hi}. \ \square
\end{aligned}$$

The proposition suggests an iterative procedure that adds simulation effort until $E_{hi} \leq \beta$ for all $h \in \mathbb{H}$ and $i \in \mathbb{I}$, where β is a target error. A key question is whether to add more design points or more sample paths at existing design points. This is a difficult question and here we provide a very simple response to it; future research should lead to better answers. The difference $|\hat{Y}_h^{(-i)}(\mathbf{x}_i) - \bar{Y}_h(\mathbf{x}_i)|$ can be large because the simulation output is far from the true value (at $\mathbf{x}_i$ or at other design points) or because of a large difference between the true value $y_h(\mathbf{x}_i)$ and the leave-one-out prediction at $\mathbf{x}_i$ that would arise if the true values were known at the other design points. In the former case, we want to add more sample paths; in the latter, we want to add more design points. The problem is that we do not know the true values, so we do not know the cause. However, if the sample size at $\mathbf{x}_i$ is already large enough to make the half-width of the confidence interval for $y_h(\mathbf{x}_i)$ very small, then it is unlikely that the cause is that $\bar{Y}_h(\mathbf{x}_i)$ is far from $y_h(\mathbf{x}_i)$, so it seems attractive to add a new design point. We add a new design point if the first term $l_h(\mathbf{x}_i, n_i; \alpha) / \left(\left|\bar{Y}_h(\mathbf{x}_i)\right| - l_h(\mathbf{x}_i, n_i; \alpha)\right)$ of E_{hi} is less than $\lambda\beta$, where $\lambda \in (0, 1)$ is a parameter of the simulation procedure whose purpose is to control the effect of Monte Carlo variability during cross-validation. In our experiments, we found that $\lambda = 1/4$ worked well. An outline of Phase II of our procedure is:

(1) For $i = 1, 2, \ldots, k$, initialize the sample size $N_i \leftarrow n_i$.

(2) For $i = 1, 2, \ldots, k$ and $h \in \mathbb{H}$, compute the sample average $\bar{Y}_h(\mathbf{x}_i)$ and sample standard deviation $s_h(\mathbf{x}_i)$ of the N_i discounted payoffs of security h on each sample path at $\mathbf{x}_i$.

(3) For all $i \in \mathbb{I}$ and $h \in \mathbb{H}$, compute E_{hi} as in Equation (4.4) but with sample size N_i.

(4) Set $(h^*, i^*) \leftarrow \arg\max_{h \in \mathbb{H}, i \in \mathbb{I}} E_{hi}$. If $E_{h^*i^*} \leq \beta$, terminate.

(5) If $l_{h^*}(\mathbf{x}_{i^*}, N_{i^*}; \alpha) / \left(\bar{Y}_{h^*}(\mathbf{x}_{i^*}) - |l_{h^*}(\mathbf{x}_{i^*}, N_{i^*}; \alpha)|\right) \geq \lambda\beta$, then:

(a) Simulate N_{i^*} additional sample paths at $\mathbf{x}_{i^*}$ and set $N_{i^*} \leftarrow 2N_{i^*}$.

(b) Update $\bar{Y}_h(\mathbf{x}_{i^*})$ and $s_h(\mathbf{x}_{i^*})$ for all $h \in \mathbb{H}$.

(c) Return to Step 3.

(6) Otherwise,

(a) Add design point $\mathbf{x}_{k+1}$ midway between $\mathbf{x}_{i^*}$ and the nearest design point, and set $\mathbb{I} \leftarrow \mathbb{I} \cup \{k+1\}$.

(b) Perform two-stage simulation at $\mathbf{x}_{k+1}$ as described in Section 4.3.1, initialize $N_{k+1} \leftarrow n_{k+1}$, and compute $\bar{Y}_h(\mathbf{x}_{k+1})$ and $s_h(\mathbf{x}_{k+1})$ for all $h \in \mathbb{H}$.

(c) Set $k \leftarrow k+1$ and return to Step 3.

4.4. Numerical Experiment Results

We tested our procedure on the example of Section 4.2. There are $2^6 = 64$ corner points coming from a hypercube $\mathcal{U}_p$ of volume $p = 0.99$ and $k = 74$ total design points in Phase I. The first-stage sample size $n_0 = 5000$. The confidence level $1 - \alpha = 0.9$ and the target levels β and γ for relative error are both 0.05.

Our figure of merit is root average relative mean squared error (RARMSE): for security h, $\text{RARMSE}(h) = \sqrt{\int_{\mathcal{X}} \text{E}[(\hat{\text{Y}}_h(\boldsymbol{x})/y_h(\boldsymbol{x}) - 1)^2]\, dF_{\boldsymbol{X}}(\boldsymbol{x})}$. To analyze the performance of our procedure, we sample $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots \boldsymbol{X}_K$ independently from the distribution of tomorrow's scenario, and use this sample to approximate the integral. In these $K = 1000$ scenarios, we compare the metamodels' predictions to very accurate estimates of the true security prices obtained from

another simulation experiment. We approximate the expectation by running $m = 30$ macro-replications of our procedure and estimate RARMSE(h) by

$$\frac{1}{mK}\sum_{i=1}^{K}\sum_{j=1}^{m}\left(\frac{\hat{\mathrm{Y}}_h^{(j)}(\boldsymbol{X}_i)}{y_h(\boldsymbol{X}_i)}-1\right)^2,$$

where $\hat{\mathrm{Y}}_h^{(j)}$ is the metamodel of the price of security h in the jth macro-replication.

Figure 4.2 contains histograms of the RARMSE of the 75 metamodels produced by our procedure. One histogram is obtained when the set $\mathbb{H}$ of securities used in cross-validation contains all 75 securities, and the other when $\mathbb{H}$ contains only five securities. In the latter case, the representative in $\mathbb{H}$ of each of the five classes of options described in Section 4.2 is the option that has the longest maturity and is deepest out of the money, which therefore has the highest ratio of payoff variance to price. The figure shows that using only five securities in cross-validation increases RARMSE only slightly; the metamodels are still quite accurate, with the biggest RARMSE around 0.72%.

The advantage of using a small set $\mathbb{H}$ for cross-validation is reduced computational cost. To study this, we simulated sample paths with a time step of two days. This is not necessary when the equity indices follow geometric Brownian motion, but some other models do require simulation with small time steps. Implemented in MATLAB 7.6 and run on a computer with a 2.4GHz CPU and 3.4GB memory under 32-bit Windows XP, the procedure took 4.2 hours when including all 75 securities in cross-validation, and 2.2 hours when including only five securities. In practice, portfolios may consist of hundreds of securities, in which case more computing power is required to run our procedure in one night. Our procedure is easy to parallelize: simulation of different sample paths and building metamodels for different securities can be allocated to separate processors.

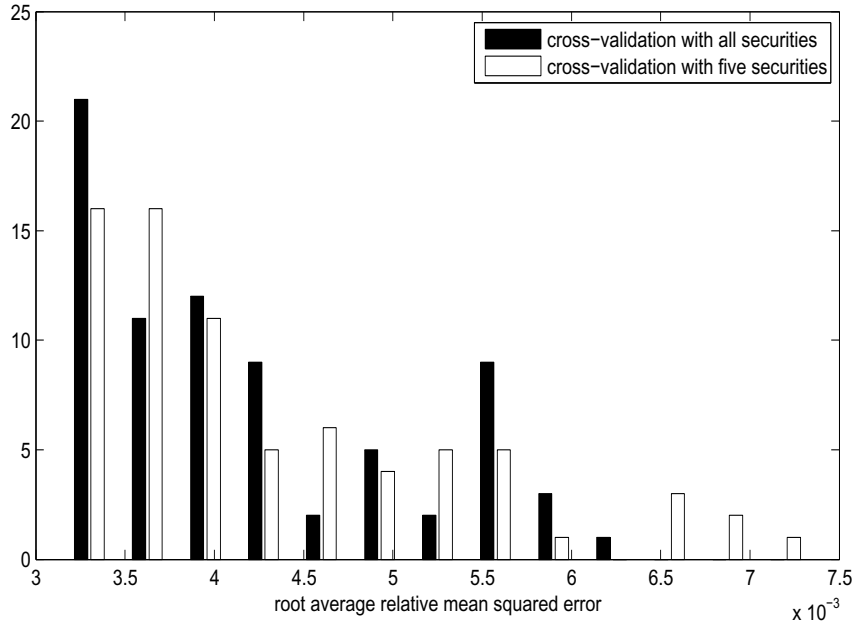

Figure 4.2. Histogram of estimated root average relative mean squared error of metamodels.

References

Ankenman, B., Nelson, B., Staum, J., 2010. Stochastic kriging for simulation metamodeling. Operations Research, forthcoming.

Barber, C. B., Dobkin, D. P., Huhdanpaa, H. T., 1996. The quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software 22 (4), 469–483.

Barton, R. R., Meckesheimer, M., 2006. Metamodel-based simulation optimization. In: Henderson, S. G., Nelson, B. L. (Eds.), Handbooks in Operations Research and Management Science: Simulation. Elsevier, New York.

Bazaraa, M., Sherali, H., Shetty, C., 2006. Nonlinear Programming Theory and Algorithms. John Wiley & Sons, Hoboken, New Jersey.

Bitran, G. R., Hax, A. C., 1981. Disaggregation and resource allocation using convex knapsack problems with bounded variables. Management Science 27, 431–441.

Boesel, J., Nelson, B. L., Kim, S., 2003. Using ranking and selection to 'clean up' after simulation optimization. Operations Research 51, 814–825.

Bretthauer, K., Ross, A., Shetty, B., 1999. Nonlinear integer programming for optimal allocation in stratified sampling. European Journal of Operational Research 116, 667–680.

Cario, M. C., Nelson, B. L., 1997. Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. Technical report, Northwestern University.

Cario, M. C., Nelson, B. L., 1998. Numerical methods for fitting and simulating autoregressive-to-anything processes. INFORMS Journal on Computing 10, 72–81.

Derman, E., 1999. Regimes of volatility. Risk 12(4), 55–59.

Durrett, R., 2005. Probability: Theory and Examples. Thomson-Brooks/Cole, Belmont, California.

Frye, J., 1998. Monte Carlo by day. Risk 11, 66–71.

Geisser, S., 1993. Predictive Inference: An Introduction. Chapman and Hall, New York.

Golub, G. H., Van Loan, C. F., 1996. Matrix Computations (3rd edition). Johns Hopkins University Press, Baltimore.

Gordy, M. B., Juneja, S., 2006. Efficient simulation for risk measurement in portfolio of CDOs. In: Perrone, L. F., Lawson, B., Liu, J., Wieland, F. P. (Eds.), Proceeedings of the 2006 Winter Simulation Conference. IEEE Press, pp. 749–756.

Gordy, M. B., Juneja, S., 2008. Nested simulation in portfolio risk measurement. Finance and Economics Discussion Series 2008-21, Federal Reserve Board.

Henderson, S. G., 2006. Mathematics for simulation. In: Henderson, S. G., Nelson, B. L. (Eds.), Simulation. Vol. 13 of Handbooks in Operations Research and Management Science. Elsevier, Amsterdam.

Kim, S.-H., Nelson, B. L., 2006. Selecting the best system. In: Henderson, S. G., Nelson, B. L. (Eds.), Simulation. Vol. 13 of Handbooks in Operations Research and Management Science. Elsevier, Amsterdam.

Kleijnen, J. P. C., 2008. Design and Analysis of Simulation Experiments. Springer-Verlag, New York.

Kleijnen, J. P. C., Beers, W. C. M., 2004. Application-driven sequential designs for simulation experiments: Kriging metamodeling. Journal of the Operational Research Society 55, 876–883.

Lan, H., 2009. Tuning the parameters of a two-level simulation procedure with screening. Working paper, Dept. of IEMS, Northwestern University.

Lan, H., Nelson, B. L., Staum, J., 2007a. A confidence interval for tail conditional expectation via two-level simulation. In: Henderson, S. G., Biller, B., Hsieh, M.-H., Shortle, J., Tew, J. D., Barton, R. R. (Eds.), Proceedings of the 2007 Winter Simulation Conference. IEEE Press, pp. 949–957.

Lan, H., Nelson, B. L., Staum, J., 2007b. Two-level simulations for risk management. In: Chick, S., Chen, C.-H., Henderson, S. G., Yücesan, E. (Eds.), Proceedings of the 2007 INFORMS Simulation Society Research Workshop. INSEAD, Fontainebleau, France, pp. 102–107.

Lan, H., Nelson, B. L., Staum, J., 2008. Confidence interval procedures for expected shortfall via two-level simulation. Working paper 08-02, Dept. of IEMS, Northwestern University.

Law, A. M., 2007. Simulation Modeling and Analysis, 4th Edition. McGraw-Hill, Boston.

Law, A. M., Kelton, W. D., 2000. Simulation Modeling and Analysis, 3rd Edition. McGraw-Hill, New York.

Lee, S.-H., 1998. Monte Carlo computation of conditional expectation quantiles. Ph.D. thesis, Stanford University.

Lesnevski, V., Nelson, B. L., Staum, J., 2007. Simulation of coherent risk measures based on generalized scenarios. Management Science 53, 1756–1769.

Lesnevski, V., Nelson, B. L., Staum, J., 2008. An adaptive procedure for simulating coherent risk measures based on generalized scenarios. Journal of Computational Finance 11, 1–31.

Liu, M., Nelson, B. L., Staum, J., 2008. An efficient simulation procedure for point estimation of expected shortfall. Working paper 08-03, Dept. of IEMS, Northwestern University.

Luenberger, D. G., 1998. Investment Science. Oxford University Press, New York.

McNeil, A. J., Frey, R., Embrechts, P., 2005. Quantitative Risk Management: Concepts, Techniques, Tools. Princeton University Press, Princeton, New Jersey.

Oakley, J., 2004. Estimating percentiles of uncertain computer code outputs. Applied Statistics 53, 83–93.

Oakley, J., O'Hagan, A., 2002. Bayesian inference for the uncertainty distribution of computer model outputs. Biometrika 89(4), 769–784.

Santner, T. J., Williams, B. J., Notz, W. I., 2003. Design and Analysis of Computer Experiments. Springer-Verlag, New York.

Shaw, J., 1998. Beyond VAR and stress testing. In: Dupire, B. (Ed.), Monte Carlo: Methodologies and Applications for Pricing and Risk Management. Risk Books, London, pp. 231–244.

Stein, C., 1945. A two-sample test for a linear hypothesis whose power is independent of the variance. Annals of Mathematical Statistics 16, 243–258.

APPENDIX A

Appendix for Chapter 2

A.1. Ranking and Selection Procedure

User input: The user specifies the computational budget C, the confidence level $1-p$ of the ES to be estimated, and k scenarios (which may be generated by an outer-level simulation).

Algorithm parameters: Choose the first-stage sample size $n_0 > 1$ and the sample size growth factor $R > 1$.

Initialization: Set $N \leftarrow 0$, $n \leftarrow n_0$, $I \leftarrow \{1, 2, \ldots, k\}$, and $j \leftarrow 0$.

Phase I:

(1) Simulate payoffs X_{ih} for $i \in I$ and $h = N+1, N+2, \ldots, N+n$ using CRN for all $i \in I$. Set $C \leftarrow C - n|I|$, $N \leftarrow N+n$, and then $n \leftarrow N(R-1)$. Calculate

$$\begin{aligned} \bar{X}_i &= \frac{1}{N}\sum_{h=1}^{N} X_{ih}, \\ S_i^2 &= \frac{1}{N-1}\sum_{h=1}^{N}(X_{ih}-\bar{X}_i)^2, \quad \text{and} \\ S_{ir}^2 &= \frac{1}{N-1}\sum_{h=1}^{N}\left(X_{ih}-X_{rh}-\left(\bar{X}_i-\bar{X}_r\right)\right)^2 \end{aligned}$$

for all $i, r \in I$ such that $r > i$. Sort $\{\bar{X}_i : i \in I\}$ to get the mapping $\pi_j(\cdot)$ and sort $\{S_i : i \in I\}$ to get the mapping $\pi_{S(j)}(\cdot)$.

(2) Use the golden section method (Bazaraa et al., 2006), or another derivative-free line search method, to find the α_j between 0 and $1/\lceil kp \rceil$ that minimizes the function $\tilde{P}(j,\cdot)$, which is computed by the following steps:

(a) For all $i, r \in I$ such that $i < r$, calculate

$$Q_{ir} = \frac{\bar{X}_i - \bar{X}_r}{S_{ir}/\sqrt{N}} \quad \text{and} \quad Q_{ri} = -Q_{ir}.$$

(b) Set $j' \leftarrow j$, $N' \leftarrow N$, $n' \leftarrow N(R-1)$, $C' \leftarrow C$, and $\tilde{I} \leftarrow I$.

(c) Set

$$\tilde{I} \leftarrow \left\{ i : \sum_{r \in \tilde{I}} 1\left\{ Q_{ir} > \frac{t_{1-\alpha_j, N'-1}}{\sqrt{N'/N}} \right\} < kp \right\}.$$

If $|\tilde{I}| = \lceil kp \rceil$, go to Step 2e.

(d) Calculate $\tau = \max\{S_{ir} : i, r \in \tilde{I}, i \neq r\}$,

$$\tilde{B} = \sum_{i=1}^{\min\{\lceil kp \rceil, |\tilde{I}| - \lceil kp \rceil\}} w_i \max_{\delta \geq 0} \left(\delta \Phi \left(\frac{-\delta}{\tau/\sqrt{N'}} \right) \right), \quad \text{and}$$

$$\tilde{V}_s = \frac{1}{C'} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_j(i)} \right)^2.$$

Set $C' \leftarrow C' - n'|\tilde{I}|$, $N' \leftarrow N' + n'$, and then $n' \leftarrow N'(R-1)$. Use the mapping $\pi_{S(j)}(\cdot)$ to construct the mapping $\pi_S(\cdot)$ from $\{1, 2, \ldots, |\tilde{I}|\}$ to $\tilde{I}$ such that $\pi_S(i)$ is the scenario in $\tilde{I}$ with the ith smallest value of S_i, and compute

$$\tilde{V}_c = \frac{1}{C'} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_S(i)} \right)^2.$$

If $\tilde{B}^2 + \tilde{V}_s > \tilde{V}_c$, set $j' \leftarrow j' + 1$ and go to Step 2c.

(e) Set $\tilde{J} = j'$ and return

$$\tilde{P}(j, \alpha_j) = \frac{(1 - \lceil kp \rceil \alpha_j)^{\tilde{J}-j+1}}{\begin{pmatrix} |\tilde{I}| \\ \lceil kp \rceil \end{pmatrix}}.$$

(3) Screening: set $I \leftarrow \left\{ i : \sum_{r \in I} 1\left\{ Q_{ir} > t_{1-\alpha_j, N-1} \right\} < kp, i \in I \right\}$. If $|I| = \lceil kp \rceil$, go to Step 5.

(4) Calculate $\tau = \max\{S_{ir}(j) : i, r \in I, i \neq r\}$,

$$\begin{aligned} B &= \sum_{i=1}^{\min\{\lceil kp \rceil, |I| - \lceil kp \rceil\}} w_i \max_{\delta \geq 0} \left(\delta \Phi \left(\frac{-\delta}{\tau / \sqrt{N}} \right) \right), \\ V_s &= \frac{1}{C} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_j(i)} \right)^2, \quad \text{and} \\ V_c &= \frac{1}{C - n|I|} \left(\sum_{i=1}^{\lceil kp \rceil} w_i S_{\pi_{S(j)}(i)} \right)^2. \end{aligned}$$

If $B^2 + V_s \geq V_c$, set $j \leftarrow j + 1$ and go to Step 1.

(5) Selection: Let $J = j$ and $\hat{\gamma} = \{\pi_J(1), \pi_J(2), \ldots, \pi_J(\lceil kp \rceil)\}$.

Phase II: For each $i = 1, 2, \ldots, \lceil kp \rceil$, compute the sample size

$$M_{\pi_J(i)} = C \frac{w_i S_{\pi_J(i)}}{\sum_{r=1}^{\lceil kp \rceil} w_r S_{\pi_J(r)}}.$$

Restart, discarding all previously simulated payoffs. For each $i \in \hat{\gamma}$, simulate M_i independent payoffs and calculate their sample average $\bar{X}_i$. Compute the expected

shortfall estimator

$$\widehat{ES}_{1-p} = \sum_{i=1}^{\lceil kp \rceil} w_i \bar{X}_{\pi_J(i)}.$$

A.2. Derivations

A.2.1. The Probability of Correct Selection

In §2.4.3, we defined $\Pr\{\text{CS}_j\} = \Pr\{\gamma \cap I_j \subseteq \hat{\gamma}\}$ as the probability of selecting, at the end of Phase I, all tail scenarios that had survived to stage j. Here we derive an approximation $\tilde{P}(j, \alpha)$ for this probability, given the information available at stage j, and making explicit the dependence on the error level α, which we need to choose. Although $\tilde{P}(j, \alpha)$ is not $\Pr\{\text{CS}_j\}$, it has similar properties. The choice α_j that maximizes $\tilde{P}(j, \cdot)$ may not maximize $\Pr\{\text{CS}_j\}$, but we can reasonably expect that it makes $\Pr\{\text{CS}_j\}$ large.

To derive $\tilde{P}(j, \alpha)$ as a function of α, recall that we are imagining that error level α will be used at all stages from j onward (§2.4.3). We also make the following forecast about the sample averages and variances in future stages:

$$\text{(A.1)} \qquad \forall j' \geq j,\ i, r \in I_{j'},\ \bar{X}_i(j') = \bar{X}_i(j),\ S_i^2(j') = S_i^2(j),\ \text{and}\ S_{ir}^2(j') = S_{ir}^2(j),$$

that is, all sample averages and variances will remain the same. Of course, the sample averages and variances will actually change from stage to stage, but given the information available at stage j, this is the obvious way to forecast them. Then, we can further forecast the number $\tilde{J}(j, \alpha)$ of stages in Phase I (Appendix A.2.1.1), get an approximation $\tilde{P}_{j'}(j, \alpha)$ to the probability of no screening mistakes at each stage $j' = j, j+1, \ldots, \tilde{J}(j, \alpha)$ (Appendix A.2.1.2), and get a lower bound $\tilde{P}_s(j, \alpha)$ on the probability of no selection mistakes at the end of Phase I.

We construct the approximation $\tilde{P}(j,\alpha)$ by treating screening mistakes at every stage and selection mistakes at the end of Phase I as if they were independent. Because the event CS_j is the event that there are no screening and selection mistakes from stage j to the end of Phase I, we define $\tilde{P}(j,\alpha)$ to have the structure

$$\tilde{P}(j,\alpha) = \left(\prod_{j'=j}^{\tilde{J}(j,\alpha)} \tilde{P}_{j'}(j,\alpha) \right) \tilde{P}_s(j,\alpha), \tag{A.2}$$

where $\tilde{P}_{j'}(j,\alpha)$ relates to the probability of a screening mistake at stage j' and $\tilde{P}_s(j,\alpha)$ relates to the probability of a selection mistake.

The lower bound

$$\tilde{P}_s(j,\alpha) = \left(\begin{array}{c} |\tilde{I}(j,\alpha)| \\ \lceil kp \rceil \end{array} \right)^{-1},$$

where $\tilde{I}(j,\alpha)$ is the set of scenarios that are forecast to survive until the end of Phase I. This is the probability of guessing blindly and correctly selecting $\lceil kp \rceil$ scenarios out of the $|\tilde{I}(j,\alpha)|$ that survive until the end of Phase I. Putting this result together with those derived in the remainder of this section, we get

$$\tilde{P}(j,\alpha) = \frac{(1-\lceil kp \rceil\, \alpha)^{\tilde{J}(j,\alpha)-j+1}}{\left(\begin{array}{c} |\tilde{I}(j,\alpha)| \\ \lceil kp \rceil \end{array} \right)}.$$

Then we choose α_j in the range $(0, 1/\lceil kp \rceil)$ to maximize $\tilde{P}(j)$.

A.2.1.1. Forecasting the Results of Screening. Here we explain how to forecast, as of stage j, the final stage $\tilde{J}(j,\alpha)$ of Phase I and the sets of surviving scenarios $\tilde{I}_{j'}(j,\alpha)$ for $j' = j+1, j+2, \ldots, \tilde{J}(j)+1$. A detailed procedure appears in Step 2 of Appendix A.1.

(1) Set $\tilde{I}_j(j,\alpha) \leftarrow I_j$ and $j' \leftarrow j$.

(2) Based on Equation (A.1) and our method of screening (Step 2c of Appendix A.1), forecast that the following scenarios will survive screening at stage j':

$$\text{(A.3)} \qquad \tilde{I}_{j'+1}(j,\alpha) = \left\{ i : \sum_{r \in \tilde{I}_{j'}(j,\alpha)} 1\left\{ \frac{\bar{X}_i(j) - \bar{X}_r(j)}{S_{ir}(j)/\sqrt{N_{j'}}} > t_{1-\alpha,N_{j'}-1} \right\} < kp \right\}.$$

(3) Evaluate the stopping rule (§2.4.5) by plugging in $\bar{X}_i(j)$ for $\bar{X}_i(j')$, $\pi_j(i)$ for $\pi_{j'}(i)$, $S_i^2(j)$ for $S_i^2(j')$, $\pi_{S(j)}(i)$ for $\pi_{S(j')}(i)$, and $\tilde{I}_{j'+1}(j,\alpha)$ for $I_{j'+1}$. If the stopping rule is satisfied, $\tilde{J}(j,\alpha) = j'$. Otherwise, set $j' \leftarrow j'+1$ and return to the previous step.

Because the forecast remaining computational budget C' in Step 2 in Appendix A.1 decreases at each stage, the stopping rule must be satisfied at a finite stage $\tilde{J}(j,\alpha)$.

Since $N_{j'} = N_j R^{j'-j}$ where $R > 1$, Equation (A.3) can be rewritten as

$$\tilde{I}_{j'+1}(j,\alpha) = \left\{ i : \sum_{r=1}^{k} 1\left\{ \frac{\bar{X}_i(j) - \bar{X}_r(j)}{S_{ir}(j)/\sqrt{N_j}} > \underbrace{\frac{t_{1-\alpha,N_{j'}-1}}{\sqrt{R^{j'-j}}}}_{W_{j'}(j,\alpha)} \right\} < kp \right\},$$

in which the threshold $W_{j'}(j,\alpha)$ is the only thing that depends on j' or α. Thus, although we are forecasting the results of screening at several future stages for multiple values of α, this does not take nearly as long as actually performing screening repeatedly.

A.2.1.2. The Probability of Screening Mistakes. To simplify the following analysis, we assume that the values of distinct scenarios $i \neq r$ are distinct: $V_i \neq V_r$. The experimental results

in §2.5.1 show that even when this assumption does not hold, our procedure can still perform well. Also continuing to treat α as the error level at all stages $j' \geq j$, we analyze the probability that no screening mistakes occur at stage j' as follows:

$$\Pr\left\{\sum_{r\in\tilde{I}_{j'}(j,\alpha)} 1\left\{\bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_{j'}-1}S_{ir}(j)}{\sqrt{N_{j'}}}\right\} < kp, \forall i \in \gamma \cap \tilde{I}_{j'}(j,\alpha)\right\}$$

$$\text{(A.4)} \quad \geq \quad 1 - \sum_{i\in\gamma\cap\tilde{I}_{j'}(j,\alpha)} \Pr\left\{\sum_{r\in\tilde{I}_{j'}(j,\alpha)} 1\left\{\bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_j-1}S_{ir}(j)}{\sqrt{N_j}}\right\} \geq kp\right\}$$

$$\text{(A.5)} \quad \geq \quad 1 - \sum_{i\in\gamma} \Pr\left\{\sum_{r\in\tilde{I}_{j'}(j,\alpha)} 1\left\{\bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_j-1}S_{ir}(j)}{\sqrt{N_j}}\right\} \geq kp\right\}$$

where (A.4) is based on the Bonferroni inequality and Equation (A.1). In (A.5) we are being conservative by considering the possibility of making a mistake by screening out any scenario $i \in \gamma$, regardless of whether it has survived to stage j': throughout this section, we imagine the random variables $X_{i1}, X_{i2}, \ldots, X_{iN_j}$ as existing, even if they were not all simulated.

To analyze the probabilities in (A.5), we consider the probability of a non-tail scenario $r \in I_0 \setminus \gamma$ beating a tail scenario $i \in \gamma$, which is

$$\Pr\left\{\bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_{j'}-1}S_{ir}(j)}{\sqrt{N_{j'}}}\right\} \quad = \quad \Pr\left\{\frac{\bar{X}_i(j) - \bar{X}_r(j)}{S_{ir}(j)/\sqrt{N_{j'}}} > t_{1-\alpha,N_{j'}-1}\right\}$$

$$\text{(A.6)} \qquad\qquad\qquad\qquad \approx \quad 1 - \Phi\left(z_{1-\alpha} + \frac{V_r - V_i}{\sigma_{ir}/\sqrt{N_{j'}}}\right)$$

where $\Phi(\cdot)$ is the standard normal distribution function and $z_{1-\alpha_j}$ is its $1-\alpha_j$ quantile. Next we consider the relative likelihood for various non-tail scenarios to beat a particular tail scenario i. Let r_i be the non-tail scenario that minimizes $(V_r - V_i)/\sigma_{ir}$. When $N_{j'}$ is large, the probability

that r_i beats i will dominate the probability that any other non-tail scenario beats i. This follows from Equation (A.6) and the exponential decay of the probability $1 - \Phi(x)$ as $x \to \infty$: for positive x, $(x^{-1} - x^{-3})(2\pi)^{-1/2} \exp(-x^2/2) \leq 1 - \Phi(x) \leq x^{-1}(2\pi)^{-1/2} \exp(-x^2/2)$ (Durrett, 2005). We use the negligible probability that any non-tail scenario other than r_i beats i to get the approximation (A.7) in the following derivation:

$$
\begin{aligned}
&\Pr\left\{ \sum_{r \in \tilde{I}_{j'}(j,\alpha)} 1\left\{ \bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_{j'}-1} S_{ir}(j)}{\sqrt{N_{j'}}} \right\} \geq kp \right\} \\
&\quad \leq \Pr\left\{ \exists r \in \tilde{I}_{j'}(j,\alpha) \setminus \gamma \ni \bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_{j'}-1} S_{ir}(j)}{\sqrt{N_{j'}}} \right\} \\
&\quad \leq \Pr\left\{ \exists r \notin \gamma \ni \bar{X}_i(j) > \bar{X}_r(j) + \frac{t_{1-\alpha,N_{j'}-1} S_{ir}(j)}{\sqrt{N_{j'}}} \right\} \\
&\quad \approx \Pr\left\{ \bar{X}_i(j) > \bar{X}_{r_i}(j) + \frac{t_{1-\alpha,N_{j'}-1} S_{ir_i(j)}(j)}{\sqrt{N_{j'}}} \right\} \qquad \text{(A.7)} \\
&\quad \leq \Pr\left\{ \frac{\bar{X}_{r_i}(j) - \bar{X}_i(j) - (V_{r_i} - V_i)}{S_{ir_i(j)}/\sqrt{N_{j'}}} < t_{\alpha,N_{j'}-1} \right\} \qquad \text{(A.8)} \\
&\quad \approx \alpha_j \qquad \text{(A.9)}
\end{aligned}
$$

where Equation (A.8) holds because $V_i \leq V_{r_i}$ and $t_{\alpha,N_{j'}-1} = -t_{1-\alpha_j,N_{j'}-1}$. From Equations (A.5) and (A.9) we obtain an approximation to the probability that there are no screening mistakes at stage j', which is $\tilde{P}_{j'}(j,\alpha) = 1 - \lceil kp \rceil \alpha$. Roughly speaking, this corresponds to thinking of any of $\lceil kp \rceil$ tail scenarios as being vulnerable to being screened out if it is beaten by all other tail scenarios and one specific non-tail scenario, and assigning probability α to the event of being beaten by that non-tail scenario.

A.2.2. Bias Estimation for the Choice of Stopping Screening

As explained in §2.4.5, we only need to consider the bias that arises from selecting $\hat{\gamma}$ from I_{j+1} if we stop after stage j. We only consider the bias induced by incorrect selection, that is, $\hat{\gamma} = \{\pi_j(i) : i = 1, 2, \ldots, \lceil kp \rceil\} \neq \gamma = \{\pi_V(i) : i = 1, 2, \ldots, \lceil kp \rceil\}$. We ignore the bias induced by ordering tail scenarios in $\hat{\gamma}$ incorrectly; this is unimportant because at most one weight among $w_1, w_2, \ldots, w_{\lceil kp \rceil}$ is different from the others.

For each $i \leq \lceil kp \rceil$, the bias induced by the possibility of excluding scenario $\pi_V(i)$ is

$$\begin{aligned} b_i &= w_i \sum_{r \in I_{j+1} \setminus \gamma} \left(V_r - V_{\pi_V(i)}\right) \Pr\left\{r \in \hat{\gamma}, \pi_V(i) \notin \hat{\gamma}\right\} \\ &\leq w_i \sum_{r \notin \gamma} \left(V_r - V_{\pi_V(i)}\right) \Pr\left\{\bar{X}_r(j) < \bar{X}_{\pi_V(i)}(j)\right\}. \end{aligned}$$

Using the approach in Appendix A.2.1.2, any of these probabilities is dominated by

$$q_i(j) = \Pr\left\{\bar{X}_{r_{\pi_V(i)}}(j) < \bar{X}_{\pi_V(i)}(j)\right\}.$$

To condense notation, define $\delta_i := V_{r_i} - V_i$ and $\tau_i := \sigma_{r_i i}$ for all $i \in \gamma$. When N_j is large, $q_i(j) \approx \Phi\left(-\delta_{\pi_V(i)}\sqrt{N_j}/\tau_{\pi_V(i)}\right)$. Then we approximate b_i by $w_i\delta_{\pi_V(i)}\Phi\left(-\delta_{\pi_V(i)}\sqrt{N_j}/\tau_{\pi_V(i)}\right)$. However, $\tau_{\pi_V(i)}$ and $\delta_{\pi_V(i)}$ are unknown. For $\tau_{\pi_V(i)}$ we substitute $\tau_j = \max\{S_{ir}(j) : i, r \in I_{j+1}, i \neq r\}$, which makes selection mistakes seem more likely, and thus increases the absolute value of our approximation of the bias. We also increase it by replacing $\delta_{\pi_V(i)}$ with $\arg\max_{\delta \geq 0} \delta\Phi(-\delta\sqrt{N_j}/\tau_j)$. Then we approximate

$$b_i \approx w_i \max_{\delta \geq 0} \delta\Phi\left(-\delta\sqrt{N_j}/\tau_j\right).$$

To approximate the total bias due to the possibility of excluding tail scenarios, we consider two cases: when $|I_{j+1}| \geq 2\lceil kp \rceil$ and when $|I_{j+1}| < 2\lceil kp \rceil$. When $|I_{j+1}| \geq 2\lceil kp \rceil$, it is possible that all tail scenarios are excluded. When $|I_{j+1}| < 2\lceil kp \rceil$, at most $|I_{j+1}| - \lceil kp \rceil$ tail scenarios can be excluded, because that is how many non-tail scenarios have survived screening. Thus we approximate the bias by

$$B(j) = \sum_{i=1}^{\min\{\lceil kp \rceil, |I_{j+1}| - \lceil kp \rceil\}} w_i \max_{\delta \geq 0} \left(\delta \Phi \left(-\delta \sqrt{N_j}/\tau_j \right) \right).$$

APPENDIX B

Appendix for Chapter 3

B.1. Optimal Budget Allocation in Stage III

In Stage III of our procedure, we want to minimize the posterior variance of the ES estimator in Equation (3.3) by allocating C inner-level simulation replications among the k design points, subject to the constraint that we have already allocated n_0 replications to each of the design points. ES is $-\sum_{i=1}^{Kp} \mathsf{Y}_{(i)}/p$, where $\mathsf{Y}_{(i)}$ is the ith lowest component of the vector $\mathbf{Y}^K$ of P&L at each prediction point. In a Bayesian interpretation of the stochastic kriging framework, ES is a random variable because $\mathbf{Y}^K$ is a random variable. Its posterior variance, given the simulation data observed in Stages I and II, is difficult to analyze, because uncertainty about $\mathbf{Y}^K$ means there is uncertainty about the order of its components. We simplify the problem by supposing, for the moment, that the order is known. That is, we consider the random variable $\boldsymbol{w}^\top \mathbf{Y}^K$ where the weight w_i is $-1/Kp$ if i is a tail scenario and 0 otherwise, treating the vector $\boldsymbol{w}$ as though it were known.

We refer to Section 3.3 for definitions of the notation in the following derivation of posterior variance. The prior distribution of the P&L and the simulation output $\left[\mathbf{Y}^K; \boldsymbol{Y}\right]$ is multivariate normal with mean vector and covariance matrix

$$\begin{bmatrix} \beta_0 \mathbf{1}^K \\ \beta_0 \mathbf{1}^k \end{bmatrix} \text{ and } \begin{bmatrix} \boldsymbol{\Sigma}^{KK} & \boldsymbol{\Sigma}^{Kk} \\ \boldsymbol{\Sigma}^{kK} & \boldsymbol{\Sigma} \end{bmatrix}$$

(Ankenman et al., 2010). Therefore $\left[\boldsymbol{w}^\top \mathbf{Y}^K; \boldsymbol{Y}\right]$ also has a multivariate normal prior distribution with mean vector and covariance matrix

$$\begin{bmatrix} -\beta_0 \\ \beta_0 \mathbf{1}^k \end{bmatrix} \text{ and } \begin{bmatrix} \boldsymbol{w}^\top \boldsymbol{\Sigma}^{KK} \boldsymbol{w} & \boldsymbol{w}^\top \boldsymbol{\Sigma}^{Kk} \\ \boldsymbol{\Sigma}^{kK} \boldsymbol{w} & \boldsymbol{\Sigma} \end{bmatrix}.$$

Then the posterior variance of $\boldsymbol{w}^\top \mathbf{Y}^K$ given $\boldsymbol{Y}$ is

$$\text{Var}\left[\boldsymbol{w}^\top \mathbf{Y}^K | \boldsymbol{Y}\right] = \boldsymbol{w}^\top \left(\boldsymbol{\Sigma}^{KK} - \boldsymbol{\Sigma}^{Kk} \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}^{kK}\right) \boldsymbol{w}. \tag{B.1}$$

The dependence of the posterior variance on the decision variable $\boldsymbol{n}$, which specifies the number of simulation replications for each design point, is buried in the matrix $\boldsymbol{\Sigma}$.

The dependence on the decision variable through the inverse of a matrix makes the optimization problem difficult to analyze. To make it more tractable, we resort to a further approximation that is justified if n_0 is large. The sum of intrinsic and extrinsic covariance matrices for the design points, $\boldsymbol{\Sigma}$, can be written as

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk} \boldsymbol{N}^{-1} = \boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0 - (\boldsymbol{C}^{kk}/n_0 - \boldsymbol{C}^{kk} \boldsymbol{N}^{-1}) = \boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0 - \boldsymbol{B}\boldsymbol{B}$$

where $\boldsymbol{B}$ is a diagonal matrix whose ith element is $\sqrt{\text{V}(\mathbf{x}_i)(1/n_0 - 1/n_i)}$. By the Sherman-Morrison-Woodbury formula (Golub and Van Loan, 1996), where $\boldsymbol{I}$ is the identity,

$$\begin{aligned} \boldsymbol{\Sigma}^{-1} &= \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0 + \boldsymbol{B}(-\boldsymbol{I})\boldsymbol{B}\right)^{-1} \\ &= \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0\right)^{-1} \\ &\quad - \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0\right)^{-1} \boldsymbol{B} \left(\boldsymbol{B} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0\right)^{-1} \boldsymbol{B} - \boldsymbol{I}\right)^{-1} \boldsymbol{B} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{C}^{kk}/n_0\right)^{-1}. \end{aligned}$$

When n_0 is large enough, $\boldsymbol{C}^{kk}/n_0$ and hence $\boldsymbol{B}$ will be small. Because the extrinsic covariance matrix $\boldsymbol{\Sigma}^{kk}$ does not depend on n_0, $\boldsymbol{B}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{B}$ is negligible compared to $\boldsymbol{I}$. This leads to the approximation

$$\begin{aligned}\boldsymbol{\Sigma}^{-1} &\approx \left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}-\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{B}(-\boldsymbol{I})\boldsymbol{B}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\\ &= \left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}+\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\left(\boldsymbol{C}^{kk}/n_0-\boldsymbol{C}^{kk}\boldsymbol{N}^{-1}\right)\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}. \qquad \text{(B.2)}\end{aligned}$$

Substituting Equation (B.2) into Equation (B.1), we get the following approximation for the posterior variance:

$$\begin{aligned}\text{Var}\left[\boldsymbol{w}^\top\mathbf{Y}^K|\boldsymbol{Y}\right] &\approx \boldsymbol{w}^\top\boldsymbol{\Sigma}^{KK}\boldsymbol{w}-\boldsymbol{w}^\top\boldsymbol{\Sigma}^{Kk}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{\Sigma}^{kK}\boldsymbol{w}\\ &\quad -\boldsymbol{w}^\top\boldsymbol{\Sigma}^{Kk}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\left(\boldsymbol{C}^{kk}/n_0\right)\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{\Sigma}^{kK}\boldsymbol{w}\\ &\quad +\boldsymbol{w}^\top\boldsymbol{\Sigma}^{Kk}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{C}^{kk}\boldsymbol{N}^{-1}\left(\boldsymbol{\Sigma}^{kk}+\boldsymbol{C}^{kk}/n_0\right)^{-1}\boldsymbol{\Sigma}^{kK}\boldsymbol{w}. \qquad \text{(B.3)}\end{aligned}$$

Only the third line of Equation (B.3) depends on the decision variable $\boldsymbol{n}$, through the matrix $\boldsymbol{N}^{-1}$. Therefore, our goal is to minimize this term, which is the objective $\boldsymbol{U}^\top\boldsymbol{C}^{kk}\boldsymbol{N}^{-1}\boldsymbol{U}$ in the optimization problem (3.4).