Stochastic Kriging for Efficient Nested Simulation of Expected Shortfall

Ming Liu
Ph.D. Candidate
Department of Industrial Engineering and Management Sciences
Robert R. McCormick School of Engineering and Applied Science
Northwestern University
Evanston, IL 60208-3119, U.S.A.
ming-liu@northwestern.edu
(847) 467-6260

Jeremy Staum
Associate Professor
Department of Industrial Engineering and Management Sciences
Robert R. McCormick School of Engineering and Applied Science
Northwestern University
Evanston, IL 60208-3119, U.S.A.
j-staum@northwestern.edu
(847) 491-2405

February 1, 2010

This paper is based upon work supported by the National Science Foundation under Grant No. DMI-0555485. The authors are grateful for the assistance of Hai Lan, discussions with Barry Nelson, and comments from Lisa Goldberg and Michael Hayes. Portions of this paper were published in the *Proceedings of the 2009 Winter Simulation Conference* under the title "Estimating Expected Shortfall with Stochastic Kriging."

Abstract

We use stochastic kriging, a metamodeling technique, to speed up nested simulation of expected shortfall, a portfolio risk measure. Evaluating a risk measure of a portfolio that includes derivative securities may require nested Monte Carlo simulation. The outer level simulates financial scenarios and the inner level of simulation estimates the portfolio value given a scenario. Spatial metamodeling enables inference about portfolio values in a scenario based on inner-level simulation of nearby scenarios, reducing the required computational effort: it is not necessary to perform inner-level simulation in every scenario. Because expected shortfall involves the scenarios that entail the largest losses, our procedure adaptively allocates more computational effort to inner-level simulation of those scenarios, which also improves computational efficiency.

Keywords: simulation, stochastic kriging, expected shortfall, conditional value at risk, tail conditional expectation, conditional tail expectation

1 Introduction

Evaluating risk measures of a portfolio may require nested simulation, especially when the portfolio contains derivative securities. 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. For example, inner-level simulation of derivative securities' payoffs in the distant future provides Monte Carlo estimates of these derivatives' values in a given scenario (Glasserman, 2004). Constructing a stochastic model of the financial markets that is adequate for risk management, and estimating its parameters, can be challenging. In this article, we assume that this task of modeling and estimation has already been performed, and we propose a remedy for another principal obstacle to successful implementation of nested risk management simulations—the large computational cost of simulating many payoffs in each of many scenarios.

In this article, we focus on expected shortfall (ES) as the risk measure. ES can be interpreted as the average of the largest losses, in the tail of the loss distribution. In particular, 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

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

where $\mathsf{Y}_{(i)}$ is the ith smallest P&L. We refer to the scenarios whose P&L is among the Kp smallest as *tail scenarios*: they belong to the tail of the loss distribution and appear in Equation (1). We refer to the other scenarios as *non-tail scenarios*. For further background on ES, we refer to Acerbi and Tasche (2002) or Liu et al. (2008).

The literature on computational efficiency of nested risk management simulations, addressing estimation of value at risk (VaR) and ES, has two branches. One branch focuses on choosing the number of inner-level simulation replications. The number of replications may depend on the scenario, but it must be strictly positive for each scenario. This branch of the literature also deals with quantifying and reducing the bias that arises due to inner-level sampling error. For a brief literature review, see Liu et al. (2008). The other branch of the literature, exemplified by Frye (1998) and Shaw (1998), proposes 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. Interpolation makes sense when there is a spatial structure of scenarios. For example, in Figure 1 below, a scenario consists of the prices of two stocks, and it lies in the positive orthant of the real plane. The present article draws upon ideas from both branches of the literature.

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., 2008). 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 Liu et al. (2008). 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 Liu et al. (2008).

The rest of the paper is structured as follows. First we give a motivating example of a risk management simulation problem in Section 2. In Section 3, we review stochastic kriging and show how to use it to estimate ES. We present our new simulation procedure in Section 4. In Section 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 and directions for future research in Section 6.

2 Motivating Example

The example is almost identical to the one we considered in Liu et al. (2008), 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 1. The example differs from that of Liu et al. (2008) only in the portfolio's positions in the options; we explain the reason for considering a different portfolio in Section 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.

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

Table 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%

volatility given in Table 1. Figure 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.

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 2. The other version uses Monte Carlo simulation, specifying a bivariate lognormal distribution for the pair of stock prices. For details, see Liu et al. (2008).

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 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., 2008) is an interpolation-based metamodeling tech-

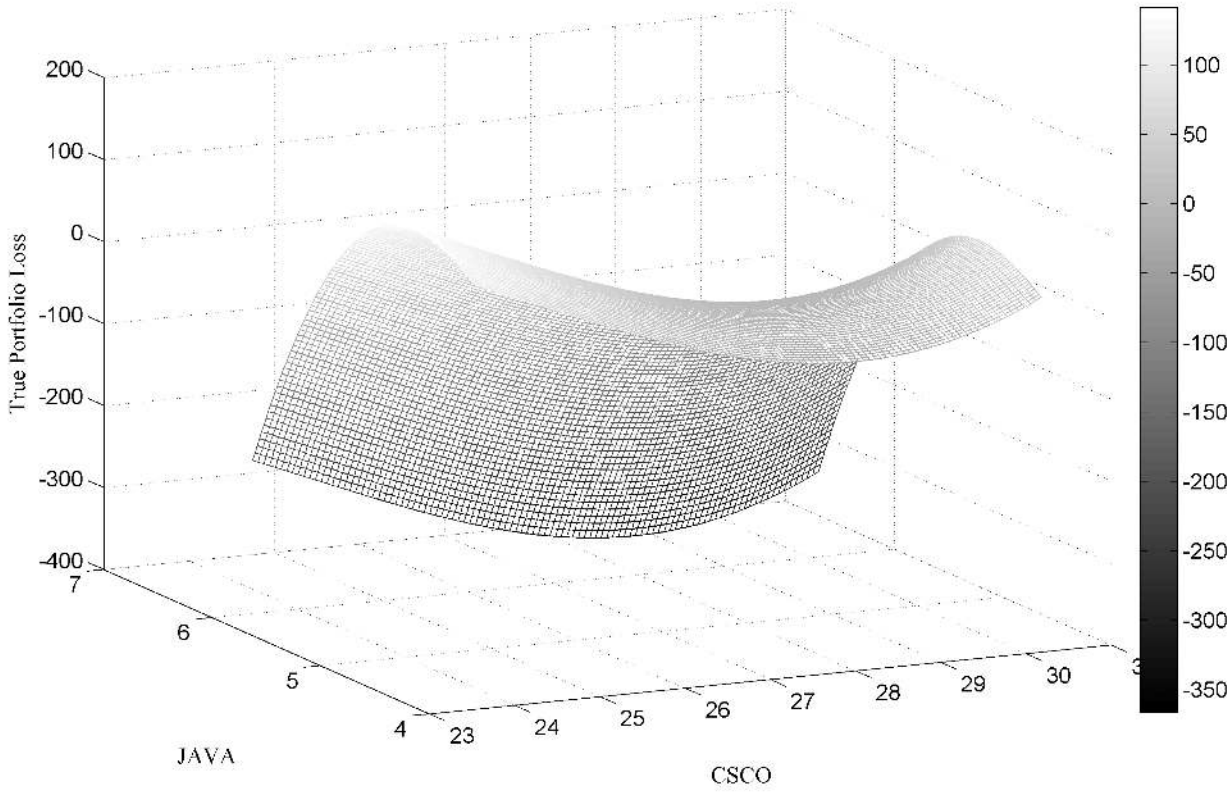

Figure 1: Portfolio Loss as a Function of Scenarios for Tomorrow's Stock Prices of Cisco (CSCO) and Sun Microsystems (JAVA).

nique. 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. (2008).

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. (2008) call this *extrinsic uncertainty*. We adopt a model frequently used in kriging, under

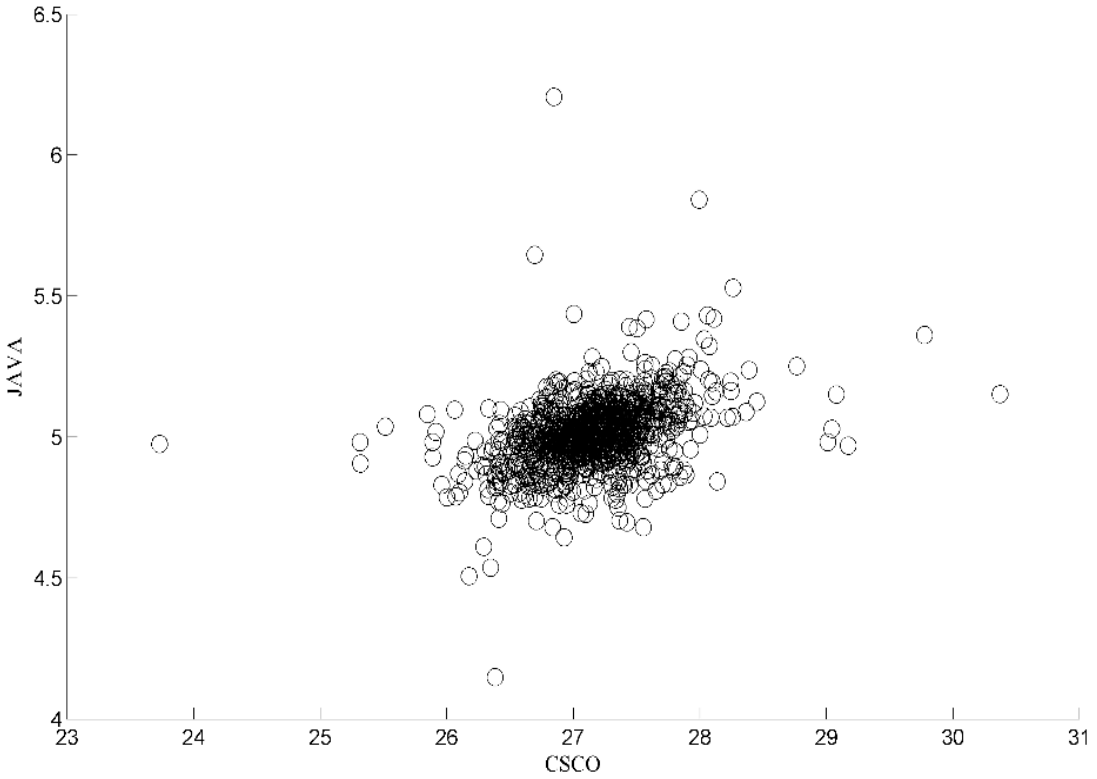

Figure 2: Scatter plot of 1000 scenarios from historical simulation.

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

$$\mathcal{Y}_j(\mathbf{x}) = \beta_0 + \mathsf{M}(\mathbf{x}) + \varepsilon_j(\mathbf{x}),$$

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

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

which is an estimator of the P&L $\mathsf{Y}(\mathbf{x}_i)$. Let $\bar{\mathcal{Y}} := [\bar{\mathcal{Y}}(\mathbf{x}_1),\ldots,\bar{\mathcal{Y}}(\mathbf{x}_k)]^\top$ 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 $\bar{\mathcal{Y}}-\mathbf{Y}^k$ is diagonal. It equals $\boldsymbol{V}\boldsymbol{N}^{-1}$ where $\boldsymbol{V}$ and $\boldsymbol{N}$ are diagonal matrices whose ith elements are respectively $\mathsf{V}(\mathbf{x}_i)$ and n_i. Define $\boldsymbol{\Sigma} := \boldsymbol{V}\boldsymbol{N}^{-1}+\boldsymbol{\Sigma}^{kk}$, 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 $\bar{\mathcal{Y}}$,

$$\widehat{\mathbf{Y}}^K = \beta_0\mathbf{1}^K + \boldsymbol{\Sigma}^{Kk}\boldsymbol{\Sigma}^{-1}(\bar{\mathcal{Y}}-\beta_0\mathbf{1}^k). \tag{2}$$

Ankenman et al. (2008) also give the covariance matrix of the Bayesian posterior distribution of $\mathbf{Y}^K$, which we use in Section 4.3.

Equation (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. (2008), 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 (2) with these estimates plugged in. Let $\widehat{\mathsf{Y}}_i$ 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 Liu et al. (2008). Our estimator of ES based on the kriging metamodel is

$$\widehat{\mathrm{ES}}_{1-p} = -\frac{1}{Kp}\sum_{i=1}^{Kp}\widehat{\mathsf{Y}}_{(i)} \tag{3}$$

where $\widehat{\mathsf{Y}}_{(i)}$ is the ith lowest value among the stochastic kriging predictions $\widehat{\mathsf{Y}}_1,\ldots,\widehat{\mathsf{Y}}_K$ at the prediction points; cf. Equation (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 $\widehat{\mathbf{Y}}^K$ 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 $\bar{\mathcal{Y}}(\mathbf{x}_i)$ is the average of these n_i replications. The simulation output is $\bar{\mathcal{Y}} := [\bar{\mathcal{Y}}(\mathbf{x}_1), \ldots, \bar{\mathcal{Y}}(\mathbf{x}_k)]^\top$.
- The variance of the simulation output for a single replication at design point $\mathbf{x}_i$ is $\mathsf{V}(\mathbf{x}_i)$, and $\boldsymbol{V}$ 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.

4 Procedure

In this section, we present our simulation procedure for estimating ES using stochastic kriging. We provide an outline in Section 4.1 and supply the details in subsequent sections.

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) 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 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 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

Stage I.

1. Generate K prediction points through outer-level simulation (historical or Monte Carlo). See Figure 2.
2. Given these prediction points, generate Stage I design points. See Section 4.2 and Figure 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 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 4.3 and Figure 6.
2. Simulate n_0 replications for the new Stage II design points. Based on the simulation outputs, create a stochastic kriging metamodel (Figure 7).

Stage III.

1. Allocate the remaining computational budget to all design points. See Section 4.4 and Figure 8.
2. Perform further simulation at the design points. Based on the simulation outputs, create a stochastic kriging metamodel (Figure 9).
3. Compute the ES estimator in Equation (3) using the final metamodel.

Figure 3: Outline of the procedure.

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. (2008, §3.3) discuss how to structure an experiment design for stochastic kriging, but not in the context 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.
- In order 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.

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 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 4 to Figure 2, which shows the prediction points.

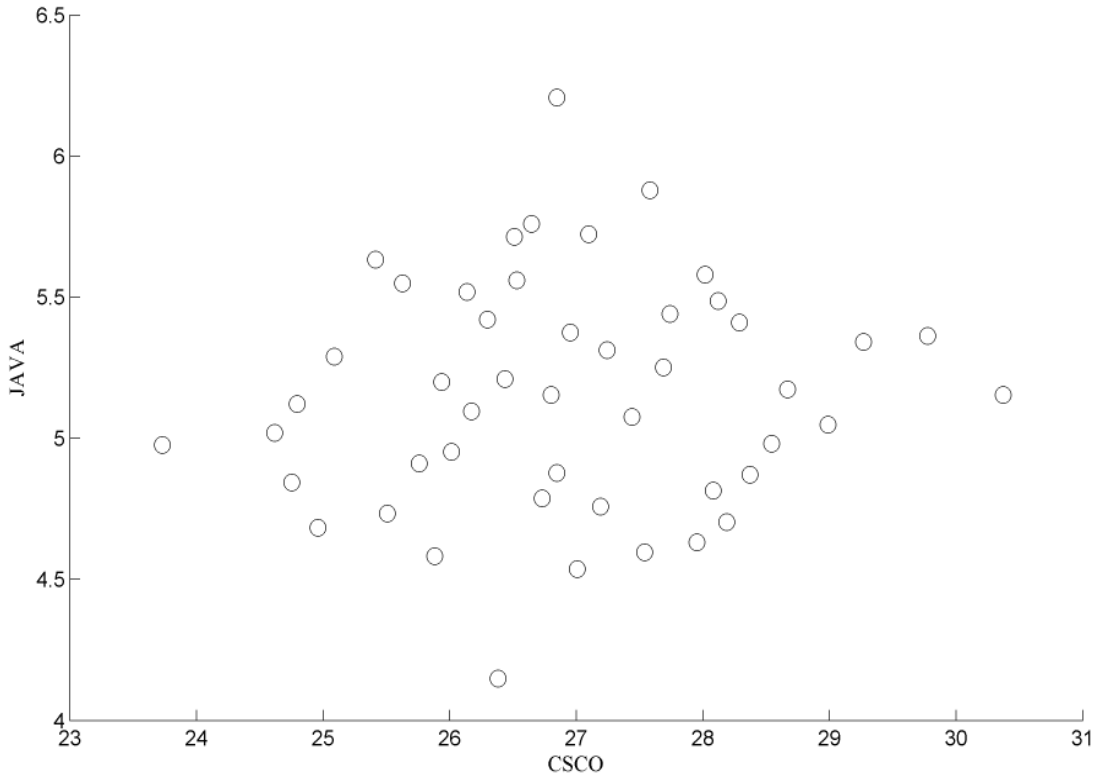

Figure 4: Design points chosen in Stage I on one run of the procedure.

Figure 5 shows the absolute value of the error $\widehat{\mathsf{Y}} - \mathsf{Y}$ 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 5 with the magnitude of P&L in Figure 1. We will see how the error shrinks after subsequent stages of the procedure.

4.3 Choosing Stage II Design Points

By comparing Equations (1) and (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.

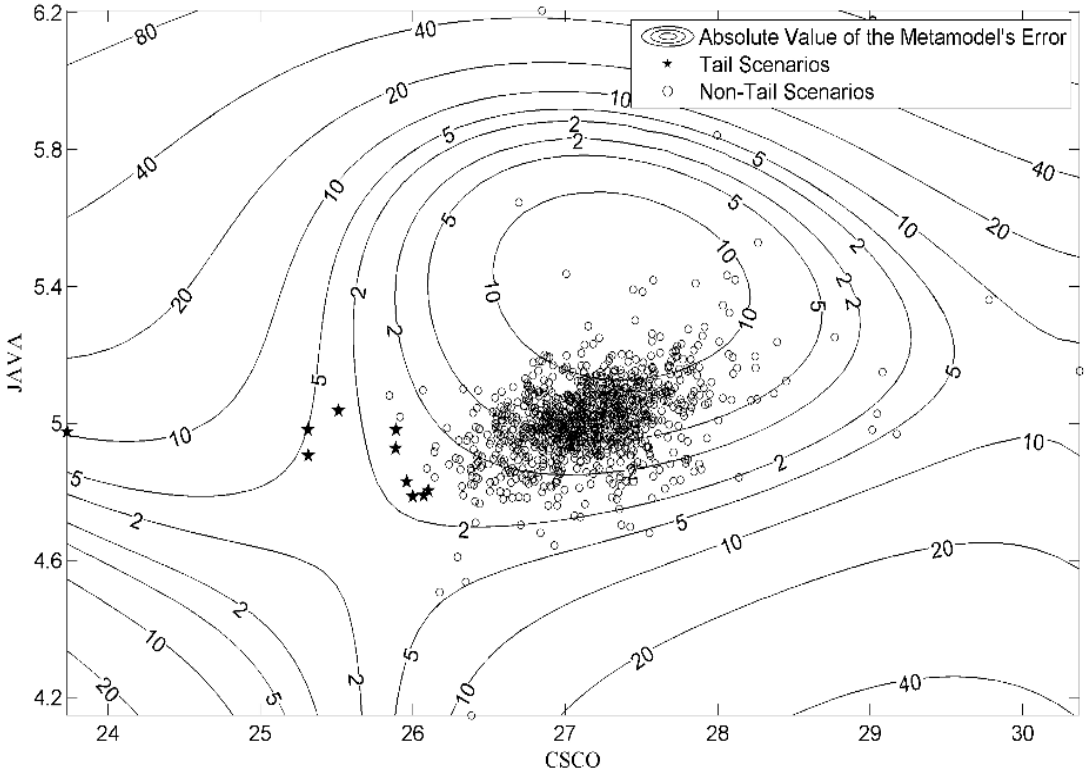

Figure 5: Absolute value of the error of the Stage I metamodel on one run of the procedure.

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., 2008). 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 $\widetilde{\mathsf{Y}}_i$ 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 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 Liu et al. (2008) 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 7 shows the absolute value of the error $\widehat{\mathsf{Y}} - \mathsf{Y}$ of the stochastic kriging metamodel built in Stage II on this run of the procedure.

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). In Appendix A, we show how to solve a simplified version of that minimization problem by solving the optimization problem (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 that $\boldsymbol{V}$ 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 $\boldsymbol{U} := (\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0)^{-1}\boldsymbol{\Sigma}^{kK}\boldsymbol{w}$. The optimization problem is to

$$\text{minimize} \quad \boldsymbol{U}^\top \boldsymbol{V}\boldsymbol{N}^{-1}\boldsymbol{U} \quad \text{subject to} \quad \boldsymbol{n}^\top \mathbf{1}^k = C,\ \boldsymbol{n} \geq n_0. \tag{4}$$

In practice, we use maximum likelihood estimates of $\boldsymbol{\Sigma}^{kk}$ and $\boldsymbol{\Sigma}^{kK}$ and we use sample variances in estimating $\boldsymbol{V}$, as discussed in Section 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 4.3.

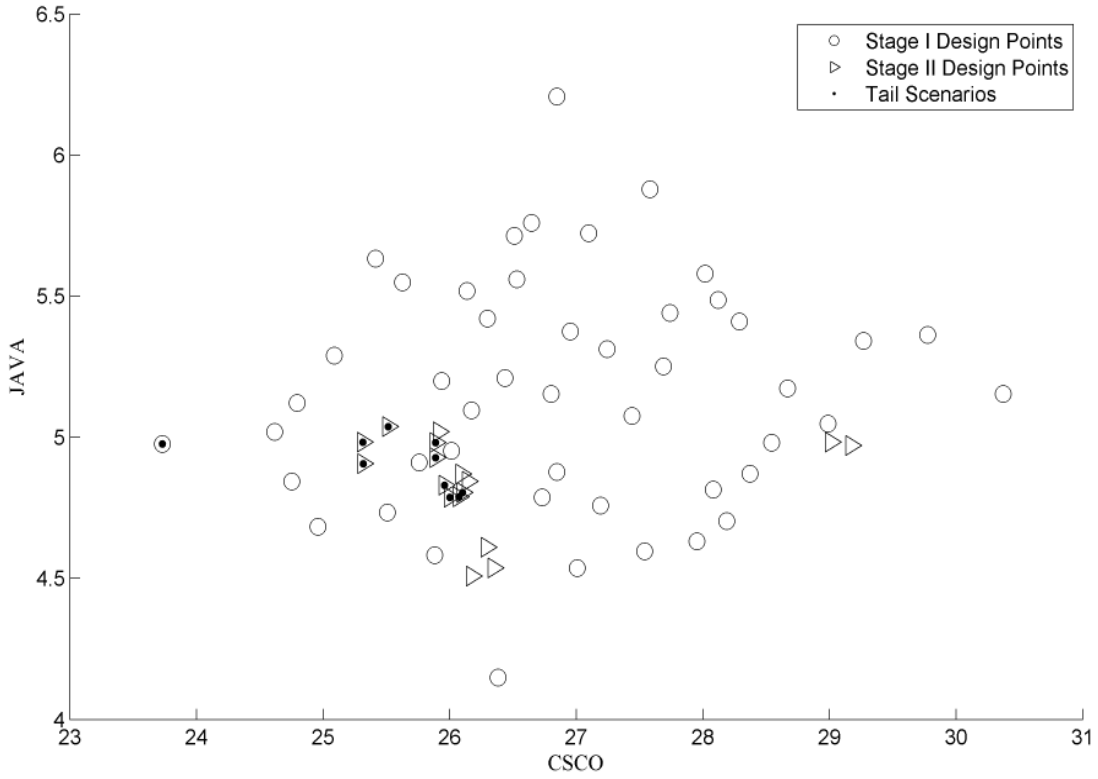

Figure 6: Design points chosen in Stages I and II on one run of the procedure.

The optimization problem (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 $n_i(m) = C(m)U_i\sqrt{\mathrm{V}(\mathbf{x}_i)}/\sum_{j\in I(m)} U_j\sqrt{\mathrm{V}(\mathbf{x}_j)}$.

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.

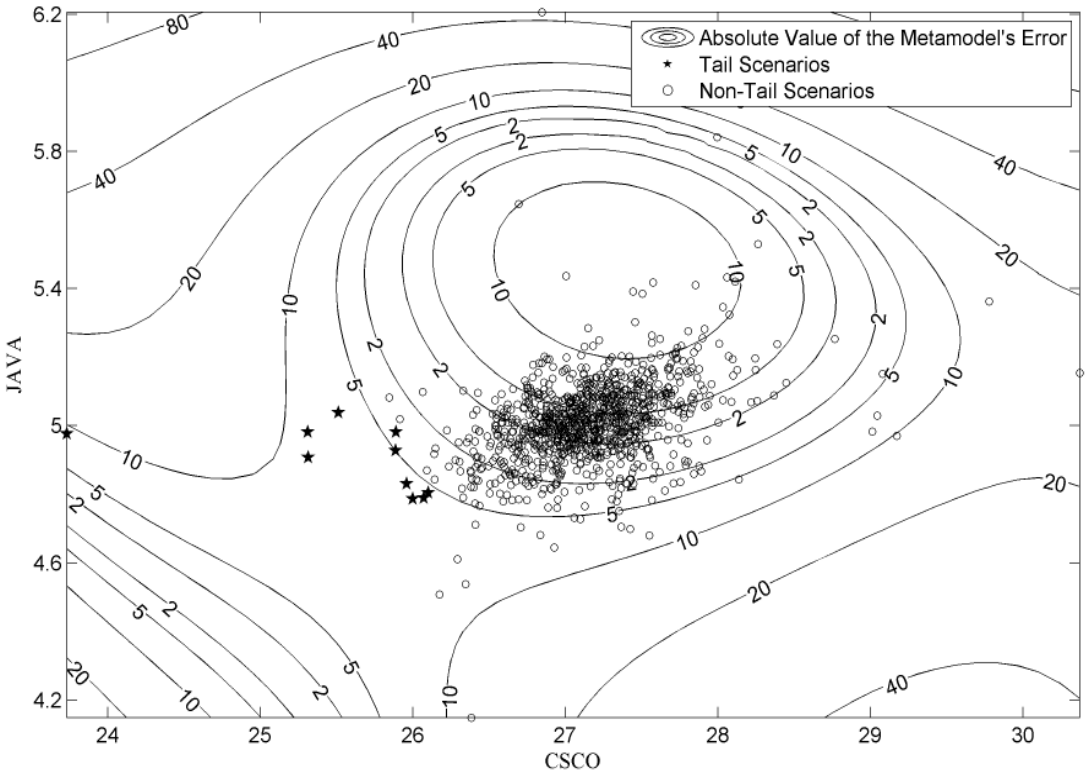

Figure 7: Absolute value of the error of the Stage II metamodel on one run of the procedure.

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

Figure 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 9 shows the absolute value of the error $\widehat{\mathsf{Y}} - \mathsf{Y}$ of the stochastic kriging metamodel built in Stage III on this run of the procedure. Comparing Figure 9 with Figure 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.

5 Numerical Study

To illustrate the performance of our procedure, we use the example described in Section 2. We present the results of simulation experiments to compare our procedure, which we call

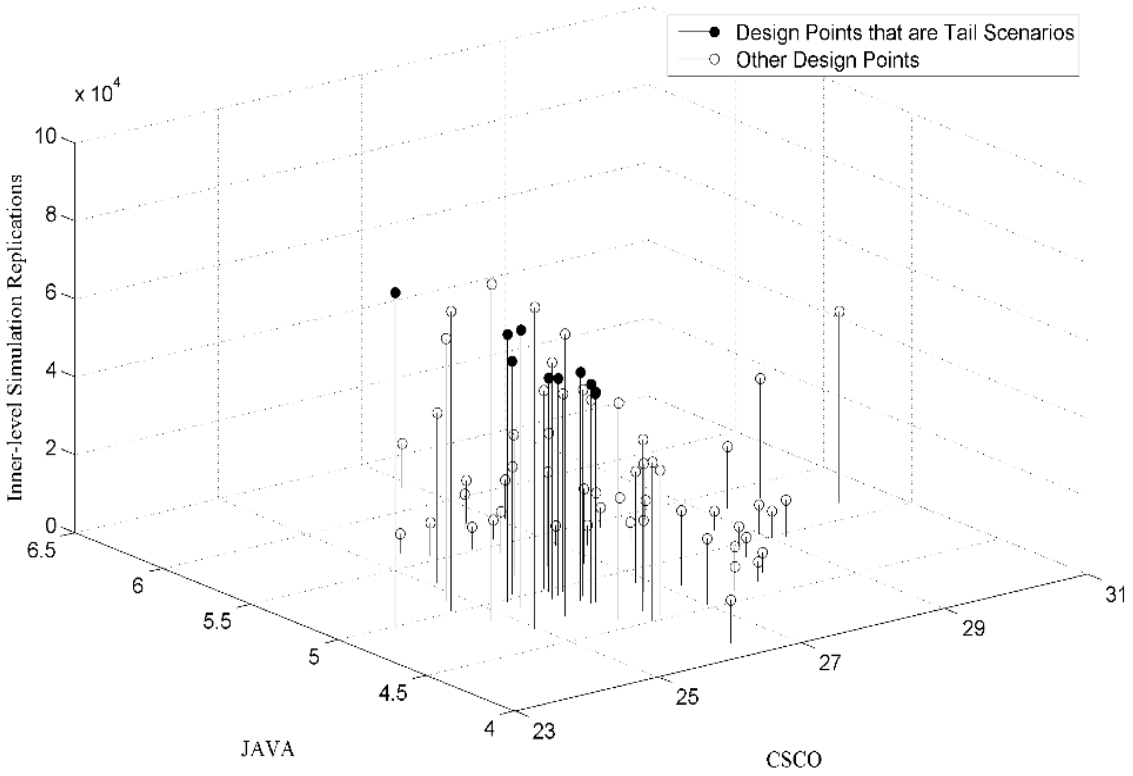

Figure 8: Number of simulation replications allocated to each design point on one run of the procedure.

the "SK procedure," to two other procedures. One is the procedure, based on methods of statistical ranking and selection, that we proposed in Liu et al. (2008), 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 Liu et al. (2008). 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.

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

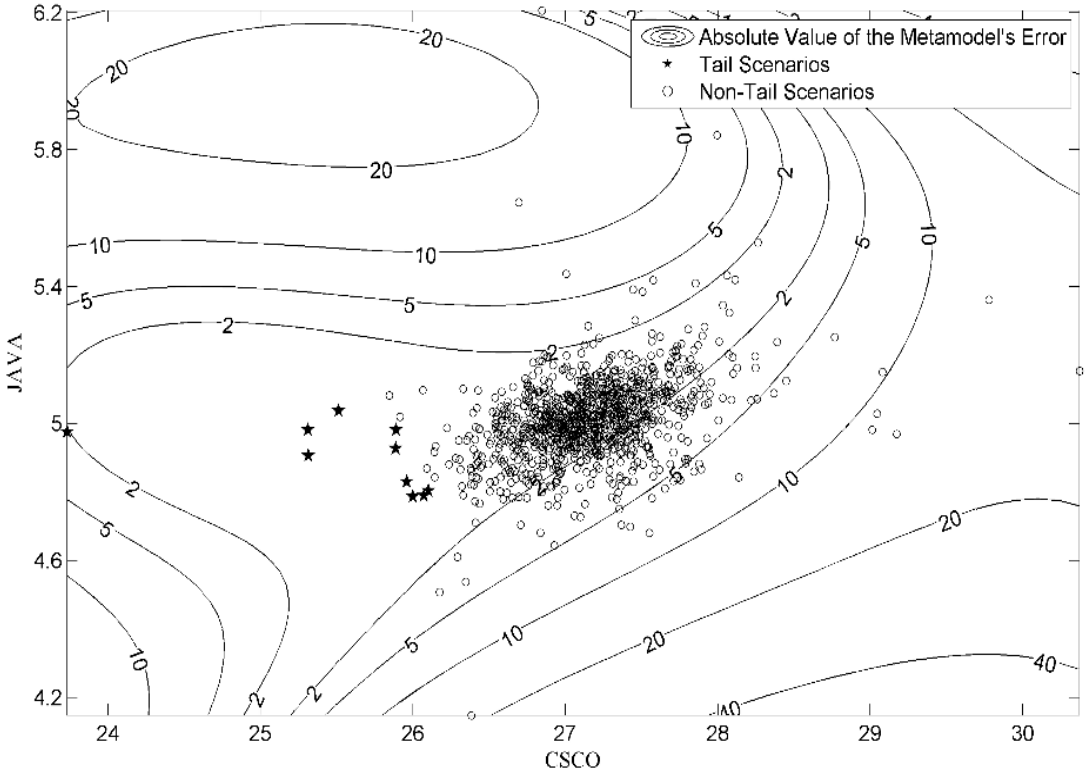

Figure 9: Absolute value of the error of the Stage III metamodel on one run of the procedure.

$k_1 = 50$ 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 Liu et al. (2008). We run 1000 macro-replications of the simulation experiments. Figure 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 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 a moderate range of budgets and do not know under what conditions, if

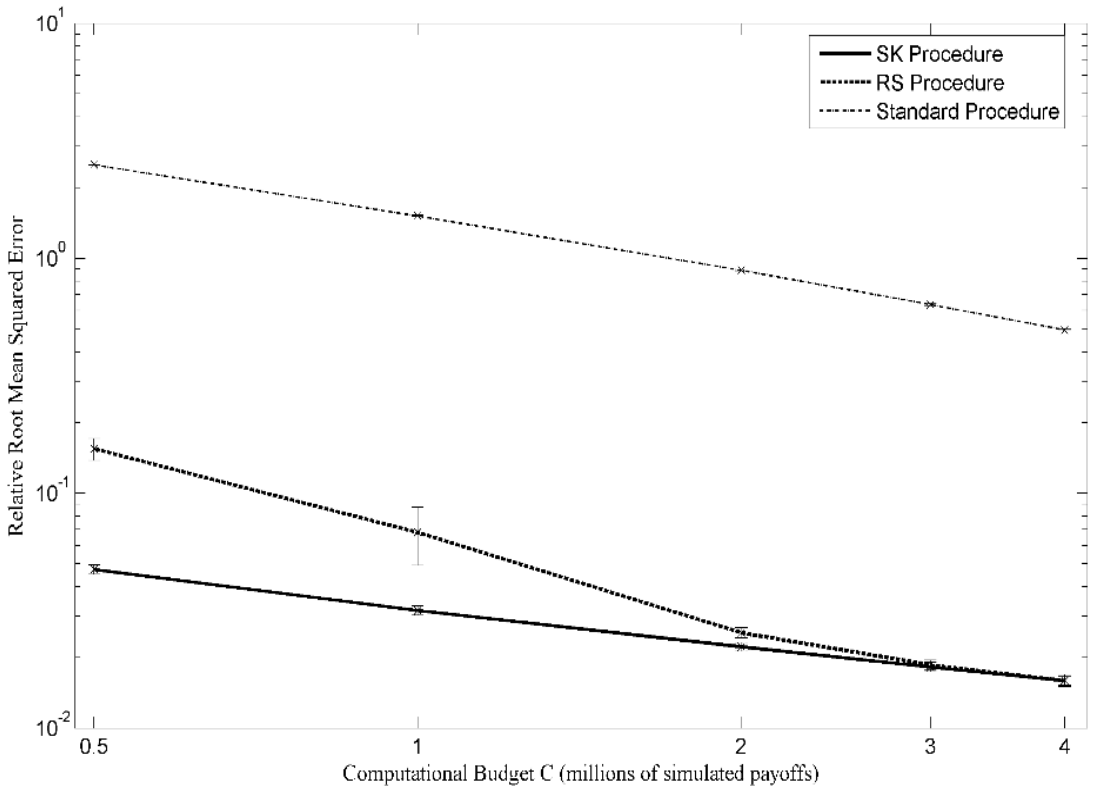

Figure 10: Accuracy in estimating expected shortfall at the 99% level for the historical simulation example.

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 11.

Comparing Figure 10 and Figure 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

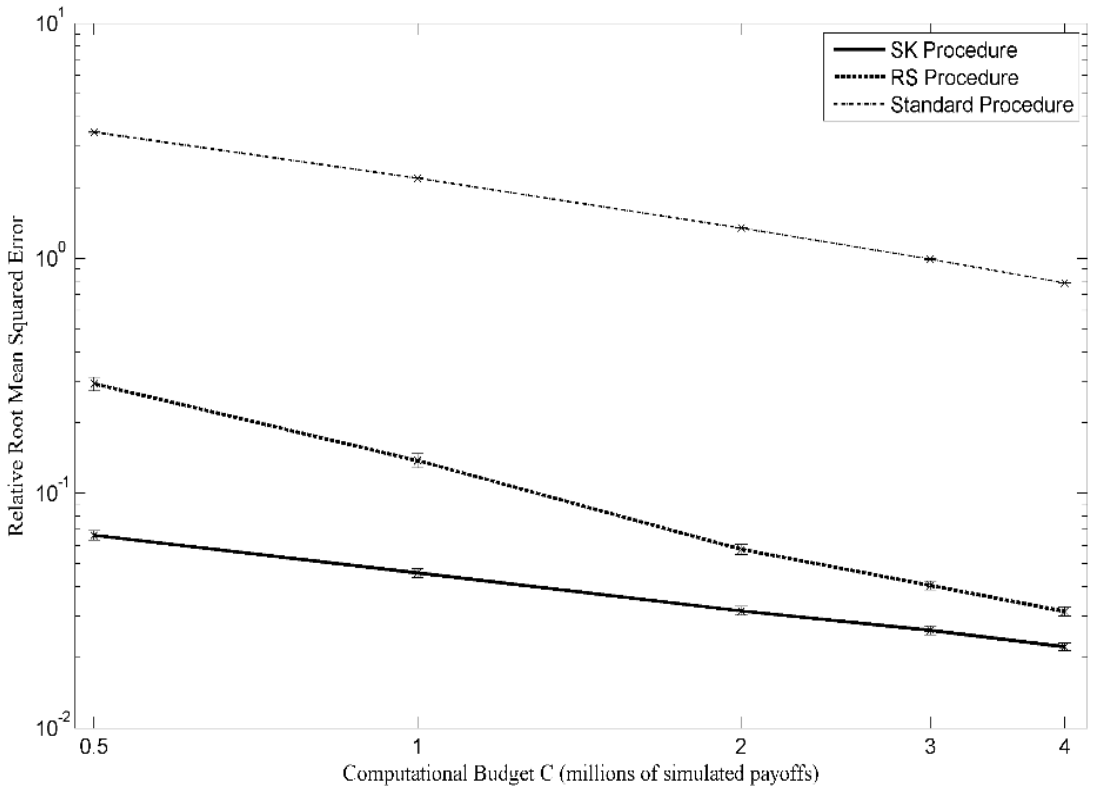

Figure 11: Accuracy in estimating expected shortfall at the 95% level for the historical simulation example.

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 (Liu et al., 2008). 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 (Liu et al., 2008). 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., 2007; Gordy and Juneja, 2008). Stochastic kriging mitigates this problem by smoothing the estimated P&L across neighboring scenarios.

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 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.

In Figure 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 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 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 12, for many simulation problems, the SK procedure with

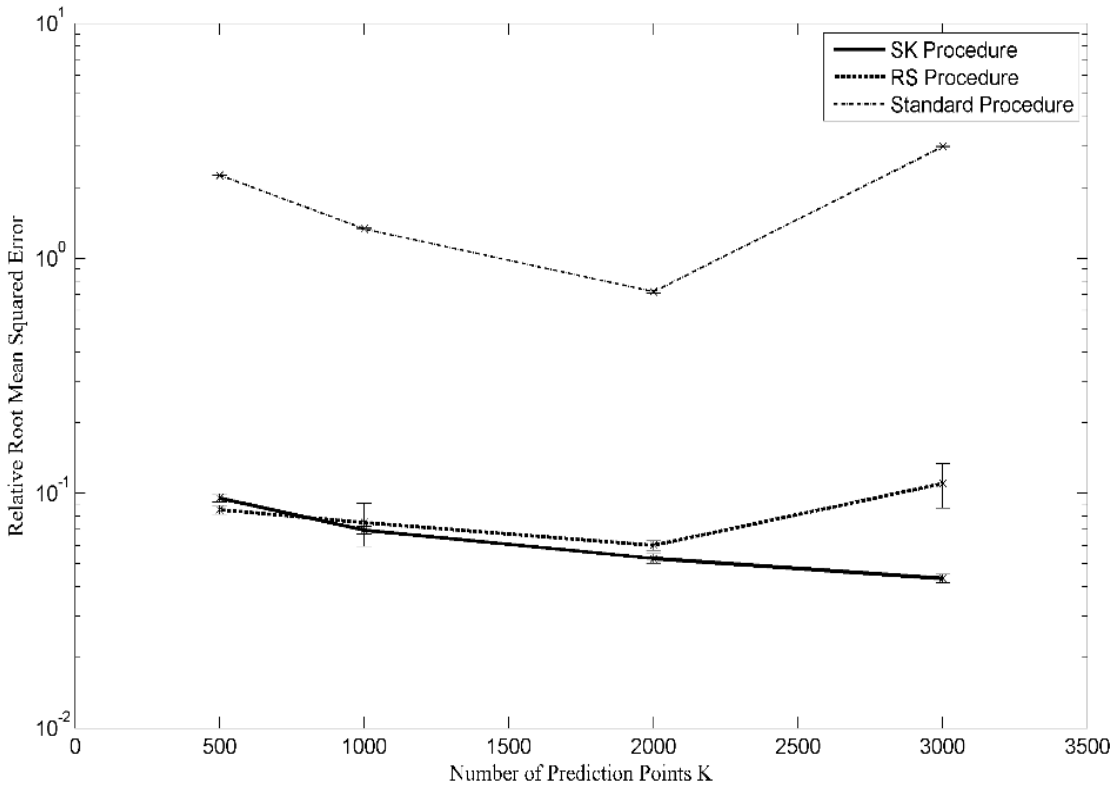

Figure 12: Accuracy in estimating expected shortfall at the 99% level for the two-level simulation example.

large K performs better than the standard and RS procedures with optimal K.

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.

References

Acerbi, C., Tasche, D., 2002. On the coherence of expected shortfall. Journal of Banking and Finance 26 (7), 1487–1503.

Ankenman, B., Nelson, B. L., Staum, J., 2008. 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, Ch. 19.

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

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

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

Glasserman, P., 2004. Monte Carlo Methods in Financial Engineering. Springer-Verlag, New York.

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

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

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., 2007. 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, available via `http://www.informs-cs.org/2007informs-csworkshop/23.pdf`.

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.

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

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.

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.

A 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) 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 for definitions of the notation in the following derivation of posterior variance. The prior distribution of the P&L and the simulation output $\left[\mathbf{Y}^K; \bar{\mathcal{Y}}\right]$ 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., 2008). Therefore $\left[\boldsymbol{w}^\top \mathbf{Y}^K; \bar{\mathcal{Y}}\right]$ 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 $\bar{\mathcal{Y}}$ is

$$\text{Var}\left[\boldsymbol{w}^\top \mathbf{Y}^K | \bar{\mathcal{Y}}\right] = \boldsymbol{w}^\top \left(\boldsymbol{\Sigma}^{KK} - \boldsymbol{\Sigma}^{Kk} \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}^{kK}\right) \boldsymbol{w}. \tag{5}$$

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

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

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

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

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

$$\begin{aligned}
\boldsymbol{\Sigma}^{-1} &\approx \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} - \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \boldsymbol{B}(-\boldsymbol{I})\boldsymbol{B} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \\
&= \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} + \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \left(\boldsymbol{V}/n_0 - \boldsymbol{V}\boldsymbol{N}^{-1}\right) \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1}. \quad (6)
\end{aligned}$$

Substituting Equation (6) into Equation (5), we get the following approximation for the posterior variance:

$$\begin{aligned}
\mathrm{Var}\left[\boldsymbol{w}^\top \mathbf{Y}^K | \bar{\mathcal{Y}}\right] &\approx \boldsymbol{w}^\top \boldsymbol{\Sigma}^{KK} \boldsymbol{w} - \boldsymbol{w}^\top \boldsymbol{\Sigma}^{Kk} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \boldsymbol{\Sigma}^{kK} \boldsymbol{w} \\
&\quad - \boldsymbol{w}^\top \boldsymbol{\Sigma}^{Kk} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} (\boldsymbol{V}/n_0) \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \boldsymbol{\Sigma}^{kK} \boldsymbol{w} \\
&\quad + \boldsymbol{w}^\top \boldsymbol{\Sigma}^{Kk} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \boldsymbol{V}\boldsymbol{N}^{-1} \left(\boldsymbol{\Sigma}^{kk} + \boldsymbol{V}/n_0\right)^{-1} \boldsymbol{\Sigma}^{kK} \boldsymbol{w}. \quad (7)
\end{aligned}$$

Only the third line of Equation (7) 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 $\boldsymbol{U}^\top \boldsymbol{V}\boldsymbol{N}^{-1}\boldsymbol{U}$ in the optimization problem (4).