INTERNATIONAL SERIES IN OPERATIONS RESEARCH AND MANAGEMENT SCIENCE

Design and Analysis of Simulation Experiments

Jack P.C. Kleijnen

Springer

Design and Analysis of Simulation Experiments

Recent titles in the INTERNATIONAL SERIES IN OPERATIONS RESEARCH & MANAGEMENT SCIENCE

Frederick S. Hillier, Series Editor, *Stanford University*

Sethi, Yan & Zhang/ *INVENTORY AND SUPPLY CHAIN MANAGEMENT WITH FORECAST UPDATES*

Cox/ *QUANTITATIVE HEALTH RISK ANALYSIS METHODS: Modeling the Human Health Impacts of Antibiotics Used in Food Animals*

Ching & Ng/ *MARKOV CHAINS: Models, Algorithms and Applications*

Li & Sun/ *NONLINEAR INTEGER PROGRAMMING*

Kaliszewski/ *SOFT COMPUTING FOR COMPLEX MULTIPLE CRITERIA DECISION MAKING*

Bouyssou et al/ *EVALUATION AND DECISION MODELS WITH MULTIPLE CRITERIA: Stepping stones for the analyst*

Blecker & Friedrich/ *MASS CUSTOMIZATION: Challenges and Solutions*

Appa, Pitsoulis & Williams/ *HANDBOOK ON MODELLING FOR DISCRETE OPTIMIZATION*

Herrmann/ *HANDBOOK OF PRODUCTION SCHEDULING*

Axsäter/ *INVENTORY CONTROL, 2nd Ed.*

Hall/ *PATIENT FLOW: Reducing Delay in Healthcare Delivery*

Józefowska & Węglarz/ *PERSPECTIVES IN MODERN PROJECT SCHEDULING*

Tian & Zhang/ *VACATION QUEUEING MODELS: Theory and Applications*

Yan, Yin & Zhang/ *STOCHASTIC PROCESSES, OPTIMIZATION, AND CONTROL THEORY APPLICATIONS IN FINANCIAL ENGINEERING, QUEUEING NETWORKS, AND MANUFACTURING SYSTEMS*

Saaty & Vargas/ *DECISION MAKING WITH THE ANALYTIC NETWORK PROCESS: Economic, Political, Social & Technological Applications w. Benefits, Opportunities, Costs & Risks*

Yu/ *TECHNOLOGY PORTFOLIO PLANNING AND MANAGEMENT: Practical Concepts and Tools*

Kandiller/ *PRINCIPLES OF MATHEMATICS IN OPERATIONS RESEARCH*

Lee & Lee/ *BUILDING SUPPLY CHAIN EXCELLENCE IN EMERGING ECONOMIES*

Weintraub/ *MANAGEMENT OF NATURAL RESOURCES: A Handbook of Operations Research Models, Algorithms, and Implementations*

Hooker/ *INTEGRATED METHODS FOR OPTIMIZATION*

Dawande et al/ *THROUGHPUT OPTIMIZATION IN ROBOTIC CELLS*

Friesz/ *NETWORK SCIENCE, NONLINEAR SCIENCE and INFRASTRUCTURE SYSTEMS*

Cai, Sha & Wong/ *TIME-VARYING NETWORK OPTIMIZATION*

Mamon & Elliott/ *HIDDEN MARKOV MODELS IN FINANCE*

del Castillo/ *PROCESS OPTIMIZATION: A Statistical Approach*

Józefowska/*JUST-IN-TIME SCHEDULING: Models & Algorithms for Computer & Manufacturing Systems*

Yu, Wang & Lai/ *FOREIGN-EXCHANGE-RATE FORECASTING WITH ARTIFICIAL NEURAL NETWORKS*

Beyer et al/ *MARKOVIAN DEMAND INVENTORY MODELS*

Shi & Olafsson/ *NESTED PARTITIONS OPTIMIZATION: Methodology And Applications*

Samaniego/ *SYSTEM SIGNATURES AND THEIR APPLICATIONS IN ENGINEERING RELIABILITY*

**** A list of the early publications in the series is at the end of the book ****

Design and Analysis of Simulation Experiments

Jack P.C. Kleijnen
Tilburg University, Tilburg, the Netherlands

Springer

Jack P.C. Kleijnen
Tilburg University
Tilburg, The Netherlands

Series Editor:
Fred Hillier
Stanford University
Stanford, CA, USA

Library of Congress Control Number: 2007926589

ISBN-13: 978-0-387-71812-5 e-ISBN-13: 978-0-387-71813-2

Printed on acid-free paper.

© 2008 by Springer Science+Business Media, LLC
All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now know or hereafter developed is forbidden.
The use in this publication of trade names, trademarks, service marks and similar terms, even if the are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

9 8 7 6 5 4 3 2 1

springer.com

To my wife, Wilma

Preface

This book is the successor of several other books that I wrote on (roughly) the same topic. My first book consisted of two volumes, and was published in 1974/1975 (and translated into Russian in 1978). Its successor was published in 1987. In 1992, Willem van Groenendaal and I wrote a more general book on simulation, which included an update of parts of my 1987 book. So I thought that it was high time to write down all I know about the statistical Design and Analysis of Simulation Experiments, which I abbreviate to DASE (and pronounce as the girl's name Daisy). This acronym is inspired by DACE, which stands for Design and Analysis of Computer Experiments; the acronym DACE is popular in deterministic simulation.

In this book, I will focus on those DASE aspects that I have a certain expertise in—I think.

Though I focus on DASE for discrete-event simulation (which includes queueing and inventory simulations), I also discuss DASE for deterministic simulation (applied in engineering, physics, etc.).

I discuss both computationally expensive and cheap simulations.

I assume that the readers already have a basic knowledge of simulation; e.g., they know concepts such as terminating simulation and steady-state simulation. They should also have a basic understanding of mathematical statistics, including concepts such as distribution functions, averages, and variances.

This book contains more than four hundred references. Yet, I have tried to eliminate older references that are mentioned in more recent references—unless the older reference is the origin of some important idea (so the readers may get a historical perspective). To improve the book's readability, I try to collect references at the end of paragraphs—as much as seems reasonable.

I recommend that the first three chapters be read in their implied order. The next chapters, however, are independent of each other, so they may be read in the order that best suits the interest of the individual reader.

I wrote this book in a foreign language (namely English, whereas Dutch is my mother tongue), so style, spelling, etc. may sometimes not be perfect: my apologies. Concerning style, I point out that I place redundant information between parentheses; the em-dash (or —) signals nonredundant, extra information. To enable readers to browse through the various chapters, I repeat the definition of an abbreviation in a given chapter—even if that abbreviation has already been defined in a preceding chapter. The book contains paragraphs starting with the word "Note", which upon first reading may be skipped.

Each website address is displayed on a separate line, because a website address may be so long that it either runs over into the right margin of the page or it must be hyphenated—but then the hyphen may be interpreted as part of the address. A comma or a period at the end of the address is not part of the address!

I wrote this book in Scientific Workplace, which also helped me (through its MuPAD computational engine) to solve some of the exercises that I formulated in this book. Winfried Minnaert (Tilburg University) introduced me to the basics of that text processor; Jozef Pijnenburg (Tilburg University) helped me with some more advanced features.

I received valuable comments on preliminary versions of various chapters from the following colleagues: Ebru Angűn (Galatasaray University, Istanbul), Russell Barton (Pennsylvania State), Victoria Chen (University of Texas at Arlington), Gabriella Dellino (Politecnico di Bari), Dick den Hertog (Tilburg University), Tony Giunta (Sandia), Yao Lin (Georgia Institute of Technology), Carlo Meloni (Politecnico di Bari), Barry Nelson (Northwestern), William Notz (Ohio State), Huda Abdullah Rasheed (al-Mustansiriyah University, Baghdad), Wim van Beers (Tilburg University), Willem van Groenendaal (Tilburg University), Jim Wilson (North Carolina State), and Bernard Zeigler (Arizona State).

I used a preliminary draft of this book to teach a course called "Simulation for Logistics" for the "Postgraduate International Program in Logistics Management Systems" at the Technical University Eindhoven. This helped me to improve parts of the book. Students solved the exercises 1.6, 2.13, 2.15. The names of these students are: Nicolas Avila Bruckner, Olla Gabali, Javier Gomes, Suquan Ju, Xue Li, María Eugenia Martelli, Kurtulus Öner, Anna Otáhalová, Pimara Pholnukulkit, Shanshan Wang, and Wei Zhang. I especially thank Xue Li and Shanshan Wang.

For that course, I also prepared PowerPoinT (PPT) slides that may also be downloaded in Portable Document Format (PDF) format from my web page:

http://center.uvt.nl/staff/kleijnen/simwhat.html.
My website also offers an update of this book, including corrections, new references, new exercises: visit
http://center.uvt.nl/staff/kleijnen/
and click “Publications”.

Contents

1
Introduction

This chapter is organized as follows. In Section 1.1, I define various types of simulation. In Section 1.2, I define the DASE approach. In Section 1.3, I define DASE symbols and terms. I finish with solutions for the exercises of this chapter.

1.1 What is simulation?

In this section, I give

1. a definition of “simulation models”
2. a simple example of a deterministic simulation model, namely the Net Present Value (NPV) calculation of a loan
3. two simple examples of random or stochastic simulation, namely a single-server queueing model and an inventory management model.

Definition 1.1 *A simulation model is a dynamic model that is meant to be solved by means of experimentation.*

This definition deserves some comments.

A simulation model may be a physical model—e.g., a miniature airplane in a windtunnel (other examples are automobiles and ships). I, however, ignore such physical models; i.e., I focus on *mathematical* models. These models are usually converted into *computer programs*—also called computer codes.

The term *dynamic* means that time plays an explicit and special role; i.e., the variables in the mathematical model have a time index—e.g., the variable x becomes $x(t)$ where t denotes a point in time. *Static* models, however, may also be solved through experimentation. For example, the well-known Newton-Ralphson method may be used to find the roots of an equation, and Interior Point (IP) methods may be used to find the optimum solution of a Linear Programming (LP) model (also see Chapter 4).

Closely related to simulation are *Monte Carlo* methods, which I define as methods that use PseudoRandom Numbers (PRNs). PRNs are generated by means of a computer program, so they are not really random, and yet they are assumed to be independently and uniformly distributed on the interval $[0, 1]$—so Monte Carlo methods involve chance, which explains the name. Monte Carlo methods are used to solve multiple integrals, which arise in physics, mathematical statistics, etc.

All the methods discussed in the preceding comments are *numerical* methods. In this book, I focus on dynamic, random simulation, but I also discuss deterministic simulation explicitly (see the next example). Moreover, many DASE methods also apply to static Monte Carlo methods. Also see my previous publications, starting with my 1974 book [181], pp. 3–22 and ending with my 2005 contribution [193].

Example 1.1 *Assume that the following data are given: θ the discount factor used by the decision maker; n the length of the planning period measured in years, and x_t the cash flow in year t with $(t = 0, \ldots, n)$. Then the NPV—also called the Present Value (PV)—(say) y may be computed through the following equation (engineers often use an alternative formula, assuming continuous time—so $\sum$ becomes $\int$, etc.):*

$$y = \sum_{t=0}^{n} \frac{x_t}{(1+\theta)^t}. \tag{1.1}$$

This NPV formula may be used to compare alternative cash flow patterns. For example, different patterns are caused by different loan types. One loan type may require a fixed amount paid back, at the end of each year (say) $z_t = z$ with $t = 1, \ldots, n$ and $z_0 = 0$, and interest payments determined by the interest rate c and the loan amount at the end of the year t, namely w_t:

$$x_t = -[\min(z_t, w_t) + c\, w_t] \text{ with } t = 1, \ldots, n. \tag{1.2}$$

where the loan amount is determined by

$$w_t = w_{t-1} - z_t \text{ with } t = 1, \ldots, n. \tag{1.3}$$

and

$$x_0 = w_0 \tag{1.4}$$

where w_0 is the original loan amount, so x_0 is the positive cash flow at the start of the planning period, whereas x_t with $t = 1, \ldots, n$ are negative *cash flows (I have already specified the initial condition $z_0 = 0$). Finally, the stopping conditions of the simulation run must also be given; in this example, the end of the planning period must be reached. Obviously, this is a simple deterministic dynamic model, including the first-order difference equation (1.3). It is easy to program such a model, e.g., by means of spreadsheet software such as Excel (a recent reference is [340]; also see [92] and [299]).*

Exercise 1.1 *Derive that $NPV = 6.238$ in case the original loan is $w_0 = 100$, $n = 2$, $c = 0.10$, and $\theta = 0.15$ (so the loaner expects to achieve a higher return on investment or ROI than the bank can).*

This *deterministic* simulation example may be augmented to a *random* simulation, if (for example) the discount factor θ or the cash flows x_t are unknown so their values are sampled from distribution functions. This type of simulation is called *Risk Analysis* (RA) or *Uncertainty Analysis* (UA); see [44], [313], [327], [340], and also recent textbooks such as [110] and [392]. I shall return to risk analysis in Section 4.5.

Complicated examples of deterministic simulation are provided by models of airplanes, automobiles, TV sets, chemical processes, computer chips, etc.—applied in *Computer Aided Engineering* (CAE) and *Computer Aided Design* (CAD)—at Boeing, General Motors, Philips, etc. Recent surveys are [65], [66], [254], [280], and [357]. (Note that [254] was published in the *AIAA Journal*, where *AIAA* stands for American Institute of Aeronautics and Astronautics.) In the last decade, Multidisciplinary Design Optimization (MDO) has emerged as a new discipline; see, e.g., [5].

Note: Deterministic simulation may show numerical inaccuracies, which make this type of simulation related to random simulation. The latter type, however, uses PRNs inside its model.

Another type of (primarily) deterministic simulation is *System Dynamics* (SD), originated by Forrester under the name "Industrial Dynamics"; see his 1961 textbook [113]. SD is more than a simulation method; it is a world view. A crucial concept in this view is *feedback*; i.e., compare an output with a norm, and react if there is an undesirable deviation. Feedback often generates counterintuitive behavior. Applications of SD include simulations of companies, industries (including supply chains), countries, and the whole globe (including the warming-up of the earth's atmosphere). In 2000, Sterman wrote a SD textbook with more than 1,000 pages; see [364].

In summary, these simulation types are usually deterministic, but their parameters (e.g., θ in (1.1)) or input variables (e.g., x_t in (1.2)) may be sampled from a given "prior" distribution so they become random. Mathematically, these simulation models are difference equations, which are nonlinear and possibly stochastic, so simulation is used to "solve" these equations.

In the preceding paragraph, I implicitly used the following two definitions, based on Zeigler's famous textbook; see [413] and Section 1.3 below.

Definition 1.2 *A (model) parameter has a value that must be inferred from data collected in the real world; it can not be observed directly in the real world.*

Definition 1.3 *An input variable of a model can be directly observed in the real world.*

Exercise 1.2 *Consider the following two applications involving the discount factor for a NPV calculation as in Example 1.1: (a) a student wishes to select the lowest NPV for several loan alternatives—each with the same interest rate, but with different amortization schemes; (b) a company wishes to select the highest NPV among several investment alternatives, such that the company maintains the ROI that it has realized during the last five years. Is the discount factor a parameter or a variable in (a); and in (b)?*

Opposed to the preceding (mainly deterministic) simulation types are *Discrete-Event Dynamic Systems* (DEDS) simulations (the name DEDS has been made popular by Ho and his collaborators; see [147]). This type of simulation is inherently stochastic; i.e., without randomness the problem would change completely. For example, a queueing (waiting) problem is caused by the randomness of the arrival or the service times. If these times were deterministic, the problem would become a so-called scheduling problem. A well-known building block of DEDS simulation (which I will also use repeatedly in this book) is the so-called M/M/1 model.

Definition 1.4 *An M/M/1 model is a queueing model with one server, and Markovian interarrival and service times.*

These Markovian times are "independently" exponentially distributed; i.e., the interarrival times are mutually independent, and they are independent of the service times. The exponential distribution has the memoryless property; the exponential distribution implies that the number of events (say, arrivals) has a Poisson distribution. Implicit in this M/M/1 notation is that the server's priority rule is First-In-First-Out (FIFO), the waiting room has infinite capacity, etc. An M/M/1 model may be simulated as follows.

Example 1.2 *Let a_{i+1} denote the interarrival time between customers i and $i+1$; s_i the service time of customer i; and r a PRN. Assume that the outputs of interest are w_i, the waiting time of customer i, and that this output is characterized by the average waiting time $\overline{w}$ defined as*

$$\overline{w} = \frac{\sum_{i=1}^{n} w_i}{n} \tag{1.5}$$

where n denotes the number of customers that stops the simulation run (so this example is a terminating simulation, not a steady-state simulation; in the latter case, n would not be prefixed or would be a "very large" number). Furthermore, assume that the simulation starts in the empty state (no customers in the system), so the customer who arrives first does not need to wait: $w_1 = 0$. The dynamics of the single-server system are specified by the so-called Lindley recurrence formula

$$w_{i+1} = \max(0, w_i + s_i - a_{i+1}). \tag{1.6}$$

The random input variables s and a in this equation are sampled (or generated) such that these variables have a service rate μ and an arrival rate λ (so the mean or expected service and interarrival times are $1/\mu$ and $1/\lambda$ respectively). To sample these variables s and a, simulation may use the PRN r as follows:

$$s_i = \frac{-\ln r_{2i-1}}{\mu} \tag{1.7}$$

and

$$a_{i+1} = \frac{-\ln r_{2i}}{\lambda} \tag{1.8}$$

where a single PRN stream $(r_1, r_2, \ldots, r_{2n-1}, r_{2n})$ is used (each of the n customers needs two PRNs, namely one PRN for the arrival time and one PRN for the service time). To program this simulation model, the analysts can choose from many simulation software packages; e.g., Swain's [373] seventh biennial survey of DEDS simulation software lists 48 software products. I think that the package that is most popular worldwide is Arena, which is very well documented in [174].

Exercise 1.3 *Consider the waiting time equation (1.6). Is it straightforward to derive a similar equation for the queue length?*

Exercise 1.4 *What are the advantages of using two separate PRN streams for a single-sever simulation with a given server priority rule (not necessarily FIFO), which has two input processes—namely the service and the arrival processes —compared with the single PRN stream in Eqs. (1.7) and (1.8)?*

Mathematical analysis of the M/M/1 model reveals that the fundamental input parameter is the so-called *traffic rate*—also called traffic intensity or traffic load—(say) $\rho = \lambda/\mu$ where λ and μ were defined above (1.7). In other words, the M/M/1 model has a single input parameter (namely, ρ), whereas its computer code has two parameters (λ and μ). In this book, I shall often use the M/M/1 model as an example—but I shall also need an example with multiple inputs. Therefore I now present another well-known building block for DEDS simulation models, namely the so-called (s, S) model.

Definition 1.5 *An (s, S) model (with $s < S$) is a model of an inventory management (or control) system with random demand (say) D. The inventory I is replenished whenever the inventory decreases to a value smaller than or equal to the reorder level s. The order quantity Q is*

$$Q = \begin{cases} S - I & \text{if } I \leq s \\ 0 & \text{if } I > s. \end{cases} \tag{1.9}$$

There are several *variations* on this basic model. For example, review of the inventory (I in Eq. 1.9) may be either continuous (in real time) or periodic. The lead time of the order may be either a nonnegative constant or a nonnegative random variable. Demand that exceeds the inventory at hand ($D > I$) may be either lost or backlogged. Costs may consist of inventory, ordering, and out-of-stock costs. These cost components are specific mathematical functions; e.g., inventory carrying (or holding) cost may be a constant per item unit, per time unit. In practice, out-of-stock costs are hard to quantify so a service (or fill rate) constraint may be specified instead; e.g., the total stockout quantity per (say) year should be smaller than 10% of the total sales during that same period.

Programming this inventory model is harder than programming the M/M/1 model; the latter has dynamics specified by the simple Eq. (1.6). For a thorough discussion of this programming, I refer to simulation textbooks such as Law's well-known textbook; see [227], pp. 48–61.

DEDS simulation and continuous simulation—which solves differential and difference equations numerically—may also be combined into so-called *hybrid* simulation. This type of simulation is also discussed by textbooks on DEDS simulation; also see the 2006 paper [125].

In summary, simulation is a widely used methodology that is applied in many disciplines. It provides a flexible, powerful, and intuitive tool for the analysis of complicated processes. The resulting insight may be used to design better systems.

Much more could be said about simulation. There are many more textbooks besides the ones I mentioned above; e.g., other well-known (recently updated) textbooks on DEDS simulation are [22] and [293] ([293] also discusses SD). For the most recent publications on DEDS simulation, I recommend the yearly proceedings of the *Winter Simulation Conference*; see its web page

http://www.wintersim.org/.

The INFORMS top journals *Management Science* and *Operations Research* publish fundamental articles on DEDS simulation; see [271] and

http://www.informs.org/.

Many other journals on Management Science/Operations Research (MS/OR) also publish on simulation—both DEDS and other types of simulation. I also refer to the recent reviews in [146] and [315]. Sensitivity analysis of simulation models is the focus of the Sensitivity Analysis of Model Output

(SAMO) conferences that have taken place every three years, since 1995; see

http://samo2007.chem.elte.hu/.

Exercise 1.5 *Do entertainment games (such as America's Army; see [373]), business games (such as the beer game in SD; see [354]) and gaming models (using concepts such as the Nash equilibrium discussed in [351]) meet the definition of "simulation"?*

1.2 What is DASE?

This book is about DASE, which is my acronym (introduced in the Preface) for the Design and Analysis of Simulation Experiments. These terms require definitions—especially because simulation is a method applied in many different scientific fields, which have their own terminologies.

Simulation implies that the analysts do not solve their model by mathematical calculus; instead, they try different values for the inputs and parameters of their model in order to learn what happens to the model's output. For example, in the NPV example (Example 1.1) the analysts may experiment with different values for the parameter θ (discount factor) and the input variable z (amount paid back every year); see again Eqs. (1.1) and (1.2). In the M/M/1 simulation, the analysts may experiment with different traffic rates and priority rules (replacing the implicit FIFO rule). In the (s, S) inventory simulation, they may try different combinations of the control limits s and S.

The *goals* of such a numerical experiment are (see [280] and also [31])

- Verification and Validation (V & V)
- Sensitivity Analysis—either global or local—or "What If" analysis
- Optimization
- Risk Analysis.

These goals require that the simulation analysts pay attention to the *design* of their experiments. For example, if the experimenters keep an input of the simulation model constant, then they cannot estimate the effect of that input on the output. In practice, however, many analysts keep many inputs constant, and experiment with a few factors only. In Chapter 6 (on Screening), I shall show that there are better ways to run simulation experiments with many factors. Another example of inferior practice is changing only one input at a time (while keeping all other inputs fixed at their so-called base values). In the next chapter, I shall prove that this approach is inefficient and does not enable the estimation of interactions among inputs.

The *design* of the experiment is intimately related to its *analysis*; indeed, I consider it a chicken-and-egg problem. An example is provided by analysts assuming that the input has a "linear" effect on the output; i.e., they assume a first-order polynomial approximation (remember the Taylor series in mathematics) or main effects only (mathematical statistics terminology). Given this assumption, it suffices to experiment with only two values of that input. Moreover, the analysts may assume that there are (say) $k > 1$ inputs that have main effects only. Then their design requires a relatively small experiment (of order k). For example, changing only one input at a time does give unbiased estimators of all the main effects. The next chapter, however, will show that minimizing the variances of these estimators requires a different design—with approximately the same size of the experiment as the one required by the one-factor-at-a-time design.

Such a polynomial approximation may be called a metamodel (see my 1975 article [182]).

Definition 1.6 *A metamodel is an approximation of the Input/Output (I/O) function that is defined by the underlying simulation model.*

Metamodels are also called response surfaces, surrogates, emulators, auxiliary models, repromodels, etc. There are different *types* of metamodels. The most popular type is a polynomial of first or second order (degree), which I shall discuss in Chapters 2 and 3. In deterministic simulation, another metamodel type is popular, namely Kriging (also called spatial correlation) models, discussed in Chapter 5. Less popular are (in alphabetical order): Classification And Regression Trees (CART), Generalized Linear Models (GLM), Multivariate Adaptive Regression Splines (MARS), (artificial) neural networks, nonlinear regression models, nonparametric regression analysis, radial functions, rational functions, splines, support vector regression, symbolic regression, wavelets, etc.; for details I refer to [16], [19], [25], [65], [66], [80], [87], [105], [129], [145], [156], [166], [237], [252], [312], [334], [339], [357], [355], [365], [370], and [390]; also see Sandia's Surfpack software

http://endo.sandia.gov/Surfpack

and the European Commission's Joint Research Centre (JRC) SIMLAB software

http://simlab.jrc.cec.eu.int/.

In theory, the analysts may combine several types of metamodels, weighing each type with its estimated accuracy. In practice, such a combination is rare, because the analysts are familiar with one or two types only. Combining different metamodel types is further discussed in [133].

The term "response surface" is used for *local* metamodels in *Response Surface Methodology* (RSM) and for *global* metamodels in deterministic simulation. *Local* means that only a small subarea of the total experimental area is considered. The limit of this "small" subarea is an area with a size that goes to zero, so partial derivatives are considered. These derivatives are the components of the gradient (gradients will be discussed further on in this chapter and in Chapter 4). RSM was introduced in 1951 by Box and Wilson (see [51]) as an iterative heuristic for optimizing real (physical) systems, namely chemical systems (a recent textbook is [268]). I shall discuss RSM for the optimization of simulated systems in Chapter 4. The oldest references to the term "response surface" in deterministic simulation that I could find quickly, are a 1985 American article ([97]) and a 1984 European monograph ([284]); more recent references—including additional references—are [25], [168], [314], and [327].

DASE has *strategic* and *tactical* aspects. Traditionally, researchers in discrete-event simulation have focused on *tactical* issues, such as the runlength of a steady-state simulation, the number of runs of a terminating simulation, and Variance Reduction Techniques (VRTs); see the classic 1963 article by Conway [83] and the current literature mentioned above (at the end of the preceding section, especially Nelson's review article [271]). In deterministic simulation—where these tactical issues vanish— statisticians have been attracted to *strategic* issues, namely which factor combinations (scenarios) to simulate and how to analyze the resulting data; see the classic 1996 publication by Koehler and Owen [222] and the 2003 textbook by Santner, Williams, and Notz [333]. Few statisticians have studied random simulations. Only some simulation analysts have focused on strategic issues. I will focus on strategic issues; I will discuss only those tactical issues that are closely related to strategic issues.

The statistical theory on *Design Of Experiments* (DOE, also spelled DoE) was developed for real, nonsimulated experiments in agriculture in the 1920s ([65] refers to a 1926 publication by Fisher), and in engineering, psychology, etc. since the 1950s. In real experiments it is impractical to investigate *many* factors; ten factors seems a maximum. Moreover, it is then hard to experiment with factors that have more than *a few* values; five values per factor seems the limit. In simulated experiments, however, these restrictions do not apply. Indeed, computer codes may have hundreds of inputs and parameters—each with many values. Consequently, a multitude of scenarios may be simulated. Moreover, simulation is well-suited to *sequential* designs instead of "one shot" designs, because simulation experiments are run on computers that typically produce output sequentially (apart from parallel computers, which are rarely used in practice) whereas agricultural experiments are run during a single growing season. So a *change of mindset* of simulation experimenters is necessary. For a more detailed discussion of simulated versus real experiments I refer to a 2005 survey article that I coauthored, [210].

In summary, DASE is needed to improve the efficiency and effectiveness of simulation; i.e., DASE is crucial in the overall process of simulation.

1.3 DASE symbols and terms

I must define some symbols and terms because DASE is a combination of mathematical statistics and linear algebra that is applied to experiments with deterministic and random simulation models; these models are applied in many scientific fields—ranging from sociology to astronomy. An excellent survey of this spectrum of simulation applications is Karplus's classic 1983 paper [173].

I had a problem when deciding on the *notation* in this book. Mathematicians use capital letters to denote matrices, whereas statisticians use capitals to denote random variables. To be consistent, I would have to denote the error term in a regression model by (say) E and the matrix of explanatory variables by $\mathbf{x}$. I have indeed used that notation in [191], but now I feel that such a notation is too orthodox. So I follow most other authors in simulation and regression analysis; i.e., I do not always use capitals for random variables; the readers should infer from the context whether a variable is random or not. I do use bold letters to denote matrices and vectors. Whenever I think that readers may be misled, I explicitly discuss the randomness of a particular variable. For example, in Chapter 3, I shall discuss Generalized Least Squares (GLS) using the covariance matrix of the simulation responses—which is estimated in practice; this estimated matrix creates statistical problems—which needs explicit discussion.

Furthermore, I use a "big hat" (instead of a "small" hat) when needed; e.g., in (3.14a) some indices are under that "hat", whereas some are not. I also use a "big bar" when I think it is needed; (2.27) gives an example

I use Greek letters to denote *parameters*, which are model quantities that have values that cannot be directly observed in the real world so these values must be inferred from other real data; see Definition 1.2. For example, the service rate μ in the M/M/1 model is estimated from the (say) n observations on the service time s (a classic estimate is $\widehat{\mu} = 1/\overline{s}$ with $\overline{s} = \sum_{i=1}^{n} s_i/n$). Note that an estimator (e.g., the sample average) is a random variable, which has a specific value—once it has been computed; this value is called an estimate.

Unlike a parameter, a variable can be directly observed in the real world. For example, the input variable service time s can be measured in a straightforward way (s is the realization of the random variable S). A variable may be either an input or an output of a model. For example, besides the input s, the M/M/1 model may have the output w, waiting time.

Both parameters and input variables may be changed in a simulation experiment; i.e., they have at least two *values* or *levels* in the experiment. Parameters and input variables together are called *factors,* in DOE. For example, a simple design in DOE is a 2^k factorial experiment; i.e., there are k factors, each with two levels; all their combinations are simulated.

These combinations are often called *scenarios* in simulation and modeling. Scenarios are usually called *design points* or *runs* by statisticians. I reserve the term "run" for a *simulation run*, which starts the simulation program in the initial conditions (e.g., the empty state in an M/M/1 simulation) and stops the simulation program once a specific event occurs (e.g., n customers have been simulated; see the discussion below Eq. 1.5).

Factors (inputs) and responses (outputs) may be either *qualitative* or *quantitative.* In the M/M/1 example, quantitative factors are the arrival and service rates; the traffic rate is the fundamental quantitative factor. In a single-server queueing simulation, a qualitative factor may be the priority rule—which may have (say) three levels, namely FIFO, LIFO (Last-In-First-Out), or SPT (Shortest-Processing Time first).

Simulation inputs and outputs may be measured on five types of *scales*:

1. Nominal: This is the only scale that applies to a qualitative (or categorical) factor. One example was the priority rule with its three nominal values (FIFO, LIFO, SPT). Another example is a simulation with two types of customers, namely A (emergencies) and B (regular). Interpolation or extrapolation makes no sense (so regression analysis must be applied with care, as I shall show in Chapter 2).

2. Ordinal: This scale ranks the input or output. For example, this scale sorts (say) n observed output values from lowest to highest, and assigns them ranks from 1 to n. *Order statistics* uses such a scale; see Conover's excellent textbook on nonparametric (distribution-free) statistics [81] (I shall use order statistics in later chapters). Another example is a survey that assigns ranks from 1 to 5 in order to measure how strongly the respondent agrees with a statement (completely agree, agree, neutral, disagree, strongly disagree).

3. Interval: This scale assigns numbers that are unique except for a linear transformation; i.e., this scale has an arbitrary zero point. An example is temperature measured in Celsius or Fahrenheit degrees. Analysts should prefer mathematical and statistical methods that are not sensitive to the scale that is used to quantify inputs or outputs. In Section 4.2, I shall discuss a scale-independent alternative for the steepest ascent method; the latter method is standard in RSM.

4. Ratio: This scale has a unique zero, so "$2x$" means "twice as much as x". Examples are length measured in centimeters or inches, and cash flow measured in euros or US dollars. Other examples are the arrival and the service rates, which depend on the time unit (e.g., seconds). Like the interval scale, the ratio scale should not change "the" conclusions of mathematical and statistical analyses.

5. Absolute: No transformation applies. An example is the number of customers arriving during the simulation run of an M/M/1 model; this is a discrete (not a continuous) variable.

A more detailed discussion of types of variables and measurement scales is given in my 1987 book; ([184], pp. 135–142).

Exercise 1.6 *Because "simulation" involves experimenting with a computer model, you are asked to program the M/M/1 defined in Example 1.2, using any software you like (e.g., Arena, C++, Pascal). Select "the" performance measure; e.g., average waiting time. Next you should experiment with your simulation model; here are some suggestions:*

1. *Change the run length (symbol n in Example 1.2) from (say) $n = 10$ (terminating simulation) to n large enough to reach the steady state; try these two n values for a "low" and a "high" traffic rate. Runs "several" replicates; e.g., $m = 10$ replicates. Ensure that the replicates are Identically and Independently Distributed (IID); i.e., use nonoverlapping PRN streams. Use either a single PRN stream for service and arrival times or use two separate streams for the arrival and service times respectively. Compare your simulation estimate with the analytical steady-state mean; use graphical plots and mathematical statistics.*

2. *Change the traffic load ($\rho = \lambda/\mu$) to estimate the I/O function. Apply either the same or different PRN seeds when comparing traffic loads: do Common Random Numbers (CRN) give better results?*

3. *Replace the exponential service time distribution by a different distribution (e.g., an Erlang distribution, namely the sum of two exponential distributions, each with a mean equal to half the original mean, to keep the traffic load constant when changing the distribution). Select some fixed value for the traffic rate, the number of customers per run, and the number of replicated runs respectively; e.g., select one of the values used above. Does the switch from an exponential to an Erlang distribution change the selected performance measure significantly?*

1.4 Solutions for exercises

	t	*payback*	*interest*	*NPV*
	0	*100*	*0*	*100*
Solution 1.1	*1*	*-50*	*-10*	$\frac{-(50+10)}{1+0.15} = -52.174$
	2	*-50*	*-5*	$\frac{-(50+5)}{(1+0.15)^2} = -41.588$
				$100 - 52.174 - 41.588 = 6.238$

Solution 1.2 *(a) For the student the discount factor is a variable, quoted by the bank; (b) for the company it is a parameter to be estimated from its investments during the last five years.*

Solution 1.3 *No.*

Solution 1.4 *Separate PRN streams improve the performance of two well-known Variance Reduction Techniques (VRTs), namely Common Random Numbers (CRN) and Antithetic Random Numbers (ARN); see any textbook on DEDS simulation.*

Solution 1.5 *Games such as entertainment games and the beer game are simulation models; gaming models are solved analytically so they are not simulation models.*

Solution 1.6 *Many answers are possible; compare your results with the results that you will obtain, once you will have read the next chapter(s).*

2

Low-order polynomial regression metamodels and their designs: basics

This long chapter is organized as follows. In Section 2.1, I discuss black-box versus white-box approaches in DASE. In Section 2.2, I cover the basics of linear regression analysis. In Section 2.3, I focus on first-order polynomial regression. In Section 2.4, I present designs for such first-order polynomials, namely so-called resolution-III designs. In Section 2.5, I augment the first-order polynomial regression model with interactions (cross-products) among the factors. In Section 2.6, I discuss resolution-IV designs, which give unbiased estimators of the main effects—even if there are two-factor interactions. In Section 2.7, I present resolution-V designs, which also estimate these individual two-factor interactions. In Section 2.8, I extend the regression model to second-order polynomials. In Section 2.9, I present designs for these second-degree polynomials, focussing on Central Composite Designs (CCDs). In Section 2.10, I briefly examine "optimal" designs and other designs. In Section 2.11, I discuss validation of the estimated regression model, including the coefficient of determination R^2 and the adjusted coefficient $R^2_{adjusted}$, Pearson's and Spearman's correlation coefficients, and cross-validation. In Section 2.12, I summarize more simulation applications of linear regression metamodeling. In Section 2.13, I summarize my conclusions. I finish with an appendix and solutions for the exercises in this chapter.

2.1 Introduction

In Chapter 1, I introduced the statistical theory on Design Of Experiments (DOE) and design of simulation experiments (DASE). That theory views the simulation model as a black box—not as a white box.

Definition 2.1 *A black-box view implies that the simulation model transforms observable inputs into observable outputs, whereas the values of internal variables and specific functions implied by the simulation's computer modules are unobservable.*

To explain the difference between black-box and white-box approaches, I now return to the M/M/1 example in Chapter 1. A *white-box* view was presented in (1.5) through (1.8); for convenience, I reproduce those equations—replacing the symbol n by c because n is a reserved symbol for another quantity in this chapter:

$$\overline{w} = \frac{\sum_{i=1}^{c} w_i}{c} \tag{2.1}$$

where $\overline{w}$ denotes the average waiting time, w_i the waiting time of customer i, and c the number of customers that terminates the simulation run.

Note: An alternative output may be the estimated 90% quantile (also called percentile) of the waiting times, denoted by $w_{(\lceil .90c+0.5 \rceil)}$ where $w_{(i)}$ denotes the order statistics—so $w_{(1)} \leq \ldots \leq w_{(i)} \leq \ldots \leq w_{(c)}$—and $\lceil 0.90c + 0.5 \rceil$ means that $0.90c$ is rounded to the next integer (recent articles on estimating quantiles in simulation are [21] and [63]). Another alternative output may be the estimated variance of the waiting time in the steady state, denoted by $\widehat{var}(w)$ or $s^2(w)$—not to be confused with $\widehat{var}(\overline{w})$, which quantifies the accuracy of the estimator defined in 2.1 (a recent article on estimating $\widehat{var}(w)$ is [64]).

The dynamics of any single-server queueing simulation with First-In-First-Out (FIFO) queueing discipline is specified by the so-called Lindley recurrence formula:

$$w_{i+1} = \max(0, w_i + s_i - a_{i+1}). \tag{2.2}$$

where a_{i+1} denotes the interarrival time between customers i and $i+1$, and s_i denotes the service time of customer i. Suppose, the simulation starts in the empty state, so $w_1 = 0$.

An M/M/1 simulation model samples the random input variables s and a such that these variables have exponential (Markovian, symbol M) service and interarrival times. A possible implementation of such an M/M/1 simulation—with a service rate μ and an arrival rate λ−using the inverse distribution function transformation of a single stream of PseudoRandom Numbers (PRNs) r with seed (initial PRN) r_0 is

$$s_i = \frac{-\ln r_{2i-1}}{\mu} \tag{2.3}$$

and

$$a_{i+1} = \frac{-\ln r_{2i}}{\lambda}. \tag{2.4}$$

Such a white-box representation is used by *Perturbation Analysis* (PA) and the *Score Function* (SF) or likelihood ratio (LR) method (to estimate the gradient for local—not global—sensitivity analysis and for optimization using infinitesimal perturbations). PA and SF are discussed in (for example) Spall's recent textbook [360]; also see Rubinstein and Shapiro's classic SF book, [319], and Ho and Cao's classic PA book, [147]. Recent reviews are [117] and [377]. (I will return to the estimation of the gradient further on in this chapter and in Chapter 4.)

In DASE, however, I do not follow a white-box approach; instead, I use a *black-box* approach. Such a black-box approach is also used by DOE for real-world experiments (see, e.g., [268]) and by Design and Analysis of Computer Experiments (DACE) for deterministic simulation experiments (see, e.g., [333]).

I now give a black-box representation of any *single-server* simulation model with output $\overline{w}$ (average waiting time) and inputs λ and μ (arrival and service rates) and r_0 (PRN seed)—and a fixed queueing discipline (e.g., FIFO), a fixed waiting room capacity, etc.:

$$\overline{w} = w(\lambda, \mu, r_0) \tag{2.5}$$

where $w(.)$ denotes the mathematical function implicitly defined by the computer program that implements (2.1) through (2.4).

My *general* black-box representation is

$$\widehat{\mathbf{\Theta}} = s(d_1, \ldots, d_k, \mathbf{r}_0) \tag{2.6}$$

where

$\widehat{\mathbf{\Theta}}$ is the vector of simulation outputs (see the next paragraph);

$s(.)$ denotes the mathematical function that is implicitly defined by the simulation program (computer code) that implements the given simulation model;

d_j with $j = 1, \ldots k$ is the j^{th} input variable (factor) of the simulation program (so $\mathbf{D} = (d_{ij})$ is the design matrix for the simulation experiment, with $i = 1, \ldots, n$ and n the number of factor combinations in that experiment),

$\mathbf{r}_0$ is the vector of PRN seeds ($\mathbf{r}_0$ is a vector because each simulation process may have its own PRN seed to improve the effectiveness of Variance Reduction Techniques (VRTs); see [227], p. 588).

I point out that d in (2.6) determines the original input variable z and the corresponding standardized input variable x; see (2.33) below. The design matrix $\mathbf{D}$ is usually standardized; e.g., a two-level (fractional) factorial has elements that are either -1 or $+1$.

The simulation output $\widehat{\mathbf{\Theta}}$ is a *multivariate* random variable that is meant to estimate $\mathbf{\Theta}$, which denotes the interesting characteristics of the output distribution (obviously, $\mathbf{\Theta}$ is not random). One example is that $\widehat{\Theta}_1 = \overline{w}$ estimates the mean of the output distribution, and $\widehat{\Theta}_2 = w_{(\lceil 0.90c+0.5 \rceil)}$ estimates the 90% quantile of that same distribution. Another example, is that $\widehat{\Theta}_1 = \overline{w}$ estimates the mean waiting time, and $\widehat{\Theta}_2 = \overline{v}$ estimates the mean queue length. A complicated case study is a nuclear waste simulation model with 13 output variables; see the recent survey in [145].

One possible metamodel of the black-box model in (2.5) is a *first-order polynomial* in the two input variables λ and μ augmented with the noise e:

$$y = \beta_0 + \beta_1 \lambda + \beta_2 \mu + e \tag{2.7}$$

where

y is the metamodel predictor of the simulation output $\overline{w}$ in (2.5);

β_0, β_1,and β_2 are the parameters of this metamodel—which may be collected in the vector $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2)'$;

e is the residual or noise—which includes both *lack of fit* of the metamodel (this metamodel is a Taylor series approximation cut off after the first-order effects) and *intrinsic noise* (caused by the PRNs).

Besides (2.7), there are many alternative metamodels. For example, a simpler metamodel is

$$y = \beta_0 + \beta_1 x + e \tag{2.8}$$

where x is the traffic rate—in queueing theory usually denoted by ρ (statisticians often use this symbol to denote a correlation coefficient; in this book, the context should clarify what the symbol ρ means):

$$x = \rho = \frac{\lambda}{\mu}. \tag{2.9}$$

This combination of the two original factors λ and μ into a single factor ρ (inspired by queueing theory) illustrates the use of *transformations.* Another useful transformation may be a logarithmic one: replacing y, λ, and μ by $\log(y)$, $\log(\lambda)$, and $\log(\mu)$ in (2.7) makes the first-order polynomial approximate relative changes; i.e., the regression parameters $\boldsymbol{\beta}$ become elasticity coefficients.

Definition 2.2 *The elasticity coefficient of (say) y with respect to x is $(\partial y/y)/(\partial x/x)$.*

Exercise 2.1 *Prove that $(\partial y/y)/(\partial \lambda/\lambda) = \beta_1$ if y is replaced by $\log(y)$ and λ by $\log(\lambda)$ in (2.7).*

Elasticity coefficients are popular in econometrics, but also in other disciplines. In an econometric case study, Van Schaik and I used the logarithmic transformation for some—but not all—explanatory variables; see

[389]; in this study we use data that are obtained through passive observation instead of active simulation experimentation. An analytical (not a simulation) software engineering study that also uses *relative* changes in the explanatory variables is [59].

The use of transformations illustrates that simulation analysts should be guided by knowledge of the real system and corresponding analytical models.

2.2 Linear regression analysis: basics

It is convenient to use the following general matrix representation for a linear regression model with multiple inputs and a single output (in case of multiple outputs, it would be necessary to use multivariate regression, which is discussed in the next chapter; the univariate regression model may be applied to each individual output):

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e} \tag{2.10}$$

where

$\mathbf{y} = (y_1, \ldots, y_n)'$ denotes the n-dimensional vector with the regression predictor (or dependent variable) y with n the number of simulation runs (or observations);

$\mathbf{X} = (\mathbf{x}_{ij})$ denotes the $n \times q$ matrix of explanatory (independent) regression variables with $\mathbf{x}_{ij}$ the value of explanatory variable j in run i $(i = 1, \ldots, n; j = 1, \ldots, q)$ (e.g., in (2.7) $q = 3$ and in (2.8) $q = 2$ including the *dummy* variable or constant $x_{i0} = 1$ corresponding with β_0);

$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)'$ denotes the q regression parameters—including the effect of a possible dummy variable (if there is such a dummy variable, then β_1 denotes the intercept in the general regression model, whereas the symbol β_0 denoted the intercept in the specific regression model (2.8));

$\mathbf{e} = (e_1, \ldots, e_n)'$ denotes the residuals in the n runs.

To select specific values (say) $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_q)'$ for the regression parameters, the criterion of *Least Squares* (LS)—also called the ordinary LS (I shall discuss generalized LS in the next chapter)—is often used; i.e., $\widehat{\boldsymbol{\beta}}$ is selected such that it minimizes the *Sum of Squared Residuals*, *SSR*:

$$\min_{\widehat{\boldsymbol{\beta}}} SSR = \sum_{i=1}^{n} (\widehat{e_i})^2 = \sum_{i=1}^{n} (\widehat{y_i} - w_i)^2 = (\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w}) \tag{2.11}$$

where $\widehat{e_i} = \widehat{y_i} - w_i$ is the estimated residual for input combination i,

$$\widehat{y_i} = \sum_{j=1}^{q} x_{ij} \widehat{\beta_j} = \mathbf{x}_i' \widehat{\boldsymbol{\beta}}, \tag{2.12}$$

and w_i denotes the simulation output of run i (e.g., the average waiting time of that run; see (2.5)).

The solution of (2.11) gives the LS estimate $\widehat{\boldsymbol{\beta}}$ of the regression parameter vector $\boldsymbol{\beta}$ in the regression model (2.10); it can be derived to be

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}. \tag{2.13}$$

Obviously, this LS estimate exists only if the matrix $\mathbf{X}$ is not collinear, so the related inverse $(\mathbf{X}'\mathbf{X})^{-1}$ does exist or this inverse is stable in numerical computations. For example, $\mathbf{X}$ is collinear in (2.7) if the two inputs λ and μ change simultaneously by the same amount; $\mathbf{X}$ is collinear in (2.9) if the input ρ is kept constant. The selection of a "good" $\mathbf{X}$ in (2.10)—and hence in (2.13)—is the focus of the next sections on various designs.

The computation of the LS estimate $\widehat{\boldsymbol{\beta}}$ does not need to use (2.13); i.e., better numerical accuracy may result when solving the set of *normal equations*

$$\mathbf{X}'\mathbf{y} = \mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}},$$

which follows from (2.10). However, the next sections provide such good design matrixes that the computation of the LS estimates becomes trivial and numerical problems are negligible.

I emphasize that the LS criterion, which is used in (2.11), is a mathematical (not a statistical) criterion. This criterion is also known as the L_2 norm (other popular mathematical criteria are the L_1 and the L_∞ norms; also see [269]). However, adding statistical assumptions about the simulation I/O data implies that the LS estimator has interesting statistical properties. Therefore I now introduce the following definition, where σ_u^2 denotes the variance of the random variable u.

Definition 2.3 *White noise (say) u is Normally, Independently, and Identically Distributed (NIID) with zero mean: $u \sim NIID(0, \sigma_u^2)$.*

This definition deserves some comments:

- There seems to be no standard definition of white noise; i.e., some publications (e.g., [191]) do not require normality.

- The simulation output w may indeed be *normally* (or Gaussian) distributed if this output is an average computed from a long enough time series of individual simulation data. These individual data are autocorrelated (serially correlated), so the classic Central Limit Theorem (CLT) does not apply. Yet it can be proven that—under specific conditions—this average tends to be Gaussian distributed. A counterexample is a simulation with the estimated quantile $w_{(\lceil 0.90c+0.5 \rceil)}$ as the output. For such a quantile, I do not expect normality—unless the simulation run c is very long. Also see Section 3.3.1.

- The simulation outputs w_i and $w_{i'}$ with $i \neq i'$ are indeed *independent* if they use PRN streams that do not overlap. So Common Random Numbers (CRN) violate this assumption (see Chapter 3).

- "*Identically* distributed" implies a constant variance (denoted by σ_u^2). However, I expect that the simulation outputs do not have the same variance when the input combinations change; i.e., the variances are heterogeneous or heteroskedastic instead of homogeneous or homoskedastic (the literature also uses the alternative spellings heteroscedastic and homoscedastic). For example, for the M/M/1 it is well-known that the variance increases as the traffic rate increases (actually, the variance increases much more than the mean). Therefore I will return to this practical problem (see Chapter 3).

For the time being, I assume that the simulation outputs w are indeed normally and independently distributed with the same variance (say, σ_w^2); obviously, the simulation outputs may have different means. I further assume that the linear regression model (2.10) is a valid metamodel, defined as follows.

Definition 2.4 *A metamodel is valid if and only if its residuals have zero means:* $E(e) = 0$*.*

Furthermore, I introduce the following related definition.

Definition 2.5 *A metamodel fits perfectly if and only if all its estimated residuals are zero:* $\forall i : \widehat{e}_i = 0$ $(i = 1, \ldots n)$*.*

These two definitions also deserve some comments:

- The metamodel is *biased* if $E(e) \neq 0$; i.e., the metamodel may either overestimate or underestimate the expected simulation output.
- A *perfectly* fitting metamodel indicates that n (number of simulation runs) is too small. (Also see the discussion of the special case $R^2 = 1$ in Section 2.11.1 below.)

If the residuals are white noise, then LS gives the *Best Linear Unbiased Estimator* (BLUE) (the condition is not "if and only if"; see the Gauss-Markov theorem; a recent discussion is given in [378]). The LS estimator is indeed a *linear* transformation of the simulation response $\mathbf{w}$:

$$\hat{\boldsymbol{\beta}} = \mathbf{L}\mathbf{w} \tag{2.14}$$

where $\mathbf{L}$ is not random—since $\mathbf{L} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ in (2.13)—and $\mathbf{w}$ is random. Such a linear estimator has the following two properties:

$$E(\hat{\boldsymbol{\beta}}) \ = \mathbf{L}[E(\mathbf{w})] \tag{2.15}$$

and

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) \ = \mathbf{L}[\mathbf{cov}(\mathbf{w})]\mathbf{L}'. \tag{2.16}$$

Exercise 2.2 *Prove that the LS estimator $\hat{\boldsymbol{\beta}}$ defined in (2.14) is an unbiased estimator of the true value $\boldsymbol{\beta}$ if $E(e) = 0$.*

The property in (2.16) implies that in case of white noise the LS estimator has the following (symmetric and positive semidefinite) covariance matrix (the noise does not need to be normally distributed):

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) \ = (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2. \tag{2.17}$$

Exercise 2.3 *Prove that the LS estimator $\hat{\boldsymbol{\beta}}$ defined in (2.14) has the covariance matrix (2.17) in case of white noise. (Hint: $(\mathbf{X}'\mathbf{X})^{-1}$ is symmetric.)*

Exercise 2.4 *Prove that the variance of the average waiting time of a simulation run with c customers—defined in (2.1)—would be σ^2/c if the individual waiting times were Identically and Independently Distributed (IID) with variance σ^2 (actually, these waiting times have different variances and are autocorrelated).*

It can be proven that among all linear unbiased estimators, the LS estimator is *best*, i.e., this estimator has the minimum variance—still assuming white noise. Obviously, the variances of the individual regression estimators $\widehat{\beta}_j$ are given by the main diagonal elements of (2.17); their covariances are given by the off-diagonal elements of the (symmetric) matrix. The matrix $(\mathbf{X}'\mathbf{X})$ is called the *information matrix.*

Note: Instead of deriving an unbiased estimator, some statisticians minimize the Mean Squared Error (MSE)—accepting possible bias. For example, in my 1987 book [184], p. 160 I discussed ridge regression. Because I do not know any applications of such an approach in simulation, I do not further discuss this issue. Instead, I refer to the literature; e.g., [52].

The *linear* LS estimator $\widehat{\boldsymbol{\beta}}$ has another interesting property if the simulation outputs $\mathbf{w}$ are *normally* distributed: this estimator $\widehat{\boldsymbol{\beta}}$ is then also normally distributed. Combining this property with the mean following from (2.15) and the covariance given by (2.17) gives

$$\widehat{\boldsymbol{\beta}} \sim N[\boldsymbol{\beta}, (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2]. \tag{2.18}$$

Consequently, the individual estimated regression parameters $\widehat{\beta_j}$ may be tested through the following t statistic:

$$t_{n-q} = \frac{\widehat{\beta_j} - \beta_j}{s(\widehat{\beta_j})} \text{ with } j = 1, \ldots, q \tag{2.19}$$

where $s(\widehat{\beta_j})$ is the square root of the j^{th} element on the main diagonal of the covariance matrix for $\widehat{\boldsymbol{\beta}}$ given in (2.17) with σ_w^2 estimated through the

Mean Squared Residuals (MSR):

$$MSR = \frac{SSR}{n-q} = \frac{(\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w})}{n-q} \tag{2.20}$$

where SSR was given in (2.11). The MSR in (2.20) assumes that *degrees of freedom* are left over after fitting the regression (meta)model: $n > q$. (An alternative estimator of the simulation output's variance uses replicates instead of the regression model's residuals; see (2.26)).

The t statistic in (2.19) may be used to test whether a specific regression parameter is zero:

$$H_0 : \beta_j = 0. \tag{2.21}$$

For example, the effect of the arrival rate may be hypothesized to be zero: $\beta_1 = 0$ in the first-order polynomial (2.7). This null-hypothesis is rejected if the computed t value is *significant*: $|t_{n-q}| > t_{n-q;1-\alpha/2}$ where $t_{n-q;1-\alpha/2}$ denotes the $1 - \alpha/2$ quantile of the distribution of t_{n-q} (this $t_{n-q;1-\alpha/2}$ is also called the upper $\alpha/2$ critical point of the t distribution).

Besides testing a single parameter, the analysts may hypothesize that *several* parameters have specific values; e.g., the effects of both the arrival rate and the service rate may be hypothesized to be zero: $\beta_1 = 0$ and $\beta_2 = 0$ in (2.7). More generally,

$$H_0 : \beta_{j'} = \ldots = \beta_q = 0 \tag{2.22}$$

where I rearranged the q parameters such that the last $q - j' + 1$ parameters are hypothesized to be zero (which holds in the immediately preceding example). To test this *composite* hypothesis, the following F statistic can be used (see, e.g., Searle's general regression textbook [345]).

1. Compute the SSR without the null-hypothesis; this is called the SSR of the *full* or *unrestricted* regression model: SSR_{full}.

2. Compute the SSR under the null-hypothesis, called the SSR of the *reduced* or *restricted* regression model: $SSR_{reduced}$. (Obviously $SSR_{reduced} \geq SSR_{full}$ because imposing the constraint (2.22) increases the minimum value of SSR.)

3. Compute
$$F_{q-j'+1;n-q} = \frac{SSR_{reduced} - SSR_{full}}{SSR_{full}}. \tag{2.23}$$

The composite null-hypothesis is rejected if $F_{q-j'+1;n-q}$ exceeds the $1 - \alpha$ quantile of the $F_{q-j'+1;n-q}$ distribution; that quantile is denoted by $F_{q-j'+1;n-q;1-\alpha}$.

The preceding linear regression formulas apply to I/O data obtained through

1. *passive* observation of a real system
2. *active* experimentation with either a real system or a simulation model of a real system.

The following formulas, however, apply only if the data are obtained through controlled experimentation; i.e., at least one combination of the explanatory variables $\mathbf{x}_i = (x_{i1}, \ldots, x_{iq})'$ in (2.10) is observed more than once. (In passive observation, the explanatory variables are not controlled, so they are actually random and the probability of multiple realizations of the same combination $\mathbf{x}_i$ is negligible.)

Definition 2.6 *A replicate (or replication) means that a given combination of the explanatory variables* $\mathbf{x}_i = (x_{i1}, \ldots, x_{iq})'$ *is observed (say)* $m_i > 1$ *times* $(i = 1, \ldots n)$.

The classic assumption is that these replicates are IID. In Discrete-Event Dynamic Simulation (DEDS), this assumption implies that the replicates use PRN streams that do not overlap. If the output is the response of a steady-state (nonterminating) simulation, then IID implies that the subrun outputs have negligible autocorrelation. If the subruns are actually renewal (or regenerative) cycles, then the IID assumption is satisfied by definition. For details on this IID property, I refer to any textbook on DEDS simulation.

Replication implies that at least one input combination $\mathbf{x}_i$ is repeated in the matrix of explanatory variables, $\mathbf{X}$. For example, if the first combination of λ and μ in (2.7) is replicated three times ($m_1 = 3$) and these values are 0.5 and 1.0 respectively, then (say) the first four rows of $\mathbf{X}$ are

$$\begin{bmatrix} 1 & 0.5 & 1.0 \\ 1 & 0.5 & 1.0 \\ 1 & 0.5 & 1.0 \\ 1 & ... & ... \end{bmatrix}.$$

In general, in case of replication the number of rows of $\mathbf{X}$ increases from n to (say)

$$N = \sum_{i=1}^{n} m_i \tag{2.24}$$

with m_i *identical* rows (because the same scenario is simulated m_i times). Consequently, the MSR has more degrees of freedom, namely, $N - q$ instead of $n - q$, as I explain now.

But first I point out that it is possible to keep the number of rows in $\mathbf{X}$ limited to the n *different* combinations. The output of the i^{th}combination then becomes the output averaged over the m_i replicates (also see (2.27)). I distinguish two situations:

- The number of replicates is constant over the n factor combinations ($m_i = m$). The LS estimate may then be computed from the n

averages,$\overline{w}_i$ $(i = 1, \dots n)$. The MSR can still be computed analogously to (2.20):

$$MSR = \frac{SSR}{n-q} = \frac{(\widehat{\mathbf{y}} - \overline{\mathbf{w}})'(\widehat{\mathbf{y}} - \overline{\mathbf{w}})}{n-q}, \tag{2.25}$$

which uses $\overline{\mathbf{w}}$ instead of $\mathbf{w}$ so this MSR has expected value $var(\overline{w}) = var(w)/m$ instead of $var(w)$.

- The number of replicates is not constant $(m_i \neq m)$. The n averages, $\overline{w}_i$ should then be weighted by the number of replicates; see (2.30) below (and also [184], p. 195).

If input combination $\mathbf{x}_i$ is replicated $m_i > 1$ times, then an alternative for the MSR estimator is the *classic* variance estimator:

$$\widehat{var}(w_i) = \widehat{\sigma}^2(w_i) = s_i^2(w) = \frac{\sum_{r=1}^{m_i}(w_{ir} - \overline{w_i})^2}{m_i - 1} \quad (i = 1, \dots n) \tag{2.26}$$

with

$$\overline{w_i} = \frac{\sum_{r=1}^{m_i} w_{ir}}{m_i}. \tag{2.27}$$

Again, I provide some comments:

- The *average* in (2.27) is computed from the m_i replicates; this average should not be confused with the average computed from the autocorrelated individual waiting times in a single simulation run; see (2.1).

- The average in (2.27) and the sample variance in (2.26) are *independent* variables if the simulation outputs w_{ir} are NIID (see any basic statistics textbook).

- The variance estimator is a *chi-square* variable with $m_i - 1$ degrees of freedom (see any statistics textbook).

- The denominator $m_i - 1$ in (2.26) makes the estimator unbiased; the Maximum Likelihood Estimator (MLE) can be proven to use the denominator m_i. However, I shall not use the Maximum Likelihood (ML) criterion in this book. (Neither will I use a Bayesian criterion.)

Because of the common variance assumption, the n variance estimators in (2.26) can be *pooled* using their degrees of freedom as weights:

$$\widehat{var}(w) = \widehat{\sigma}_w^2 = s^2(w) = \frac{\sum_{i=1}^{n}(m_i - 1)s_i^2}{\sum_{i=1}^{n}(m_i - 1)}. \tag{2.28}$$

So now there are the following two variance estimators:

- The *MSR*—defined in (2.20) for nonreplicated combinations ($m = 1$), and in (2.25) for an equal number of replicates per input combination ($m_i = m > 1$)—which uses the fitted regression model. If the regression model is not valid, then the MSR obviously overestimates the true variance.

- The *pooled* variance estimator in (2.28), which uses $m_i > 1$ replicates. This estimator does not use the fitted regression model; it is unbiased assuming the simulation outputs for a replicated combination are IID (not necessarily NIID; however, the F statistic in (2.30) does assume normality).

These two estimators can be compared through the following so-called *lack-of-fit F-statistic*, still assuming each factor combination i is replicated a constant number of times ($m_i = m$) (also see (3.35):

$$F_{n-q;n(m-1)} = \frac{m}{(n-q)} \frac{(\overline{\mathbf{w}} - \widehat{\mathbf{y}})'(\overline{\mathbf{w}} - \widehat{\mathbf{y}})}{\sum_{i=1}^{n} \widehat{var}(w_i)/n}. \tag{2.29}$$

where $(\sum_{i=1}^{n} \widehat{var}(w_i)/n)/m$ is an unbiased estimator of $var(\overline{w}) = var(w)/m$, and $(\overline{\mathbf{w}} - \widehat{\mathbf{y}})'(\overline{\mathbf{w}} - \widehat{\mathbf{y}})/(n-q)$ is an unbiased estimator of the same quantity only if the regression model is correct. If the number of replicates per combination is not a constant, then this statistic becomes (see, e.g., [268], p. 52):

$$F_{n-q;N-n} = \frac{\sum_{i=1}^{n} m_i(\overline{w_i} - \widehat{y}_i)^2/(n-q)}{\sum_{i=1}^{n}\sum_{r=1}^{m_i}(w_{ir} - \overline{w_i})^2/(N-n)}. \tag{2.30}$$

The numerator uses the MSR computed from the *average* simulation output per combination; at least one combination is replicated (usually, the center of the experimental area is replicated when applying classic DOE to simulation). Obviously, the regression model is rejected if the lack-of-fit F-statistic is significantly high.

I finish this section with the following notes.

- Alternative tests for the validation of the fitted metamodel will be presented in Section 2.11.2. Those tests do not assume white noise. Moreover, they may be applied to other metamodel types, e.g., Kriging models.

- In 2006, [94] discussed plots for assessing lack of fit for linear regression models under the white noise assumption—assuming no replicates (so the lack-of-fit F test in (2.30) does not apply).

- The LS estimator $\widehat{\boldsymbol{\beta}}$ is also the MLE under the white noise assumption.

- There are many—more complex—types of black-box metamodels. Examples are Kriging models and the other metamodel types mentioned in Chapter 1. Now, however, I focus on the simplest—and hence most popular—type that has established a track record in both random and deterministic simulations, namely low-order polynomial regression models.

In summary, in this section I reviewed classic linear regression analysis, which provides tools that give simple metamodels for simulation.

2.3 Linear regression analysis: first-order polynomials

To estimate the parameters of whatever black-box metamodel (e.g., $\boldsymbol{\beta}$ in the linear regression model (2.10)), the analysts must experiment with the simulation model; i.e., they must change the inputs of the simulation, run the simulation, and analyze the resulting I/O data. In this section, I assume that a first-order polynomial is a valid metamodel.

2.3.1 First-order polynomial with a single factor

I start with the simplest metamodel, namely a first-order polynomial with a single factor; see (2.8) above, which has $q = 2$ regression parameters. Elementary mathematics proves that—to fit a straight line—it suffices to have only *two* I/O observations; also see Figure 2.1. This figure displays the expected value of the number of jobs (or customers) in the system in the steady state (this number equals the queue length plus the job being served); i.e., it is the value after “very many” jobs have been simulated (so the effects of the initial state—namely the empty state—has disappeared; see, e.g., [227]). The figure further displays two approximations, namely a first-order polynomial for low traffic rates, and a second-order polynomial for higher traffic rates.

Note: In [413] , Zeigler et al. call the experimental area (see Figure 2.1) the *experimental frame*. I would also call it the domain of admissible scenarios—given the goals of the simulation study (various goals are discussed in [211] and [227]).

I now prove that selecting those two values *as far apart as possible* gives the “best” estimator of the effect of this factor. I use the following two standard assumptions:

- The simulation responses have a *constant variance*; i.e., $var(w_i) = \sigma_w^2$ $(i = 1, \ldots, n)$.
- The n simulation responses are statistically *independent*.

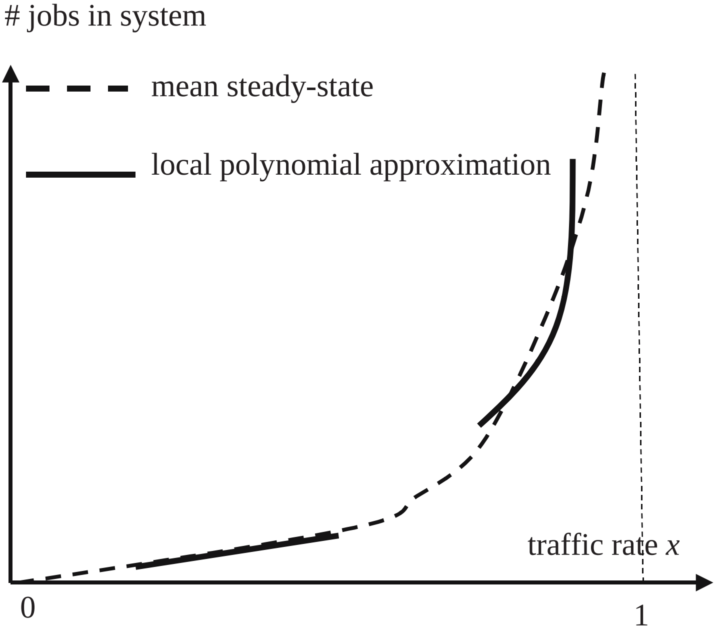

Figure 2.1: M/M/1 with first-order and second-order polynomial approximations of the mean steady-state number of jobs in sytem

These assumptions imply $\mathbf{cov}(\mathbf{w}) = \sigma_w^2\mathbf{I}$ so $\mathbf{cov}(\hat{\boldsymbol{\beta}}) = \sigma_w^2(\mathbf{X}'\mathbf{X})^{-1}$; see (2.17). I denote the lower value of the factor $x = \rho$ in (2.8) by l and the upper value by u.

Exercise 2.5 *Prove that the OLS estimator $\widehat{\beta_1}$ has minimum variance if l and u (lower and upper factor values) are as far apart as possible.*

2.3.2 First-order polynomial with several factors

A first-order polynomial with $k > 1$ factors (inputs) may be represented as follows (I use the classic notation, which denotes the dummy factor by $x_0 = 1$ and its effect by β_0):

$$E(y) = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k. \tag{2.31}$$

So in the general linear regression model (2.10) the variable q (number of regression parameters) now equals $k+1$. An example of this general model is the first-order polynomial for the two factors λ and μ in (2.7).

In practice, a first-order polynomial may be very useful when trying to estimate the *optimal* values for the inputs of a simulation model. For example, the analysts may wish to find the input values that maximize the profit of the simulated company. There are many methods for estimating the optimal input combination (see Chapter 4). Some of these methods use the gradient, which is defined as follows.

Definition 2.7 *The gradient of a function* $w(x_1, \ldots, x_k)$*—usually denoted as* $\nabla(w)$*—is the vector with the first-order partial derivatives:* $\nabla(w) = (\partial w/x_1, \ldots, \partial w/\partial x_k)$.

So the gradient quantifies *local* marginal effects. To estimate the gradient, many mathematicians change one factor at a time—using two or three values per factor (see Section 4.4). From the statistical theory on DOE, however, it follows that it is more efficient to estimate the gradient through a (full or fractional) factorial design and to fit a first-order polynomial to the resulting I/O data.

More general (i.e., not only in optimization), I claim that the LS estimation of the $k+1$ parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ in (2.31) often uses one of the following two design types:

- One-factor-at-a-time designs
- Full factorial designs

In practice, analysts often change each factor one at a time (called the *ceteris paribus* approach in econometrics). DOE, however, may use a 2^k design where k denotes the number of factors and 2 denotes the number of levels (values) per factor. Obviously, two values suffice for the first-order polynomial metamodel (2.31). Before I further discuss these two design types (in Section 2.4), I discuss coding.

It is convenient and traditional in DOE to use *coded*—also called *standardized* or *scaled*—factor values. If each factor has only two levels in the whole experiment with n factor combinations, then these levels may be denoted by -1 and +1. This implies the following linear transformation with z_j denoting the quantitative factor j measured on the original scale, l_j the lower value of z_j in the experiment, and u_j the upper value:

$$x_{ij} = a_j + b_j z_{ij} \text{ with } a_j = \frac{l_j + u_j}{l_j - u_j};\ b_j = \frac{2}{u_j - l_j};\ j = 1, \ldots, k;\ i = 1, \ldots n. \tag{2.32}$$

This transformation implies

$$x_{ij} = \frac{z_{ij} - \overline{z_j}}{(u_j - l_j)/2} \tag{2.33}$$

where $\bar{z}_j$ denotes the average value of input j in a *balanced* experiment, which means that each input has the lower value in half of the n factor combinations (and hence the upper value in the other half); the denominator $(u_j - l_j)$ in (2.33) is known as the *range* of input j (the range is a well-known quantitative measure for the variation; another measure is the variance).

Exercise 2.6 *Suppose that you simulate an M/M/1 queue with a traffic rate between 0.2 and 0.5, and that you fit a first-order polynomial; see (2.8). Code this polynomial metamodel using (2.32). Suppose further that you wish to use the metamodel (2.8) to predict the simulation output for a traffic rate of 0.3 and 0.4 respectively. Which x values correspond with the original traffic rates 0.3 and 0.4?*

The original scale of z in (2.32) may be an interval, a ratio, or an absolute scale (see the discussion of scales at the end of Chapter 1). If the original variable z has a nominal or ordinal scale and it has only two levels, then the coding remains simple: arbitrarily associate one level with -1 and the other level with $+1$. For example, one level may mean that the FIFO priority rule applies, whereas the other level means that LIFO (Last-In-First-Out) applies in a queueing simulation. In another example one level may mean that some patients have preemptive priority (e.g., emergency patients in a hospital simulation), whereas the other level means that this priority does not apply (so all patients are served FIFO); therefore -1 may mean that a rule does not apply or is *switched off*.

In practice, simulation analysts also consider inputs with *nominal scales with more than two levels*. For example, in [188] I present a simulation study on the use of sonar to search for mines at the bottom of the sea. This bottom consists of clay, sand, or rocks—which affects the sonar's output. The simulation analysts erroneously coded these three bottom types as -1, 0, and $+1$. The correct coding of a nominal scale with two or more levels may be done through *multiple binary* variables—each coded as 0 and 1—instead of a single variable that is coded as -1 and $+1$; see the Appendix.

Standardization such that each factor (either quantitative or qualitative) varies between -1 and $+1$ is useful when *comparing* the effects of multiple factors. The example in Figure 2.2 shows two quantitative factors with different ranges (assuming the same scale; if the two scales were different, then two horizontal axes would be needed). The marginal effect of factor 2 is higher than the marginal effect of factor 1. Nevertheless, because the range of factor 1 is much bigger, "the" effect of this factor is larger. If the standardization defined in (2.33) is applied, then the standardized effect of the first factor exceeds that of the second factor.

Elsewhere (namely [40] and [216], p. 178) I present the following first-order polynomial model in the *original* factors centered around their average values in the experiment:

$$E(y) = \delta_0 + \delta_1(z_1 - \overline{z_1}) + \ldots + \delta_k(z_k - \overline{z_k}). \tag{2.34}$$

Obviously, this model implies that the *marginal* effect of factor j is δ_j (the average $\overline{z}_j$ is a constant, determined before the experiment is carried out).

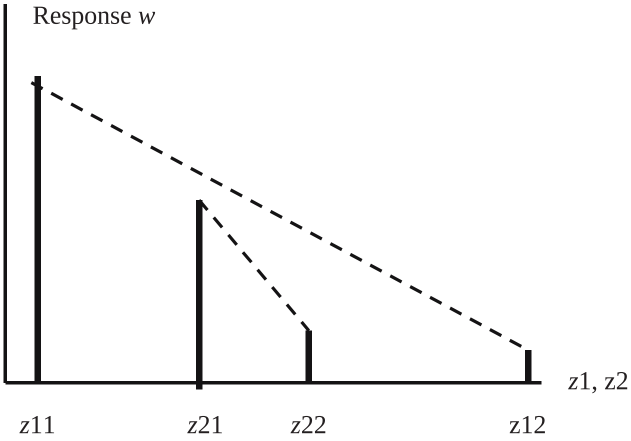

Figure 2.2: Scaling effects when comparing factors

The *total effect* over the range of this factor is

$$\delta_j(u_j - l_j) = 2\beta_j \ (j = 1, \ldots, k)$$

where β_j is the marginal effect of the standardized factor j; all standardized factors have the same range, namely $(1-(-1)) = 2$. The conclusion is that to *rank* the factor effects, the absolute values of the standardized effects β_j should be sorted—if a first-order polynomial is a valid metamodel (else, interactions should also be considered; see Section 2.12 below).

There is a third formulation of the metamodel, namely one using the original noncentered factors:

$$E(y) = \gamma_0 + \gamma_1 z_1 + \ldots + \gamma_k z_k. \tag{2.35}$$

The intercept in the first-order polynomial with standardized factors estimates the simulation output at the center of the experimental area: $E(y) = \beta_0$ if $x_j = 0$ for all j. When using the original non-centered factors in (2.35), the intercept estimates the simulation output when $z_j = 0$ for all j—which may be very far away from the experimental area!

I point out that a factor may be significant when tested through the t statistic defined in (2.19), but may be unimportant—especially when compared with other factors in the experiment. For example, [54] uses many replicates (namely, $m = 500$); all factors turn out to be significant. Reversely, a factor may not be significant, but may still be kept in the metamodel. For example, [93] kept two nonsignificant main effects in the regression model, because these two effects correspond to two decision variables in the simulated Decision Support System (DSS) that is to be optimized. I point out that a nonsignificant estimated effect is still the BLUE.

Note: When comparing a metamodel with the underlying simulation model, the probability that their outputs $\widehat{y}$ and $\overline{w}$ differ significantly increases as the number of replicates increases. Their difference may be important or not—depending on the goals of the metamodel and the simulation model; see [211].

Now I return to one-factor-at-a-time designs versus factorial designs. I first discuss the simplest example with multiple factors, in detail.

Example 2.1 *I suppose that the number of factors is only two: $k = 2$. To select a design type, I compare the variances of the factor effects estimated through a one-factor-at-a-time design and a full factorial design respectively—assuming a first-order polynomial suffices to approximate the simulation I/O behavior. I assume that there are no replicates: $m_i = 1$.*

The one-factor-at-a-time design may be represented by Figure 2.3. This is only one of the possible designs that belong to this popular design class. Other designs in this class use three (instead of two) values (but I have pointed out that two values suffice for a first-order polynomial). Moreover, I assume that the combination denoted by (1) in this figure, is the so-called base value (e.g., the current scenario); the other two combinations increase factor 1 and 2 respectively. Obviously, the design could also be "mirrored" so the first combination would become $(+1, +1)$ instead of $(-1, -1)$.

This figure corresponds with the following design matrix:

$$\mathbf{D} = \begin{bmatrix} -1 & -1 \\ +1 & -1 \\ -1 & +1 \end{bmatrix}.$$

Hence, $\mathbf{X}$ in the general linear regression model (2.10) becomes

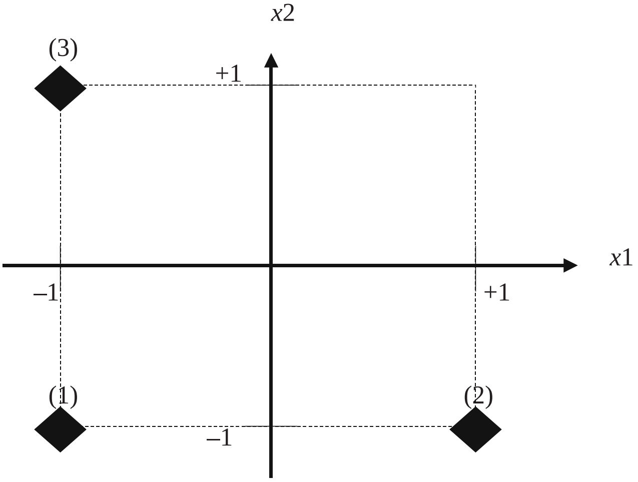

Figure 2.3: A one-factor-at-a-time design for two factors

$$\mathbf{X} = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \\ +1 & -1 & +1 \end{bmatrix}.$$

Assuming (for convenience) that $\sigma_w^2 = 1$ *gives*

$$\mathbf{cov}(\widehat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix}^{-1} = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.25 & 0.5 & 0.25 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}.$$

and

$$\widehat{\boldsymbol{\beta}} = \begin{bmatrix} \widehat{\beta_0} \\ \widehat{\beta_1} \\ \widehat{\beta_2} \end{bmatrix} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} =$$

$$\begin{bmatrix} 0 & 0.5 & 0.5 \\ -0.5 & 0.5 & 0 \\ -0.5 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 0.5w_2 + 0.5w_3 \\ 0.5w_2 - 0.5w_1 \\ 0.5w_3 - 0.5w_1 \end{bmatrix}.$$

This LS estimate agrees with common sense; e.g., β_2 *is estimated by the difference between the third observation in Figure 2.3 and the base observation (combination 1). Note that each of the three regression parameters is estimated from only two of the three outputs.*

The 2^2 *design adds a fourth combination to Figure 2.3, namely the combination* $(+1, +1)$*. Hence* $\mathbf{X}$ *in the general linear regression model (2.10) becomes*

$$\mathbf{X} = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \\ +1 & -1 & +1 \\ +1 & +1 & +1 \end{bmatrix}.$$

The LS formulas give

$$\mathbf{cov}(\widehat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}^{-1} = \begin{bmatrix} 0.25 & 0 & 0 \\ 0 & 0.25 & 0 \\ 0 & 0 & 0.25 \end{bmatrix}$$

and

$$\widehat{\boldsymbol{\beta}} = \begin{bmatrix} \widehat{\beta_0} \\ \widehat{\beta_1} \\ \widehat{\beta_2} \end{bmatrix} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} = \begin{bmatrix} 0.25w_1 + 0.25w_2 + 0.25w_3 + 0.25w_4 \\ 0.25w_2 - 0.25w_1 - 0.25w_3 + 0.25w_4 \\ 0.25w_3 - 0.25w_2 - 0.25w_1 + 0.25w_4 \end{bmatrix}.$$

This LS estimate again agrees with common sense; e.g., β_2 *is now estimated by subtracting the average of the first and second outputs from the average of the third and fourth outputs—which agrees with Figure 2.3, augmented with the fourth combination. Furthermore, each of the three regression parameters is now estimated from all four outputs!*

The variances of each estimated parameter is 0.25 *in the factorial design, whereas these variances are* 0.5 *for the one-at-a-time design. These variances, however, should be corrected for the number of combinations:* $4 \times 0.25 = 1.0$ *and* $3 \times 0.5 = 1.5$ *so the factorial design is more "efficient". (I shall also discuss examples with exactly equal numbers of combinations for both design types.) Moreover, the estimated parameters are uncorrelated* *in the factorial design; in the one-at-a-time design, the correlations are* $0.25/0.5 = 0.5$. *Under the normality assumption, zero correlation implies independence; independent estimators simplify the statistical analysis.*

I give one more example of a 2^k design.

Example 2.2 *Consider a* 2^k *design with* $k = 3$. *Then the* 8×3 *design matrix (say)* $\mathbf{D}$ *is as follows, where (as is conventional in DOE) I give only the signs of the elements (so* $-$ *means* -1, *and* $+$ *means* $+1$*):*

$$\mathbf{D} = \begin{bmatrix} - & - & - \\ + & - & - \\ - & + & - \\ + & + & - \\ - & - & + \\ + & - & + \\ - & + & + \\ + & + & + \end{bmatrix}.$$

It is easy to verify that all the columns of this $\mathbf{D}$ *are orthogonal. Furthermore, each column has the same number of pluses and minuses, namely* $2^{k-1} = 4$ *, so this design is balanced (this property helps to check for typos).*

In general, a 2^k design results in an *orthogonal* matrix of explanatory variables for the first-order polynomial (2.31):

$$\mathbf{X}'\mathbf{X} = n\mathbf{I} \text{ with } n = 2^k. \tag{2.36}$$

This property follows directly from the following general procedure for constructing a 2^kdesign (also see the preceding example with $k = 3$):

1. Select the first 2 elements of column 1 (factor 1) to be $(-1, +1)'$; repeat these two elements—until the column is filled; all columns have $n = 2^k$ elements.

2. The first 2^2 elements of column 2 are $(-1, -1, +1, +1)'$ respectively; repeat these 2^2 elements—until this column is filled.

3. The first 2^3 elements of column 3 are $(-1, -1, -1, -1, +1, +1, +1, +1)'$ respectively; repeat these 2^3 elements—until this column is filled.

4. Repeat this procedure—until the last column is filled, as follows.

5. The first 2^{k-1} elements of the last column (column k) consists of 2^{k-1} consecutive elements -1, followed by 2^{k-1} consecutive elements $+1$.

The orthogonality property simplifies the LS estimator: substituting (2.36) into (2.13) gives

$$\hat{\boldsymbol{\beta}} =(n\mathbf{I})^{-1}\mathbf{X}'\mathbf{w} = \mathbf{X}'\mathbf{w}/n = (\mathbf{x}_j\mathbf{w}/n) = \left(\frac{\sum_{i=1}^{n} x_{ij}w_i}{n}\right) \; (j = 1, \dots q). \tag{2.37}$$

In this equation no matrix inversion is needed. Because—for each j—half the x_{ij} equal -1 and the other half equal $+1$, the estimate $\widehat{\beta}_j$ is simply the difference between two averages:

$$\widehat{\beta}_j = \frac{\sum_{i=1}^{n} x_{ij} w_i/(n/2)}{2} = \frac{\overline{w_{1j}} - \overline{w_{2j}}}{2} \tag{2.38}$$

where $\overline{w_{1j}}$is the average output when factor j is $+1$; $\overline{w_{2j}}$ is the average output when factor j is -1.

Furthermore, the orthogonality property simplifies the covariance matrix (2.17) to

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) = (n\mathbf{I})^{-1}\sigma_w^2 = \mathbf{I}\frac{\sigma_w^2}{n}. \tag{2.39}$$

So all estimators have the same variance, and they are independent.

Note: To rank the estimated effects in the order of their importance, either the estimated effects $\widehat{\beta}_j$ themselves or their t values can be used because the estimated effects have the same estimated variances; see (2.19).

Finally, back in 1952, Box ([46]) proved that the variances of $\widehat{\beta_j}$ (the elements on the main diagonal of (2.17)) are minimal if $\mathbf{X}$ is orthogonal. (These orthogonal matrixes are related to so-called Hadamard matrixes; see [109], [122] and [398] and also [332]).

Altogether, 2^k designs have many attractive properties. Unfortunately, the number of combinations ($n = 2^k$) grows exponentially with the number of factors (k). At the same time, the number of effects is only $q = k + 1$, so these designs become inefficient for high values of k. For example, if $k = 7$, then $2^7 = 128$ whereas $q = 8$. Therefore I shall next present designs that require only a fraction of these 2^k combinations.

Definition 2.8 *An incomplete design has fewer combinations than the corresponding full factorial design.*

This definition deserves the following comments:

- The simplest incomplete designs are 2^{k-p} designs, which are a fraction 2^{-p} of the 2^k design. For example, if $k = 7$, then a 2^{7-4} design (with only $n = 8$ combinations) suffices to fit a first-order polynomial. Details follow in the next section.

- There are also fractions of *mixed-level* designs; e.g., $2^{k_1}3^{k_2}$ designs. I will not discuss these designs in detail, because they are rather complicated, and I have never applied them; also see Section 2.10.

2.4 Designs for first-order polynomials: resolution-III

Definition 2.9 *A resolution-III design gives unbiased estimators of the parameters of a first-order polynomial, assuming such a polynomial is a valid approximation.*

I provide the following comments.

- My definition goes back to Box and Hunter's definition in 1961; see [49] (also see [265]).
- These designs are also known as *Plackett-Burman* designs, published back in 1946; see [296].
- Plackett-Burman designs have as a subclass *fractional factorial two-level* or 2^{k-p} designs; see Section 2.4.1. Obviously, the latter subclass has its number of combinations equal to a power of two. Plackett-Burman designs have their number of combinations equal to a multiple of four and at least equal to $k+1$ (e.g., for $8 \leq k \leq 11$ the Plackett-Burman design has $n = 12$ combinations, which is not a power of two); see Section 2.4.2.

2.4.1 2^{k-p} designs of resolution-III

I start with the simplest example of a 2^{k-p} design, namely $k = 3$. A 2^3 design would require $n = 8$ combinations; see again Example 2.2. The number of parameters is only $q = k + 1 = 4$. Therefore I prefer a 2^{3-1} design, which requires only $n = 4$ combinations. Because this design has resolution-III, it is denoted as a 2^{3-1}_{III}design in the literature. Table 2.1 gives one of the two possible 2^{3-1} designs. The heading "Combi." stands for "Factor combination"; the heading "$\mathbf{3} = \mathbf{1.2}$" is a shorthand notation for $x_{i3} = x_{i1}x_{i2}$ with $i = 1, \ldots n$. Hence, the first element ($i = 1$) in the last column is $x_{13} = x_{11}x_{12} = (-1)(-1) = +1$ so the entry is a plus (+). The DOE literature calls "$\mathbf{3} = \mathbf{1.2}$" a design *generator*; I will discuss generators in more details, after I shall have discussed interactions.

It is easy to verify that Table 2.1 gives an orthogonal $\mathbf{X}$; i.e., (2.36) is satisfied. The design is also balanced (two minuses and two pluses per column).

Figure 2.4 shows the design that corresponds with Table 2.1 This figure has the following *geometric* property: each factor combination corresponds with a vertex that cannot be reached via traversing only one edge of the cube.

Next I discuss Table 2.2. This design belongs to the same *family* as the design in Table 2.1. In this simple example, these two designs together form the full factorial design that was listed in Example 2.2.

Combi.	1	**2**	**3** $= 1.2$
1	$-$	$-$	$+$
2	$+$	$-$	$-$
3	$-$	$+$	$-$
4	$+$	$+$	$+$

Table 2.1: A fractional-factorial two-level design for three factors with generator $3 = 1.2$

The choice between these two designs is *arbitrary* (random). (The association between the three factors and the three columns in the design is also arbitrary; e.g., factor 1 may be associated with column 3. The association between the original levels and the $+$ and $-$ signs is also arbitrary; e.g., the highest value of a factor may be associated with the minus sign—even though this may confuse some users.)

Now I present the next simplest example of a 2^{k-p} design, namely a design with $n = 2^3 = 8$ combinations. The number of factors follows from $2^{k-p} = 8$ or $k - p = 3$ with positive integers k and p, and $2^{k-p} > k$. The solution is $k = 7$ and $p = 4$. This gives Table 2.3, which is the analogue of Table 2.1. It is again easy to check that this design gives an orthogonal $\mathbf{X}$, and it is balanced ($2^{7-5} = 4$ minuses and pluses per column).

The design in Table 2.3 belongs to a bigger *family*. This family is formed by substituting a minus sign for the (implicit) plus sign in one or more generators; e.g., substituting $\mathbf{4} = -\mathbf{1.2}$ for $\mathbf{4} = \mathbf{1.2}$ in Table 2.3 gives one other member of the family. All the ($2^7/2^{7-4}$ =) 16 family members together form the unique (full-factorial two-level) 2^7 design.

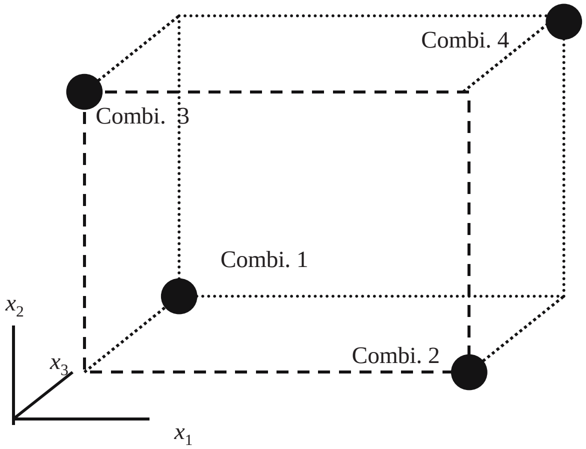

Figure 2.4: A fractional-factorial two-level design for three factors with generator $3 = 1.2$

Combi.	1	**2**	$\mathbf{3} = -1.2$
1	−	−	−
2	+	−	+
3	−	+	+
4	+	+	−

Table 2.2: A fractional-factorial two-level design for three factors with generator 3 = -1.2

Table 2.3 gives a so-called *saturated* design for seven factors; Tables 2.1 and 2.2 gave *saturated* designs for three factors.

Definition 2.10 *A saturated design has as many combinations as the number of parameters to be estimated.*

This definition leads to the following comments.

- In symbols, the definition means $n = q$ in (2.10).
- Hence, no degrees of freedom are left for the MSR in (2.20), so the lack-of-fit F-test in (2.30) cannot be applied. This problem can be easily solved: select one or more combinations from another member of the family, and also simulate this combination; the easiest selection is random.

After discussing the 2^{3-1} and 2^{7-4} designs, I now consider *intermediate* k values: $4 \leq k \leq 6$. Table 2.3 can still be used: for $k = 4$ delete three columns (e.g., the last three columns); for $k = 5$ delete two columns; for $k = 6$ delete one column. Obviously, the resulting designs are not saturated anymore. (Of course, the analysts may also add one or more extra factors to their original list with $4 \leq k \leq 6$ factors; these extra factors do not require a bigger experiment: n remains eight.)

The next example (after Table 2.1 with $n = 4$ and Table 2.3 with $n = 8$) has $n = 2^{k-p} = 16$. So a saturated design implies $k = 15$. Hence $k - p = 4$ implies $p = 15 - 4 = 11$. This 2^{15-11} design may be constructed through the following simple algorithm.

Combi.	**1**	**2**	**3**	**4 = 1.2**	**5 = 1.3**	**6 = 2.3**	**7 = 1.2.3**
1	-	-	-	+	+	+	-
2	+	-	-	-	-	+	+
3	-	+	-	-	+	-	+
4	+	+	-	+	-	-	-
5	-	-	+	+	-	-	+
6	+	-	+	-	+	-	-
7	-	+	+	-	-	+	-
8	+	+	+	+	+	+	+

Table 2.3: A one-fourth fractional factorial design for seven factors

Algorithm 2.1 *1. Construct the (full factorial two-level)* 2^4 *design; i.e., write down the* 16×4 *design matrix.*

2. Add all $(4 \times (4 - 1)/2 = 6)$ *pairwise generators:* $\mathbf{5} = \mathbf{1.2}$, $\mathbf{6} = \mathbf{1.3}$, $\mathbf{7} = \mathbf{1.4}$, *...,* $\mathbf{10} = \mathbf{3.4}$.

3. Add the following four triplet generators: $\mathbf{11} = \mathbf{1.2.3}$, $\mathbf{12} = \mathbf{1.2.4}$, $\mathbf{13} = \mathbf{1.3.4}$, $\mathbf{14} = \mathbf{2.3.4}$.

4. Add the following quadruple generator: $\mathbf{15} = \mathbf{1.2.3.4}$.

The final example that I give (after $n = 4, 8, 16$) has $n = 32$. So a saturated design implies $k = 31$. Hence $k - p = 5$ (so $2^5 = 32$) implies $p = 31 - 5 = 26$. The construction of this 2^{31-26} design remains quite simple, but tedious. A computer procedure is then helpful. To check the computed results, the orthogonality and balance of the resulting design may be verified. It is simple to write such a procedure. I also refer to [332], p. 366 for a different procedure (based on so-called Walsh functions; also see [327]).

2.4.2 Plackett-Burman designs of resolution-III

As I mentioned above, Plackett-Burman designs have 2^{k-p} designs as a subclass. I speak of a Plackett-Burman design *in the narrow sense* if its number of combinations equals a multiple of four, but not a power of two. Actually, Plackett and Burman published such designs for $12 \leq n \leq 96$. In my 1974/1975 book ([181], pp. 332–333), I reproduced these designs (including a misprint: 38 is obviously not a multiple of four; the correct value is 36). These designs are also reproduced in [268], p. 170) for $12 \leq n \leq 36$. The only Plackett-Burman design in the narrow sense that I have ever applied, is the smallest one; see Table 2.4, which has $n = 12$ and $k = 11$. Plackett-Burman designs are again balanced and orthogonal (but they are "nonregular"; see [405]).

Combi.	**1**	2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	+	-	+	-	-	-	+	+	+	-	+
2	+	+	-	+	-	-	-	+	+	+	-
3	-	+	+	-	+	-	-	-	+	+	+
4	+	-	+	+	-	+	-	-	-	+	+
5	+	+	-	+	+	-	+	-	-	-	+
6	+	+	+	-	+	+	-	+	-	-	-
7	-	+	+	+	+	+	+	-	+	-	-
8	-	-	+	+	-	-	+	+	-	+	-
9	-	-	-	+	+	+	-	+	+	-	+
10	+	-	-	-	+	+	+	-	+	+	-
11	-	+	-	-	-	+	+	+	-	+	+
12	-	-	-	-	-	-	-	-	-	-	-

Table 2.4: The Plackett-Burman design for eleven factors

Exercise 2.7 *Apply a resolution-III design to a simulation model of your own choice, provided this model enables you to experiment with (say) between five and twenty factors. Select the ranges of these factors to be "small" (e.g., 1% changes from the base values) so that a first-order polynomial is a valid metamodel. If the simulation model is random, then simulate (say) five replicates. Estimate the main effects of these factors, using the coded and the original factor values respectively. Test whether these effects are significantly different from zero. Give a list that sorts the factors in their order of importance.*

2.5 Regression analysis: factor interactions

Definition 2.11 *Interaction means that the effect of one factor depends on the levels of one or more other factors.*

I offer the following comments on this definition.

- In analytical terms, interaction means $E\left(w|\, x_j = -1\right) - E\left(w|\, x_j = +1\right) = f(x_{j'})$ with $j \neq j'$.
- If the I/O function is continuous, then the preceding expression implies $\partial E(w)/\partial dx_j = f(x_{j'})$ with $j \neq j'$.
- In geometric terms, interaction means that the response curves $E\left(w|\, x_j, x_{j'} = c\right)$ are not parallel for different c values; see the simple example in Figure 2.5, which uses the coded values -1 and $+1$ for the two factors.

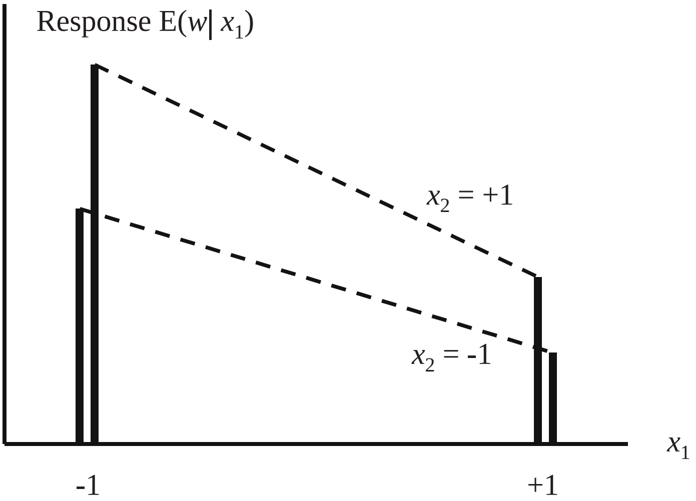

Figure 2.5: Interaction between two factors, x_1 and x_2

The analytical expression for the metamodel corresponding with this example is

$$E(y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{1;2} x_1 x_2. \tag{2.40}$$

This equation implies $\partial E(y)/\partial x_1 = \beta_1 + \beta_{1;2} x_2$, so the effect of x_1 indeed depends on x_2.

- If the interaction between two factors is positive, the factors are called *complementary*. If this interaction is negative, the factors are *substitutes* for each other.

- Sobol' [358] generalizes these definitions from classic linear regression—or better, ANalysis Of VAriance (ANOVA)—to nonlinear models—also called "functional ANOVA"; also see [44], [145], [168], [238], [263], [279], [327], [329], [333], p. 193, and [341].

In general, augmenting the first-order polynomial in (2.31) with two-factor (also called two-way or pairwise) interactions yields

$$E(y) = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k-1} \sum_{j'=j+1}^{k} \beta_{j;j'} x_j x_{j'}. \tag{2.41}$$

It is easy to prove that the total number of interactions in this equation is $k(k-1)/2$, so the total number of parameters is $q = 1 + k + k(k-1)2 = 1 + k(k+1)/2$. The formulation of $\mathbf{X}$ (matrix of explanatory variables) for the metamodel (2.41) follows straightforwardly from $\mathbf{D}$ (design matrix):

$$\mathbf{X} = (x_{ij}) = (1, d_{i1}, \ldots, d_{ik}, d_{i1}d_{i2}, \ldots, d_{i;k-1}d_{ik}). \tag{2.42}$$

In the following case study a first-order polynomial did not give a valid metamodel, but augmenting this polynomial with two-factor interactions did give an adequate approximation.

Example 2.3 *In 1988, Standridge and I published a case study on a Flexible Manufacturing System (FMS) that we did for Pritsker & Associates (see [212]). The factors of our simulation experiment determine the machine mix for our FMS; i.e., z_1, z_2, and z_3 are the number of machines performing operation #1, #2, and #3 respectively; z_4 denotes the number of flexible machines (robots) that may perform any of these three operations. It was easy to derive that the experimental domain should be defined by the following constraints: $5 \leq z_1 \leq 6$, $1 \leq z_2 \leq 2$, $2 \leq z_3 \leq 3$, and $0 \leq z_4 \leq 2$. This domain is quite small indeed, so a first-order polynomial may result in a valid metamodel. Originally, Standridge intuitively specified an incomplete design with $n = 8$ combinations (for details see [212]).Now, however, we select a 2^{4-1} design, which has the same number of combinations; see*

Table 2.3 above with the last three columns deleted (so the generator is $\mathbf{4} = \mathbf{1}.\mathbf{2}$*). Both designs give I/O data, to which we fit first-order polynomials; see (2.31) with* $k = 4$*. As is to be expected, the intuitive design gives bigger variances for the estimated regression parameters; e.g., the variance for the estimated effect of* z_4 *is nearly four times higher (because we use the original scales instead of the standardized scales, the* 2^{4-1} *design does not give constant variances for these parameters). Further analysis of the fitted metamodel (based on the data from the* 2^{4-1} *design) suggests that the first-order polynomial is not adequate, and that the effects of* z_1 *and* z_3 *are negligible (see cross-validation discussed in Section 2.11.2 below). So next, we fit a first-order polynomial in the remaining two factors—adding their interaction; see (2.41) with* $k = 2$*. This model is fitted to the "old" I/O data resulting from the* 2^{4-1} *design. Our analysis suggests that the resulting metamodel is valid. From this metamodel we conclude that the machines in groups #2 and #4 are the bottlenecks of the FMS, and—because the interaction has a negative sign—that machine group #4 (the robots) can serve as a substitute for machine group #2.*

This example demonstrates the usefulness of first-order polynomials augmented with two-factor interactions. The ANOVA literature uses higher-order interactions, e.g., three-factor interactions:

$$E(y) = \beta_0+\sum_{j=1}^{k}\beta_j x_j+\sum_{j=1}^{k-1}\sum_{j'=j+1}^{k}\beta_{j;j'}x_j x_{j'}+\sum_{j=1}^{k-2}\sum_{j'=j+1}^{k-1}\sum_{j''=j'+1}^{k}\beta_{j;j';j''}x_j x_{j'} x_{j''} \tag{2.43}$$

I will not give the definition of these high-order interactions, for two reasons:

1. High-order interactions are hard to interpret (i.e., difficult to explain to the simulation clients).
2. High-order interactions are often unimportant in practice.

Throughout this book, I assume that interactions among three or more factors are unimportant. Of course, this assumption should be checked; see the "lack of fit" and "validation" discussed throughout this book.

2.6 Designs allowing two-factor interactions: resolution-IV

Definition 2.12 *A resolution-IV design gives unbiased estimators of the parameters of a first-order polynomial, even if two-factor interactions are nonzero; all other effects are assumed to be zero.*

Back in 1951, Box and Wilson [51] proved the *foldover* theorem. I formulate their theorem briefly as follows (I quoted the full theorem in [181], p. 343):

Theorem 2.1 *If a resolution-III design* $\mathbf{D}$ *is augmented with its so-called mirror design* $-\mathbf{D}$*, then the resulting design is a resolution-IV design.*

So the price for proceeding from a resolution-III to a resolution-IV design is that the number of combinations doubles.

I now give the following three examples:

1. Only $k = 3$ factors, originally investigated through a 2_{III}^{3-1}design
2. $k = 7$ factors and a 2_{III}^{7-4} design
3. $k = 11$ factors and a Plackett-Burman design.

Example 2.4 *Table 2.1 gave the* 2^{3-1}*design with generator* $\mathbf{3} = \mathbf{1.2}$*. The mirrored design was shown in Table 2.2 , which is the* 2^{3-1}*design with generator* $\mathbf{3} = -\mathbf{1.2}$*. Combining these two designs into a single design gives a* 2^3 *design. This design results in an* $\mathbf{X}$ *with* $n = 8$ *rows and* $q = 1 + 3(3 + 1)/2 = 7$ *columns corresponding with the intercept, the three first-order effects, and the three two-factor interactions. Because all these columns are orthogonal,* $\mathbf{X}$ *is certainly not collinear, and LS estimation is possible. The* $8 - 7 = 1$ *degree of freedom left, could be used to estimate the three-factor interaction; see (2.43). However, if this high-order interaction is assumed to be zero, then this degree of freedom can be used to estimate the common variance* σ_w^2 *through* MSR *defined in (2.20).*

The following example demonstrates that adding the mirror design gives unbiased estimators of the first-order (or main) effects, but does not always enable unbiased estimation of the individual two-factor interactions.

Example 2.5 *Table 2.3 gave a* 2^{7-4} *design. Combining this design with its mirrored design gives a design with* $n = 16$ *combinations (namely, a* 2^{7-3} *design; see below).* $\mathbf{X}$ *corresponding with (2.41) has* $n = 16$ *rows and* $q = 1{+}7(7{+}1)/2 = 29$ *columns, so this* $\mathbf{X}$ *is collinear. Hence, LS estimation of the* 29 *regression parameters is impossible. It is possible, however, to compute the LS estimator of the intercept and the seven first-order effects; see the next exercise.*

Exercise 2.8 *Derive* $\mathbf{X}$ *for the intercept and the seven first-order effects, using the combined design in Example 2.5. Check that—for example—the column for the interaction between the factors 6 and 7 is balanced and orthogonal to the columns for the first-order effects of the factors 6 and 7.*

Now I demonstrate some useful manipulations with the generators in Table 2.1, which gave the 2^{3-1}design with the generator $\mathbf{3} = \mathbf{1.2}$. Remember that $\mathbf{3} = \mathbf{1.2}$ stands for $x_{i3} = x_{i1}x_{i2}$ with $i = 1, \ldots, n$. So postmultiplying both sides of this equation by x_{i3} gives $(x_{i3})^2 = x_{i1}x_{i2}x_{i3}$. Because x_{i3} is either -1 or $+1$ in a 2^{k-p} design, I write $(x_{i3})^2 = +1$. Hence, $x_{i1}x_{i2}x_{i3} = +1$. Moreover, the dummy factor (which has the effect β_0) implies $x_{i0} = +1$.

So, $x_{i1}x_{i2}x_{i3} = x_{i0}$; i.e., the estimates $\widehat{\beta}_0$ and $\widehat{\beta}_{1;2;3}$ are identical. The DOE literature calls $\widehat{\beta}_0$ and $\widehat{\beta}_{1;2;3}$ *confounded* or *aliased.* It is quite easy to prove that $E(\widehat{\beta}_0) = \beta_0 + \beta_{1;2;3}$; i.e., only if $\beta_{1;2;3} = 0$, the estimator $\widehat{\beta}_0$ is unbiased. But in this book I do assume that high-order interactions are zero!

These manipulations may also be written in short-hand notation, using mod(2). I start again with the generator $\mathbf{3} = \mathbf{1.2}$. Postmultiplying both sides with $\mathbf{3}$ gives $\mathbf{3.3} = \mathbf{1.2.3}$ or $\mathbf{3}^2 = \mathbf{1.2.3}$. Applying mod(2) to the exponent gives $\mathbf{3}^0 = \mathbf{1.2.3}$ where $\mathbf{3}^0 = \mathbf{I}$ with $\mathbf{I}$ a column with n ones. So $\mathbf{1.2.3} = \mathbf{I}$, which means that $\widehat{\beta}_0$ and $\widehat{\beta}_{1;2;3}$ are confounded. The literature calls $\mathbf{I} = \mathbf{1.2.3}$ the *defining relation.* It can be proven that this relation has 2^p members—called *words.*

Similar manipulations can be used to derive that more effects are confounded in this example. I start again with the generator $\mathbf{3} = \mathbf{1.2}$ or $\mathbf{I} = \mathbf{1.2.3}$. So $(\mathbf{2.3})\mathbf{I} = (\mathbf{2.3})(\mathbf{1.2.3}) = \mathbf{1.2}^2.\mathbf{3}^2 = \mathbf{1.2}^0.\mathbf{3}^0 = \mathbf{1.I.I} = \mathbf{1}$. So $\mathbf{2.3} = \mathbf{1}$; i.e., $E(\widehat{\beta}_1) = \beta_1 + \beta_{2;3}$. However, using Table 2.1 (a 2^{3-1}design with $\mathbf{3} = \mathbf{1.2}$) assumes that a first-order polynomial (no interactions) is valid, so this design is a resolution-III design. Likewise, it is easy to derive that $\mathbf{1.3} = \mathbf{2}$. Summarizing these equations in the order of the main effects gives $1 = \mathbf{2.3}$, $\mathbf{2} = \mathbf{1.3}$, and $\mathbf{3} = \mathbf{1.2}$.

Table 2.2 gave the 2^{3-1}design with the generator $\mathbf{3} = -\mathbf{1.2}$. It is easy to derive that this generator implies $\mathbf{1} = -\mathbf{2.3}$, $\mathbf{2} = -\mathbf{1.3}$, and $\mathbf{3} = -\mathbf{1.2}$, so $E(\widehat{\beta}_1) = \beta_1 - \beta_{2;3}$, etc.

Similarly, I manipulate the generators of the 2^{7-4} design that were given in Table 2.3. This design has four generators (in general, a 2^{k-p} design has p generators): $\mathbf{4} = \mathbf{1.2}$, $\mathbf{5} = \mathbf{1.3}$, $\mathbf{6} = \mathbf{2.3}$, and $\mathbf{7} = \mathbf{1.2.3}$. Hence $\mathbf{I} = \mathbf{1.2.4} = \mathbf{1.3.5} = \mathbf{2.3.6} = \mathbf{1.2.3.7}$. So $\mathbf{1} = \mathbf{2.4} = \mathbf{3.5} = \mathbf{1.2.3.6} = \mathbf{2.3.7}$. Assuming that high-order interactions are zero, the latter equations give $\mathbf{1} = \mathbf{2.4} = \mathbf{3.5}$. Analogously, it follows that the other main effect estimators are not confounded with any other main effect estimators; the main effect estimators are confounded with two-factor interaction estimators. So this is a resolution-III design.

Exercise 2.9 *Derive the expected value of the main effect estimator for factor 2 in a 2^{7-4} design with the generators $\mathbf{4} = \mathbf{1.2}$, $\mathbf{5} = \mathbf{1.3}$, $\mathbf{6} = \mathbf{2.3}$, and $\mathbf{7} = \mathbf{1.2.3}$, assuming that all high-order interactions are zero.*

A resolution-IV design for $k = 7$ factors may be constructed by adding the mirror design of the preceding 2_{III}^{7-4} design. This gives a design with $n = 16$ combinations. In my 1974/1975 book [181], pp. 336–344, I showed how to derive the generators of a 2_{IV}^{k-p} design (resolution-IV fractional-factorial two-level design). I further showed that $n = 16$ combinations give a resolution-IV design for eight factors (a 2_{IV}^{8-4} design); i.e., one extra factor may be studied when augmenting the 2_{III}^{7-4} with its mirror design. In general, adding the mirror design to a resolution-III design for k factors gives a resolution-IV design for $k + 1$ factors (with $n_{IV} = 2n_{III}$ and n_{III} a multiple of four, possibly a power of two). For example, $k = 11$ requires a Plackett-Burman (resolution-III) design with $n_{III} = 12$ combinations; see (2.4), so a resolution-IV design with $n_{IV} = 24$ combinations enables the estimation of $k = 12$ main effects unbiased by two-factor interactions.

The construction of resolution-IV designs is easy, once a resolution-III design is available. A $\mathbf{D}_{III}$ design (a Plackett-Burman design) is simply augmented with its mirror design, $-\mathbf{D}_{III}$. For the Plackett-Burman subclass of $2_{III}^{(k-1)-p}$ designs, the 2_{IV}^{k-p} designs may be constructed by first defining the full-factorial design in $k - p$ factors, and then aliasing the remaining p factors with high-order interactions among these first $k - p$ factors. For example, $k = 8$ and $n = 16 = 2^4$ leads to a 2^{8-4} design. So first a 2^4 design in four factors is written down. Suppose these four factors are labeled 1, 2, 3, and 4. Next, the following main generators may be used: $\mathbf{5} = \mathbf{1.3.4}$, $\mathbf{6} = \mathbf{2.3.4}$, $\mathbf{7} = \mathbf{1.2.3}$, and $\mathbf{8} = \mathbf{1.2.4}$. It can be derived that the 28 two-factor interactions are confounded in seven groups of size four; see [181], pp. 336–344 and [184]pp. 303–305.

The third (and last) example uses a (resolution-III) Plackett-Burman design in the narrow sense. These designs do not have the simple confounding patterns of 2^{k-p} designs. The latter designs use design generators, which imply that a given column is identical to some other column of $\mathbf{X}$ when that $\mathbf{X}$ includes columns for all the interactions among these k factors. Plackett-Burman designs in the narrow sense lead to an $\mathbf{X}$ that also has $q = 1 + k + k(k-1)/2$ columns. Linear algebra proves that $n < q$ implies that this $\mathbf{X}$ is collinear. Hence, the columns for the main effects and the intercept must be orthogonal to the two-factor interaction columns (since it is a resolution-IV design), but the latter $k(k-1)/2$ columns are not necessarily mutually orthogonal or identical. (The expected value of a specific two-factor interaction estimator is a linear combination of the other two-factor interaction estimators; in 2^{k-p} designs these linear combinations have weights either zero or one—if the principal generators are used.)

The resolution-IV designs discussed so far imply that the number of combinations increases with jumps of eight ($n_{IV} = 8, 16, 24, 32, 40, ...$), because the underlying resolution-III designs have a number of combinations that jump with four ($n_{III} = 4, 8, 12, 16, 20, ...$). Back in 1968, Webb [397] derived resolution-IV designs with n increasing in smaller jumps: $n_{IV} = 2k$ where k does not need to be a multiple of four. He also used the foldover theorem. Because I have never seen any applications of these designs in simulation, I refer to my 1974/1975 book [181], pp.344–348 for details of these designs and their analysis.

In practice, a single simulation run may require so much computer time that a resolution-IV design is hardly possible. The following procedure may help.

1. Simulate all combinations of the resolution-III design.
2. Use the I/O data resulting from step 1, to estimate the first-order polynomial metamodel.
3. Use the metamodel resulting from step 2, to predict the simulation responses of the mirror design of the resolution-III design (the original resolution-III design plus its mirror design form the resolution-IV design).
4. Initialize a counter (say) i: $i = 1$.
5. Simulate combination i of the mirror design.
6. Compare the metamodel prediction from step 3 and the simulation response from step 5; if the prediction error is not acceptable, then increase the counter to $i+1$ and return to step 4; else stop simulating.

I conclude this section on resolution-IV designs with a general discussion of *confounding*. Suppose that a valid linear regression metamodel is

$$E(w) = E(y) = \mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2. \tag{2.44}$$

An example is an $\mathbf{X}_1$ corresponding with the intercept and the main effects collected in $\boldsymbol{\beta}_1$, and an $\mathbf{X}_2$ corresponding with the two-factor interactions $\boldsymbol{\beta}_2$. Suppose that the analysts use the simple metamodel without these interactions. Then they estimate the first-order polynomial coefficients through

$$\widehat{\boldsymbol{\beta}}_1 = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{w}. \tag{2.45}$$

So combining (2.45) and (2.44) gives

$$\begin{aligned} E(\widehat{\boldsymbol{\beta}}_1) = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'E(\mathbf{w}) =& (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'(\mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2) = \\ = \boldsymbol{\beta}_1 + (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2\boldsymbol{\beta}_2 & \end{aligned} \tag{2.46}$$

where $(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2$ is known as the *alias matrix* (see [48]). Equation (2.46) implies an unbiased estimator of $\boldsymbol{\beta}_1$ if either $\boldsymbol{\beta}_2 = \mathbf{0}$ or $\mathbf{X}_1'\mathbf{X}_2 = \mathbf{0}$. Indeed, resolution-III designs assume that $\boldsymbol{\beta}_2 = \mathbf{0}$ where $\boldsymbol{\beta}_2$ consists of the two-factor interactions; resolution-IV designs ensure that $\mathbf{X}_1'\mathbf{X}_2 = \mathbf{0}$ (the two-factor interaction columns are orthogonal to the main effects and intercept columns).

2.7 Designs for two-factor interactions: resolution-V

Definition 2.13 *A resolution-V design enables LS estimation of the first-order effects, the two-factor interactions, and the intercept; all other effects are assumed to be zero.*

The FMS case study in Example 2.3 illustrated that estimation of the individual two-factor interactions may be desirable, in practice. In that example, the number of factors was small (k was four originally, but reduced to two after the analysis of the original 2_{III}^{4-1} design; elimination of the two nonsignificant factors gave a 2^2 design replicate twice).

In the preceding section, I presented a 2_{IV}^{8-4} design. LS estimation of the $q = 1 + 8(8 + 1)/2 = 37$ regression parameters was impossible. Obviously, $n = 64$ combinations enables LS estimation of these 37 parameters—provided these combinations are selected correctly. A correct selection is a 2^{8-2} design. Such a design has $p = 2$ generators. To avoid aliasing among the relevant effects (namely, the two-factors interactions, the main effects, and the intercept), these generators should multiply more than two factors; e.g., a bad generator is $\mathbf{7} = \mathbf{1.2}$ because it gives $\mathbf{I} = \mathbf{1.2.7}$ so $\mathbf{1} = \mathbf{2.7}$, $\mathbf{2} = \mathbf{1.7}$, and of course $\mathbf{7} = \mathbf{1.2}$. Another bad generator is $\mathbf{7} = \mathbf{1.2.3}$, because it implies $\mathbf{I} = \mathbf{1.2.3.7}$ so $\mathbf{1.2} = \mathbf{3.7}$, etc. Actually, the construction of a 2^{8-2} design implies that a full-factorial 2^6 design is constructed first. Next, the generators $\mathbf{7} = \mathbf{1.2.3.4}$ and $\mathbf{8} = \mathbf{1.2.5.6}$ is a good choice, because it implies $\mathbf{I} = \mathbf{1.2.3.4.7} = \mathbf{1.2.5.6.8} = \mathbf{3.4.5.6.7.8}$ where the last equality follows from multiplying the first two members of the identity relation. Hence, these generators imply confounding of two-factor interactions with interactions among three or more factors—the latter (high-order) interactions are assumed to be zero, in this book.

Exercise 2.10 *Prove that the following two generators confound main effects and two-factor interactions (e.g.,* $\mathbf{5} = \mathbf{6.7}$*) if there are seven factors:* $\mathbf{6} = \mathbf{1.2.3.4.5}$ *and* $\mathbf{7} = \mathbf{1.2.3.4}$.

In general, the first-order polynomial augmented with all the two-factor interactions implies that q (number of regression parameters) becomes $1 + k + k(k-1)/2 = (k^2 + k)/2 + 1$, so the number of parameters becomes order k^2 and many more combinations need to be simulated compared with a first-order polynomial. Back in 1961, Box and Hunter [50] published a table with generators for 2^{k-p} designs of resolution V and higher; I reproduced their table in my 1974/1975 book [181], p. 349, and do so again in Table 2.5. Note that this table gives some designs with a resolution higher than V; the definition of these higher resolution is unimportant for DASE.

Recently, Sanchez and Sanchez [332] published a computer procedure for constructing resolution-V designs in case the number of factors is very large: k may be as high as 120. An example is a $2_V^{120-105}$ design. Unfortunately, 2^{k-p} designs—except for the 2_V^{5-1} design—require relatively many combinations to estimate the regression parameters. One example is the 2_{VI}^{9-2} design in Table 2.5, which requires 128 combinations to estimate $q = 1+9(9+1)/2 = 46$ parameters so its "efficiency" is only $46/128 = 0.36$. Another example is the $2_V^{120-105}$ design, which requires $n = 32,768$ whereas $q = 7,261$ so its efficiency is only $7261/32768 = 0.22$.

k	n	generators
5	$2_V^{5-1} = 16$	$\mathbf{5} = \mathbf{1.2.3.4}$
6	$2_{VI}^{6-1} = 32$	$\mathbf{6} = \mathbf{1.2.3.4.5}$
7	$2_{VII}^{7-1} = 64$	$\mathbf{7} = \mathbf{1.2.3.4.5.6}$
8	$2_V^{8-2} = 64$	$\mathbf{7} = \mathbf{1.2.3.4}$; $\mathbf{8} = \mathbf{1.2.5.6}$
9	$2_{VI}^{9-2} = 128$	$\mathbf{9} = \mathbf{1.4.5.7.8}$; $\mathbf{10} = \mathbf{2.4.6.7.8}$
10	$2_V^{10-3} = 128$	$\mathbf{8} = \mathbf{1.2.3.7}$; $\mathbf{9} = \mathbf{2.3.4.5}$; $\mathbf{10} = \mathbf{1.3.4.6}$
11	$2_V^{11-4} = 128$	generators for $k = 10$ plus $\mathbf{11} = \mathbf{1.2.3.4.5.6.7}$

Table 2.5: Generators for fractional-factorial two-level designs of resolution V and higher (VI, VII)

There are resolution-V designs that require fewer runs. For example, [255] gives a design for 47 factors that requires 2,048 combinations, so its efficiency is $1,129/2,048 = 0.55$ (whereas [332] requires 4,096 combinations, so its efficiency is 0.28). And [141] gives a resolution-V design for 64 factors and 4,096 combinations, so its efficiency is 0.51([332] requires 8,192 combinations, so its efficiency is 0.25). For further comparisons among these three types of designs, I refer to [332], pp. 372–373.

Actually, if a simulation run takes much computer time, then *saturated* designs are much more attractive. Back in 1967, Rechtschaffner [311] published simple saturated nonorthogonal fractions of two-level (and three-level) designs; see Table 2.6 (and also [181], p. 352). Their construction is simple: the *generators* are permuted in the different factor combinations; see the design for $k = 4$ factors in Table 2.7 and for $k = 5$ factors in [181], p. 352.

Exercise 2.11 *Compute the variances of the estimated regression parameters that result from the design in Table 2.7. What would these variances have been, had there been an orthogonal saturated design of resolution-V for $k = 4$?*

I applied Rechtschaffner's design in the following case study.

Example 2.6 *The Dutch OR Society organized a competition, challenging the participants to find the combination of six factors that maximizes the output of a simulated system. This challenge was accepted by twelve teams from academia and industry—including one of my graduate students (Pala) and me. Because each team could run only 32 combinations, Pala and I used Rechtschaffner's saturated resolution-V design. So we simulated* $1 + 6 + 6(6-1)/2 = 22$ *combinations; see Table 1 in [208].*

Effect type	Generator
Intercept	$(-1, \ldots, -1)$ for all k factors
Main effect	$(-1, +1, \ldots, +1)$ for all k factors
Two-factor Interaction	$(1, 1, -1, \ldots, -1)$ for $k > 3$ factors

Table 2.6: Generators for Rechtschaffner's resolution-V designs

Combi.	Generator	**1**	**2**	3	**4**
1	$(-1, \ldots, -1)$	-1	-1	-1	-1
2	$(-1, +1, \ldots, +1)$	-1	$+1$	$+1$	$+1$
3		$+1$	-1	$+1$	$+1$
4		$+1$	$+1$	-1	$+1$
5		$+1$	$+1$	$+1$	-1
6	$(+1, +1, -1, \ldots, -1)$	$+1$	$+1$	-1	-1
7		$+1$	-1	$+1$	-1
8		$+1$	-1	-1	$+1$
9		-1	$+1$	$+1$	-1
10		-1	$+1$	-1	$+1$
11		-1	-1	$+1$	$+1$

Table 2.7: Rechtschaffner's design for four factors

2.8 Regression analysis: second-order polynomials

The Taylor series argument implies that—as the experimental area gets bigger or the I/O function gets more complicated—a better metamodel may be a *second-order polynomial*. An example is the M/M/1 simulation: a valid metamodel for the I/O behavior for higher traffic rates in Figure 2.1 may be

$$E(y) = \beta_0 + \beta_1 x + \beta_2 x^2. \tag{2.47}$$

Obviously, estimation of the three parameters in (2.47) requires the simulation of at least *three* input values. Indeed, practitioners often use a one-factor-at-a-time design with three values per factor (they even do so, when fitting a first-order polynomial; above, I showed that this practice is inferior). DOE also provides designs with three values per factor; e.g., 3^k designs. However, more popular in simulation are Central Composite Designs (CCDs), which have five values per factor (see Section 2.9 below).

I emphasize that the second-order polynomial in (2.47) is nonlinear in $\mathbf{x}$ (explanatory regression variables), but linear in $\boldsymbol{\beta}$ (regression parameters). Consequently, such a metamodel remains a linear regression model, which was specified in (2.10).

The general second-order polynomial in k factors is

$$E(y) = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k} \sum_{j' \geq j}^{k} \beta_{j;j'} x_j x_{j'}. \tag{2.48}$$

So this metamodel adds k *purely quadratic* effects $\beta_{j;j}$ to (2.41); consequently, q (total number of effects) becomes $(k+1)(k+2)/2$. In practice, second-order polynomials are applied either locally or globally. *Local* fitting may be used when searching for the optimum input combination; an example is the competition in Example 2.6. I shall return to searching for the optimum in Chapter 4. *Global* fitting (for $0 < x < 1$ in the M/M/1 queueing example) using second-order polynomials has been applied, but I think that Kriging provides better metamodels; see Chapter 5 and also [404].

2.9 Designs for second-degree polynomials: Central Composite Designs (CCDs)

A *CCD* augments a resolution-V design such that the purely quadratic effects can also be estimated. Figure 2.6 gives an example for two factors. In general, a CCD adds the *central* point and $2k$ *axial* points that form a *star design*, where—in the coded factors—the central point is $(0, \dots 0)'$, and the "positive" axial point for factor j (with $j = 1, \dots, k$) is the point with $x_j = +c$ and all other $k-1$ factors fixed at the center (so $x_{j'} = 0$ with $j' = 1, \dots, k$ and $j' \neq j$) and the "negative" axial point for factor j is the point with $x_j = -c$ and $x_{j'} = 0$. Selecting $c = k^{1/2}$ results in a *rotatable* design; i.e., this design gives a *constant* variance for the predicted output at a *fixed* distance from the origin (so the contour functions are circles). Note that a CCD does not give an orthogonal $\mathbf{X}$; hence, the estimated parameters of the second-degree polynomial are correlated. Note further that if n_{CCD} denotes the total number of combinations in a CCD, then $n_{CCD} = n_V + 1 + 2k$; e.g., $k = 2$ implies $n_{CCD} = 2^2 + 1 + 2 \times 2 = 9$; see again Figure 2.6. For $k = 120$, the design in [332] implies $n_{CCD} = 32,768 + 1 + 2 \times 120 = 33,009$. Often only the central point is replicated, to estimate the common variance and to compute the lack-of-fit F-statistic

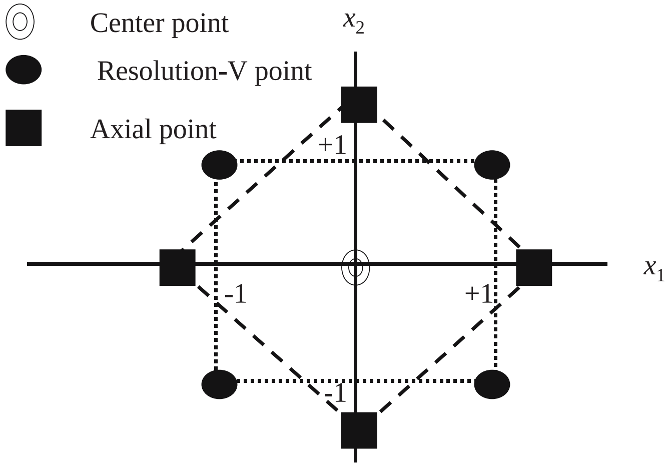

Figure 2.6: A CCD for two factors

defined in (2.30). For further discussion of CCDs, I refer to Myers and Montgomery's classic textbook on RSM [268], and NIST/SEMATECH's e-handbook of statistical methods [275].

Exercise 2.12 *By definition, a rotatable CCD gives a constant variance for the predicted output at a given distance from the origin. Will this constant variance increase or decrease as the output is predicted at a distance farther away from the origin?*

CCDs are rather inefficient because they use inefficient resolution-V designs and add $2k$ axial points so—together with the center point—five values per factor result. Therefore, in Example 2.6, Pala and I simulated only half of the star design; e.g., if the better outputs seem to lie in the southwestern corner of Figure 2.6, then it is efficient to simulate only the two points $(-c, 0)'$ and $(0, -c)'$. I have already emphasized that classic resolution-V designs are very inefficient, so I prefer Rechtschaffner's saturated designs. In my 1987 book [184], pp. 314–316, I discuss three other types of saturated designs for second-order polynomials (due to Koshall, Scheffé, and Notz respectively), but I have never seen any simulation applications of these designs. More designs for second-order polynomials are surveyed in [25], which also references [261].

Exercise 2.13 *Select a model with a known (unconstrained) optimum in your favorite literature (e.g., the Operations Research/Management Science literature on inventory management). Fit a second-order polynomial in the neighborhood of the true optimum, using the coded and the original input values respectively. To fit that polynomial, use a design that enables unbiased estimation of all the coefficients of this polynomial (e.g., a CCD with axial points with a coded value equal to $\sqrt{k}$ where k denotes the number of inputs in your simulation experiment). Replicate only the center point of this design $m > 1$ times. Then estimate the optimal input and output of this simulation model, using the fitted polynomial (again in coded and original values).*

2.10 Optimal designs and other designs

What is an *optimal* design? I discuss the following optimality criteria, which include the so-called alphabetic optimality criteria.

- *A-optimality*: minimize the *trace* of $\mathbf{cov}(\widehat{\boldsymbol{\beta}})$. Obviously, this criterion is related to the criterion that I have (implicitly) used so far, namely minimize the individual variances of the estimated regression parameters, $var(\widehat{\beta_j})$ with $j = 1, \ldots, q$. The A-optimality criterion neglects the off-diagonal elements of $\mathbf{cov}(\widehat{\boldsymbol{\beta}})$; these elements are incorporated in the following criterion.

- *D-optimality*: minimize the determinant of $\mathbf{cov}(\widehat{\boldsymbol{\beta}})$.
- *G-optimality:* minimize the maximum prediction variance, $var(\widehat{y})$.
- *IMSE-optimality:* minimize the MSE integrated over the experimental area. Note that MSE was defined in (2.20); a related criterion is the Root MSE, $RMSE = \sqrt{MSE}$.

Optimal designs do not need to be *orthogonal.*

There is quite some literature on optimal designs. In 1959, Kiefer and Wolfowitz published their classic article on optimal designs; see [178]. And in 1972, Fedorov published his famous book on the same topic; see [111]. A standard text is Pukelsheim's 1993 book, [302]. A recent book is [256]; recent articles are, e.g., [56], [65], [233], [243], and [355]. The recent monograph [265] uses the so-called Minimum Aberration (MA) criterion; also see [405]. I shall return to the construction of optimal designs in the chapter on optimization, Chapter 4, and the chapter on Kriging, Chapter 5. All these criteria assume that n (number of combinations) and q (number of parameters) are fixed (in Chapter 5, however, n will not be a constant; i.e., the design is sequential; [320] discusses Bayesian two-stage designs for low-order polynomial metamodels). References to older literature are given in my 1987 book [184], pp. 335–336.

The DOE literature gives many more design types.

- Whereas resolution-V designs enable the estimation of *all* $k(k-1)/2$ two-factor interactions, some designs enable the estimation of *specific* two-factor interactions only—besides the k main effects and the intercept. In 2005, [2] derived such designs; moreover, these designs enable the estimation of specific three-factor interactions. These designs are optimal in the sense that they are orthogonal. The 2006 paper [124] also assumes that not all two-factor interactions are important; that paper investigates how to discriminate among regression models with different subsets of two-factor interactions. I also refer to the 1974 publication [362] and the 2006 publications [167] and [232].
- In *mixed-level* designs, some factors have two levels, some have three levels, some have four levels, etc. This happens, e.g., when some factors are qualitative with more than two levels, and some are quantitative with two levels. A 2005 article that gives an algorithm for the construction of orthogonal mixed-level designs is [224]. Another 2005 article, namely [242], discusses aliasing among effects. A textbook that includes these designs is [402].
- I have not discussed *blocked* designs, because I assume that blocking may be important in real-life experiments, but not in simulation experiments. More specifically, I assume that in real life the environment cannot be controlled, so undesirable effects may occur. Examples are learning effects during experiments with humans, and extra

wear during experiments with car tires (the right-front tire may wear more than any of the other three tires). In simulation experiments, such undesired effects simply do not occur, because everything is completely controlled—except for the PRNs. Antithetic Random Numbers (ARN) and CRN can be used as a block factor—as originally proposed by [343] (for an update see [96]). I shall briefly return to blocking in a case study discussed in Section 2.12. and in my detailed discussion of Latin Hypercube Sampling (LHS) in Section 4.5.

Randomization is another issue that I claim to be unimportant in simulation experiments, whereas in real life the order in which experimental units (such as tires or cars) are assigned to specific treatments may be important (so this assignment should be in random order, to reduce systematic effects).

- In *weighing* designs the factor levels sum-up to 100%, as is the case in experiments where the factors denote the proportion of chemicals that are used to produce a product; a recent article is [60], which provides more references.

- The usual experimental area is a k-dimensional rectangle (or square if the factors are standardized; see equation 2.32). In some applications, however, the experimental area does not have simple “box” constraints, so different shapes result when the factors must satisfy general constraints. In [215] my coauthors and I study this problem for classic designs used in random simulation (in our harbor simulation the factors should have values such that the traffic rate remains smaller than 100%). A more recent publication , namely [367], solves this problem for maximin designs in deterministic simulation.

- Whereas classic designs keep the factor levels constant during a simulation run, *Frequency Domain Experimentation* (FDE) oscillates these levels during a run. More precisely, each factor has its own oscillation frequency. FDE tries to find which input oscillations affect output oscillations. Originally (in 1987), Schruben and Cogliano [342] proposed this approach. Recently (2005), Sanchez et al. [331] applied FDE for second-order polynomial metamodels with an arbitrary number of factors; they also studied a kanban simulation with 34 factors. Unfortunately, FDE requires rather complicated Fourier spectral analysis.

- The Internet gives information on software for the generation and analysis of more designs. This software includes Stat-Ease’s “Design-Ease” and “Design-Expert” and Crary’s “WebDOE”, and generic statistical software such as Genstat, Minitab, S-Plus, and Statistica; see, e.g., [136], [227], and

http://www.scientific-computing.com/scwfebmar06computational.html.
A library of over 200 orthogonal arrays is maintained by Sloane at A&T; see
http://www.research.att.com/˜njas/oadir/.
More website addresses for metamodeling software are given in [355], which documents a 2002 panel discussion.

2.11 Validation of metamodels

In practice, the simulation analysts do not know over which experimental area (say) a first-order polynomial gives a valid metamodel. This validity depends on the *goals* of the simulation study. The goal may be to find the optimal factor combination of the simulation model; a local metamodel may then be used to estimate the local gradient—which is used to search for the optimum, in a sequence of steps. A different goal may be to identify the factors that are important in a given experimental area; a global metamodel is then needed. Sargent and I discuss this issue in [211].

In Section 2.2, I presented the lack-of-fit F-test, which assumes white noise. In this section, I present the following alternatives:

1. two related coefficients of determination (including R^2) and two related correlation coefficients

2. cross-validation.

These alternatives may be applied to deterministic and random simulation, and to other metamodels than linear regression models; e.g., neural networks (see Section 2.12) and Kriging models.

2.11.1 Coefficients of determination and correlation coefficients

R^2 is a very popular statistic in passive observation of real systems; in active experimentation with replication, the lack-of-fit F-statistic is more popular (see page 23). Whether or not replications are available, R^2 may be defined as follows (also see, e.g., [98], p. 33):

$$R^2 = \frac{\sum_{i=1}^{n}(\widehat{y}_i - \overline{\overline{w}})^2}{\sum_{i=1}^{n}(\overline{w_i} - \overline{\overline{w}})^2} = 1 - \frac{\sum_{i=1}^{n}(\widehat{y}_i - \overline{w_i})^2}{\sum_{i=1}^{n}(\overline{w_i} - \overline{\overline{w}})^2} \tag{2.49}$$

where $\widehat{y}_i$ denotes the metamodel predictor defined in (2.12), $\overline{w_i}$denotes the simulation response of combination i averaged over its $m_i \geq 1$ replicates defined in (2.27), and $\overline{\overline{w}} = \sum_{i=1}^{n} \overline{w_i}/n$ denotes the overall average simulation response. The right-most equality in (2.49) shows that $R^2 = 1$ if

$\widehat{y}_i = \overline{w}_i$ for all i values. R^2 measures how much of the variation in the simulation response is explained by the regression model; see the denominator in (2.49), which is the numerator of the classic variance estimator computed over the n combinations—analogous to (2.26).

I do not define R^2 as a function of the *individual* outputs w_{ir}, because I accept the metamodel as valid if it adequately predicts the *expected* output of the simulation model. Defining R^2 as a function of the individual outputs would decrease the value of R^2 because of the larger variability of the individual outputs per combination.

R^2 may also be used in *deterministic* simulation. In such simulation, the analysts do not obtain any replicates so $\overline{w}_i$ becomes w_i and $\overline{\overline{w}}$ becomes $\overline{w}$ in (2.49).

Obviously, if $n = q$ (no degrees of freedom left; saturated design), then $R^2 = 1$. This value is misleading. Therefore R^2 *adjusted* for the number of explanatory variables is defined as follows:

$$R^2_{adjusted} = 1 - \frac{n-1}{n-q}(1 - R^2). \tag{2.50}$$

Obviously, if $q = 1$, then $R^2_{adjusted} = R^2$.

Lower critical values for either R^2 or $R^2_{adjusted}$ are unknown, because these statistics do not have well-known distributions. Analysts therefore use subjective lower thresholds. In 2006, Deflandre and I demonstrated how the distributions of these two statistics can be obtained through *bootstrapping* (or resampling); see [200]. I shall further discuss the bootstrap approach in the next chapter.

R^2 is also called the *multiple correlation coefficient.* However, R^2 should be distinguished from the *Pearson correlation coefficient*—usually denoted by ρ. As any statistics textbook explains, this ρ quantifies the strength of the linear relationship between two random variables (say) x and w (in classic DOE, x is deterministic—so regression analysis instead of correlation analysis is used). Like R^2, the statistic ρ ranges between -1 and $+1$. A value of $+1$ implies that the two variables are related perfectly by an increasing (positive slope) linear relationship. A value of -1 implies a perfect, decreasing linear relationship. Now I present the formal definition of ρ.

Formally, assume that the (vector) random variable (x, w) is *Bivariate Normally Independently Distributed* with parameters $E(x) = \mu_x$, $E(w) = \mu_w$, $var(x) = \sigma^2_x$, $var(w) = \sigma^2_w$, and $cor(x, w) = \rho(x, w) = \rho$ (so $cov(x, w) = \rho\sigma_x\sigma_w$):

$$(x, w) \sim NID_2(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \text{ with } \boldsymbol{\mu} = (\mu_x, \mu_w)', \ \boldsymbol{\Sigma} = \begin{bmatrix} \sigma^2_x & \rho\sigma_x\sigma_w \\ \rho\sigma_x\sigma_w & \sigma^2_w \end{bmatrix}, \tag{2.51}$$

where the subscript 2 of NID denotes that the vector variate has dimension 2. Then it can be derived that

$$E\left(w|\, x\right) = \beta_0 + \beta_1 x \text{ with } \beta_0 = \mu_w - \beta_1 \mu_x \text{ and } \beta_1 = \rho \frac{\sigma_w}{\sigma_x}. \tag{2.52}$$

It may seem that the relationships in (2.52) can be used to validate a metamodel, as follows. Let the metamodel have output y that approximates the simulation model's output w. Then (2.52) seems to imply that a perfect metamodel has $E\left(y|\, w\right) = w$ if $\beta_0 = 0$ or $\mu_y = \mu_w$ and $\beta_1 = 1$ or $\rho = 1$ and $\sigma_w = \sigma_y$. However, the output w does not have a normal distribution; in fact, w is a function of the simulation inputs that depend deterministically on the design $\mathbf{D}$. Nevertheless, this type of validation may be used in trace-driven simulation; see the Note on page 60.

The parameters μ_x, μ_w, σ_x^2, and σ_w^2 can be estimated in the classic way, analogous to (2.27) and (2.26) respectively. The covariance is then estimated through

$$\widehat{cov}(x, w) = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(w_i - \overline{w})}{n-1}, \tag{2.53}$$

so the correlation is estimated through

$$\widehat{\rho(x, w)} = \widehat{\rho} = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(w_i - \overline{w})}{\sqrt{\sum_{i=1}^{n} (x_i - \overline{x})^2} \sqrt{\sum_{i=1}^{n} (w_i - \overline{w})^2}}. \tag{2.54}$$

A special case is $\rho = 0$. In this case, x and w are independent (zero correlation does not imply independence for nonnormally distributed variables!). To test $H_0 : \rho = 0$, the following t statistic can be used:

$$t_{n-2} = \frac{\widehat{\rho}}{\sqrt{1 - \widehat{\rho}^2}} \sqrt{n-2}. \tag{2.55}$$

The general case of confidence intervals for the correlation coefficient is discussed in [372].

It may happen that the two variables x and w are related, but not through the linear relationship $E\left(w|\, x\right) = \beta_0 + \beta_1 x$ in (2.52). An example of an alternative relationship is $E\left(w|\, x\right) = \beta_0 x^{\beta_1}$. Such an increasing monotonic relationship may be quantified through *Spearman's rank correlation coefficient* (say) η. This coefficient is Pearson's coefficient computed—not from the original pairs (x_i, w_i)—but from the ranked pairs $(r(x_i), r(w_i))$, as follows:

1. The smallest value of x_i is assigned a rank of 1 (so $r(x_i) = 1$ if $x_i = \min_{i'} x_{i'}$), ..., the largest value gets rank n (so $r(x_i) = n$ if $x_i = \max_{i'} x_{i'}$).

2. In case of a "tie" (two or more values happen to be the same), the average rank is assigned to the members of that tie.

3. The ranks for w are computed in the same manner as for x.

To test the null-hypothesis $H_0 : \eta = 0$, Table A10 in [81] can be used. If $n \geq 30$, then this hypothesis may also be tested through $z = \widehat{\eta}\sqrt{n-1}$ where z denotes the standard normal variable so $z \sim N(0,1)$; again see [81], p. 456.

Note: Details on the use of the two related correlation coefficients ρ and η to identify important factors in simulation (not to quantify the adequacy of a metamodel) are given in my article with Helton, [204]; also see [44] and [145].

Example 2.7 *In [204],Helton and I use Pearson's and Spearman's correlation coefficients to identify important factors in a large-scale simulation developed at Sandia National Laboratories in Albuquerque, New Mexico (NM). This simulation estimates the probability of (low-radiation) leakage from the Waste Isolation Pilot Plant (WIPP) near Carlsbad, NM (the radiation may result from nuclear medical treatment; the leakage may be caused by drilling intrusions into the WIPP; most parts of the simulation model are deterministic because they represent physical processes, but some parts are random because they represent human actions). Several performance measures (over a planning horizon of 10,000 years) are considered, in order to obtain permission for building the WIPP. The number of values per factor is one hundred (*$n = 100$*), which is high compared with the two-level designs and the CCDs, which use only two or five values respectively (the WIPP simulation experiment uses LHS, to sample the* n *values per factor; see Section 4.5 on Risk Analysis). The two correlation coefficients quantify the strength of the relationships between an individual factor and a specific simulation output.*

Note: Another measure of dependence (or association) is Kendall's tau; this measure is compared with Spearman's measure in [114].

2.11.2 Cross-validation

Before I discuss cross-validation, I discuss the following type of validation that is often used for the validation of the predictive adequacy of any model in any scientific discipline.

1. First, the analysts use the model to compute a prediction (say) $\widehat{y}$. In DASE, this $\widehat{y}$ may be the outcome of the metamodel; in other areas $\widehat{y}$ may be the outcome of a simulation model or some other model.

2. Next, the analysts observe the actual outcome (say) w. In DASE, w is the simulation outcome; in the other areas, w is the outcome of the real system.

3. Finally, the analysts compare the two outcomes; are the outcomes close? In DASE, this comparison may go as follows.

Assume that the analysts compute their prediction through a linear regression metamodel with parameters $\boldsymbol{\beta}$ estimated from n factor combinations, each replicated m_i times ($i = 1, \ldots, n$). The analysts use this metamodel to predict the actual simulation outcome for a *new* combination $\mathbf{x}_{n+1}$:

$$\widehat{y_{n+1}} = \mathbf{x}'_{n+1}\widehat{\boldsymbol{\beta}} \tag{2.56}$$

where for simplicity I assume that the OLS estimator $\widehat{\boldsymbol{\beta}}$ is used (a more complicated estimator would be the Estimated GLS, EGLS; see Chapter 3).

To estimate the expected simulation outcome for the same combination $\mathbf{x}_{n+1}$, the analysts obtain $m_i > 1$ replicates of the simulation output and compute the average

$$\overline{w_{n+1}} = \frac{\sum_{r=1}^{m_{n+1}} w_{n+1;r}}{m_{n+1}}. \tag{2.57}$$

To compare the outcomes of (2.56) and (2.57), the analysts may use the Studentized statistic

$$t_\nu = \frac{\overline{w_{n+1}} - \widehat{y_{n+1}}}{\sqrt{\widehat{var(\overline{w_{n+1}})} + \widehat{var(\widehat{y_{n+1}})}}} \tag{2.58}$$

where $var(\overline{w_{n+1}})$ is estimated through the classic estimator

$$\widehat{var(\overline{w_{n+1}})} = \frac{\sum_{r=1}^{m_{n+1}} (w_{n+1;r} - \overline{w}_{n+1})^2}{m_{n+1}(m_{n+1} - 1)}$$

and $var(\widehat{y_{n+1}})$ is estimated through the analogue of (2.16):

$$\widehat{var(\widehat{y_{n+1}})} = \mathbf{x}'_{n+1}\widehat{\mathbf{cov}(\widehat{\boldsymbol{\beta}})}\mathbf{x}_{n+1}.$$

Because the two variables $\overline{w_{n+1}}$ and $\widehat{y_{n+1}}$ have different variances, the correct value for ν (degrees of freedom) in (2.58) is not so easy to determine (this is known as the Behrens-Fisher problem; see [100] and also [181]). I think that a simple solution is

$$\nu = \min_{1 \leq i' \leq n+1} m_{i'} - 1.$$

If the statistic in (2.58) is not significant, then the analysts may accept the metamodel as being valid. Next, they may use the "new" observations $w_{n+1;r}$ ($r = 1, \ldots m_{n+1}$) to *re-estimate* the regression parameters $\boldsymbol{\beta}$. The resulting new estimate is expected not to deviate much from the old estimate—assuming the metamodel is valid.

Actually, once the analysts have included the new I/O data in the old data set, the new and the old data may change roles; e.g., $\mathbf{x}_1$ may replace $\mathbf{x}_{n+1}$ in the preceding equations. This idea leads to cross-validation.

Cross-validation is applied not only in linear regression analysis, but also in nonlinear regression analysis, Kriging, neural networks, etc.; see, e.g., [105] and [382]. The basic idea of cross-validation is quite old; see, e.g., Stone's 1974 article [369]. Here I give the so-called *leave-one-out cross-validation* procedure ([355] claims that "leave-k-out cross-validation" may be better for the validation of Kriging metamodels). For ease of presentation, I first assume that $\mathbf{X}$ has only n rows (not $N = \sum_{i=1}^{n} m_i$ rows), so I assume that the number of replicates is constant, possibly one: $m_i = m \geq 1$. If the number of replicates is indeed a constant ($m > 1$), then the LS estimate may replace w_{ir} (individual simulation output for combination i) by $\overline{w_i}$ (average simulation output for combination i); see page 24.

Note: If $m_i > 1$ and $m_i \neq m$ (different replication numbers), then the white noise assumption implies $var(\overline{w_i}) = \sigma_w^2/m_i$; i.e., the variance of $\overline{w_i}$ is not constant. In case of such heterogeneity of variance, the LS formulas need correction (see the next chapter).

The leave-one-out cross-validation procedure runs as follows.

1. Delete I/O combination i from the complete set of n combinations, to obtain the remaining I/O data set $(\mathbf{X}_{-i}, \overline{\mathbf{w}_{-i}})$. I assume that this step results in a noncollinear matrix $\mathbf{X}_{-i}$ $(i = 1, \ldots, n)$; see (2.59) below. To satisfy this assumption, the original matrix $\mathbf{X}$ must satisfy the condition $n > q$. Counterexamples are saturated designs; a simple solution is to simulate one more combination, e.g., the center point if the original design is not a CCD.

2. Recompute the LS estimator of the regression parameters:

$$\widehat{\boldsymbol{\beta}_{-i}} = (\mathbf{X}'_{-i}\mathbf{X}_{-i})^{-1}\mathbf{X}'_{-i}\overline{\mathbf{w}_{-i}}. \tag{2.59}$$

3. Use this recomputed estimator $\widehat{\boldsymbol{\beta}_{-i}}$ to compute the regression prediction for the combination deleted in step 1:

$$\widehat{y_{-i}} = \mathbf{x}'_i\widehat{\boldsymbol{\beta}_{-i}}. \tag{2.60}$$

4. Repeat the preceding three steps, until all n combinations have been processed. This results in n predictions $\widehat{y_{-i}}$ with $i = 1, \ldots, n$.

5. Use a scatterplot with the n pairs $(w_i, \widehat{y_{-i}})$ to judge whether the metamodel is valid.

Note: It is wrong to proceed as follows. Start with the $N \times q$ matrix $\mathbf{X}_N$ (instead of the $n \times q$ matrix $\mathbf{X}$) and the corresponding N-dimensional vector of simulation outputs $\mathbf{w}$ (instead of $\overline{\mathbf{w}}$). Next, delete one row of this

$\mathbf{X}_N$ and the corresponding $\mathbf{w}$ (so $\mathbf{X}_N$ becomes $\mathbf{X}_{N-1}$). From the remaining I/O data, recompute the LS estimator $\widehat{\boldsymbol{\beta}}$ and the regression predictor $\widehat{y}$. I emphasize that this predictor uses $m_i - 1$ simulation outputs for scenario i, so it does not challenge the metamodel to correctly predict the mean simulation output for this scenario! Obviously, if $m_i = 1$, then this procedure is not wrong.

The following two case studies use the cross-validation procedure outlined above:

- a deterministic spreadsheet simulation for the economic appraisal of natural gas projects; see [387]
- a random simulation for the control of animal diseases; see [391].

Note: Scatterplots with $(w_i, \widehat{y_i})$—not $(w_i, \widehat{y_{-i}})$—are used in many deterministic simulations; an example is the simulation of the earth's climate in [139], not using cross-validation of a linear regression metamodel but straightforward validation of a Kriging metamodel. These scatterplots should be distinguished from scatterplots for the validation of *trace-driven* simulation models. The former plots use different factor combinations as inputs; i.e., $(w_i, \widehat{y_i})$ and $(w_{i'}, \widehat{y_{i'}})$ use the combinations $\mathbf{d}_i$ and $\mathbf{d}_{i'}$ with $i \neq i'$. The latter plots use the same factor combination but different realizations of the random input variables. For example, for M/M/1 queueing systems the simulation uses the arrival times that are observed for the real system, in historical order; i.e., the simulated system and the real system are assumed to have the same arrival rate λ, and both systems use the same realizations of the random interarrival time a; also see (2.4). In [196], my coauthors and I prove that this scatterplot gives a line like (2.52) with a slope $\beta_1 < 1$ (so the line does not have a 45 degrees tilt) and an intercept $\beta_0 > 0$—if the simulation model is valid.

Exercise 2.14 *Prove that $\beta_1 < 1$ and $\beta_0 > 0$ if the simulation model with output w is a "valid" model of the real system with output (say) x so $\mu_w = \mu_x$ and $\sigma_w = \sigma_x$, but the simulation model is not "perfect" so $\rho < 1$.*

Back in 1983 (see [183]), I proposed the following alternative for the subjective judgment in step 5, inspired by (2.58): Compute

$$t_{m-1}^{(i)} = \frac{\overline{w_i} - \widehat{y_{-i}}}{\sqrt{\widehat{var(\overline{w_i})} + \widehat{var(\widehat{y_{-i}})}}} \quad (i = 1, \ldots, n) \tag{2.61}$$

where $\widehat{var(\overline{w_i})} = \widehat{var(w_i)}/m$ (and $\widehat{var(w_i)}$ was given in (2.26)) and $\widehat{var(\widehat{y_{-i}})}$ follows from (2.60) and the analogue of (2.16):

$$\widehat{var(\widehat{y_{-i}})} = \mathbf{x}_i' \widehat{\mathbf{cov}(\widehat{\boldsymbol{\beta}}_{-i})} \mathbf{x}_i \tag{2.62}$$

where

$$\widehat{\mathbf{cov}(\widehat{\boldsymbol{\beta}_{-i}})} = \widehat{var(\overline{w_i})}(\mathbf{X}'_{-i}\mathbf{X}_{-i})^{-1}. \tag{2.63}$$

Because (2.61) gives n values (because $i = 1, \ldots, n$), the regression metamodel is rejected if

$$\max_i t^{(i)}_{m-1} > t_{m-1;1-[\alpha/(2n)]} \tag{2.64}$$

where the right-hand side follows from *Bonferroni's inequality*, which implies that the classic type-I error rate (in this case $\alpha/2$) is replaced by the same value divided by the number of tests (in this case n)—resulting in the "experimentwise" or "familywise" type-I error rate α. (Recent references on Bonferroni's inequality are given in [135].)

There is a *shortcut* for the n computations in the cross-validation procedure given above; modern software uses this shortcut. The technique uses the so-called *hat matrix* **H** (see, e.g., [258], pp. 201–202, and also [216], pp. 156–157):

$$\mathbf{H} = (\mathbf{h}_{ii'}) = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \text{ with } i, i' = 1, \ldots, n. \tag{2.65}$$

This **H** is implicitly used in (2.12) where $\widehat{y_i} = \mathbf{x}'_i\widehat{\boldsymbol{\beta}}$, since this equation implies the vector $\widehat{\mathbf{y}} = (\widehat{y_i}) = \mathbf{X}\widehat{\boldsymbol{\beta}}$, which together with (2.13) gives

$$\widehat{\mathbf{y}} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}} = \mathbf{H}\overline{\mathbf{w}}. \tag{2.66}$$

In other words, **H** projects the vector of observations $\overline{\mathbf{w}}$ onto the subspace spanned by **X**. Such a projection matrix is idempotent: $\mathbf{HH} = \mathbf{H}$. Obviously, **H** is an $n \times n$ matrix, so it assumes that the number of replicates is constant, possibly one.

Note: If $m_i > 1$ and $m_i \neq m$ (different replication numbers), then the white noise assumption implies $var(\overline{w_i}) = \sigma^2_w/m_i$; i.e., the variance of $\overline{w_i}$ is not constant. Then a more complicated definition of the hat matrix becomes necessary for the shortcut (see the next chapter, and also [216], p. 157)) (Validation of nonlinear regression metamodels including a modified hat matrix is discussed in [334] and [335].)

From (2.65) it follows that the i^{th} element on the main diagonal of **H** is h_{ii}. It can be proven (see, e.g.,[19], p. 18, 24) that the numerator of (2.61) can be written as

$$\overline{w_i} - \widehat{y_{-i}} = \frac{\overline{w}_i - \widehat{y_i}}{1 - h_{ii}}$$

and (2.61) itself can be written as

$$t_{m_i-1} = \frac{\overline{w_i} - \widehat{y_i}}{\sqrt{\widehat{var(\overline{w_i})}}\sqrt{1 - h_{ii}}} \; (i = 1, \ldots, n) \tag{2.67}$$

so the cross-validation computations can be based solely on the *original* I/O data, $(\mathbf{X}, \mathbf{w})$, which give $\widehat{y_i}$ and h_{ii} (the subscript is i, not $-i$).

Note: If the analysts assumed that the metamodel is valid, then $\widehat{var(\overline{w_i})}$ could be computed from MSE defined in (2.20); also see [283].

I have already pointed out (see page 31) the difference between *significance* and *importance (*a factor may be significant but not important, and vice versa). In situations with many simulation replications, a metamodel may give a predicted value that differs significantly from the simulation output, and yet the metamodel may adequately serve its purpose. For example, [54] uses $m = 500$ replicates when comparing the outcomes of a first-order polynomial approximation and the original simulation for a new scenario, using (2.58). That publication gives a significant difference. Yet the metamodel adequately helps identify the important factors (even though the metamodel is not perfect; i.e., it does not give a scatterplot with all pairs $(w_i, \widehat{y_{-i}})$ on the 45° line).

I emphasize that in *deterministic* simulation, the statistic defined in (2.64) should not be applied, for the following reason. Deterministic simulation implies that the term $\widehat{var(\overline{w_i})}$ in (2.61) is set to zero. The term $\widehat{var(\widehat{y_{-i}})}$ may be computed from (2.62) where (2.63) uses the factor $\widehat{var(\overline{w_i})}$, which may now be computed from the MSR in (2.20). But the worse the metamodel fits, the bigger this MSR gets—so the smaller the test statistic in (2.61) becomes, so the smaller the probability of rejecting this false metamodel becomes! Therefore I propose to compute the *relative* prediction errors $\widehat{y_{-i}}\ /w_i$, and decide whether these errors are acceptable—practically speaking. (In other words, instead of Studentizing the prediction errors, I now standardize the prediction errors by using relative errors.) An alternative remains the scatterplot described in Step 5 of the cross-validation procedure above (on page 59).

Cross-validation not only affects the regression predictions $\widehat{y_{-i}}$, but also the estimated regression parameters $\widehat{\boldsymbol{\beta}_{-i}}$; see (2.60). So the analysts may be interested not only in the predictive performance of the metamodel, but also in its *explanatory* performance—as the following example demonstrates.

Example 2.8 *In this example, I return to Example 2.3, which concerned a case study on a FMS. There are four factors in the simulation experiment, denoted by* z_1 *through* z_4*. The experiment uses a* 2^{4-1} *design. This experiment gives I/O data, to which Standridge and I fit a first-order polynomial. Moreover, we apply cross-validation. In this deterministic simulation experiment, the first-order polynomial gives high relative prediction errors (namely, between* -38% *and* $+33\%$*), and negligible effects of* z_1 *and* z_3*. So next, we fit a first-order polynomial in the remaining two factors augmented with their interaction. We fit this model to the "old" I/O data from the* 2^{4-1} *design. Cross-validation of the new metamodel gives smaller* *relative prediction errors (between* −16% *and* +14%*) and stable important main effects and interaction for z_2 and z_4.*

The regression literature proposes several so-called *diagnostic* statistics that are related to (2.67); e.g., PRESS, DEFITS, DFBETAS, and Cook's D; see [216], p. 157. The simulation literature proposes validation measures that are related to MSE; e.g., $RMSE$, Average Absolute Error (AAE), and Average Absolute Relative Error (AARE). Instead of taking the Mean (see M in the preceding acronyms) or Average (see A), the analysts may take the maximum. The mean is relevant for risk-neutral users, whereas the maximum is for risk-averse users. For further discussion, I refer to my article with Sargent [211] and to [138], [235] and Chapter 5 (on Kriging).

Outside linear regression analysis, the literature also uses either the absolute value or the squared value of the numerator in (2.61)—and ignores the denominator. This gives either $RMSE$ or AAE. For details, I refer to, e.g., [382]. The distribution of such criteria may be estimated through bootstrapping; see [68].

Exercise 2.15 *Simulate the M/M/1 model (also see Exercise 1.6). Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time, or the mean waiting time of (say) the first 100 or 1000 customers. Select two different experimental areas; e.g., the traffic load $\rho = \lambda/\mu$ varies between 0.1 and 0.3 and between 0.5 and 0.8. Select these two areas such that you are pretty sure that a first-order polynomial gives good and bad fit respectively (for "high" traffic rates the first-order polynomial is not a valid metamodel; see Figure 2.1 above). To select these areas, you may "cheat" as follows: draw a plot of the analytical steady-state mean against the traffic rate. Use m_i replicated simulation runs (with nonoverlapping PRN streams). Either ignore the variance heterogeneity within the experimental area or use more replicates for the higher traffic rate; see (3.27) in the next chapter. You may use either a single PRN stream or two streams for arrival and service times—whatever you find convenient. To simplify your analysis, do not apply CRN for different traffic rates. Now validate your metamodel, using different techniques (e.g., the lack-of-fit F-test and cross-validation).*

2.12 More simulation applications

Besides the case studies that I presented above, there are many more (but not enough) applications of linear regression metamodels and experimental designs in simulation (any simulation should use such designs, including their analysis). A 2002 panel with five representatives from industry and government presented more applications of metamodels, including low-order polynomials for deterministic engineering simulations; see [355].

These applications use various designs and metamodels, e.g., a resolution-V fractional factorial design for 10 and 30 factors respectively, and a CCD for 11 factors (built stagewise, starting with a Plackett-Burman design, followed by a foldover design, etc.).

To further illustrate simulation applications of linear regression metamodels, I summarize a few recent applications—more or less in random order.

- Sensitivity Analysis aimed at better understanding (not at prediction) of a simulated automated manufacturing system is performed in [101]. Its authors are especially interested in interactions and the relative importance of factors. They select eight factors, and simulate a 2_V^{8-2} design with ten replicates. They standardize these factors as in (2.33). $R^2_{adjusted}$ is only 0.83 for a first-order polynomial, but increases to 0.96 when adding two-factor interactions. One main effect turns out to be nonsignificant; nine (out of 28) two-factor interactions are significant. To validate the fitted metamodel (keeping only significant effects), the authors randomly select ten combinations that are not a part of the original 2_V^{8-2} design. They compute the AREs $|\overline{w_i} - \widehat{y}_i| / \overline{w_i}$ (with $i = 1, \ldots 10$), and consider both the average and the maximum of these AREs over the ten combinations—as Sargent and I propose in [211]. Because interactions are important, a factor's relative importance is not measured by the absolute value of its main effect only. These authors measure this importance in a special way (see [101], p. 28); I wonder whether a simple overall measure for factor importance is necessary and possible.

- To perform Sensitivity Analysis of a simulated inventory management system in Internet retailing, [20] uses a 2^5 design combined with "design blocking". In this application, the block factor is demand correlation, which has four levels (this correlation may be either high or low, and either positive or negative). Furthermore, the validity of the metamodel is measured through R^2 and $R^2_{adjusted}$. (The normality and variance homogeneity assumptions are checked through graphical analysis of the residuals; see Chapter 3.) This application results in significant main effects and two-factor interactions; high-order interactions are not significant. Two-factor interactions are illustrated through plots analogous to Figure 2.5. The blocking effect is significant, but the authors find it difficult to explain this effect—which supports my claim that simulation experiments should not use blocking, except for the control of ARN and CRN.

- To optimize (also see Chapter 4) a simulated DSS for a production line, [93] uses a 2^5 design with $m = 5$ replicates per combination (and no CRN). The authors estimate a regression metamodel including all interactions (including the five-factor interaction). They keep two nonsignificant main effects, because they want to optimize all five factors (which are decision variables in the DSS). They check the fit of the metamodel through $R^2_{adjusted}$. Moreover, they validate the fitted metamodel by predicting the outputs of eight randomly selected combinations that are not a part of the original 2^5 design. They compute the ARE to quantify the metamodel's validity; they average the ARE over these eight combinations; I point out that an alternative may be the maximum ARE over these combinations—depending on the risk attitude of the users. (The authors check the white noise assumptions through residual plots delivered by Minitab; also see Chapter 3.).

- To predict the performance of a Dial-up Modem Pool (DMP), [339] uses simulation, and analyzes the resulting I/O data through first-order polynomials augmented with two-factor interactions—applying transformations (such as $\ln(x)$ and $1/x$) of the independent variables. Its authors also fit two neural networks and a nonlinear data mining metamodel, using commercial software. They experiment with five factors; e.g., number of modem pools. Each factor has either three or four levels. They use twenty replicates, and eliminate the apparent warm-up period. They select six performance measures; e.g., mean time in queue. To validate their metamodel's predictions, they use a design with the same factors but different levels. To quantify the fit and the validity of their metamodels, they use R^2, the Mean Squared Deviation (MSD), and the Mean Absolute Deviation (MAD). For some factor combinations—but not all combinations—the linear regression metamodels perform significantly poorer than the more complicated metamodels (similar results will be given in Chapter 5 for low-order polynomials versus Kriging). The authors point out that metamodels might be used not only for prediction, but also for explanation; the low-order polynomials best serve explanation resulting in insight (as the FMS case study in Example 2.3 demonstrated).

- To explain (understand) and optimize the parameters of evolutionary search strategies, [26]—also see [27]—applies DOE, linear regression, and GLM (see page 8). At the start there are nine factors; e.g., one factor is the number of parent individuals and another factor is the recombination operator. The output variable (say) w is the quality of the best solution found by the search strategy. A 2^{9-5}_{III} design finds the important factors, after transforming the output w into $\log(w)$. Then only five of these nine factors turn out to be significant. Next, for the four significant quantitative factors, a 2^4 design is used. The resulting I/O data give only two important factors. These two factors are further explored in a CCD design (see my Figure 2.6) with ten replicates for each factor combination.

- Like the preceding publication (namely [26]), another publication (namely [306]) investigates search strategies (including evolutionary strategies, but also Tabu Search, Simulated Annealing, and a hill-climbing procedure). The application concerns the location of ambulances. Its authors distinguish three qualitative factors (e.g., distribution type of demand for ambulances) and two outputs (quality of best solution found by the search strategy, and time to find that solution). To compare the four heuristics, the authors use DOE, accounting for two-factor and three-factor interactions; fortunately, the (hard to interpret) three-factor interaction was nonsignificant.
- Manufacturing simulation and DOE are combined in [225]. This application has three response variables and seven factors, including one nominal factor. It uses a 2^{7-1} design augmented with two central points. These central points have the nominal factor at its two levels, while all other coded, quantitative factors are zero. So the number of simulation runs is 64 ($= 2^{7-1}$) plus the two central points (all together 66 runs). Obviously, no main effects or two-factor interactions are aliased with each other. Its authors use the Design Ease and Minitab software. (They test the white noise assumption through residual plots delivered by Minitab.) Next, they apply RSM—for only two factors and one response variable.

2.13 Conclusions

In this chapter, I gave a tutorial explaining the basics of linear regression models—especially first-order and second-order polynomial models—and the corresponding statistical designs—namely, designs of resolution III, IV, and V, and CCDs. I also discussed the validation of the estimated regression model, including the coefficient of determination R^2 and the adjusted coefficient $R^2_{adjusted}$, Pearson's and Spearman's correlation coefficients, and cross-validation. Throughout this chapter I assumed white noise, meaning that the residuals of the fitted linear regression model are Normally, Independently, and Identically Distributed (NIID) with zero mean. In the next chapter, I shall drop the white-noise assumption, and explain the consequences.

2.14 Appendix: coding of nominal factors

In my first book ([181] p. 299), I discussed the following example to illustrate how to model nominal factors with two or more levels. The example has two factors, called A and B; A has three levels, and B has two levels; no replicates are obtained.

So $\mathbf{X}$ (matrix of explanatory variables) in the general linear regression model (2.10) is

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{2.68}$$

where column 1 corresponds with the dummy factor, columns 2 through 4 with factor A, and columns 4 and 5 with factor B. Row 1 means that in the first factor combination, A is at its first level, and B is also at its first level. Row 2 means that in the second combination, A is still at its first level, but B is at its second level. Row 3 means that in the third combination, A is at its second level, and B is at its lowest level. And so on, until the last combination (row 6) where A is at its third level, and B is at its second level.

In this example, the column of regression parameters in (2.10) becomes $\boldsymbol{\beta} = (\beta_0, \beta_1^A, \beta_2^A, \beta_3^A, \beta_1^B, \beta_2^B)'$. If w denotes the simulation output, then β_0 is the overall or grand mean:

$$\beta_0 = \frac{\sum_{i=1}^{3} \sum_{j=1}^{2} E(w_{ij})}{6}. \tag{2.69}$$

The main effect of factor A at level i is

$$\beta_i^A = \frac{\sum_{j=1}^{2} E(w_{ij})}{2} - \beta_0 \ (i = 1, 2, 3) \tag{2.70}$$

—also see Figure 2.7—and the main effect of factor B at level j is

$$\beta_j^B = \frac{\sum_{i=1}^{3} E(w_{ij})}{3} - \beta_0 \ (j = 1, 2); \tag{2.71}$$

see Figure 2.8, especially the Legend sub 1. These last three equations (namely, (2.69), (2.70), and (2.71)) imply the following constraints:

$$\beta_1^A + \beta_2^A + \beta_3^A = 0 \tag{2.72}$$

and

$$\beta_1^B + \beta_2^B = 0, \tag{2.73}$$

because the three main effects of factor A are defined as the deviations from the average response, as is illustrated in Figure 2.7 where this average is the dotted horizontal line; for factor B a similar argument applies.

If a factor is quantitative, then *interpolation* makes sense; see the dashed line that connects the two responses in Figure 2.8, especially the legend sub 2. (For factor A it seems that a second-order polynomial is a more adequate approximation.) Then the coding with -1 and $+1$ of the main text (instead of 0 and $+1$ in this appendix) may be used, so β_0 becomes the intercept of the polynomial, β^B becomes the marginal effect $\partial E(w)/\partial B$ (which is an

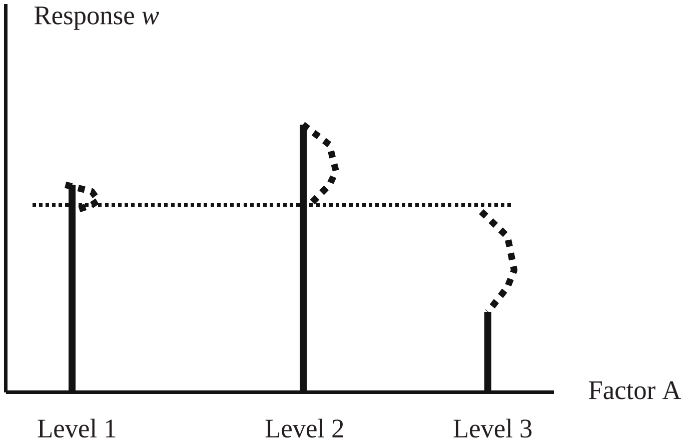

Figure 2.7: Factor A with three levels

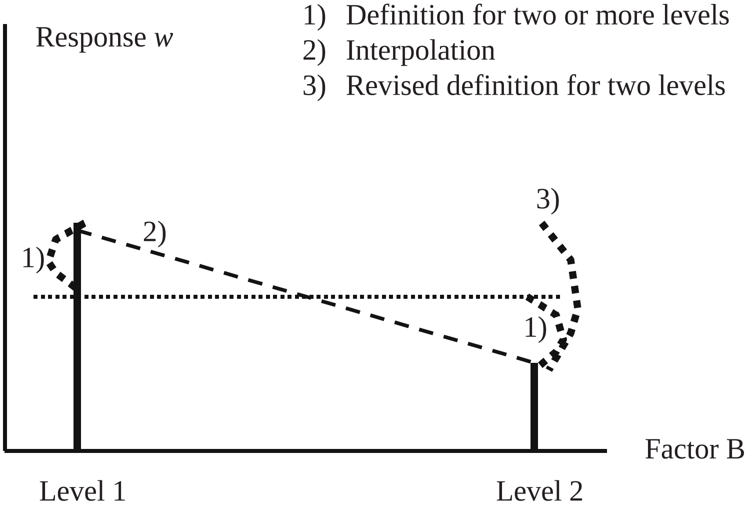

Figure 2.8: Factor B with two levels only

element of the gradient) or the slope of the first-order polynomial, etc. If the factors have two levels only, then an alternative definition also makes sense; see the legend sub 3 in the figure. This alternative defines "the" effect of a factor—not as the deviation from the average—but as the difference between the two mean outputs averaged over all levels of the other factors:

$$\beta^B = \frac{\sum_{i=1}^{3} E(w_{i1})}{3} - \frac{\sum_{i=1}^{3} E(w_{i2})}{3}. \tag{2.74}$$

Note that this definition gives values twice as big as the original one.

The 6 × 6 matrix $\mathbf{X}$ in (2.68) is not of full rank; e.g., summing the columns 2 through 4 or the columns 5 and 6 gives column 1. It can be proven that the rank of $\mathbf{X}$ is four. The normal equations together with the two constraints (2.72) and (2.73) give the unique LS estimate $\widehat{\boldsymbol{\beta}}$; see, e.g., [32] and its references to classic textbooks. These computations are standard in ANOVA software.

2.15 Solutions for exercises

Solution 2.1 $\log(y) = \beta_0 + \beta_1 \log\lambda + ...$ *so* $y = e^{\beta_0+\beta_1 \log\lambda+\cdots}$. *Hence*
$\frac{d}{d\lambda}(e^{\beta_0+\beta_1 \log\lambda}) = \beta_1 e^{\beta_0}\lambda^{\beta_1-1}$,
which upon substitution into the expression for the elasticity coefficient
$(dy/y)/(d\lambda/\lambda) = (dy/d\lambda)(\lambda/y)$
gives
$(\beta_1 e^{\beta_0}\lambda^{\beta_1-1})(\lambda/e^{\beta_0+\beta_1 \log\lambda}) = \lambda\beta_1 \frac{e^{\beta_0}}{e^{\beta_0+(\ln\lambda)\beta_1}}\lambda^{\beta_1-1}$,
which upon some manipulation reduces to β_1.

Solution 2.2 $E(\hat{\boldsymbol{\beta}}) = \mathbf{L}[E(\mathbf{w})] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'[\mathbf{X}\boldsymbol{\beta}] = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})\boldsymbol{\beta} = \boldsymbol{\beta}$.

Solution 2.3 $\mathbf{cov}(\hat{\boldsymbol{\beta}}) = \mathbf{L}[\mathbf{cov}(\mathbf{w})]\mathbf{L}' = [(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'][\sigma_w^2\mathbf{I}][(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}']'$.
Because $(\mathbf{X}'\mathbf{X})^{-1}$ *is symmetric, this expression becomes*
$[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'][\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}]\sigma_w^2 = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1})\sigma_w^2 = (\mathbf{X}'\mathbf{X})^{-1})\sigma_w^2$.

Solution 2.4 *(2.1) in matrix notation becomes*
$\overline{w} = \mathbf{L}\mathbf{w}$ *with* $\mathbf{L} = (1, \ldots, 1)/c$ *and* $\mathbf{w} = (w_1, \ldots, w_c)'$.
Assuming that these waiting times are independent with constant variance σ^2 *gives*
$\mathbf{cov}(\mathbf{w}) = \sigma^2\mathbf{I}$.
Combining with (2.16) gives var$(\overline{w}) = [(1, \ldots, 1)/c][\sigma^2\mathbf{I}][(1, \ldots, 1)'/c] = \sigma^2[(1, \ldots, 1)/c][(1, \ldots, 1)'/c] = \sigma^2[c/(c^2)] = \sigma^2/c$.

Solution 2.5 *From* $\mathbf{cov}(\hat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{X})^{\mathbf{-1}}\sigma_w^2$ *in (2.17) with (say)* $\sigma_w^2 = 1$ *and*

$$\mathbf{X} = \begin{bmatrix} 1 & l \\ 1 & u \end{bmatrix}$$

follows

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 1 & 1 \\ l & u \end{bmatrix}\begin{bmatrix} 1 & l \\ 1 & u \end{bmatrix} = \begin{bmatrix} 2 & l+u \\ l+u & l^2+u^2 \end{bmatrix} \textit{ so}$$

$$(\mathbf{X}'\mathbf{X})^{\mathbf{-1}} = \begin{bmatrix} 2 & l+u \\ l+u & l^2+u^2 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{l^2+u^2}{-2lu+l^2+u^2} & \frac{-l-u}{-2lu+l^2+u^2} \\ \frac{-l-u}{-2lu+l^2+u^2} & \frac{2}{-2lu+l^2+u^2} \end{bmatrix} \textit{ so}$$

var$(\hat{\beta}_1) = \frac{2}{-2lu+l^2+u^2} = \frac{2}{(u-l)^2}$.

This variance is minimal if the denominator $(u-l)^2$ *is maximal, which occurs if* l *and* u *are as far apart as possible.*

Solution 2.6 *The experimental area* $0.2 \leq z \leq 0.5$ *implies*
$a = (0.2 + 0.5)/(0.2 - 0.5) = -2.\,333$
and
$b = 2/(0.5 - 0.2) = 6.667.$
Hence
$x = -2.333 + 6.667z$ *so*
$x_{\min} = -2.333 + (6.667)(0.2) = -1$
and
$x_{\max} = -2.333 + (6.667)(0.5) = 1.$
Further, $z = 0.3$ *implies* $x = -2.333 + (6.667)(0.3) = -0.333.$
Likewise $z = 0.4$ *implies* $x = -2.333 + (6.667)(0.4) = 0.333.$

Solution 2.7 *The answer depends on the simulation model that you selected.*

Solution 2.8 *Table 2.3 gives*

Combi.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	-	-	-	+	+	+	-
2	+	-	-	-	-	+	+
3	-	+	-	-	+	-	+
4	+	+	-	+	-	-	-
5	-	-	+	+	-	-	+
6	+	-	+	-	+	-	-
7	-	+	+	-	-	+	-
8	+	+	+	+	+	+	+

so adding its mirror design and adding the column **6.7** gives

Combi.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	6.7
1	-	-	-	+	+	+	-	-
2	+	-	-	-	-	+	+	+
3	-	+	-	-	+	-	+	-
4	+	+	-	+	-	-	-	+
5	-	-	+	+	-	-	+	-
6	+	-	+	-	+	-	-	+
7	-	+	+	-	-	+	-	-
8	+	-	-	+	+	+	+	+
9	+	+	+	-	-	-	+	-
10	-	+	+	+	+	-	-	+
11	+	-	+	+	-	+	-	-
12	-	-	+	-	+	+	+	+
13	+	+	-	-	+	+	-	-
14	-	+	-	+	-	+	+	+
15	+	-	-	+	+	-	+	-
16	-	-	-	-	-	-	-	+

Solution 2.9 $\mathbf{I} = \mathbf{1.2.4} = \mathbf{1.3.5} = \mathbf{2.3.6} = \mathbf{1.2.3.7}$ *implies* $\mathbf{2} = \mathbf{1.4} = \mathbf{1.2.3.5} = \mathbf{3.6} = \mathbf{1.3.7}$. *Assuming zero high-order effects gives* $\mathbf{2} = \mathbf{1.4} = \mathbf{3.6}$, *so* $E(\widehat{\beta}_2) = \beta_2 + \beta_{1;4} + \beta_{3;6}$.

Solution 2.10 $\mathbf{6} = \mathbf{1.2.3.4.5}$ *implies* $\mathbf{I} = \mathbf{1.2.3.4.5.6}$.
$\mathbf{7} = \mathbf{1.2.3.4}$ *implies* $\mathbf{I} = \mathbf{1.2.3.4.7}$.
So $\mathbf{6.7} = (\mathbf{1.2.3.4.5})(\mathbf{1.2.3.4}) = \mathbf{5}$.

Solution 2.11 *Adding the dummy column for the intercept to Rechtschaffner's design gives*

$$\mathbf{X} = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 \end{bmatrix} \textit{ so } \mathbf{X}'\,\mathbf{X} = \begin{bmatrix} 11 & 1 & 1 & 1 & 1 \\ 1 & 11 & -1 & -1 & -1 \\ 1 & -1 & 11 & -1 & -1 \\ 1 & -1 & -1 & 11 & -1 \\ 1 & -1 & -1 & -1 & 11 \end{bmatrix}$$

$$\textit{so } (\mathbf{X}'\,\mathbf{X})^{-1} = \begin{bmatrix} \frac{2}{21} & -\frac{1}{84} & -\frac{1}{84} & -\frac{1}{84} & -\frac{1}{84} \\ -\frac{1}{84} & \frac{2}{21} & \frac{1}{84} & \frac{1}{84} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{2}{21} & \frac{1}{84} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{2}{21} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{2}{21} \end{bmatrix},$$

whereas an orthogonal design matrix would imply $var(\widehat{\beta_j}) = 1/n = 1/11 = 0.09 < 2/21 = 0.95$.

Solution 2.12 *The variance of the predicted output increases as the input combination moves away from the center of the experimental area; also see the text between the equations (4.3) and (4.4).*

Solution 2.13 *The answer depends on the simulation model that you selected.*

Solution 2.14 *A valid simulation model implies* $\sigma_w = \sigma_x = \sigma$, *which implies* $\beta_1 = \rho\sigma/\sigma = \rho$. *A less than perfect simulation model implies* $\rho < 1$, *so* $\beta_1 < 1$. *A valid simulation model also implies* $\mu_w = \mu_x = \mu$, *which implies that* $\beta_0 = \mu_w - \beta_1\mu_x$ *reduces to* $\beta_0 = \mu - \beta_1\mu = \mu(1 - \beta_1) = \mu(1 - \rho)$. *So* $\beta_0 > 0$ *if* $0 < \rho < 1$ *(not a perfectly valid simulation model) and* $\mu > 0$, *which holds for most queuing simulation outputs (e.g., mean waiting time and mean queue length are positive).*

Solution 2.15 *The answer depends on the metamodel, experimental area, etc. that you selected.*

3
Classic assumptions revisited

This chapter is organized as follows. In Section 3.1, I define the classic assumptions. In Section 3.2, I discuss multivariate simulation output. In Section 3.3, I address possible nonnormality of the simulation output, including tests of normality, transformations of simulation I/O data, jackknifing, and bootstrapping. In Section 3.4, I cover variance heterogeneity of the simulation output. In Section 3.5, I discusses cross-correlated simulation outputs, created through Common Random Numbers (CRN). In Section 3.6, I discuss nonvalid low-order polynomial metamodels. In Section 3.7, I summarize the major conclusions of this chapter. I finish with solutions for the exercises of this chapter.

3.1 Introduction

In this chapter, I return to the assumptions that I used in Chapter 2, in which I discussed classic linear regression analysis and its concomitant designs. These classic assumptions stipulate univariate output and white noise. In practice, however, these assumptions usually do not hold.

Indeed, my general black-box representation in (2.6) implies that the simulation output $\widehat{\mathbf{\Theta}}$ is a *multivariate* random variable. In one example, $\widehat{\Theta}_1$ estimated the mean waiting time and $\widehat{\Theta}_2$ estimated the 90% quantile (also called percentile) of the waiting time distribution. Another example may be that $\widehat{\Theta}_1$ estimates the mean waiting time, and $\widehat{\Theta}_2$ estimates the mean queue length. More examples will follow in Section 3.2.

For the readers' convenience, I repeat the definition of *white noise* that was given in the preceding chapter.

Definition 3.1 *White noise (say) u is Normally, Independently, and Identically Distributed (NIID) with zero mean:* $u \sim NIID(0, \sigma_u^2)$.

This definition implies the following assumptions.

- Normally distributed simulation responses
- No use of CRN across the (say) n factor (or input) combinations
- Common variance of the simulation responses across the n combinations
- Valid metamodel, so the expected values of the residuals of the fitted metamodel are zero.

In this chapter, I shall address the following questions, in the order that they are listed here.

1. How realistic are the classic assumptions, which were used in the preceding chapter?
2. How can these assumptions be tested, if "needed" (i.e., if it is not obvious that the assumption is violated; a counterexample is CRN, which obviously violates the independence assumption)?
3. If an assumption is violated, can the simulation's I/O data be transformed such that the assumption holds?
4. If not, which statistical methods do then apply?

The answers to these questions are scattered throughout the literature on statistics and simulation; in this chapter, I try to answer these questions in a coherent way.

3.2 Multivariate simulation output

In practice, the simulation model often results in multivariate output. A class of practical examples concerns *inventory* simulation models with the following two outputs:

1. the sum of the holding and the ordering costs, averaged over the simulated periods
2. the service (or fill) rate, averaged over the same simulation periods.

The precise definitions of these two outputs vary with the applications. For example, the holding costs may have fixed and variable components; the service rate may be the fraction of total demand per year that is delivered from stock at hand. Moreover there may be multiple inventory items or Stock Keeping Units (SKUs). Inventory simulations are discussed in simulation textbooks such as [227] and in many Management Science/Operations Research (MS/OR) textbooks (I report on inventory simulations together with several coauthors; see [12], [162], and [209]).

A *case study* relevant in this context concerns a Decision Support System (DSS) for production planning based on a simulation model, in which I was involved; see [186]. Originally, this simulation model had a multitude of outputs. However, to support decision making, it turns out that it suffices to consider only the following two outputs (these two DSS criteria form a bivariate response):

1. the total production of steel tubes manufactured (which is of major interest to the production manager)
2. the 90% quantile of delivery times (which is the sales manager's concern).

I shall return to this case study in the chapter on optimization (Chapter 4).

In (2.6), I have already introduced a general black-box representation. Now it is convenient to replace the symbol $\widehat{\boldsymbol{\Theta}}$ by $\mathbf{w}$ to obtain

$$\mathbf{w} = s(d_1, \ldots, d_k, \mathbf{r}_0) \tag{3.1}$$

where

$\mathbf{w}$ denotes the vector of r simulation outputs, so $\mathbf{w} = (w_0, \ldots, w_{r-1})'$ (it is convenient to label the r outputs starting with zero instead of one; see Chapter 4);

$s(.)$ denotes the mathematical function implicitly defined by the computer code that implements the given simulation model;

d_j with $j = 1, \ldots k$ is the j^{th} factor (input variable) of the simulation program (so $\mathbf{D} = (d_{ij})$ is the design matrix for the simulation experiment, with $i = 1, \ldots, n$ and n the number of factor combinations in that experiment);

$\mathbf{r}_0$ is the vector of PseudoRandom Number (PRN) seeds.

Analogous to my assumption in the preceding chapter on univariate output, my current assumption is that the multivariate I/O function $s(.)$ in (3.1) is approximated by r univariate low-order polynomials (in the preceding chapter, I assumed $r = 1$):

$$\mathbf{y}_h = \mathbf{X}\boldsymbol{\beta}_h + \mathbf{e}_h \text{ with } h = 0, \ldots r-1 \tag{3.2}$$

where

$\mathbf{y}_h = (y_{1;h}, \ldots, y_{n;h})'$ denotes the n-dimensional vector with the regression predictor y_h for simulation output w_h;

$\mathbf{X} = (\mathbf{x}_{ij})$ denotes the common $n \times q$ matrix of explanatory variables with $\mathbf{x}_{ij}$ the value of explanatory variable j in combination i $(i = 1, \ldots, n; j = 1, \ldots, q)$; for simplicity, I assume that all fitted regression metamodels are polynomials of the same order (e.g., either first order or second order); if $q \geq 2$ (including an intercept), then the metamodel is called a *multiple* regression model;

$\boldsymbol{\beta}_h = (\beta_{1;h}, \ldots, \beta_{q;h})'$ denotes the q regression parameters for the h^{th} metamodel;

$\mathbf{e}_h = (e_{1;h}, \ldots, e_{n;h})'$ denotes the residuals for the h^{th} metamodel, in the n combinations.

This multivariate regression model violates the classic assumptions, as the following simplistic example illustrates.

Example 3.1 *Suppose that there are only two factor combinations ($n = 2$). Suppose further that each input combination gives three outputs ($r = 3$). No CRN are used. Finally, suppose that the variances and covariances do not vary with the combinations. These assumptions give the following covariance matrix, where I show only the elements on and above the main diagonal because the matrix is symmetric:*

$$\mathbf{cov}(\mathbf{e}) = \begin{bmatrix} \sigma_1^2 & \sigma_{1;2} & \sigma_{1;3} & 0 & 0 & 0 \\ & \sigma_2^2 & \sigma_{2;3} & 0 & 0 & 0 \\ & & \sigma_3^2 & 0 & 0 & 0 \\ & & & \sigma_1^2 & \sigma_{1;2} & \sigma_{1;3} \\ & & & & \sigma_2^2 & \sigma_{2;3} \\ & & & & & \sigma_3^2 \end{bmatrix}.$$

This example illustrates that multivariate residuals $\mathbf{e}$ have the following two properties.

1. The univariate residuals e_h have variances that vary with the output variable w_h $(h = 1, \ldots, r)$; i.e., $\sigma_h^2 \neq \sigma^2$. Practical examples are simulated inventory costs and service percentages, which have different variances so $\sigma_1^2 \neq \sigma_2^2$.

2. The univariate residuals e_h and $e_{h'}$ are not independent for a given input combination i; i.e., $\sigma_{h;h';i} \neq 0$ for $h \neq h'$ Obviously, if these covariances (like the variances) do not vary with the combination i, then this property may be written as $\sigma_{h;h';i} = \sigma_{h;h'} \neq 0$ for $h \neq h'$. For example, "unusual" PRN streams in a given combination i may result in inventory costs that are "relatively high"—that is, higher than expected—and a relatively high service percentage, so these two outputs are positively correlated: $\sigma_{1;2} > 0$.

Because of these two properties ($\sigma_h^2 \neq \sigma^2$ and $\sigma_{h;h'} \neq 0$ for $h \neq h'$), the classic assumptions do not hold. Consequently, it seems that the univariate Ordinary Least Squares (OLS) estimators should be replaced by the *Generalized Least Squares* (GLS) estimator of the parameter vector in the corresponding *multivariate regression* model. Such an approach tends to be rather complicated, so simulation analysts tend to be daunted; also see [126]. Fortunately, in 1967 Rao [310] proved that GLS reduces to OLS computed per output if the *same design* matrix of independent variables (denoted by $\mathbf{X}$) is used (as is the case in simulation with multivariate output); i.e., the Best Linear Unbiased Estimator (BLUE) of $\boldsymbol{\beta}_h$ in (3.2) is

$$\widehat{\boldsymbol{\beta}_h} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}_h \; (h = 0, \ldots, r-1) \tag{3.3}$$

where $\mathbf{w}_h$ was defined below (3.1) and $\mathbf{D} = (d_{ij})$—defined below (3.1)—determines $\mathbf{X}$ in (3.2) and (3.3). More recent references are [244] and [321], p. 703.

Given Rao's result, the simulation analysts can easily obtain confidence intervals and statistical tests for the regression parameters per type of output variable; i.e., the analysts may use the classic formulas presented in the preceding chapter.

3.2.1 Designs for multivariate simulation output

To the best of my knowledge, there are no *general* designs for multivariate output. I consider a simple, artificial example (inspired by [54]).

Example 3.2 *The analysts are interested in two simulation response variables (so $r = 2$ in (3.1)). The number of simulation inputs is 15 (so $k = 15$ in (3.1)). First the analysts try to estimate the first-order effects, so they use a resolution-III design. Obviously, an attractive resolution-III design is a 2^{15-11} design (see Chapter 2). After running this design, they find that the factors labeled 1 through 7 have important main effects for response type 0, while the factors labeled 6 through 15 have important main effects for response type 1. In the next stage of their investigation, the analysts want to estimate the two-factor interactions between those factors that turned out to have important main effects in the first stage. This approach means that the analysts (implicitly) use the "strong heredity" assumption in [402]. That assumption states that if a factor has no important main effect, then this factor does not interact with any other factor (also see Chapter 6). Because the number of two-factor interactions is $k(k-1)/2$, this number sharply increases with the number of factors in the experiment. In this example it is therefore efficient to estimate the interactions in two separate experiments, namely one experiment for each simulation response type. So the analysts split the original group of $k = 15$ factors into two subgroups, namely one subgroup with $k_0 = 7$ factors for the simulation response labeled* 0 *and*

$k_1 = 10$ *factors for the simulation response labeled* 1 *(the factors labeled 6 and 7 are members of both subgroups). The original group with* 15 *factors would require* $1+15+15\times(15-1)/2 = 121$ *factor combinations at least (but classic resolution-V designs are often not saturated at all; see Chapter 2). Now the first subgroup requires at least* $1+7+7\times(7-1)/2 = 29$ *combinations, and the second subgroup requires at least* $1+10+10\times(10-1)/2 = 56$ *combinations. So, together the two subgroups require at least* $29 + 56 = 85$ *instead of* 121 *combinations; i.e., a "divide and conquer" strategy pays off indeed.*

3.3 Nonnormal simulation output

I repeat a remark made in the preceding chapter: Least Squares (LS) is a mathematical criterion, so LS does not assume a normal distribution. Only if the simulation analysts require statistical properties—such as BLUEs, confidence intervals, and tests—then they usually assume a normal (Gaussian) distribution (alternative distributions corresponding with alternative criteria such as the L_1 and the L_∞ norms are discussed in [269]). In this section, I try to answer the following questions (already formulated more generally in Section 3.1):

1. How realistic is the normality assumption?
2. How can this assumption be tested?
3. How can the simulation's I/O data be transformed such that the normality assumption holds?
4. Which statistical methods can be applied that do not assume normality?

3.3.1 Realistic normality assumption?

By definition, *deterministic* simulation models do not have a normally distributed output for a given factor combination; this output is a single fixed value. Nevertheless, simulation analysts often assume a normal distribution for the *residuals* of the fitted metamodel. An example is my case study on coal mining, using deterministic System Dynamics simulation; see [189]. Another case study models global heating caused by the CO_2 greenhouse effect; my coauthors and I use deterministic simulation in [218]. I also refer to the chapter on Kriging (Chapter 5). Indeed, the simulation analysts might argue that so many things affect the residuals that the classic Central Limit Theorem (CLT) applies and a normal distribution is a good assumption. I shall return to the CLT below (immediately after Definition 3.2).

In this subsection, I focus on *random* simulation models. In the next paragraph (and in the Kriging chapter, Chapter 5) I need the following definition.

Definition 3.2 *The time series w_t is a stationary covariance process if it has a constant mean (say) $E(w_t) = \mu$, a constant variance $var(w_t) = \sigma^2$, and covariances depending only on the lag $|t - t'|$ so $cov(w_t, w_{t'}) = \sigma_{|t-t'|}$.*

In practical and academic simulation models the normality assumption often holds *asymptotically*; i.e., if the "sample" size is large, then functions of the simulation data —in particular the sample average of those data—are nearly normal. Basic statistics books mention that the CLT explains why an average is often normally distributed. The CLT assumes that this average has independent components. In simulation, however, the output of a simulation run is often an average computed over that run so the components are *autocorrelated* (serially correlated). Fortunately, there are (sophisticated) variations of the CLT that explain why and when this correlation does not destroy the normality of the average in many simulations. For example, [179] discusses the *Functional Central Limit Theorem* (FCLT) and gives references including Billingsley's 1968 classic textbook [43]. Furthermore, Lehmann's textbook ([229], Chapter 2.8) implies that the average of a *stationary covariance process* remains asymptotically normally distributed if the covariances tend to zero sufficiently fast for large lags.

I add that in inventory simulations the output is often the costs averaged over the simulated periods; this average is probably normally distributed. Another output of an inventory simulation may be the service percentage calculated as the fraction of demand delivered from on-hand stock per (say) week, so "the" output is the average per year computed from these 52 weekly averages. This yearly average may be normally distributed—unless the service goal is "close" to 100%, so the average service rate is cut off at this threshold and the normal distribution is a bad approximation.

Finally, I point out that confidence intervals based on the t statistic are quite insensitive to nonnormality; see the many references in my 1987 book, [184]. However, the lack-of-fit F-statistic is more sensitive to nonnormality; again see [184].

In summary, a limit theorem may explain why random simulation outputs are asymptotically normally distributed. Whether the actual simulation run is long enough, is always hard to know. Therefore it seems good practice to check whether the normality assumption holds (as I shall explain in the next subsection).

3.3.2 Testing the normality assumption

Basic statistics textbooks (also see the recent articles [17],[121], and [170]) and simulation textbooks (see [184] and [227]) propose several *visual plots* and *goodness-of-fit statistics*—such as the chi-square, Kolmogorov-Smirnoff, and Anderson-Darling statistics—to test whether a set of observations come from a specific distribution type such as a normal distribution. These plots and statistics can also be generated through software that is available as an add-on to simulation or statistics software (such software is mentioned throughout this book). A basic assumption is that these observations are Identically and Independently Distributed (IID). Simulation analysts may therefore obtain "many" (say, $m = 100$) replicates for a specific factor combination (e.g., the base scenario) if such an approach is computationally feasible. However, if a single simulation run takes relatively much computer time, then only "a few" (say, $2 \leq m \leq 10$) replicates are feasible, so the plots are too rough and the goodness-of-fit tests lack power. (To obtain more observations on an expensive simulation in an inexpensive way, the analysts may bootstrap a goodness-of-fit test; see [69].)

Actually, the white-noise assumption concerns the metamodel's residuals e, not the simulation model's outputs w. The estimated residuals $\widehat{e_i} = \widehat{y_i} - w_i$ with $i = 1, \ldots n$ were defined in (2.11); an alternative definition was given in (2.25), namely $\widehat{\widehat{e_i}} = \overline{w_i} - \widehat{y_i}$. These two definitions coincide in case of passive observation (as is the case in, e.g., econometric studies), so no replicates are available. I, however, assume that the simulation analysts obtain at least a few replicates for each factor combination ($\forall i : m_i > 1$). For simplicity of presentation, I further assume that the number of replicates is constant ($m_i = m$). If the simulation outputs w have a constant variance (σ_w^2), then $\sigma_{\overline{w}}^2$ ($= \sigma_w^2/m$) is also constant. Unfortunately, even if the average simulation outputs have a constant variance ($\sigma_{\overline{w}}^2$) and are independent (no CRN), the *estimated* residuals do not have a constant variance and they are not independent; it can be proven that

$$\mathbf{cov}(\widehat{\mathbf{e}}) = [\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}']\sigma_{\overline{w}}^2 \tag{3.4}$$

where $\mathbf{X}$ is the $n \times q$ matrix of explanatory regression variables; also see (2.67). This equation uses the well-known hat matrix $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$.

An example of normality testing in simulation is [20]; this publication checks the normality assumption (and the variance homogeneity assumption; see Section 3.4) through a graphical analysis of the estimated residuals (I have already discussed this publication in Section 2.12). Several more simulation publications apply visual inspection of residual plots, which are standard output of many statistical packages; see, e.g., [276].

3.3.3 Transformations of simulation I/O data, jackknifing, and bootstrapping

The simulation output w may be transformed to obtain better normality; e.g., $v = \log(w)$ may be more normally distributed than the original simulation output w. The logarithmic transformation is a special case of the

Box-Cox power transformation:

$$v = \frac{w^{\lambda} - 1}{\lambda} \text{ if } \lambda \neq 0; \text{ else } v = \ln(w) \tag{3.5}$$

where λ is estimated from the original simulation output data. A complication is that the metamodel now explains the behavior of the transformed output, but not the behavior of the original output! See the recent textbook [19] and the articles [77] and [115].

Note: *Outliers* may occur more frequently whenever the actual distribution has "fatter" tails than the normal distribution. *Robust regression analysis* might therefore be applied; see the recent articles [35] and [325] and textbook [19]. However, I have not seen any applications of this approach in simulation.

Normality is not assumed by two general computer-intensive statistical procedures that use the original simulation I/O data $(\mathbf{D}, \mathbf{w})$, namely jackknifing and bootstrapping (actually, the jackknife is a linear approximation of the bootstrap; see the excellent textbook on bootstrapping by Efron and Tibshirani, [104]). Both procedures have become popular since powerful and cheap computers have become available to the analysts (the bootstrap is also used in [325], albeit for robust regression). Special bootstrap procedures are available in many statistical software packages, including the BOOT macro in SAS and the "bootstrap" command in S-Plus; see [277].

Jackknifing

In general, *jackknifing* solves the following two types of problems:

1. How to compute *confidence intervals* in case of nonnormal observations?

2. How to reduce possible *bias* of estimators?

Examples of nonnormal observations are the estimated service rate close to 100% in inventory simulations, and extreme quantiles such as the 99.99% point in Risk Analysis (see the nuclear waste simulations in [204], summarized in Example 2.7). Examples of biased estimators will follow in Section 3.4.

Jackknifing was proposed by Quenouille in 1949 for bias reduction, and by Tukey in 1969 for confidence interval construction; see Miller's classic 1974 review article on jackknifing, [260]. To explain jackknifing, I will use the following linear regression problem.

Suppose the analysts want a confidence interval for the regression coefficients $\boldsymbol{\beta}$ in case the simulation output has a very nonnormal distribution. So the linear regression metamodel is still (2.10). For simplicity, I assume that each factor combination i is replicated an equal number of times ($m_i = m > 1$). The original OLS estimator (also see (2.13)) is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}}. \tag{3.6}$$

Jackknifing resembles cross-validation, in the sense that both procedures drop observations; see Figure 3.1. Leave-one-out cross-validation deletes I/O combination i from the complete set of n combinations, to obtain the remaining I/O data set $(\mathbf{X}_{-i}, \overline{\mathbf{w}_{-i}})$; see Section 2.11.2), whereas jackknifing deletes the r^{th} replicate among the m IID replicates.

The jackknife recomputes the estimator for which a confidence interval is wanted

$$\widehat{\boldsymbol{\beta}_{-r}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}_{-r}}(r = 1, \ldots, m) \tag{3.7}$$

where the n-dimensional vector with average simulation outputs $\overline{\mathbf{w}_{-r}} = (\overline{w_{1;-r}}, \ldots, \overline{w_{i;-r}}, \ldots, \overline{w_{n;-r}})'$ has elements that are the averages of the $m-1$ replicates after deleting replicate r:

$$\overline{w_{i;-r}} = \frac{\sum_{r' \neq r}^{m} w_{i;r'}}{m-1} \tag{3.8}$$

where for the case $r = m$ the summation runs from 1 to $m - 1$ (not m) (a more elegant but more complicated mathematical notation would have been possible).

Obviously, (3.7) gives the m correlated estimators $\widehat{\boldsymbol{\beta}_{-1}}, \ldots, \widehat{\boldsymbol{\beta}_{-m}}$. For ease of presentation, I focus on β_q (the last of the q regression parameters

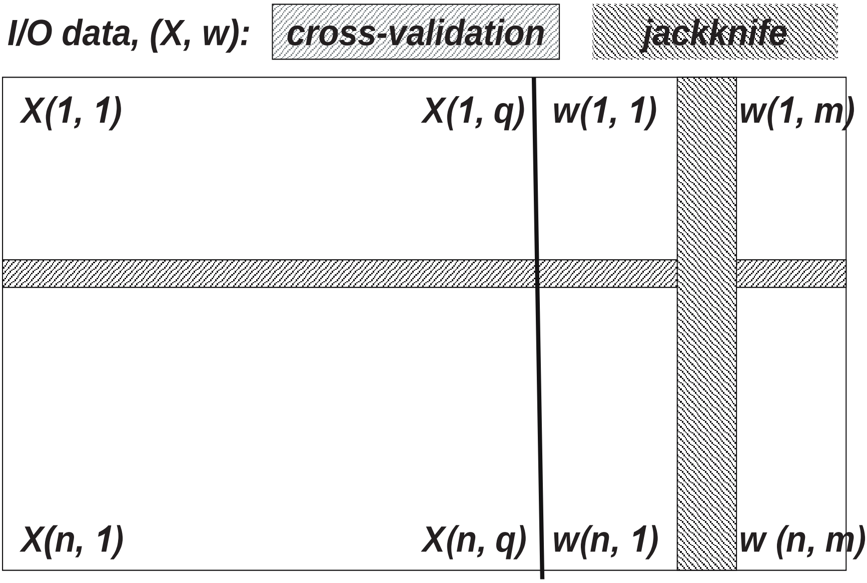

Figure 3.1: Jackknife and cross-validation

in the vector $\boldsymbol{\beta}$). Jackknifing uses the *pseudovalue* (say) J, which is defined as the following weighted average of $\widehat{\beta_q}$ (the original estimator) and $\widehat{\beta_{q;-r}}$ (the q^{th} element of the jackknifed estimator $\widehat{\boldsymbol{\beta}_{-r}}$ defined in (3.7)) with the number of observations as weights:

$$J_r = m\widehat{\beta_q} - (m-1)\widehat{\beta_{q;-r}}. \tag{3.9}$$

In this example both the original and the jackknifed estimators are unbiased, so the pseudovalues also remain unbiased estimators. Otherwise, it can be proven that the bias is reduced by the *jackknife point estimator*

$$\overline{J} = \frac{\sum_{r=1}^{m} J_r}{m}, \tag{3.10}$$

which is simply the average of the m pseudovalues.

To compute a confidence interval, jackknifing treats the pseudovalues as if they were NIID:

$$P(\overline{J} - t_{m-1;1-\alpha/2}\widehat{\sigma_{\overline{J}}} < \beta_q < \overline{J} + t_{m-1;1-\alpha/2}\widehat{\sigma_{\overline{J}}}) = 1 - \alpha \tag{3.11}$$

where $t_{m-1;1-\alpha/2}$ denotes the $1-\alpha/2$ quantile (upper $\alpha/2$ point) of the distribution of the t statistic with $m-1$ degrees of freedom, and

$$\widehat{\sigma_{\overline{J}}} = \sqrt{\frac{\sum_{r=1}^{m}(J_r - \overline{J})^2}{m(m-1)}}. \tag{3.12}$$

The interval in (3.11) may be used for testing whether the true regression parameter is zero; see the null-hypothesis in (2.21).

Applications of jackknifing in simulation are numerous. For example, in [205].my coauthors and I apply jackknifing to obtain confidence intervals for a LS estimator that uses the estimated covariance matrix of the simulation output, $\widehat{\mathbf{cov}(\mathbf{w})}$. With other coauthors I apply jackknifing to reduce the bias and compute confidence intervals for a Variance Reduction Technique (VRT) called control variates or regression sampling; see [207]. Jackknifing may also be applied in the renewal analysis of steady-state simulation (renewal analysis uses ratio estimators, which are biased); see the 1992 textbook that Van Groenendaal and I wrote, [216], pp. 202–203.

Exercise 3.1 *Apply jackknifing to derive a confidence interval for the average waiting time of the first (say) c customers arriving into the M/M/1 system with a traffic rate of (say)* 0.8. *Vary c between (say)* 10 *(terminating simulation) and* 10^7 *(steady-state simulation), and m (number of replicated simulation runs) between (say)* 10 *and* 10^2. *Does this interval cover the analytical steady-state value?*

Exercise 3.2 *Apply jackknifing to derive a confidence interval for the slope (say) β_1 in the simple regression model $w_{ir} = \beta_0 + \beta_1 x_i + e_{ir}$ where e_{ir} is* *nonnormally distributed* $(i = 1, \ldots n; r = 1, \ldots m)$*, e.g.,* e_{ir} *is lognormally distributed with expected value equal to zero. Design a Monte Carlo experiment with (say)* $\beta_0 = 0$ *and* $\beta_1 = 1$*,* $x_i = 1, 2$ *(so* $n = 2$*),* $m = 5$ *and* $m = 25$ *respectively and* 1000 *macro-replicates; sample* e_{ir} *from a lognormal distribution with standard deviation (say)* $\sigma_e = 0.1$ *and shifted such that* $E(e) = 0$*.*

Bootstrapping

In 1982, Efron published his famous monograph on *bootstrapping*; see [103]. In 1993 this monograph was followed by Efron and Tibshirani's classic textbook on bootstrapping, [104]. Recent textbooks on bootstrapping are [73], [90], [134], and [241]; recent articles are [68], [69], and [89] (more references will follow below).

Bootstrapping may be used to solve two types of problems:

1. The relevant distribution is not Gaussian
2. The statistic is not standard.

Sub 1: Nonnormal distribution

As an example, I consider the same problem as I used for jackknifing; i.e., the analysts want a confidence interval for the regression coefficients $\boldsymbol{\beta}$ in case of nonnormal simulation output. Again I assume that each of the n factor combinations is replicated an equal number of times, $m_i = m > 1$ $(i = 1, \ldots, n)$. The original LS estimator was given in (3.6).

The bootstrap distinguishes between the *original observations* w and the *bootstrapped observations* (say) w^* (note the superscript). Standard bootstrapping assumes that the original observations are IID (bootstrapping of time series is discussed in [68], [151], [226], [250], [288] and [301]). In the example, there are $m_i = m$ IID original simulated observations per factor combination i, namely $w_{i;1}, \ldots, w_{i;m}$. These observations give the average simulation output for combination i, namely $\overline{w_i}$. In turn, these averages give the vector $\overline{\mathbf{w}}$, which is a factor in the OLS estimator (3.6).

The bootstrap observations are obtained by *resampling with replacement* from the original observations. This resampling may result in the original observation $w_{i;1}$ being sampled m times, and—because the sample size is kept constant, at m—all the other $m - 1$ original observations $w_{i;2}, \ldots, w_{i;m}$ being sampled zero times. Obviously, this sampling outcome has low probability, but is not impossible. In general, resampling in this example implies that the bootstrapped observations $w^*_{i;1}, \ldots, w^*_{i;m}$ occur with frequencies $f_1, \ldots, f_m$ such that $f_1 + \ldots + f_m = m$ so these frequencies follow the multinomial (or polynomial) distribution with parameters m and $p_1 = \ldots = p_m = 1/m$ (for the multinomial distribution, I refer to any statistics textbook).

This resampling is executed for each combination i ($i = 1, \ldots n$).

The resulting bootstrapped outputs $w^*_{i;1}, \ldots, w^*_{i;m}$ give the bootstrapped average simulation output $\overline{\mathbf{w}^*}$. Substitution into the LS formula (3.6) gives the bootstrapped LS estimator

$$\widehat{\boldsymbol{\beta}^*} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}^*}. \tag{3.13}$$

To reduce sampling variation (also called "sampling error"), this resampling is repeated (say) B times; B is known as the *bootstrap sample size.* A typical value for B is 100 or 1,000. This gives $\widehat{\boldsymbol{\beta}^*}_1, \ldots, \widehat{\boldsymbol{\beta}^*}_B$, which may also be denoted as $\widehat{\boldsymbol{\beta}^*}_b$ with $b = 1, \ldots, B$.

As in the jackknife example, I focus on the single regression parameter β_q. The bootstrap literature gives several alternative procedures for constructing a two-sided confidence interval (including the double bootstrap; see [68]). In practice, the most popular confidence interval is

$$P(\widehat{\beta^*_{q}}_{(\lfloor B\alpha/2)\rfloor)} < \beta_q < \widehat{\beta^*_{q}}_{(\lfloor B(1-\alpha)/2)\rfloor)}) = 1 - \alpha \tag{3.14a}$$

where $\widehat{\beta^*_{q}}_{(\lfloor B\alpha/2)\rfloor)}$ (the left endpoint of the interval) is the (lower) $\alpha/2$ quantile of the Empirical Density Function (EDF) of the bootstrap estimate $\widehat{\beta^*_q}$; i.e., the values of the bootstrap estimate $\widehat{\beta^*_q}$ are sorted from low to high, and $\widehat{\beta^*_{q}}_{(\lfloor B\alpha/2)\rfloor)}$ and $\widehat{\beta^*_{q}}_{(\lfloor B(1-\alpha)/2)\rfloor)}$ are the lower and upper limits of the interval.

Note: The recent article [132] describes bootstrapping as "an artificial bootstrap world is constructed, conditional on the observed data".

I have applied bootstrapping in many situations where classic statistics did not seem appropriate. For example, in [197] my coauthors and I apply bootstrapping to validate trace-driven simulation models in case of serious nonnormal outputs (the test statistic is the difference between the average output of the real and the simulated systems).

Exercise 3.3 *Apply bootstrapping to derive a confidence interval for the average waiting time of the first (say) c customers arriving into the M/M/1 system with a traffic rate of (say)* 0.8*. Vary c between (say)* 10 *(terminating simulation) and* 10^7 *(steady-state simulation?), and m (number of replicated simulation runs) between (say)* 10 *and* 10^2*. Does this interval cover the analytical steady-state value? (Also see Exercise 3.1.)*

Exercise 3.4 *Apply bootstrapping to derive a confidence interval for the slope (say)* β_1 *in the simple regression model* $w = \beta_0 + \beta_1 x + e$ *where* e_{ir} *is nonnormally distributed* $(i = 1, \ldots n; r = 1, \ldots m)$*, e.g.,* e_{ir} *is lognormally distributed with expected value equal to zero. To evaluate this bootstrapping, design a Monte Carlo experiment with (say)* $\beta_0 = 0$ *and* $\beta_1 = 1$*,* $x_i = 1, 2$ *(so* $n = 2$*),* $m = 5$ *and* $m = 25$ *respectively and 1,000 macro-replicates;*

sample e_{ir} from a lognormal distribution with standard deviation (say) $\sigma_e = 0.1$ and shifted such that $E(e) = 0$. (Also see Exercise 3.2.)

Sub 2: Nonstandard statistic

Besides classic statistics such as the t and F statistics, the simulation analysts may be interested in statistics that have no tables with critical values, which provide confidence intervals—assuming normality. For example, R^2 is such a statistic; in [200], Deflandre and I bootstrap R^2 to test the validity of regression metamodels in simulation. (I wonder whether the Sobol' variance decomposition—mentioned below (2.40)—could benefit from bootstrapping.)

In *expensive* simulation there may be only a few replicates; e.g., $m = 1$ or $m = 2$ ([153] uses only two replicates in a Kriging metamodel used for simulation optimization). In such a situation, the *distribution-free bootstrapping* does not work; i.e., resampling with replacement gives the same result "many" times. However, there is also *parametric bootstrapping*; i.e., the analysts assume a specific type of distribution (e.g., a Gaussian distribution); they estimate the distribution's parameters from the original data (e.g., they estimate the mean and the variance of the assumed Gaussian distribution). Next they use PRNs to sample bootstrapped observations from the resulting distribution; i.e., parametric bootstrapping is a specific type of *Monte Carlo* experiment. I shall give examples in Sections 4.3 and 5.2.

I emphasize that using bootstrapping to test a *null-hypothesis* (like $H_0 : E(e) = 0$) requires some more care than estimating a confidence interval for some parameter (like β_q). Indeed, [348], p. 189 warns: "bootstrap hypothesis testing ... is not a well-developed topic." A recent discussion of hypothesis testing versus confidence interval estimation in bootstrapping is [250]; also see [304]. My coauthors and I give examples of bootstrapping for testing the null-hypothesis of a valid simulation model and a valid regression metamodel respectively in [197] and [200].

In general, it is better not to bootstrap the original statistic of interest but the so-called *pivotal* statistic. A statistic is called pivotal if its distribution does not depend on unknown nuisance parameters; e.g., the sample average $\overline{x}$ has the distribution $N(\mu, \sigma^2/n)$ with the unknown nuisance parameter σ, whereas the Studentized statistic $(\overline{x} - \mu)/(s/\sqrt{n})$ has a t_{n-1} distribution, which does not depend on σ so the latter statistic is pivotal. Instead of bootstrapping $\widehat{\beta_q}$ in (3.14a), it is better to bootstrap the Studentized version $\widehat{\beta_q}/s(\widehat{\beta_q})$; also see (2.19)). For further discussion, I refer to [68] and [304].

I end this section on nonnormality with the following remarks.

- A *Generalized Linear Model* (GLM; see page 8) assumes a distribution from the exponential family (instead of a distribution from the normal family). A simulation application is [26].

- More *technical* details on how to bootstrap in simulation can be found in the tutorial that Deflandre and I wrote; see [199].
- Recently, [283] presented several *asymptotic* prediction intervals for regression models with nonnormal outputs. Its author assumes that the residuals are IID. That author modifies the classic interval based on the t statistic; see (2.67). But he also points out that an obvious alternative method is bootstrapping.
- At the same time, [94] discussed bootstrapping and graphical methods for assessing lack-of-fit for linear and nonlinear regression models, allowing nonnormality and heteroskedasticity—assuming that there are no replicates.
- Bootstrapping is used in [112] to estimate a percentile (proportion) in the optimization of simulated manufacturing systems

3.4 Heterogeneous simulation output variances

By definition, *deterministic* simulation models give a single fixed value for a given factor combination, so the conditional variance is zero: $var(w|\mathbf{x}) = 0$. Simulation analysts often assume a normal distribution for the residuals of the metamodel fitted to the I/O data of the deterministic simulation model, as I discussed at the beginning of Section 3.3.1. Usually, the analysts then assume a normal distribution with a *constant* variance (also see the Kriging chapter, Chapter 5). I do not know a better assumption that works in practice for deterministic simulation models.

In this subsection, I further focus on *random* simulation models. I try to answer the following questions (formulated more generally in Section 3.1):

1. How *realistic* is the common (homoscedastic) variance assumption?
2. How can this assumption be *tested*?
3. How can the simulation's I/O data be *transformed* such that the common variance assumption holds?
4. Which statistical *analysis* methods can be applied that allow nonconstant (heteroskedastic, heterogeneous) variances?
5. Which statistical *design* methods can be applied that allow nonconstant variances?

3.4.1 Realistic constant variance assumption?

In practice, random simulation outputs usually have heterogeneous variances, as factor combinations change. For example, in the M/M/1 queueing simulation not only the expected value (mean, first moment) of the steady-state waiting time changes as the traffic rate changes—the variance of this output changes even more (see, e.g., [71], [72], and [404]).

3.4.2 Testing for constant variances

As the previous subsection demonstrated, it may be *a priori* certain that the variances of the simulation outputs are not constant at all. In some applications, however, the analysts may hope that the variances are (nearly) constant. Unfortunately, the variances are unknown so they must be estimated. If there are m_i replicates, then the classic unbiased variance estimator was given in (2.26). This estimator itself has high variance. Using the classic assumption of normally distributed output, any statistics textbook gives

$$var(\widehat{\sigma}^2) = \frac{2\sigma^4}{m}$$

where $(m-1)\widehat{\sigma}^2$ has a chi-square distribution with $m-1$ degrees of freedom. Under the same classic assumption, two independent variance estimators (say) $\widehat{\sigma_1}^2$ and $\widehat{\sigma_2}^2$ may be compared through the F statistic:

$$F_{m_1-1.m_2-1} = \frac{(m_1-1)\widehat{\sigma_1}^2}{(m_2-1)\widehat{\sigma_2}^2}.$$

Actually, there are n ($> k$) combinations of the k factors in the simulation experiment, so n variance estimators $\widehat{\sigma_i}^2$ need to be compared. This problem may be solved in many different ways (my 1987 book [184], p. 225 shows that at that time there were approximately 60 different tests for the same problem). Some examples are:

1. In 1950, Hartley [140] proposed

$$F_{\max} = \frac{\max_{1\leq i\leq n}(\widehat{\sigma_i}^2)}{\min_{1\leq i\leq n}(\widehat{\sigma_i}^2)}. \tag{3.15}$$

2. Scheffé proposes ANOVA, treating the data as a one-way layout (an experiment with a single factor) and n levels. Because the variance estimators have chi-square distributions (whereas ANOVA assumes normality), the analysts may apply a normalizing transformation such as the Box-Cox transformation defined in (3.5). Details are given in [338].

3. Conover gives a distribution-free test; see [81].

Exercise 3.5 *Apply bootstrapping to derive the distribution of Hartley's statistic defined in (3.15) for the following simple case:* $w_{ir} \sim NID(\mu_i, \sigma_i^2)$

$(i = 1, \ldots n; r = 1, \ldots m)$ *with homogeneous variances* $(\sigma_i^2 = \sigma^2)$ *and* $n = 3$. *Design a Monte Carlo experiment with (say)* $\mu_i = 0$ *and* $\sigma_i^2 = 1$, $m = 25$, *and 1,000macro-replicates. Compare the results with Table 31 in [290]. Repeat the experiment for heterogeneous variances* $(\sigma_i^2 \neq \sigma^2)$. *Repeat for nonnormally distributed* w_{ir}.

3.4.3 Variance stabilizing transformations

The logarithmic transformation, which is a special case of the Box-Cox transformation in (3.5), may be used not only to obtain normal output but also to obtain outputs with constant variances. A problem may again be that the metamodel now explains the transformed output instead of the original output.

3.4.4 LS estimators in case of heterogeneous variances

In case of heterogeneous variances, the LS criterion still gives an *unbiased* estimator $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ (vector of regression parameters). To prove this lack of bias, it suffices to assume that the residuals have zero mean, $E(\mathbf{e}) = 0$; see again the solution of Exercise 2.2.

The *variance* of the LS (or OLS) estimator, however, is no longer given by (2.17). Actually, this variance is given by the main diagonal of the covariance matrix that follows from (2.16):

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) = [(\mathbf{X}_N'\mathbf{X}_N)^{-1}\mathbf{X}_N'\mathbf{cov}(\mathbf{w})\mathbf{X}_N(\mathbf{X}_N'\mathbf{X}_N)^{-1}, \tag{3.16}$$

where this time I explicitly show the number of rows $N = \sum_{i=1}^{n}$ of $\mathbf{X}_N$ (obviously, $\mathbf{X}_N$ is an $N \times q$ matrix). This formula deserves the following comments.

- The matrix $\mathbf{cov}(\mathbf{w})$ in the right-hand side is a *diagonal* matrix if the simulation outputs have different variances but are independent (no CRN used).
- If there are *no* replicates, then $\mathbf{X}$ (matrix of explanatory variables) is $n \times q$, so $\mathbf{cov}(\mathbf{w})$ is also an $n \times n$ matrix with the i^{th}element on its main diagonal being $var(w_i)$ $(i = 1, \ldots, n)$.
- If factor combination i is replicated m_i times, then $\mathbf{X}_N$ is $N \times q$ so $\mathbf{cov}(\mathbf{w})$ is also an $N \times N$ matrix with the first m_1 elements on its main diagonal all being equal to $var(w_1)$, ..., the last m_n elements on its main diagonal being $var(w_n)$.
- If the number of replicates is constant $(m_i = m)$, then the LS estimator may be written as

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}} \tag{3.17}$$

where $\mathbf{X}$ is $n \times q$ and $\overline{\mathbf{w}}$ denotes the vector with the n simulation outputs averaged over the m replicates; see $\overline{w_i}$in (2.27) with $m_i = m$.

Exercise 3.6 *Prove that (3.16) (general formula for the covariance matrix of the LS estimator) reduces to the classic formula in (2.16) if* $\mathbf{cov}(\mathbf{w}) = \sigma_w^2 \mathbf{I}$.

In [185], I study confidence intervals for the q individual OLS estimators in (3.17). Their standard errors follow from the following corrected covariance matrix (also see (3.16)):

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) \ = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'[\mathbf{cov}(\overline{\mathbf{w}})]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}. \tag{3.18}$$

Confidence intervals may then be computed through a t statistic with $m-1$ degrees of freedom. In Section 3.5, I shall present an alternative method that does not require the estimation of $\mathbf{cov}(\overline{\mathbf{w}})$ in (3.18) (see (3.32))—but that alternative does require m computations of the OLS estimator. One more alternative is presented in [399].

Though the OLS estimator remains unbiased, it is no longer the BLUE. It can be proven that the BLUE is now the *Weighted LS* (or WLS) estimator

$$\widetilde{\boldsymbol{\beta}} \ = (\mathbf{X}_N^{'}[\mathbf{cov}(\mathbf{w})]^{-1}\mathbf{X}_N)^{-1}\mathbf{X}_N'[\mathbf{cov}_N(\mathbf{w})]^{-1}\mathbf{w}. \tag{3.19}$$

Analogously to (3.17), a constant number of replicates ($m_i = m$) implies that the WLS estimator may be written as

$$\widetilde{\boldsymbol{\beta}} \ = (\mathbf{X}^{'}[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\mathbf{X})^{-1}\mathbf{X}'[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\overline{\mathbf{w}} \tag{3.20}$$

where $\mathbf{X}$ is $n \times q$ (also see (3.2)) and $\mathbf{cov}(\overline{\mathbf{w}}) = \mathbf{cov}(\mathbf{w})/m$ where $\mathbf{cov}(\mathbf{w})$ is $n \times n$. The covariance matrix of this WLS estimator can be proven to be

$$\mathbf{cov}(\widetilde{\boldsymbol{\beta}}) \ = (\mathbf{X}^{'}[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\mathbf{X})^{-1}. \tag{3.21}$$

Furthermore, it can be proven that the WLS estimator can also be computed through classic LS software replacing the original I/O data (x_{ij}, w_i) by $(x_{ij}/\sigma_i, w_i/\sigma_i)$ where σ_i is the standard deviation of w_i ($i = 1, \ldots, N$ and $j = 1, \ldots, q$). Obviously, the transformed outputs have a constant variance, which is equal to one. It can also be proven that WLS minimizes the the sum of squared residuals weighted with $1/\sigma_i^2$.

In practice, $\mathbf{cov}(\mathbf{w})$ is unknown so this covariance matrix must be estimated. I distinguish two types of situations (as I did on page 23):

1. *passive* observation of a real system: no replicates

2. *active* experimentation with either a real system or a simulation model of a real system: replicates.

In situations of type 1, the covariance matrix $\mathbf{cov}(\mathbf{w})$ is estimated from the residuals; see any econometrics textbook or the recent article [132]. In type-2 situations, $var(w_i)$ is estimated from (2.26). I focus on the latter type of situations, because simulation practitioners usually do obtain replicates.

Substituting the estimated response variances into the main diagonal of $\mathbf{cov}(\mathbf{w})$ gives $\widehat{\mathbf{cov}(\mathbf{w})}$. Substituting this estimated covariance matrix into the classic WLS estimation formula (3.19) gives the *Estimated WLS* (EWLS) or Aitken estimator. For a constant number of replicates this EWLS estimator is

$$\widehat{\widehat{\boldsymbol{\beta}}} = (\mathbf{X}^{'}[\widehat{\mathbf{cov}(\overline{\mathbf{w}})}]^{-1}\mathbf{X})^{-1}\mathbf{X}'[\widehat{\mathbf{cov}(\overline{\mathbf{w}})}]^{-1}\overline{\mathbf{w}}. \tag{3.22}$$

Obviously, *this EWLS is not a linear estimator.* Consequently, the statistical analysis becomes more complicated. For example, the covariance matrix of $\widehat{\widehat{\boldsymbol{\beta}}}$ (the EWLS estimator) does no longer follow from (2.16). The analogue of (3.21) holds only asymptotically (under certain conditions); see, e.g., [18], [132], and [198]:

$$\mathbf{cov}(\widehat{\widehat{\boldsymbol{\beta}}}) \approx (\mathbf{X}^{'}[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\mathbf{X})^{-1}. \tag{3.23}$$

Confidence intervals are no longer similar to (2.19). I have already presented relatively simple solutions for this type of problems, namely jackknifing and bootstrapping (see the subsubsections 3.3.3 and 3.3.3). For EWLS these two techniques may be applied as follows.

Jackknifing the EWLS estimator is done by my coauthors and myself in [205]. So we delete the r^{th} replicate among the m IID replicates, and recompute the EWLS estimator (analogous to the jackknifed OLS estimator (3.7)):

$$\widehat{\widehat{\boldsymbol{\beta}_{-r}}} = (\mathbf{X}^{'}[\widehat{\mathbf{cov}(\overline{\mathbf{w}})}_{-r}]^{-1}\mathbf{X})^{-1}\mathbf{X}'[\widehat{\mathbf{cov}(\overline{\mathbf{w}})}_{-r}]^{-1}\overline{\mathbf{w}_{-r}} \; (r = 1, \ldots, m)$$

where $\overline{\mathbf{w}_{-r}}$ consists of the n averages of the $m-1$ replicates after deleting replicate r, and $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}_{-r}$ is computed from the same replicates. From these $\widehat{\widehat{\boldsymbol{\beta}_{-r}}}$ and the original $\widehat{\widehat{\boldsymbol{\beta}}}$ in (3.22) we compute the pseudovalues, which give the desired confidence interval.

Bootstrapping the EWLS estimator is discussed in [200]; also see [132] and [410].

I recommend that analysts compute both the OLS estimate and the EWLS estimate, and check whether these two estimates give the same qualitative conclusions; e.g., which factors are important. EWLS tends to give more significant estimates (because the standard errors tend to be smaller).

3.4.5 Designs in case of heterogeneous variances

If the output variances are not constant, classic designs still give the *unbiased* OLS estimator $\hat{\boldsymbol{\beta}}$ and WLS estimator $\widetilde{\boldsymbol{\beta}}$ for the vector of regression parameters $\boldsymbol{\beta}$. The DOE literature pays little attention to the derivation of alternative designs for cases with heterogeneous output variances. A recent exception is [60], which discusses A-optimal designs for heterogeneous variances; unfortunately, the article considers situations that are not typical for simulation applications (it discusses real-life, chemical experiments).

In a 1995 article [217], Van Groenendaal and I investigate designs in which the n factor combinations are replicated so many times that the estimated variances of the averages per combination are (approximately) constant. The definition of these averages $\overline{w_i}$in (2.27) implies

$$var(\overline{w_i}) = \frac{\sigma_i^2}{m_i} \; (i = 1, \ldots, n).$$

To get a common variance (say) $\sigma^2 = 1/c_0$, the number of replicates should satisfy

$$m_i = c_0 \sigma_i^2 \tag{3.24}$$

where c_0 is a common positive constant such that the m_i become integers. In other words, the higher the variability of the simulation output w_i is, the more replicates are simulated. The allocation of the total number of simulation runs ($N = \sum_{i=1}^{n} m_i$; also see (2.24)) according to (3.24) is not necessarily optimal, but it simplifies the regression analysis and the design of the simulation experiment (an alternative allocation rule replaces the variances σ_i^2 by the standard deviations σ_i); also see [217]. Indeed the regression analysis can now apply OLS to the averages $\overline{w_i}$ to get the BLUE.

In practice, however, the variances of the simulation outputs must be estimated. A *two-stage* procedure takes a *pilot sample* of size (say) $m_0 \geq 2$ for each factor combination, and estimates the variances σ_i^2 through

$$s_i^2(m_0) = \frac{\sum_{r=1}^{m_0} \left[w_{ir} - \overline{w_i}(m_0)\right]^2}{m_0 - 1} (i = 1, \ldots n) \tag{3.25}$$

with

$$\overline{w_i}(m_0) = \frac{\sum_{r=1}^{m_0} w_{ir}}{m_0}. \tag{3.26}$$

Combining (3.25) and (3.24), Van Groenendaal and I propose in [217] to select additional replicates $\widehat{m_i} - m_0$ (in the second stage) where

$$\widehat{m_i} = m_0 \left\lfloor \frac{s_i^2(m_0)}{\min_{1 \leq i \leq n} s_i^2(m_0)} \right\rfloor \tag{3.27}$$

with $\lfloor x \rfloor$ denoting the integer closest to x (so, in the second stage no additional replicates are simulated for the combination with the smallest estimated variance). After the second stage, all $\widehat{m_i}$ replicates are used to estimate the average output and its variance. OLS is applied to these averages. We estimate the covariance matrix $\mathbf{cov}(\widehat{\boldsymbol{\beta}})$ through (3.16) with $\mathbf{cov}(\mathbf{w})$ estimated through a diagonal matrix with diagonal elements $s_i^2(\widehat{m_i})/\widehat{m_i}$. We base the confidence intervals for the estimated regression parameters on the classic t statistic with degrees of freedom equal to $m_0 - 1$.

After the second stage these variance estimates $s_i^2(\widehat{m_i})/\widehat{m_i}$ may still differ considerably. Therefore, the two-stage approach may be replaced by a (purely) *sequential* approach. The latter approach adds one replicate at a time, until the estimated variances of the average simulation outputs have become practically constant; for details see [217] The sequential procedure requires fewer simulation responses, but is harder to understand, program, and implement.

Exercise 3.7 *Simulate the M/M/1 model, as follows. Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time. Select an experimental area; e.g., the traffic load is 0.3 and 0.5. Fit a first-order polynomial. Use m_i replicated simulation runs; each run should be "sufficiently long". Simulate more replicates for the higher traffic rate, using (3.27). Do not apply CRN for different traffic rates. Now estimate the parameters of your metamodel and the simulation output at a 0.4 traffic load including a confidence interval; does this interval cover the analytical solution? Also see Exercise 2.15.*

3.5 Common random numbers (CRN)

The use of CRN creates correlation between the simulation outputs $w_{i;r}$ and $w_{i';r}$ with $i, i' = 1, \ldots, n$; $r = 1, \ldots, m$; $m = \min_i m_i$ (obviously, individual simulation output data—such as individual waiting times—are autocorrelated or serially correlated, within a simulation run for a given factor combination). In Figure 3.2, I display the $n \times q$ matrix of explanatory variables $\mathbf{X}$ and the two vectors of simulation outputs $\mathbf{w}_r$ and $\mathbf{w}_{r'}$ with $r, r' = 1, \ldots, m$. I assume that two different replicates use nonoverlapping PRN streams, so their outputs $w_{i;r}$ and $w_{i';r}$ with $r \neq r'$ are independent (i.e., I assume that no Antithetic Random Numbers or ARN are used).

The goal of CRN is to reduce the variance of the estimated regression effects; i.e., to decrease $var(\widehat{\beta_j})$ with $j = 1, \ldots, q$. Actually, the variance of the intercept increases when CRN are used.

Exercise 3.8 *Prove that $var(\widehat{\beta_1})$ increases if CRN are used and β_1 denotes the intercept (assume that no replicates are used and that CRN does "work"; i.e., $cov(w_{i;i'}) > 0$).*

So CRN may be useful to better explain the factor effects, as scenarios are compared under the "same circumstances". CRN is also useful to better predict the output of combinations not yet simulated, provided the lower accuracy of the estimated intercept is outweighed by the higher accuracy of all other estimated effects.

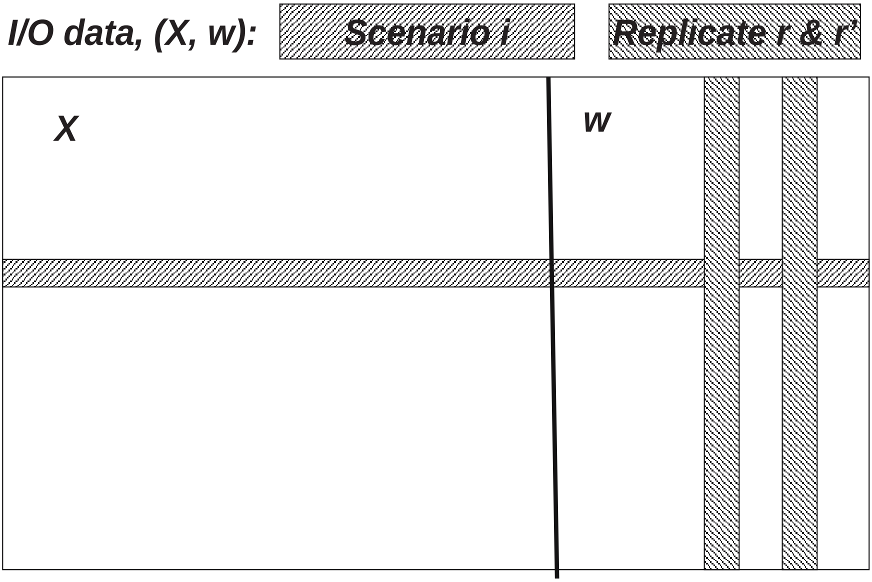

Figure 3.2: Common Random Numbers

I again try to answer the following questions (formulated more generally in Section 3.1):

1. How *realistic* is it to assume the use of CRN?
2. Which statistical *analysis* methods can be applied that allow CRN?
3. Which statistical *design* methods can be applied to account for CRN?

3.5.1 Realistic CRN assumption?

Obviously, simulation analysts apply CRN in *random* simulation only. In practice, the analysts use CRN very often; actually, CRN is the *default* of much simulation software (e.g., Arena starts with the same PRN seed—unless the users change the seed).

3.5.2 Alternative analysis methods

Because CRN violates the classic assumptions of regression analysis, the analysts have two options (which are analogous to the options in the case of heterogeneous output variances):

1. Continue to use OLS
2. Switch to GLS

OLS

The variance of the OLS estimator is similar to (3.16), but now $\mathbf{cov}(\mathbf{w})$ is not a diagonal matrix:

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) \ = (\mathbf{X}_N'\mathbf{X}_N)^{-1}\mathbf{X}_N'\mathbf{cov}(\mathbf{w})\mathbf{X}_N(\mathbf{X}_N'\mathbf{X}_N)^{-1}, \tag{3.28}$$

which for $m_i = m$ (the usual situation in case of CRN) becomes

$$\mathbf{cov}(\hat{\boldsymbol{\beta}}) \ = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{cov}(\overline{\mathbf{w}})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \tag{3.29}$$

with $n\times q$ matrix $\mathbf{X}$. The $n\times n$ covariance matrix $\mathbf{cov}(\overline{\mathbf{w}})$ may be estimated by $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$ with the elements (also see (2.26) and (2.53))

$$\widehat{cov(\overline{w}_i, \overline{w}_{i'})} = \frac{\sum_{r=1}^{m}(w_{i;r} - \overline{w_i})(w_{i';r} - \overline{w_{i'}})}{(m-1)m}. \tag{3.30}$$

This $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$ is *singular* if the number of replicates is “too small”; i.e., if $m \leq n$; see [102].

In [185], I show that confidence intervals for the q individual OLS estimators may be computed from a t statistic with $m-1$ degrees of freedom—provided $m > n$. In this t statistic, the standard errors $s(\widehat{\beta_j})$ are the square roots of the elements on the main diagonal of the corrected covariance matrix in (3.29).

An *alternative* method does not require the estimation of $\mathbf{cov}(\overline{\mathbf{w}})$ to derive confidence intervals for the OLS estimators, so it suffices that $m > 1$. This alternative requires m computations of the OLS estimator; i.e., from replicate r, the analysts estimate the (true) vector of regression parameters $\boldsymbol{\beta}$ through

$$\widehat{\boldsymbol{\beta}_r} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}_r \ (r = 1, \ldots, m). \tag{3.31}$$

The n elements of the vector $\mathbf{w}_r$ are correlated (because they use CRN) and may have different variances. The m estimators of a specific regression parameter β_j, however, are independent (because they use nonoverlapping PRN streams) and have a common standard deviation (say) $\sigma(\widehat{\beta_j})$. Therefore (2.19) is replaced by

$$t_{m-1} = \frac{\overline{\widehat{\beta}_j} - \beta_j}{s(\overline{\widehat{\beta_j}})} \text{ with } j = 1, \ldots, q \tag{3.32}$$

with

$$s(\overline{\widehat{\beta_j}}) = \sqrt{\frac{\sum_{r=1}^{m}(\widehat{\beta_{j;r}} - \overline{\widehat{\beta}_j})^2}{m(m-1)}}.$$

Note: Law’s textbook [227], p. 627 inspired me to derive confidence intervals for regression coefficients estimated with CRN and multiple replicates through (3.32).

GLS

It can be proven that CRN implies that the BLUE is not the OLS but the GLS estimator, which is analogous to (3.19):

$$\widetilde{\boldsymbol{\beta}} = (\mathbf{X}^{'}[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\mathbf{X})^{-1}\mathbf{X}'[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\overline{\mathbf{w}}). \qquad (3.33)$$

The covariance matrix of the GLS estimator is analogous to (3.21):

$$\mathbf{cov}(\widetilde{\boldsymbol{\beta}}) = (\mathbf{X}^{'}[\mathbf{cov}(\overline{\mathbf{w}})]^{-1}\mathbf{X})^{-1}. \qquad (3.34)$$

Again, in practice $\mathbf{cov}(\overline{\mathbf{w}})$ is unknown so it must be estimated. The matrix $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$ has the elements given by (3.30). This matrix is *singular* if the number of replicates is "too small"; i.e., if $m \leq n$.

Substituting $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$ (estimated covariance matrix) into the classic GLS estimation formula (3.33) gives *Estimated GLS* (EGLS), which is analogous to EWLS in (3.22). The EGLS estimator can again be analyzed through jackknifing and bootstrapping. In [185], however, I compare OLS and EGLS relying on the asymptotic covariance matrix of the EGLS estimator—given by (3.23) with nondiagonal $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$. However, [89] claims that "bootstrap tests ... yield more reliable inferences than asymptotic tests in a great many cases."

In conclusion, CRN with EGLS may give better point estimates of the factor effects than CRN with OLS, but the EGLS estimate requires "many" replicates—namely $m > n$—to obtain a nonsingular $\widehat{\mathbf{cov}(\overline{\mathbf{w}})}$.

Exercise 3.9 *Simulate the M/M/1 model, as follows. Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time. Select an experimental area; e.g., the traffic load is 0.3 and 0.5. Each run should be "sufficiently long". Apply CRN for the different traffic rates. Use m replicated simulation runs; vary the number of replicates between its minimum 2 and (say) 10. Fit a first-order polynomial. Now estimate the parameters of your metamodel, including a confidence interval for the interpolated output at a traffic rate of (say) 0.4; does this interval cover the analytical solution? Also see Exercise 3.7.*

3.5.3 Designs in case of CRN

The literature pays no attention to the derivation of alternative designs for cases with CRN. (In [343], Schruben and Margolin do discuss how CRN and ARN should be combined if classic designs are applied; for an update see [96].)

In two recent papers [213] and [384], Van Beers and I propose *sequential* procedures to select the next factor combination to be simulated. We allow the simulation model to be either deterministic or random. However, we assume that the simulation I/O data $(\mathbf{D}, \mathbf{w})$ are analyzed through a Kriging metamodel, which allows the simulation outputs of different factor combinations to be correlated; see Chapter 5.

3.6 Nonvalid low-order polynomial metamodel

In this section, I try to answer the following questions (again, these questions were formulated more generally in Section 3.1):

1. How can the validity of the low-order polynomial metamodel be tested?
2. If this metamodel is not valid, how can the simulation's I/O data be transformed such that a low-order polynomial becomes valid ?
3. Which alternative metamodels can be applied?

3.6.1 Testing the validity of the metamodel

Definition 2.4 defined a valid metamodel as a metamodel with zero mean residuals: $E(e) = 0$. To test this assumption, the classic *lack-of-fit F-statistic* was defined in (2.29) and (2.30) for white-noise situations.

If the analysts apply CRN, then they may apply the following variant of (2.29) derived by Rao in 1959 (see [309] and also [185]):

$$F_{n-q;m-n+q} = \frac{m-n+q}{(n-q)(m-1)}(\overline{\mathbf{w}} - \widehat{\widehat{\mathbf{y}}})'[\widehat{\mathbf{cov}(\overline{\mathbf{w}})}]^{-1}(\overline{\mathbf{w}} - \widehat{\widehat{\mathbf{y}}}) \qquad (3.35)$$

where $n > q$, $m > n$, and $\widehat{\widehat{\mathbf{y}}}$ denotes the EGLS estimator. Obviously, Rao's test also allows EWLS instead of EGLS. Normality of the simulation output is an important assumption for both the classic test and Rao's test. In case of nonnormality, the analysts may apply jackknifing or bootstrapping. Deflandre and I bootstrap Rao's statistic and the classic R^2 statistic in [200].

Note: If the number of replicates tends to infinity, then both the classic test and Rao's test converge in distribution to $\chi^2_{n-q}/(n-q)$ if the metamodel is valid.

An alternative test uses *cross-validation*; see (2.64). Now the OLS estimator $\widehat{y}$ may be replaced by the estimator $\widehat{\widehat{y}}$ to account for EWLS or EGLS estimation. The t statistic is less sensitive to nonnormality than the F statistic is; see the extensive Monte Carlo study in my 1992 article [185]. Moreover, this t statistic requires fewer replications, namely $m > 1$ instead of $m > n$ if OLS or EWLS is used (see the discussion of (3.30)).

Besides these quantitative tests, the analysts may use *graphical* methods to judge the validity of a fitted metamodel (be it a linear regression model or some other type of metamodel such as a Kriging model). Scatterplots have already been discussed in Section 2.11.2. In the 2002 paper [355], a panel also emphasizes the importance of visualization; also see [145]. (Recently, [138] proposed three other two-dimensional plots for judging the validity of metamodels in deterministic simulation—but I do not know any applications of these plots, except for the one application given in that paper itself.)

If these validation tests give significant approximation errors, then the analysts may consider the following alternatives.

3.6.2 Transformations of independent and dependent regression variables

In (2.9), I have already demonstrated that a transformation that combines two simulation inputs (arrival rate λ and service rate μ) into a single independent regression variable ($x = \lambda/\mu$) may give a better metamodel. Another useful transformation already discussed in (2.7) replaced y, λ, and μ by $\log(y)$, $\log(\lambda)$, and $\log(\mu)$, to make the first-order polynomial approximate relative changes.

Another simple transformation assumes that the I/O function of the underlying simulation model is *monotonic*. Then it makes sense to replace the dependent and independent variables by their ranks, which results in so-called *rank regression*; see Conover and Iman's 1981 article [82]; also see [327] and [328].

Note: Spearman's correlation coefficient also uses the rank transformation, but for only two correlated random variables; see Example 2.7. In [204], Helton and I use Spearman's coefficient and rank regression to find the most important factors in a simulation model of nuclear waste disposal; also see Example 2.7.

Transformations may also be applied to make the simulation output (dependent regression variable) better satisfy the assumptions of normality (see (3.5)) and variance homogeneity. Unfortunately, different goals of the transformation may conflict with each other; e.g., the analysts may apply the logarithmic transformation to reduce nonnormality, but this transformation may give a metamodel in variables that are not of immediate interest.

3.6.3 Adding high-order terms to a low-order polynomial metamodel

In the preceding chapter, I discussed designs with different resolutions. Resolution-III designs assume first-order polynomial metamodels, whereas resolution-IV and resolution-V designs assume two-factor interactions. Moreover, I discussed designs for second-order polynomials, especially CCDs. If these designs do not give valid metamodels, then I recommend to look for *transformations,* as discussed in the preceding subsection. I do not recommend routinely adding higher-order terms to the metamodel, because these terms are hard to interpret. However, if the goal is not to better *understand* the underlying simulation model but to better *predict* the output of an expensive simulation model, then high-order terms may be added. Indeed, (full factorial) 2^k designs enable the estimation of all interactions (e.g., the interaction among all k factors). If more than two levels per factor are simulated, then other types of metamodels may be considered (see the next subsection).

3.6.4 Nonlinear metamodels

On page 8, I have already mentioned several alternative metamodel types (e.g., Kriging models). These alternatives may give better predictions than low-order polynomials do. However, these alternatives are so complicated that they do not help the analysts better understand the underlying simulation model—except for sorting the simulation inputs in order of their importance.

Furthermore, these alternative metamodels require *alternative design types.* I shall discuss Latin Hypercube Sampling in section 4.5.1.

3.7 Conclusions

In this chapter, I discussed the assumptions of classic linear regression analysis and the concomitant statistical designs (detailed in the preceding chapter) when these methods are applied in simulation practice. I pointed out that multiple simulation outputs can still be analyzed through OLS per output type. I addressed possible nonnormality of simulation output, including normality tests, normalizing transformations of simulation I/O data, and the distribution-free methods called jackknifing and bootstrapping. I presented analysis and design methods for simulation outputs that do not have a common variance. I discussed how to analyze simulation I/O data that uses CRN, so the simulation outputs are correlated (across different factor combinations, within the same replicate). I discussed possible lack-of-fit tests for low-order polynomial metamodels, transformations to improve the metamodel's validity, and alternative metamodels and designs. Throughout this chapter, I gave many references for further study of these issues. (Paraphrasing the old saying "Crime does not pay", I now claim: "but assumptions do".)

3.8 Solutions for exercises

Solution 3.1 *The jackknife results for this M/M/1 simulation depend on the PRN stream; see [199] for examples.*

Solution 3.2 *The jackknife results for this Monte Carlo experiment depend on the PRN stream; see [216], pp. 141–146 and also [185] and [205] for examples.*

Solution 3.3 *The bootstrap results for this M/M/1 simulation depend on the PRN stream; see [199] for an example.*

Solution 3.4 *The bootstrap results for this Monte Carlo experiment depend on the PRN stream; see [200] for examples.*

Solution 3.5 *See Section 3.3.3 on bootstrapping.*

Solution 3.6 *If* $\mathbf{cov}(\mathbf{w}) = \sigma_w^2\mathbf{I}$*, then* $\mathbf{cov}(\hat{\boldsymbol{\beta}}) =$
$(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{cov}(\mathbf{w})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} =$
$= \sigma_w^2(\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1} = \sigma_w^2(\mathbf{X}'\mathbf{X})^{-1}$.

Solution 3.7 *The results for this M/M/1 simulation depend on the specific PRNs, etc.*

Solution 3.8 *Let the intercept be estimated through* $\widehat{\beta_1} = \sum_{i=1}^{n} w_i/n = \mathbf{1}'\mathbf{w}/n$ *with* $\mathbf{1}' = (1, \ldots, 1)$ *assuming no replicates. Then* $var(\widehat{\beta_1}) = \mathbf{1}'\mathbf{cov}(\mathbf{w})\mathbf{1}/n^2 = \sum_{i=1}^{n}\sum_{i'=1}^{n} cov(w_{i;i'})/n^2$*, which increases if CRN "works"; i.e.,* $cov(w_{i;i'}) > 0$*.*

Solution 3.9 *The results for this M/M/1 simulation depend on the specific PRNs, etc.*

4

Simulation optimization

This chapter is organized as follows. In Section 4.1, I introduce the central issues in simulation optimization. In Section 4.2, I summarize classic Response Surface Methodology (RSM), and the Adapted Steepest Ascent (ASA) search direction (developed by my coauthors and myself in [201]). In Section 4.3, I summarize Generalized RSM (GRSM) for simulation with multiple responses (developed by my coauthors and myself in [12]). In Section 4.4, I summarize a procedure for testing whether an estimated optimum is truly optimal—using the Karush-Kuhn-Tucker (KKT) conditions (developed by coauthors and myself in [13] and [39]). In Section 4.5, I discuss Risk Analysis (RA), also called Uncertainty Analysis (UA). In Section 4.6, I explore Robust Optimization. In Section 4.7, I present conclusions. I finish with solutions for the exercises of this chapter.

4.1 Introduction

The practical importance of *optimizing* engineered systems (man-made artifacts) is emphasized in the 2006 NSF report on simulation-based engineering, [280] (also see [315]). That report also emphasizes the crucial role of *uncertainty* in the input data for simulation models, which—in my opinion—implies that *robust* optimization is important. An essential difference with Mathematical Programming (MP) models is that in simulation models the objective function (which is the function to be minimized or maximized) is not known explicitly; actually, this function is defined implicitly by the simulation model (computer code, computer program). Moreover, in random simulation these functions give random outputs, which only estimate the true outputs (e.g., the mean or 90% quantile). The academic importance of simulation optimization is demonstrated by the many sessions on this topic at the yearly Winter Simulation Conferences; see http://www.wintersim.org/.

The simplest optimization problem has no constraints for the input or output, no uncertain environmental variables, and concerns the expected value of a single (univariate) simulation output. This expected value may also represent the probability of a binary variable having the value one (use the indicator function). The expected value, however, excludes quantiles (such as the median and the 95% quantile or percentile) and the mode of the output distribution. Furthermore, the simplest problem assumes that the inputs are continuous variables (not discrete or nominal; see the various scales discussed in Section 1.3).

The assumption of continuous inputs implies that there is an infinite number of systems, so *Ranking and Selection* (R & S) procedures and *Multiple Comparison Procedures* (MCPs) are excluded (since these procedures assume a limited fixed set of competing systems). R & S procedures use the so-called indifference zone approach. Recent discussions of these R & S procedures are [180], [295], and also part of the tutorial review of simulation optimization in [118]. A historical survey is included in [374]. An older discussion of R & S and MCPs is given in my 1974/1975 book, [181]. Bootstrapping of R & S procedures is discussed in [68]. Related to R & S is *Ordinal Optimization* (OO); see [7], [148], [163], and [285]. Personally, I have never applied any of these procedures in practice, so I do not give more details.

An academic example of a simple optimization problem is an (s, Q) inventory management simulation where (say) w_0 is the total inventory costs (the sum of inventory carrying, reorder, and out-of-stock costs), and the decision variables are the reorder level $z_1 = s$ and the order quantity $z_2 = Q$. (Implicit input constraints are that these two decision variables must be nonnegative; the symbols w_0, z_1, and z_2 are also used in the general problem formulation later in this chapter.)

In practice, however, simulation models have *multiple outputs*. Examples are many practical inventory models that require the inventory system to satisfy a minimum service rate (or fill rate), because the out-of-stock costs are hard to quantify (an example is [162]). I shall formalize this type of problems in Section 4.3.

There are *many different methods* for simulation optimization; i.e., there are many approaches to the estimation of the optimal solution of a simulated system; see [25], [118], [146], and [377]. In Chapter 1, I pointed out that a simulation model can be either random or deterministic. In the present chapter, I focus on random simulation (but I shall briefly mention deterministic simulation).

Numerous methods have been developed to optimize real, nonsimulated systems or to find the optimal solution of a mathematical model (such as a Linear Programming or LP model). Many of these optimization methods can also be applied to simulated systems; see [116], [360], and also [137].

Simulation optimization methods may be classified as either *black-box* or *white-box* methods. By definition, black-box methods observe only the inputs and outputs of the simulation model—be it a random or a deterministic model. White-box methods use the explicit mathematical functions inside the simulation model, to estimate gradients (these gradients are used to estimate the optimum; see below). Examples of white-box methods are methods that use Perturbation Analysis (PA) and Score Function (SF) methods. For details, I refer to [117], [130], [142], [147], [264], [319], and [377] (also see my discussion of (2.3) and (2.4)).

Examples of black-box methods are *metaheuristics*; e.g., (Artificial) Ant Colonies, Evolutionary Algorithms (EAs) and the related Genetic Algorithms (GAs), Scatter Search, Simulated Annealing (SA, not to be confused with Stochastic Approximation), and Tabu Search (TS). These methods can also be used to optimize a simulated system. They are global search methods; i.e., they are meant to escape from local optima. Recent overviews are [1], [8], [27], [55], [78], [99], [150], [245], [281], [306], [324], [371], [377], and [406]. For Tabu Search, I also refer to the web

http://www.tabusearch.net/.

Other black-box methods are Spall's *Simultaneous Perturbation Stochastic Approximation (*SPSA; see [24], [119], [359], [360], and
http://www.jhuapl.edu/SPSA/,
Rubinstein's *Cross Entropy* (see [318] and
http://iew3.technion.ac.il/CE/about.php,
and Wieland and Schmeiser's method based on the Control Variates (CV) Variance Reduction Technique (VRT); see [400]. SPSA requires the simulation of only two input combinations to estimate the gradient—whatever the number of inputs is; an application of SPSA is [344]. The CV-based method requires a single input combination, but that combination must be replicated.

Pitchitlamken and Nelson's procedure in [294] combines *Nested Partitioning* (see [350]) and *Ranking and Selection* (R & S). The former component is meant to avoid getting trapped in a local optimum.

Instead of applying a simulation optimization method to the simulation model itself, the method may be applied to a metamodel based on the simulation model at hand. Any of the metamodeling methods for simulation mentioned in Chapter 1 may be combined with an optimization method. If the metamodel is explicit (e.g., the locally or globally fitted second-order polynomials in Chapter 2, the Kriging metamodels in Chapter 5), then Nonlinear Mathematical Programming (NMP) may be applied; see the examples in [42] and [112]. Otherwise (e.g., support vector regression models), some search method (e.g., a crude grid search, a genetic algorithm, a gradient based method) may be used; see the examples of spline metamodels in [25] and [175]. Recently, Huyet ([159]) applied data mining and machine learning to simulated systems, in order to both optimize (via evolutionary algorithms) and obtain insight (via so-called induction graphs). Also see [25] and [133].

Note: Recently, [361] studied the convergence rates and efficiency of a few stochastic optimization approaches, in a monograph that contains many more mathematically advanced contributions to Robust Optimization.

Software for optimization based on various types of metamodels is provided by Sandia's DAKOTA, which stands for Design Kit for Optimization and Terascale Applications; see [129] and the website

http://endo.sandia.gov/DAKOTA.

Software for the optimization of System Dynamics models includes DYSMOD's pattern search; see [88]. More references will follow below.

Note: In theory, the analysts may apply several types of optimization methods, checking whether the estimated optima differ really. In practice, however, such a combined approach is rare, because the analysts are familiar with one or two optimization methods only.

In summary, there is a bewildering number of methods for simulation optimization. Nobody seems expert in more than a few methods. In this chapter, I focus on the black-box method known as RSM, which Box and Wilson published in 1951; see [51]. This method is often ignored in the literature on metaheuristics. Nevertheless, RSM is often applied in real-life experiments; see Section 4.2 and also the Design-Expert software detailed in

www.statease.com.

Note: Some authors outside the discrete-event simulation area speak of RSM, but mean what I call the what-if regression-metamodeling approach, not the sequential (iterative) optimization approach; see, e.g., [97] and [284]. Other authors speak of RSM, but use global Kriging metamodels instead of local low-order polynomials; see [168].

In this chapter, I further focus on *expensive* simulation, meaning that it takes much computer time to compute a single realization of the time path of the simulated system. For example, 36 to 160 hours of computer time are needed to simulate a crash model at Ford Motor Company; see the panel discussion reported in [355]. This panel also reports the example of a (so-called "cooling") problem with 12 inputs,10 constraints, and 1 objective function. For such expensive simulations, many simulation optimization methods are unpractical. An example is the popular software called OptQuest, which combines Tabu Search, Neural Networks, and Scatter Search; it is an add-on to simulation software such as Arena, CrystallBall, MicroSaint, ProModel, and Simul8; see [118] and also [1], [27], [150], and [306]. OptQuest requires relatively many simulation replicates and factor combinations; see the inventory example in my recent publications with Wan, [219] and [395]. Genetic Algorithms are also unpractical for expensive simulations; see [108], [153], and [312]. Fortunately, the mathematical and statistical computations required by the RSM and GRSM search heuristics are negligible—compared with the computer time required by the "expensive" simulation runs.

4.2 RSM: classic variant

Classic RSM started with the 1951 article by Box and Wilson, [51] (the origin of RSM is nicely discussed by Box himself in [47]). Those authors search for the combination of quantitative "decision variables" (inputs, factors) that minimizes the univariate output of a *real-world* system (or maximizes that output: simply add a minus sign in front of the output, before minimizing it). Numerous applications are given in an excellent survey that covers the period 1966–1988; see [267], pp. 147–151. Recent textbooks on classic RSM are [177] and [268]. Recent software for classic RSM is Minitab's "Response Optimizer" and Stat-Ease's "Design-Ease" and "Design-Expert". (I also refer back to Sections 1.2 and 2.10.)

This classic RSM has also been applied to *simulation*—be it random or deterministic. I refer to several of my own publications; e.g., my 1998 survey [191], my 1993 case study [186], my 1981 case study with several coauthors [206], the extensive discussion in my 1974/1975 textbook [181]. Recent case studies of RSM optimization of random simulations by other authors are [25], [26], [161], [257], [276], [316], and [407]. A case study of RSM for deterministic simulation is [33]. RSM is also discussed in the classic simulation textbook by Law,[227], pp. 646–655, and survey articles such as [25] and [377]. Unfortunately, RSM (unlike Tabu Search and other alternatives) has not yet been implemented as an add-on to any of the Commercial Off The Shelf (COTS) simulation software packages. (Nevertheless, in 2007 Google gives nearly seventeen million hits for "Response Surface Methodology".)

As I have already stated in Section 4.1, the simplest optimization problem has no constraints and no uncertain environmental variables, and concerns the expected value of a univariate simulation output (also see (2.6)):

$$\min_{\mathbf{z}} E(w_0|\mathbf{z}) \tag{4.1}$$

where

$\mathbf{z} = (z_1, \ldots, z_k)'$ where z_j $(j = 1, \ldots k)$ denotes the j^{th} original (non-standardized) decision variable of the simulation program**;**

$E(w_0|\mathbf{z})$ is the goal (or objective) output of the simulation model, which is to be minimized through the choice of $\mathbf{z}$:

$$E(w_0|\mathbf{z}) = \int_0^1 \cdots \int_0^1 s(\mathbf{z}, \mathbf{r}) d\mathbf{r}$$

where $s(\mathbf{z}, \mathbf{r})$ denotes the computer simulation program, which is a mathematical function that maps the inputs $\mathbf{z}$ and the PseudoRandom Numbers (PRNs) $\mathbf{r}$ to the random simulation response (output) w_0.

Classic RSM (applied to real-world or simulated systems) has the following *characteristic*s.

1. RSM is an *optimization heuristic* that tries to estimate the input combination that minimizes a given goal function, like the one in (4.1) above. Because RSM is a heuristic, success is not guaranteed (see below).

2. RSM is a *stepwise* (multi-stage) method.

3. In these steps, RSM uses first-order and second-order *polynomial* regression (meta)models (response surfaces) locally. RSM assumes that the responses have white noise locally, so Ordinary Least Squares (OLS) gives the Best Linear Unbiased Estimator (BLUE); see Chapters 2 and 3.

4. To fit (estimate, calibrate) these first-order polynomials, RSM uses *classic designs* of resolution III; for the second-order polynomial, RSM uses a Central Composite Design (CCD); details on these designs were given in Chapter 2.

5. To determine in which direction the factors will be changed in a next step, RSM uses the *gradient* that is implied by the first-order polynomial fitted in the current step. This gradient is used in the mathematical (not statistical) technique of *steepest descent* (or steepest ascent, in case the output is to be maximized, not minimized).

6. In the final step, RSM applies the mathematical technique of *canonical analysis* to the second-order polynomial metamodel, to examine the shape of the optimal (sub)region: does that region have a unique minimum, a saddle point, or a ridge (stationary points)?

More specifically, I distinguish the following eight *steps*; also see Figure 4.1 (a more detailed description is given in [25]).

1. The analysts begin by selecting a *starting point*; see the point (0) in Figure 4.1 They may select the factor combination currently used in practice if the simulated system already exists. Otherwise, they should use intuition and prior knowledge (as in many other search strategies).

2. The analysts explore the Input/Output (I/O) behavior of the simulated system in the *neighborhood* of this starting point; see the dotted square with the point (0) in the lower-left. They approximate this behavior through a local first-order polynomial (as the Taylor series expansion suggests). Hence they need to estimate the intercept β_0 and the k main effects β_j with $j = 1, \ldots, k$. Therefore they use a resolution-III design. Unfortunately, there are no general guidelines

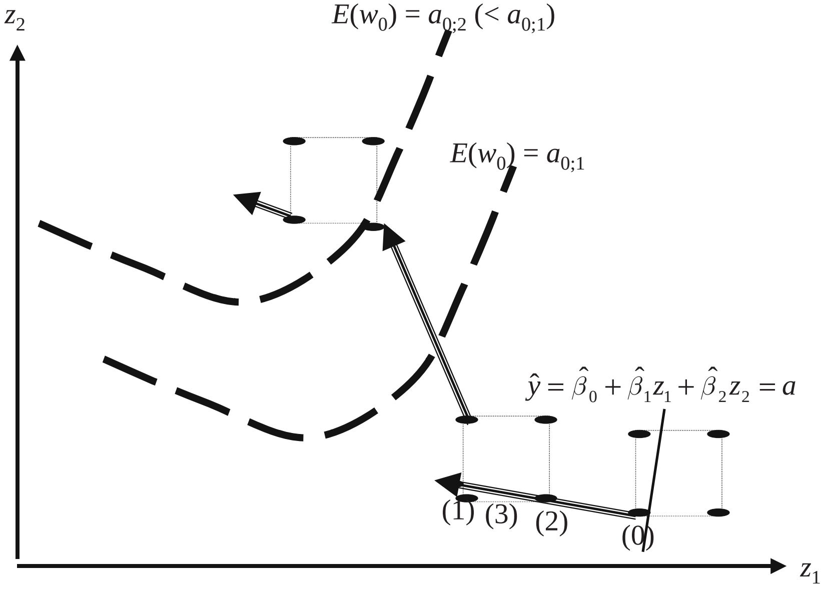
z_2
$E(w_0) = a_{0;2}$ ($< a_{0;1}$)
$E(w_0) = a_{0;1}$
$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 z_1 + \hat{\beta}_2 z_2 = a$
(1)
(3)
(2)
(0)
z_1

Figure 4.1: RSM example

to determine the appropriate size of this local area; intuition and prior knowledge are again important. (Finite differencing—which replaces the resolution-III design by a less efficient one-factor-at-a-time design—also faces the problem of selecting an appropriate size for the local area; the optimal size depends on the unknown variance and second-order derivatives; see [53], [327], and [412].)

3. To decide on the next input combination to be explored by simulation, the analysts follow the *steepest descent path*, which uses the local gradient. For example, if the estimated local first-order polynomial is $\widehat{y} = \widehat{\beta_0} + \widehat{\beta_1} z_1 + \widehat{\beta_2} z_2$, then a corresponding contour line is $\widehat{y} = a$ where a denotes some constant (if the goal output w_0 denotes costs, then the contour is also called the iso-costs line). The steepest descent path is *perpendicular* to the local contour lines. This path implies that if $\widehat{\beta_1} >> \widehat{\beta_2}$, then z_1 is decreased much more than z_2. (Special designs have been developed to estimate the slope accurately; i.e., these designs may replace the classic resolution-III designs; see [267], pp. 142–143. However, D-optimal designs seem good enough to estimate the steepest descent path; see [375], p. 121.) Unfortunately, the steepest descent method is *scale dependent* (see [268], pp. 218–220). Fortunately, in [202] and [201] my coauthors and I present a scale-independent variant, which I shall summarize at the end of this section.

4. Unfortunately, the steepest descent technique does not quantify the *step size* along its path. The analysts may therefore try some value for the step size; see point (1) in Figure 4.1. If that value yields an inferior simulation output (i.e., a significantly higher instead of lower output), then they may reduce the step size; e.g., halve the step size as in point (2). In the figure, point (2) turns out to be better than the point (0), so the step size is again increased to point (3). In the figure it turns out that the best point along the steepest descent path is (2), after all. (There are more sophisticated mathematical procedures for selecting step sizes; see [323] and Section 4.3 below.)

5. The preceding step illustrates that after a number of steps along the steepest descent path, the simulation output will *deteriorate* (i.e., increase instead of decrease), because the first-order polynomial is only a local approximation of the implicit I/O function defined by the simulation model itself. When this deterioration happens, the analysts explore the subarea around the best point found so far; i.e., they simulate the $n > k$ factor combinations specified by a *resolution-III design* centered around the current best point. So the analysts may use the same coded design as in step 2, but translate that design into different values for the original variables (the current best combination may be one of the corners of the design—like the lower-right corner denoted by (2) of the next square in Figure 4.1). Next the analysts estimate the first-order effects in the new local polynomial approximation And so their search continues; see the other two arrows in the figure.

6. However, it is intuitively clear that a *plane* (implied by the most recent local first-order polynomial) cannot adequately represent a *hill top* (when searching for the maximum; the analogue holds for the minimum). So in the neighborhood of the optimum, a first-order polynomial shows serious *lack of fit.* A popular and simple diagnostic measure is the coefficient of determination R^2 (see Section 2.11.1). A related diagnostic tests whether all estimated first-order effects (and hence the gradient) are zero (see equation 2.22 and the KKT test in (4.4) below). Instead of these diagnostics, the analysts might use cross-validation (see Section 2.11.2). If the most recently fitted first-order polynomial turns out to be inadequate, then the analysts fit a *second-order polynomial.* To estimate this metamodel, they may simulate the combinations specified by a CCD (see Section 2.9). This is not shown in the figure.

7. From this second-order polynomial, the analysts estimate the optimal values of the decision variables by straightforward *differentiation* or by more sophisticated *canonical analysis* to examine the shape of the optimum; see [268] p. 208.

8. If time permits, then the analysts may try to escape from a local minimum and *restart* their search from a different initial local area—which brings them back to Step 1; also see [274].

Exercise 4.1 *Apply RSM to the following problem:*

$$min_{\mathbf{z}} E[5(z_1-1)^2 + (z_2-5)^2 + 4z_1z_2 + e]$$

where $e \backsim N(0,1)$*. RSM treats this example as a black box; i.e., you select the input combination* $\mathbf{z}$*, sample* e*, and use these input data to compute the output (say)* w*. You (not RSM) may use the explicit function to derive the true optimum solution,* (z_1^o, z_2^o)*.*

In step 3, I mentioned that my coauthors and I give a variant of steepest descent in [201] and [202] (also see [291]). I summarize our results as follows. *Adapted Steepest Descent* (ASD) accounts for the covariances between the elements of the estimated gradient $\widehat{\boldsymbol{\beta}_{-0}} = (\widehat{\beta_1}, \ldots, \widehat{\beta_k})'$, where the subscript -0 means that the intercept $\widehat{\beta_0}$ of the estimated first-order polynomial vanishes in the estimated gradient so $\widehat{\boldsymbol{\beta}} = (\widehat{\beta_0}, \widehat{\boldsymbol{\beta}_{-0}})'$.

This $\mathbf{cov}(\widehat{\boldsymbol{\beta}_{-0}})$ follows from the (classic) white noise assumption:

$$\mathbf{cov}(\widehat{\boldsymbol{\beta}}) = \sigma_w^2(\mathbf{Z}'\mathbf{Z})^{-1} = \sigma_w^2 \begin{pmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{C} \end{pmatrix} \tag{4.2}$$

where

σ_w^2 denotes the variance of the (goal) simulation output w;

$\mathbf{Z}$ is the $N \times (1+k)$ matrix of explanatory regression variables including the column with N one's;

$N = \sum_{i=1}^{n} m_i$ is the total number of simulation runs;

n is the number of simulated input combinations;

m_i is the number of Identically and Independently Distributed (IID) replicates for combination i;

a is a scalar;

$\mathbf{b}$ is a k-dimensional vector;

$\mathbf{C}$ is a $k \times k$ matrix such that $\mathbf{cov}(\widehat{\boldsymbol{\beta}_{-0}}) = \sigma_w^2\mathbf{C}$.

Note: $\mathbf{Z}$'s first column corresponds with the intercept β_0. Furthermore, $\mathbf{Z}$ is determined by the resolution-III design, transformed into the original values of the inputs in the local area. To save computer time, only the center of the local area may be replicated (the center is not part of the resolution-III design). Replicates use the same input combination $\mathbf{z}_i (i = 1, \ldots n)$ but different PRNs. For more details I refer back to Chapter 2.

Under this assumption, the variance is estimated through the Mean Squared Residuals (MSR):

$$\widehat{\sigma_w}^2 = \frac{\sum_{i=1}^{n}\sum_{r=1}^{m_i}(w_{i;r} - \widehat{y_i})^2}{(\sum_{i=1}^{n} m_i) - (k+1)} \tag{4.3}$$

where $\widehat{y_i} = \mathbf{z}_i'\widehat{\boldsymbol{\beta}}$; also see Chapter 2.

It is easy to prove that the predictor variance $var(\widehat{y}|\mathbf{z})$ increases as $\mathbf{z}$ (point to be predicted) moves further away from the local area where the gradient is estimated. The point with the minimum predictor variance is the point $-\mathbf{C}^{-1}\mathbf{b}$.

The new point to be simulated is

$$\mathbf{d} = -\mathbf{C}^{-1}\mathbf{b} - \lambda\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}_{-0}} \tag{4.4}$$

where

- $-\mathbf{C}^{-1}\mathbf{b}$ is the point where the local search starts, namely the point with the minimum variance in the local area

- λ is the step size.

- $\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}_{-0}}$ is the (classic) steepest descent direction (namely, $\widehat{\boldsymbol{\beta}_{-0}}$) adapted for $\mathbf{cov}(\widehat{\boldsymbol{\beta}_{-0}})$. It is easy to see that if $\mathbf{C}$ is a diagonal matrix, then the higher the variance of a factor effect is, the less the search moves into the direction of that factor.

Exercise 4.2 *Prove that the search direction in (4.4) does not change the steepest descent direction if the design matrix is orthogonal (so* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*).*

Accounting for $\mathbf{cov}(\widehat{\boldsymbol{\beta}_{-0}})$ gives a *scale independent* search direction, which in general outperforms the steepest descent direction.

Note: A Bayesian approach to gradient estimation is given in [272]; also see my brief discussion on Bayesian analysis in Section 4.5.

4.3 Generalized RSM: multiple outputs and constraints

In practice, simulation models have multiple responses (multivariate output; also see Section 3.2). Several approaches to solve the resulting issues are surveyed in [316]. Furthermore, the RSM literature also offers several approaches for such situations; see the surveys in [12], [176], [273], and [286]. However, I find these approaches less attractive than the following approach—which I call Generalized RSM or GRSM—that my coauthors and I present in [12].

We assume that one simulation output should be minimized, while all the other outputs must satisfy given constraints (so we do not use multi-objective optimization). More specifically, GRSM has the following characteristics.

- We generalize the steepest descent search direction (applied in classic RSM), using the "affine scaling search direction" and borrowing ideas from *Interior Point* (IP) methods (a variation on Karmarkar's algorithm) in Mathematical Programming; see [23]. Our search direction moves faster to the optimum than steepest descent, since our search avoids creeping along the boundary of the feasible area (this feasible area is determined by the constraints on the random outputs and the deterministic inputs; see below). Moreover, our search tries to stay inside the feasible area, so the simulation program does not crash. Finally, our search direction is scale independent, which is an important characteristic for both practitioners and researchers.

- We use our search direction *iteratively* (as classic RSM does). Because we assume expensive simulation experiments, the search should quickly reach a neighborhood of the true optimum.

- Though we develop our heuristic for *random* simulations, we can easily adapt it for *deterministic* simulations and *real-world* (nonsimulated) systems—analogous to classic RSM.

Formally, we extend the classic RSM problem formulated in (4.1) to the following *constrained nonlinear random optimization problem*:

$$\min_{\mathbf{z}} E(w_0|\mathbf{z}) \tag{4.5}$$

such that the other $(r-1)$ random outputs (also see equation 3.1) satisfy the constraints

$$E(w_{h'}|\mathbf{z}) \geq a_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{4.6}$$

and the k deterministic inputs satisfy the so-called *box constraints*

$$l_j \leq z_j \leq u_j \text{ with } j = 1, \ldots, k. \tag{4.7}$$

An example is the following inventory simulation. The sum of the expected inventory carrying costs and ordering costs should be minimized, The expected service percentage (or fill rate) should be at least (say) 90% so $a_1 = 0.9$ in (4.6). Both the reorder quantity $z_1(= Q)$ and the reorder level $z_2(= s)$ should be nonnegative, so $z_1 \geq 0$ and $z_2 \geq 0$; see (4.7). (Note the similarity of the constraints on the random outputs and the deterministic inputs.)

Note: Stricter input constraints may be formulated; e.g., the reorder level should at least cover the expected demand during the expected order lead time. Input constraints more complicated than these box constraints (namely, geometry constraints) are discussed in [365] and [368]. Simple linear constraints for the inputs are used in [93] and [294]. Input constraints resulting from output constraints are discussed in [273].

Analogously to the first steps of classic RSM, we locally approximate the multivariate I/O function (see equation 3.1) by r univariate first-order polynomials (see equation 3.2):

$$\mathbf{y}_h = \mathbf{Z}\boldsymbol{\beta}_h + \mathbf{e}_h \text{ with } h = 0, \ldots r-1. \tag{4.8}$$

We assume that *locally* the white noise assumption holds; i.e., the residuals e are Normally IID with zero mean and constant variance: $e \sim NIID(0, \sigma^2)$ (see Chapter 3). The following OLS estimators are then the BLUEs:

$$\widehat{\boldsymbol{\beta}_h} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}_h \text{ with } h = 0, \ldots, r-1. \tag{4.9}$$

Then $\widehat{\boldsymbol{\beta}_0}$ (OLS estimator for first-order polynomial approximation of goal function) and the goal function (4.5) itself result in

$$\min_{\mathbf{z}} \widehat{\boldsymbol{\beta}_{0;-0}}\mathbf{z} \tag{4.10}$$

where $\widehat{\boldsymbol{\beta}_{0;-0}}$ denotes the OLS estimate of the local regression parameters for the goal output (which explains the first subscript 0) excluding the intercept (which explains the second subscript -0); i.e., $\widehat{\boldsymbol{\beta}_{0;-0}} = (\widehat{\beta_{0;1}}, \ldots, \widehat{\beta_{0,k}})'$ is the *estimated local gradient* of the goal function. (The alternative, more complicated symbol $\widehat{\boldsymbol{\beta}_{0;-0}}(\mathbf{z})$ would emphasize that the gradient depends on the local area being explored.)

The $r-1$ estimates $\widehat{\boldsymbol{\beta}_{h'}}$ with $h' = 1, \ldots, r-1$ in (4.9) combined with the original output constraints (4.6) give

$$\widehat{\boldsymbol{\beta}_{h';-0}}\mathbf{z} \geq c_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{4.11}$$

where $\widehat{\boldsymbol{\beta}_{h';-0}} = (\widehat{\beta_{h';1}}, \ldots, \widehat{\beta_{h',k}})'$ denotes the estimated local gradient of constraint function h', and $c_{h'} = a_{h'} - \widehat{\beta_{h';0}}$denotes the modified right-hand side of this constraint function.

The box constraints in (4.7) remain unchanged.

Now the $(r-1)$ k-dimensional vectors $\widehat{\boldsymbol{\beta}_{h';-0}}$ in (4.11) are collected in the $(r-1) \times k$ matrix called $\mathbf{B}$. Likewise, the $(r-1)$ elements $c_{h'}$ are collected in the vector $\mathbf{c}$. And the k-dimensional vectors with the nonnegative *slack variables* $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ are introduced. Altogether this gives

$$\begin{array}{ll} \text{minimize} & \widehat{\boldsymbol{\beta}_{h';-0}}\mathbf{z} \\ \text{subject to} & \mathbf{Bz} - \mathbf{s} = \mathbf{c} \\ & \mathbf{z} + \mathbf{r} = \mathbf{u} \\ & \mathbf{z} - \mathbf{v} = \mathbf{l}. \end{array} \tag{4.12}$$

This (local) optimization problem is *linear* in the decision variables $\mathbf{z}$ (the OLS estimates $\widehat{\boldsymbol{\beta}_{0;-0}}$ and $\widehat{\boldsymbol{\beta}_{h';-0}}$ in $\mathbf{B}$ use the property that this problem is also linear in the regression parameters). We do not solve this LP problem, but use this problem only to derive the following novel *search direction* $\mathbf{d}$:

$$\mathbf{d} = -\left(\mathbf{B}^{'}\mathbf{S}^{-2}\mathbf{B} + \mathbf{R}^{-2} + \mathbf{V}^{-2}\right)^{-1}\widehat{\boldsymbol{\beta}_{0;-0}} \tag{4.13}$$

where $\mathbf{S}$, $\mathbf{R}$, and $\mathbf{V}$ are diagonal matrixes with as main-diagonal elements the current estimated slack vectors $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ in (4.12); the factor $-\widehat{\boldsymbol{\beta}_{0;-0}}$ is the estimated classic steepest descent direction. Our search direction can be proven to be scale independent; i.e., the linear transformations in (2.32) do not affect this search direction. For details, I refer to [12].

As the value of a slack variable in (4.13) decreases (so the corresponding constraint gets tighter), our search direction deviates more from the steepest descent direction. Possible singularity of the various matrices in (4.13) is discussed in [12].

Following the search direction (or path) defined by (4.13), we must decide on the *step size* (say) λ along this path. We derive an explicit step size in [12], assuming that the local metamodel (4.11) holds *globally*:

$$\lambda = 0.8 \min_{h'}\left[\frac{c_{h'} - \widehat{\boldsymbol{\beta}_{h';-0}}^{'}\mathbf{z}_c}{\widehat{\boldsymbol{\beta}_{h';-0}}^{'}\mathbf{d}}\right] \tag{4.14}$$

where the factor 0.8 is chosen to decrease the probability that the local metamodel is misleading when applied globally; $\mathbf{z}_c$ denotes the current input combination, so the new combination becomes $\mathbf{z}_c + \lambda\mathbf{d}$. Obviously, the box constraints (4.7) for the deterministic inputs hold globally, so it is easy to check the solution in (4.14) against these constraints.

Analogously to classic RSM, we proceed *stepwise*; i.e., after each step along our search path we test the following two hypotheses:

1. $w_0(\mathbf{z}_c + \lambda\mathbf{d})$ (the simulation output of the new combination) is *no improvement* over $w_0(\mathbf{z}_c)$ (the output of the old combination); i.e. this step increases the goal output w_0 (pessimistic null-hypothesis):

$$H_0 : E[w_0(\mathbf{z}_c + \lambda\mathbf{d})] \geq E[w_0(\mathbf{z}_c)]. \tag{4.15}$$

2. This step is *feasible*; i.e., the new solution satisfies the $(r-1)$ constraints in (4.6):

$$H_0 : E(w_{h'}|\mathbf{z}_c + \lambda\mathbf{d})) \geq a_{h'}\text{with } h' = 1, \ldots, r-1. \tag{4.16}$$

To test these hypotheses, I propose the following simple statistical procedures (more complicated parametric bootstrapping is used in [12], which permits non-normality and tests the relative improvement $w_0(\mathbf{z}_c + \lambda\mathbf{d})/w_0$ $(\mathbf{z}_c)$ and the relative slacks $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$; see Section 3.3.3 and the following exercise).

Exercise 4.3 *Which statistical problem arises when testing the ratio of the slack at the new solution and the slack at the old solution, $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$?*

Sub 1: To test the hypothesis in (4.15), the classic t statistic may be applied. A paired t statistic may be applied if CRN are used to obtain the two simulation outputs $w_0(\mathbf{z}_c + \lambda\mathbf{d})$ and $w_0(\mathbf{z}_c)$. To estimate the standard error of their difference, $m \geq 2$ replicates suffice: the t statistic has $v = m-1$ degrees of freedom. The hypothesis is rejected if significant improvement is observed. (Note that [171] also uses a t test in a simulation optimization context, but that article uses the so-called quasi-Newton method instead of our Interior Point method.)

Sub 2: Again a t statistic with $m-1$ degrees of freedom may be applied. Because $r-1$ hypotheses are implied by (4.16), Bonferroni's inequality may be used (if $r > 2$).

Actually, a better solution may lie somewhere between $\mathbf{z}_c$ (the old combination) and $\mathbf{z}_c + \lambda\mathbf{d}$ (the new combination at the "maximum" step size). We therefore recommend to apply a *binary search*; i.e., simulate a combination that lies halfway between these two combinations (and is still on the search path). We may apply this halving of the stepsize a number of times.

Next we proceed analogously to classic RSM. So around the best combination found so far, we select a new local area. We again use a resolution-III design to select the simulation runs to be generated. We again may replicate only the new center $m > 1$ times. And we fit r first-order polynomials, which gives a *new* search direction. And so on.

We apply GRSM to two examples, namely an inventory simulation with a service-level constraint so the solution is unknown (see [28]), and an artificial example with known solution (most test functions in simulation optimization are unconstrained; see, e.g., the seven functions in [274] and the seven multi-modal functions in [175]). The results of these examples are encouraging, as GRSM finds solutions that are both feasible and give drastically lower goal functions.

Figure 4.2 gives an example, which deserves the following comments.

- There are two decision variables; see the two axes labeled z_1 and z_2.
- There is one goal function; the figure shows only two contour functions, namely $E(w_0) = a_{0;1}$ and $E(w_0) = a_{0;2}$ with $a_{0;2} < a_{0;1}$.
- There are two constrained random outputs; see $E(w_1) = a_1$ and $E(w_2) = a_2$, which correspond with the boundary of the feasible area.
- The search starts in the lower right local area, where a 2^2 design is executed; see the four elongated points.
- This design and the (not shown) replicates give a search direction; see the arrow leaving from point (0).

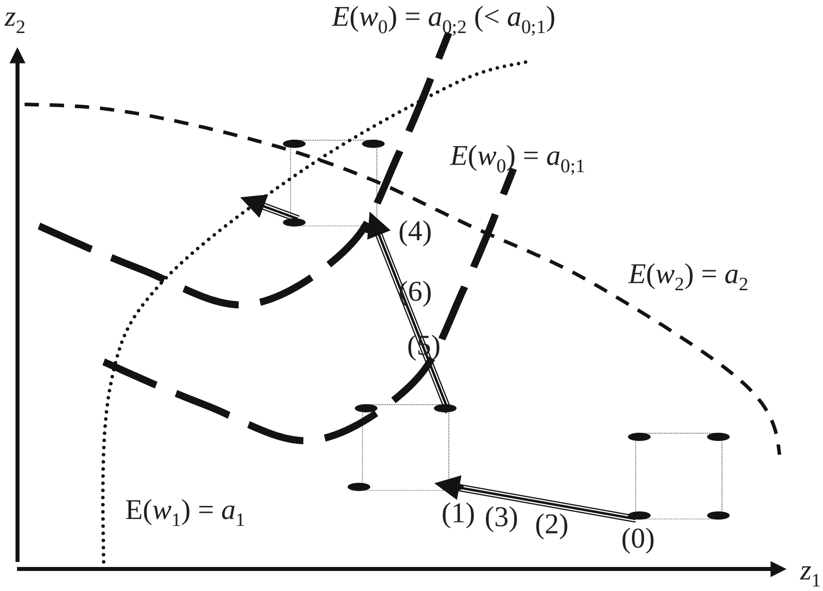

Figure 4.2: GRSM example

- The maximum step size along this path takes the search from point (0) to point (1).

- The binary search takes the search back to point (2), and next to point (3).

- Because the best point so far turns out to be point (1), the 2^2 design is simulated at the new local area with this point as one of its input combinations.

- This design gives a new search direction, which avoids the boundary.

- The maximum step size now takes the search to point (4).

- The binary search takes the search back to point (5), and next to point (6).

- Because the best point so far is now point (4), the 2^2 design is simulated at the local area with this point as one of its points.

- A new search direction is estimated; etc. (the remaining search is not displayed).

Exercise 4.4 *Apply GRSM to the following artificial example reproduced from [12]:*

$$\begin{array}{ll} \textit{minimize} & E[5(z_1-1)^2+(z_2-5)^2+4z_1z_2+e_0] \\ \textit{subject to} & E[(z_1-3)^2+z_2^2+z_1z_2+e_1] \leq 4 \\ & E[z_1^2+3\,(z_2+1.061)^2+e_2] \leq 9 \\ & 0 \leq z_1 \leq 3,\ -2 \leq z_2 \leq 1 \end{array} \tag{4.17}$$

where e_0, e_1, and e_2 are the components of a multivariate normal variate with mean **0***, variances $\sigma_{0,0}=1$ (so $\sigma_0=1$), $\sigma_{1,1}=0.0225$ (or $\sigma_1=0.15$), $\sigma_{2,2}=0.16$ (or $\sigma_2=0.4$), and correlations $\rho_{0,1}=0.6$, $\rho_{0,2}=0.3$, $\rho_{1,2}=-0.1$.*

4.4 Testing an estimated optimum: KKT conditions

By definition, it is uncertain whether the optimum estimated by a heuristic (e.g., GRSM) is close enough to the true optimum. In *deterministic* Nonlinear Mathematical Programming, the KKT first-order optimality conditions have been derived; see, e.g., [127].

Figure 4.3 illustrates the same type of problem as the one in Figure 4.2. In the present example there is again a goal function $E(w_0)$, for which three

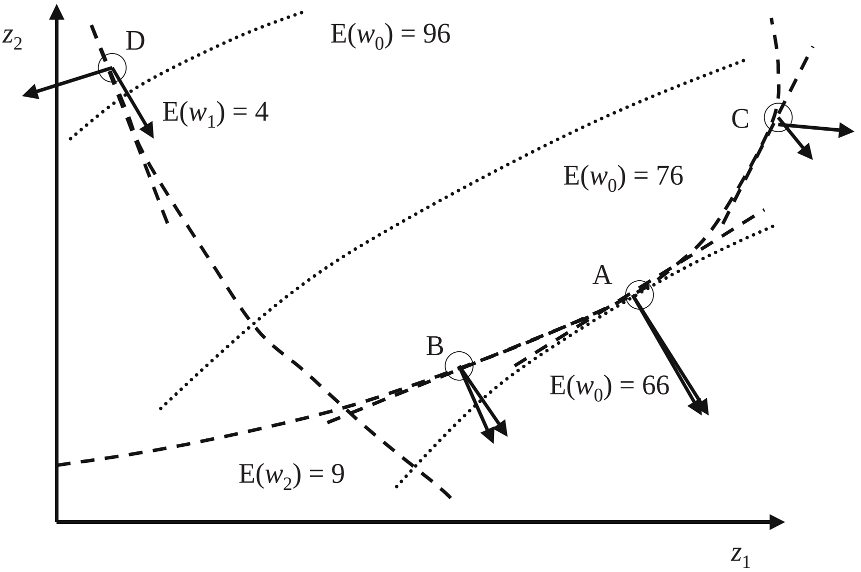

Figure 4.3: A constrained nonlinear random optimization problem

contour plots are displayed corresponding to the values 66, 76, and 96; see (4.5). There are two constrained simulation outputs, namely $E(w_1) \geq 4$ and $E(w_2) \geq 9$. The optimum combination is point A. Points B and C lie on the boundary $E(w_2) = 9$; point D lies on the boundary $E(w_1) = 4$; also see (4.6). Obviously, point D is far away from the optimum combination A. The figure also displays the (local) gradients at these four points for the goal function and the *binding constraint*; i.e., the constraint with a zero slack value in (4.6). These gradients are perpendicular to the local tangent lines; those lines are shown only for the binding constraint (not for the goal function). At the optimum, the gradients for the goal function and the binding constraint coincide; at point D, these two gradients point in very different directions.

Let $\mathbf{z}^0$ denote a local minimizer for the deterministic variant of our problem. The KKT first-order necessary optimality conditions for $\mathbf{z}^0$ are then

$$\begin{aligned} &\boldsymbol{\beta}_{0;-0} = \sum_{h \in A(\mathbf{z}^0)} \lambda_h^0 \boldsymbol{\beta}_{h;-0} \\ &\lambda_h^0 \geq 0 \\ &h \in A\left(\mathbf{z}^0\right) \end{aligned} \tag{4.18}$$

where $\boldsymbol{\beta}_{0;-0}$ denotes the k-dimensional vector with the gradient of the goal function (see (4.10)); $A\left(\mathbf{z}^0\right)$ is the index set with the indices of the constraints that are binding at $\mathbf{z}^0$; λ_h^0 is the Lagrange multiplier for binding constraint h; $\boldsymbol{\beta}_{h;-0}$ is the gradient of the output in that binding constraint (in Figure 4.3, there is only one binding constraint at the points A through D). The KKT conditions imply that the gradient of the objective can be expressed as a nonnegative linear combination of the gradients of the binding constraints, at $\mathbf{z}^0$. Note that there is a certain constraint qualification that is relevant when there are nonlinear constraints in the problem; see [127], p. 81. There are several types of constraint qualification, but many are only of theoretical interest; a practical constraint qualification for nonlinear constraints is that the $r-1$ constraint gradients at $\mathbf{z}^0$ be linearly independent.

In Figure 4.3 point A satisfies the KKT conditions; point B has two gradients that point in different but similar directions—and so does point C. Point D, however, has two gradients that point in completely different directions.

Note: If the optimum occurs *inside* the feasible area, then there are no binding constraints. The KKT conditions then reduce to the condition that the goal gradient is zero. In classic RSM, the analysts test for a zero gradient, estimated from a second-order polynomial; see Step 7 in Section 4.2. This test may use a classic F-test (see Section 4.2 and Chapter 2). Recently, [172] considered an unconstrained optimization problem, and estimated the gradient of the goal function through the SF method. I do not consider such situations any further.

Unfortunately, in *random* simulation the analysts must estimate the gradients. Moreover, to check which constraints are binding, the analysts must estimate the slacks of the constraints. This estimation turns the KKT conditions (4.18) into a problem of nonlinear statistics.

Angün and I derive an asymptotic test in [13] to check whether the optimum has indeed been found. Together with two other coauthors, I derive an alternative, bootstrap test in [39]. I focus on the latter test, because it is simpler (the former test uses the so-called Delta method and a generalized form of the so-called Wald statistic) and it allows a small number of replicates (as is the case in expensive simulation). Both tests assume a problem like the one formulated in the preceding section; i.e., there is one random simulation output to be minimized and there are $r-1$ constrained random simulation outputs; see (4.5) and (4.6).

We assume that estimators of the gradients are available. Some (iterative, heuristic) simulation optimization methods do estimate these gradients; some do not. For example, RSM, PA, and the SF method do give estimated gradients (using either multiple input combinations like RSM does or a single run like PA and SF do; also see [400]); most metaheuristics do not estimate gradients. However, when the analysts apply a metaheuristic, then they may follow-up with a local experiment to estimate the gradients at the estimated optimum and use these gradients as a stopping criterion—instead of using rather arbitrary criteria such as a prefixed computer budget.

Note: Whenever a metaheuristic is used to estimate gradients while treating the simulation model as a *black box*, the analysts should not change one factor at a time followed by some type of finite differencing, such as forward or central finite differences. Nevertheless, such an approach is proposed in, e.g., [108], [171], [228], [360], and [412]. Instead, the analysts should use classic designs to fit first-order or second-order polynomials locally; e.g., the tangent lines in Figure 4.3 may be interpreted as first-order polynomials. To fit such polynomials, classic RSM uses highly efficient resolution-III designs and a CCD (see Section 4.2 and also [53] and [169]). Obviously, the estimated gradient is biased if second-order effects are important and yet a first-order polynomial is fitted.

The technical details of the bootstrap KKT test are presented in the remainder of this section (so readers not interested in these details should skip to Section 4.5). As in classic RSM, I assume *locally constant (co)variances* for each of the r simulation outputs; i.e., when moving to a new local area, the (co)variances may change. For example, the points A through D in Figure 4.4 do not have the same variance for the goal output. This assumption is part of the white-noise assumption. The latter assumption implies that OLS applied per univariate simulation output gives the BLUE, $\widehat{\boldsymbol{\beta}_h}$ $(h = 0, 1, \ldots, r-1)$ defined in (4.9). Even if the (co)variances do not remain locally constant, the OLS estimators remain unbiased; see Section 3.4.4.

However, only under locally constant (co)variances do the OLS estimators have the following estimated covariance matrix (also see [297]:

$$\mathbf{cov}(\widehat{\widehat{\boldsymbol{\beta}}_h, \widehat{\boldsymbol{\beta}}_{h'}}) = \mathbf{cov}(\widehat{w_h, w_{h'}}) \otimes (\mathbf{Z}'\mathbf{Z})^{-1} \; (h, h' = 0, \ldots, r-1) \qquad (4.19)$$

where $\otimes$ is the well-known “Kronecker product” operator (also see [265], p. 13) and $\mathbf{cov}(\widehat{w_h, w_{h'}}))$ is an $r \times r$ matrix with the classic estimators of the (co)variances based on the m replicates at the local center (also see equation 2.26):

$$\mathbf{cov}(\widehat{w_h, w_{h'}}) = (\widehat{\sigma_{h;h'}}) = (\sum_{l=1}^{m} (w_{h;l} - \overline{w_h})(w_{h';l} - \overline{w_{h'}}))\, \frac{1}{m-1}. \qquad (4.20)$$

The Kronecker product implies that $\mathbf{cov}(\widehat{\widehat{\boldsymbol{\beta}}_h, \widehat{\boldsymbol{\beta}}_{h'}})$ is an $rq \times rq$ matrix (where q still denotes the number of parameters of the univariate regression model per simulation output; for example $q = 1 + k$ in a first-order polynomial regression metamodel), formed from the $r \times r$ matrix $\mathbf{cov}(\widehat{w_h, w_{h'}})$ by multiplying each of its elements by the entire $q \times q$ matrix $(\mathbf{Z}'\mathbf{Z})^{-1}$ (in (4.2), $\mathbf{Z}$ was an $N \times (1 + k)$ matrix). The matrix $\mathbf{cov}(\widehat{w_h, w_{h'}})$is singular if $m \leq r$; e.g., the case study in [186] has $r = 2$ response types and $k = 14$ inputs so $m \geq 3$ replicates of the center point are required. Of course, the higher m is, the higher is the power of the tests that use these replicates.

Another reason for replicating the center point is that this point is used to test whether a constraint is binding in the current local area ; see (4.21) below. The center point is more representative of the local behavior than the points of the resolution-III design. Classic RSM also uses replication of the center point when using a CCD.

I also assume that the r-variate simulation output is multivariate *Gaussian*. Then (as in classic RSM) the *validity* of the local metamodel may be tested through the classic lack-of–fit F-statistic; see (2.30). This test also assumes that no CRN are applied. In our GRSM variant there are multiple simulation responses, so this classic test is combined with *Bonferroni's inequality*; i.e., the classic type-I error rate α is replaced by the “experimentwise” or “familywise” error rate α/r. (A more complicated, multivariate variant of this test is given in [317].)

If the metamodel is rejected, then there are two options:

- Decrease the local area; e.g., halve each factor's range.
- Increase the order of the polynomial; e.g., switch from a first-order to a second-order polynomial.

I do not explore these options further in this chapter, but refer to the RSM literature.

Testing the KKT conditions in random simulation implies testing the following three null-hypotheses, denoted by the superscripts (1) through (3):

1. The current solution is *feasible* and at least one constraint is *binding*; see (4.6):

$$H_0^{(1)}: E(w_{h'}|\mathbf{d}=\mathbf{0}) = a_{h'} \text{ with } h' = 1, \ldots, r-1 \tag{4.21}$$

   where $\mathbf{d}=\mathbf{0}$ corresponds with the center of the local area expressed in the inputs standardized through (2.32).

2. The expected value of the estimated goal gradient may be expressed as the expected value of a *linear* combination of the estimated gradients of the simulation outputs in the binding constraints; i.e., in (4.18) the deterministic quantities are replaced by their random estimators:

$$H_0^{(2)}: E(\widehat{\boldsymbol{\beta}}_{0;-0}) = E\left(\sum_{j\in A(\mathbf{z}^0)} \widehat{\lambda_j^0 \boldsymbol{\beta}_j)}\right). \tag{4.22}$$

3. The Lagrange multipliers in (4.22) are nonnegative:

$$H_0^{(3)}: E(\widehat{\boldsymbol{\lambda}}) \geq \mathbf{0}. \tag{4.23}$$

Note: Each of these three hypotheses requires multiple tests, so Bonferroni's inequality is applied. Moreover, these three hypotheses are tested sequentially, so it is hard to control the final type-I and type-II error probabilities. However, classic RSM has the same type of problems (multiple and sequential tests), and nevertheless RSM has acquired a track record in practice.

Sub 1: To save simulation runs, a local experiment should start at its center point including replicates. If it turns out that either no constraint is binding or at least one constraint is violated, then the other hypotheses need not be tested so the remainder of the local design is not simulated.

To test the hypothesis (4.21), the following t statistic may be used:

$$t_{m-1}^{(h')} = \frac{\overline{w_{h'}(\mathbf{d}=\mathbf{0})} - a_{h'}}{\sqrt{\widehat{\sigma_{h';h'}}/m}} \text{with } h' = 1, \ldots, r-1 \tag{4.24}$$

where both the numerator and the denominator are based on the m replicates at the local center point (also see (4.20)).

The t statistic in (4.24) may give the following three different results.

- The statistic is *significantly positive*. The analysts may then conclude that the constraint for output h' is not binding. If none of the $(r-1)$

constraints is binding, then the optimal solution is not yet found (assuming that at the optimum at least one constraint is binding; otherwise, classic RSM applies). The search for better solutions continues (also see Section 4.3 above).

- The statistic is *significantly negative*. The analysts may conclude that the current local area does not give feasible solutions; i.e., the optimal solution is not yet found. The search should back up into the feasible area (again see Section 4.3).
- The statistic is *nonsignificant*. The analysts may conclude that the current local area gives feasible solutions, and the constraint for output h' is binding. The index of this gradient is then included in $A\left(\mathbf{z}^0\right)$; see (4.22). And the analysts proceed to test whether they have indeed found the optimal solution—as follows.

Sub 2: The hypothesis (4.22) is that the expected value of the goal gradient may be expressed as the expected value of a *linear* combination of the estimated gradients of the binding constraints. To estimate this linear combination, we apply OLS using as explanatory variables the estimated gradients of the (say) J binding constraints (so the explanatory variables become random). We collect the latter gradients in the $k \times J$ matrix $\widehat{\mathbf{B}_{J;-0}}$. The parameters estimated through OLS are $\widehat{\boldsymbol{\lambda}}$. Let $\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}$ denote the OLS estimator of the goal gradient; i.e.

$$\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}} = \widehat{\mathbf{B}_{J;-0}}(\widehat{\mathbf{B}_{J;-0}}'\widehat{\mathbf{B}_{J;-0}})^{-1}\widehat{\mathbf{B}_{J;-0}}'\widehat{\boldsymbol{\beta}_{0;-0}} = \widehat{\mathbf{B}_{J;-0}}\widehat{\boldsymbol{\lambda}} \tag{4.25}$$

where $\widehat{\boldsymbol{\lambda}} = (\widehat{\mathbf{B}_{J;-0}}'\widehat{\mathbf{B}_{J;-0}})^{-1}\widehat{\mathbf{B}_{J;-0}}'\widehat{\boldsymbol{\beta}_{0;-0}}$ is the OLS estimator of the Lagrange multipliers in the KKT conditions. Obviously, if $\widehat{\boldsymbol{\beta}_{0;-0}}$ and the vectors in $\widehat{\mathbf{B}_{J;-0}}$ point in the same direction, then all the components of $\widehat{\boldsymbol{\lambda}}$ are positive; if $\widehat{\boldsymbol{\beta}_{0;-0}}$ and $\widehat{\mathbf{B}_{J;-0}}$ are perpendicular, then $\widehat{\boldsymbol{\lambda}}$ is zero; if $\widehat{\boldsymbol{\beta}_{0;-0}}$ and $\widehat{\mathbf{B}_{J;-0}}$ point in opposite directions, then $\widehat{\boldsymbol{\lambda}}$ is negative; also see the points A through D in Figure 4.3. The expression in (4.25) is highly nonlinear; bootstrapping is a classic analysis method for nonlinear statistics.

There are several statistics for quantifying the accuracy of a model fitted through OLS. One of the statistics for quantifying the validity of our linear approximation is the k-dimensional vector of residuals

$$\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}}) = \widehat{\widehat{\boldsymbol{\beta}_{0;-0}}} - \widehat{\boldsymbol{\beta}_{0;-0}}. \tag{4.26}$$

The hypothesis (4.22) implies $E(\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}})) = \mathbf{0}$. Section 3.3.3 focused on distribution-free bootstrapping. In expensive simulation, however, only the center point is replicated a few times so this type of bootstrapping does not give good results. Therefore *parametric bootstrapping* should be used; i.e., a specific distribution type is assumed and its parameters are estimated from the simulation's I/O data at hand (therefore the bootstrap is called "data driven"). Like in classic RSM, we assume a normal distribution.

Altogether, our KKT test procedure uses three layers of models:

1. The simulation model, which GRSM treats as a black box.
2. The regression metamodel, which uses the simulation I/O data $(\mathbf{Z}, \mathbf{w})$ as input and estimates the gradients of the goal response $(\widehat{\boldsymbol{\beta}_{0;-0}})$ and of the constrained responses including the binding constraints collected in $\widehat{\mathbf{B}_{J;-0}}$. The regression analysis also estimates $\mathbf{cov}(\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}, \widehat{\mathbf{B}_{J;-0}})$, the covariance matrix of these estimated gradients.
3. The bootstrap model, which uses the regression output $(\widehat{\boldsymbol{\beta}_{0;-0}}, \widehat{\mathbf{B}_{J;-0}}$, $\mathbf{cov}(\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}, \widehat{\mathbf{B}_{J;-0}}))$ as parameters of the multivariate normal distribution of its output $\widehat{\boldsymbol{\beta}^*_{0;-0}}$ and $\widehat{\mathbf{B}^*_{J;-0}}$ where the superscript $*$ denotes bootstrapped values.

More specifically, our bootstrap procedure consists of the following four steps.

1. Use the *Monte Carlo* method to sample $vec(\widehat{\boldsymbol{\beta}^*_{0;-0}}, \widehat{\mathbf{B}^*_{J;-0}})$, which is a $(k+kJ)$-dimensional vector formed by "stapling" or "stacking" the k-dimensional goal gradient vector and the J k-dimensional vectors of the $k \times J$ matrix $\widehat{\mathbf{B}^*_{J;-0}}$:

$$vec(\widehat{\boldsymbol{\beta}^*_{0;-0}}, \widehat{\mathbf{B}^*_{J;-0}}) \sim N(vec(\widehat{\boldsymbol{\beta}_{0;-0}}, \widehat{\mathbf{B}_{J;-0}}), \mathbf{cov}[vec(\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}, \widehat{\mathbf{B}_{J;-0}})]) \tag{4.27}$$

   where $\mathbf{cov}[vec(\widehat{\widehat{\boldsymbol{\beta}_{0;-0}}}, \widehat{\mathbf{B}_{J;-0}})]$ is the $(k+kJ) \times (k+kJ)$ matrix computed through (4.19).

2. Use the bootstrap values resulting from Step 1, to compute the *OLS* estimate of the bootstrapped goal gradient using the bootstrapped gradients of the binding constraints as explanatory variables; i.e., use (4.25) adding the superscript $*$ to all random variables—resulting in $\widehat{\widehat{\boldsymbol{\beta}^*_{0;-0}}}$ and $\widehat{\boldsymbol{\lambda}}^*$.

3. Use $\widehat{\widehat{\boldsymbol{\beta}^*_{0;-0}}}$ from Step 2 and $\widehat{\boldsymbol{\beta}^*_{0;-0}}$ from Step 1 to compute the *bootstrap residual*$\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}^*_{0;-0}}}}) = \widehat{\widehat{\boldsymbol{\beta}^*_{0;-0}}} - \widehat{\boldsymbol{\beta}^*_{0;-0}}$ analogous to (4.26). Furthermore,

determine whether any of the *bootstrapped Lagrange multipliers* $\widehat{\boldsymbol{\lambda}}^*$ (found in Step 2) is negative; i.e., augment a counter (say) c^* with the value 1 if this event occurs.

4. *Repeat* the preceding three steps (say) 1,000times (this is the bootstrap sample size, denoted by B in Section 3.3.3). This gives the Estimated Density Function (EDF) of $\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}}}{}^*_{0;-0;j})$ (the bootstrapped residuals for input j with $j = 1, \ldots, k$), and the final value of the counter c^*. Reject the hypothesis in (4.22) if this EDF implies a two-sided $(1 - \alpha/(2k))$ confidence interval that does not cover the value 0 (the factor k is explained by Bonferroni's inequality). Reject the hypothesis in (4.23) if the fraction $c^*/1{,}000$ is significantly higher than 50% (if the true Lagrange multiplier is only "slightly" larger than zero, then "nearly" 50% of the bootstrapped values is negative). To test the latter fraction, the binomial distribution may be approximated through the normal distribution with mean 0.50 and variance $(0.50 \times 0.50)/1{,}000 = 0.00025$.

The numerical examples that we report in [39] are encouraging:

1. The classic t test for zero slacks and the classic F test for lack-of-fit perform as expected.

2. Our bootstrap tests give observed type I error rates close to the prespecified rates; the type II error rate (complement of the power) decreases as the input combination tested moves farther away from the true optimum (see the points A through D in Figure 4.3).

4.5 Risk analysis

In Section 1.1, I mentioned that a *deterministic* simulation model—such as Example 1.1 with its Net Present Value (NPV) spreadsheet computation—may be augmented to a *random* simulation model—if inputs such as the discount factor θ or the cash flows x_t are unknown so their values are sampled from distribution functions. The latter type of simulation is called *Risk Analysis* (RA) or *Uncertainty Analysis* (UA); see again [44], [313], [327], [340], and recent textbooks such as [110] and [392].

In that same section, I also mentioned that complicated examples of deterministic simulation models are provided by models of airplanes, automobiles, chemical processes, computer chips, etc.—applied in *Computer Aided Engineering* (CAE) and *Computer Aided Design* (CAD). I referred to the recent surveys [5], [65], [66], [254], [280], and [357]; additional examples are [298] and [365].

Another type of deterministic simulation is used in *routing protocols* in telematics and *project planning* through the “Critical Path Method” (CPM) and “Programme Evaluation and Review Technique” (PERT). Classic models assume known values for the components of the total routing or project respectively. Simulation models allow the input values to be random; e.g., durations are sampled from beta distributions. See [107] and [240].

In general, even a *deterministic* simulation model generates *random* output if the model’s input variables and parameters are sampled from a (prior) distribution because their values are not known exactly. This uncertainty is called *subjective* or *epistemic*; see [145]. The latter publication includes references to methods for obtaining subjective distributions based on *expert opinions*. Alternative representations of this uncertainty—such as fuzzy sets and evidence theory—are given in [30], [144] and [145].

Random simulations (such as Discrete-Event Dynamic Systems or DEDS simulations) have objective, *aleatory* or inherent uncertainty; again see [145] (that reference also cites publications that use Importance Sampling for rare events with major consequences). I add that DEDS simulation models represent real systems that without this inherent uncertainty would have a completely different character; e.g., a queueing model without uncertain arrival and service times is not a queueing problem anymore; it becomes a scheduling problem. I refer to the recent tutorial [69], and also to [10] and [416].

I claim that RA answers different questions than Sensitivity Analysis (SA) does. SA answers the question: “Which are the most important factors in the simulation model of a given real system”? RA answers the question: “What is the probability of a given event happening; e.g., what is the probability of a nuclear or a financial disaster happening in the system under investigation (a nuclear reactor, a bank)”? A nuclear waste example was discussed in Example 2.7; also see [145]. A financial risk example is the estimation of the 5% quantile of the NPV distribution in [118]. Food safety risks (e.g., foot and mouth disease, terrorist food poisoning, natural disasters such as extreme weather) are discussed in [45].

Note: SA may help identify those inputs for which the distributions in RA need further refinement; see [145]. (Bayesian approaches are discussed below). SA may use DASE, because DASE gives better answers; i.e., the common sense approach changing one factor at a time gives estimators of factor effects that have higher standard errors, and does not enable estimation of interactions among factors; see Chapter 2.

RA may proceed as follows.

1. RA uses the Monte Carlo method (briefly discussed in Section 1.1) to sample a combination of factor values (a scenario) from the joint distribution of possible factor values. (If the factors are assumed to be independent, then this joint distribution is simply the product of the marginal distributions.)

2. RA uses this combination as input into the simulation model of the real system.

3. RA uses the given simulation model to transform this input into output (response), which is also called "propagation of uncertainty'.

4. RA repeats Steps 1 through 3 a number of times (say, 100 or 1,000 times), to obtain an Estimated Distribution Function (EDF) of the response of interest.

5. RA uses the EDF of Step 4 to estimate the probability that is being asked. (Also see the discussion of bootstrapping in Section 3.3.3.)

Personally I was involved in the following applications, using both RA and SA.

- Helton and I report on the probability of leakage of low-radiation nuclear waste; see [204] and also Example 2.7.

- Van Groenendaal and I report on the NPV distribution of an environmental investment in a biogas plant in China. We apply the same RA and SA as I do with Helton. See [388].

- Gaury and I perform RA using an academic simulation model of a production line, to estimate the probability of a managerial disaster (or accident)—for different production pull-control systems such as a Kanban system; see [203] and Section 4.6 below.

Exercise 4.5 *Perform a RA of an M/M/1 simulation, as follows. Suppose that you have available n IID observations on the interarrival time, and on the service time respectively: a_i and s_i $(i = 1, \ldots, n)$. Actually, you sample these values from exponential distributions with parameter $\lambda = \rho$ and $\mu = 1$ where ρ is the traffic rate that you select. Use bootstrapping to sample interarrival times and service times, which you use to estimate the arrival and service rates λ and μ. This pair of estimated rates you use as input to your M/M/1 simulation. In this simulation, you observe the output that you are interested in (e.g., the estimated steady-state waiting time). Perform m replications, to estimate the aleatory uncertainty. Repeat the bootstrap, to find different values for the pair of estimated rates; again simulate to estimate the epistemic uncertainty. Compare the effects of both types of uncertainty.*

I further discuss the similarities and dissimilarities between RA and SA in [187] and [190]; I also refer to [249] and [279].

An *expensive* simulation model requires much computer time per run. RA may then sample, not this expensive simulation model, but its metamodel approximation. For example, [129] uses crude Monte Carlo, Latin Hypercube Sampling (LHS; see Section 4.5.1 below), and orthogonal arrays to sample from specific metamodel types, namely Kriging models and Multivariate Adaptive Regression Splines (MARS). It turns out that the true mean output can be better estimated through sampling many "cheap" values from the metamodel; this metamodel is estimated from relatively few I/O values obtained from the expensive simulation (because that publication estimates an expected value, it does not perform a true RA); also see [106]. Another example is [249], which samples a Kriging metamodel to assess output uncertainty. Kriging is also used for robust design in [220]. The use of Kriging metamodels and Bayesian RA is also briefly discussed in [346].

This RA resembles the *Bayesian* approach, since both approaches assume the parameters of the simulation model to be unknown and assume specific distributions for these parameters. The Bayesian paradigm selects these prior distributions in a more formal way (e.g., so-called conjugate priors), obtains simulation I/O data, and *calibrates* the metamodel's parameters; i.e., it computes the posterior distribution (or likelihood) using the well-known Bayes theorem. Recent references, which include many additional references, are [31], [68], [75], [131], [139], [272], [278], [279], [307], and [416] (also see my own comments in [192]).

Bayesian model averaging formally accounts—not only for the uncertainty of the input parameters—but also for the uncertainty in the form of the (simulation) model itself; see [74], [307], and [416]. Also see *Bayesian melding* in [346].

Sample size determination in Bayesian RA is the focus of [272]; i.e. that publication focuses on the allocation of the limited sampling budget to the various input parameters that can be better estimated when additional data are collected (also see [74]). (For a classic, frequentist approach see again [145] and also [326].)

I think that the Bayesian approach is very interesting, especially from an academic point of view. Practically speaking, however, the classic frequentist RA has been applied many more times (see, e.g., the applications at Sandia).

4.5.1 Latin Hypercube Sampling (LHS)

In 1979, McKay et al. published LHS for the design of experiments with deterministic simulation models or "computer codes"; later on, LHS became so popular that this article was republished in 2000; see [253]. Nowadays, LHS is applied in both deterministic and random simulation experiments, analyzed through a metamodel that is more complicated than a low-order polynomial (examples of such metamodels are Kriging, Bayesian model averaging, etc.).

Popular *software* for LHS is "Crystal Ball", "@Risk", and "Risk Solver", which are add-ins to Microsoft's Excel spreadsheet software; see the software reviews in [149] and [337], and also see

http://www.solver.com/risksolver.htm.

These RA software packages enable crude Monte Carlo and LHS. LHS can also be generated through the MATLAB Statistics toolbox subroutine "lhs" (see [153]), and Sandia's DAKOTA software (see [106], [129] and

http://endo.sandia.gov/DAKOTA)

and the European Commission's Joint Research Center (JRC) SIMLAB software (see [329]) on

http://simlab.jrc.cec.eu.int/.

In their case study (concerning the WIPP, which was also discussed in Example 2.7), [143] finds that crude Monte Carlo and LHS give similar results if the common sample size is large enough. In general, however, LHS is meant to improve results in simulation applications.

Note: Technically, LHS is a type of stratified sampling based on the classic *Latin Square* design, which is a square matrix such that each level of the factor of interest occurs exactly once in each row and each column; the column and row correspond with two factors that are nuisance or block factors (also see the discussion of blocking in Section 2.10). An example with 5 levels is Table 4.1; see [385]. Here factor 1 is the factor of interest, whereas factors 2 and 3 are the nuisance factors. This example requires only $5 \times 5 = 25$ combinations instead of $5^3 = 125$ combinations. (For a discussion of Latin and Graeco-Latin squares, I also refer to [66].) Another Latin square—this time, constructed in a *systematic* way—is shown in Table 4.2. This design, however, may give a biased estimator of the effect of interest. For example, suppose that the factor of interest (factor 1) is wheat, and wheat comes in five varieties. Suppose further that this table determines the way wheat is planted on a piece of land; factor 2 is the type of harvesting machine, and

	factor 3's level				
factor 2's level	1	2	3	4	5
1	1	4	2	5	3
2	4	1	3	2	5
3	3	2	5	4	1
4	2	5	1	3	4
5	5	3	4	1	2

Table 4.1: A Latin square with three factors, each at five levels

	factor 3's level				
factor 2's level	1	2	3	4	5
1	1	2	3	4	5
2	5	1	2	3	4
3	4	5	1	2	3
4	3	4	5	1	2
5	2	3	4	5	1

Table 4.2: A systematic Latin square with three factors at five levels

factor 3 is the type of fertilizer. If the land shows a very fertile strip that runs from north-west to south-east (see the main diagonal in this table), then the effect of wheat type 1 is overestimated. Therefore *randomization* should be applied to protect against unexpected effects. Randomization makes such bias unlikely—but not impossible. Therefore random selection may be corrected if its realization happens to be too systematic (e.g., LHS may be corrected to give a "nearly" orthogonal design; see [165]).

In general, LHS software proceeds as follows (see, e.g., [145]).

1. LHS divides the range of each factor into $n > 1$ mutually exclusive and exhaustive intervals of equal probability. For example, if the distribution of factor values is uniform on $[a, b]$, then each interval has length $(b - a)/n$. However, if the distribution is Gaussian, then intervals near the mode are shorter than in the tails.

2. LHS randomly selects one value for the first factor x_1 from each interval, without replacement. Hence n values are sampled, namely $x_{1;1}$ through $x_{1;n}$.

3. LHS pairs these n values with the n values of the second factor, x_2, randomly without replacement.

4. LHS combines these n pairs with the n values of the third input, x_3, randomly without replacement to form n triplets.

5. And so on, until a set of n k-tuples is formed.

Table 4.3 and the corresponding Figure 4.4 give a LHS example with $n = 5$ combinations of two factors. In the table, each factor has five discrete levels, which are labelled 1 through 5. If the factors are continuous, then the label (say) 1 may denote a value within interval 1; see the figure. Some LHS variations place that value at the middle of the interval instead of sampling its precise value according to the distribution of the factor values.

I point out that in LHS there is no strict mathematical relationship between n (number of factor combinations) and k (number of factors), whereas in classic designs there is such a relationship; e.g., $n = 2^{k-p}$ with $0 \leq p < k$ in a fractional-factorial two-level design (see Chapter 2).

	factor 1's level				
factor 2's level	1	2	3	4	5
1			*		
2		*			
3				*	
4					*
5	*				

Table 4.3: LHS example with five combinations of two factors

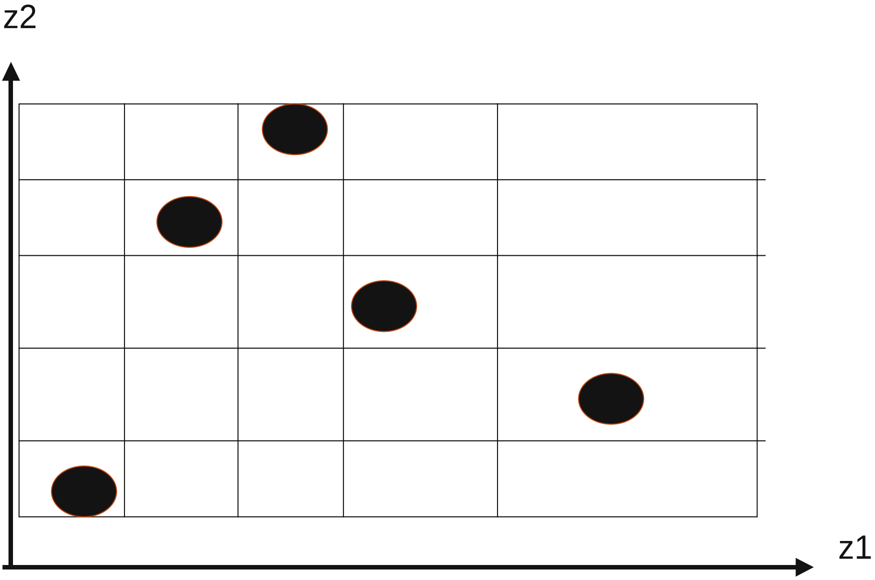

Figure 4.4: LHS example with five combinations of two factors

Obviously, more factor combinations do not hurt. Rules of thumb can be found in the literature; e.g., [144] suggests that $n = 100$ suffices in LHS and crude Monte Carlo sampling. This rule of thumb is supported by the results in [129] for a simple test function (namely the two-dimensional so-called Rosenbrock function).

Above, I mentioned that a factor combination should be sampled from the *joint* distribution for the values of the factors in the experiment. The most popular assumption is that the factors are statistically independent so their joint distribution becomes the product of their individual marginal distributions. The next simplest procedure assumes a multivariate Gaussian distribution, which is characterized by its covariances and means; see (2.51) for the bivariate normal distribution. For nonnormal joint distributions, Spearman's correlation coefficient was discussed in Section 2.11.1. Iman and Conover's procedure uses Spearman's correlation coefficient for LHS and crude Monte Carlo sampling; see [145] and [326].

Note: Helton and his coauthors (see, e.g., [143] and [145]) partition their LHS with sample size $n = 300$ into three subsamples of equal size (namely, 100), to test the stability of RA and SA results. As an alternative, I would suggest bootstrapping; i.e., resampling (without replacement) the original 300 observations. A complication, however, is that these 300 observations are not strictly independent in LHS.

A desirable property of LHS is that if a factor turns out to be unimportant, then the design may still be space filling in the experimental domain for the remaining factors; i.e., *projecting* an LHS point (combination) in the original k-dimensional space onto any axis gives a uniform spacing. Such a design is also called *noncollapsing*; i.e., when an unimportant factor is eliminated, no points become identical. However, projections onto two or more dimensions may give "bad" designs, so standard LHS is further refined. For details, I refer to [29], [158], [233], [365], and [408]; for projection properties of non-LHS designs see [381].

Besides LHS there are many other design types, such as minimax, maximin designs and orthogonal arrays; see the recent articles [66], [153], [305], and [386], the classic textbook [333], the recent dissertations [157] and [365], and the websites

http://lib.stat.cmu.edu

and

http://www.spacefillingdesigns.nl/.

These designs are also popular in Kriging, discussed in Chapter 5.

4.6 Robust optimization: Taguchian approach

The practical importance of Robust Optimization is emphasized in the 2002 panel report [355] (also see [155]). Indeed, I think that robustness is crucial, given today's increased complexity and uncertainty in organizations and their environment.

My approach to Robust Optimization is inspired by *Taguchi*'s view, but I do not use his techniques. Taguchi is a Japanese engineer and statistician; see his 1987 book [376], the more recent book [402], and
http://en.wikipedia.org/wiki/Genichi_Taguchi.
His techniques (which I do not use) include certain experimental designs; e.g., "orthogonal arrays" (see [1], [242], and [265]). His view distinguishes between the following two types of factors:

- Decision (or control) factors
- Environmental (or noise) factors

The first type of factors are under the control of the users; e.g., in a queuing problem the number of servers and their service rates may be controllable; in inventory management, the reorder levels and order quantities may be controllable. The second type of factors are not controlled by the users; examples may be the arrival rate of customers in a queuing system, and demand and lead times in inventory management. In practice, the controllability of a factor depends on the specific situation; e.g., the users may change the customer arrival rate through an advertising campaign. More examples of controllable and environmental factors will follow in the case study presented in Section 4.6.1.

Note: Other authors distinguish between environmental uncertainty (e.g., demand uncertainty) and system uncertainty (e.g., yield uncertainty); see [266].

Implementation errors may also be a source of uncertainty. These errors occur whenever recommended (optimal) values of controllable factors are to be realized in practice; see [366] and also [365]. Continuous values are hard to realize in practice, since only limited accuracy is then possible. For example, in a simple inventory model the optimal order quantity may turn out to be the square root of some expression; in practice, however, only a discrete number of units may be ordered.

Besides implementation errors, there are validation errors of the simulation model (compared with the real system) and the metamodel (compared with the simulation model); see [211]. The search for an "optimal" solution should also account for these errors; see [366].

Ben-Tal and Nemirovsky present an interesting theory for Robust Mathematical Programming; see [34]. They assume that all values of the environmental factors are equally likely in a given area; that area may be a multidimensional box or ellipsoid. In other words, they focus on the worst case within this area. Their solution turns out to be much better than the standard solution. Stinstra and Den Hertog apply that theory to linear regression and Kriging metamodels of deterministic simulation models; again see [366]. So, Robust Mathematical Programming finds elegant (tractable) solutions for its type of robustness problems; also see the recent overview by Bertsimas and Thiele in [37] (nonuniform input distributions are the focus of another paper by Bertsimas et al.; see [36]). Its models, however, are neither dynamic nor stochastic, whereas random simulation models are (by definition). Moreover, the approach assumes that all points inside the box or ellipsoid are equally likely and important, whereas a point just outside that area is completely unimportant. I assume a distribution function (e.g., a Gaussian distribution) that decreases the likelihood and importance smoothly to zero—as points are farther removed from the most likely scenario.

Note: The goal of Robustness Analysis (discussed in this section) is the design of robust products or systems, whereas the goal of Risk Analysis (discussed in Section 4.5 above) is to quantify the risk of a given design; that design may turn out to be not robust at all. Robustness Analysis should result in (for example) reengineered "flexible", "agile" or "resilient" supply chains; see [79] and Section 4.6.1 below. In the section on Risk Analysis, I also discussed the *Bayesian* approach. The latter approach may also be used for Robust Optimization; see [307] and [308]. The Taguchian approach is related to the *six sigma* approach; see [220] and [221].

Whereas optimization is a "hot" topic in simulation, Robust Optimization is neglected—except for a few publications; see [3], [66], [330], [347], and [380], which reference several more simulation studies using Taguchi's methods. Taguchi's approach is combined with a Genetic Algorithm (GA) in [4]. Evolutionary heuristics combined with penalty functions to avoid constraint violations are discussed in [409]. I, however, will focus on RSM in the remainder of this section (because I expect that RSM requires relatively few runs with the—possibly expensive—simulation model.)

In [203], Gaury and I derive an optimal solution assuming a specific—namely the most likely—combination of environmental factor values. Next, we estimate the robustness of this solution when the environment changes; technically, we generate these combinations through LHS. In this section, however, I wish to find a solution that—from the start of the analysis—accounts for all possible environments, including their likelihood (see below). For example, I wish to select the reorder level (say) s and the order-up-to level S accounting for the probabilities of different values for demand D—not only the most likely value (estimated from past demand data); see (1.9). In other words, whereas Gaury and I perform *ex post* robustness analysis, I now wish to perform an *ex ante* analysis.

As I mentioned above, I use Taguchi's view but not his statistical methods. My reason is that simulation experiments enable the exploration of many more factors, factor levels, and combinations of factor levels than real-life experiments do. Taguchi and his followers focus on real-life (not simulated) experiments—for designing robust products (not complete production systems such as supply chains). Moreover, I do not use a Taguchian scalar output (such as the signal/noise ratio); instead I allow a vector of multiple outputs, using a Mathematical Programming approach (which minimizes one output, while satisfying the constraints for the remaining outputs).

My analysis continues a recent research project in which my co-authors and I applied robustness analysis to a supply chain of the Ericsson company in Sweden; see [194] and also Section 4.6.1 and Chapter 6. For the controllable factors we used a second-order polynomial regression metamodel. This metamodel serves as a quick (inexpensive) predictive model for optimization. (An alternative metamodel would be a Kriging model, which has been used by other authors for optimization; see Section 4.1 and Chapter 5.) For the environmental factors we generate combinations of environmental factor values through LHS. We use this LHS to quantify the variability of the simulation output. (LHS is typically a technique not used by Taguchians.)

Technically, I consider both the expected simulation output and the output's variance caused by environmental disturbances (like Taguchians, I assume no input and output constraints—unlike Section 4.3 on GRSM). The expected value and variance can be managed through the controllable factors. Unlike Taguchians, I do not propose to combine the mean and variance into a single criterion (using the signal/noise ratio); i.e., I think that a *Taguchian loss function* is too restrictive.

So the (say) k controllable factors are to be optimized. Inspired by RSM, I propose to approximate the local I/O behavior of the simulation model through a second-order polynomial in these factors—once the search for the optimum seems to have reached the area of the true optimum. To estimate the coefficients of this polynomial, classic RSM uses a Central Composite Design (CCD). (Also see again Section 4.2.)

Regarding the environmental factors, I am *not* interested in their functional relationship with the output; i.e., I do not wish to estimate (say) a low-order polynomial in these factors. Following Taguchi, I consider these factors as noise. Unlike Taguchi, I propose to use LHS to sample (say) n environmental factor combinations. Unlike a CCD, LHS does not impose a relationship between the number of environmental scenarios and the number of environmental factors (see Section 4.5.1 above). If there is no apriori information about the likelihood of the environmental factor values, then I propose to assume independent uniform distributions per environmental factor (I also refer to Bayesian uninformative prior distributions; again see Section 4.5).

Next, I propose to use one of Taguchi's design techniques; i.e., *cross* (or combine) the *inner array*—namely, the CCD for the controllable factors—with the *outer array*—the LHS design for the environmental factors; see Table 4.4, which implies that the total number of scenarios simulated is $n_{CCD}n_{LHS}$. (Instead of such an approach, the controllable and the environmental factors may be combined in a single design that enables the estimation of a low-order polynomial in both types of factors; see [223].)

Whereas classic optimization assumes a single scenario (e.g., the most likely scenario), I estimate the parameters in the polynomial from the CCD simulation outputs averaged over all simulated LHS scenarios.

In Robustness Optimization, an important characteristic is the output variability (besides the output mean). Taguchians often quantify this variability through the variance; an alternative may be the standard deviation or (as Gaury and I did) the probability of a specific disaster happening. Anyhow, I propose to model the estimated mean and variability as two separate second-order polynomials in the controllable factors.

To evaluate the reliability of the Robust Optimization solution, I propose to apply bootstrapping (obviously, the estimated solution is a nonlinear function of the simulation output so standard confidence intervals do not hold). However, I leave this bootstrapping for future research. Also see Section 3.3.3.

	LHS			
CCD	1	2	...	n_{LHS}
1				
2				
...				
n_{CCD}				

Table 4.4: A cross design combining CCD and LHS

In summary, to maximize the mean simulation output, the analysts should select specific values for the controllable factors. To minimize the variance of the simulation output, however, they may have to select other values. The final decision is up to management; they should select compromise values depending on their risk attitude. Note that [347] (p. 3837) also uses plots to decide on a compromise solution; also see Figure 4.5 where the horizontal double-pointed arrows denote the (bootstrap) confidence intervals for the optimal solutions for the mean and variance respectively (which do not overlap in this example).

The resulting "robust"solution may be compared with the "classic" optimum solution for the controllable factors; the latter solution assumes a single scenario, namely the base scenario. More precisely, the mean and variance of the simulation output for the robust solution and the classic solution may be estimated. These estimates may be computed from new environmental scenarios; i.e., the old LHS values are replaced by new samples (the old scenarios would favor the robust solution, since this solution uses estimates based on these scenarios). I expect that these results will show that risk considerations do make a difference!

In future research, Robust Simulation Optimization may be extended to multiple simulation outputs. I have already discussed GRSM for Simulation Optimization with constraints for the simulation inputs and outputs, in Section 4.3.

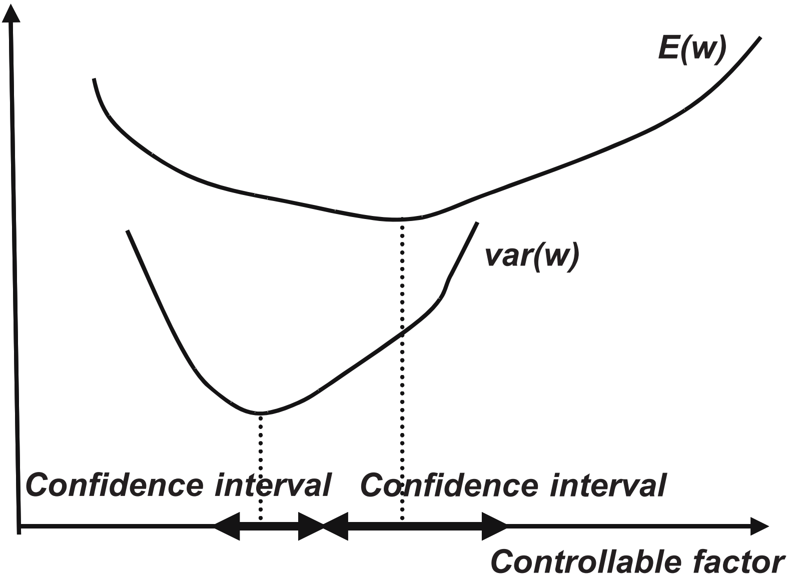

Figure 4.5: Example of Robust Optimization

4.6.1 Case study: Ericsson's supply chain

In this section, I demonstrate "robustness" through a case study, simulating three supply chain configurations of Ericsson's mobile communications industry in Sweden (these three configurations concern the past, present, and future situations; see [282]). Unfortunately, my co-authors and I could not finish this case study so I cannot report definitive results.

Note: The supply chain literature distinguishes between robustness and flexibility. A flexible supply chain can react to a changing environment by adapting its operations (see [414]). A robust supply chain keeps its design fixed, but can still accommodate many changes in its environment. So the two concepts focus on operational and strategic decisions respectively.

The newer the supply chain design is, the fewer operations and tests that configuration has; i.e., newer designs are "lean and mean". A crucial environmental factor is process yield, which is the percentage of products that passes a test. A defective product is sent to a repair unit, which decides whether to repair the product or to scrap it.

Each of these three supply chain designs is simulated (these three simulation models are programmed in the Taylor II simulation software; see [160]). The simulation models include buffers (inventories), located before and after every test station and operation. Products are transported between all operations and test stations. More details are given in [195] and [292].

Controllable factors concern the manufacturing processes, logistic partners for transportation, etc. Environmental factors are demand for products, process yield, and scrap percentage at each test station. My coauthors and I focus on a single output, namely the total weekly costs for the whole supply chain.

Note: We assume that management is interested in steady-state output (transient output would be relevant in short-term operational control; our study, however, concerns strategic design decisions). We select a warm-up period of four weeks, and an additional run length of sixteen weeks.

Originally there were 92 factors, but after a screening experiment (see Chapter 6) only three *controllable* factors remain (all three factors concern transportation, namely, internal transportation within the circuit board factory, transportation between factories, and transportation between Surface Mounted Devices and test stations). After this screening, six important *environmental* factors remain (namely the demand for the product and the yields at five different stations).

Because we wish to optimize these three controllable factors, we use a second-order polynomial for these factors in the experimental area (the factors typically change by only 5% of their base values). To estimate the coefficients of this polynomial, we use the following reduced CCD (we reduce the CCD because the simulations take much computer time; e.g., the whole experiment for one of the three simulation models takes 42 hours on a Pentium II 600 MHz PC):

1. To estimate the main effects (first-order effects) and two-factor interactions (cross-products), we use a 2^3 full factorial design.

2. To estimate the purely quadratic effects, we need at least three values per controllable factor. We decide to simulate the "axial" factor combinations with coded values -0.5 (besides the values -1 and $+1$ of the 2^3 design). So—unlike a classic CCD—we use only the "lower" half of the star design. The value -0.5 is rather arbitrary (a popular value is $-\sqrt{k} = -\sqrt{3} = -1.7$). We select -0.5 instead of $+0.5$, because we expect that this choice decreases the costs: we have reasons to assume that all main effects are nonnegative (also see Chapter 6).

3. We also simulate the base scenario (the center of the experimental area).

To sample combinations of the six environmental factors, we use LHS with a sample size of ten. Because we have no *a priori* information about the likelihood of the factor values, we assume independent uniform distributions for each factor.

Note: To this LHS sample, my co-authors and I add two extreme scenarios, namely an optimistic scenario (all factors at their lowest values) and a pessimistic scenario (all factors at their highest values). These scenarios give "extreme" outputs; i.e., their outputs straddle the outputs for the ten LHS scenarios.

Next we cross the "inner array"—our reduced CCD for the controllable factors—with the "outer array"—our LHS design for the environmental factors.

Note: Our crossed design is conceptually related to the design in [396]. However, we cross a reduced CCD and a LHS design, whereas the latter publication crosses factorial designs with all controllable factors at three levels, and lattice points for all environmental factors.

We do not *replicate* the whole crossed design, because we find that the estimated standard error computed from four replicates at the center point (with different PRNs) is ten times smaller than the standard error estimated from the ten LHS environmental scenarios combined with the center point; see again Table 4.4.

When we try to optimize the controllable factors, we account for the *box constraints* on these factors; see (4.7). These constraints may be binding; i.e., the optimal values may indeed turn out to lie on the border of the experimental area. That border, however, is fixed rather arbitrarily (namely, to changes of 5%), so we may reconsider these constraints.

Technically, we account for these constraints through Lagrange multipliers, which quantify the shadow prices of the constraints. Indeed, we find that the three controllable factors have nonzero shadow prices (–133,162, -307,731, and –208,537 respectively). These shadow prices are negative, because the mean costs decrease as the constraints are made less tight.

We also estimate the output variance (instead of the mean) as a second-order polynomial function of the controllable factors. Using these estimated effects, we find that one controllable factor again minimizes this output (estimated variance, not mean) at its lower boundary (which has a coded value of -1), whereas optimal values for the other two factors are –0.13 and –0.65. The Lagrange multiplier for the former factor is 22,215 (of course, only the constraint for this factor has a non-zero shadow price).

Our preliminary conclusion is that both the mean and the variance of the simulation output (namely, costs) are minimized by selecting the minimum value for the first controllable factor. The other two controllable factors, however, have conflicting optimal values when considering both outputs (mean and variance). However one of these two factors has estimated optimal values –1 and –0.65, so maybe the true optimal values are the same? To estimate the accuracy of our estimated optimal values, we might derive a (bootstrap) confidence region. Management may then use the (bootstrap) confidence region to select a robust solution; see again Figure 4.5.

Note: Comparing the robust solutions for the three supply chains shows that the future supply chain gives the lowest expected value and variance for its costs.

4.7 Conclusions

In this chapter, I first summarized classic RSM, assuming a single response variable. I added the Adapted Steepest Ascent (ASA) search direction, which improves the classic direction.

Next, I summarized GRSM for simulation with a multivariate response, assuming that one univariate response is to be minimized while all the other responses must meet given constraints. Moreover, the (deterministic) inputs must satisfy given box constraints.

Then, I summarized a procedure for testing whether an estimated optimum is truly optimal—using the KKT conditions. This procedure combines classic t and F tests with bootstrapped tests.

Next, I discussed RA.

Finally, I discussed Robust Optimization, focusing on a Taguchian approach.

4.8 Solutions for exercises

Solution 4.1 $(z_1^o, z_2^o) = (-5, 15)$*; also see [12].*

Solution 4.2 *If* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*,then (4.2) implies* $\mathbf{C} = \mathbf{I}/N$*. Hence, (4.4) does not change the steepest descent direction.*

Solution 4.3 *The ratio of two normal variables has a Cauchy distribution so its expected value does not exist; its median does.*

Solution 4.4 $(z_1^o, z_2^o) = (1.24, 0.52)$*; also see [12].*

Solution 4.5 *The results depend on your choice of* n*, etc.*

5

Kriging metamodels

This chapter is organized as follows. In Section 5.1, I introduce Kriging (the name refers to the South African mining engineer Krige; Kriging is also called spatial correlation modeling). In Section 5.2, I present the basic Kriging assumptions and formulas. In Section 5.3, I present some relatively new results, including Kriging for random simulation and estimating the true variance of the Kriging predictor through bootstrapping. In Section 5.4, I discuss one-shot designs such as Latin Hypercube Sampling (LHS) and sequentialized, customized designs. In Section 5.5, I present conclusions.

5.1 Introduction

In the preceding chapters, I focussed on *low-order polynomial regression* metamodels. Such metamodels are fitted to the Input/Output (I/O) data of the local or global experiment with the underlying simulation model. These metamodels may be used for the explanation of the simulation model's behavior, and for prediction of the expected simulation output for combinations of factor values (scenarios) that have not yet been simulated. The final goals of the metamodel may be validation of the simulation model, Sensitivity Analysis, and Robust Optimization.

In the present chapter, I focus on *Kriging* metamodels. Typically, Kriging models are fitted to data that are obtained for larger experimental areas than the areas used in low-order polynomial regression metamodels; i.e., Kriging models are *global* rather than local. These models are used for prediction; the final goals are Sensitivity Analysis and Robust Optimization.

Kriging was originally developed in *geostatistics* (also known as spatial statistics) by Krige. The mathematics were further developed by Matheron; see his 1963 article [251]. A classic geostatistics textbook is Cressie's 1993 book [86], which has 900 pages. A more recent textbook was published in 1999; see [363]. I also mention the references 17 through 21 in [248].

Later on, Kriging models were applied to the I/O data of *deterministic simulation* models. These models have k-dimensional input where k is a given positive integer (whereas geostatistics considers only two or three dimensions); see the classic article [322] published in 1989 by Sacks et al. More recent publications are Jones et al.'s 1998 summary article [168], Simpson et al.'s 2001 article [356], and Santner et al.'s 2003 textbook [333].

Only recently, Kriging has also been applied to *random simulation* models; see my 2003 article with Van Beers [383]. Although Kriging in random simulation is still rare, I strongly believe that the track record Kriging achieved in deterministic simulation holds promise for Kriging in random simulation! Also see the 2007 paper that I wrote together with three coauthors; [42].

Note: Searching for "Kriging" via Google (on February 15, 2007) gave 631,000 hits, which illustrates the popularity of this mathematical method. Searching for "Operations Research" within these pages gave 81,000 hits.

5.2 Kriging basics

I start with highlighting the differences between linear regression—especially low-order polynomial regression—and Kriging models. So, I repeat a few formulas from the previous chapters. I again present my general black-box representation (2.6), but now I limit myself to a single (univariate, scalar) simulation output because most Kriging models also assume such output:

$$w = s(d_1, \ldots, d_k, \mathbf{r}_0) \tag{5.1}$$

where

w is the output of the underlying simulation model;

$s(.)$ denotes the mathematical function implicitly defined by the computer code implementing this simulation model;

d_j with $j = 1, \ldots k$ is the j^{th} input variable (factor) of the simulation program, so $\mathbf{D} = (d_{ij})$ is the design matrix for the simulation experiment, with $i = 1, \ldots, n$ and n the number of factor combinations in that experiment,

$\mathbf{r}_0$ is the vector of PseudoRandom Number (PRN) seeds (which vanishes in deterministic simulation).

Remember that $\mathbf{D}$ determines the original input variables z and the corresponding standardized input variables x. The design matrix $\mathbf{D}$ is usually standardized; e.g., a two-level (fractional) factorial has elements that are either -1 or $+1$; also see (2.5).

In practice, a simulation model has multiple outputs, and univariate Kriging is applied to each output independently.

The *first-order polynomial* regression metamodel for (5.1) is

$$y_{reg} = \beta_0 + \beta_1 d_1 + \ldots \beta_k d_k + e_{reg} \tag{5.2}$$

where

y_{reg} is the metamodel predictor of the simulation output w in (5.1); I now add the subscript *reg* to distinguish this metamodel from the Kriging metamodel in this chapter;

$\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ is the vector with the parameters of this metamodel;

e_{reg} is the residual or noise—which includes both lack-of-fit of the metamodel and intrinsic noise (caused by the PRNs).

The general *linear regression* model was given in (2.10) and is repeated here:

$$\mathbf{y}_{reg} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}_{reg} \tag{5.3}$$

where

$\mathbf{y}_{reg}$ denotes the n-dimensional vector with the regression predictor;

$\mathbf{X} = (\mathbf{x}_{ij})$ denotes the $n \times q$ matrix of explanatory regression variables with $\mathbf{x}_{ij}$ the value of variable j in combination i ($i = 1, \ldots, n; j = 1, \ldots, q$) (e.g., (5.2) implies $q = 1 + k$ including the dummy variable or constant $x_{i0} = 1$ corresponding with β_0);

$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)'$ denotes the q-dimensional vector of regression parameters (if there is a dummy variable, then β_1 denotes the intercept in the general regression model, whereas the symbol β_0 denoted the intercept in the specific regression model in (5.2));

$\mathbf{e}_{reg}$ is the vector of residuals in the n combinations.

The Least Squares (LS) estimator (say) $\widehat{\boldsymbol{\beta}}$ of the regression parameter vector $\boldsymbol{\beta}$ in the linear regression model (5.3) can be derived to be

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} \tag{5.4}$$

where $\mathbf{w} = (w_1, \ldots, w_n)'$ is the n-dimensional vector with "the" output of the simulation model with input $\mathbf{D} = (d_{ij})$; "the" output of combination i is the average output of a constant number of replications, $m_i = m$:

$$\overline{w_i} = \frac{\sum_{r=1}^{m} w_{ir}}{m}. \tag{5.5}$$

Obviously, deterministic simulation implies $m = 1$.

Hence, the regression estimator for a simulation input (say) $\mathbf{d} = (d_1, \ldots, d_k)'$ is

$$\widehat{\mathbf{y}}_{reg}(\mathbf{d}) = \mathbf{x}(\mathbf{d})'\hat{\boldsymbol{\beta}} = \mathbf{x}(\mathbf{d})'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} \tag{5.6}$$

where the vector of explanatory variables $\mathbf{x}(\mathbf{d})$ is determined by the vector of simulation inputs $\mathbf{d}$; e.g., the first-order polynomial model (5.2) implies $\mathbf{x}(\mathbf{d}) = (1, d_1, \ldots, d_k)'$. The input $\mathbf{d}$ may be a new or an old combination (the old combination is one of the rows in $\mathbf{D}$).

I focus on the simplest type of Kriging called *Ordinary Kriging*, which assumes

$$w(\mathbf{d}) = \mu + \delta(\mathbf{d}) \tag{5.7}$$

where

μ is the simulation output averaged over the experimental area;

$\delta(\mathbf{d})$ is the additive noise that forms a stationary covariance process with zero mean (also see Definition 3.2).

Note that this metamodel with its constant μ does not imply a *flat* response surface; see [322]. Instead of the constant μ in (5.7), *Universal* Kriging uses a regression model. However, Ordinary Kriging often suffices in practice; see [65], [247], [248]), and [322].

Kriging is used—quite successfully—in deterministic simulation. At first sight it may seem strange that the *random* (meta)model (5.7) can be applied to a *deterministic* simulation model. My interpretation is that the deviations of the simulation output w from its mean μ form a random process—with the characteristics of a "stationary covariance process" (with zero mean); see δ in (5.7).

Ordinary Kriging—which from now on, I briefly call Kriging—uses the following *linear* predictor:

$$y(\mathbf{d}) = \boldsymbol{\lambda}(\mathbf{d}, \mathbf{D})'\mathbf{w}(\mathbf{D}) = \boldsymbol{\lambda}'\mathbf{w} \tag{5.8}$$

where the weights $\boldsymbol{\lambda}(\mathbf{d}, \mathbf{D})$—abbreviated to $\boldsymbol{\lambda}$—are not constants (whereas $\boldsymbol{\beta}$ in (5.3) remains constant) but decrease with the *distance* between the input $\mathbf{d}$ to be predicted and the "old" points $\mathbf{D}$ (remember that the $n \times k$ design matrix $\mathbf{D} = (d_{ij})$ has already been defined below (5.1)). This $\mathbf{D}$ determines the simulation output vector $\mathbf{w}$, so the explicit notation is $\mathbf{w}(\mathbf{D})$ and the simpler notation is $\mathbf{w}$.

To select the *optimal* values for the weights $\boldsymbol{\lambda}$ in (5.8), a criterion must be selected. In linear regression, the Sum of Squared Residuals is the criterion—which gives the LS estimator (5.4). Kriging selects the *Best Linear Unbiased Predictor (BLUP)*, which (by definition) minimizes the Mean Squared Error (MSE) of the predictor:

$$\min_{\boldsymbol{\lambda}} MSE[y(\mathbf{d})] = \min_{\boldsymbol{\lambda}}[E\{y(\mathbf{d}) - w(\mathbf{d})\}^2] \tag{5.9}$$

where $\mathbf{d}$ may be any point (factor combination) in the experimental area. Moreover, this minimization must account for the condition that the predictor be *unbiased*:

$$E[y(\mathbf{d})] = E[w(\mathbf{d})]. \tag{5.10}$$

Obviously, in deterministic simulation $E[w(\mathbf{d})]$ reduces to $w(\mathbf{d})$. It can be proven that the solution of the constrained minimization problem defined by (5.9) and (5.10) implies that the weights of the linear predictor (5.8) must satisfy the following condition:

$$\sum_{i=1}^{n} \lambda_i = 1 \tag{5.11}$$

or (in matrix notation) $\mathbf{1}'\boldsymbol{\lambda} = 1$ where $\mathbf{1} = (1, \ldots, 1)'$ is an n-dimensional vector with each element being the value 1.

Furthermore, the *optimal* weights can be proven to have the values

$$\boldsymbol{\lambda}_o = \boldsymbol{\Gamma}^{-1}[\boldsymbol{\gamma} + \mathbf{1}\frac{1 - \mathbf{1}'\boldsymbol{\Gamma}^{-1}\boldsymbol{\gamma}}{\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1}}] \tag{5.12}$$

where

$\boldsymbol{\Gamma} = (cov(w_i, w_{i'}))$ with $i, i' = 1, \ldots, n$ is the $n \times n$ symmetric and positive semi-definite matrix with the covariances between the "old" outputs (i.e., outputs of input combinations that have already been simulated);

$\boldsymbol{\gamma} = (cov(w_i, w_0))$ is the n-dimensional vector with the covariances between the n "old" outputs w_i and w_0, the output of the combination to be predicted (which may be "new" or "old").

Note: Many publications use the symbol $\mathbf{R}$ instead of $\boldsymbol{\Gamma}$; I use Greek letters to denote unknown parameters (such as the covariances).

Finally, it can be proven (see, e.g., [237]) that (5.7), (5.8), and (5.12) imply

$$y(\mathbf{d}) = \widehat{\mu} + \boldsymbol{\gamma}(\mathbf{d})'\boldsymbol{\Gamma}^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}) \tag{5.13}$$

with

$$\widehat{\mu} = (\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1})^{-1}\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{w}. \tag{5.14}$$

and $\mathbf{d}$ denoting the input of the output w_0 that is to be predicted.

Exercise 5.1 *Derive the expected value of the Kriging predictor defined in (5.13).*

Exercise 5.2 *Is the conditional expected value of the predictor (5.13) smaller, equal, or larger than the unconditional mean μ if that condition is as follows: $w_1 > \mu, w_2 = \mu, \ldots, w_n = \mu$?*

Note that $\nabla(y) = (\partial y/\partial d_1, \ldots, \partial y/\partial d_k)$—the *gradient* of the Kriging predictor—follows from (5.13) and (5.14), where $\boldsymbol{\gamma}$ is a function of the input $\mathbf{d} = (d_1, \ldots, d_k)'$ for the output w_0. Gradients are used in simulation optimization. In Exercise 5.5, I shall return to this gradient, after introducing specific covariance functions such as the Gaussian function.

Obviously, the optimal values for the Kriging weights in (5.12) depend on the covariances—or equivalently the correlations—between the simulation outputs in the Kriging model (5.7) (for the correlation concept see (2.51)).

Kriging assumes that these correlations are determined by the "distance" between the inputs of the outputs w_i and $w_{i'}$ or w_i and w_0—or (expressed more succinctly) between w_i and w_g with $g = 0, 1, \ldots, n$.

In geostatistics, Kriging often uses the *Euclidean distance* (say) h between these simulation inputs:

$$h = \|\mathbf{h}\|_2 = \|\mathbf{d}_0 - \mathbf{d}_i\|_2 \tag{5.15}$$

where $\|\|_2$ denotes the L_2 norm; $\mathbf{d}_0$ is the input of the "new" simulation output w_0 (to be predicted), and $\mathbf{d}_i$ is the input of the "old" simulation outputs that have already been simulated. So, $cov(w_0, w_i) = \sigma(h)$. The analogue holds for the correlations between the "old" outputs themselves.

In simulation, however, Kriging assumes that the correlation function for a k-dimensional input vector is the *product* of k one-dimensional functions:

$$\rho(w(\mathbf{d}_i), w(\mathbf{d}_g)) = \prod\nolimits_{j=1}^{k} \rho(d_{ij}, d_{gj}). \tag{5.16}$$

Moreover, Kriging assumes a *stationary covariance process*, which implies that the correlations depend only on

$$h_j(i, g) = |d_{ij} - d_{gj}| \ \ (j = 1, \ldots, k) \ (i = 1, \ldots, n)(g = 0, 1 \ldots, n). \tag{5.17}$$

So, $\rho(d_{ij}, d_{gj})$ in (5.16) reduces to $\rho(h_j(i, g))$. I point out that transforming the standardized design points d_j into the original simulation inputs z_j makes the distances scale dependent; also see [85].

Note: Instead of the correlation function, geostatisticians use the *variogram*—which quantifies the same information because it equals the (constant) variance $\sigma_0 = \sigma^2$ minus the (decreasing) covariance function.

There are several types of stationary covariance processes; Figure 5.1 illustrates three popular shapes for a single input so $h_j = h$ in (5.17) with parameter $\theta > 0$ \:

- Linear correlation function: $\rho(h) = \max(1 - \theta h, 0)$
- Exponential correlation function: $\rho(h) = \exp(-\theta h)$
- Gaussian correlation function: $\rho(h) = \exp(-\theta h^2)$ (its point of inflection can be proven to be $1/\sqrt{2\theta}$).

Note: Even a simulation textbook as old as the 1966 book [270], pp. 118–121 discusses the first two types, assuming h denotes the "lag" in a time series; see Definition 3.2).

In Kriging, a popular correlation function (with h and $\mathbf{h}$ defined in (5.15)) is

$$\rho(\mathbf{h}) = \exp[-\sum_{j=1}^{k} \theta_j h_j^{p_j}] = \prod\nolimits_{j=1}^{k} \exp[-\theta_j h_j^{p_j}] \tag{5.18}$$

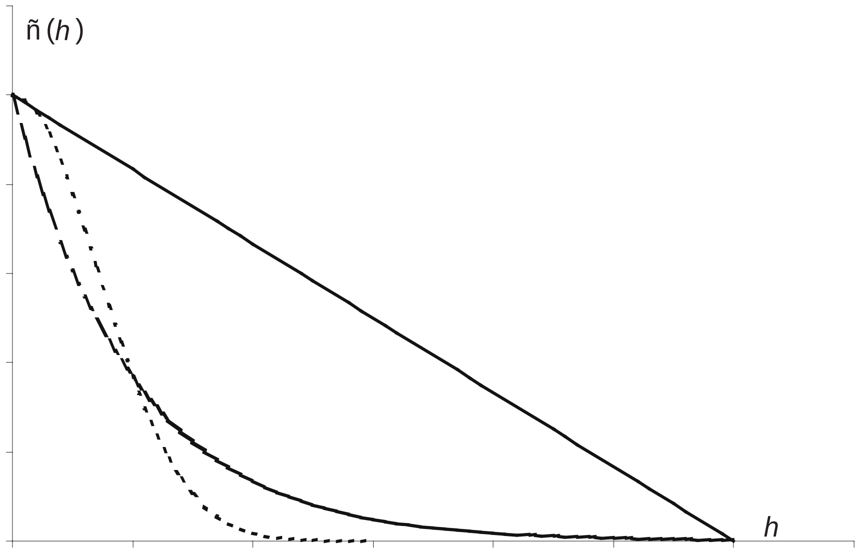

Figure 5.1: Three types of correlation functions that depend on distance h

where

θ_j denotes the importance of factor j; i.e., the higher θ_j is, the less effect input j has;

p_j denotes the smoothness of the correlation function; e.g., $p_j = 2$ implies an infinitely differentiable function. Figure 5.1 has already illustrated an exponential and a Gaussian function, which have $p = 1$ and $p = 2$ respectively.

Exercise 5.3 *What is the value of $\rho(h)$ in (5.18) with $p > 0$, when $h = 0$ and $h = \infty$ respectively?*

Exercise 5.4 *What is the value of θ_k in (5.18) with $p_k > 0$, when input k has no effect on the output?*

Exercise 5.5 *Suppose there is a single input (so $\mathbf{d}$ becomes d) and a Gaussian correlation function. Derive the gradient $\nabla(y) = (\partial y/\partial d)$.*

Correlation functions that decrease as the distance increases, imply that the optimal weights are relatively high for inputs close to the input to be predicted. Furthermore, some of the weights may be *negative*. Finally, the weights imply that for an "old" input (so $\mathbf{d}$ is a row within $\mathbf{D}$) the predictor equals the observed simulation output at that input:

$$y(\mathbf{d}_i) = w(\mathbf{d}_i) \text{ if } \mathbf{d}_i \in \mathbf{D}, \tag{5.19}$$

so all weights are zero except the weight of the observed output. This property implies that the Kriging predictor is an exact *interpolator*, whereas the regression predictor minimizes the Sum of Squared Residuals (SSR) so it is not an exact interpolator—unless $n = q$; see (2.11). In (5.9), $y(\mathbf{d}) - w(\mathbf{d})$ may be replaced by the residual $e(\mathbf{d})$, which is the analogue of $\mathbf{e}_{reg}$ in (5.3). Because of (5.19), the residuals observed at the old design points are exactly zero: $e_i = 0$ $(i = 1, \ldots, n)$.

A major problem is that the optimal Kriging weights λ_i depend on the correlation function of the underlying simulation model—*but this correlation function is unknown*. Therefore both the type (see Figure 5.1) and its parameter values must be estimated. Estimators for covariances have already been shown in (3.30). Now, however, the number of observations for a covariance of a given distance h decreases as that distance increases. Given these estimates for various values of h, a correlation function (such as the ones in Figure 5.1) is fitted. To estimate the parameters of such a correlation function, the standard software and literature uses Maximum Likelihood Estimators (MLEs). A MLE requires constrained maximization. This optimization is a hard problem, because matrix inversion is necessary, multiple local maxima may exist, etc.; see [246] and [248].

Note: Besides the MLE criterion, [248] uses cross-validation. For the linear correlation function, Van Beers and I use the LS criterion because this criterion gives a simpler estimator; see [213].

For the estimation of the correlation functions and the optimal weights through (5.12), my coauthors and I have been using the MATLAB Kriging toolbox DACE, which is free of charge; see [239]. Alternative free software is available via

http://www.stat.ohio-state.edu/˜comp_exp/

and

http://endo.sandia.gov/Surfpack.

If the number of simulation inputs does not exceed three, then geographical Kriging software can also be applied. An example of commercial geographical software is Isatis; see

http://www.geovariances.com/.

Unfortunately, the DACE software uses lower and upper limits for θ_j (the correlation parameters), which the analysts usually find hard to specify. Different limits may give completely different $\widehat{\theta_j}$ (MLE); see the examples in [237].

Note: There are also many publications that interpret Kriging models in a *Bayesian* way; a recent article is [139]; also see [65], [66], [246], and some references below.

Note: Kriging seems related to Moving Least Squares (MLS), which is described in, e.g., [379] and [382]. MLS fits regression models *locally* with higher *weights* given to nearby data points. The weight function seems related to the stationary covariance function used in Kriging. Like Kriging, MLS uses coefficients that change with the point to be predicted. These relationships deserve further research.

5.3 Kriging: new results

The interpolation property in (5.19) is attractive in *deterministic* simulation, because the observed simulation output is unambiguous (ignoring numerical noise that may occur when deterministic simulation software is executed; see [379]). In *random* simulation, however, the observed output is only one of the many possible values. In [383], Van Beers and I study random simulation and replace $w(\mathbf{d}_i)$ in (5.9) by the average observed output, which was also defined in (2.27) as

$$\overline{w_i} = \frac{\sum_{r=1}^{m_i} w_{ir}}{m_i} \; (i = 1, \ldots, n). \tag{5.20}$$

These n averages, however, are still random, so the property in (5.19) loses its intuitive appeal. Nevertheless, Kriging may be attractive in random simulation because it may decrease the prediction bias (and hence the MSE) at input combinations close together. In [383], we give examples of Kriging predictions based on (5.20) that are much better than the regression predictions. (Regression metamodels may be useful for other goals such as understanding, screening, and validation; see Section 1.2.)

Note: Santner et al's 2003 textbook [333] has an appendix with a computer program in C, which is called PErK and allows random output. When using PErK with the "RandomError = Yes" option in the job file, this software includes a white noise term in the Kriging model. The Kriging predictor is then no longer an exact interpolator. I have not yet applied this program, neither do I know any applications in random simulation. Also see [353].

The Kriging model in (5.7) assumes a stationary covariance process, which implies a constant variance (say) σ_δ^2. However, in experiments with random simulation models such as queueing models, the analysts know that the output variances $var(w_i)$ are not constant at all! Fortunately, in [214] Van Beers and I demonstrate that the Kriging model in (5.7) is not very sensitive to this variance heterogeneity.

I emphasize the following property that is ignored in the Kriging literature: replacing the weights in (5.8) by the estimated optimal weights (say) $\widehat{\boldsymbol{\lambda}_0}$ implies that the Kriging predictor becomes a *nonlinear* estimator (also see the discussion on EWLS, defined in (3.22)). The literature uses the predictor variance—*given* the Kriging weights $\boldsymbol{\lambda}$; i.e., this variance is conditional on the weights. At a fixed point $\mathbf{d}$, this variance follows directly from (5.12) (also see [86], p. 122):

$$var[y(\mathbf{d})|\boldsymbol{\lambda})] = 2\sum_{i=1}^{n}\lambda_i cov(w_0, w_i) - \sum_{i=1}^{n}\sum_{i'=1}^{n}\lambda_i\lambda_{i'} cov(w_i, w_{i'}). \qquad (5.21)$$

Exercise 5.6 *Use (5.21) to derive the variance in case w_0 equals one of the points already simulated; e.g., $w_0 = w_1$.*

Ignoring the randomness of the estimated optimal weights tends to underestimate the true variance of the Kriging predictor. Moreover, the unconditional and the conditional variances do not reach their maxima at the same factor combination. To solve this problem, I distinguish between deterministic and random simulations. Because I focus on random simulations in this book, I start with that type of simulation.

- In *random* simulation, each factor combination is replicated a number of times; also see (5.20) Therefore a simple method for estimating the true predictor variance solution is *distribution-free bootstrapping.* I discussed the general principles of bootstrapping in Section 3.3.3. Van Beers and I resample—with replacement—the m_i replicated observations. This results in the n bootstrapped averages $\overline{w_i^*}$ $(i = 1, \ldots, n)$. From these $\overline{w_i^*}$, we compute the estimated optimal weights $\widehat{\boldsymbol{\lambda}_0}^*$ and the corresponding y^*. To decrease sampling effects, this whole procedure is repeated B times, which gives y_b^* with $b = 1, \ldots, B$. We estimate the variance of the Kriging predictor from these B values. Details are given in [384].

- For *deterministic* simulation, my coauthors and I apply *parametric* bootstrapping in [91]. We assume a Gaussian stationary covariance process with parameters estimated from the given simulation I/O data. Our empirical results demonstrate that ignoring the random character of the estimated Kriging weights may seriously underestimate the true predictor variance. For alternative approaches (namely, cross-validation and Akaike's Information Criterion), I refer to [248].

Note: In [91], we focus on the Gaussian correlation function. We use the DACE toolbox to estimate the parameters $\boldsymbol{\theta}$ in (5.18) from the simulation I/O data $(\mathbf{D}, \mathbf{w})$. Next, we substitute these estimated parameters $\widehat{\boldsymbol{\theta}}$ into (5.18). Then we use the Monte Carlo method to sample both old I/O data $(\mathbf{D}, \mathbf{w}^*)$ and the new data $(\mathbf{d}_0, w_0^*)$ in "a single shot" (from the multivariate normal distribution with parameters estimated from the simulation data), because the n old outputs and the new output are correlated. An alternative method is used in [246], which also finds that ignoring the uncertainty of the true parameters in the correlation function underestimates the true variance of the Kriging predictor.

Note: Kriging metamodels may also be analyzed through *functional ANOVA*; see Section 2.5, especially the references to the Sobol' ANOVA. In such an approach, the metamodel also helps understand (not only predict) the underlying simulation model: which are the important factors?

5.4 Designs for Kriging

Simulation analysts often use *LHS* to generate the I/O simulation data to which they fit a Kriging (meta)model. As I explain in Section 4.5.1, LHS was not invented for Kriging but for Risk Analysis. (Other designs related to LHS are mentioned in Section 4.5.1.)

LHS assumes that an adequate metamodel is more complicated than a low-order polynomial (which is assumed by classic designs such as fractional factorials). LHS, however, does not assume a specific metamodel or simulation model. Instead, LHS focuses on the design space formed by the k–dimensional unit cube defined by the standardized simulation inputs. LHS is one of the space filling types of design: LHS samples that space according to some prior distribution for the inputs, such as independent uniform distributions on [0, 1]; see again Section 4.5.1.

As an alternative for LHS, Van Beers and I introduce *sequentialized* designs —analyzed through Kriging—for deterministic and random simulation respectively, in [213] and [384]. We make our designs sequential for the following reasons:

- Sequential statistical procedures are known to be more “efficient”; i.e., they require fewer observations than fixed-sample (one-shot) procedures; see, e.g., [123] and [289]. Nevertheless, sequential procedures may be less efficient computationally; e.g., re-estimating the Kriging parameters may be costly; see [120].

- Computer experiments proceed sequentially (unless parallel computers are used; our procedure also fits parallel computers).

Our procedure has the following six steps, in both deterministic and random simulation (details follow after the discussion of Figure 5.2, which is displayed below).

1. We start with a *pilot* experiment, using some space-filling design with only a few factor combinations (see Section 4.5.1 on LHS and related designs). Its (say) n_0 combinations form the input into the simulation model, and gives the corresponding simulation outputs.

2. We fit a *Kriging* model to the I/O simulation data resulting from Step 1.

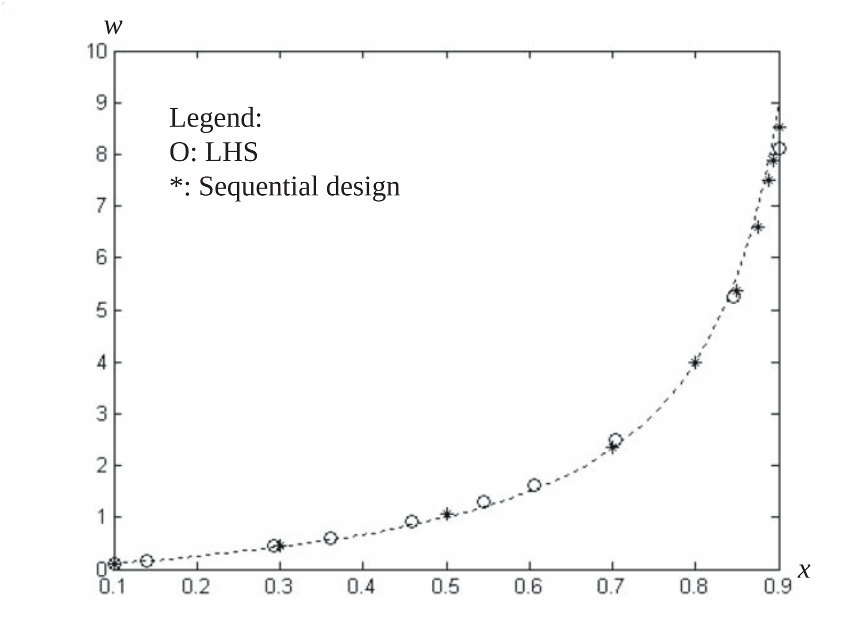

Figure 5.2: LHS and sequentialized, customized design for M/M/1 simulation

3. We consider (but do not yet simulate) a set of *candidate* combinations that have not yet been simulated and that are selected through some space-filling design; we select as the next combination to be actually simulated, the candidate combination that has the highest *predictor variance*.

4. We use the combination selected in Step 3 as the input to the simulation model, run the (expensive) simulation, and obtain the corresponding simulation output.

5. We re-fit a Kriging model to the I/O data that is augmented with the I/O data resulting from Step 4.

6. We return to Step 3 until we are satisfied with the Kriging metamodel.

Note: In Step 5, we re-fit the Kriging model, using re-estimated correlation parameters $\widehat{\theta}_j$; some researchers, however, prefer to save computer time and do not re-estimate θ_j; see [235].

Our designs are also *customized* (tailored or application-driven, not generic); i.e., which combination has the highest predictor variance is determined by the underlying simulation model. For example, if the simulation model has an I/O function (response surface) that is a simple hyperplane within a subspace of the total experimental area, then our procedure does not select points in that part of the input space; see the one-dimensional example in Figure 5.2 (reproduced from [384]). This figure displays a LHS design with $n = 10$ prefixed values for the traffic rate x in an M/M/1 simulation with experimental area $0.1 \leq x \leq 0.9$, and our sequentialized and customized design that we stop after we have simulated the same number of observations (namely 10). The figure illustrates that our design selects more input values in the part of the input range that gives a drastically increasing (highly nonlinear) I/O function. It turns out that our design gives better Kriging predictions than the fixed LHS design does—especially for small designs, which are used in expensive simulations.

I point out that "customization" requires "learning"—which is a dynamic process, so it requires sequential designs.

In the M/M/1 simulation in Figure 5.2 we use Common Random Numbers (CRN). Moreover, we take so many renewal cycles that the average output has reached a "prespecified accuracy"; i.e., the 95% confidence interval for the mean output has a relative error of no more than 15% (this is a standard procedure, which is also used in, e.g., [227]). We take a small sample size for our distribution-free bootstrap, namely $B = 50$.

We also simulate the M/M/1 without CRN, and obtain $\widehat{\theta}$ in the Gaussian correlation function: $\rho(h) = \exp(-\theta h^2)$. The simulation outputs at different traffic rates (between 0.2 and 0.8) are then less correlated, so we expect that the function in Figure 5.1 decreases faster or $\widehat{\theta}$ increases. Indeed, we find $11 < \widehat{\theta} < 16$ in five macro-replicates without CRN; with CRN we find $0.05 < \widehat{\theta} < 0.10$.

Now I present some details on Step 3 in the procedure presented above. First, I discuss random simulation models; next, I discuss deterministic simulations.

5.4.1 Predictor variance in random simulation

To estimate the variance of the Kriging predictor in random simulation, Van Beers and I use *bootstrapping*. Because a simulation model such as an M/M/1 model may have output that is not Gaussian distributed, we use *distribution-free* (non-parametric) bootstrapping; i.e., we resample the Identically and Independently Distributed (IID) outputs for a specific input combination (e.g., a specific traffic value); also see Definition 2.6. To obtain such IID observations for the simulation of the steady-state waiting time of the M/M/1 model, we use renewal analysis. For transient-state output, we would have replicated the simulation run m_i times with $i = 1, \ldots, n$ where n now denotes the number of input combinations so far simulated in the sequential design.

Note: Besides the M/M/1 simulation model, we also investigate an (s, S) inventory simulation in [384]. Our design analyzed by Kriging gives better predictions than a 4^2 full factorial design with four levels per factor (also called a 4×4 grid design) analyzed through a second-order polynomial; the latter design and analysis are also performed by Law in [227], pp. 645–655. Our design also gives better predictions than a fixed-size (one-shot) LHS design analyzed by Kriging. Our design again concentrates its combinations in the steeper part of the response surface (instead of spreading out evenly).

Note: A sequential design is also used in [404], combined with a nonlinear regression metamodel with a single explanatory variable (so $k = 1$). That article builds on previous work that I did with my co-author Cheng in [71].

5.4.2 Predictor variance in deterministic simulation

For deterministic simulation, Van Beers and I do not use bootstrapping; instead, we use cross-validation and jackknifing. We compare our design to a sequential design based on (5.21), which approximates the variance of the Kriging predictor ignoring the random character of the estimated weights. The latter design selects as the next point the input value that maximizes this variance; i.e., there is no need to specify candidate points. It turns out that this approach selects as the next point the input farthest away from the old inputs, so the final design spreads all its points evenly across the experimental area—like space filling designs do. In our approach, however, we estimate the true predictor variance through *cross-validation*. In Section 2.11.2, I discussed cross-validation for linear regression models. For Kriging, we proceed in an analogous way (for alternative cross-validation approaches, I refer to [248]). An interesting research issue is the fast computation of Kriging models in cross-validation—analogous to the shortcut (2.67) for linear regression that uses the hat matrix; also see [154] discussing fast cross-validation for Principle Component Analysis (PCA).

So, we successively delete one of the n I/O observations already simulated, which gives the data set $(\mathbf{D}_{-i}, \mathbf{w}_{-i}).(i = 1, \ldots, n)$. Next, we recompute the Kriging predictor, based on the recomputed correlation function parameters and the corresponding optimal Kriging weights; see Figure 5.3. This figure shows the three Kriging predictions for the original data set (no data deleted), and after deleting observation 2 and 3 respectively, for each of three candidate points; we do not delete the two extreme inputs (namely $x = 0$ and $x = 10$) because Kriging does not extrapolate very well. This figure shows that the point most difficult to predict is the output at the candidate point $x = 8.33$. To quantify this prediction uncertainty, we use the jackknifed variance—as follows.

In Section 3.3.3, I discussed jackknifing in general. Now, we calculate the jackknife's pseudovalue J for candidate j as the weighted average of the original and the cross-validation predictors:

$$J_{j;i} = n\widehat{y_j} - (n-1)\widehat{y_{j;-i}} \text{ with } j = 1, \ldots, c \text{ and } i = 1, \ldots, n$$

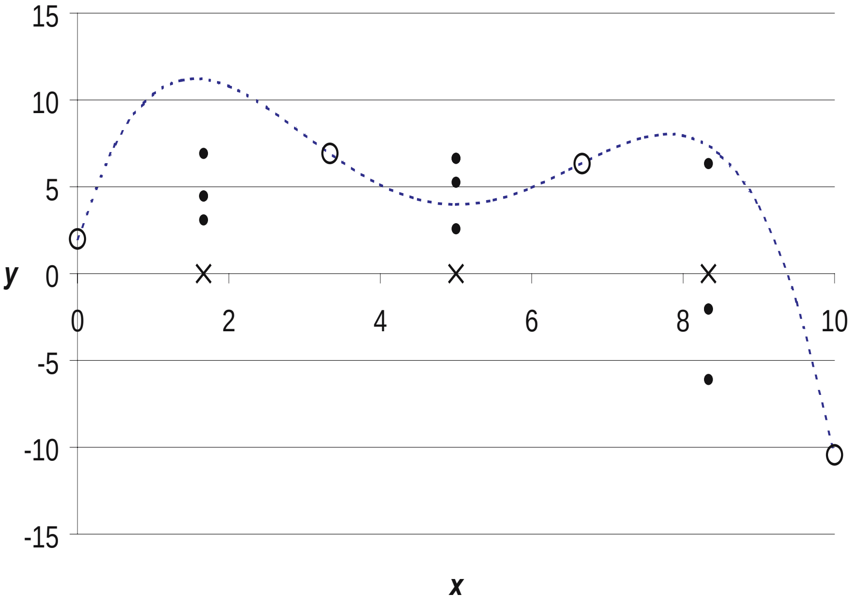

Figure 5.3: Cross-validation in fourth-order polynomial example with four pilot observations and three candidate input values

where c denotes the number of candidate points and n the number of points that have really been simulated so far and are deleted successively. From these pseudovalues we compute the classic variance estimator; see (3.12):

$$\widehat{var}(J_j) = \frac{\sum_{i=1}^{n}(J_{j;i} - \overline{J_j})^2}{n(n-1)}.$$

Like in Figure 5.2, our design favors input combinations in subareas that have more interesting I/O behavior. One artificial example is the fourth-degree polynomial I/O function with two local maxima and three local minima in Figure 5.3 (two minima occur at the border of the experimental area). Figure 5.4 shows the candidate points that are selected for actual simulation. We start with a pilot sample of four equally spaced points; also see Figure 5.3. Our design selects relative few input values in the subareas that generate an approximately linear I/O function; it selects many input values near the edges, where the function changes much.

Note: In [235], Lin et al. criticize the use of cross-validation for Kriging in deterministic simulation, but their study concerns the validation of the Kriging metamodel—not the estimation of the prediction error to select the next design point.

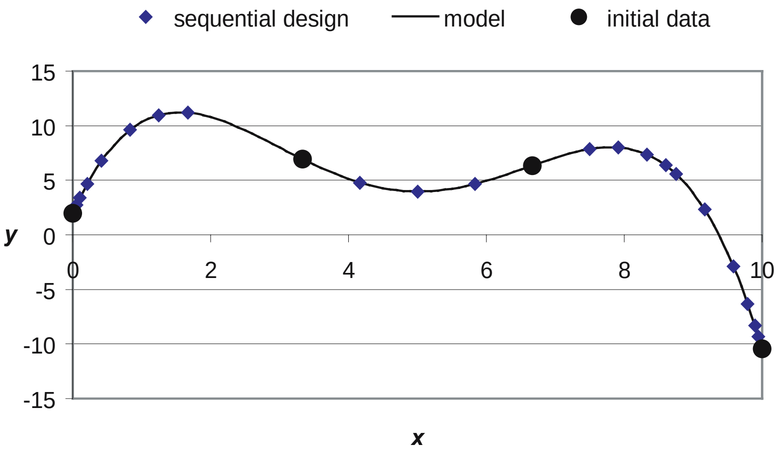

Figure 5.4: A fourth-degree polynomial example of a sequentialized and customized design

5.4.3 Related designs

I finish this section on designs for Kriging with a brief review of the literature on related sequentialized and customized designs for simulation. I review these publication, starting with the most recent publications.

- In [152], Huang et al. derive sequential designs for the optimization of random and deterministic simulation models, using Kriging and so-called Efficient Global Optimization (EGO), which maximizes the Expected Improvement (EI) following a Bayesian approach; also see [153], [168], and [336]. (In [312], Regis and Shoemaker try to balance local and global search; they use radial basis functions instead of Kriging, and instead of the EI criterion they require new points to be a prespecified distance away from old points.)

- In [355], Simpson et al. report on a panel discussion, which also emphasizes the importance of sequential and adaptive sampling.

- In [237], Lin et al. use a Bayesian approach to derive a sequential design based on prediction errors for the optimization of deterministic simulation models. That publication includes a number of interesting references. (Moreover, the related publication [236] uses a second metamodel to predict the predictor errors.)

- In [175], Keys and Rees sequentially re-estimate a spline metamodel. They use splines for optimization (instead of Sensitivity Analysis), applying the so-called “Hooke and Jeeves” search method over a grid.

- In [164], Jin et al. study sequential designs for Kriging metamodels, using (5.21), which assumes known parameters of the underlying covariance process.
- In [84], Crary discusses G-optimal and I-optimal designs, which I also discuss in Section 2.10.
- In [401], Williams et al. use a Bayesian approach to derive sequential IMSE designs, which I discuss in Section 2.10.
- In [287], Park and Faraway assume IID residuals and nonparametric regression metamodels in their sequential designs for response curve estimation.
- In [61], Chang et al. approximate deterministic nonlinear functions, focussing on splines and on grid designs (not LHS). They use "an asymptotic analysis that yields a closed-form relationship" (not small-sample cross-validation or bootstrap analysis). We find LHS designs easier to construct and more flexible than grid designs. Note that these authors also mention the extension of their approach to multivariate (instead of univariate) outputs.

In this chapter, I focus on Sensitivity Analysis; other authors, however, focus on optimization—still using Kriging; see [120], [153], [168], and [336]. They try to balance local and global search (e.g., using the EI criterion)—assuming a single simulation output (no constrained multiple outputs) and a Gaussian distribution for the stationary covariance process (instead of our distribution-free bootstrapping, jackknifing, and cross-validation). Note that in Figure 5.4 our method selects input values not only near the "top" but also near the "bottom" of the I/O function; if we were searching for a maximum, we would certainly adapt our procedure such that it would not collect data near an obvious minimum.

Note: Sequentialized and customized design procedures may benefit from *asymptotic proofs* of their performances; e.g., does the design approximate the optimal design? (Optimal designs were discussed in Section 2.10; also see [300].)

Note: In [278], Oakley estimates the 95% quantile of the output of a deterministic simulation model with uncertain inputs. Because he is interested in this quantile only, he is not interested in the whole experimental area. He combines Kriging, the Bayesian approach, and two-stage sampling.

5.5 Conclusions

This chapter may be summarized as follows. I started with a review of the basic assumption of Kriging, namely "old" simulation observations closer to the new point to be predicted, should receive more weight. This assumption is formalized through a stationary covariance process with correlations that decrease as the distances between observations increase. The Kriging model is an interpolator; i.e., predicted outputs equal observed simulated outputs at old points. Next, I reviewed some more recent results for random simulation, and I explained how the true variance of the Kriging predictor can be estimated through bootstrapping. I finished with a discussion of one-shot versus sequential designs for simulation experiments to be analyzed through Kriging.

5.6 Solutions for exercises

Solution 5.1 *Equation (5.13) with the shorthand notation* $\mathbf{c}' = \boldsymbol{\gamma}'\boldsymbol{\Gamma}^{-1}$ *implies* $E(y) = E(\widehat{\mu}) - \mathbf{c}'[E(\mathbf{w}) - E(\widehat{\mu})\mathbf{1}] = E(\widehat{\mu}) - \mathbf{c}'[E(\widehat{\mu})\mathbf{1}] - E(\widehat{\mu})\mathbf{1}] = E(\widehat{\mu}) = \mu$ *where the last equality holds because*

$$E(\widehat{\mu}) = (\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1})^{-1}\mathbf{1}'\boldsymbol{\Gamma}^{-1}E(\mathbf{w}) = (\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1})^{-1}\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mu\mathbf{1} =$$
$$(\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1})^{-1}(\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1})\mu = \mu.$$

Solution 5.2 $E(y|w_1 > \mu, w_2 = \mu, \ldots, w_n = \mu) > \mu$ *because* $\boldsymbol{\gamma}'\boldsymbol{\Gamma}^{-1} > \mathbf{0}'$.

Solution 5.3 *When* $h = 0$ *then* $\rho = 1/\exp(0) = 1/1 = 1$. *When* $h = \infty$ *then* $\rho = 1/\exp(\infty) = 1/\infty = 0$.

Solution 5.4 *When input* k *has no effect on the output, then* $\theta_k = \infty$ *in (5.18), so the correlation function drops to zero.*

Solution 5.5 *The Kriging predictor (5.13) implies*

$$\frac{\partial}{\partial d}\left(\widehat{\mu} + \gamma(\mathbf{d})'\Gamma^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1})\right) =$$
$$0 + \left(\frac{\partial}{\partial d}\gamma(\mathbf{d})'\right) \cdot \Gamma^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}) =$$
$$\left(\partial e^{-\theta(d_0 - d_1)^2}/\partial d, \ldots, \partial e^{-\theta(d_0 - d_n)^2}/\partial d\right) \cdot \Gamma^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}) =$$
$$\left(-2\theta(d_0 - d_1)e^{-\theta(d_0 - d_1)^2}, \ldots, -2\theta(d_0 - d_n)e^{-\theta(d_0 - d_n)^2}\right) \cdot \Gamma^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}).$$

Solution 5.6 *If* $w_0 = w_1$, *then* $\lambda_1 = 1$ *and* $\lambda_2 = \ldots = \lambda_n = 0$ *so* $var[y(\mathbf{d})|\boldsymbol{\lambda})] = 2cov(w_1, w_1) - [cov(w_1, w_1) + cov(w_1, w_1)] = 0$.

6
Screening designs

This chapter is organized as follows. In Section 6.1, I introduce "screening"; i.e., the search for the really important factors in experiments with simulation models that have very many factors (or inputs). In Section 6.2, I present a screening method that may be most efficient and effective, namely Sequential Bifurcation (abbreviated to SB). Subsection 6.2.1 gives an outline of the simplest type of SB. Subsection 6.2.2 covers some mathematical details of this simplest SB. Subsection 6.2.3 summarizes a case study, namely a supply-chain simulation for Ericsson in Sweden. Subsection 6.2.4 extends SB, accounting for two-factor interactions. In Section 6.3, I present conclusions.

6.1 Introduction

Why is there a need for screening? The *Pareto* principle or *20-80* rule implies that only a few factors (simulation inputs) are really important (or "active", as some authors say). The *parsimony* principle or *Occam's razor* implies that a simpler explanation is preferred to a more complex explanation—all other things being equal. In the DASE context, *screening* means that the simulation analysts are searching for the really important factors among the many factors (often hundreds or more) that can be varied in the simulation experiment. In other words, effects are assume to be "sparse". The *curse of dimensionality* is also mentioned in many publications, including the 2002 panel report [355]. In his famous article [259], the psychologist Miller claims that people cannot handle more than "seven plus or minus two" factors when processing information.

For deterministic simulation and random simulation respectively, I give the following examples of simulation with many factors in which I was personally involved.

- Deterministic simulation: The Dutch organization RIVM developed a simulation model (called "IMAGE"), which tries to explain the worldwide increase of temperatures known as the greenhouse phenomenon. In [41], Bettonvil and I vary 281 factors in a submodel of this simulation model. After simulating only 154 factor combinations (scenarios), we identify a shortlist with 15 factors, including some factors that the ecological experts had not expected to have important effects! This shortlist was used to support national policy makers in their decision-making. It is also important to know which factors are "certainly" unimportant so the decision-makers are not bothered by details about these factors.

- Random simulation: Originally, Persson and Olhager developed a supply chain simulation for the Ericsson company in Sweden, and simulated only nine combinations of factor values; see [292]. In [195], my coauthors and I revisit this simulation model, and find that this model actually has 92 factors. Even if we wished to experiment with the minimum number of values per factor (namely 2), we could not simulate all combinations; e.g., $2^{92} \approx 5 \times 10^{27}$, which is close to infinity. And changing one factor at a time still requires 93 simulation runs if not more than two values per factor are simulated; moreover, this approach does not enable the estimation of any factor interactions. In Subsection 6.2.3, I shall show how we actually simulate only 21 combinations—each combination replicated five times—to identify a shortlist with the 11 most important factors among the original 92 factors. Note that one replicate takes 40 minutes in this case study (after modification of the simulation code, which originally took three hours per replicate). I also discuss this case study in Section 4.6.1, focussing on Risk Analysis and Robust Optimization.

The importance of factors depends on the *experimental domain* (also called the experimental area or experimental frame; see Section 2.3). The users should supply information on this domain, including realistic ranges of the individual factors and limits on the admissible factor combinations (e.g., some factor values must add up to 100%). Therefore, in practice, user involvement is crucial for the application of screening methods.

There are several types of screening designs. All these designs treat the simulation as a *black box* (see Definition 2.1).

- *Classic* two-level designs and Frequency Domain Experimentation (FDE)—discussed in Chapter 2—are often considered to provide screening designs. Especially resolution-III designs are often called screening designs in the literature; see, e.g., [122] and [411]. So-called *conference designs* have twice as many combinations as there are factors (so $n = 2k$); see [109].

- *Supersaturated* designs have fewer combinations than factors (so $n < k$). (Classic designs are called "saturated" if $n = q$ where q denotes the number of regression parameters; e.g., $q = 1 + k$ in a first-order polynomial metamodel.) These supersaturated designs assume that the designs are not sequential, so they are relatively inefficient as I explained in Section 5.4. I discuss these designs in [181] (my 1974/1975 book). For recent discussions of supersaturated designs, I refer to [6], [62], [128], [231], [402], [403], and [415].

  Note: [382] also gives a screening procedure with $n < k$ but uses Moving Least Squares (MLS) and cross-validation; also see Section 3.4.4.

- *Group-screening* designs aggregate (or confound) individual factors into groups so that k factors may be evaluated in less than k combinations. Consequently, these designs are supersaturated—but they are executed in two or more steps (stages). There are several types of these screening designs. Examples are One-factor-At-a-Time (OAT), Morris's OAT, Cotter's design, Andres's Iterated Fractional Factorial Design (IFFD), multi-stage group screening, and Sequential Bifurcation (SB); see [57], [58], [95], [181], [262], [327], [329], and [341].

Different designs are based on different mathematical assumptions concerning the *smoothness* of the I/O function implied by the underlying simulation model, possible *monotonicity* of this function, etc. I focus on SB because it is a very efficient and effective method if its assumptions are satisfied; see Section 6.2 below. (SB resembles binary search, which is a well-known procedure in computer science; SB, however, not only estimates *which* factors are important, but also estimates the *magnitudes* of the effects of the important factors.)

The *fixed* sample-size assumption of classic and supersaturated designs does not hold if the next factor combination is selected after the preceding Input/Output (I/O) simulation data are analyzed. This analysis may give designs that are purely sequential, multi-stage, or two-stage (also called double sampling). Moreover, these designs are "customized"; i.e., they account for the specific simulation model. I also refer to Section 5.4.

In practice, simulation models have *multivariate* output. This problem has not yet been touched in the screening literature. As a simple rule, I propose to declare a (sub)group of factors to be important if at least one of the multiple outputs changes significantly when changing the level of this group.

Note: IFFD is used, e.g., in [11] and identifies 8 important factors among the 3800 individual factors in 512 simulation runs with a Risk Analysis model of nuclear waste disposal in Canada. This method assumes a second-order polynomial metamodel. Also see [9].

Note: A Bayesian analysis of screening experiments is presented in [76].

6.2 Sequential Bifurcation

Originally, Bettonvil developed SB in his doctoral dissertation, [38]. He and I summarized his dissertation in [41] and [195]. A few other authors extended SB; see [67], [70], [349], [393], and [394]. The specific SB extensions made in these publications will be mentioned below.

SB uses the following assumptions, which will be spelled out below:

Assumption 1(a): first-order polynomial metamodel

Assumption 1(b): first-order polynomial augmented with two-factor interactions, which replaces Assumption 1(a)

Assumption 2: known signs of the first-order effects

Assumption 3: strong heredity if Assumption 1(b) holds

6.2.1 Outline of simplest SB

The SB procedure is *sequential;* i.e., it consists of a sequence of steps. The first step aggregates all factors into a single group, and tests whether or not that group of factors has an important effect (this test will be detailed in Subsection 6.2.2). If the group has indeed an important effect, then the second step splits the group into two subgroups—*bifurcates*—and tests each of these subgroups for importance. The next steps continue in a similar way; i.e., SB splits important subgroups into smaller subgroups, and discards unimportant subgroups. In the final steps, all individual factors that are not in subgroups identified as unimportant, are estimated and tested.

The simplest type of SB uses the following two assumptions.

Assumption 1(a): a valid metamodel is a first-order polynomial plus noise:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + e. \tag{6.1}$$

Note: I repeat some basic concepts that I have already covered in Chapter 2. The input variables x_j $(j = 1, \ldots, k)$ are standardized such that they are either -1 or $+1$; see (2.32). This scaling implies that the factors may be ranked (sorted) by their main effects; i.e., the most important factor is the one with the largest absolute value of its first-order effect or main effect; the least important factor is the one with the effect closest to zero. The larger the range of an untransformed (original) factor is, the larger the response difference and hence the main effect of that factor is (also see the "unit cost" effects in [70]). The noise e in (6.1) arises from approximation error and—in case of random simulation—the use of Pseudo-Random Numbers (PRNs); also see the comment below (2.7). If the metamodel is valid, then this noise has zero expected value: $E(e) = 0$.

To estimate the parameters in this simple metamodel, it is most efficient to experiment with only two levels (values) per factor (see again Chapter 2). In practice, it is important that these levels are realistic extreme values; so the users of the underlying simulation model should provide these values. Also see the discussion of scaling in [393].

Assumption 2: the signs of all main effects are known and are nonnegative:

$$\beta_j \geq 0 \ (j = 1, \ldots, k).$$

Without Assumption 2, main effects might cancel each other. However, if Assumption 2 holds, then the analysts can define the two levels of an individual factor such that changing the level from the standardized value -1 to $+1$ does not decrease the expected simulation output (i.e., that change either increases the output or does not change it at all). For example, if the arrival rate is increased, then the expected steady-state waiting time increases. If the queuing discipline changes from FIFO (First-In-First-Out) to SPT (Shortest-ProcessingTime-first), then the expected waiting time decreases, so the level -1 should correspond with SPT and $+1$ with FIFO.

Figure 6.1 illustrates that the "known signs" assumption is related to the "monotonicity" of the I/O function, defined as follows.

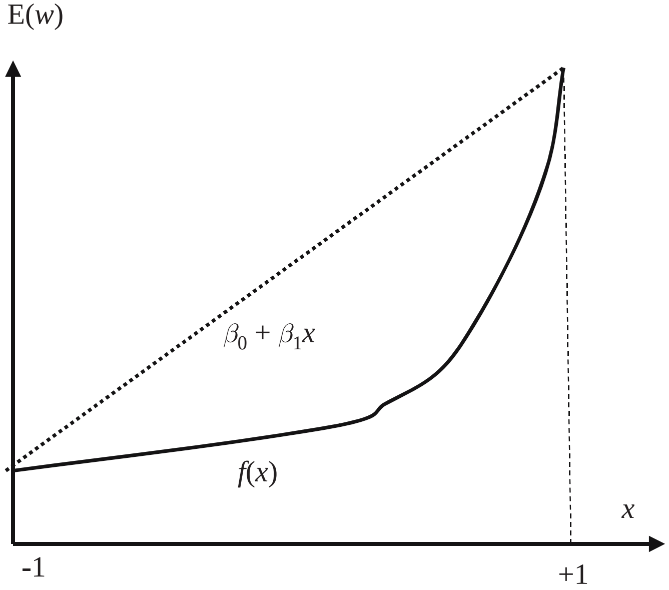

Figure 6.1: Known signs and monotonicity

Definition 6.1 *The function $w = f(x_1, \ldots, x_k)$ is called monotonically increasing if $\partial w/\partial x_j \geq 0$ for all j and all values of $x_{j'}$ ($j, j' = 1, \ldots, k$; $j \neq j'$).*

Obviously, the factors can be defined such that if the function is monotonically decreasing in the original factors, this function becomes monotonically increasing in the standardized factors.

My experience is that Assumption 2 is easy to satisfy in practice; i.e., it is straightforward to define the upper and lower levels of each factor such that changing a factor from its lower to its upper level does not decrease the expected response. For example, in the Ericsson supply-chain case-study some factors refer to transportation speeds: the higher these speeds, the lower the Work In Process (WIP) and hence the lower the cost—which is the output of interest in the screening experiment. Other authors give more examples; see [15], [230], [234], and [352] ([15] proposes wavelet based estimators; [234] proposes "isotonic regression", allowing Common Random Numbers, CRN; both publications assume $k = 1$ factor).

Note: In unconstrained optimization, the function to be maximized or minimized is assumed not to be monotonically increasing; otherwise, the maximum or minimum lies at the limits of the experimental area. This assumption may still be compatible with the "known signs" assumption, as Figure 6.2 illustrates. In this figure, switching the standardized factor value from -1 to $+1$ increases the output (so this factor will be found

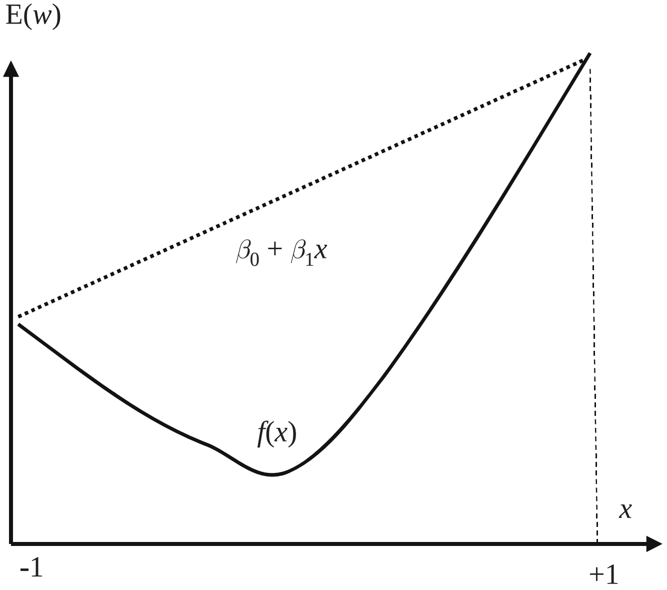

Figure 6.2: Known signs and non-monotonicity

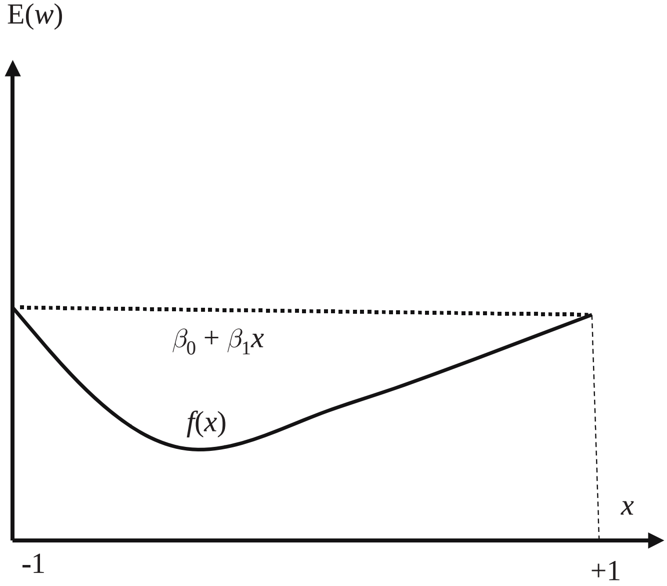

Figure 6.3: Non-monotonic I/O function with misleading sign

to have an important effect). Figure 6.3, however, gives a "pathological" counterexample; i.e., the I/O function is not monotonic, and happens to give the same output values at the two observed input levels −1 and +1 so the factor effect seems to be zero and this factor will be eliminated by SB.

Nevertheless, if in a particular case study it seems hard to satisfy Assumption 2 for a few specific factors, then these factors should be treated *individually*; i.e., none of these factors should be grouped with other factors in SB. For example, [95] creates some subgroups of size one in a multi-stage group-screening design; this design is less efficient than SB, but it also uses aggregation. Treating such factors individually is safer than assuming negligible probability of cancellation within a subgroup.

The *efficiency* of SB—measured by the number of simulated factor combinations (and hence simulation time)—improves if the individual factors are labeled such that factors are placed in increasing order of importance; see [38], p. 44 (consequently, the important factors are clustered). To realize this efficiency gain, it is crucial to utilize prior knowledge of users and analysts about the real system being simulated. For example, if they conjecture that environmental factors are most important, then these factors should be placed at the end of the list of factors. Indeed, in the Ericsson case study we place the environmental factor "demand" at the very end of the list with 92 individual factors.

The efficiency further improves when placing similar factors within the same subgroup. In the Ericsson case study, we group all "test yield" factors together; our conjecture is that if one yield factor is unimportant, then all yield factors are likely to be unimportant too.

Finally, the efficiency increases if factor subgroups are split such that the number of factors for the first new subgroup is a power of two; e.g., we split the first 48 factors into a subgroup of 32 (= 2^5) factors and a subgroup of the remaining 16 factors (so the important factors are placed into the smallest subgroup, assuming the factors are sorted from unimportant to most important). However, I do not recommend this splitting if it implies splitting up a group of related factors. Anyhow, splitting a subgroup into subgroups of *equal* size (like some authors do) does not need to be optimal. Also see [38], pp. 40–43.

The way SB proceeds may be interpreted through the following *metaphor*; also see Figure 6.4. Imagine a lake that is controlled by a dam. The goal of the experiment is to identify the highest (most important) rocks (actually, SB not only identifies, but also measures the height of these "rocks"). The dam is controlled in such a way that the level of the murky water slowly drops. Obviously, the highest rock first emerges from the water! The most-important-but-one rock turns up next; etc.. SB stops when the simulation analysts feel that all the "important" factors are identified. Once SB stops, the analysts know that all remaining (unidentified) factors have smaller effects than the effects of the factors that have been identified. (I will further discuss this figure below.)

The aggregated effect of a given subgroup is an upper limit (say) U for the value of any individual main effect within that subgroup. If the analysts

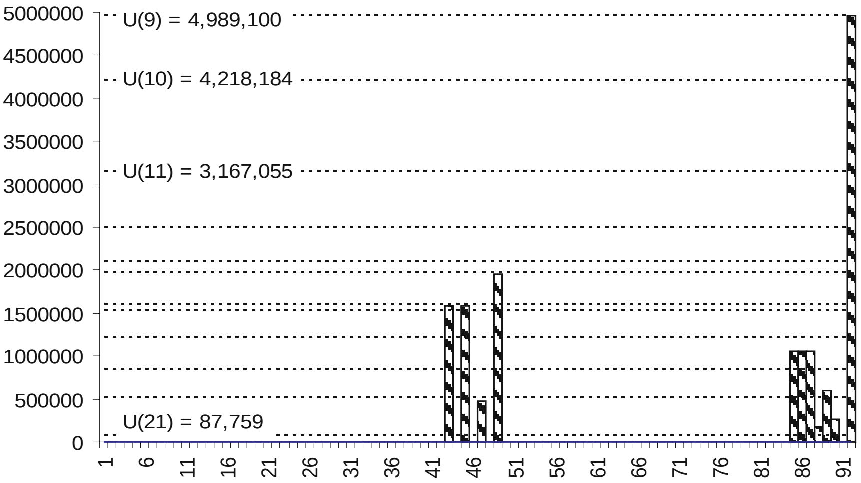

Figure 6.4: Upper limit $U(i)$ after step i ($i = 9, \dots, 21$) and individual main effect estimates (shaded bars) versus the factor label j ($j = 1, \dots, 92$) in the Ericsson supply-chain simulation

must terminate SB prematurely (e.g., because their computer breaks down or their clients get impatient), then SB still allows identification of the factors with the largest main effects. For example, if in Figure 6.4, SB is terminated after Step 11, then the most important factor has already been identified and its main effect has been estimated (none of the other factors has a main effect exceeding that of the factor labeled 92).

SB is extended in [394], improving the control over the type-I error rates ("false positives"), using either a two-stage approach or a fully sequential approach. Theoretically, this control does not satisfy the classic statistical requirements concerning a prespecified experimentwise error rate and a prespecified power for the *final* results—after *all* stages have been executed. Nevertheless, the numerical results in that publication look very promising.

SB is extended to the so-called *polytope* method in [14]. The latter method is more efficient (requiring fewer combinations to be simulated), but is also more complicated (requiring the solution of a Linear Programming or LP problem after each additional observation). Moreover this method assumes main effects only (interactions will be discussed in Subsection 6.2.4). Note that the LP problem arises because this method computes the Ordinary Least Squares (OLS) estimate (i.e., it minimizes the Sum of Squared Residuals, SSR, defined in (2.11)) under the constraint stipulating that all regression coefficients be nonnegative (see Assumption 2 above).

6.2.2 Mathematical details of simplest SB

To explain some mathematical details of SB, I use the following additional notation.

$w_{(j);r}$: observed simulation output with the factors 1 through j set to their high levels and the remaining factors set to their low levels, in replication r;

$\beta_{j'-j}$: sum of main effects of factors j' through j; that is

$$\beta_{j'-j} = \sum_{h=j'}^{j} \beta_h. \tag{6.2}$$

A simple estimate (a complicated estimate is given in [14]) of this group effect based on replication r is

$$\widehat{\beta_{j'-j};r} = \frac{w_{(j);r} - w_{(j'-1);r}}{2}. \tag{6.3}$$

SB starts with simulating the two most extreme scenarios: in scenario 1 all k factors are set at their low levels so $x_j = -1$; in scenario 2 all these factors are high so $x_j = 1$ ($j = 1, \ldots, k$). If the metamodel in (6.1) is valid, then

$$E(w_{(0)}) = \beta_0 - \beta_1 - \ldots - \beta_k \tag{6.4a}$$

and

$$E(w_{(k)}) = \beta_0 + \beta_1 + \ldots + \beta_k \tag{6.5a}$$

so

$$E(w_{(k)}) - E(w_{(0)}) = 2(\beta_1 + \ldots + \beta_k), \tag{6.6}$$

which shows that the group effect estimator defined in (6.3) is *unbiased.*

Likewise it follows that the individual main effect is estimated through the analogue of (6.3):

$$\widehat{\beta_{j;r}} = \frac{w_{(j)r} - w_{(j-1);r}}{2}. \tag{6.7}$$

Analogous to (3.32), the (say) m replicates enable the estimation of the mean and the variance for each (aggregated or individual) estimated effect. For example, (6.7) gives

$$\overline{\widehat{\beta_j}} = \frac{\sum_{r=1}^{m} \widehat{\beta_{j;r}}}{m} \text{ and } s(\overline{\widehat{\beta_j}}) = \sqrt{\frac{\sum_{r=1}^{m} (\widehat{\beta_{j;r}} - \overline{\widehat{\beta_j}})^2}{m(m-1)}}. \tag{6.8}$$

This variance estimator allows unequal response variances and CRN.

To *test* the importance of the estimated (either aggregated or individual) main effects statistically, SB uses a t statistic; see (2.19). Different scenarios probably produce observations with different variances, and may use CRN; see (6.8). SB applies a one-sided test because all individual main effects are assumed to be nonnegative. SB uses a prespecified type-I error rate per test (e.g., $\alpha = 0.05$); i.e., SB does not adjust for multiple testing (RSM is also a sequential procedure that does not control the type-I and type-II error rates over the whole procedure; see Section 4.2). However, [393] does use multiple testing procedures in its SB.

To *verify* (or validate) the shortlist produced by SB, I recommend to test the effects of the "unimportant" factors through the following two scenarios, each simulated m times:

1. Set all factors that SB declared to be unimportant at their low levels, while keeping the important factors fixed (for example, at their base levels).

2. Switch all these unimportant factors to their high levels, still keeping the important factors fixed.

Obviously, these two scenarios are not used in SB if verification fixes the important factors at base values (coded as 0) that are not extreme values (coded as either -1 or 1). The difference between the outputs of these two scenarios may be tested through a t 'statistic; this difference should not differ significantly from zero.

How SB proceeds sequentially is illustrated in the following case study. A formal procedure for the SB steps is given in [394].

6.2.3 Case study: Ericsson's supply chain

For the Ericsson simulation model my coauthors and I distinguish $k = 92$ factors and obtain $m = 5$ replicates. Table 6.1 gives the replicates for the two extreme scenarios. This table shows that the scenario with all factors at their low levels has an average output $\overline{w_{(0)}} =$ 3,981,627. The other scenario has all factors at their high levels; its average output is $\overline{w_{(92)}} =$ 34,013,832. So, the estimated group effect of all 92 factors is obtained from (6.2), (6.6), and (6.8), and is $\widehat{\beta_{1-92}} =$ (34,013,832 - 3,983,627)/2 = 15,016,102. The standard error of this estimated group effect follows from this table and (6.8): $s(\widehat{\beta_{1-92}}) = 94{,}029.3/\sqrt{5} = 42{,}051$. So this effect is very significant!

Note: On hindsight, this early stage might have used fewer replicates; e.g., only $m = 2$ replicates would have shown that one or more factors among the 92 factors must be important; also see the next exercise.

Exercise 6.1 *Derive the value of the* t *statistic from the first two replicates only, in Table 6.1.*

Note: If this simulation were deterministic without numerical noise, then $m = 1$ replicate would have sufficed. The ratio $w_{(92)}/w_{(0)}$ would have clearly shown that one or more factors must be important.

Figure 6.5 shows the successive SB steps for this case study (a figure with a different layout for a related Ericsson model is given in [195]). For example, this figure shows that the next step after the initial step with its two extreme scenarios, divides the current group of 92 factors into two subgroups. Into the first subgroup (in the left-hand side of the figure) we decide to place all the 79 "decision" factors; into the other subgroup we put all 13 "environmental" factors (controllable and environmental factors are discussed in Section 4.6). Simulation of this scenario gives an expected output between the expected outputs of the preceding extreme scenarios (values are not displayed). Comparison of $\overline{w_{(79)}}$ and $\overline{w_{(0)}}$ gives $\widehat{\beta_{1-79}}$. Similarly, comparison of $\overline{w_{(92)}}$ and $\overline{w_{(79)}}$ gives $\widehat{\beta_{80-92}}$. So, this step splits the

Replicate	$\overline{w_{(0)}}$	$\overline{w_{(92)}}$	$\widehat{\beta_{1-92}}$
1	3,954,024	34,206,800	15,126,388.0
2	3,975,052	33,874,390	14,949,669.0
3.	3,991,679	33,775,326	14,891,823.5
4	4,003,475	34,101,251	15,048,888.0
5	3,983,905	34,111,392	15,063,743.5
Average	3,981,627	34,013,832	15,016,102.4
Standard Error	18,633	180,780	94,029.3

Table 6.1: First two combinations replicated five times in Ericsson's supply chain

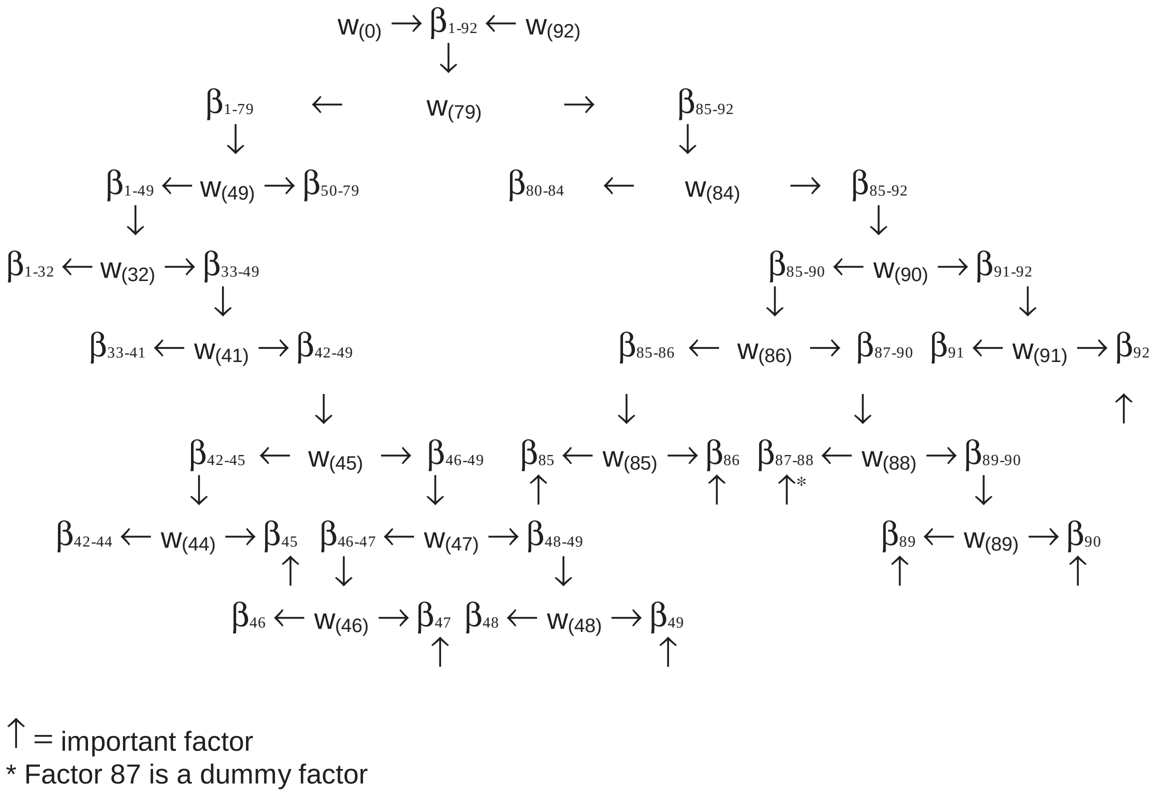

Figure 6.5: SB steps in Ericsson case study

total effect $\widehat{\beta_{1-92}}$ into its two additive components. This step decreases the upper limit U for any individual effect in the first subgroup and the second subgroup respectively; see again Figure 6.4.

SB does not split a subgroup any further when its estimated aggregated main effect is nonsignificantly positive (if the estimate were significantly negative in a two-sided t test, then Assumption 2 would be rejected); e.g., the estimated aggregated main effect of factors 50 through 79 turns out to be a small negative value.

In this case study, SB stops after 21 steps. The upper limit, $U(21)$, for the main effect of any remaining individual factor is then reduced to 87,759; see again Figure 6.4. Our shortlist has 11 factors; the most important factor is factor 92. To improve the SB efficiency, we try to label the factors from least important to most important; we now conclude that factor 92 is indeed the most important factor and that no factor labelled smaller than 43 is declared to be important. This figure also shows that the most important individual factor (namely, factor 92) has already been identified and estimated after only ten steps; the next important factor (namely, factor 49) is identified after 16 observations.

6.2.4 SB with two-factor interactions

In this section, I summarize SB for situations in which Assumption 1(a) is replaced by Assumptions 1(b) and 3.

Assumption 1(b): a valid metamodel is a first-order polynomial augmented with two-factor interactions and noise:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \beta_{1;2} x_1 x_2 + \ldots + \beta_{k-1;k} x_{k-1} x_k + e. \quad (6.9)$$

Note: The signs of the two-factor interactions are irrelevant as we shall see.

Assumption 3: if a factor has no important main effect, then this factor does not interact with any other factor.

Assumption 3 is called the *strong heredity* assumption; see [402] and also [327]. Strong heredity is related to *functional marginality*, which was recently discussed in [381].

SB enables the estimation of first-order effects unbiased by two-factor interactions if the *foldover* principle is applied (in Section 2.6, I applied Theorem 2.1 to resolution-III designs to obtain resolution-IV designs). This principle means that SB simulates the “mirror” combination besides the original factor combination (“mirror” observations will be defined in the next paragraph). Hence, the number of simulated combinations doubles. Furthermore, the mirror combinations imply that more combinations are simulated; however, it may happen that fewer replications per scenario are needed in random simulation; see [393]. To further improve the efficiency, CRN may be applied separately to all positive levels and negative levels respectively; see [394].

More specifically, let $w_{-(j)}$ be the *mirror* observation of $w_{(j)})$; i.e., $w_{-(j)}$ is the simulation output with the factors 1 through j set to their low levels and the remaining factors set to their high levels. For example, for $j = 48$ the analogue of (6.4a) and (6.5a) is

$$\begin{aligned} E(w_{-(49)}) &= \beta_0 + (-\beta_1 - \ldots - \beta_{49}) + (\beta_{50} + \ldots + \beta_{92}) + \\ &\quad +(\beta_{1;2} + \ldots + \beta_{48;49}) + \\ &\quad +(-\beta_{1;50} - \ldots - \beta_{49;92}) + \\ &\quad +(\beta_{50;51} + \ldots + \beta_{91;92}) \end{aligned}$$

and

$$\begin{aligned} E(w_{(49)}) &= \beta_0 + (\beta_1 + \ldots + \beta_{49}) + (-\beta_{50} - \ldots - \beta_{92}) + \\ &\quad +(\beta_{1;2} + \ldots + \beta_{48;49}) + \\ &\quad +(-\beta_{1;50} - \ldots - \beta_{49;92}) + \\ &\quad +(\beta_{50;51} + \ldots + \beta_{91;92}) \end{aligned}$$

so subtracting these two equations cancels all interactions!

The analogue of (6.3) gives the unbiased group estimator

$$\widehat{\beta_{j'-j};}r = \frac{(w_{(j);r} - w_{-(j);r}) - (w_{(j'-1);r} - w_{-(j'-1);r})}{4}. \tag{6.10}$$

The analogue of (6.7) gives the unbiased individual estimator

$$\widehat{\beta_{j};}r = \frac{(w_{(j);r} - w_{-(j);r}) - (w_{(j-1);r} - w_{-(j-1);r})}{4}. \tag{6.11}$$

Exercise 6.2 *What is the mirror scenario of the extreme scenario with all factors at their low levels?*

SB augmented with mirror scenarios may still give misleading results if (say) two factors have unimportant main effects but their interaction is important. Therefore SB assumes *strong heredity.* If the analysts suspect that this assumption is violated for a specific factor, then they should investigate that factor after the screening phase.

SB with mirror scenarios does not enable estimation of *individual* interactions, but it does show whether interactions are important—as follows. Estimate the main effects from the original scenarios ignoring the mirror scenarios. If the analyses of the mirror scenarios and of the original scenarios give the same conclusions, then interactions are unimportant. This happens, e.g., in the ecological simulation reported in [38] and [41]. In that study, the factor values change relatively little (larger changes give unrealistic simulation output), so a first-order polynomial is adequate. In the

Ericsson case study, however, interactions turn out to be important. (In a follow-up experiment with the factors declared to be important in SB, the sizes of the individual interactions are estimated from a resolution-V design; see [194].) The foldover design may give a different path through the list of individual factors; e.g., the path in Figure 6.5 may change.

More details on SB using mirror scenarios—applied to the Ericsson model —are given in [195]; also see Section 4.6.1. These details include programming and validation of the simulation model, the role of two-factor interactions and "dummy" factors (individual factors in SB that do not occur in the simulation model itself, so they are known to have zero effects; also see the next exercise), steady-state analysis (including estimation of a warm-up period).

Exercise 6.3 *The Ericsson study concerns three variants of the supply chain, such that the oldest variant has more factors (namely 92) than the current variant (which has 78 factors). Hence, applying SB to the current variant uses 14 dummy factors. Will the group effect after simulating the two extreme scenarios for the current variant be smaller or larger than for the old variant?*

Note: In [195], we also discuss the need for *software* that implements sequential screening of simulation experiments. That software should generate an input file, once a particular design type (e.g., SB) has been chosen. Such a file can then be executed sequentially (and efficiently) in batch mode; i.e., no human intervention is required while the computer executes the sequential design (including rules for selecting the next input combination, based on all preceding observations). Good computer programming avoids fixing the inputs at specific numerical values within the code; instead, the computer reads input values so that the program can be run for many combinations of these values. (Of course, the computer should check whether these values are admissible; i.e., are these combinations within the experimental domain?) Such a practice can automatically provide a long list of potential factors.

6.3 Conclusions

This chapter may be summarized as follows. I started with an overview of different screening designs, including resolution-III, supersaturated, and group-screening designs. Then I focused on SB. I detailed the various assumptions of SB. These assumptions may not be too restrictive in practice, as the Ericsson case study illustrated. If the SB assumptions are satisfied, then this screening method is a most efficient and effective method that may be applied to deterministic and random simulations!

6.4 Solutions for exercises

1. $\overline{\widehat{\beta_{1-92}}} = (15,126,388.0 + 14,949,669.0)/2 = 15,038,000$ and
$s(\overline{\widehat{\beta_{1-92}}}) = 694,610/\sqrt{2} = 491,160$ so
$t = 15,038,000/491,160 = 30.62$.

2. The mirror scenario of the extreme scenario with all factors at their low levels, is the other extreme scenario with all factors at their high levels.

3. The group effect of the two extreme scenarios for the current variant of the supply chain is smaller than for the old variant.

7
Epilogue

I summarize this book as follows.

In Chapter 1—called Introduction—I first discussed various types of simulation. Next, I described the DASE approach. Then I defined symbols and terms for DASE.

In Chapter 2, I gave a detailed tutorial explaining the basics of linear regression models—especially first-order and second-order polynomial models—and the corresponding statistical designs—namely, designs of resolution III, IV, and V and Central Composite Designs (CCDs). I also discussed the validation of the estimated regression model, including the coefficient of determination R^2 and the adjusted coefficient $R^2_{adjusted}$, Pearson's and Spearman's correlation coefficients, and cross-validation. Throughout that chapter, I assumed white noise, meaning that the residuals of the fitted linear regression model are Normally, Independently, and Identically Distributed (NIID) with zero mean.

In Chapter 3, I dropped the white-noise assumption, and explained the consequences; i.e., I discussed regression analysis and experimental designs for simulation practice. I pointed out that multivariate simulation output can still be analyzed through OLS. I addressed possible nonnormality of simulation output, including normality tests, transformations of simulation Input/Output (I/O) data, jackknifing, and bootstrapping. I presented analysis and design methods for heteroscedastic simulation output. I discussed how to analyze simulation I/O data that uses Common Random Numbers (CRN), so the simulation outputs are correlated across different factor combinations. I discussed possible lack-of-fit tests for low-order polynomial metamodels, transformations to improve the metamodel's validity, and alternative metamodels and designs.

In Chapter 4, I first summarized classic Response Surface Methodology (RSM), assuming a single response variable. I added the Adapted Steepest Ascent (ASA) search direction, which improves the classic steepest ascent direction. Next, I summarized Generalized RSM (GRSM) for simulation with multivariate responses, assuming that one response is to be minimized while all the other responses must meet given constraints. Moreover, the (deterministic) inputs must satisfy given box constraints. Then, I summarized a procedure for testing whether an estimated optimum is truly optimal—using the Karush-Kuhn-Tucker (KKT) conditions. This procedure combines classic tests and bootstrapped tests. Next, I discussed Risk Analysis (RA) or Uncertainty Analysis (UA). Finally, I discussed Robust Optimization, focusing on a Taguchian approach.

In Chapter 5, I started with a review of the basic assumption of Kriging, namely "old" simulation observations closer to the new point to be predicted, should receive more weight. This assumption is formalized through a stationary covariance process with correlations that decrease as the distances between observations increase. The Kriging model is an interpolator; i.e., predicted outputs equal observed simulated outputs at old points. Next, I reviewed some more recent results for random simulation, and I explained how the true variance of the Kriging predictor can be estimated through bootstrapping. I finished with a discussion of one-shot versus sequential designs for simulation experiments to be analyzed through Kriging.

In Chapter 6, I started with an overview of different screening designs, including resolution-III, supersaturated, and group-screening designs. Then I focused on SB. I detailed the various assumptions of SB. These assumptions may not be too restrictive in practice, as the Ericsson case study illustrated. If its assumptions are satisfied, then SB is a most efficient and effective screening method that may be applied to deterministic and random simulations.

References

[1] Adenso-Diaz, B. and M. Laguna (2006), Fine-tuning of algorithms using fractional experimental designs and local search. *Operations Research*, 54, no. 1, pp. 99–114 [103, 104, 130]

[2] Aggarwal, M.L. and M.D. Mazumder (2005), Optimal fractional factorial plans using minihypers. *Statistics & Probability Letters*, 75, no. 4, pp. 291–297 [52]

[3] Al-Aomar, R. (2002), A robust simulation-based multicriteria optimization methodology. *Proceedings of the 2002 Winter Simulation Conference*, edited by E. Yücesan, C.H. Chen, J.L. Snowdon and J.M. Charnes, pp. 1931–1939 [131]

[4] Al-Aomar, R. (2006), Incorporating robustness into genetic algorithm search of stochastic simulation outputs. *Simulation Modelling Practice and Theory*, 14, pp. 201–223 [132]

[5] Alexandrov N.M. and M.Y. Hussaini (1997), Multidisciplinary design optimization—state of the art. *Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, SIAM Proceedings Series [3, 123]

[6] Allen, T.T. and M. Bernshteyn (2003), Supersaturated designs that maximize the probability of finding the active factors. *Technometrics*, 45, no. 1, pp. 1–8 [159]

[7] Alrefaei M.H. and M. Almomani (2007), Subset selection of best simulated systems. *Journal of the Franklin Institute*, in press [102]

[8] Andradóttir, S. (2006), An overview of simulation optimization via random search. *Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, Elsevier/North Holland, pp. 617–631 [103]

[9] Andres, T.H. (1997), Sampling methods and sensitivity analysis for large parameter sets. *Journal of Statistical Computation and Simulation*, 57, no. 1/4, pp. 77–110 [160]

[10] Andres, T. (2002), Uncertainty analysis guide. Report AECL-12103, Whiteshell Laboratories, Pinawa, Manitoba R0E 1L0, Canada http://www.aecl.ca/Canada/aecl%20library%20site%20Canada/key%20files/serrepaecl.htm [124]

[11] Andres, T.H. and W.C. Hajas (1993), Using iterated fractional factorial design to screen parameters in sensitivity analysis of a probabilistic risk assessment model, *Proceedings of the Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applications*, Karlsruhe, Germany, April 19 to 23, volume 2, pp. 328–37 [160]

[12] Angün, E., D. den Hertog, G. Gürkan, and J.P.C. Kleijnen (2006), Response surface methodology with stochastic constraints for expensive simulation. Working Paper, Tilburg University, Tilburg, Netherlands [75, 101, 110, 113, 116, 138]

[13] Angün, E. and J.P.C. Kleijnen (2006), An asymptotic test of optimality conditions in multiresponse simulation-based optimization. Working Paper, Tilburg University, Tilburg, Netherlands [101, 118]

[14] Ankenman, B.E., R.C.H. Cheng, and S.M. Lewis (2006), A polytope method for estimating factor main effects efficiently. *Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 369–375 [165]

[15] Antoniadis, A., J. Bigot, and I. Gijbels (2007), Penalized wavelet monotone regression. *Statistics & Probability Letters*, in press [162]

[16] Antionidis, A. and D.T. Pham (1998), Wavelet regression for random or irregular design. *Computational Statistics and Data Analysis,* 28, pp. 353–369 [8]

[17] Arcones, M.A. and Y. Wang (2006), Some new tests for normality based on U-processes. *Statistics & Probability Letters*, 76, no. 1, pp. 69–82 [79]

[18] Arnold, S.F. (1981), *The theory of linear models and multivariate analysis.* Wiley, New York [91]

[19] Atkinson, A. and M. Riani (2000), *Robust diagnostic regression analysis.* Springer, New York [8, 61, 81]

[20] Ayanso, A., M. Diaby, and S.K. Nair (2006), Inventory rationing via drop-shipping in Internet retailing: a sensitivity analysis. *European Journal of Operational Research*, 171, no. 1, pp. 135–152 [64, 80]

[21] Bakshi, Y. and D.A. Hoeflin (2006), Quantile estimation: a minimalist approach. *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 2140–2145 [16]

[22] Banks, J., J.S. Carson, B.L. Nelson, and D.M. Nicol (2005), *Discrete-event simulation, fourth edition.* Prentice-Hall, Upper Saddle River, N.J. [6]

[23] Barnes, E.R. (1986), A variation on Karmarkar's algorithm for solving linear programming problems. *Mathematical Programming*, 36, pp. 174–182 [111]

[24] Bartkute, V. and L. Sakalauskas (2007), Simultaneous perturbation stochastic approximation of nonsmooth functions. *European Journal of Operational Research*, in press [103]

[25] Barton, R.R. and M. Meckesheimer (2006), Metamodel-based simulation optimization. Chapter 18 in *Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, Elsevier/North Holland, pp. 535–574 [8, 9, 51, 102, 104, 105, 106]

[26] Bartz-Beielstein, T. (2003), Experimental analysis of evolutionary strategies; overview and comprehensive introduction. Technical report no. CI-157/03, University Dordmund, Dordmund, Germany [65, 66, 86, 105]

[27] Bartz-Beielstein, T. (2006), *Experimental research in evolutionary computation; the new experimentalism.* Springer, Berlin [65, 103, 104]

[28] Bashyam, S. and M.C. Fu (1998), Optimization of (s, S) inventory systems with random lead times and a service level constraint. *Management Science*, 44, pp. 243–256 [114]

[29] Bates, S.J., J. Sienz, and V.V. Toropov (2005), Formulation of the optimal Latin Hypercube design of experiments using a permutation genetic algorithm. *45th AIAA/ASME/ASCE/AHS/ASC SDM Conference*, Palm Springs, California, paper AIAA-2004–2011 [130]

[30] Baudrit, C., I. Couso, and D. Dubois (2007), Joint propagation of probability and possibility in risk analysis: towards a formal framework. *International Journal of Approximate Reasoning*, in press [124]

[31] Bayarri, M.J. et al. (2005), A framework for validation of computer models. Working Paper, National Institute of Statistical Sciences (NISS), Research Triangle Park, North Carolina [7, 126]

[32] Bayhan, G.M. (2004), An alternative procedure for the estimation problem in 2^n factorial experimental models. *Computers & Industrial Engineering*, 47, pp. 1–15 [69]

[33] Ben-Gal, I. and J. Bukchin (2002), Ergonomic design of working environment via rapid prototyping tools and design of experiments, *IIE Transactions*, 34, no. 4, pp. 375–391 [105]

[34] Ben-Tal, A. and A. Nemirovski (2002), Robust optimization: methodology and applications. *Mathematical Programming*, 92, no. 3, pp. 453–380 [131]

[35] Bertaccini, B. and R. Varriale (2007), Robust ANalysis Of VAriance: an approach based on the forward search. *Computational Statistics & Data Analysis*, in press [81]

[36] Bertsimas, D., D. Pachamanova, and M. Sim (2004), Robust linear optimization under general norms. *Operations Research Letters*, 32, pp. 510–516 [131]

[37] Bertsimas, D. and A. Thiele (2006), A robust optimization approach to inventory theory. *Operations Research*, 54, no. 1, pp. 150–168 [131]

[38] Bettonvil, B. (1990), *Detection of important factors by sequential bifurcation.* Ph.D. dissertation, Tilburg University Press, Tilburg [160, 163, 164, 170]

[39] Bettonvil, B., E. del Castillo, and J.P.C. Kleijnen (2006), Statistical testing of optimality conditions in multiresponse simulation-based optimization. Working Paper, Tilburg University, Tilburg, Netherlands [101, 118, 123]

[40] Bettonvil, B. and J.P.C. Kleijnen (1990), Measurement scales and resolution IV designs. *American Journal of Mathematical and Management Sciences*, 10, nos. 3 & 4, pp. 309–322 [30]

[41] Bettonvil, B. and J.P.C. Kleijnen (1997), Searching for important factors in simulation models with many factors: sequential bifurcation. *European Journal of Operational Research*, 96, pp. 180–194 [158, 160, 170]

[42] Biles, W.E., J.P.C. Kleijnen, W.C. M. van Beers, and I. van Nieuwenhuyse (2007), Kriging metamodeling in constrained simulation optimization: an explorative study. *Proceedings of the 2007 Winter Simulation Conference*, edited by S.G. Henderson, B. Biller, M.H. Hsieh, J. Shortle, J.D. Tew, and R.R. Barton, under review [103, 140]

[43] Billingsley, P. (1968), *Convergence of probability measures.* Wiley, New York [79]

[44] Borgonovo, E. and L. Peccati (2007), Global sensitivity analysis in inventory management. *International Journal of Production Economics*, in press [3, 41, 57, 123]

[45] Bourlakis, M.A. and P.W.H. Weightman, editors (2004), *Food supply chain management.* Blackwell Publishing, Oxford [124]

[46] Box, G.E.P. (1952), Multi-factor designs of first order. *Biometrika*, 39, no. 1, pp. 49–57 [35]

[47] Box, G.E.P. (1999), Statistics as a catalyst to learning by scientific method, part II - a discussion. *Journal of Quality Technology*, 31, no. 1, pp. 16–29 [105]

[48] Box, G.E.P. and N.R. Draper (1959), A basis for the selection of a response surface design. *Journal American Statistical Association*, 54, pp. 622–654 [46]

[49] Box, G.E.P. and J.S. Hunter (1961), The 2^{k-p} fractional factorial designs, Part I. *Technometrics*, 3, pp. 311–351 [36]

[50] Box, G.E.P. and J.S. Hunter (1961), The 2^{k-p} fractional factorial designs, Part II. *Technometrics*, 3, pp. 449–458 [47]

[51] Box, G.E.P. and K.B. Wilson (1951), On the experimental attainment of optimum conditions. *Journal Royal Statistical Society, Series B*, 13, no. 1, pp. 1–38 [9, 42, 104, 105]

[52] Breiman, L. and J.H. Friedman (1997), Predicting multivariate responses in multiple linear regression. *Journal Royal Statistical Society, Series B*, 59, no. 1, pp. 3–54 [22]

[53] Brekelmans, R., L. Driessen, H. Hamers, and D. den Hertog (2005), Gradient estimation schemes for noisy functions. *Journal of Optimization Theory and Applications,* 126, no. 3, pp. 529–551 [107, 118]

[54] Breukers, A. (2006), *Bio-economic modelling of brown rot in the Dutch potato production chain.* Doctoral dissertation, Wageningen University, Wageningen, The Netherlands [31, 62, 77]

[55] Burke, E.K. and G. Kendall (2005), editors, *Search methodologies; introductory tutorials in optimization and decision support techniques*. Springer. [103]

[56] Bursztyn, D. and D.M. Steinberg (2006), Comparison of designs for computer experiments. *Journal of Statistical Planning and Inference*, 136, pp. 1103–1119 [52]

[57] Campolongo, F., J. Cariboni, and A. Saltelli (2007), An effective screening design for sensitivity analysis of large models. *Environmental Modelling & Software*, in press [159]

[58] Campolongo, F., J.P.C. Kleijnen, and T. Andres (2000), Screening methods. In: *Sensitivity Analysis*, edited by A. Saltelli, K. Chan, and E.M. Scott, Wiley, Chichester (England), pp. 65–89 [159]

[59] Cangussu, J.W., R.A. DeCarlo, and A.P. Mathur (2003), Using sensitivity analysis to validate a state variable model of the software test process. *IEEE Transactions on Software Engineering*, 29, no. 5, pp. 430–443 [19]

[60] Ceranka, B., M. Graczyk, and K. Katulska (2006), A-optimal chemical balance weighing design with nonhomogeneity of variances of errors *Statistics & Probability Letter*s, 76, no. 7, pp. 653–665 [53, 92]

[61] Chang, J.Z., J.P. Allebach, and C.A. Bouman (1997), Sequential linear interpolation of multidimensional functions. IEEE Transactions on Image Processing, 6, no. 9, pp. 1231–1245 [155]

[62] Chatterjee, K., A. Sarkar and D.K.J. Lin (2007), Supersaturated designs with high searching probability. *Journal of Statistical Planning and Inference,* in press [159]

[63] Chen, E.J. and W.D. Kelton (2006), Quantile and tolerance-interval estimation in simulation. *European Journal of Operational Research*, 168, no. 2, pp. 520–540 [16]

[64] Chen, H. and C. Yeh (2005), Asymptotic results for batch-variance methods in simulation output analysis. *Computers & Industrial Engineering*, 48, no. 1, pp. 23–37 [16]

[65] Chen, V.C.P., K.-L. Tsui, R.R. Barton, and J.K. Allen (2003), A review of design and modeling in computer experiments. In: *Handbook of Statistics; Volume 22*, edited by R. Khattree and C.R. Rao, Elsevier, Amsterdam, pp. 231–261 [3, 8, 9, 52, 123, 142, 146]

[66] Chen, V.C.P., K.-L. Tsui, R.R. Barton, and M. Meckesheimer (2006), A review on design, modeling, and applications of computer experiments. *IIE Transactions*, 38, pp. 273–291 [3, 8, 123, 127, 130, 131, 146]

[67] Cheng, R.C.H. (1997), Searching for important factors: sequential bifurcation under uncertainty. *Proceedings of the 1997 Winter Simulation Conference*, edited by S. Andradóttir, K.J. Healy, D.H. Withers, and B.L. Nelson, pp. 275–280 [160]

[68] Cheng, R.C.H. (2006), Resampling methods. *Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, pp. 415–453 [63, 84, 85, 86, 102, 126]

[69] Cheng, R.C.H. (2006), Validating and comparing simulation models using resampling. *Journal of Simulation*, 1, pp. 53–63 [80, 84, 124]

[70] Cheng, R.C.H. and W. Holland (1999). Stochastic sequential bifurcation: practical issues. *Proceedings of UK Simulation Society Conference*, edited by D. Al-Dabass and R.C.H. Cheng, UKSIM, Nottingham, pp. 74–80 [160]

[71] Cheng, R.C.H. and J.P.C. Kleijnen (1999), Improved design of queueing simulation experiments with highly heteroscedastic responses. *Operations Research*, 47, no. 5, pp. 762–777 [88, 152]

[72] Cheng, R.C.H., J.P.C. Kleijnen, and V.B. Melas (2000), Optimal design of experiments with simulation models of nearly saturated queues. *Journal of Statistical Planning and Inference,* 85, no. 1–2, pp. 19–26 [88]

[73] Chernik, M.R. (1999), *Bootstrap methods; a practitioner's guide.* Wiley, New York [84]

[74] Chick, S.E. (2006), Subjective probability and Bayesian methodology. *Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, Elsevier/North Holland, pp. 225–257 [126]

[75] Chick, S.E. (2006), Bayesian ideas and discrete event simulation: why, what and how.*Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 96–106 [126]

[76] Chipman, H. (2006), Prior distributions for Bayesian analysis of screening experiments. In: *Screening: methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 235–267 [160]

[77] Cho, K. and W.-Y. Loh (2006), Bias and convergence rate of the coverage probability of prediction intervals in Box Cox transformed linear models. *Journal of Statistical Planning and Inference*, 136, no. 10, pp. 3614–3624 [81]

[78] Chootinan, P. and A. Chen (2006), Constraint handling in genetic algorithms using a gradient-based repair method. *Computers & Operations Research*, 33, no. 8, pp. 2263–2281 [103]

[79] Christopher, M. and H. Peck (2005), Building the resilient supply chain. Working Paper, Cranfield School of Management [131]

[80] Clarke, S.M., J.H Griebsch, and T.W., Simpson (2003), Analysis of support vector regression for approximation of complex engineering analyses. *Proceedings of DETC '03, ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Chicago. [8]

[81] Conover, W.J. (1999), *Practical nonparametric statistics: third edition.* Wiley, New York [11, 57, 88]

[82] Conover, W.J. and R.L. Iman (1981), Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician*, 35, no. 3, pp124–133 [98]

[83] Conway, R.W. (1963), Some tactical problems in digital simulation. *Management Science*, 10, no 1, pp. 47–61 [9]

[84] Crary, S.B. (2002), Design of computer experiments for metamodel generation. *Analog Integrated Circuits and Signal Processing*, 32, pp. 7–16 [155]

[85] Crary, S.B., D.M. Woodcock, and A. Hieke (2001), Designing efficient computer experiments for metamodel generation. *Proceedings Modeling and Simulation of Microsystems Conference*, pp. 132–135 [144]

[86] Cressie, N.A.C. (1993), *Statistics for spatial data: revised edition.* Wiley, New York [140, 148]

[87] Cuyt, A., R.B. Lenin, S. Becuwe, and B. Verdonk (2003), Adaptive multivariate rational data fitting with applications in electromagnetics. Working Paper, Antwerp University, Antwerp, Belgium [8]

[88] Dangerfield, B. and C. Roberts (1996), An overview of strategy and tactics in system dynamics optimization, *Journal of the Operational Research Society*, 47, pp. 405–423 [104]

[89] Davidson, R. and J.G. MacKinnon (2007), Improving the reliability of bootstrap tests with the fast double bootstrap. *Computational Statistics & Data Analysis*, in press [84, 96]

[90] Davison, A.C. and D.V. Hinkley (1997), *Bootstrap methods and their application.* Cambridge University Press, Cambridge [84]

[91] Den Hertog, D., J.P.C. Kleijnen, and A.Y.D. Siem (2006), The correct Kriging variance estimated by bootstrapping. *Journal of the Operational Research Society*, 57, no. 4, pp. 400–409 [148]

[92] Denardo, E.V. (2001), The science of decision-making. *OR/MS Today*, pp. 30–32 [3]

[93] Dengiz, B., T. Bektas, and A. E. Ultanir (2006), Simulation optimization based DSS application: a diamond tool production line in industry. *Simulation Modelling Practice and Theory*, 14, no. 3, pp. 296–312 [31, 64, 111]

[94] Deschepper, E., O. Thas and J.P. Ottoy (2006), Regional residual plots for assessing the fit of linear regression models. *Computational Statistics & Data Analysis*, 50, no. 8, pp. 1995–2013 [26, 87]

[95] De Vos, C, H.W. Saatkamp, M. Nielen, and R.B.M. Huirne (2006), Sensitivity analysis to evaluate the impact of uncertain factors in a scenario tree model for classical swine fever introduction. *Risk Analysis*, 26, no. 5, pp. 1311–1322 [159, 163]

[96] Donohue, J.M. (1995), The use of variance reduction techniques in the estimation of simulation metamodels. *Proceedings of the 1995 Winter Simulation Conference*, edited by C. Alexopoulos, K. Kang, W.R. Lilegdon, and D. Goldsman, Institute of Electrical and Electronics Engineers, Piscataway, NJ, pp. 195–199 [53, 96]

[97] Downing, D.J., R.H. Gardner, and F.O. Hoffman (1985), An examination of Response-Surface Methodologies for uncertainty analysis in assessment models. *Technometrics*, 27, no. 2, pp. 151–163 (Discussion: 1986, 28, no. 1, pp. 91–93) [9, 104]

[98] Draper, N.R. and H. Smith (1981), *Applied regression analysis; second edition.* Wiley, New York [54]

[99] Dréo, J., A. Pétrowski, E. Taillard, and A. Chatterjee (2006), *Metaheuristics for hard optimization; methods and case studies.* Springer, New York [103]

[100] Dudewicz, E.J., Y. Ma, E. Mai, and H. Su (2007), Exact solutions to the Behrens–Fisher problem. *Journal of Statistical Planning and Inference*, in press [58]

[101] Durieux, S. and H. Pierreval (2004), Regression metamodeling for the design of automated manufacturing system composed of parallel machines sharing a material handling resource. *International Journal of Production Economics*, 89, pp. 21–30 [64]

[102] Dykstra, R.L. (1970), Establishing the positive definiteness of the sample covariance matrix. *The Annals of Mathematical Statistics*, 41, no. 6, pp. 2153–2154 [95]

[103] Efron, B. (1982), *The jackknife, the bootstrap and other resampling plans.* CBMS-NSF Series, Siam, Philadelphia [84]

[104] Efron, B. and R.J. Tibshirani (1993), *An introduction to the bootstrap.* Chapman & Hall, New York [81, 84]

[105] El Tabach, E., L. Lancelot, I. Shahrour, and Y. Najjar (2007), Use of artificial neural network simulation metamodelling to assess groundwater contamination in a road project. *Mathematical and Computer Modelling*, in press [8, 59]

[106] Eldred, M.S., A.A. Giunta, S.F. Wojtkiewicz, and T.G. Trucano (2002), Formulations for surrogate-based optimization under uncertainty. *Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization,* Atlanta, GA, Sept. 4–6, 2002, paper AIAA-2002-5585 [126, 127]

[107] Elmaghraby, S.E. (1995), Activity nets: a guided tour through some recent developments. *European Journal of Operational Research*, 82, no. 3, pp. 383–408 [124]

[108] Els, P.S., P.E. Uys, J.A. Snyman, and M.J. Thoresson (2006), Gradient-based approximation methods applied to the optimal design of vehicle suspension systems using computational models with severe inherent noise. *Mathematical and Computer Modelling*, 43, no. 7 & 8, pp. 787–801 [104, 118]

[109] Elster, C. and A. Neumaier (1995), Screening by conference designs. *Biometrika*, 82, no. 3, pp. 589–602 [35, 159]

[110] Evans, J.R. and D.L. Olson (1998), *Introduction to simulation and risk analysis.* Prentice-Hall, Upper Saddle River, New Jersey [3, 123]

[111] Fedorov, V.V. (1972), *Theory of optimal experiments.* Academic Press, New York [52]

[112] Feyzioglu, O., H. Pierreval, and D. Deflandre (2005), A simulation-based optimization approach to size manufacturing systems. *International Journal of Production Research*, 43, no. 2, pp. 247–266 [87, 103]

[113] Forrester, J.W. (1961), *Industrial dynamics.* MIT Press, Cambridge, Massachusetts [3]

[114] Fredricks, G.A. and R.B. Nelsen (2006), On the relationship between Spearman's rho and Kendall's tau for pairs of continuous random variables. *Journal of Statistical Planning and Inference*, in press [57]

[115] Freeman, J. and R. Modarres (2006), Inverse Box Cox: the power-normal distribution. *Statistics & Probability Letters*, 76, no. 8, pp. 764–772 [81]

[116] Fu, M.C. (2002), Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing*, 14, no. 3, pp. 192–215 [103]

[117] Fu, M.C. (2006), Gradient estimation. *Handbooks in Operations Research and Management Science, Volume 13*, Elsevier/North Holland, pp. 575–616 [17, 103]

[118] Fu, M.C., F.W. Glover, and J. April (2005), Simulation optimization: a review, new developments, and applications. *Proceedings of the 2005 Winter Simulation Conference*, edited by M.E. Kuhl, N.M. Steiger, F.B. Armstrong, and J.A. Joines, Institute of Electrical and Electronics Engineers, Piscataway, New Jersey, pp. 83–95 [102, 104, 124]

[119] Fu, M. C. and S. D. Hill (1997), Optimization of discrete event systems via simultaneous perturbation stochastic approximation. *IIE Transactions*, 29, pp. 233–243 [103]

[120] Gano, S.E., J.E. Renaud, J.D. Martin, and T.W. Simpson (2006), Update strategies for Kriging models for using in variable fidelity optimization. *Structural and Multidisciplinary Optimization*, 32, no. 4, pp. 287–298 [149, 155]

[121] Gel, Y.R., W. Miao, and J.L. Gastwirth (2007), Robust directed tests of normality against heavy-tailed alternatives. *Computational Statistics & Data Analysis*, in press [79]

[122] Georgiou, S.D. (2007), New two-variable full orthogonal designs and related experiments with linear regression models. *Statistics & Probability Letters*, in press [35, 159]

[123] Ghosh, B.K. and P.K. Sen (editors) (1991), *Handbook of sequential analysis.* Marcel Dekker, New York [149]

[124] Ghosh, S. and Y. Tian (2006), Optimum two level fractional factorial plans for model identification and discrimination. *Journal of Multivariate Analysis*, 97, no. 6, pp. 1437–1450 [52]

[125] Giambiasi, N. and J.C. Carmona (2006), Generalized discrete event abstraction of continuous systems: GDEVS formalism. *Simulation Modelling: Practice and Theory*, 14, no. 1, pp. 47–70 [6]

[126] Gilbert, S. and P. Zemčík (2006), Who's afraid of reduced-rank parameterizations of multivariate models? Theory and example. *Journal of Multivariate Analysis*, 97, no. 4, pp. 925–945 [77]

[127] Gill, P.E., W. Murray, and M.H. Wright (2000). *Practical optimization, 12th edition.* Academic Press, London [116, 117]

[128] Gilmour, S.G. (2006), Factor screening via supersaturated designs. In: *Screening: Methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 169–190 [159]

[129] Giunta, A.A., McFarland, J.M., Swiler, L.P., and M.S. Eldred (2006), The promise and peril of uncertainty quantification using response surface approximations. *Structure and Infrastructure Engineering*, 2, nos. 3 - 4, pp. 175–189 [8, 104, 125, 127, 129]

[130] Glasserman, P. (1991), *Gradient estimation via perturbation analysis.* Kluwer, Dordrecht (Netherlands) [103]

[131] Gluhovsky, I. (2007), Determining output uncertainty of computer system models. *Performance Evaluation*, 64, pp. 103–125 [126]

[132] Godfrey, L.G. (2006), Tests for regression models with heteroskedasticity of unknown form. *Computational Statistics & Data Analysis,* 50, no. 10, pp. 2715–2733 [85, 91]

[133] Goel, T., R. T. Haftka, W. Shyy, and N.V. Queipo (2007), Ensemble of surrogates. *Structural and Multidisciplinary Optimization,* in press [8, 104]

[134] Good, P.I. (2005), *Resampling methods: a practical guide to data analysis; third edition.* Birkhäuser, Boston [84]

[135] Gordon, A.Y. (2007), Unimprovability of the Bonferroni procedure in the class of general step-up multiple testing procedures. *Statistics & Probability Letters,* in press [61]

[136] Grant, F. (2006), Planning for the unexpected. *Scientific Computing World*, no. 86, pp. 24–27 [53]

[137] Haijema, R., N. van Dijk, J. van der Wal, and C. Smit Sibinga (2007), Blood platelet production with breaks: optimization by SDP and simulation. *International Journal of Production Economics,* in press [103]

[138] Hamad, H. and S. Al-Hamdan (2007), Discovering metamodels quality-of-fit for simulation via graphical techniques. *European Journal of Operational Research*, 178, no. 2, pp. 543–559 [63, 98]

[139] Hankin, R.K.S. (2005), Introducing BACCO, an R bundle for Bayesian analysis of computer code output. *Journal of Statistical Software*, 14, no. 16, pp. 1–21 [60, 126, 146]

[140] Hartley, H.O. (1950), The maximum F-ratio as a short-cut test for heterogeneity of variance. *Biometrika*, 50, pp. 187–194 [88]

[141] Hedayat, A.S., N.J.A. Sloane, and J. Stufken (1999), *Orthogonal arrays: theory and applications*. Springer, New York [48]

[142] Heidergott, B., F.J. Vázquez-Abad, and W. Volk-Makarewicz (2007), Sensitivity estimation for Gaussian systems. *European Journal of Operational Research*, in press [103]

[143] Helton, J.C., F.J. Davis, and J.D. Johnson (2005), A comparison of uncertainty and sensitivity results obtained with random and Latin hypercube sampling. *Reliability Engineering and Systems Safety*, 89, pp. 305–330 [127, 129]

[144] Helton, J.C., J.D. Johnson, W.D. Oberkampf, and C.J. Sallaberry (2006), Sensitivity analysis in conjunction with evidence theory representations of epistimic uncertainty. *Reliability Engineering and Systems Safety*, 91, pp. 1414–1434 [124, 129]

[145] Helton, J.C., J.D. Johnson, C.J. Sallaberry, and C.B. Storlie (2006), Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering and Systems Safety*, 91, pp. 1175–1209 [8, 18, 41, 57, 98, 124, 126, 128, 129]

[146] Henderson, S.G. and B.L. Nelson (2006), editors, *Handbooks in operations research and management science, Volume 13*, Elsevier/North Holland [6, 102]

[147] Ho, Y. and X. Cao (1991), *Perturbation analysis of discrete event dynamic systems*. Kluwer, Dordrecht (Netherlands) [4, 17, 103]

[148] Ho, Y., Q-C. Zhao, and Q.S. Jia (2007), *Ordinal optimization: soft computing for hard problems*. Springer, New York [102]

[149] Holland, W. (2005), Crystal Ball v7.01 Professional. *OR/MS Today*, 32, no. 2, pp. 54–57 [126]

[150] Hong, L.J. and B.L. Nelson (2006), Discrete optimization via simulation using COMPASS. *Operations Research*, 54, no. 1, pp. 115–129 [103, 104]

[151] Hsieh, K-L. and Y-K. Chen (2007), Bootstrap confidence interval estimates of the bullwhip effect. *Simulation Modelling Practice and Theory*, in press [84]

[152] Huang, D., T.T. Allen, W. Notz, and R.A. Miller (2006), Sequential Kriging optimization using multiple fidelity evaluation. Working Paper, Ohio State University [154]

[153] Huang, D., T.T. Allen, W. Notz, and N. Zheng (2006), Global optimization of stochastic black-box systems via sequential Kriging metamodels. *Journal of Global Optimization*, 34, 441–466 [86, 104, 127, 130, 154, 155]

[154] Hubert, M. and S. Engelen (2007), Fast cross-validation of high-breakdown resampling methods for PCA. *Computational Statistics & Data Analysis*, in press [152]

[155] Huele, A.F. and J. Engel (2006), A response surface approach to tolerance design. *Statistica Neerlandica*, 60, no. 3, pp. 379–395 [130]

[156] Hussain, M.F., R.R. Barton, and S.B. Joshi (2002), Metamodeling: radial basis functions versus polynomials. *European Journal of Operational Research*, 138, no.1, pp. 142–154 [8]

[157] Husslage, B.G.M. (2006), Maximin designs for computer experiments. Doctoral dissertation, Tilburg University, Tilburg, Netherlands [130]

[158] Husslage, B., G. Rennen, E.R. van Dam, and D. den Hertog (2006), Space-filling Latin hypercube designs for computer experiments. CentER Discussion Paper, no. 2006–18, Tilburg University, Tilburg, Netherlands [130]

[159] Huyet, A.L. (2006), Optimization and analysis aid via data-mining for simulated production systems. *European Journal of Operational Research*, 173, no. 3, pp. 827–838 [104]

[160] Incontrol (2003), Incontrol Enterprise Dynamics (www.EnterpriseDynamics.com) [135]

[161] Irizarry, M., J.R. Wilson, and J. Trevino (2001), A flexible simulation tool for manufacturing-cell design, II: response surface analysis and case study. *IIE Transactions*, 33, pp. 837–846 [105]

[162] Ivanescu, C., W. Bertrand, J. Fransoo, and J.P.C. Kleijnen (2006), Bootstrapping to solve the limited data problem in production control: an application in batch processing industries. *Journal of the Operational Research Society*, 57, number 1, pp. 2–9 [75, 102]

[163] Jia, Q.-S., Y.-C. Ho, and Q.-C. Zhao (2006), Comparison of selection rules for ordinal optimization. *Mathematical and Computer Modelling*, 43, no. 9–10, pp. 1150–1171 [102]

[164] Jin, R, W. Chen, and A. Sudjianto (2002), On sequential sampling for global metamodeling in engineering design. *Proceedings of DET '02, ASME 2002 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, DETC2002/DAC-34092, September 29 - October 2, 2002, Montreal, Canada [155]

[165] Jin, R, W. Chen, and A. Sudjianto (2005), An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference*, 134, no. 1, pp. 268–287 [128]

[166] Johnson, R.T., F. Yang, B.E. Ankenman, and B.L. Nelson (2004), Nonlinear regression fits for simulated cycle time vs. throughput curves for semiconductor manufacturing, *Proceedings of the 2004 Winter Simulation Conference*, eds. R.G. Ingalls, M.D. Rossetti, J.S. Smith, B.A. Peter, Institute of Electrical and Electronics Engineers, Piscataway, NJ, pp. 1951–1955 [8]

[167] Jones, B.A., W. Li, C.J. Nachtsheim, and K.Q. Ye (2007), Model discrimination another perspective on model-robust designs. *Journal of Statistical Planning and Inference*, in press [52]

[168] Jones, D.R., M. Schonlau, and W.J. Welch (1998), Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13, pp. 455–492 [9, 41, 104, 140, 154, 155]

[169] Joshi, S., H.D. Sherali, and J.D. Tew (1998), An enhanced response surface methodology (RSM) algorithm using gradient deflection and second-order search strategies. *Computers and Operations Research*, 25, no. 7/8, pp. 531–541 [118]

[170] Jurečková, J. and J. Picek (2007), Shapiro–Wilk-type test of normality under nuisance regression and scale. *Computational Statistics & Data Analysis*, in press [79]

[171] Kao, C. and S.-P. Chen (2006), A stochastic quasi-Newton method for simulation response optimization. *European Journal of Operational Research*, 173, no. 1, pp. 30–46 [114, 118]

[172] Karaesman, I and G. van Ryzin (2004), Overbooking with substitutable inventory classes. *Operations Research*, 52, no. 1, pp. 83–104 [117]

[173] Karplus, W.J (1983), The spectrum of mathematical models. *Perspectives in Computing*, 3, no. 2, pp. 4–13 [10]

[174] Kelton, W.D., R.P. Sadowski, and D.T. Sturrock (2004), *Simulation with Arena; third edition.* McGraw-Hill, Boston [5]

[175] Keys, A.C. and L.P. Rees (2004), A sequential-design metamodeling strategy for simulation optimization. *Computers & Operations Research*, 31, no. 11, pp. 1911–1932 [104, 114, 154]

[176] Khuri, A.I. (1996), Multiresponse surface methodology. *Handbook of Statistics, volume 13*, edited by S. Ghosh and C. R. Rao, Elsevier, Amsterdam [110]

[177] Khuri, A.I. and J.A. Cornell (1996), *Response surfaces: design and analysis, second edition.* Marcel Dekker, New York [105]

[178] Kiefer, J. and J. Wolfowitz (1959), Optimum designs in regression problems. *Annals Mathematical Statistic*s, 30, pp. 271–294 [52]

[179] Kim, S., C. Alexopoulos, K. Tsui, and J.R. Wilson (2007), Distribution-free tabular CUSUM chart for autocorrelated data. *IIE Transactions*, 39, pp. 317–330 [79]

[180] Kim, S. and B.L. Nelson (2006), Selecting the best system. *Handbooks in Operations Research and Management Science, Volume 13*, pp. 501–534 [102]

[181] Kleijnen, J.P.C. (1974/1975), *Statistical techniques in simulation (in two parts).* Marcel Dekker, New York (Russian translation, Publishing House 'Statistics', Moscow, 1978) [2, 39, 42, 44, 45, 47, 48, 58, 66, 102, 105, 159]

[182] Kleijnen, J.P.C. (1975), A comment on Blanning's metamodel for sensitivity analysis: the regression metamodel in simulation. *Interfaces*, 5, no. 3, pp. 21–23 [8]

[183] Kleijnen, J.P.C. (1983), Cross-validation using the t statistic. *European Journal of Operational Research*, 13, no. 2, pp. 133–141 [60]

[184] Kleijnen, J.P.C. (1987), *Statistical tools for simulation practitioners.* Marcel Dekker, New York [12, 22, 25, 45, 51, 52, 79, 88]

[185] Kleijnen, J.P.C. (1992), Regression metamodels for simulation with common random numbers: comparison of validation tests and confidence intervals. *Management Science*, 38, no. 8, pp. 1164–1185 [90, 95, 96, 97, 100]

[186] Kleijnen, J.P.C. (1993), Simulation and optimization in production planning: a case study. *Decision Support Systems*, 9, pp. 269–280 [75, 105, 119]

[187] Kleijnen, J.P.C. (1994), Sensitivity analysis versus uncertainty analysis: when to use what? *Predictability and nonlinear modelling in natural sciences and economics,* edited by J. Grasman and G. van Straten, Kluwer, Dordrecht, Netherlands, pp. 322–333 [125]

[188] Kleijnen, J.P.C. (1995), Case study: statistical validation of simulation models. *European Journal of Operational Research*, 87, no. 1, pp. 21–34 [30]

[189] Kleijnen, J.P.C. (1995), Sensitivity analysis and optimization of system dynamics models: regression analysis and statistical design of experiments. *System Dynamics Review*, 11, no. 4, pp. 275–288 [78]

[190] Kleijnen, J.P.C. (1997), Sensitivity analysis and related analyses: a review of some statistical techniques. *Journal Statistical Computation and Simulation*, 57, nos. 1–4, pp. 111–142 [125]

[191] Kleijnen, J.P.C. (1998), Experimental design for sensitivity analysis, optimization, and validation of simulation models. *Handbook of simulation*, edited by J. Banks, Wiley, New York, pp. 173–223 [10, 20, 105]

[192] Kleijnen, J.P.C. (2001), Comments on M.C. Kennedy & A. O'Hagan's Bayesian calibration of computer models. *Journal Royal Statistical Society, Series B*, 63, Part 3, 2001, pp. 458–459 [126]

[193] Kleijnen, J.P.C. (2005), Invited review: an overview of the design and analysis of simulation experiments for sensitivity analysis. *European Journal of Operational Research*, 164, no. 2, pp. 287–300 [2]

[194] Kleijnen, J.P.C., B. Bettonvil, and F. Persson (2003), Robust solutions for supply chain management: simulation, optimization, and risk analysis. Working Paper, Tilburg University, Tilburg, Netherlands [132, 171]

[195] Kleijnen, J.P.C., B. Bettonvil, and F. Persson (2006), Screening for the important factors in large discrete-event simulation: sequential bifurcation and its applications. In: *Screening: Methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 287–307 [135, 158, 160, 167, 171]

[196] Kleijnen, J.P.C., B. Bettonvil, and W. van Groenendaal (1998), Validation of trace-driven simulation models: a novel regression test. *Management Science*, 44, no. 6, pp. 812–819 [60]

[197] Kleijnen, J.P.C., R.C.H. Cheng, and B. Bettonvil (2001),Validation of trace-driven simulation models: bootstrapped tests. *Management Science*, 47, no. 11, pp. 1533–1538 [85, 86]

[198] Kleijnen, J.P.C., P. Cremers, and F. van Belle (1985), The power of weighted and ordinary least squares with estimated unequal variances in experimental designs. *Communications in Statistics, Simulation and Computation*, 14, no. 1, pp. 85–102 [91]

[199] Kleijnen, J.P.C. and D. Deflandre (2003), Statistical analysis of random simulations: bootstrap tutorial. *Simulation News Europe*, issue 38/39, pp. 29–34 [87, 100]

[200] Kleijnen, J.P.C. and D. Deflandre (2006), Validation of regression metamodels in simulation: bootstrap approach. *European Journal of Operational Research*, 170, no. 1, pp. 120–131 [55, 86, 91, 97, 100]

[201] Kleijnen, J.P.C., D. den Hertog, and E. Angün (2004), Response surface methodology's steepest ascent and step size revisited. *European Journal of Operational Research*, 159, pp. 121–131 [101, 107, 109]

[202] Kleijnen, J.P.C., D. den Hertog, and E. Angün (2006), Response surface methodology's steepest ascent and step size revisited: correction. *European Journal of Operational Research*, 170, pp. 664–666 [107, 109]

[203] Kleijnen, J.P.C. and E.G.A. Gaury (2003), Short-term robustness of production-management systems: a case study. *European Journal of Operational Research*, 148, no. 2, pp. 452–465 [125, 132]

[204] Kleijnen, J.P.C. and J.C. Helton (1999), Statistical analyses of scatter plots to identify important factors in large-scale simulations, 1: review and comparison of techniques. *Reliability Engineering and Systems Safety*, 65, no. 2, pp. 147–185 [57, 81, 98, 125]

[205] Kleijnen, J.P.C., P.C.A. Karremans, W.K. Oortwijn, and W.J.H. van Groenendaal (1987), Jackknifing estimated weighted least squares: JEWLS. *Communications in Statistics, Theory and Methods*, 16, no. 3, pp. 747–764 [83, 91, 100]

[206] Kleijnen, J.P.C., F. Keijzer, E. Mullenders, and A. van Reeken (1981), Optimization of priority class queues, with a computer center case study. *American Journal of Mathematical and Management Sciences*, 1, no. 4, pp. 341– 358 (Reprinted in: Dudewicz, E.J. and Z.A. Karian, *Modern design and analysis of discrete-event computer simulations.* IEEE Computer Society Press, Washington D.C., 1985, pp. 298–310) [105]

[207] Kleijnen, J.P.C., J. Kriens, H. Timmermans, and H. Van den Wildenberg (1989), Regression sampling in statistical auditing: a practical survey and evaluation (including Rejoinder). *Statistica Neerlandica*, 43, no. 4, pp. 193–207 (p. 225) [83]

[208] Kleijnen, J.P.C. and O. Pala (1999), Maximizing the simulation output: a competition. *Simulation*, 73, no. 3, pp. 168–173 [48]

[209] Kleijnen, J.P.C. and P.J. Rens (1978), IMPACT revisited: a critical analysis of IBM's inventory package "IMPACT". *Production and Inventory Management, Journal of the American Production and Inventory Control Society*, 19, no. 1, pp. 71–90 [75]

[210] Kleijnen, J.P.C., S.M. Sanchez, T.W. Lucas, and T.M. Cioppa (2005), State-of-the-art review: a user's guide to the brave new world of designing simulation experiments. *INFORMS Journal on Computing*, 17, no. 3, pp. 263–289 [9]

[211] Kleijnen, J.P.C. and R.G. Sargent (2000). A methodology for the fitting and validation of metamodels in simulation. *European Journal of Operational Research*, 120, no. 1, pp. 14–29 [27, 32, 54, 63, 64, 131]

[212] Kleijnen, J.P.C. and C. Standridge (1988), Experimental design and regression analysis: an FMS case study. *European Journal of Operational Research*, 33, no. 3, pp. 257–261 [41]

[213] Kleijnen, J.P.C. and W.C.M. van Beers (2004), Application-driven sequential designs for simulation experiments: Kriging metamodeling. *Journal of the Operational Research Society,* 55, no. 9, pp. 876–883 [96, 146, 149]

[214] Kleijnen, J.P.C. and W.C.M. van Beers (2005), Robustness of Kriging when interpolating in random simulation with heterogeneous variances: some experiments. *European Journal of Operational Research*, 165, no. 3, pp. 826–834 [147]

[215] Kleijnen, J.P.C., A.J. van den Burg, and R.Th. van der Ham (1979), Generalization of simulation results: practicality of statistical methods. *European Journal of Operational Research*, 3, pp. 50–64 [53]

[216] Kleijnen, J.P.C. and W. van Groenendaal (1992), *Simulation: a statistical perspective.* Wiley, Chichester (England) [30, 61, 63, 83, 100]

[217] Kleijnen, J.P.C. and W. van Groenendaal (1995), Two-stage versus sequential sample-size determination in regression analysis of simulation experiments. *American Journal of Mathematical and Management Sciences*, 15, nos. 1&2, pp. 83–114 [92, 93]

[218] Kleijnen, J.P.C., G. van Ham, and J. Rotmans (1992), Techniques for sensitivity analysis of simulation models: a case study of the CO_2 greenhouse effect. *Simulation*, 58, no. 6, pp. 410–417 [78]

[219] Kleijnen, J.P.C. and J. Wan (2007), Optimization of simulated systems: OptQuest and alternatives. *Simulation Modelling Practice and Theory*, 15, pp. 354–362 [104]

[220] Koch, P.N., B. Wujek, O. Golovidov, and T.W. Simpson (2002), Facilitating probabilistic multidisciplinary design optimization using Kriging approximation models, 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, AIAA-2002-5415 [126, 131]

[221] Koch, P.N., R.-J. Yang, and L. Gu (2004), Design for six sigma through robust optimization. *Journal of Structural and Multidisciplinary Optimization*, 26, no. 3-4, pp. 235–248 [131]

[222] Koehler, J.R. and A.B. Owen (1996), Computer experiments. *Handbook of statistics, volume 13*, edited by S. Ghosh and C.R. Rao, Elsevier, Amsterdam, pp. 261–308 [9]

[223] Kolaiti, E. and C. Koukouvinos (2006), On the use of three level orthogonal arrays in robust parameter design. *Statistics & Probability Letters*, 76, no. 3, pp. 266–273 [133]

[224] Kuhfeld, W.F. and C. Suen (2005), Some new orthogonal arrays OA($4r$, $r^1 2^p, 2$). *Statistics & Probability Letters*, 75, no. 3, pp. 169–178 [52]

[225] Kumar, S. and D.A. Nottestad (2006), Capacity design: an application using discrete-event simulation and designed experiments. *IIE Transactions*, 38, pp. 729–736 [66]

[226] Lahiri, S.N. (2003), *Resampling methods for dependent data.* Springer, New York [84]

[227] Law, A.M. (2007), *Simulation modeling and analysis; fourth edition.* McGraw-Hill, Boston [6, 17, 27, 53, 75, 79, 95, 105, 151, 152]

[228] L'Ecuyer, P. and G. Perron (1994), On the convergence rates of IPA and FDC derivative estimators. *Operations Research*, 42, no. 4, pp. 643–656 [118]

[229] Lehmann, E.L. (1999), *Elements of large-sample theory*, Springer, New York [79]

[230] Lewis, S.M. and A.M. Dean (2001), Detection of interactions in experiments on large numbers of factors (including discussion). *Journal Royal Statistical Society B,* 63, pp. 633–672 [162]

[231] Li, L. and W. Li (2005), Tabu search and perturbation methods in the construction of supersaturated designs. *American Journal of Mathematical and Management Sciences*, 25, nos. 1 & 2, pp. 189–205 [159]

[232] Li, W. (2006), Screening designs for model selection. In: *Screening: Methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 207–234 [52]

[233] Liefvendahl, M. and R. Stocki (2006), A study on algorithms for optimization of Latin hypercubes. *Journal of Statistical Planning and Inference*, 136, no. 9, pp. 3231–3247 [52, 130]

[234] Lim, E. and P.W. Glynn (2006), Simulation-based response surface computation in the presence of monotonicity. *Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 264–271 [162]

[235] Lin, Y., F. Mistree, K.-L. Tsui, and J.K. Allen (2002), Metamodel validation with deterministic computer experiments. *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, September 4-6, 2002, Atlanta, GA, Paper Number AIAA-2002-5425* [63, 150, 153]

[236] Lin, Y., F. Mistree, J.K. Allen, K.-L. Tsui, and V.C.P. Chen (2004), A sequential exploratory design method: development of appropriate empirical models in design. *Proceedings of DETC 2004, 2004 ASME Design Engineering Technical Conference*, September 28 - October 2, 2004, Salt Lake City, Utah [154]

[237] Lin, Y., F. Mistree, J.K. Allen, K.-L. Tsui, and V.C.P. Chen (2004), Sequential metamodeling in engineering design. *10th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Albany, NY, August 30-September 1, 2004, Albany, NY, Paper Number AIAA-2004-4304* [8, 143, 146, 154]

[238] Liu, R. and A.B. Owen (2006), Estimating mean dimensionality of Analysis of Variance decompositions. *Journal of the American Statistical Association*, 101, no. 474, pp. 712–721 [41]

[239] Lophaven, S.N., H.B. Nielsen, and J. Sondergaard (2002), DACE: a Matlab Kriging toolbox, version 2.0. IMM Technical University of Denmark, Lyngby [146]

[240] Lloyd-Smith, B., A.A. Kist, R.J. Harris, and N. Shrestha (2004), Shortest paths in stochastic networks. *Proceedings 12th IEEE International Conference on Networks 2004*, volume 2, pp. 492–496 [124]

[241] Lunneborg, C.E. (2000), *Data analysis by resampling: concepts and applications*. Duxbury Press, Pacific Grove, California [84]

[242] Mandal, A. (2005), An approach for studying aliasing relations of mixed fractional factorials based on product arrays. *Statistics & Probability Letters*, 75, no. 3, pp. 203–210 [52, 130]

[243] Mandal, S. and B. Torsney (2006), Construction of optimal designs using a clustering approach. *Journal of Statistical Planning and Inference*, 136, no. 3, pp. 1120–1134 [52]

[244] Markiewicz, A. and A. Szczepanska (2007), Optimal designs in multivariate linear models. *Statistics & Probability Letters*, 77, pp. 426–430 [77]

[245] Marti, R. (2006), Editorial: Scatter search—wellsprings and challenges. *European Journal of Operational Research*, 169, no. 2, pp. 351–358 [103]

[246] Martin, J.D., and T.W. Simpson (2004), A Monte Carlo simulation of the Kriging model. *10th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-2004-4483* [146, 148]

[247] Martin, J.D., and T.W. Simpson (2004), On using Kriging models as probabilistic models in design. *SAE Transactions Journal of Materials & Manufacturing*, 5, pp. 129–139 [142]

[248] Martin, J.D., and T.W. Simpson (2005), Use of Kriging models to approximate deterministic computer models. *AIAA Journal*, 43, no. 4, pp. 853–863 [140, 142, 146, 148, 152]

[249] Martin, J.D. and T.W. Simpson (2007), A methodology to manage uncertainty during conceptual designs. *ASME Journal of Mechanical Design*, Special Issue on “Risk based and robust design”, in press [125, 126]

[250] Martin, M.A. (2007), Bootstrap hypothesis testing for some common statistical problems: a critical evaluation of size and power properties. *Computational Statistics & Data Analysis*, in press [84, 86]

[251] Matheron, G. (1963), Principles of geostatistics. *Economic Geology*, 58, no. 8, pp. 1246–1266 [140]

[252] McCullagh, P. and J.A. Nelder (1989), *Generalized linear models; second edition.* Chapman and Hall, London [8]

[253] McKay, M.D., R.J. Beckman, and W.J. Conover (1979), A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21, no. 2, pp. 239–245 (reprinted in *Technometrics*, 42, no. 1, 2000, pp. 55–61) [126]

[254] Meckesheimer, M., R.R. Barton, T.W. Simpson, and A.J. Booker (2001), Computationally inexpensive metamodel assessment strategies. *AIAA Journal*, 40. no. 10, pp. 2053–2060 [3, 123]

[255] Mee, R.W. (2004), Efficient two-level designs for estimating all main effects and two-factor interactions. *Journal Quality Technology*, 36, pp. 400–412 [48]

[256] Melas, V.B. (2006), *Functional approach to optimal experimental design* Springer, Berlin [52]

[257] Merkuryeva, G. (2005), Response surface-based simulation metamodelling methods with applications to optimisation problems. In: *Supply chain optimisation product / process design, facility location and flow control*, edited by A. Dolgui, J. Soldek, and O. Zaikin, Springer, p.205 –215 [105]

[258] Miller, A.J. (1990), *Subset selection in regression.* Chapman and Hall, London [61]

[259] Miller, G.A. (1956), The magical number seven plus or minus two: some limits on our capacity for processing information. *The Psychological Review*, 63, pp. 81–97 [157]

[260] Miller, R.G. (1974), The jackknife—a review. *Biometrika*, 61, pp. 1–15 [81]

[261] Morris, M.D. (2000), A class of three-level experimental designs for response surface modelling. *Technometrics*, 42, pp. 111–121 [51]

[262] Morris, M.D. (2006), An overview of group factor screening. In: *Screening: Methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 191–206 [159]

[263] Morris, M.D., L.M. Moore, and M.D. McKay (2006), Sampling plans based on balanced incomplete block designs for evaluating the importance of computer model inputs. *Journal of Statistical Planning and Inference*, 136, no. 9, pp. 3203–3220 [41]

[264] Mourani, I., S. Hennequin, and X. Xie (2007), Simulation-based optimization of a single-stage failure-prone manufacturing system with transportation delay. *International Journal of Production Economics*, in press [103]

[265] Mukerjee, R. and C.F.J. Wu (2006), *A modern theory of factorial designs.* Springer, New York [36, 52, 119, 130]

[266] Mula, J., R. Poler, J.P. García-Sabater, and F.C. Lario (2006), Models for production planning under uncertainty : A review. *International Journal of Production Economics*, 103, no. 1, pp. 271–285 [131]

[267] Myers, R.H., A.I. Khuri, and W.H. Carter (1989), Response surface methodology: 1966-1988. *Technometrics*, 31, no. 2, pp. 137–157 [105, 107]

[268] Myers, R.H. and D.C. Montgomery (2002), *Response surface methodology: process and product optimization using designed experiments; second edition.* Wiley, New York [9, 17, 26, 39, 51, 105, 107, 108]

[269] Narula, S.C. and J.F. Wellington (2007), Multiple criteria linear regression. *European Journal of Operational Research*, in press [20, 78]

[270] Naylor, T.H., J.L. Balintfy, D.S. Burdick, and K. Chu (1966), *Computer simulation techniques.* Wiley, New York [144]

[271] Nelson, B.L. (2004), Stochastic simulation research in *Management Science. Management Science*, 50, no. 7, pp. 855–868 [6, 9]

[272] Ng, S.H. and S.E. Chick (2006), Reducing parameter uncertainty for stochastic systems. *ACM Transactions on Modeling and Computer Simulation*, pp. 1–24 [110, 126]

[273] Ng, S.H. K. Xu, and W.K. Wong (2006), Optimization of multiple response surfaces with secondary constraints. Working Paper, National University of Singapore [110, 111]

[274] Nicolai, R. and R. Dekker (2005), Automated Response Surface Methodology for stochastic optimization models with unknown variance. Working Paper, Erasmus University, Rotterdam, Netherlands [109, 114]

[275] NIST/SEMATECH (2006), *e-Handbook of statistical methods* http://www.itl.nist.gov/div898/handbook/ [51]

[276] Noguera, J.H. and E.F. Watson (2006), Response surface analysis of a multi-product batch processing facility using a simulation metamodel. *International Journal of Production Economics*, 102, no. 2, pp. 333–343 [80, 105]

[277] Novikov, I. and B. Oberman (2007), Optimization of large simulations using statistical software. *Computational Statistics & Data Analysis*, 51, no. 5, pp. 2747–2752 [81]

[278] Oakley, J. (2004), Estimating percentiles of uncertain computer code outputs. *Applied Statistics*, 53, part 1, pp. 83–93 [126, 155]

[279] Oakley, J. and A. O'Hagan (2004), Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal Royal Statistical Society, Series B*, 66, part 3, pp. 751–769 [41, 125, 126]

[280] Oden, J.T. (2006), *Revolutionizing engineering science through simulation.* National Science Foundation (NSF), Blue Ribbon Panel on Simulation-Based Engineering Science [3, 7, 101, 123]

[281] Ólafsson, S. (2006), Metaheuristics. *Handbooks in Operations Research and Management Science, Volume 13*, Elsevier/North Holland, pp. 633–654 [103]

[282] Olhager, J., J.F. Persson, B. Parborg, and S. Rosén (2002), Supply chain impacts at Ericsson: from production units to demand-driven supply units. *International Journal of Technology Management*, 23, nos. 1/2/3, pp. 40–59 [135]

[283] Olive, D.J. (2006), Prediction intervals for regression models. *Computational Statistics & Data Analysis*, in press [62, 87]

[284] Olivi, L. (1984), editor, *Response surface methodology; handbook for nuclear reactor safety.* EUR 9600, Commission of the European Communities, Joint Research Centre, Ispra, Italy [9, 104]

[285] Oon, S.J. and L.H. Leehe (2006), The impact of ordinal on Response Surface Methodology. *Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 406–413 [102]

[286] Osborne, D.M., R.L. Armacost, and J. Pet-Edwards (1997), State of the art in multiple Response Surface Methodology. *Proceedings of the IEEE/SMC Conference*, Orlando, Florida [110]

[287] Park, D. and J.J. Faraway (1998), Sequential design for response curve estimation. *Journal for Nonparametric Statistics*, 9, pp. 155–164 [155]

[288] Park, D.S., Y.B. Kim, K.I. Shin, and T.R. Willemain (2001), Simulation output analysis using the threshold bootstrap. *European Journal of Operational Research*, 134, pp. 17–28 [84]

[289] Park, S., J.W. Fowler, G.T. Mackulak, J.B. Keats, and W.M. Carlyle (2002), D-optimal sequential experiments for generating a simulation-based cycle time-throughput curve. *Operations Research*, 50, no. 6, pp. 981–990 [149]

[290] Pearson, E.S. and H.O. Hartley (1970), *Biometrika tables for statisticians, volume I, third edition*, Cambridge University Press, London [89]

[291] Perez-Salvador, B.R. and F.J. O'Reilly (2005), Some modifications in response surface methodology, Working Paper, IIMAS, UNAM, Mexico City [109]

[292] Persson, J.F. and J. Olhager (2002), Performance simulation of supply chain designs. *International Journal of Production Economics*, 77, pp. 231–245 [135, 158]

[293] Pidd, M. (2004), *Computer simulation in management science, fifth edition.* Wiley, Chichester, England [6]

[294] Pitchitlamken, J. and B.L. Nelson (2003), A combined procedure for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation*, 13, no. 2, pp. 155–179 [103, 111]

[295] Pichitlamken, J., B.L. Nelson, and L.J. Hong (2006), A sequential procedure for neighborhood selection-of-the-best in optimization via simulation. *European Journal of Operational Research*, 173, no. 1, pp. 283–298 [102]

[296] Plackett, R.L. and J.P. Burman (1946), The design of optimum multifactorial experiments. *Biometrika*, 33, pp. 305–325 [36]

[297] Porta Nova, A.M. and J.R. Wilson (1989), Estimation of multiresponse simulation metamodels using control variates. *Management Science,* 35, no. 11, pp. 1316–1333 [119]

[298] Post, J., G. Klaseboer, E.D. Stinstra, and J. Huétink (2004), A DACE study on a three stage metal forming process made of Sandvik Nonoflex. *Proceedings of Numiform 2004*, Ohio, pp. 475–480 [123]

[299] Powell, S.G. (1997), Leading the spreadsheet revolution. *OR/MS Today*, pp. 8–10 [3]

[300] Pronzato, L. (2006), On the sequential construction of optimum bounded designs. *Journal of Statistical Planning and Inference*, 136, no. 8, pp. 2783–2804 [155]

[301] Psaradakis, Z. (2006), Blockwise bootstrap testing for stationarity. *Statistics & Probability Letters*, 76, no. 6, pp. 562–570 [84]

[302] Pukelsheim, F. (1993), *Optimal design of experiments.* Wiley, New York [52]

[303] Qu, X. (2007), Statistical properties of Rechtschaffner designs. *Journal of Statistical Planning and Inference*, in press

[304] Racine, J.S. and James G. MacKinnon (2007), Inference via kernel smoothing of bootstrap values P values. *Computational Statistics & Data Analysis*, in press [86]

[305] Rafajlowicz, E. and R. Schwabe (2006), Halton and Hammersley sequences in multivariate nonparametric regression. *Statistics & Probability Letters*, 76, pp. 803–812 [130]

[306] Rajagopalan, H.K., F.E. Vergara, C. Saydam, and J. Xiao (2007), Developing effective meta-heuristics for a probabilistic location model via experimental design. *European Journal of Operational Research*, 177, no. 1, pp. 83–101 [66, 103, 104]

[307] Rajagopal, R. and E. del Castillo (2005), Model-robust process optimization using Bayesian model averaging. *Technometrics*, 47, no. 2, pp. 152–163 [126, 131]

[308] Rajagopal, R., E. del Castillo, and J.J. Peterson (2005), Model and distribution-robust process optimization with noise factors. *Journal of Quality Technology*, 37, no. 3, pp. 210–222 (corrigenda: 2006, 38, no. 1, pp. 83) [131]

[309] Rao, C.R. (1959), Some problems involving linear hypothesis in multivariate analysis. *Biometrika*, 46, pp. 49–58 [97]

[310] Rao, C.R. (1967), Least squares theory using an estimated dispersion matrix and its application to measurement of signals. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability,* I, pp. 355–372 [77]

[311] Rechtschaffner, R.L. (1967), Saturated fractions of 2^n and 3^n factorial designs. *Technometrics*, 9, pp. 569–575 [48]

[312] Regis, R.G. and C.A. Shoemaker (2005), Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31, no. 1, pp. 153–171 [8, 104, 154]

[313] Ridlehoover, J. (2004), Applying Monte Carlo simulation and risk analysis to the facility location problem. *The Engineering Economist*, 49, no. 3, pp. 237–252 [3, 123]

[314] Rikards, R. and J. Auzins (2002), Response surface method for solution of structural identification problems. *Fourth International Conference on Inverse Problems in Engineering*, Rio de Janeiro, Brazil [9]

[315] Robinson, S. (2005), Discrete-event simulation: from the pioneers to the present, what next? *Journal Operational Research Society*, 56, no. 6, pp. 619–629 [6, 101]

[316] Rosen, S.C., C.M. Harmonosky, and M.T. Traband (2007), A simulation optimization method that considers uncertainty and multiple performance measures. *European Journal of Operational Research*, in press [105, 110]

[317] Roy, S.N., R. Gnanadesikan, and J.N. Srivastava (1971), *Analysis and design of certain quantitative multiresponse experiments.* Pergamon Press, Oxford [119]

[318] Rubinstein, R.Y. and D.P. Kroese (2004), *The cross-entropy method; a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning.* Springer, New York [103]

[319] Rubinstein, R.Y. and A. Shapiro (1993), *Discrete-event systems: sensitivity analysis and stochastic optimization via the score function method.* Wiley, New York [17, 103]

[320] Ruggoo, A. and M. Vandebroek (2007), Model-sensitive sequential optimal designs. *Computational Statistics & Data Analysis*, in press [52]

[321] Ruud, P. A. (2000), *An introduction to classical econometric theory.* Oxford University Press, New York [77]

[322] Sacks, J., W.J. Welch, T.J. Mitchell, and H.P. Wynn (1989), Design and analysis of computer experiments (includes Comments and Rejoinder). *Statistical Science*, 4, no. 4, pp. 409–435 [140, 142]

[323] Safizadeh, M.H. and R. Signorile (1994), Optimization of simulation via quasi-Newton methods. *ORSA Journal on Computing*, 6, no. 4, pp. 398–408 [108]

[324] Sakalauskas, L. and J. Krarup (2006), Editorial; heuristic and stochastic methods in optimization. *European Journal of Operational Research,* 171, no. 3, pp. 723–724 [103]

[325] Salibian-Barrera, M. (2007), Bootstrapping MM-estimators for linear regression with fixed designs. *Statistics & Probability Letters*, in press [81]

[326] Sallaberry, C.J., J.C. Helton, and S.C. Hoara (2006), Extension of Latin hypercube samples with correlated variables. Working Paper, Sandia National Laboratories, Albuquerque, New Mexico [126, 129]

[327] Saltelli, M. Ratto, S. Tarantola, and F. Campolongo (2005), Sensitivity analysis of chemical models. *Chemical Reviews*, 105, no. 7, pp. 2811–2827 [3, 9, 39, 41, 98, 107, 123, 159, 169]

[328] Saltelli, A. and I.M. Sobol (1995), About the use of rank transformation in sensitivity analysis of model output. *Reliability Engineering and System Safety*, 50, pp. 225–239 [98]

[329] Saltelli, A., S. Tarantola, F. Campolongo, and M. Ratto (2004), *Sensitivity analysis in practice; a guide to assessing scientific models.* Wiley, Chichester, England [41, 127, 159]

[330] Sanchez, S.M. (2000) Robust design: seeking the best of all possible worlds, *Proceedings of the 2000 Winter Simulation Conference*, edited by J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, pp. 69–76 [131]

[331] Sanchez, S.M., F. Moeeni, and P.J. Sanchez (2006), So many factors, so little time...simulation experiments in the frequency domain. *International Journal of Production Economics*, 103, no. 1, pp. 149–165 [53]

[332] Sanchez, S.M. and P.J. Sanchez (2005), Very large fractional factorial and central composite designs. *ACM Transactions of Modeling and Computer Simulation*, 15, no. 4, pp. 362–377 [35, 39, 47, 48, 50]

[333] Santner, T.J., B.J. Williams, and W.I. Notz (2003), *The design and analysis of computer experiments.* Springer-Verlag, New York [9, 17, 41, 130, 140, 147]

[334] Santos, M.I. and A.M.O. Porta Nova (2006), Statistical fitting and validation of nonlinear simulation metamodels: a case study. *European Journal of Operational Research*, 171, no. 1, pp. 53–63 [8, 61]

[335] Santos, M.I. and A.M.O. Porta Nova (2007), Using simulation to estimate and validate nonlinear regression metamodels. *Communications in Statistics-Simulation and Computation*, in press [61]

[336] Sasena, M.J, P. Papalambros, and P. Goovaerts (2002), Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization,* 34, no.3, pp. 263–278 [154, 155]

[337] Savage, S. (2006), Interactive simulation. *OR/MS Today*, 33, no. 2, pp. 62–63 [126]

[338] Scheffé, H. (1964), *The analysis of variance; fourth printing.* Wiley, New York [88]

[339] Schikora, P.F. and M.R. Godfrey (2003), Efficacy of end-user neural network and data mining software for predicting complex system performance. *International Journal of Production Economics*, 84, pp. 231–253 [8, 65]

[340] Scholtes, S. (2004), Insight 2.0. *OR/MS Today*, 31, no. 1, pp. 42–45 [3, 123]

[341] Schonlau, M. and W.J. Welch (2006), Screening the input factors to a computer code via Analysis of Variance and visualization. In: *Screening: Methods for experimentation in industry, drug discovery, and genetics*, edited by A. Dean and S. Lewis, Springer-Verlag, New York, pp. 308–327 [41, 159]

[342] Schruben, L.W. and V.J. Cogliano (1987), An experimental procedure for simulation response surface model identification. *Communications ACM*, 30, no. 8, pp. 716–730 [53]

[343] Schruben, L.W. and B.H. Margolin (1978), Pseudorandom number assignment in statistically designed simulation and distribution sampling experiments. *Journal American Statistical Association*, 73, no. 363, pp. 504–525 [53, 96]

[344] Schwartz, J.D., W. Wang, and D.E. Rivera (2006), Simulation-based optimization of process control policies for inventory management in supply chains. *Automatica*, 42, no. 8, pp. 1311–1320 [103]

[345] Searle, S.R. (1971), *Linear models.* Wiley, New York [23]

[346] Ševčíková, H., A.E. Raftery, and P.A. Waddell (2006), Assessing uncertainty in urban simulations using Bayesian melding *Transportation Research Part B: Methodological, Transportation Research B*, 41, 652–669 [126]

[347] Shang, J.S., S. Li, and P. Tadikamalla (2004), Operational design of a supply chain system using the Taguchi method, response surface methodology, simulation, and optimization. *International Journal of Production Research*, 42, no. 18, pp. 3823–3849 [131, 134]

[348] Shao, J. and D. Tu (1995), *The jackknife and bootstrap.* Springer, New York [86]

[349] Shen, H. and H. Wan (2006), A hybrid method for simulation factor screening. *Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 382–389 [160]

[350] Shi, L. and S. Olafsson (2000), Nested partitions method for global optimization. *Operations Research*, 48, no. 3, pp. 390–407 [103]

[351] Shubik, M. (2002), Game theory and Operations Research: some musings 50 years later. *Operations Research*, 50, no. 1, pp. 192–196 [7]

[352] Siem, A.Y.D., E. de Klerk, and D. den Hertog (2007), Discrete least-norm approximation by nonnegative (trigonometric) polynomials and rational functions, *Structural and Multidisciplinary Optimization*, in press [162]

[353] Siem, A.Y.D. and D. den Hertog (2007), Kriging models that are robust with respect to simulation errors. Working Paper, Tilburg University, Tilburg, Netherlands [147]

[354] Simchi-Levi, D., P. Kaminsky, and E. Simchi-Levi (2003), *Designing and managing the supply chain: concepts, strategies, and case studies;second edition*. Irwin/McGraw-Hill, Boston [7]

[355] Simpson, T.W., A.J. Booker, D. Ghosh, A.A. Giunta, P.N. Koch, and R.-J. Yang (2004), Approximation methods in multidisciplinary analysis and optimization: a panel discussion. *Structural and Multidisciplinary Optimization*, 27, no. 5, pp. 302–313 [8, 52, 54, 59, 63, 98, 104, 130, 154, 157]

[356] Simpson, T.W., T.M. Mauery, J.J. Korte, and F. Mistree (2001), Kriging metamodels for global approximation in simulation-based multidisciplinary design optimization. *AIAA Journal*, 39, no. 12, pp. 2233–2241 [140]

[357] Simpson, T.W., J. Peplinski, P.N. Koch, and J.K. Allen (2001), Metamodels for computer-based engineering design: survey and recommendation. *Engineering with Computers*, 17, no. 2, pp. 129–150 [3, 8, 123]

[358] Sobol', I.M. (1990), Sensitivity estimates for non-linear mathematical models. *Matematicheskoe Modelirovanie*, 2, pp. 112–118 [41]

[359] Spall, J.C. (2000), Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45, no. 10, pp. 1839–1853 [103]

[360] Spall, J.C. (2003), *Introduction to stochastic search and optimization; estimation, simulation, and control*. Wiley, New York [17, 103, 118]

[361] Spall, J.C., S.D. Hill, and D.R. Stark (2006), Theoretical framework for comparing several stochastic optimization approaches. In *Probabilistic and randomized methods for design under uncertainty*, edited by G. Calafiori and F. Dabbene, Springer, London [104]

[362] Srivastava, J.N. (1975), Designs for searching nonnegligible effects. *A survey of statistical design and linear models*, edited by J.N. Srivastava, North-Holland, Amsterdam [52]

[363] Stein, M.L. (1999), *Interpolation of spatial data: some theory for Kriging*. Springer, New York [140]

[364] Sterman, J.D. (2000), *Business dynamics: systems thinking and modeling for a complex world*. McGraw-Hill, Homewood, Illinois [3]

[365] Stinstra, E.D. (2006), *The meta-model approach for simulation-based design optimization*. Doctoral dissertation, Tilburg University, Tilburg, Netherlands [8, 111, 123, 130, 131]

[366] Stinstra, E. and D. den Hertog (2007), Robust optimization using computer experiments. *European Journal of Operational Research*, in press [131]

[367] Stinstra, E.D., D. den Hertog, H.P. Stehouwer, and A.P.A. Vestjens (2003), Constrained maximin designs for computer experiments. *Technometrics*, 45, no. 4, pp. 340–346 [53]

[368] Stinstra, E.D., H.P. Stehouwer, and J. van der Heijden (2003), Collaborative tube design optimization: an integral meta-modeling approach. *Proceedings of the 5th ISSMO Conference on Engineering Design Optimization* [111]

[369] Stone, M. (1974), Cross-validatory choice and assessment of statistical predictions. *Journal Royal Statistical Society, Series B*, 36, no. 2, pp. 111–147 [59]

[370] Storlie, C.B. and J.C. Helton (2006), Multiple predictor smoothing methods for sensitivity analysis. Working Paper, Sandia National Laboratories, Albuquerque, New Mexico [8]

[371] Suman, B. and P. Kumar (2006), A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society*, 57, pp. 1143–1160 [103]

[372] Sun, Y. and A.C.M. Wong (2007), Interval estimation for the normal correlation coefficient. *Statistics & Probability Letters*, in press [56]

[373] Swain, J.J. (2005), "Gaming" reality. *OR/MS Today*, 32, no. 6, pp. 44–55 [5, 7]

[374] Swisher, J.R., S.H. Jacobson, and E. Yücesan (2003), Discrete-event simulation optimization using ranking, selection, and multiple comparison procedures: a survey. *ACM Transactions on Modeling and Computer Simulation*, 13. no. 2, pp. 134–154 [102]

[375] Sztendur, E. (2005), *Precision of the path of steepest ascent in response surface methodology.* Doctoral dissertation, Victoria University of Technology, School of Computer Science and Mathematics, Melbourne, Australia [107]

[376] Taguchi, G. (1987), *System of experimental designs, volumes 1 and 2.* UNIPUB/Krauss International, White Plains, New York [130]

[377] Tekin, E. and I. Sabuncuoglu (2004), Simulation optimization: a comprehensive review on theory and applications. *IIE Transactions*, 36, pp. 1067–1081 [17, 102, 103, 105]

[378] Tian, Y. and D.P. Wiens (2007), On equality and proportionality of ordinary least squares, weighted least squares and best linear unbiased estimators in the general linear model. *Statistics & Probability Letters*, in press [21]

[379] Toropov, V.V., U. Schramm, A. Sahai, R. Jones, and T. Zeguer (2005), Design optimization and stochastic analysis based on the Moving Least Squares method. *6th World Congress of Structural and Multidisciplinary Optimization*, Rio de Janeiro, paper no. 9412 [146, 147]

[380] Tsai, C.S (2002), Evaluation and optimisation of integrated manufacturing system operations using Taguchi's experiment design in computer simulation. *Computers & Industrial Engineering*, 43, no. 3, pp. 591–604 [131]

[381] Tsai, P-W., S.G. Gilmour and R. Mead (2007), Three-level main-effects designs exploiting prior information about model uncertainty. *Journal of Statistical Planning and Inference,* in press [130, 169]

[382] Tu, J. and D.R. Jones (2003), Variable screening in metamodel design by cross-validated Moving Least Squares method, *Proceedings 44th AIAA/ASME/ASCE/AHS/ASC Structures,* Structural Dynamics and Materials Conference, AIAA-2003-1669, Norfolk, Virginia, April 7-10, 2003 [59, 63, 146, 159]

[383] Van Beers, W. and J.P.C. Kleijnen (2003), Kriging for interpolation in random simulation. *Journal of the Operational Research Society*, no. 54, pp. 255–262 [140, 147]

[384] Van Beers, W.C.M. and J.P.C. Kleijnen (2006), Customized sequential designs for random simulation experiments: Kriging metamodeling and bootstrapping. Working Paper, Tilburg University, Tilburg, Netherlands [96, 148, 149, 151]

[385] Van Berkum, E. (2005), Proefopzetten, *STAtOR*, 6, no. 4, pp. 11–16. [127]

[386] Van Dam, E.R., B.G.M. Husslage, D. den Hertog, and J.B.M. Melissen (2007), Maximin Latin hypercube designs in two dimensions, *Operations Research*, 55, 158–169 [130]

[387] Van Groenendaal, W.J.H. (1998), *The economic appraisal of natural gas projects*. Oxford University Press, Oxford [60]

[388] Van Groenendaal, W.J.H. and J.P.C. Kleijnen (2002), Deterministic versus stochastic sensitivity analysis in investment problems: an environmental case study. *European Journal of Operational Research*, 141, no. 1, pp. 8–20 [125]

[389] Van Schaik, F.D.J. and J.P.C. Kleijnen (2004), Sealed-bid auctions: case study. Working Paper, Faculty of Economics and Business Administration, Tilburg University [19]

[390] Vladislavleva, E. and G. Smits (2007), Order of nonlinearity as a complexity measure for models generated with symbolic regression via genetic programming, Working Paper, Tilburg University, Tilburg, Netherlands [8]

[391] Vonk Noordegraaf, A. (2002), *Simulation modelling to support national policy making in the control of bovine herpes virus*. Doctoral dissertation, Wageningen University, Wageningen, The Netherlands [60]

[392] Vose, D. (2000), *Risk analysis; a quantitative guide; second edition.* Wiley, Chichester, United Kingdom [3, 123]

[393] Wan, H., B.E. Ankenman, and B.L. Nelson (2006), Controlled sequential bifurcation: a new factor- screening method for discrete-event simulation. *Operations Research*, 54, no. 4, pp. 743–755 [160, 161, 166, 169]

[394] Wan, H., B.E. Ankenman, and B.L. Nelson (2006), Simulation factor screening with controlled sequential bifurcation in the presence of interactions, Working Paper, Purdue University, West Lafayette, Indiana [160, 165, 166, 169]

[395] Wan, J. and J.P.C. Kleijnen (2006), Simulation for the optimization of (s, S) inventory system with random lead times and a service level constraint by using Arena and OptQuest. Working Paper, Hebei University of Technology [104]

[396] Wang, Y., D.K.J. Lin, and K.T. Fang (1995), Designing outer array points. *Journal of Quality Technology*, 27, no. 3, pp. 226–241 [136]

[397] Webb, S. (1968), Non-orthogonal designs of even resolution. *Technometrics*, 10, pp. 291–299 [45]

[398] Weisstein, E.W. (2006), Hadamard matrix, *MathWorld–A Wolfram Web Resource.*
http://mathworld.wolfram.com/HadamardMatrix.html [35]

[399] Wen, M-J., S-Y. Chen, and H.J. Chen (2006), On testing a subset of regression parameters under heteroskedasticity. *Computational Statistics & Data Analysis,* in press [90]

[400] Wieland, J.R. and B.W. Schmeiser (2006), Stochastic gradient estimator using a single design point. *Proceedings of the 2006 Winter Simulation Conference*, edited by L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, pp. 390–397 [103, 118]

[401] Williams, B.J., T.J. Santner, and W.I. Notz (2000), Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica*, 10, pp. 1133–1152 [155]

[402] Wu, C.F.J. and M. Hamada (2000), *Experiments; planning, analysis, and parameter design optimization.* Wiley, New York [52, 77, 130, 159, 169]

[403] Yamada,S., M. Matsui, T. Matsui, D.K.J. Lin, and T. Takahashi (2006), A general construction method for mixed-level supersaturated design. *Computational Statistics & Data Analysis*, 50, no. 1, pp. 254–265 [159]

[404] Yang, F., B. Ankenman, and B. Nelson (2006), Efficient generation of cycle time-throughput curves through simulation and metamodeling. *Naval Research Logistics*, accepted [50, 88, 152]

[405] Yang, G. and N.A. Butler (2006), Nonregular two-level designs of resolution IV or more containing clear two-factor interactions. *Statistics & Probability Letters, i*n press [39, 52]

[406] Yang, T., H-P. Fu, and K-Y. Yang (2007), An evolutionary-simulation approach for the optimization of multi-constant work-in-process strategy—a case study. *International Journal of Production Economics,* 107, pp. 104–114 [103]

[407] Yang, T. and L. Tseng (2002), Solving a multi-objective simulation model using a hybrid response surface method and lexicographical goal programming approach: a case study on integrated circuit ink-marking machines. *Journal of the Operational Research Society*, 53, no. 2, pp. 211–221 [105]

[408] Yeh, K., W. Li, and A. Sudjianto (2000), Algorithmic construction of optimal symmetric Latin Hypercube designs. *Journal of Statistical Planning and Inference*, 90, pp. 145–159 [130]

[409] Yeomans, J.S. (2007), Solid waste planning under uncertainty using evolutionary simulation-optimization. *Socio-Economic Planning Sciences*, 41, pp. 38–60 [132]

[410] You, J. and G. Chen (2006), Wild bootstrap estimation in partially linear models with heteroscedasticity. *Statistics & Probability Letters*, 76, no. 4, pp. 340–348 [91]

[411] Yu, H-F. (2007), Designing a screening experiment with a reciprocal Weibull degradation rate. *Computers & Industrial Engineering*, in press [159]

[412] Zazanis, M.A. and R. Suri (1993), Convergence rates of finite-difference sensitivity estimates for stochastic systems. *Operations Research*, 41, no. 4, pp. 694–703 [107, 118]

[413] Zeigler B.P., H. Praehofer, and T.G. Kim (2000), *Theory of modeling and simulation; second edition.* Academic Press, San Diego [4, 27]

[414] Zhang, Q., M.A. Vonderembse, and J.S. Lim (2003), Manufacturing flexibility: defining and analyzing relationships among competence, capability, and customer satisfaction. *Journal of Operations Management*, 21, pp. 173–191 [135]

[415] Zhang, Q-Z., R-C. Zhang, and M-Q. Liu (2006), A method for screening active effects in supersaturated designs. *Journal of Statistical Planning and Inference,* in press [159]

[416] Zouaoui, F. and J.R. Wilson (2004), Accounting for input-model and input-parameter uncertainties in simulation. *IIE Transactions*, 36, pp. 1135–1151 [124, 126]

Index

Early Titles in the
INTERNATIONAL SERIES IN OPERATIONS RESEARCH & MANAGEMENT SCIENCE
Frederick S. Hillier, Series Editor, *Stanford University*

Saigal/ *A MODERN APPROACH TO LINEAR PROGRAMMING*
Nagurney/ *PROJECTED DYNAMICAL SYSTEMS & VARIATIONAL INEQUALITIES WITH APPLICATIONS*
Padberg & Rijal/ *LOCATION, SCHEDULING, DESIGN AND INTEGER PROGRAMMING*
Vanderbei/ *LINEAR PROGRAMMING*
Jaiswal/ *MILITARY OPERATIONS RESEARCH*
Gal & Greenberg/ *ADVANCES IN SENSITIVITY ANALYSIS & PARAMETRIC PROGRAMMING*
Prabhu/ *FOUNDATIONS OF QUEUEING THEORY*
Fang, Rajasekera & Tsao/ *ENTROPY OPTIMIZATION & MATHEMATICAL PROGRAMMING*
Yu/ *OR IN THE AIRLINE INDUSTRY*
Ho & Tang/ *PRODUCT VARIETY MANAGEMENT*
El-Taha & Stidham/ *SAMPLE-PATH ANALYSIS OF QUEUEING SYSTEMS*
Miettinen/ *NONLINEAR MULTIOBJECTIVE OPTIMIZATION*
Chao & Huntington/ *DESIGNING COMPETITIVE ELECTRICITY MARKETS*
Weglarz/ *PROJECT SCHEDULING: RECENT TRENDS & RESULTS*
Sahin & Polatoglu/ *QUALITY, WARRANTY AND PREVENTIVE MAINTENANCE*
Tavares/ *ADVANCES MODELS FOR PROJECT MANAGEMENT*
Tayur, Ganeshan & Magazine/ *QUANTITATIVE MODELS FOR SUPPLY CHAIN MANAGEMENT*
Weyant, J./ *ENERGY AND ENVIRONMENTAL POLICY MODELING*
Shanthikumar, J.G. & Sumita, U./ *APPLIED PROBABILITY AND STOCHASTIC PROCESSES*
Liu, B. & Esogbue, A.O./ *DECISION CRITERIA AND OPTIMAL INVENTORY PROCESSES*
Gal, T., Stewart, T.J., Hanne, T. / *MULTICRITERIA DECISION MAKING: Advances in MCDM Models, Algorithms, Theory, and Applications*
Fox, B.L. / *STRATEGIES FOR QUASI-MONTE CARLO*
Hall, R.W. / *HANDBOOK OF TRANSPORTATION SCIENCE*
Grassman, W.K. / *COMPUTATIONAL PROBABILITY*
Pomerol, J-C. & Barba-Romero, S. */ MULTICRITERION DECISION IN MANAGEMENT*
Axsäter, S. */ INVENTORY CONTROL*
Wolkowicz, H., Saigal, R., & Vandenberghe, L. / *HANDBOOK OF SEMI-DEFINITE PROGRAMMING: Theory, Algorithms, and Applications*
Hobbs, B.F. & Meier, P. / *ENERGY DECISIONS AND THE ENVIRONMENT: A Guide to the Use of Multicriteria Methods*
Dar-El, E. / *HUMAN LEARNING: From Learning Curves to Learning Organizations*
Armstrong, J.S. / *PRINCIPLES OF FORECASTING: A Handbook for Researchers and Practitioners*
Balsamo, S., Personé, V., & Onvural, R./ *ANALYSIS OF QUEUEING NETWORKS WITH BLOCKING*
Bouyssou, D. et al. / *EVALUATION AND DECISION MODELS: A Critical Perspective*
Hanne, T. / *INTELLIGENT STRATEGIES FOR META MULTIPLE CRITERIA DECISION MAKING*
Saaty, T. & Vargas, L. / *MODELS, METHODS, CONCEPTS and APPLICATIONS OF THE ANALYTIC HIERARCHY PROCESS*
Chatterjee, K. & Samuelson, W. / *GAME THEORY AND BUSINESS APPLICATIONS*
Hobbs, B. et al. / *THE NEXT GENERATION OF ELECTRIC POWER UNIT COMMITMENT MODELS*
Vanderbei, R.J. / *LINEAR PROGRAMMING: Foundations and Extensions, 2nd Ed.*
Kimms, A. / *MATHEMATICAL PROGRAMMING AND FINANCIAL OBJECTIVES FOR SCHEDULING PROJECTS*
Baptiste, P., Le Pape, C. & Nuijten, W. / *CONSTRAINT-BASED SCHEDULING*
Feinberg, E. & Shwartz, A. / *HANDBOOK OF MARKOV DECISION PROCESSES*: *Methods and Applications*
Ramík, J. & Vlach, M. / *GENERALIZED CONCAVITY IN FUZZY OPTIMIZATION AND DECISION ANALYSIS*

Early Titles in the
INTERNATIONAL SERIES IN OPERATIONS RESEARCH & MANAGEMENT SCIENCE
(Continued)

Song, J. & Yao, D. / *SUPPLY CHAIN STRUCTURES: Coordination, Information and Optimization*
Kozan, E. & Ohuchi, A. / *OPERATIONS RESEARCH/ MANAGEMENT SCIENCE AT WORK*
Bouyssou et al. / *AIDING DECISIONS WITH MULTIPLE CRITERIA: Essays in Honor of Bernard Roy*
Cox, Louis Anthony, Jr. / *RISK ANALYSIS: Foundations, Models and Methods*
Dror, M., L'Ecuyer, P. & Szidarovszky, F. / *MODELING UNCERTAINTY: An Examination of Stochastic Theory, Methods, and Applications*
Dokuchaev, N. / *DYNAMIC PORTFOLIO STRATEGIES: Quantitative Methods and Empirical Rules for Incomplete Information*
Sarker, R., Mohammadian, M. & Yao, X. / *EVOLUTIONARY OPTIMIZATION*
Demeulemeester, R. & Herroelen, W. / *PROJECT SCHEDULING: A Research Handbook*
Gazis, D.C. / *TRAFFIC THEORY*
Zhu/ *QUANTITATIVE MODELS FOR PERFORMANCE EVALUATION AND BENCHMARKING*
Ehrgott & Gandibleux*/ MULTIPLE CRITERIA OPTIMIZATION: State of the Art Annotated Bibliographical Surveys*
Bienstock/ *Potential Function Methods for Approx. Solving Linear Programming Problems*
Matsatsinis & Siskos/ *INTELLIGENT SUPPORT SYSTEMS FOR MARKETING DECISIONS*
Alpern & Gal*/ THE THEORY OF SEARCH GAMES AND RENDEZVOUS*
Hall/*HANDBOOK OF TRANSPORTATION SCIENCE - 2nd Ed.*
Glover & Kochenberger*/ HANDBOOK OF METAHEURISTICS*
Graves & Ringuest/ *MODELS AND METHODS FOR PROJECT SELECTION: Concepts from Management Science, Finance and Information Technology*
Hassin & Haviv/ *TO QUEUE OR NOT TO QUEUE: Equilibrium Behavior in Queueing Systems*
Gershwin et al*/ ANALYSIS & MODELING OF MANUFACTURING SYSTEMS*
Maros/ *COMPUTATIONAL TECHNIQUES OF THE SIMPLEX METHOD*
Harrison, Lee & Neale/ *THE PRACTICE OF SUPPLY CHAIN MANAGEMENT: Where Theory and Application Converge*
Shanthikumar, Yao & Zijm/ *STOCHASTIC MODELING AND OPTIMIZATION OF MANUFACTURING SYSTEMS AND SUPPLY CHAINS*
Nabrzyski, Schopf & Węglarz/ *GRID RESOURCE MANAGEMENT: State of the Art and Future Trends*
Thissen & Herder/ *CRITICAL INFRASTRUCTURES: State of the Art in Research and Application*
Carlsson, Fedrizzi, & Fullér/ *FUZZY LOGIC IN MANAGEMENT*
Soyer, Mazzuchi & Singpurwalla/ *MATHEMATICAL RELIABILITY: An Expository Perspective*
Chakravarty & Eliashberg/ *MANAGING BUSINESS INTERFACES: Marketing, Engineering, and Manufacturing Perspectives*
Talluri & van Ryzin/ *THE THEORY AND PRACTICE OF REVENUE MANAGEMENT*
Kavadias & Loch/*PROJECT SELECTION UNDER UNCERTAINTY: Dynamically Allocating Resources to Maximize Value*
Brandeau, Sainfort & Pierskalla/ *OPERATIONS RESEARCH AND HEALTH CARE: A Handbook of Methods and Applications*
Cooper, Seiford & Zhu/ *HANDBOOK OF DATA ENVELOPMENT ANALYSIS: Models and Methods*
Luenberger/ *LINEAR AND NONLINEAR PROGRAMMING, 2nd Ed.*
Sherbrooke/ *OPTIMAL INVENTORY MODELING OF SYSTEMS: Multi-Echelon Techniques, Second Edition*
Chu, Leung, Hui & Cheung/ *4th PARTY CYBER LOGISTICS FOR AIR CARGO*

Early Titles in the
INTERNATIONAL SERIES IN OPERATIONS RESEARCH & MANAGEMENT SCIENCE
(Continued)

Simchi-Levi, Wu & Shen/ *HANDBOOK OF QUANTITATIVE SUPPLY CHAIN ANALYSIS: Modeling in the E-Business Era*

Gass & Assad/ *AN ANNOTATED TIMELINE OF OPERATIONS RESEARCH: An Informal History*

Greenberg/ *TUTORIALS ON EMERGING METHODOLOGIES AND APPLICATIONS IN OPERATIONS RESEARCH*

Weber/ *UNCERTAINTY IN THE ELECTRIC POWER INDUSTRY: Methods and Models for Decision Support*

Figueira, Greco & Ehrgott/ *MULTIPLE CRITERIA DECISION ANALYSIS: State of the Art Surveys*

Reveliotis/ *REAL-TIME MANAGEMENT OF RESOURCE ALLOCATIONS SYSTEMS: A Discrete Event Systems Approach*

Kall & Mayer/ *STOCHASTIC LINEAR PROGRAMMING: Models, Theory, and Computation*

***** A list of the more recent publications in the series is at the front of the book ****