MODELING AND OPTIMIZATION OF PROCESS ENGINEERING PROBLEMS CONTAINING BLACK-BOX SYSTEMS AND NOISE

by

EDGAR FRANKLIN DAVIS

A Dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Chemical and Biochemical Engineering

written under the direction of

Professor Marianthi G. Ierapetritou

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2008

ABSTRACT OF THE DISSERTATION

Modeling and Optimization of Process Engineering Problems
Containing Black-Box Systems and Noise

by EDGAR FRANLKIN DAVIS

Dissertation Director:
Dr. Marianthi G. Ierapetritou, Ph.D.

This thesis addresses the optimization of systems whose behavior is described by noisy input-output data instead of model equations. Process models may not exist, as in the case of emergent technologies, or may be inaccessible if they are embedded within a legacy computational code. When a functional form for the input-output relationships is unavailable, the process behavior is symbolically described using black-box models. Two cases motivate the need to address problems containing black-box models: 1) building a case for obtaining continued research funding during the early product life cycle, when the system information is limited to a sparse sampling set, and 2) process train optimization for systems that have been retrofitted or exhibit behavior which results in suboptimal performance. The challenge is to determine the best operating conditions which satisfy some objective, such as maximizing reaction yield or minimizing utilities costs, based on a limited amount of additional sampling that can be performed.

Surrogate data-driven models can be alternatively generated, but many substitute models may need to be built, especially in the case of process synthesis problems. Although model reliability can be improved using additional information, resource constraints can limit the number of additional experiments allowed. Since it may not be possible to *a priori* estimate the problem cost in terms of the number of experiments

required, there is a need for strategies targeted at the generation of sufficiently accurate surrogate models at low resource cost. The problem addressed in this work focuses on the development of model-based optimization algorithms targeted at obtaining the best solutions based on limited sampling. A centroid-based sampling algorithm for global modeling has also been developed to accelerate accurate global model generation and improve subsequent local optimization. The developed algorithms enable the superior local solutions of problems containing black-box models and noisy input-output data to be obtained when the problem contains both continuous and integer variables and is defined by an arbitrary convex feasible region. The proposed algorithms are applied to many numerical examples and industrial case studies to demonstrate the improved optima attained when surrogate models are built prior to optimization.

ACKNOWLEDGMENTS

I would like to acknowledge Professor Marianthi Ierapetritou for choosing me as a research collaborator, for your guidance, encouragement, and patience over the past five years – thank you always.

I would like to thank Dr. Alkis Constantinides and Dr. Henrik Pedersen for giving me several opportunities to instruct and assist them in undergraduate coursework. My lab friends:

I remain grateful for the funding provided by both the NIH and NSF.

Finally, I would like to thank my committee members: Dr. Ioannis P. Androulakis, Dr. Charlie Roth, and Dr. David Coit.

Rutgers Grad Students: Lauren Britton, Eric Yang, Meric Ovacik, Pegy Fontineu

Rutgers PSE Lab: Dr. Aditya Bindal, Dr. Dan Wu, Dr. Vishal Goyal, Dr. Ipsita Banerjee, Dr. Zhenya Jia, Dr. Nripen Sharma, Dr. Patricia Portillo, Dr. Georgios Saharidis, Dr. Vidya Iyer, Hong Yang, Steve Guzikowski, Zukui Li, Kai He, Beverly Smith, Yijie Gao, and Mehmet Orman.

CBE and SOE administrative staff: Debora Moon, Lynn DiCaprio, Ursula Wolf, Stephanie Thomas, Joyce Creteau, Sadhna Patel, Tannie Wan. A special thank you to all of you for being so generous and gracious with your time to help me out in all things big and small.

DEDICATION

In 1981, a baby was found in a Taipei phone booth, wrapped in newspaper. Remaining unclaimed, he was taken to an orphanage. The child was frail, having been born with a bilateral cleft lip and cleft palate. A survival prognosis was grim. Early death was likely.

In 1982, a picture of this boy was left in a U.S.A. mailbox by a stranger. Adoption soon followed, and on July 16, 1983, this child arrived at the Denver, Colorado airport. During the next fifteen years, he would be a surgical pediatrics patient at Texas Children's Hospital twenty-plus times. Always afraid each time he made the lonely walk to the OR, this child would later touch his new scars, wondering if he would ever be normal.

What, if any, contribution could a broken child ever give to the world? Many times, this child asked for a reason to believe. Many times, his mother prayed for a hope she could give. Her hope became a flame of encouragement for him, burning brightly, enduring.

His mother died on April 8, 2008. One of her last wishes was that he finish his Ph.D.. Her son, although inconsolable, stayed the course, persevered, and finished three months later. Sadly, he would never be able to send the following note to his beloved mother:

Dear Mom,

I finished my thesis. I'm almost a doctor!

Love, Eddie

Mom, this is for you.

To everybody else, named and unnamed, who have guided me along my life's journeys, all the struggles I endured as a child have led to a greater appreciation for each new day I am able to live. This work is also dedicated with love and gratitude to all those who gave me life, who gave my life meaning, who gave me a future. You sacrificed so much that I might have the opportunity to give back to the world. I hope to always make you proud.

Birthparents: Taken from you all too soon, I would never remember the faces of those who saw my first breath, my first smile, my first tears. You gave me up so I could live.

My brother, John: Though our mother is gone, we still have each other. She will always watch over us, and she would not wish for us to be afraid in the days ahead. It is now your turn to fulfill your dreams.

Grandparents Jessy F. Gray and Margery E. Walker, and all other relatives.

Godfamily: Jerry, Sandy, Melissa, and Alexander McMahen.

Craniofacial/ENT surgical team leads: Dr. Samuel Stal, Dr. Daniel Franklin, Dr. Newton Duncan.

Middlesex Endodontics: Drs. Avery Lafkowitz, Dr. Dimple Malavia, and all their wonderful staff.

TAMS: Dr. Shilpa Miniyar, Dr. Margaret Boyden, Dr. Sarah Hayter, Dr. Atisha Patel, Dr. Shalin Patel, Dr. Diana Dugas, Nathan Wozny, Laura Christian, Jennifer Shih, Matthew Frank, Diana Cheng, David Yin, and Courtney Davis.

With profound gratitude to these special friends and their families: John and Patricia Bentz, Gary, Kathy, Robby and Chase Anderson, James, Jeanie, and Cody Herrington, Margaret Cottrill, Jeffrey and Katy O'Neal, Tim and Lois Manier, Tom and Patricia Synatschk-Silva, Carlos, Jennifer, Morgan, and Rachel Morris, Albert, Sharon, and Rachael Liles, Brandon and Sarah Satrom, Todd, Debbie, Kaly, and Kyle Paulson, Russell and Rhonda Serpas, Carol Mendelovitz and all the wonderful staff at Faith Community Hospice, Jodi Picoult, Deirdre Shannon, Matthew Gilsenan, Dr. Robert Karlsberg, Dr. Steven Thompson, and Dr. Elaine Donnelly.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Decision-making is complicated at different product life cycle steps – funding allocation for research lines, process synthesis, and product optimization – when accurate model equations are lacking, and input-output data, possibly noisy, provide the only description of system behavior. Schematics describing the sampling and funding decision-making stages are shown in Figures 1.1(a) and 1.1(b), respectively.

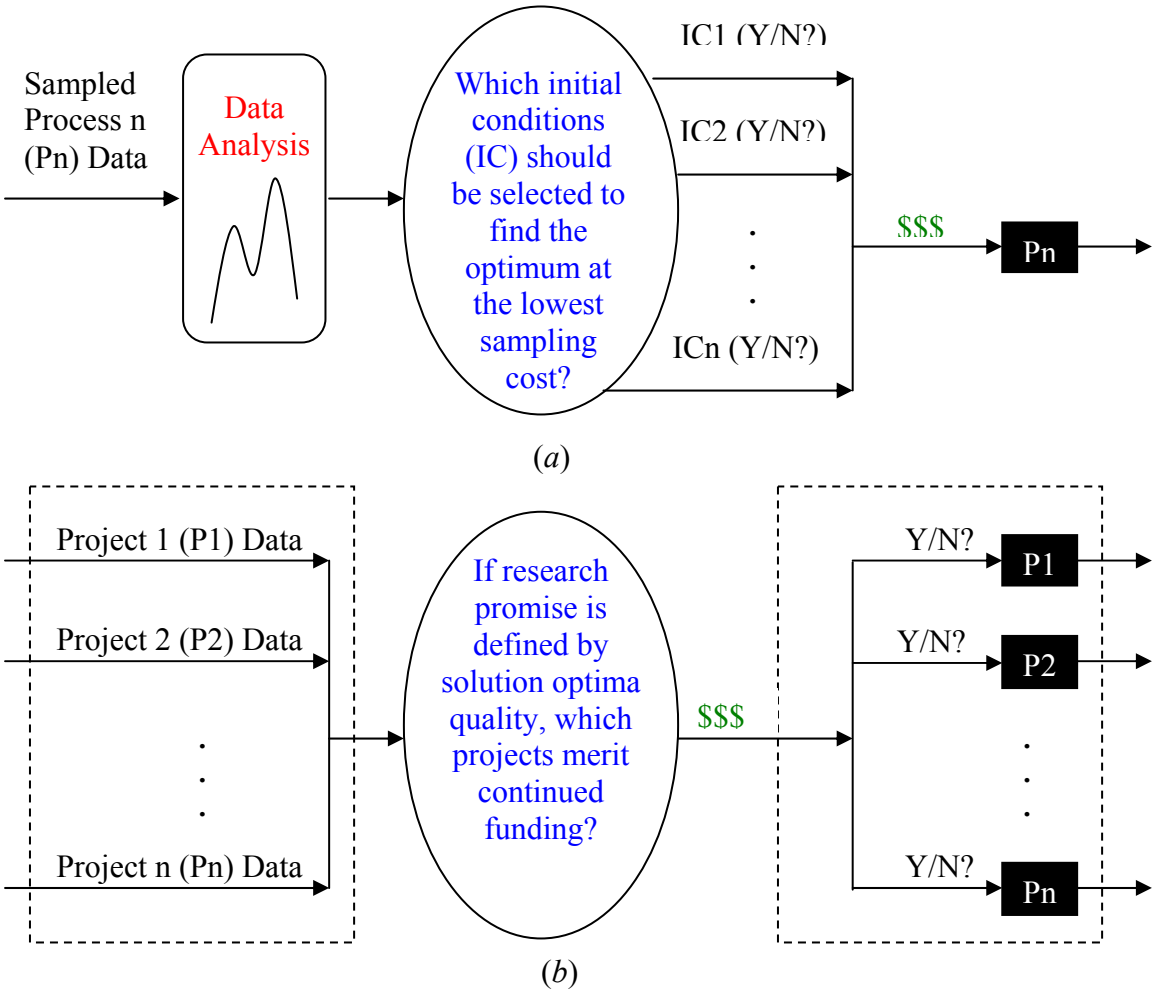


Figure 1.1. Decision stages for (a) sampling and (b) funding allocation.

Continued project funding or production line existence may depend upon the determination of design and/or operating conditions at which the most efficient operation is attained, a task defined as process optimization. These conditions are defined as *process optimal* and quantify a system's promise and/or performance. If this information can be efficiently determined, decision-making quality is improved.

A set of input conditions can be verified as locally process optimal if two optimality conditions are satisfied. A locally optimal set of input parameters can also be globally optimal, but the criteria are more difficult to satisfy. This thesis work is directed at the solution of a class of problems for which global optimality cannot be guaranteed, so the following presentation of the local optimality criteria is sufficient in order to explain the difficulty associated in finding an optimum when accurate process models are absent.

For ease of presentation, let the term input conditions colloquially refer to the inclusive set of design and/or operating conditions at which process optimality is to be attained. Operation efficiency is expressed in terms of either a single or combination of quantified objective functions f(x), such as product purity/yield maximization, or cost/side product minimization. For a vector of input conditions x and a prespecified tolerance ε, a local minimizer is a point x^* satisfying the condition:

$$f(x^*) \leq f(x) \text{ for all } x \text{ such that } ||x - x^*|| < \varepsilon.$$

Process operation at the operating conditions x leads to a locally minimized objective function having value $f(x^*)$ if the following *optimality conditions* are satisfied:

1) The value of the objective function gradient, f'(x), or a reduced objective function gradient, f_r'(x), is zero. A reduced objective function f_r(x) is created when system constraints are present and is formulated as the sum of f(x) and the system constraints, each weighted by a Lagrange multiplier.
2) The sign of the Hessian matrix is greater than zero. The Hessian matrix is the matrix of all second-order partial derivatives of f(x) or f_r(x), as appropriate, and the Hessian is its determinant. A locally maximized objective is conversely attained when the Hessian is less than zero.

The application of gradient techniques results in the generation of a function's first and second-order derivatives, from which the above two optimality conditions can then be evaluated for hypothesized optimal input conditions. When process models are absent, the two conditions can be applied to sampling data information from which surrogate models may or may not have been constructed. However, any surrogate model remains an approximation to the theoretically accurate system equations. Therefore, the task of guaranteeing an input condition as process optimal is made more difficult as an optimization framework must be targeted at not only finding good search directions based on sampling data alone, but also capable of identifying good search directions even though a substitute model is only an approximation to the true behavior.

Process models may be absent in the case of emergent technologies such as nanoscale-level research, in which a "first-principles" system description may not yet be fully developed. Alternatively, accurate models may be absent if the theoretical models are either obsolete due to process retrofitting or system noise, or inaccessible if embedded within legacy computational codes or if expert knowledge is no longer available. The

traditional reliance upon a macroscopic continuum-level "first-principles" description of the physical system that presumes the accuracy of closed-form equations is therefore no longer an effective strategy for achieving process optimization. When a functional form for the input-output relationships is unavailable, the process behavior is symbolically described using black-box models as shown in Figure 1.2.

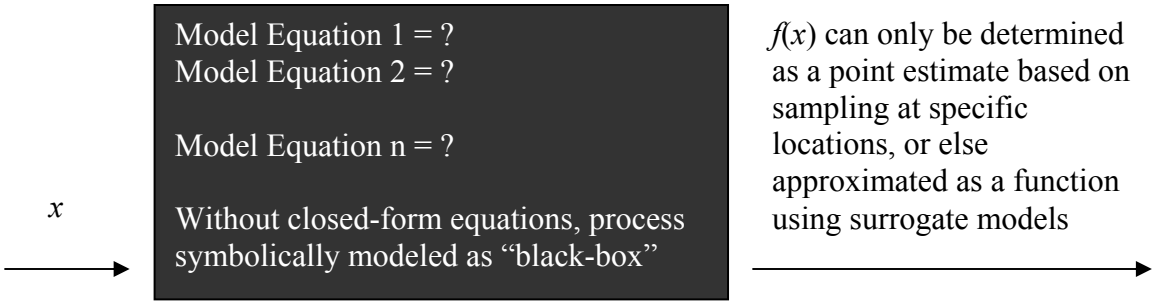


Figure 1.2. Modeling a process as black-box when system models are unavailable.

Problems containing available, yet mathematically intractable, equations constitute a third possible case in which the problem can be considered black-box, since conventional gradient-based methods may fail to find all solutions of interest. One example described by this problem class is reaction rate determination in simultaneous diffusion-reaction systems. A specific problem is taken from Lucia et. al[1] and described in more detail in Chapter 6.5. For this problem, the model equations are given by heat and mass PDE along with system-specific initial/boundary conditions. This system of nonlinear equations may have multiple physically meaningful solutions. Equation-solving routines may require user-specified input conditions, such as warm-start iterates or a specified subregion for search, in order for the entire solution set to be successfully identified; one such example is that of the GAMS NLP solver CONOPT. The discovery of all realistic

solutions is motivated by a practical issue in which there is some metric quantifying one of the solutions as the best. For the diffusion-reaction example, the solution set consists of a set of possible reaction rates. If only a subset of the solutions are found, and the best possible reaction rate remains undiscovered, this reaction system may be discarded from further study beyond the conceptual design level. Therefore, algorithms which address problems having black-box characteristics can also be applied as equation-solving methods.

In addition to being described as black-box, a process may exhibit noisy behavior as well. Process behavior is considered to be noisy when replicate experiments are conducted at identical input conditions and multiple output values are attained, instead of a single value. Noise factors affecting process operation are typically responsible for output variability, with some examples being high batch variability for input species flowrates/compositions, deteriorating mechanical performance due to aging process equipment, process contamination due to environmental toxins such as sulfurs leaking through valve and tank orifices, or adverse process controllability factors due to electronic delays for automated process controllers. Although process troubleshooting is used to identify and minimize noise sources, it may not be possible to eliminate all noise factors. The application of modeling techniques such as kriging, in which noise is considered to be a natural system feature, rather than induced by external environment variables, can lead to the generation of models providing a better system description than models based solely on deterministic models, since process variability is now taken into account.

The research achievements in this dissertation are the development and computational implementation of four optimization algorithms for problems containing black-box models. The algorithms have been developed by modifying two data-driven techniques known as kriging and response surface methodology (RSM), in order to address the solution of problems outside the standard applications for these two procedures.

RSM employs gradient descent after sampling-based models are built in order to determine search directions for an optimum. In the absence of prior knowledge, a quadratic model is commonly employed as the response surface functionality since the behavior near an optimum is quadratic. Far away from an optimum however, the quadratic model may poorly approximate system behavior, and unnecessary resources may be utilized in the search for an optimum due to the identification of poor intermediate search directions. Better approximations are achieved when the region modeled by a quadratic surface is small; consequently system accuracy increases using only localized response surfaces. Although a solution may be found, no mechanism exists enabling x* to be classified as a global optimum, or at least a local optimum that is superior among a set of ancillary local optima that have also been attained.

Kriging is a complementary method to response surface techniques in that it is directed at obtaining a global model of system behavior. The method originated out of the geostatistics literature and is so named after a South African mining engineer, Daniel Krige. The method was originally targeted at describing mineral deposit concentration distributions in caves and mines, in order to determine the best drilling locations maximizing the attainment of ores having a desired grade. This method is typically

applied in only three dimensions in keeping with its original application, due to the spatial visualization attained in the form of 3D mappings.

The proposed techniques in this work extend the capabilities of kriging and response surfaces in order to address a more general class of problems than the primary applications. The advantages of the proposed methods are: 1) efficient identification of locally optimal solutions for problems described by general convex feasible regions, based on sequential optimization of response surfaces built using adaptive experimental designs (Chapter 2), 2) increased probability of finding a problem's global optimum, based on the application of a unified kriging-RSM algorithm (Chapter 3), 3) successful identification of a problem's global optimum when integer variables are present outside the black-box models, based on sequential application of Branch-and-Bound (B&B), kriging, and RSM (Chapter 4), 4) successful identification of a problem's global optimum when integer variables are present both inside and outside the black-box models, based on sequential application of Branch-and-Bound (B&B), kriging, RSM, and direct search (Chapter 5), and attainment of accurate global kriging models via iterative refinement at lower sampling expense, relative to a strategy employing randomized/heuristic sampling, based on application of a centroid-based sampling method (Chapter 6). Although a global optimum cannot be theoretically confirmed due to the fact that true system equations are assumed absent, and that the kriging/response surface models are only approximations to the true behavior, a local solution x_i* to any given optimization problem is colloquially referred to as a global optimum if it is corresponding objective function f(x*) has the lowest value relative to the objective values of the remaining local solutions x_j*. A schematic of the problems addressed in each chapter, the proposed method employed,

and type of solution to be attained – a local or global optimum – is provided in Figure 1.3.

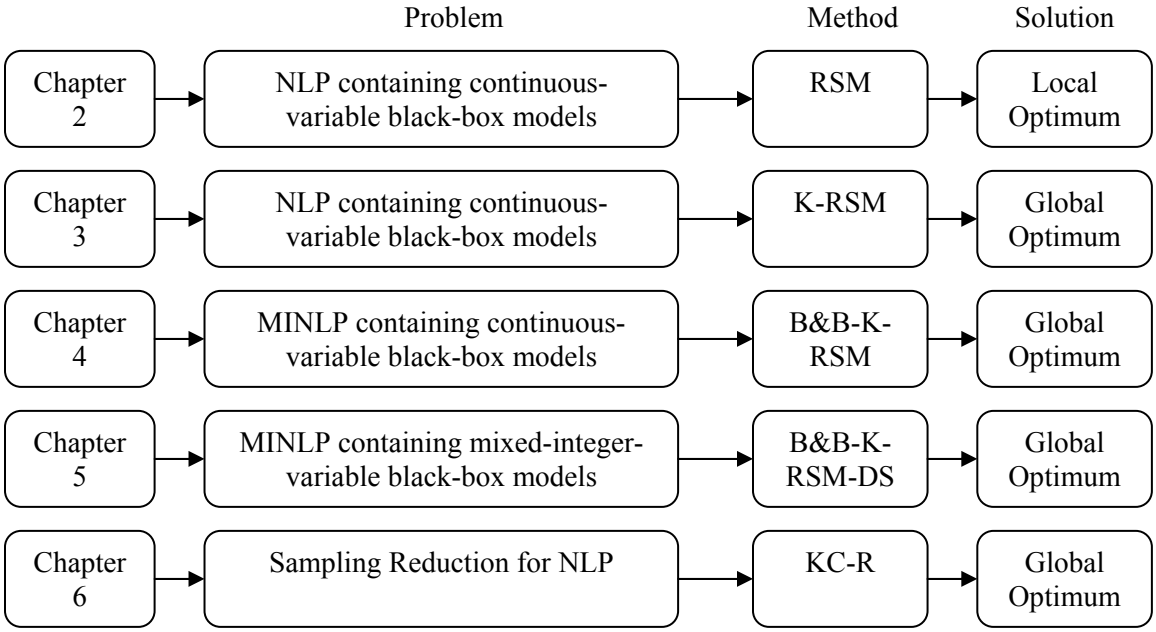


Figure 1.3. Description of the optimization problems addressed in Chapters 2 – 6, along with the corresponding solution algorithms employed and the optima type attained.

Chapter 2

Local Optimization Employing Response Surface Models Constructed From Adaptive Experimental Designs

Response surface methodologies are local optimization techniques that can be employed to find the solution of problems containing black-box models and noisy input-output data. Optimization occurs based on sequential local optimization of quadratic polynomial response surfaces. Each response surface is built from input-output data obtained according to an experimental design template centered around the current best solution, or iterate. The contribution of the work in this chapter is the development of adaptive experimental design templates used in sampling set generation whenever iterates are near constraint boundaries. The templates are employed to 1) enable problem optima to be found whenever the feasible region is a general convex region, and not simply box-constrained, and 2) reduce the overall sampling expense required for discovery of problem optima, based on the construction of lower-D response surfaces whenever iterates are located near boundaries. The adaptive designs presented are employed within three different response surface methodologies to a real case study[1,2].

2.1 Introduction

A response surface is a closed-form analytic model that describes input-output variable relationships according to an additive sum of weighted basis functions. The weights corresponding to each basis function are determined by least-squares fitting, in which the sum of squared errors between the set of model predictions and sampled output data is minimized. Any set of basis functions can be used in model generation[3-6]; however, the simplest, and most widely applied set corresponds to the family of polynomials.

The motivation for using response surfaces is targeted at finding process optima; therefore, based on the optimality conditions provided in Chapter 1, one reasonable requirement of the model used is that it be twice continuously differentiable so that the Hessian can be evaluated. If the Hessian is indeed positive, the neighborhood containing a process optimum is considered to have been found, and refinement of the current best process conditions can be initiated using more localized models. One simple twice-continuously differentiable response surface model is that of a quadratic polynomial containing bilinear interaction terms as given by Equation (2.1):

$$\tilde{z}_2(x) = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i}^{n} \sum_{j=i+1}^{n} a_{ij} x_i x_j + \sum_{i=1}^{n} a_{ii} x_i^2 \qquad (2.1)$$

where x_i , n, and $\tilde{z}_2(x)$ denote the i^{th} input variable, total number of input variables, and the modeled output variable, respectively. The vector [a_0, a_i, a_{ij}, a_{ii}] corresponds to the et of weights for each polynomial basis function.

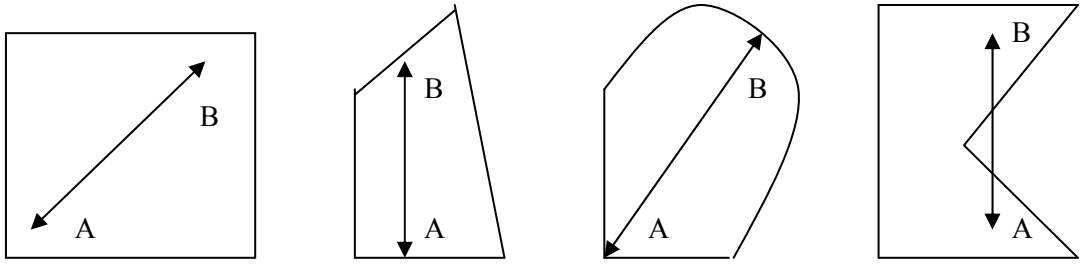
Response surfaces are frequently employed as surrogate models for black-box systems for three main reasons. Firstly, the analytic model employed, such as the polynomial given by Equation (2.1), is often very simple in form, enabling correspondingly simplistic cause-effect relationships between input and output variables to be easily determined based on changes in the operating conditions. Secondly, the computational expense associated with model generation is low, enabling input-output behavior to be quickly characterized for time-sensitive processes in which quick decision-making is essential for sustained acceptable process performance. When analytically simple basis functions are employed for the model structure, the corresponding weights, or coefficients, can be determined at low computational cost using computationally inexpensive least-squares fitting algorithms. Thirdly, the computational expense associated with model optimization is also low, since the application of gradient methods to a simple model enables process optima to be identified quickly. The acquisition of optimal process conditions at low computational cost can be vital for automated processes in which input variable set points can be quickly adjusted to the process optimal values without sustained deterioration in product quality.

For clarification, a response surface methodology, or RSM, is an algorithm for finding process optima using simple analytic models referred to as response surfaces. In its simplest framework, RSM consists of: 1) first building linear models centered around an iterate until the neighborhood of the optimum has been reached, and then 2) applying gradient-based methods to quadratic polynomial models, in order to identify the best search directions leading to an optimum. The neighborhood of the optimum is considered to be identified when the center sampling point iterate has the lowest objective value

relative to the set of objective function values for all sampling points that participate in the current model construction.

RSM has been extensively applied for the solution of low-D box-constrained problems[7,8]. A box-constrained problem is one in which the only constraints are lower and upper bounds for each one of the input variables, and an example of a 2-D box-constrained problem is shown in Figure 2.1(a). However, many process systems are described by feasible regions containing additional linear and nonlinear constraints in the form of sizing and operability constraints. A general framework for the solution of high-D problems whose feasible region is comprised of an arbitrary number of constraints, and not simply defined by lower/upper bound ranges for the problem variables, e.g., box-constrained, has not been found in literature extending over the past thirty years.

The contribution of the research in this chapter is the presentation of a response surface algorithm that can be applied to problems whose feasible region is convex. A convex feasible region is one in which a line can be drawn between any two points lying within or on the boundaries of the feasible region, and for which all points on this line also reside in the feasible region. Examples of convex feasible regions are shown in Figures 2.1(a) – 2.1(c) where it is seen that a box-constrained feasible region is easily ascertained as a convex feasible region. For completeness, a nonconvex feasible region is also shown in Figure 2.1(d).



(a) (b) (c) (d)

Figure 2.1. Convex feasible regions defined by: (a) box constraints, (b) linear non-box constraints, and (c) nonlinear constraints. Nonconvex feasible region (d).

The new ideas employed within RSM are the application of adaptive experimental designs for the creation of *n*-D response surfaces and lower-D projected response surfaces whenever iterates approach constraint boundaries during the search for an optimum. Since the new algorithm can be applied to problems whose feasible regions are asymmetrical and defined by an arbitrary number of convex constraints, as shown in Figures 2.1(b) and 2.1(c), it is an improvement upon standard response surface methodology since a box-constrained feasible region is no longer required in order to apply RSM.

2.1.1 Literature Review

When closed-from process models are absent, the identification of optimal operating conditions can be identified using techniques which do and do not employ model-building as part of the search methodology.

Optimization algorithms are classified as zero, first, or second-order, depending upon whether gradient information is used to guide search. For the class of zero-order methods, no gradient information is employed, and search is instead based on the application of heuristics with respect to sampled data in order to determine promising operating conditions for future testing. Some examples of zero-order methods are the pattern search methods (Hooke-Jeeves, Nelder-Mead, DIRECT, multilevel coordinate search)[9-11], genetic algorithms, and differential evolution[12].

Pattern search methods are applied by generating a sampling set according to a template, identifying the optimum as the vector yielding the best objective value, centering a new template around the new best solution, and repeating the process until convergence in the optimum has been attained. Genetic algorithms and differential evolution techniques are applied by sampling at a randomly selected set of vectors and then performing crossover operations on components of the set of vectors yielding optimal process performance in order to determine new operating conditions believed to yield improved system performance. Hooke-Jeeves and Nelder-Mead represent two pattern search methods in which the optimum is locally found by sequential movement of a sampling set spatially arranged over a small subregion of the feasible region. DIRECT (Divided Rectangles), multilevel coordinate search, on the other hand, are two other pattern search methods that, along with genetic algorithms and differential evolution, belong to a class of global optimization algorithms since the search is based on sampling at vectors dispersed throughout the entire feasible region in an ordered or random manner. The biggest drawback to using these methods is that convergence to an optimum may occur asymptotically.

The application of gradient-based methods can lead to the identification of better search directions towards an optimum than those identified using zero-order methods. Gradient-based methods are categorized as being first-order or second-order depending upon whether strictly first-order information, or both first- and second-order information, respectively, are employed in combination with sampling information to identify future operating conditions at which improved process performance can potentially be achieved. Although gradient-based techniques can be applied to sampling data alone, following search directions based on gradient information obtained from noisy input-output data can lead to the discovery of poor search directions. The search directions will be based on noise-contaminated information rather than on the underlying deterministic input-output functionality. If a surrogate model is first built from the sampling data, and is created based on the application of a methodology minimizing the effect of the noise information, improved search directions may be obtained since the model may provide a better approximation to the underlying system behavior in contrast to the sampling data information alone. For ease of presentation, the term "gradient-based techniques" used in the context of RSM refers to the process of, 1) evaluating first-order derivative information from quadratic polynomial response surface models, and 2) solving the resulting linear system that is obtained from setting the gradient equations to zero. The solution set obtained corresponds to the model optimum.

Although additional sampling expense may be required for surrogate model creation, the associated resource costs are justified by the possibility that the search for a minimum may be accelerated if model construction is added as an intermediate step between sampling and the application of optimization heuristics. The total sampling expense

required to obtain an optimum may therefore be lower than the corresponding cost required from the application of methods lacking a model-building step, provided that the required sampling and model-building expense for reliable iterative model construction is not prohibitive. It should be noted that surrogate model-building is not a requirement for an algorithm to be classified as first- or second-order, as the identification of search directions based on first- and second-order adaptive finite differencing methods[13] such as implicit filtering[14,15] does not require prior model construction. However, if surrogate model-building is indeed one step of a first- or second-order method, search directions are identified based on gradient information directly obtained from the model. Since surrogate models are used to predict system behavior at unsampled data locations, the locations of where an output variable exhibits interesting behavior such as peaks, valleys, or ridge systems can provide additional operability information.

When both the input and output variables are stochastic, a stochastic response surface model, such as the one proposed by Isukapalli[16], can be developed using a stochastic response surface method (SRSM) whose functionality z is of similar form as the equation given in (2.1). In the stochastic response surface model, the deterministic variables x are now replaced by assumed probability distributions pdfs(x) which model the uncertainty associated with the behavior of the now stochastic inputs. As a result, z_2 is therefore also modeled as a stochastic variable having statistical properties such as mean and variance for a given sampling vector. The stochastic response surface model can be viewed as an alternative to the kriging model presented in Chapter 3 since in both cases z is described as a stochastic variable. However, the current implementation of SRSM given by

Isukapalli requires a strict ordered sampling arrangement in order that the set of β coefficients be accurately estimated.

The primary modeling algorithms used in this dissertation are response surface methodologies, presented in this chapter, and kriging (Chapter 3), procedures used in building local and global models, respectively. Based on information obtained from model predictions, an optimization algorithm can then be applied in order to obtain improved process conditions relative to the best candidates currently identified. If the performance of an optimization algorithm, such as a gradient-based method, is computationally inexpensive, the computational burden may instead be associated strictly with the model-building phase. The reason for this is that predictions are generated at a high number of test points in order to obtain a comprehensive model of the system over its entire feasible region. However, provided that a substitute model accurately describes system behavior over a given subregion, the application of surrogate model-based techniques can result in the identification of the search direction leading to an optimum at a lower sampling expense when compared to methods that lack a model-building step.

An optimization algorithm can be considered both efficient and effective if an optimum is successfully discovered at the lowest theoretical sampling expense. This expense can be quantified as the minimum number of field or computer experiments required in order for the optimum conditions to be determined. This number is system-independent and, while it is not known *a priori*, would only be attained if an optimization methodology always identified the best search direction to move in based on the available sampling information. Therefore, one way of measuring the efficiency of a zero, first, or second-order algorithm is by the number of sampling experiments required

to identify a single or subset of optima. Based on this metric, the performance of a new algorithm can be compared against the corresponding efficiency determined for other algorithms in order to evaluate the advantages and limitations of new techniques.

However, sampling expense may not be the only reliable indicator of algorithmic effectiveness and efficiency. When surrogate models are first constructed as part of the optimization algorithm, the computational expense required for model construction must also be considered as another efficiency metric. If the time required for sampling is low, but the computational cost of model building is prohibitively high, the algorithm is considered to be ineffective and inefficient since the time it takes to obtain the optima information may not be justified in terms of the quality of optima information obtained.

For both modeling and optimization algorithms, the computational and sampling costs required for the attainment of optima information generally increases when the number of input variables n, designated as the problem dimensionality, also increases. This is because the number of possible search directions increases multiplicatively as the number of input variables increases, and the identification of poor search directions can result in the sampling expense rising prohibitively with little improvement being attained in the search for optimal process conditions.

2.1.2 Problem Definition

The mathematical formulation for the class of optimization problems addressed using RSM is given by Equation (2.2):

$$
\begin{aligned}
&min \;\; F(x, z_1, z_2) \\
&s.t. \;\; g(x, z_1, z_2) \le 0 \\
&\quad h(x, z_1) = 0 \\
&\quad z_2(x) = \Gamma(\mathrm{x}) + \varepsilon(x) \\
&\quad \varepsilon(x) \in N\left(x \mid \mu, \sigma^2\right) \\
&\quad N\left(x \middle| \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \\
&\quad x \in \Re^n
\end{aligned}
\qquad (2.2)
$$

The variables x, z_1, and z_2, represent the vectors of continuous input variables, output variables whose input-output models are known, e.g. $z_1 = 8x_1^2 - x_2$, and output variables whose input-output relationship is unknown, respectively. In addition, n is used to denote the problem dimensionality in terms of the number of input variables x. The equations for F, g, h, and Γ represent the objective function, closed-form operability constraints, closed-form balance equations such as mass/energy relationships, and black-box functions symbolically describing an unknown functionality between x and z_2. If a closed-form functionality describing the behavior of z_2 as a function of x is developed, it can be substituted into (2.3), replacing Γ, and the optimization problem can be solved using gradient-based techniques.

The closed-form functionality is known as a surrogate data-driven model, defined as such due to its creation from a set of sampled input-output data. The error term $\varepsilon(x)$ is considered to be additive noise term to the deterministic output variable $z_2(x)$ and is

modeled as behaving according to a normally distributed function having mean zero and variance σ^2. Data-driven model terms may not be theoretically significant in terms of physically meaningful terms such as rate, driving force, and resistance variables. However, the identification of cause-effect behavior based on the manipulation of a few input parameters can enable a simplistic understanding of system behavior to be attained at low computational cost.

2.2 Solution Approach

By definition, the model built from any modeling algorithm can be considered as a response surface, since there is some response variable being represented as a function of a set of input variables. However, the term "response surface" usually refers to either a linear or quadratic polynomial model, in contrast to a model comprised of non-polynomial basis functions. Therefore, models built from radial basis function methods, which employ exponential basis functions, or from artificial neural network techniques, which employ signomial and lognormal functions for signal processing applications, or inverse distance weighting methods, such as kriging, which employ a wider family of nonlinear basis functions, may not be referred to as response surfaces in the literature.

As mentioned previously, a response surface is an analytic functionality comprised of a sum of basis functions, each multiplied by an unknown coefficient. The coefficient vector, given as $[a_0, a_i, a_{ij}, a_{ii}]$ in Equation (2.1) for a quadratic polynomial response surface, is determined using least-squares fitting based on the minimization of the total squared error between sampled output data and corresponding model predictions. The number of sampling vectors required for modeling is fixed according to a user-specified experimental design. This design template is centered around an iterate x_c representing

the current set of operating conditions yielding the best process performance. If x_c resides far away from the operating constraint boundaries, the sampling vectors are identified using a design targeting the construction of a response surface built with respect to n independent variables. Conversely, if x_c is near any r constraints, the sampling vectors are identified using a design targeting the construction of a response surface built with respect to $(n - r)$ independent variables, where n designates the problem dimension whose value is given by the number of input variables.

Once the response surface has been built and minimized, a new iterate x_c, defined as the new best set of operating conditions, is determined once sampling has occurred at the response surface minimum. If convergence in the problem objective $F(x_c)$ has not yet occurred, the procedure is repeated whereby another response surface is constructed around the new x_c; otherwise, the algorithm is terminated. An illustration of the RSM algorithm being applied is presented in Figure 2.2. A flowchart containing the basic steps of this standard method, referred to as the RSM-S method, is given in Figure 2.3.

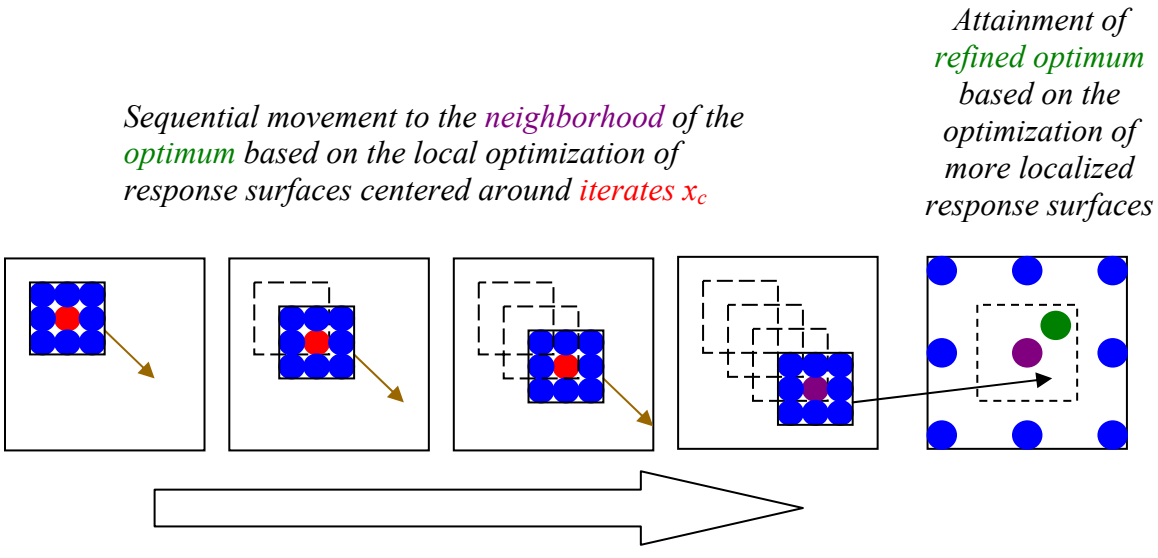


Figure 2.2. Application of the RSM-S algorithm for finding a system optimum.

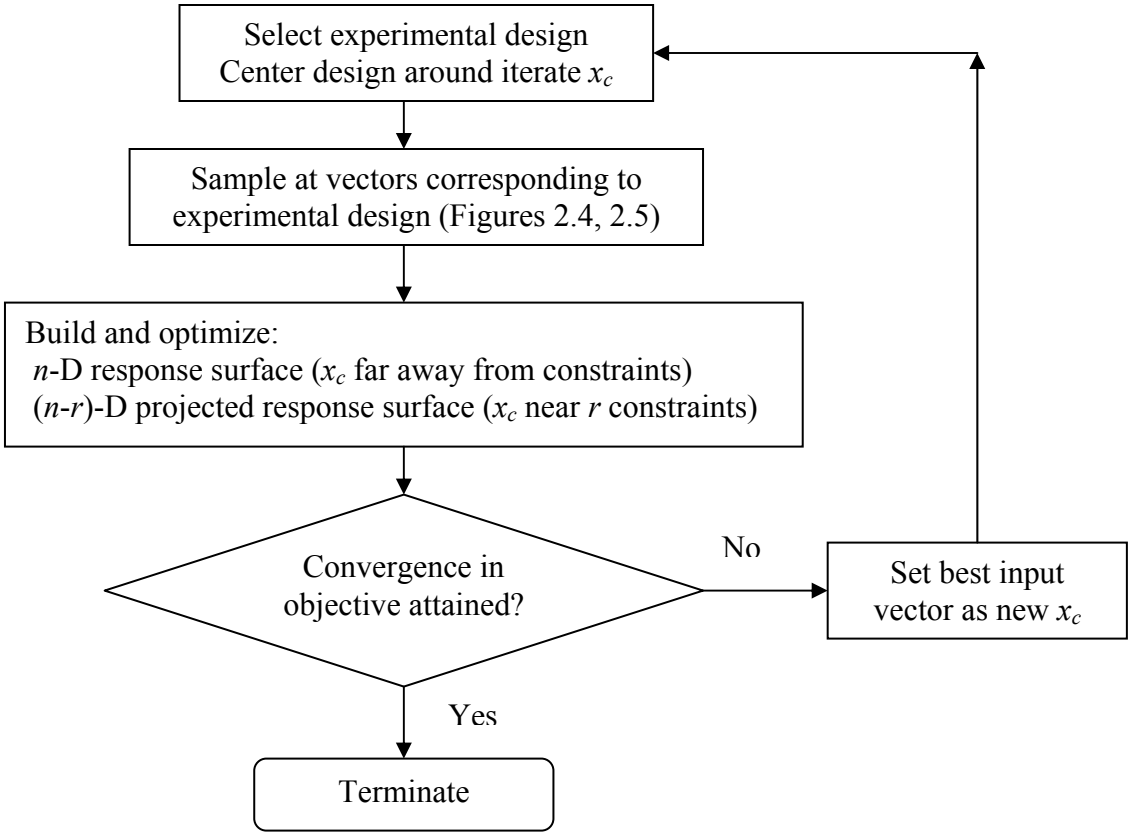


Figure 2.3. Flowchart of an RSM algorithm employing projected response surfaces in order to efficiently find optima residing at boundaries of constrained problems.

The promise of the RSM-S method, is based on the possibility that the early movement along a "fine" search direction will accelerate search towards the optimum However, since the response surface model used in this dissertation is a quadratic polynomial, accurate models are expected only in regions which contain linear or quadratic system behavior. Linear behavior is generally observed only in localized subregions, and quadratic behavior is observed in the neighborhood of a process optimum. Since many systems exhibit more nonlinear behavior over the majority of the

feasible region, and since it is unlikely that a random starting iterate will be located within the vicinity of an optimum, the value of the information obtained from building quadratic response surfaces is likely to be poor. Therefore, the sampling expense associated with building a full response surface during the early stages of RSM is not considered justified.

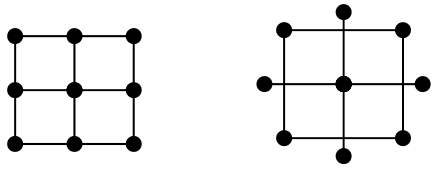
In order to reduce the sampling expense required for attainment of the problem optimum, two additional response surface methodologies, denoted as the DS-RSM and RSM-G algorithms, are applied. In the first method, response surfaces are built only at the later stages of optimization; in the second method, the initial response surface is optimized with respect to the global feasible region in contrast to a local subregion as done with the standard RSM-S algorithm. The methodology of the DS-RSM technique is presented in Section 2.2.5.

Each one of the response surface algorithms - DS-RSM, RSM-S, and RSM-G employ experimental designs for the determination of sampling vector locations. In Section 2.2.1, the equations for the factorial and central composite designs are presented. In Section 2.2.2, conditions are provided which specify both the radius and selection of a particular experimental design template to employ for response surface construction whenever an iterate is near linear or nonlinear constraints. In Section 2.2.3, a template is presented for response surface construction whenever operability or model equations are present in the form of equality constraints $h(x,z_1)$ as given in Problem (2.1). In Section 2.2.4., a template is presented for lower-D response surface construction whenever iterates approach constraints, which results in lower-D response surfaces being projected onto nearby constraints. In Sections 2.2.5 and 2.2.6, two variations of the standard response

surface algorithm described above are presented which target overall sampling expense reduction in the search for problem optima.

2.2.1 Factorial and Central Composite Experimental Designs

The sampling set used in building response surfaces conforms to a stencil arrangement known as an experimental design that typically consists of sampling vectors symmetrically located around a center iterate. Two commonly employed experimental designs are the factorial and central composite designs, shown in Figure 2.4.



(a) (b)

Figure 2.4. 2-D factorial design (a) and central composite design (b).

The factorial design shown in Figure 2.4(a) is a template used for determining sampling locations at all factor-level combinations. Let the variables x_i, l_i, and $l_{tot,i}$ denote the i^{th} input variable, i^{th} operating condition, such as low/medium/high dosage level, and total number of operating conditions for x_i, respectively. For this design, $\prod_{i=1}^{n} l_{tot,i}$ sampling experiments are required, a number that increases quickly if many levels are considered or if many input variables exist. Due to the increased sampling expense, which can become prohibitively high, this design may not be appropriate if either one of the above two conditions are present.

The central composite design (CCD), shown in Figure 2.4(b) for a problem containing two input variables, requires $(1 + 2n + 2^n)$ sampling points for a problem of dimension n. The CCD requires a lower sampling expense relative to a factorial design since data are not obtained at every factor-level combination. It should be noted that a 2-level factorial design is identical to a CCD lacking its axial points.

The procedure used to generate a sampling set using the factorial design or CCD is now described. The sampling set is comprised of peripheral sampling vectors x_p symmetrically arranged about a center iterate x_c. Each peripheral sampling vector x_p is obtained as the sum of the corresponding center iterate x_c plus a vector containing scalar multiples of a constant b. The value of b is considered to be a user-defined bound, or radius, of the resulting response surface, which is created with respect to n input variables. In the absence of prior knowledge, an example value for b can be a fraction, such as one-tenth, of an input variable's feasible range. The equations for generating x_p for the first two components of x_c, $x_{c,1}$ and $x_{c,2}$, respectively, are given in Figures 2.5 and 2.6 for the factorial design and CCD, respectively.

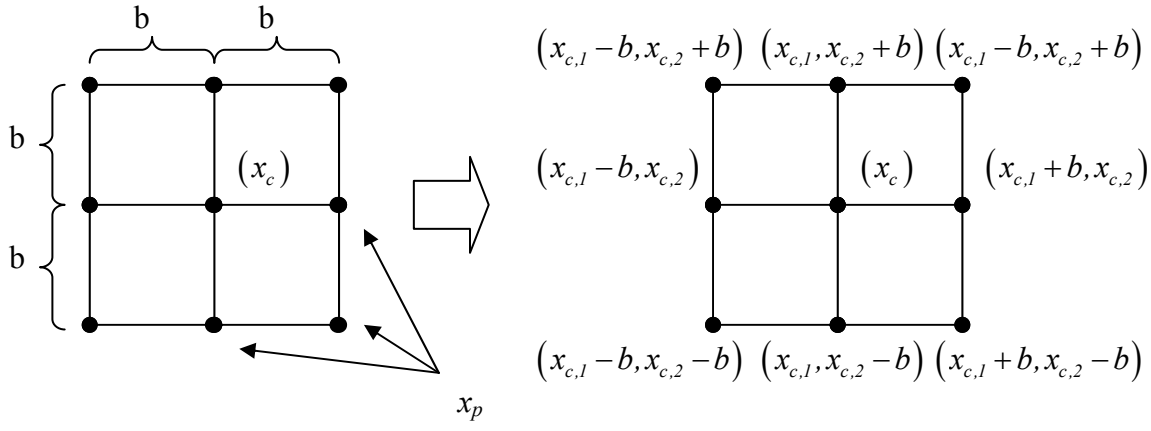


Figure 2.5. Sampling set generation based on a factorial design centered around x_c.

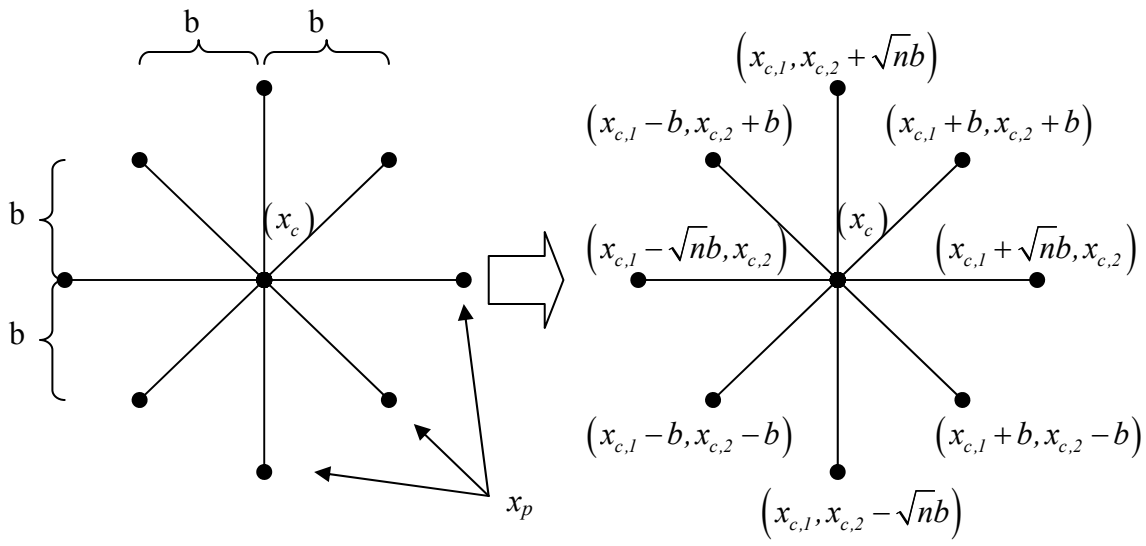


Figure 2.6. Sampling set generation based on a CCD centered around x_c.

2.2.2 Experimental Design Selection and Radius Sizing for Response Surface Construction Near Convex Constraints

If the convex feasible region is of arbitrary shape and iterates x_c approach nonlinear or multiple boundaries, infeasible collocation points may be identified using symmetrical design stencils. Since more reliable search directions are usually obtained from response surfaces constructed over symmetrical design stencils, in order to retain feasibility, the experimental design for collocation point generation is adaptively changed depending upon iterate proximity to the boundary. If iterates are near nonlinear boundaries or non-orthogonal boundaries whose interior angle is acute, the central composite design is used, as shown in Figure 2.7(a) and 2.7(b), respectively. Conversely, if the iterate is near a linear boundary or a set of linear boundaries whose interior angle is orthogonal or obtuse, a factorial design is applied, as shown in Figure 2.7(b). The particular designs used according to the above stated conditions provide a more advantageous search of the

respective subregions defined by the surrounding constraints over their counterparts. However, due to the lower associated sampling expense relative to the factorial design, the CCD is a more attractive design template to use as the problem dimensionality increases regardless of iterate proximity to linear boundaries. To further ensure feasibility for linearly constrained problems, the maximum design radius is set as the minimum of the distance between the iterate and all constraints, shown in Figure 2.7(b) and illustrated in more detail in Figure 2.8.

(a) (b)

Figure 2.7. Application of a CCD or factorial design depending upon iterate proximity to a linear or nonlinear boundary (a). Ensuring feasibility by setting the maximum radius as the minimum iterate-constraint distance (b).

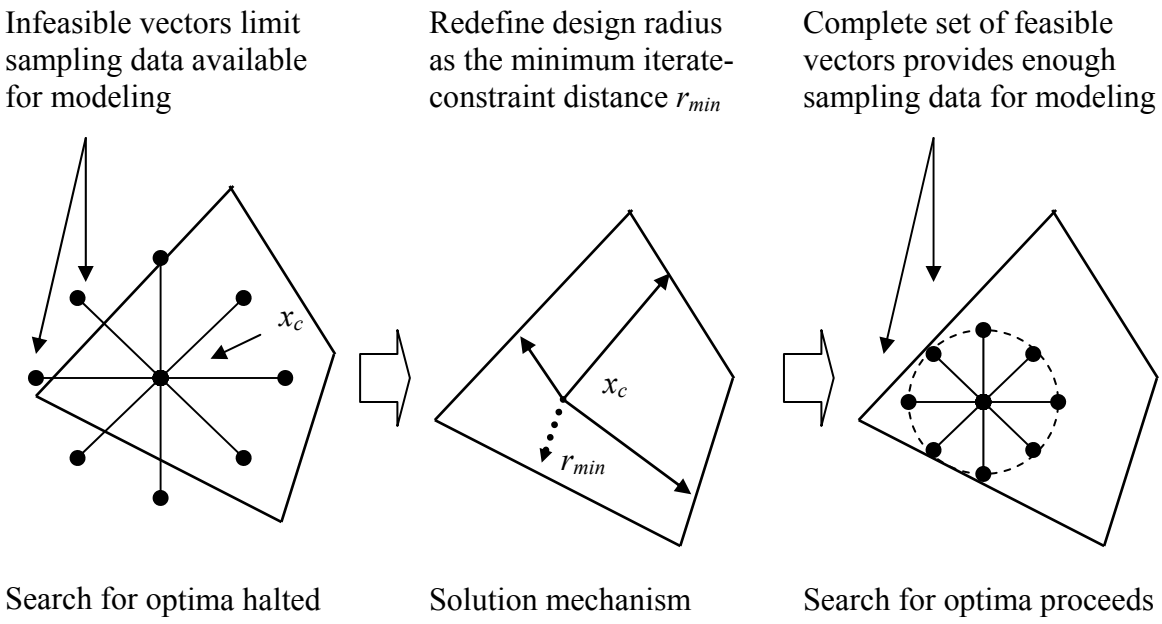


Figure 2.8. Determination of the maximum allowable design radius in order to ensure all sampling points remain feasible for modeling.

2.2.3 Experimental Designs for Equality-Constrained Response Surface Construction

The determination of each $\{x_1 \ldots x_n\}$ component for each sampling vector x_p must be modified if equality constraints $h(x,z_1)$ are present, since these constraints may result in the generation of infeasible input variable vectors. Consider an imaginary test problem involving a box-constrained optimization problem having n input variables. A box-constrained problem is one in which the operability region is defined simply by lower and upper bounds on all input variables. Suppose an operability condition $h_{oc}(x,z_1) = 0$ is added to the problem formulation in which the value of one input variable must assume a

fixed value relative to another. As an example, suppose that $h_{oc}(x,z_1) = 2x_1 - x_2 = 0$. Based on this constraint, the value of x_1 must be twice that of x_2 for any and all sampling vectors. From a given design template, as shown in Figure 2.6, x_1 and x_2 are also determined by a separate set of equations based on x_c, b, and n. If the x_1- and x_2-components are determined according to the design template equations, they may be infeasible with respect to $h_{oc}(x)$, causing the referenced sampling vector x_p to also be infeasible.

In order to ensure feasibility in all sampling vectors, the response surface must then be built with respect to (n-1) independent variables, where the $\{x_2 \ldots x_n\}$ components are determined according to the equations for an (n-1)-dimensional design. The x_1-component for each sampling vector is now determined in terms of x_2 according to $h_{oc}(x)$ as shown in Equation (2.3).

$$h_{oc}(x_1 \ldots x_n) = 0 \Rightarrow x_1 = h_{oc}(x_2 \ldots x_n) \tag{2.3}$$

This technique can be generalized whereby if an optimization problem contains r operability conditions prescribed as equality constraints, the response surface is then built with respect to (n-r) independent variables.

2.2.4 Experimental Designs for Response Surfaces Projected onto Convex Constraints

The limitation of the rule which enforces a maximum radius r_{min} for feasibility purposes as described in Section 2.2.2 is that as iterates x_c approach a boundary $g_{close}(x_1 \ldots x_n,z_1)$, r_{min} can become small. Local optimization of response surfaces constructed over a small region results in small steps taken towards the optimum,

resulting in only a limited improvement being attained in the objective at the cost of the sampling required to build the local model. A more efficient application of the amount of sampling information can be attained if the response surface is "projected" onto the nearby constraint as shown in Figure 2.9. Projection is applied once r_{min} falls within a small percent of the distance between g_{close}, x_c, and the opposite boundary g_{far} in the normal direction between g_{close} and x_c.

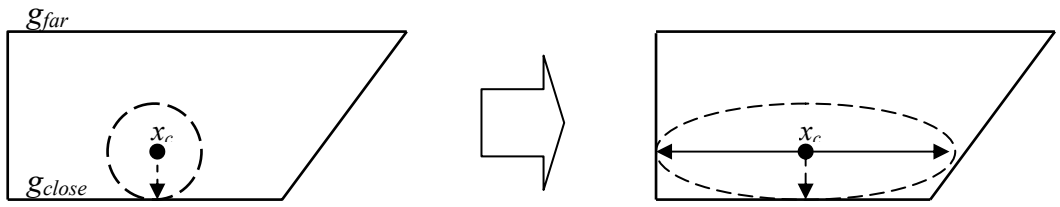


Figure 2.9. Projection of an *n*-dimensional response surface onto constraints to avoid taking small steps towards the optimum.

The projection is accomplished by treating $g_{close}(x_1 \ldots x_n, z_1)$ as an equality constraint as shown in Figure 2.10, where this particular inequality constraint is removed from the set of *t1* inequality constraints $g(x,z_1)$ and added to the set of *t2* equality constraints $h(x,z_1)$. This technique results in one of the *x*-components of g_{close}, e.g. $\{x_1 \ldots x_n\}$, no longer being treated as an independent variable, as discussed in the concluding section of Section 2.2.3.

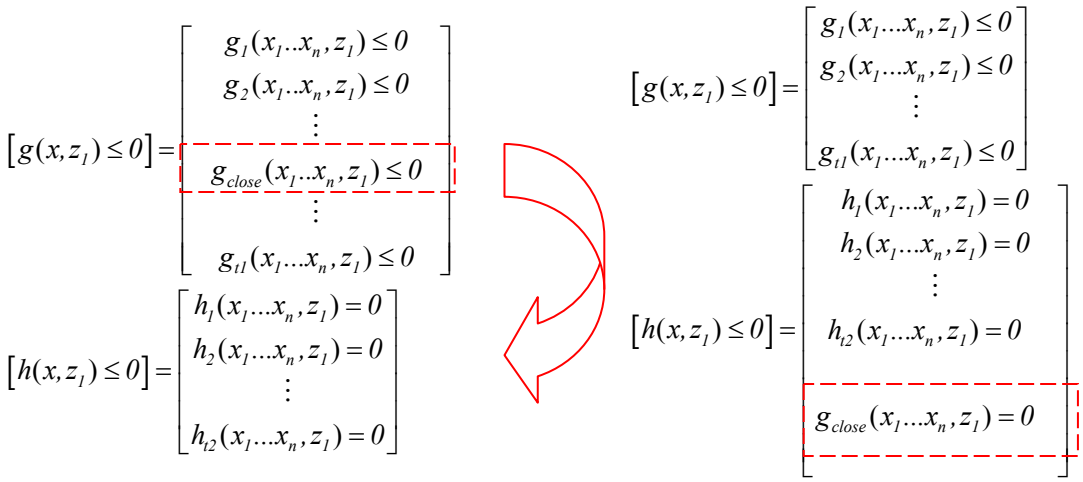


Figure 2.10. Conversion of a nearby inequality constraint $g_{close}(x,z_1)$ into an equality constraint $h_{new}(x,z_1)$ in order to enable lower-D response surface construction projected onto $g_{close}(x,z_1)$.

The projected response surface is constructed with respect to $(n - r - 1)$ independent variables as shown in Equation (2.4). One benefit of applying this technique is that significantly fewer sampling points are required to construct an $(n - r - 1)$-dimensional response surface when compared to the number required for an n-dimensional model.

$$g_{close}(x_1...x_n)\le 0 \Rightarrow g_{close}(x_1...x_n)=0 \Rightarrow x_1 = g_{close,rearranged}(x_2...x_n) \qquad (2.4)$$

The method of using projected response surfaces is illustrated in Figures 2.11 and 2.12 for a problem containing three input variables. In Figure 2.11, seven 3-D response surfaces are required before the optimum is attained. Each response surface is built from sampling information taken at vectors corresponding to a CCD, since only fifteen data points are required in contrast to the twenty-seven required by a three-level factorial design. The solution is found after the sequential optimization of seven response surfaces. An approximate total of (7)(15) = 185 sampling experiments are required. In Figure 2.12,

application of the projection technique results in the optimum being found after five response surfaces are built. Nine sampling experiments are required for a 2-D CCD, and so an estimated total of (3)(15)+(2)(9) = 63 experiments are needed before the optimum is attained, a 68% reduction in resource costs when compared to the 185 required if projection is not employed. The total sampling values of 185 and 63 represent the maximum number of experiments required, since a sampling vector may correspond to the design points of both an old and new response surface, negating the need for additional experimentation. In order to further reduce the sampling expense for higher-dimensional problems, line search techniques can also be incorporated into the algorithm whereby sampling is conducted at a sequence of points obtained along the path defined by the current and previous iterates.

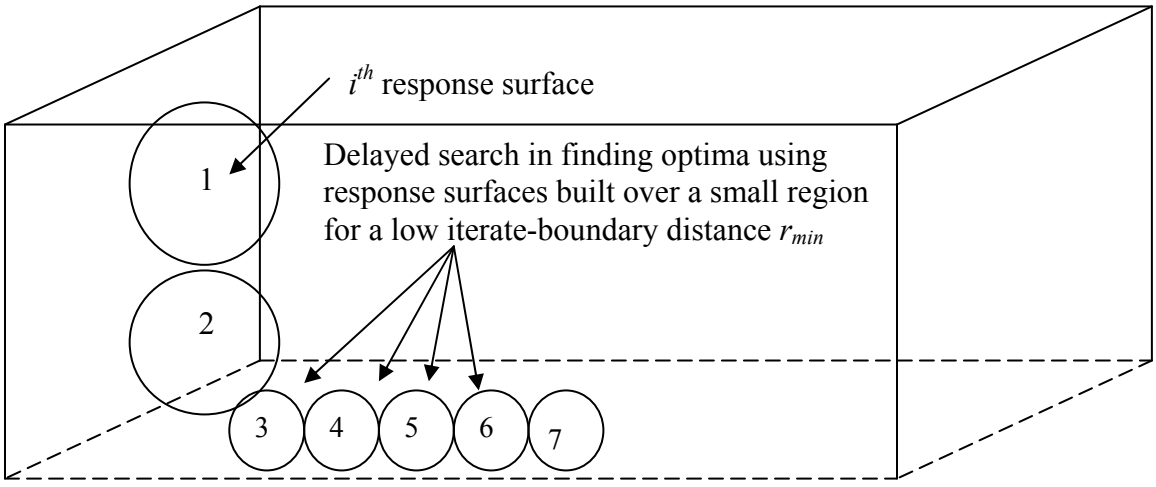


Figure 2.11. Delayed search for the optimum when movement based on low-radius response surface models (4 – 7) results in small steps being taken towards the optimum.

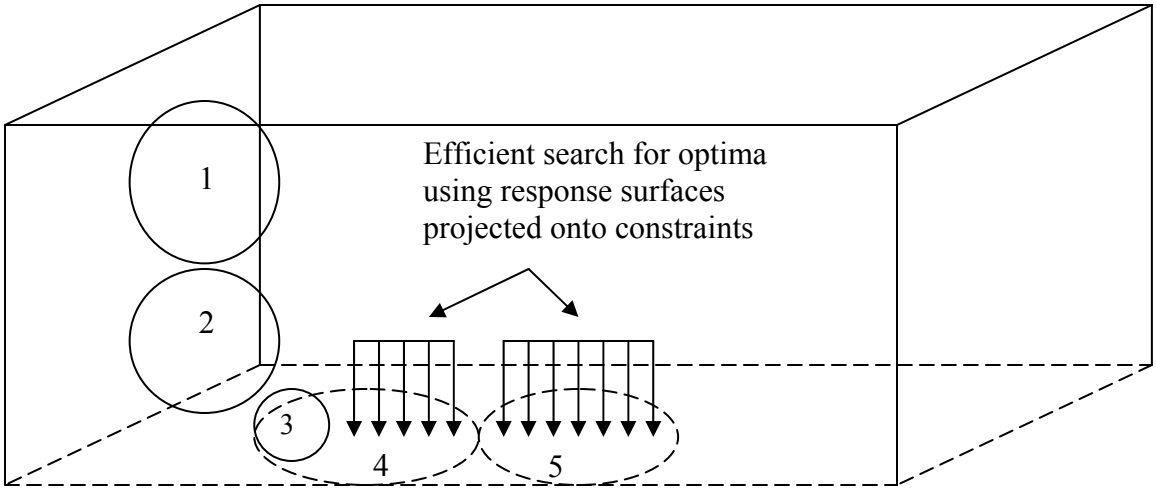


Figure 2.12: Accelerated search for the optimum when movement based on high-radius response surface models (4 – 5) enables large steps being taken towards the optimum.

The construction and optimization of projected response surfaces onto constraints using adaptive experimental designs is incorporated into the DS-RSM, RSM-S, and RSM-G response surface algorithms, enabling search towards the optimum to be accelerated for constrained high-D problems. The methodology of the DS-RSM algorithm is presented in the next section.

2.2.5 DS-RSM Algorithm

The methodology of the DS-RSM algorithm is based on the sequential application of, 1) direct search in the early stages, and 2) local optimization of response surfaces using gradient methods in the later stages once the neighborhood of an optimum has been identified.

First, the iteration index j is initialized at unity. Sampling is performed at input conditions $x_{c,j}$ selected as a first guess for the optimal process parameters. A 2-level factorial design having nominal bounds b_j is centered at $x_{c,j}$. In the presented examples, the initial value of b_j is set between 10% and 25% of the maximum feasible region radius. The 2-level design simply calls for low and high process sampling conditions relative to $x_{c,j}$ and each peripheral sampling point x_p is identified according to the equations given in Figure 2.13.

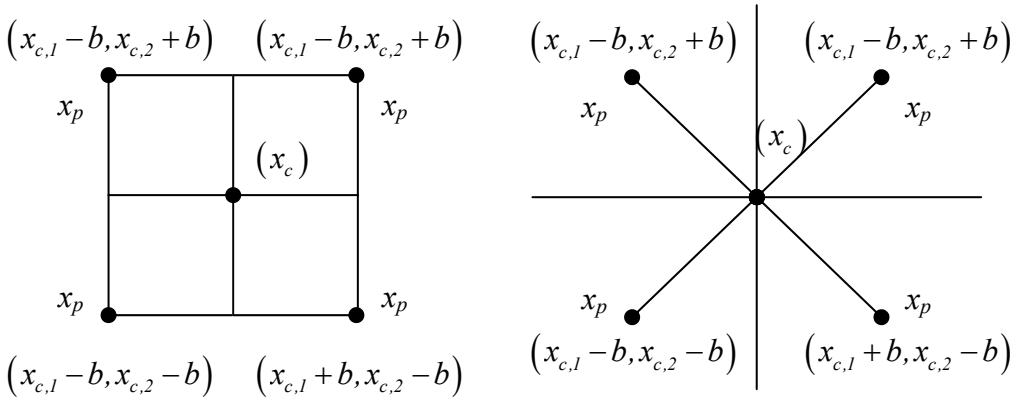
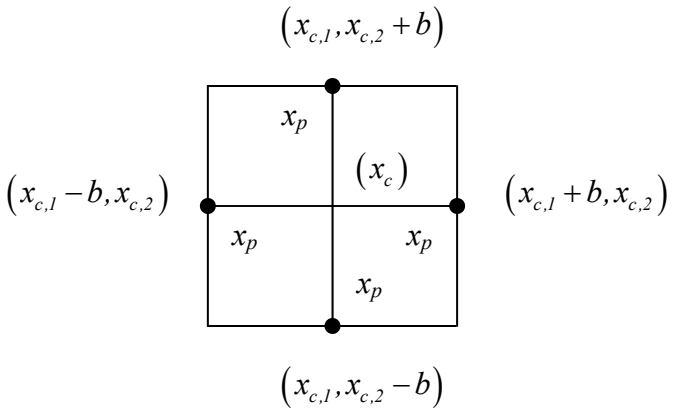


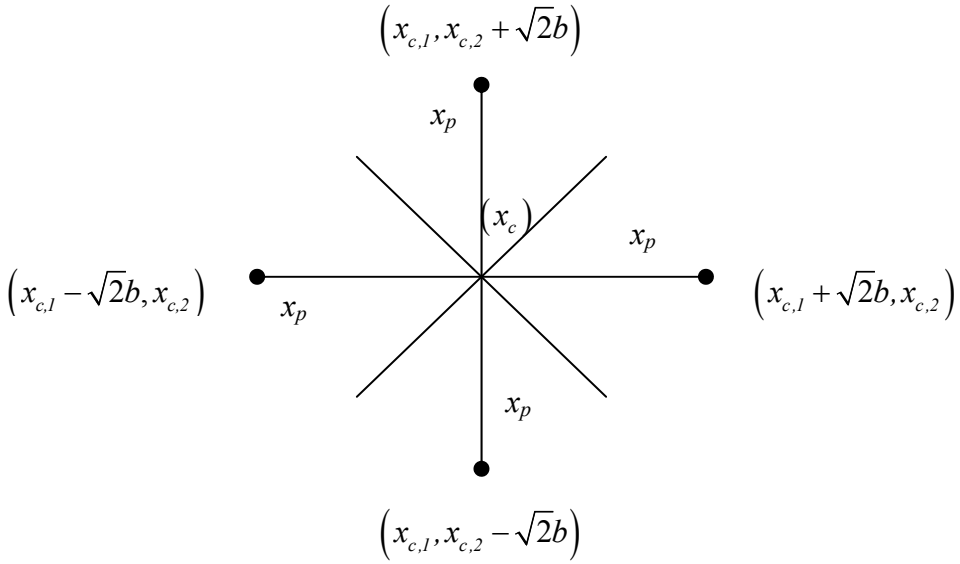
Figure 2.13. 2-D 2-level factorial design used as a base template for the 3-D factorial design (a) and CCD (b) in response surface construction.

Once sampling has been performed at all x_p, the vector at which the lowest objective value is attained is defined as $F_{min}(x_p)$. If $F_{min}(x_p)$ is less than $F(x_{c,j})$, the corresponding $x_{p,j}$ location becomes the new center iterate $x_{c,j+1}$. The iteration index j is increased by unity and a new set of sampling points are generated around the new $x_{c,j}$ vector. If $F(x_{c,j})$ is lower than $F_{min}(x_p)$, sampling is performed at the unsampled points corresponding to a 3-level factorial design, or, for high-dimensional problems, the CCD as given by the

equations for x_p in Figures 2.14(a) and 2.14(b), respectively. The value of $F_{min,j}$ is updated based on the additional objective function values obtained from sampling at additional x_p locations.



(a)



(b)

Figure 2.14. Additional x_p sampling location equations for the 3-level factorial (a) and central composite (b) designs in 2-D.

The response surface is then fitted by determining coefficients $[a_0,a_j,a_{jk},a_{kk}]$ such that the squared error between the objective values based on sampled data, and the model objective function values, is minimized. A predicted optimum objective function value $P_{min,j}$, located at $x_{test,j}$, is then determined from local minimization of the response surface over the design region.

Sampling is performed at $x_{test,j}$ to confirm whether process operation at $x_{test,j}$ leads to improved product quality. The new best solution vector $x_{c,j+1}$ is then determined as the one having lowest corresponding objective value from the set $\{F(x_{c,j}),F(x_{p,j}),F(x_{test,j})\}$. If the difference between the current and previous optimal objective values $F(x_{c,j+1})$ and $F(x_{c,j})$ exceeds the value of a stopping tolerance *tol*, a new design radius b_{j+1} is defined in preparation for building another response surface. The new radius is half the value of the old radius if $x_{c,j+1}$ and $x_{c,j}$ are identical, which signifies the need for more localized response surface construction around the current solution vector in order to refine the optimum. If $x_{c,j+1}$ and $x_{c,j}$ are different, b_{j+1} is set as the Euclidean distance between the two vectors. The iteration index j is increased by unity and another response surface is created. Conversely, if the difference between $F(x_{c,j+1})$ and $F(x_{c,j})$ falls below the value of *tol*, the procedure terminates and $x_{c,j+1}$ is considered to be process optimal having corresponding objective value $F(x_{c+1,j})$. The DS-RSM algorithm is illustrated in Figure 2.15 and a flowchart of the methodology is given in Figure 2.16.

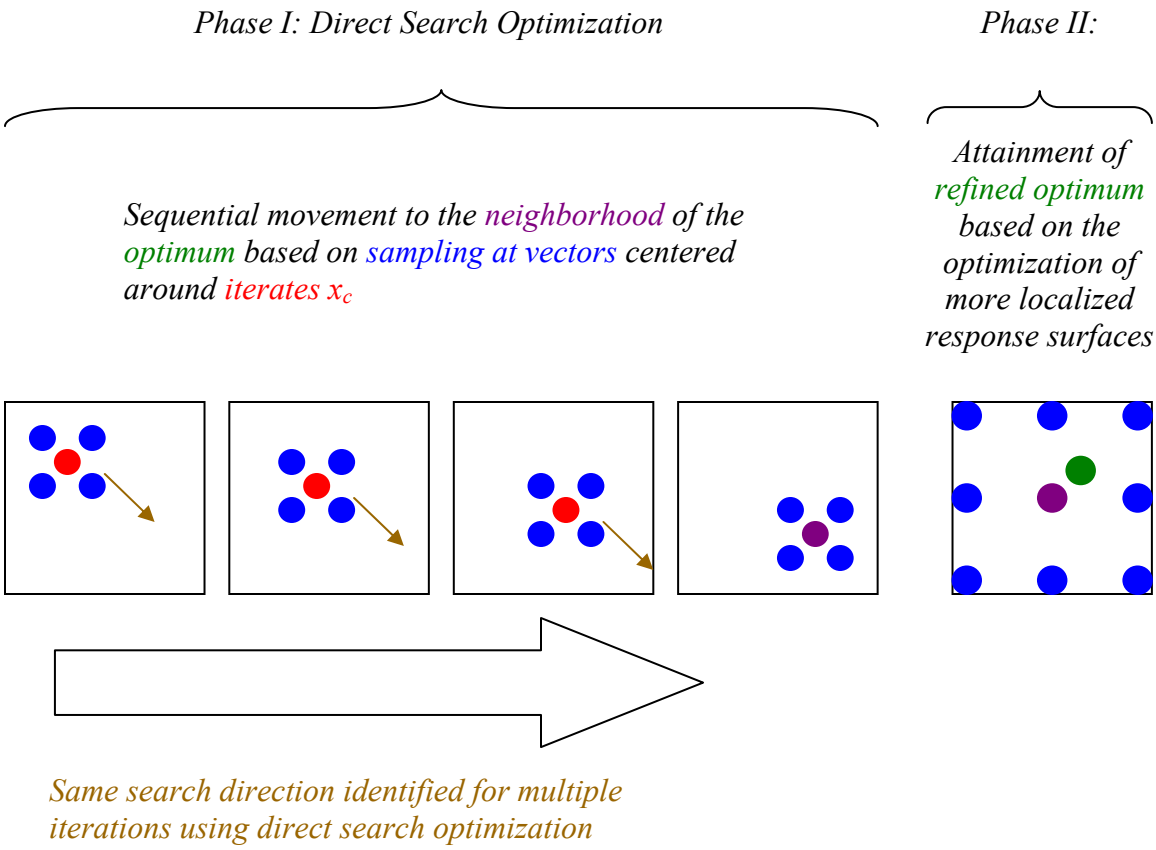


Figure 2.15. Application of the DS-RSM algorithm to find a system optimum.

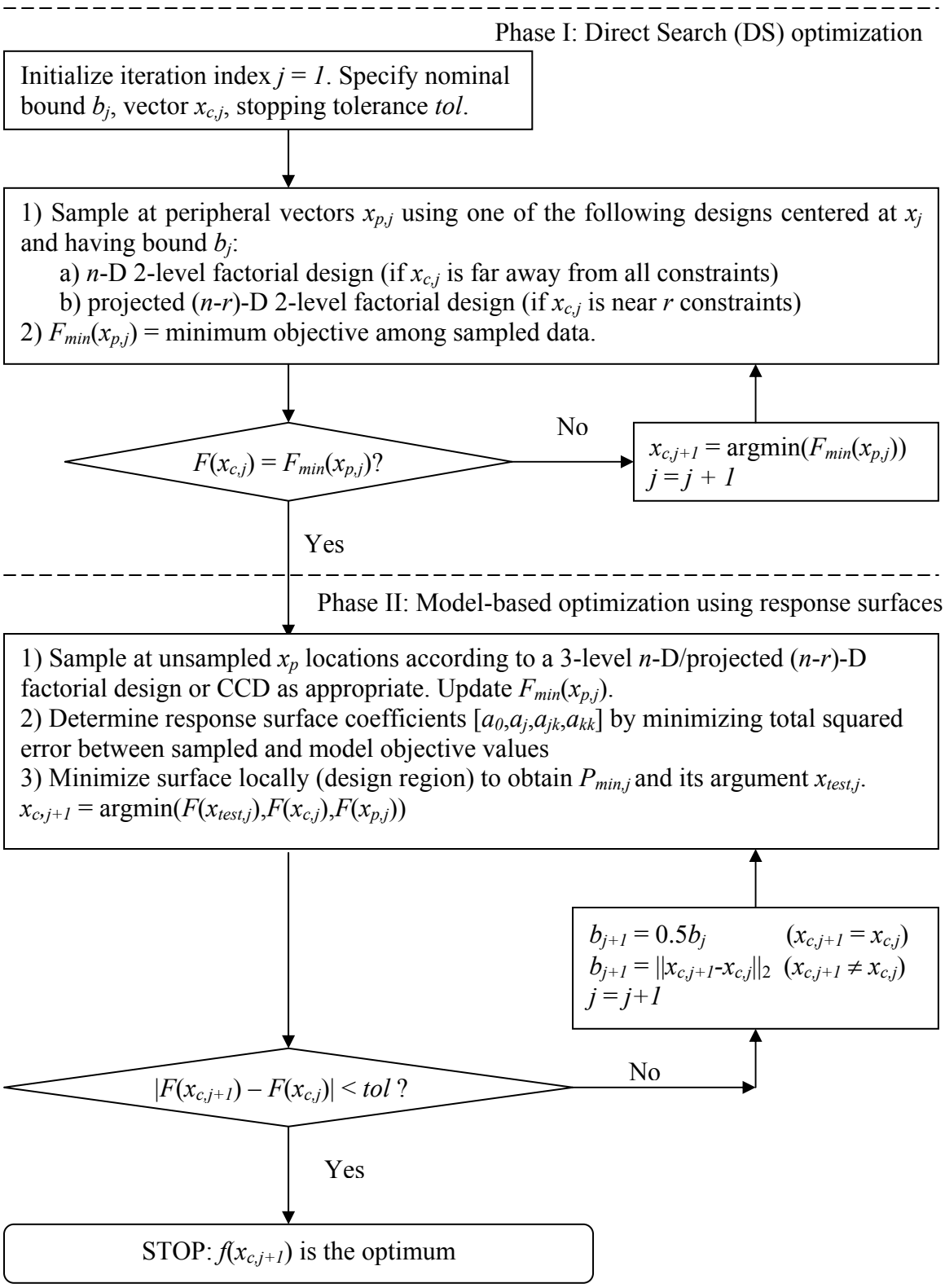


Figure 2.16. Flowchart of the DS-RSM algorithm.

When the DS-RSM algorithm is applied, movement towards the optimum can only be accomplished by following the best of a limited set of search directions defined as the path between the center iterate and each one of the peripheral sampling vectors. The available search directions of the DS-RSM algorithm correspond to a "coarse" search direction, in contrast to the availability of a family of "fine" search directions identified from applying gradient methods to a response surface built at each iteration as done for the RSM-S algorithm.

The primary benefit of applying the DS-RSM algorithm relative to the RSM-S technique is the lower sampling expense incurred during the early stages of optimization. Since fine search directions are needed at only the end stages of optimization, in order to refine the best candidate solution, the benefit of applying the RSM-S algorithm using quadratic polynomial response surfaces fails to be realized until an iterate approaches the search path leading directly to the optimum. It is at this time that response surfaces are likely to be built for the DS-RSM technique; therefore the increased sampling expense associated from building a response surface is motivated by the strong probability of attaining an improved problem optimum.

2.2.6 RSM-G Algorithm

The methodology of the RSM-G algorithm is based on the global optimization of a response surface built at each iteration in order to identify an updated iterate x_c residing close to the optimum. The promise of the RSM-G algorithm is based on the possibility that while an iterate may reside far away from the optimum, the search path between the iterate and optimum is approximately a straight line. The application of this method leads

to a possibly accelerated search for the optimum when it resides far away from the nominal iterate.

The benefit of globally optimizing the response surface is realized when either 1) the underlying system behavior is approximately quadratic, or 2) when optima lie near feasible region boundaries, and the locally optimal search direction remains unchanged for subsequent models created along this path as shown in Figure 2.17. A flowchart of the RSM-S and RSM-G algorithms is presented in Figure 2.18.

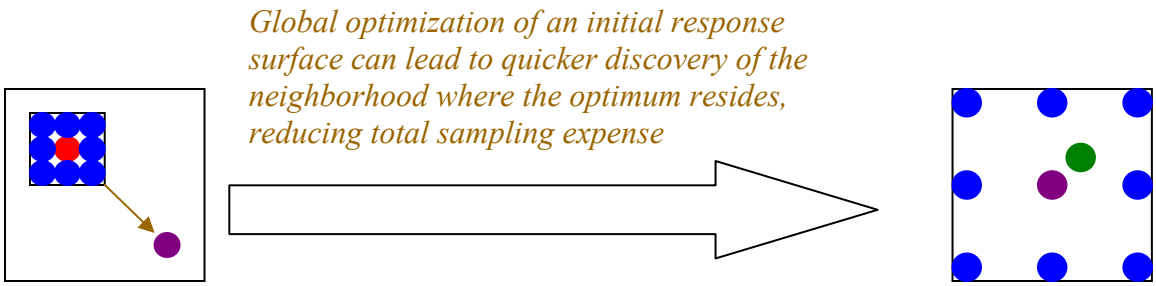


Figure 2.17. Application of the RSM-G algorithm for finding a system optimum.

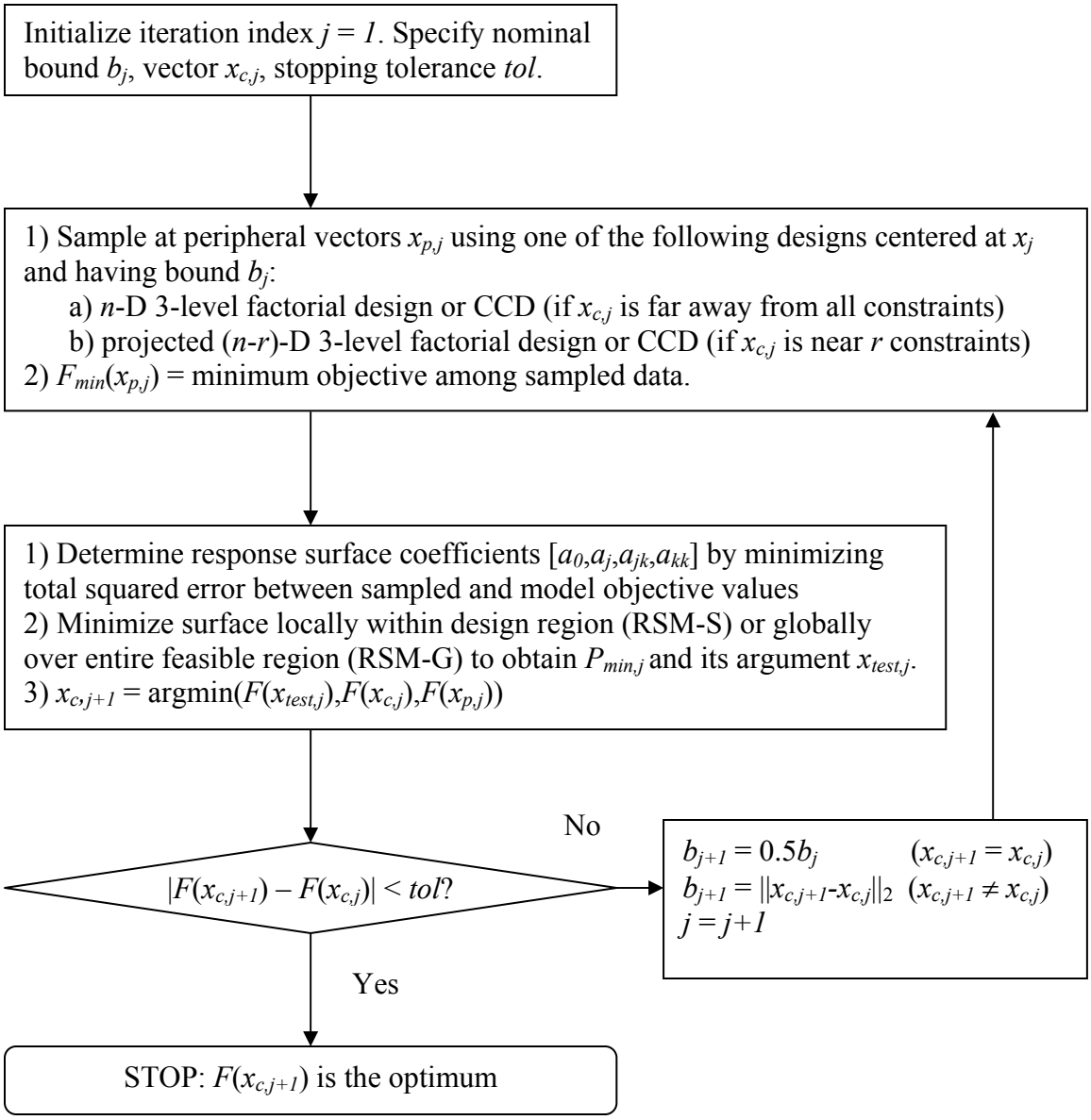


Figure 2.18. Flowchart of the RSM-S and RSM-G algorithms.

2.3 Kinetics Case Study

In this section, the performance the DS-RSM, RSM-S, and RSM-G algorithms is evaluated based on the application of each method to a reaction engineering case study[17,18]. The goal is to identify the set of input species concentrations which minimize an objective F, given by Equation (2.9), that is a nonlinear function of the corresponding

output variable concentrations. The performance of the response surface algorithms as a class is also evaluated against other optimization methods mentioned in the Literature Review (Section 2.1.2).

The problem is to determine optimal CSTR reaction conditions when closed-form rate equations are unavailable. A seven-reaction network involves five species A, B, C, D, and E, whose kinetic behavior adheres to a modification of the Fuguitt and Hawkins mechanism. The reactions are as follows: A $\rightarrow$ E, A $\rightarrow$ D, B $\leftrightarrow$ D, C $\leftrightarrow$ 2D, and 2A $\rightarrow$ C. The rate constants are k_1^f = 3.33384 s^{-1}, k_2^f = 0.26687 s^{-1}, k_3^f = 0.29425 s^{-1}, k_3^r = 0.14940 s^{-1} , k_4^f = 0.011932 s^{-1}, k_4^r = 1.8957e-3 m^3/(mol s), k_5^f = 9.598e-6 m^3/(mol s). The reactor volume V is 0.1 m^3, the total input flowrate F_1 is 0.008 m^3/s, and only species A and C enter the reactor. The concentration ranges for A and C are specified as 3e3 < C_A^0 [mol/m^3] < 3e4 and 0 < C_C^0 [mol/m^3] < 1e4.

If the mathematical form of the rate equations is assumed to be unknown, the dynamic behavior can be described using black-box models as given by Equation (2.5):

$$\frac{dC_i}{dt} = \Gamma\left(k_1^f, k_2^f, k_3^f, k_3^r, k_4^f, k_4^r, k_5^f, C_A^0, C_C^0, F_1, V, t\right) \qquad i = A, B, C, D, E \tag{2.5}$$

where the LHS describes the rate at which species i changes with respect to time. The variables k_j^f and k_j^r are rate constants for the j^{th} forward and reverse reaction, respectively. C_A^0 and C_C^0 denote initial species concentrations of A and C, F_1 is the total mass flow to the reactor, V is the reactor volume, and t is time. For completeness, the true rate equations are given by (2.6), even though they are not directly employed in the optimization.

$$\frac{dC_A}{dt} = \frac{F_1}{V}\left(C_A^0 - C_A\right) - k_1^f C_A - k_2^f C_A - k_5^f C_A^2 \tag{2.6a}$$

$$\frac{dC_B}{dt} = \frac{F_1}{V}(-C_B) - k_3^f C_B + k_3^r C_D \tag{2.6b}$$

$$\frac{dC_C}{dt} = \frac{F_1}{V}\left(C_C^0 - C_C\right) - k_4^f C_C + 0.5k_4^r C_D^2 + 0.5k_5^f C_A^2 \tag{2.6c}$$

$$\frac{dC_D}{dt} = \frac{F_1}{V}(-C_D) + k_2^f C_A + k_3^f C_B - k_3^r C_D + 2k_4^r C_C - k_4^r C_D^2 \tag{2.6d}$$

$$\frac{dC_E}{dt} = \frac{F_1}{V}(-C_E) + k_1^f C_A \tag{2.6e}$$

The NLP is given by problem (2.7) and the objective function F depends upon the steady-state concentrations C_C^{SS} and C_D^{SS} attained for species C and D.

$$\begin{aligned}
min\ & F = 4\left(X - 0.6\right)^2 + 4\left(Y - 0.4\right)^2 + sin^3\left(\pi X\right) + 0.4 \\
s.t.\ & X = 0.1428C_C^{SS} - 0.357C_D^{SS} \\
& Y = -0.1428C_C^{SS} + 2.857C_D^{SS} + 1.0 \\
& C_C^{SS} = \Gamma\left(C_A^0, C_C^0, F_1, V, t\right) \\
& C_D^{SS} = \Gamma\left(C_A^0, C_C^0, F_1, V, t\right) \\
& F_1 = 8\,L\,s^{-1} \\
& V = 100\,L \\
& 3 \le C_A^0 \le 30 \\
& 0 \le C_C^0 \le 10
\end{aligned} \tag{2.7}$$

In keeping with the formulation as presented in (2.1), the optimization problem can be recast using the vectors of continuous and output variables x, z_1, and z_2 as shown in problem (2.8) and the equations given in (2.9):

$$
\begin{aligned}
&min\ F = 4\left(z_{1,1} - 0.6\right)^2 + 4\left(z_{1,2} - 0.4\right)^2 + sin^3\left(\left(\pi\right)\left(z_{1,1}\right)\right) + 0.4 \\
&s.t.\ z_{1,1} = 0.1428z_{2,3} - 0.357z_{2,4} \\
&\quad z_{1,2} = -0.1428z_{2,3} + 2.857z_{2,4} + 1.0 \\
&\quad z_{2,8} = \Gamma\left(x_1, x_2, x_3, x_4, x_5\right) \\
&\quad z_{2,9} = \Gamma\left(x_1, x_2, x_3, x_4, x_5\right) \\
&\quad x_1 = 8 \\
&\quad x_2 = 100 \\
&\quad 3 \le x_3 \le 30 \\
&\quad 0 \le x_4 \le 10
\end{aligned}
\tag{2.8}
$$

where x, z_1, and z_2 are defined as follows:

$$\left(x_1, x_2, x_3, x_4, x_5\right) = \left(F_1, V, C_A^0, C_C^0, t\right) \tag{2.9a}$$

$$\left(z_{1,1}, z_{1,2}\right) = \left(X, Y\right) \tag{2.9b}$$

$$\left(z_{2,1}, z_{2,2}, z_{2,3}, z_{2,4}, z_{2,5}\right) = \left(\frac{dC_A}{dt}, \frac{dC_B}{dt}, \frac{dC_C}{dt}, \frac{dC_D}{dt}, \frac{dC_E}{dt}\right) \tag{2.9c}$$

$$\left(z_{2,6}, z_{2,7}, z_{2,8}, z_{2,9}, z_{2,10}\right) = \left(C_A^{SS}, C_B^{SS}, C_C^{SS}, C_D^{SS}, C_E^{SS}\right) \tag{2.9d}$$

$$x = \left[x_1, x_2, x_3, x_4, x_5\right] \tag{2.9e}$$

$$z_1 = \left[z_{1,1}, z_{1,2}\right] \tag{2.9f}$$

$$z_2 = \left[z_{2,1}, z_{2,2}, z_{2,3}, z_{2,4}, z_{2,5}, z_{2,6}, z_{2,7}, z_{2,8}, z_{2,9}, z_{2,10}\right] \tag{2.9g}$$

A contour plot of the objective as a function of the input variables C_A^0 and C_C^0 is shown in Figure 2.19.

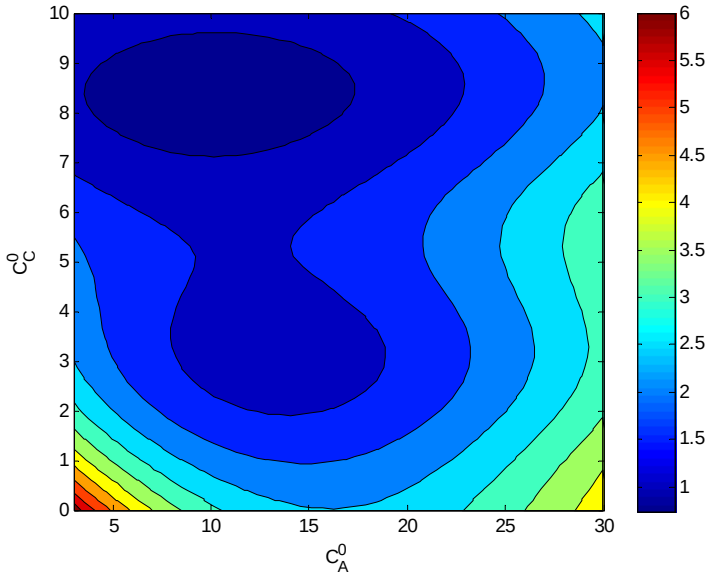


Figure 2.19. Contour plot of the objective function given in Problem (2.7).

The deterministic solution set consists of a global optimum of F = 0.7422 at $[C_A^0, C_C^0]$ = [10.117, 8.378] and a local optimum of F = 1.2364 at [13.202,3.163]. In order to introduce black-box complications, the rate equations are assumed to be unknown, so a microscopic model is used instead, represented using a lattice containing P_{tot} particles. The microscopic model is generated by first expressing the fraction of each species present in terms of a given number of molecular particles, which is the corresponding fraction of P_{tot} total particles.

The microscopic system is evolved using the Gillespie algorithm[19], and the number of molecular species particles is then converted back to the corresponding macroscale variable species concentrations. The Gillespie algorithm is a method for evolving reaction networks based on the assignation of an event probability to each reaction and choosing one to occur by generating a random number. After the number of molecular variables for each species has been updated, another random number is obtained and the corresponding

reaction is selected. This procedure continues occurring until little change is observed in the molecular variables. The noise in the concentration arises as a function of how coarse the microscopic model is, or how low the value of P_{tot} is. Steady-state solution vectors are obtained from an initial point by running the microscale simulations for a long time horizon, after which the objective function can be evaluated. The variance of the microscale system error is evaluated as:

$$\sigma^2 = Var(\varepsilon) = Var\left\{F^i(C_A^0, C_C^0, P_{tot}) \mid i = 1...k\right\} \tag{2.10}$$

In the next section, a detailed methodology of the Gillespie algorithm is presented.

2.3.1 Gillespie Algorithm for Microscopic Model Evolution

Based on the optimization formulation given in problem (2.7), steady-state species concentrations C_C^{SS} and C_D^{SS} are required in order to solve the optimization problem. In the absence of rate equations, the Gillespie algorithm[19] can alternatively be employed to obtain C_C^{SS} and C_D^{SS}. The Gillespie algorithm is a stochastic population-based method used to simulate the dynamic behavior of reaction networks at a microscale level, based on an initial number of particles P_{tot} present.

Even when a set of deterministic rate equations are available, the Gillespie algorithm provides an attractive computational alternative for obtaining a dynamic profile. The reason for this is that a stochastic simulation of a kinetics system can avoid the numerical intractabilities associated with solving a system of many coupled differential equations since numerical instabilities can arise through the application of both numerical and analytical techniques. The five-species, seven-reaction system is selected as a

simplification of a problem in which there are, say, a thousand reactions and the system is effectively black-box since a numerical solution can be computationally intractable.

The Gillespie algorithm proceeds as follows. Species concentrations C_i at the macroscale level are expressed in terms of a corresponding number of particles P_i at the microscale level according to a weighted percentage of P_{tot}. The idea is to evolve the system to steady-state by selecting a single reaction to occur at each time step t_j for a long time horizon t_f, as given by Equation (2.11), or until there is very little change observed in the species particle population counts P_i for a time step series.

$$t_j = \frac{j}{P_{tot}} \quad , \quad j = 1 \ldots t_f P_{tot} \tag{2.11}$$

At steady-state, the particle equivalents are then mapped back to the corresponding macroscale species concentration values. The conversion between concentrations C_A^0 and C_C^0 and particles is accomplished through simple weighting as shown in Equation (2.12):

$$P_i = P_{tot} \frac{C_i^0}{\sum_{i=A}^{E} C_i^0} \qquad i = A, B, C, D, E \tag{2.12}$$

where P_{tot} represents the total particle population, P_j represents the number of particles for species i, and C_i^0 represents the species concentration i at the macroscale level, in terms of mol/L. Once C_C^{SS} and C_D^{SS} have been determined, the optimization problem can now be solved.

Reaction networks are evolved based on the assignment of an event probability to each reaction and choosing one to occur based on the generation of a random number. At each time step, the kinetic system evolves whereby one of the seven reactions occurs probabilistically as a function of: (1) how many particles are present for reaction i and (2)

its respective rate constant. After the species particle count has been updated, another random number is obtained and the corresponding reaction is selected. This procedure continues occurring until little change is observed in the molecular variables P_i. Each time a steady-state vector is obtained, the microscale outputs are then mapped back to macroscale concentrations as shown in Equation (2.13), a reciprocal form of Equation (2.12), from which the objective function F can subsequently be evaluated.

$$C_i^{SS} = \frac{P_i}{P_{tot}} \sum_{i=A}^{E} C_i^0 \qquad , \qquad i = A, B, C, D, E \tag{2.13}$$

As the value of P_{tot} increases, the length of the time step decreases, and the CPU time required for evolution of the reaction system to steady-state increases. As P_{tot} approaches infinity, and the time interval approaches zero, the microscopic model behavior approaches the continuous behavior described by the macroscopic rate equations, though at a higher computational cost when compared to model evolution based on a lower value for P_{tot}. For repeated simulations of the microscopic model, different steady-state vectors will be generated due to the stochastic nature of the algorithm. As P_{tot} decreases, the variance of the set of these steady-state vectors increases since the microscopic system evolution is less accurately represented due to the kinetic system evolving over longer time steps. When the value of P_{tot} is specified as one million, the standard deviation of the objective value obtained from the set of one hundred stochastic simulations evolved from identical starting concentrations $[C_A^0, C_C^0]$ is 0.004. As expected, at high values of P_{tot} the steady-state behavior obtained from population-based stochastic simulation approaches that of the deterministic steady-state behavior obtained from solution of the rate equations given by the equations in (2.6).

The Gillespie algorithm is therefore an effective tool for obtaining input-output information for response surface modeling. The input data consist of initial concentrations C_A^0 and C_C^0. The output data consist of steady-state species concentrations from which F can then be evaluated via the intermediate equations for X and Y. The response surface is built with respect to F as a function of the inputs $[C_A^0, C_C^0]$, since it is F that is being minimized, not C_C^{SS} or C_D^{SS}. Once the Gillespie algorithm has been employed at each of the sampling vectors required for model construction, the response surface can be built. Local or global response surface minimization results in the attainment of a model-optimal value F.

The noise, quantified in terms of the standard deviation σ around the objective F for the R replicate simulations, decreases for increasing P_{tot}. The value of σ is used as the stopping tolerance *tol* for the algorithms presented in Figures 2.16 or 2.18. If the value of *tol* is lower than σ, it is possible for premature termination to occur in that a solution vector will be attained that is far from the true optimum. At the same time, the imposition of a strict criterion can lead to an increased sampling expense if the optimization of sequential response surfaces results in minimal improvement in the objective F.

2.3.2 Computational Results

The NLP formulated in problem (2.7) is now solved by employing the DS-RSM, RSM-S, and RSM-G algorithms. The best solution F is attained after sequential optimization of response surfaces for twenty-five trials each of varying microscale system size P_{tot}. For each trial, the nominal solution x_c, or, "initial best guess" is randomly selected in order to demonstrate successful convergence to an optimum

regardless of where the starting vector is located. Alternatively, this experiment is performed in order to show that the response surface method is robust against even a poor initial guess for the optimal vector (C_A^0, C_C^0). The initial radius for the 2-level, and later 3-level factorial design, is set as 0.1. A sample collocation set is presented in Figure 2.20 for a nominal iterate $x_c = [0.7,0.7]$.

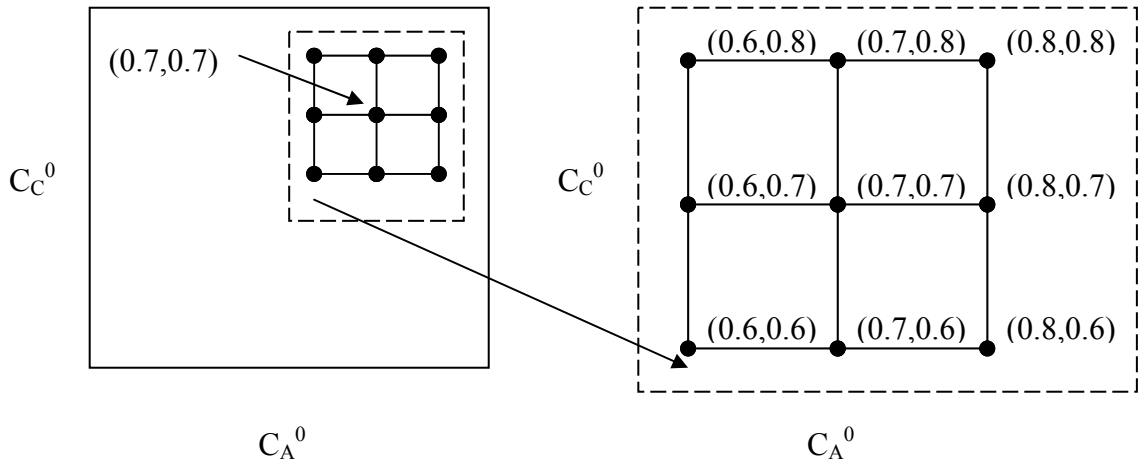


Figure 2.20. Generation of sampling vectors for response surface modeling based on a factorial design template centered at $x_c = [0.7,0.7]$ and having initial bounds $b_1 = 0.1$.

The solution for each of the computational trials is comprised of the optimal process conditions $(C_A^0, C_C^0)^{opt}$ and its corresponding objective function F. Results for the set of trials in which the global optimum is attained are presented in Tables 2.1 and 2.2[1].

Table 2.1: Global solutions obtained based on application of the DS-RSM algorithm.

Microscale model size	Design radius for initial response surface	C_A^0		C_C^0		CPU time (s)	
P_{tot}	b_1	μ	σ	μ	σ	μ	σ
10,000	0.1	9.977	0.703	8.37	0.106	66	20
10,000	0.15	10.354	0.637	8.38	0.124	45	11
10,000	0.2	9.956	0.72	8.368	0.108	44	8
10,000	0.25	10.171	0.743	8.348	0.12	44	7
50,000	0.1	10.27	0.532	8.379	0.068	290	77
50,000	0.15	10.089	0.597	8.365	0.063	254	63
50,000	0.2	10.12	0.527	8.347	0.07	236	46
50,000	0.25	10.205	0.484	8.382	0.096	201	36
100,000	0.1	10.257	0.477	8.375	0.039	494	141
100,000	0.15	10.225	0.466	8.38	0.047	407	96
100,000	0.2	10.207	0.416	8.353	0.057	345	77
100,000	0.25	10.209	0.507	8.378	0.077	337	66

Table 2.2: Global solutions obtained based on application of the RSM-G algorithm.

Microscale model size	Design radius for initial response surface	C_A^0		C_C^0		CPU time (s)	
P_{tot}	b_1	μ	σ	μ	σ	μ	σ
10,000	0.1	10.131	0.787	8.395	0.128	94	30
10,000	0.15	10.145	0.802	8.36	0.137	60	16
10,000	0.2	10.156	0.571	8.347	0.128	74	21
10,000	0.25	10.291	0.656	8.354	0.125	77	17
50,000	0.1	10.172	0.643	8.398	0.129	290	79
50,000	0.15	10.184	0.705	8.358	0.072	264	61
50,000	0.2	10.186	0.485	8.359	0.081	261	61
50,000	0.25	10.275	0.482	8.34	0.102	277	66
100,000	0.1	10.203	0.559	8.388	0.085	624	190
100,000	0.15	10.173	0.432	8.355	0.059	496	123
100,000	0.2	10.284	0.353	8.332	0.066	496	112
100,000	0.25	10.248	0.475	8.337	0.067	529	132

For each (P_{tot},b_1) combination, where b_1 designates the experimental design radius corresponding to the initial response surface, the same set of 100 random starting points is used. Data represent averages of the subset of experiments converging to F = 0.7422. Similar results are obtained for convergence to the local optimum. As the initial size of the design region in both RSM-based algorithms increases, the accuracy of the optimum value achieved generally decreases. The reason for this is that if the vector at which refinement of the best solution is located just inside the basin of the true optimum, shrinkage of initially large experimental design regions may still result in determining additional points outside this region.

Sequential quadratic programming using NPSOL is used to generate the solution of problem (2.7) when the microscale system size P_{tot} is adaptively refined during the course of optimization in order to reduce the computational expense during the early stages[18]. Results are given in Table 2.4. Table 2.5 provides additional results based on the performance of four other optimization algorithms that are employed to solve problem (2.7).

Table 2.3. Optimization results obtained for Problem (2.7), based on application of an adaptive gradient-based NLP algorithm.

Solution Method	P_{tot}	$(C_A^0,C_C^0)^{opt}$		F	CPU Time (s)	# of calls to microscale simulator
		μ	σ			
Standard	10^6	(10.167,8.426)	(0.2,0.05)	0.7425	2346	62
Adaptive	10^4-10^6	(10.166,8.475)	(0.21,0.04)	0.744	1052	96

For the adaptive method, 75% of the overall CPU time is spent making function calls to the simulator. Each function call requires a unique reaction network evolution of the micrsocale model which is computationally intensive as the value of P_{tot} increases. For the results obtained using the DS-RSM and RSM-G response surface methods, presented in Tables 2.4 and 2.5, the corresponding percentage of CPU time spent making function calls to the microscale simulator is above 85% since the fitting and optimization of the quadratic model are inexpensive operations compared to the evolution of the reaction network. The application of the adaptive algorithm results in a substantial reduction in CPU time compared to when the system size is fixed at a high resolution value of 10^6 total particles, yet also requires a larger number of function calls. Since the value of N is initially lower for the adaptive algorithm, more iterations are required before the optimum region is reached. The standard deviation of the solution for ten replicate runs is comparable.

Table 2.4. Performance of various optimization algorithms in finding the global optimum to problem (2.7) for a microscale system size of P_{tot} = 50,000.

Method	P_{tot}	F	$(C_A^0, C_C^0)^{opt}$	CPU Time (s)	# calls to microscale simulator
Nelder-Mead	50,000	0.7432	(9.505,8.460)	794	56
MCS	50,000	0.7422	(10.482,8.367)	1879	200
DS-RSM	50,000	0.7443	(10.243,8.376)	208	26
RSM-G	50,000	0.7447	(10.113,8.367)	333	36

Table 2.5. Performance of various optimization algorithms in finding the global optimum to problem (2.7) for a microscale system size of P_{tot} = 100,000.

Method	P_{tot}	F	$(C_A^0, C_C^0)^{opt}$	CPU Time (s)	# calls to microscale simulator
Nelder-Mead	100,000	0.7424	(10.449,8.382)	1948	67
Hooke-Jeeves	100,000	0.7422	(10.054,8.339)	2671	150
SQP	100,000	0.7471	(10.327,8.478)	1321	69
MCS	100,000	0.7423	(10.359,8.399)	3369	201
DS-RSM	100,000	0.7442	(10.203,8.378)	443	28
RSM-G	100,000	0.7442	(10.194,8.360)	512	35

For a microscale system size of P_{tot} = 50,000, the response surface algorithms require approximately half the number of function calls required by Nelder-Mead, a purely direct search method. Because a combination of direct search and gradient-based optimization are employed by the DS-RSM algorithm, it can be considered as a hybrid between Nelder-Mead and the standard RSM-S technique. The solution attained by MCS is identical to the deterministic solution; however, it is attained at significantly higher computational expense relative to the response surface methods and Nelder-Mead. Even though the objective values obtained by the non-MCS methods are slightly inferior to the value of 0.7422 attained using MCS, the associated computational expense is an order of magnitude cheaper than that required by MCS. For a microscale system size of Ptot = 100,000, the response surface algorithms again require approximately half the number of function calls to the microscale simulator relative to the number required by its nearest competitors in the Nelder-Mead and Hooke-Jeeves methods.

2.4 Summary

The novel contribution of the work presented in this chapter has been the presentation of new experimental design templates employed within RSM optimization frameworks for the solution of NLP described by convex constraints. The new designs include techniques for 1) ensuring sampling set feasibility when iterates are located near boundaries, 2) generating sampling sets for lower-D response surfaces when equality constraints are present, and 3) generating sampling sets corresponding to response surfaces projected onto constraints to ensure that large steps remain being taken towards an optimum, rather than being limited in length by low iterate-constraint distances. A sequential direct search-model based response surface methodology and a global response surface methodology are presented as techniques which target sampling expense reduction in the search for an optimum. Both of the new response surface techniques are applied to a kinetics case study to illustrate proof of concept, and the problem's global optimum is attained at lower sampling expense relative the amount required when additional zero- and first-order algorithms are employed.

Chapter 3

Global Optimization Employing Kriging and Response Surface Models

Kriging predictors are data-driven models in which noisy input-output behavior is considered to be a natural system feature instead of a contaminant. Therefore, one main advantage of the method is that it specifically targets accurate model generation for noisy processes. Kriging was originally applied towards the identification of optimum drilling locations for mining applications and has been applied frequently for 3-D visualization in geostatistical applications[20]. Accurate kriging models can be constructed from dispersed sampling data at lower sampling expense relative to that required by an experimental design for response surface construction. The contribution of the work in this chapter is the development of a kriging-RSM algorithm targeted at the identification of global optima for problems of arbitrary dimension and whose feasible region is described by convex constraints[2]. Kriging is used to first generate a global model of input-output behavior over the entire feasible region in order to identify the best warm-start iterates for local optimization using RSM. The benefit attained from applying the unified kriging-RSM algorithm relative to the stand-alone RSM algorithms presented in Chapter 2 is that the chances of finding a problem's global optimum increase since the initial iterates selected for local optimization are chosen based on global model information rather than

simply being chosen by random selection. The performance of the kriging-RSM algorithm is compared to the stand-alone response surface techniques and its effectiveness is evaluated in terms of the number of function calls required, number of times the global optimum is found, and computational time required before the optimum is found.

3.1 Introduction

The global solution of process design problems lacking closed-form model equations is difficult to obtain when a local optimization technique such as RSM, whose methodology is presented in the previous chapter, is applied. The primary limitation of the response surface model employed in the previous chapter is that it generally describes system behavior accurately only in the vicinity of an optimum. The reason why model accuracy is limited to a specific subregion is due to the simplicity of its quadratic polynomial functionality, which describes output variable behavior as a quadratic function of the set of continuous input variables. Since optima may be located in remote parts of the feasible region, and because output variable behavior can be a highly nonlinear function of the input variables, a quadratic polynomial response surface functionality may fail to accurately describe system behavior over a majority of its feasible region space.

Kriging is a global modeling technique that can sufficiently model arbitrary nonlinear behavior; thereby the use of kriging models can overcome the modeling disadvantage of the quadratic response surface. However, the sampling expense associated with building accurate kriging models is generally higher than that required for local modeling since

the model is describing system behavior over the entire feasible region, which is usually more complex, in terms of mathematical geometry, in contrast to behavior described over a localized area. However, the additional sampling expense incurred from building global models is justified by the possibility of identifying promising warm-start iterates for local optimization, enabling more rapid determination of the complete set of refined local and global optima, in contrast to local models initialized at randomly determined iterates.

In order to overcome the problem of high sampling expense, the technique of building iteratively updated global models can be applied in which an initial kriging model is built using a low number of sampling points, and subsequent models are then constructed based on the incorporation of additional sampling information. This practice has the effect of minimizing resource costs attributed to sampling at locations where the contribution to model development is low. However, one disadvantage of applying kriging is that it may not be *a priori* known how many sampling points are needed for accurate model construction, or where their locations should be. A naïve technique is to build kriging models from sampling data obtained at a user-specified number of randomly chosen locations. In Chapter 6, a standardized algorithm is presented to overcome the uncertainty associated with random sampling for initial model construction.

In this chapter, a unified kriging-RSM algorithm is presented as a method for finding the solution of nonlinear programs (NLP) containing black-box functions and noisy variables. The kriging-RSM technique extends the capabilities of current methods to handle convex feasible regions defined by arbitrary linear and nonlinear constraints. In the proposed method, kriging is first used to construct iteratively refined global models in order to find the set of regions containing potential optima. Response surface techniques

are then applied to local models in order to refine the set of local solutions. The global model building expense is offset by an increased probability of finding the global optimum in contrast to the application of RSM alone.

3.1.1 Literature Review

A kriging predictor is a global model employing normally distributed basis functions, so both an expected sampling value and its variance are obtained for each test point[21-23]. Kriging was first developed as an inverse distance weighting method to describe the spatial distribution of mineral deposits. The global model that was obtained was used to determine additional locations possessing the same grade characteristics, thereby enabling the generation of improved monthly forecast estimates for different mine sections. The seminal article by Krige[20] was reviewed in context of other classical statistical methods used for geostatistics[24].

Kriging has not only been used to generate models based on field data, but also when input-output information is obtained via computer experiments due to the increasing use of simulation for the study of complex processes[25-26]. It is to be noted that in the proposed algorithm, the function of kriging is to determine improved locations for local search thereby enabling the response surface techniques to be applied to "warm-start" iterates. Due to the black-box functions present in the problem formulation, even though the global optimum is sought, it is impossible to theoretically guarantee global optimality. The confidence limits obtained using of kriging may be similar to those attained using αBB[28] but αBB assumes that the functionality is known which is not the case for the NLP class addressed in this chapter.

Kriging is an interpolation technique whereby a prediction $\tilde{z}_2$ at test point x_k is made according to a weighted sum of the observed function values at nearby sampling points. The kriging predictor is modeled after a normally distributed random function, so a prediction variance is also obtained at the test point. As a result, the sampling value is expected to fall within the interval specified by the prediction and corresponding variance. If the feasible region is discretized, both a prediction and variance mapping can be obtained over a test point set having uniform coverage. The variance mapping describes prediction uncertainty, which will be high in regions with a low number of sampling points. Kriging predictors can be improved by incorporating additional sampling information obtained within these regions, thereby reducing model uncertainty. The existing literature usually compares kriging and RSM methods but has not combined them together for the purpose of global optimization as presented in this chapter.

In the next section, the mathematical formulation of the problem is given. Following this, the basic steps of the kriging-RSM algorithm will be presented.

3.1.2 Problem Definition

The problem addressed in this work has the following form:

$$
\begin{aligned}
&\textit{min} \quad F(x, z_1, z_2) \\
&\textit{s.t.} \quad g(x, z_1, z_2) \le 0 \\
&\qquad h(x, z_1) = 0 \\
&\qquad z_2(x) = \Gamma(\mathrm{x}) + \varepsilon(x) \\
&\qquad \varepsilon(x) \in N\left(x \mid \mu, \sigma^2\right) \\
&\qquad N\left(x \middle| \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \\
&\qquad x \in \Re^n
\end{aligned}
\tag{3.1}
$$

The problem formulation is the same as that given by Equation (2.1); however, the kriging-RSM methodology presented in this chapter targets the attainment of a global optimal solution instead of an arbitrary local optimum found using RSM alone as presented in Chapter 2.

In this formulation, x represents continuous variables that are process inputs. This set of variables is distinct from the set of input variables z_2 whose values are obtained from sampling data obtained at upstream units based on a subset of the variables x. Deterministic output variables z_1 describe outputs whose modeling equations $h(x,z_1)$ are known. Stochastic output variables z_2 represent the black-box part of the model and are simulated by deterministic functions $\Gamma(x)$ perturbed by an additive noise term $\varepsilon(x)$. The noise is described by a normally distributed error having mean zero and variance σ^2 whose properties can change depending upon the spatial location of x. The noise function is stochastic in that replicate experiments performed under the same conditions lead to different values for $\varepsilon(x)$ and in turn $z_2(x)$. Design constraints such as operating specifications are given by $g(x,z_2)$.

3.2 Solution Approach

In the proposed approach, a sequential kriging-RSM algorithm is employed for the optimization of systems containing black-box functions and noisy variables. The basic idea is to first use kriging to obtain a global picture of system behavior by generating predictions at test points, and then to apply RSM to a set of regions containing potential local optima in order to find the global optimum. Although the kriging predictor is iteratively improved as additional sampling is conducted in regions where prediction

ability is poor, the CPU cost associated with generating many predictions over the course of the modeling can become high, especially if a set of discretized test points obtained using high-resolution grids are employed in order to obtain more accurate optima. Since response surfaces can be inexpensively generated, a better strategy for obtaining potential optima is to generate a kriging surface over a lower-resolution grid and then use sequential response surfaces to refine the best set of candidate vectors.

Since it is possible that local optima lie along the boundaries of the feasible region, the application of current response surface techniques which are suited for the solution of box-constrained problems would fail to improve upon the current values of the possible optima. In Chapter 2, these limitations have been overcome based on the application of 1) adaptive experimental designs to retain feasibility, and 2) projection of the *n*-dimensional response surface onto the feasibility space limited by the problem constraints. The incorporation of these techniques has extended the capabilities of RSM, enabling the solution of NLP to be obtained when the feasible region is described by an arbitrary set of convex constraints.

Since the kriging-RSM algorithm is targeted at finding both interior and boundary solutions for constrained NLP described by 1) black box functions, 2) noisy variables, and 3) arbitrary convex feasible regions, it can serve as a backbone algorithm for the solution of problems containing integer variable complexities whose corresponding algorithms rely on the solution of relaxed NLP subproblems. In Chapters 4 and 5, Branch-and-Bound and Direct Search techniques will be combined with the kriging-RSM algorithm to address this more difficult MINLP problem class. It should be emphasized that even though global optimality is not theoretically guaranteed when the kriging-RSM

method is applied, the chances of finding a vector close to the deterministic optimum is increased relative to when a local technique such as RSM is used by itself. The reason for this is due to the global search feature obtained from building kriging models over the entire feasible region. In the next section, the details of the kriging method will now be presented.

3.2.1 Kriging Methodology

In kriging, sampling data are treated as the realizations of a random function in order to improve the modeling of a black-box system assumed to be inherently stochastic. Since the kriging model is based on a random function, at each test point S_k, both a point value and variance estimate are obtained. The global model is built by mapping the kriging predictions with respect to the input variables. The corresponding variance mapping is primarily used to determine sampling locations for model refinement at high-variance regions. However, it can also be employed as an alternative measure of model reliability whereby further refinement is terminated once the maximum variance falls below a fraction of its initial value. Each kriging estimate at test point S_k is generated as a weighted sum of nearby sampled function values. The weights are generated as a function of the Euclidean distance between the sampling vectors close to S_k in a manner similar to inverse distance weighting methods, in which higher weighting is generally given to sampled function values whose vectors are close to the test point.

Kriging models can be generated with respect to the process output variables z_2 for the black-box units; however, the model that is directly used in the kriging-RSM algorithm is the one built with respect to the NLP objective. This is done since it is the

objective function which is being optimized instead of the process outputs. The weights are determined using a covariance function, which measures correlation strength between two objective values based on a Euclidean distance equation as given by Equation 3.2:

$$d_{i,j} = \sqrt{S_i(x) - S_j(x)} \tag{3.2}$$

where it should be noted that the complete representation of any sampling vector S_i is given as $S_i(x)$ if only continuous variables are present, and as the triplet $S_i(x,y_1,y_2)$ if any integer variables are also present. The integer variable classifications for y_1 and y_2 are provided in Chapter 5.1, and if any y_2 variables are relaxed and therefore can temporarily assume continuous values, the equation for $d_{i,j}$ is modified as given by Equation (3.3):

$$d_{i,j} = \sqrt{S_i(x, y_2) - S_j(x, y_2)} \tag{3.3}$$

The two vectors employed in Equation (3.2) can be either two sampling points $\{S_i,S_j\}$, or a sampling point and test point $\{S_i,S_k\}$. Although coefficient values must first be determined for the covariance function, a reliable covariance function can usually be built using limited sampling information, enabling a set of predictions, and, in turn, a reliable global model, to be generated at low cost. The basic steps for building the model are as follows: 1) determination of covariance function coefficients based on sampling data; 2) calculation of the covariance $Cov(d_{i,k})$ between the test point and each nearby sampling point; 3) generation of weighting values λ for each sampling point S_i close to S_k after solving the linear system $C\lambda = D$, where the elements of C and D are $\{S_i,S_j\}$ and $\{S_i,S_k\}$ covariance values, respectively; and 4) estimation of the kriging predictor as given by Equation (3.4).

$$F(S_k) = \sum_{i=1}^{k_{Cluster}} F(S_i)\lambda(S_i) \tag{3.4}$$

In (3.3), $F(S_k)$ represents the prediction value obtained at test point S_k, $k_{Cluster}$ denotes the number of nearby sampling points, and $F(S_i)$ denotes a corresponding sampled output value at S_i. The estimation of $F(S_k)$ is generally improved when the set of the Euclidean distances between the test point and its nearby sampling points are all different. Lower weighting generally occurs as the $d_{i,k}$ distance between S_i and S_k increases, a behavior similar to that observed with inverse distance methods.

The methodology for obtaining the covariance function coefficients is now presented. From the set of S_{Tot} sampling data, squared output value differences $F_{i,j}$ are calculated and plotted relative to sampling-pair distances as given by Equation (3.5):

$$F_{i,j} = \left[F(S_i) - F(S_j) \right]^2 \qquad i, j = 1 \ldots S_{Tot}, i \neq j \qquad (3.5)$$

From a scatter plot of $F_{i,j}$ as a function of $d_{i,j}$, a semivariance function is then fitted. Several standard semivariance models from the literature are typically tested in order to ascertain which one provides the best fit[20]. Due to the plot complexity as shown in Figure 3.1(a), the best fit to one of the established semivariance models in the literature is not usually immediately apparent.

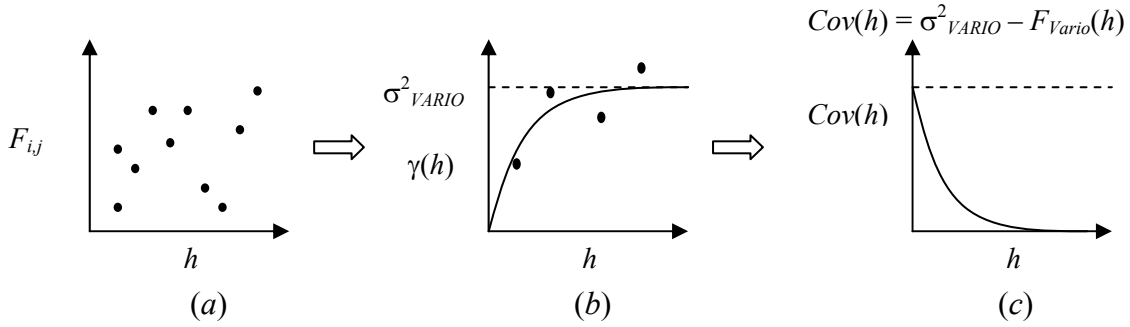


Figure 3.1. Data smoothing applied to squared function differences $F_{i,j}$ (a) in order to obtain a semivariogram fit (b) and covariance function fit (c).

To alleviate this problem, data smoothing is applied, and the semivariance function is then fitted to a reduced set of scatterpoints known as semivariances γ as shown in Figure 3.1(b). A set of P equidistant intervals are defined between zero and the maximum $d_{i,j}$ distance. The p^{th} interval midpoint is denoted by h_p, and the semivariance corresponding to the p^{th} interval, $\gamma(h_p)$, is obtained by averaging the set of squared function differences falling inside this interval as given by Equation (3.6):

$$\gamma(h_p) = \frac{1}{2N(h_p)} \sum_{r=1}^{N(h_p)} F_{i,j,r} \qquad , \qquad p = 1...P, \quad i \neq j \tag{3.6}$$

where $N(h_p)$ is the number of sampling pairs $\{S_i,S_j\}$ whose separation distance $d_{i,j}$ lies inside the p^{th} interval. The semivariance function behavior typically rises from zero to an asymptotic maximum known as the sill σ^2_{VARIO}. In order to generate the corresponding covariance function as displayed in Figure 3.1(c), the semivariance function is then reflected between the x-axis and the sill. Once the covariance function has been obtained, the covariance between any two sampling vectors can then be determined by substituting $d_{i,j}$ or $d_{i,k}$ into the model equation. The kriging weights are then obtained as the solution of a linear system of equations in which the LHS consists of a matrix of $\{S_i,S_j\}$ covariances, and the RHS consists of a vector of $\{S_i,S_k\}$ covariances. If the weights are forced to sum to unity, the linear system can be recast in a Lagrangian formulation as given by Equation (3.7):

$$\begin{bmatrix} \lambda(S_k) \\ \lambda'(S_k) \end{bmatrix} = \begin{bmatrix} Cov(d_{i,j}) & 1 \\ 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} Cov(d_{i,k}) \\ 1 \end{bmatrix} \quad i,j = 1...k_{Cluster}\,,\ i \neq j \tag{3.7}$$

where $\lambda(S_k)$ and $\lambda'(S_k)$ represent the weight vector and the Lagrange multiplier, respectively. Once the weights are obtained, the kriging prediction $F(S_k)$ and its expected variance $\sigma_k^2(S_k)$ are obtained according to Equations (3.4) and (3.8), respectively:

$$\sigma_k^2(S_k) = \sigma_{VARIO}^2 - \sum_{i=1}^{k_{Cluster}} \lambda(S_i)\, Cov(d_{ik}) - \lambda'(S_k) \tag{3.8}$$

The methodology is then applied to another test point, and once all k_{Test} kriging predictions have been obtained, the global mapping can be constructed. If additional sampling is performed, a new covariance function can be generated. Based on the updated covariance function, new kriging estimates can be generated for all k_{Test} sampling points and a refined global model can be created. For each global model, its corresponding average predictor value μ is compared against its counterpart based on the previous model. Once convergence has been achieved in μ, further refinement is terminated. Let the iteration index m refer to any property based on the m^{th} kriging model, and let Tol_{Krig} be a percentage stopping tolerance. A sample range for Tol_{Krig} would be any value between one and ten percent. The m^{th} average prediction value μ_m is defined as the average of the set of kriging predictions and sampled function values as given by Equation (3.9):

$$\mu_m = \frac{1}{\left(S^m + k_{Test,m}\right)}\left(\sum_{i=1}^{S^m} F_i(S_i) + \sum_{r=1}^{k_{Test,m}} F_r(S_k)\right) \tag{3.9}$$

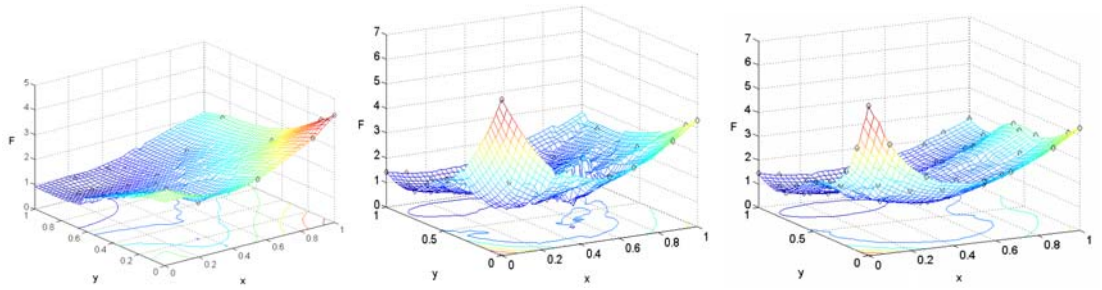
where S^m and $k_{Test,m}$ refer to the number of sampled function values and test points employed in constructing the m^{th} global model. Based on this notation, S^1 denotes the nominal sampling set used to create the first global model. The nominal value of μ_0 is obtained by averaging the sampled function values from S^1. Once the m^{th} global model

has been constructed, μ_m/μ_{m-1} is evaluated. If this ratio falls inside the interval $(1\pm Tol_{Krig})$, the m^{th} global model is considered accurate, and no additional updating occurs.

If, on the other hand, μ_m/μ_{m-1} falls outside $(1\pm Tol_{Krig})$, another model is built using additional sampling data. To increase model reliability, sampling is performed in regions of high uncertainty characterized by either high prediction variances, and at points whose kriging prediction value has changed the most between iterations, relative to the prediction value changes corresponding to the remaining set of k_{Test} points[16]. In order to emphasize global model improvement, an additional criterion is enforced whereby all new sampling points reside some minimum distance apart from one another. This practice ensures that the new sampling set will not consist of clustered points located at a single high-variance region, thereby de-emphasizing local model refinement. It should again be noted that the corresponding mapping of the sampling data with respect to the process output z_2, rather than the partially relaxed NLP objective, can be obtained by substitution of z_2 in place of F for the equations given in (3.4), (3.5), (3.6), and (3.9).

Local optimization using response surfaces is performed after global model reliability has been confirmed. However, the number of sampling points needed for local model construction can become high if the black-box process behavior is described using more than five variables. To alleviate this problem, each new sampling set used for global model refinement also includes vectors whose kriging predictors yield the best objective values for the current iteration. At regions where locally optimal kriging predictions are obtained, refined grids are generated and the corresponding set of new vectors is added to the set of current test points. A set of global models is presented in Figure 3.2 in order to

illustrate the stabilization that occurs after the kriging predictor has been improved using the current sampling rules for model refinement.



(a) (b) (c)

Figure 3.2. Kriging model generated at initial (a), intermediate (b), and final (c) stages.

The procedure for obtaining a prediction at S_k, referred to as the kriging algorithm, can be summarized as follows. First, the feasible region is characterized and the iteration index m is initialized at unity. If the black-box model represents an intermediate process within a process train, output sampling data z_2 obtained for some upstream processes may be required in order to fully define the feasibility constraints $g(x,z_2)$ and $h(x,z_1,)$, or alternatively the corresponding feasibility constraints $g(\bullet)$ and $h(\bullet)$ presented in Chapters 4 – 6, as appropriate, if the problem formulation also contains integer variables. A nominal sampling set S^m is specified which contains as few as ten points even when the MINLP is described by as many as forty input variables. As long as the starting size of S^m is not too small, the number of iterations required to achieve convergence in μ_m will be relatively insensitive to the number of sampling vectors in the nominal sampling set. However, the initial number of sampling points which comprise S^m should be kept low in order to place emphasis on further sampling as needed during the iterative stages of predictor refinement. Semivariances are then generated using all sampling data within

S^m. The best semivariance model is fitted using least squares, and the complementary covariance function is then obtained. The matrices on the RHS of Equation (3.7) are then constructed from submatrices $h_{Cluster}$, h_0, C, and D, as given by Equation (3.10). The matrices C and D are augmented in order to remain consistent with the Lagrangian formulation given in Equation (3.7), in which the weights are required to sum to unity.

$$h_{Cluster} = \begin{bmatrix} 0 & d_{1,2} & \cdots & d_{1,k_{Cluster}} \\ d_{2,1} & 0 & \cdots & d_{2,k_{Cluster}} \\ \vdots & \vdots & \ddots & \vdots \\ d_{k_{Cluster,1}} & d_{k_{Cluster,2}} & \cdots & 0 \end{bmatrix} = \left[d_{i,j} \right] \qquad i, j = 1 \ldots k_{Cluster} \tag{3.10a}$$

$$h_0 = \begin{bmatrix} d_{1,k} \\ d_{2,k} \\ \vdots \\ d_{k_{Cluster},k} \end{bmatrix} = \left[d_{i,k} \right] \qquad i = 1 \ldots k_{Cluster} \tag{3.10b}$$

$$C = \begin{bmatrix} Cov(h_{Cluster}) & 1 \\ 1 & 0 \end{bmatrix} \tag{3.10c}$$

$$D = \begin{bmatrix} Cov(h_0) \\ 1 \end{bmatrix} \tag{3.10d}$$

The location S_k is specified and $k_{Cluster}$ nearest-neighbor sampling points are chosen from S^m that are nearest to S_k as given by Equations (3.2) or (3.3), depending upon whether any integer variables y_2 exist and are relaxed in the corresponding relaxed NLP subproblem. The value of $k_{Cluster}$ usually varies between five and ten regardless of problem dimension, although the estimate of $F(S_k)$ may be skewed if sampling information is too sparse. The kriging weights λ are then obtained from solving the linear

system of equations as presented in Equation (3.7) and the prediction $F(S_k)$ and its variance $\sigma_k^2(S_k)$ are determined using Equations (3.4) and (3.8), respectively.

The weights are then recalculated for each of the k_{Test} sampling vectors in order to generate corresponding estimates for $F(S_k)$. Once the global mapping has been constructed μ_m is determined from Equation (3.9) and compared against μ_{m-1}. If convergence is not achieved, the iteration index m is advanced by unity and additional sampling is performed based on application of the sampling rules. A new covariance function is built, new kriging estimates $F(S_k)$ are generated, and an updated mapping is built. The procedure is terminated once convergence has been achieved in μ_m. The best local solutions are then identified for sequential local optimization using RSM. RSM targets the optimization of x, or, if appropriate, x and y_2, as described in Chapter 5.3. A flowchart of the kriging algorithm is presented in Figure 3.3.

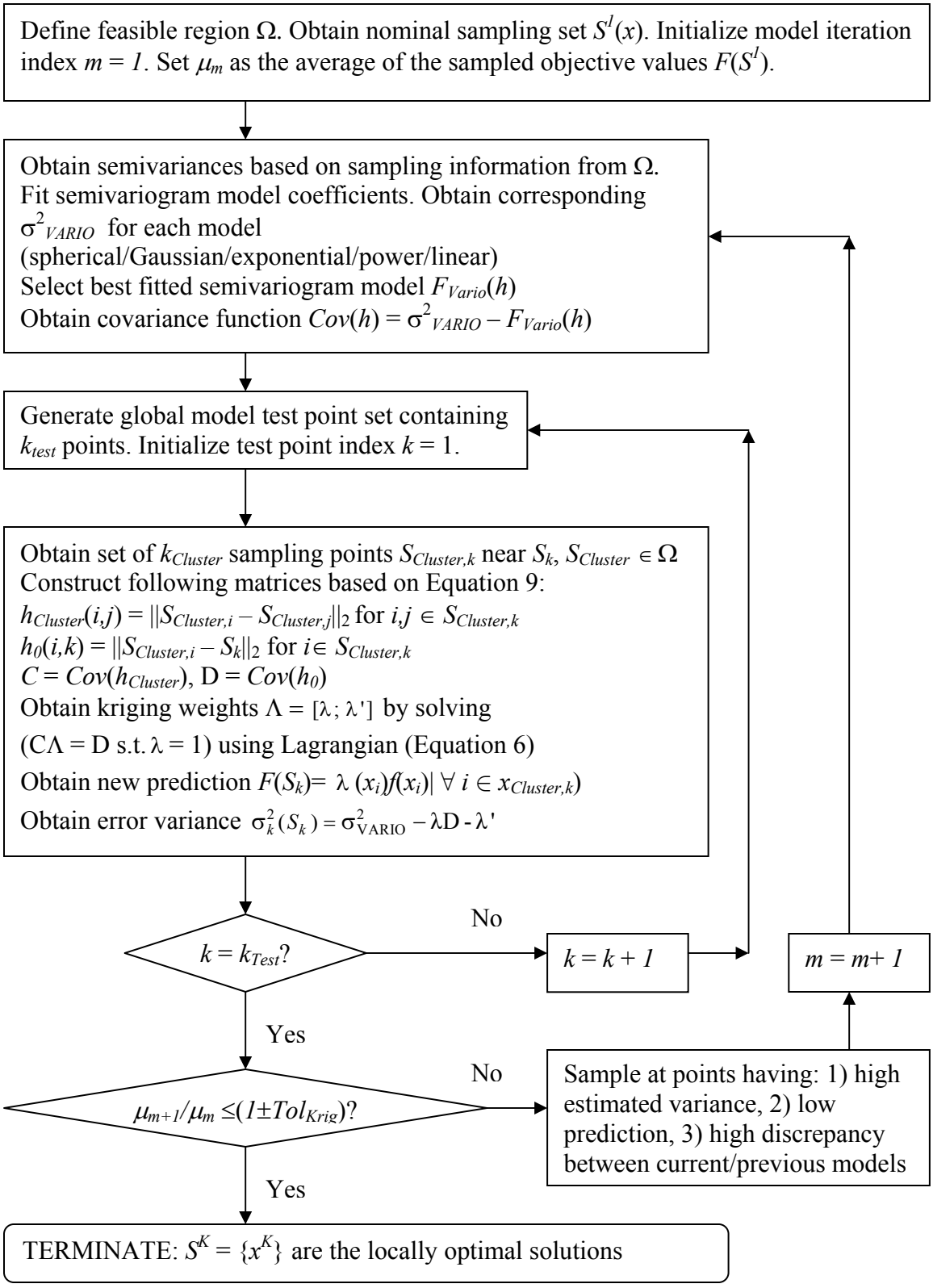


Figure 3.3. Kriging algorithm flowchart for building/refining a data-driven global model.

3.2.2 Kriging-RSM Algorithm

In this section, the details of the kriging-RSM algorithm are presented. First, a set of nominal sampling information is obtained and a set of feasible test points is generated over which to build the kriging predictor. As mentioned before, it is important that the number of test points not be too high as model building costs will increase. Discretization can be used to generate the set of test points for 2- or 3- dimensional problems, but the number may become too high for problems of higher dimension. For the presented examples in which the problem dimensionality was greater than three, it is found that one thousand randomly generated points are sufficient for building the kriging predictor without substantial increase in the model building costs.

A test point x_k is selected and both its kriging prediction and variance are obtained according to the kriging algorithm described in Figure 3.3. Once all kriging predictions have been obtained for the set of k_{Test} points, the average value of the kriging predictor is then obtained and compared to the corresponding value obtained in the previous iteration. If the difference between these values exceeds a stopping tolerance Tol_{Krig}, a set of additional candidate vectors is obtained whose corresponding kriging variances are the highest for their respective local region. The parameter Tol_{Krig} is set at 0.01 and is a generally noise-independent parameter as the number of test points used to build the kriging mapping exceeds the number of points at which sampling data are obtained.

Due to the fact that black-box functions prevent knowledge of NLP problem convexity as given by Equation (3.1), the global optimum may still be missed when kriging or any other global modeling technique is applied. In order to increase the chances of finding the subregion containing the global optimum, the current kriging

model is iteratively improved based on the information obtained from a new sampling set obtained for each iteration. The new sampling set is comprised of three subsets each having an equal user-determined number of data points. For the examples presented in this and the remaining chapters, this number is set at three. Each subset has a family of points conforming to one of the following characteristics: 1) high variance, 2) high difference between prediction estimates for consecutive iterations, and 3) minimum prediction values. In addition, the sampling locations within each family subset are separated by a minimum L^2-normed distance in order to maximize the amount of global information obtained. After performing additional sampling, the kriging predictor is then refined by generating new predictions again at all test points. Once the difference in the average kriging predictor falls below Tol_{Krig} for consecutive iterations, refinement of the current best candidate vectors yielding the lowest kriging predictions in M local regions occurs according to the RSM algorithm. The set of M kriging solutions S_m^K, $m = 1...M$, are then rearranged in order of increasing objective function value F; therefore S_1^K refers to the kriging solution having the lowest objective value. The iteration index m represents the m^{th} kriging solution to be locally optimized.

At the start of the RSM algorithm, the iteration index w is initialized at a value of unity. A response surface is built around a kriging-optimal solution S_m^K by fitting sampling data obtained from a collocation set $S_{coll,w}$. The vectors which comprise $S_{coll,w}$ are determined by applying either one of the factorial or CC design templates for a predetermined initial model radius b_w. For the examples presented in Section 6.3, the nominal value of b_w is set at ten percent of each variable's operability range as defined by the difference in corresponding lower and upper bounds. The vector S^K and its

corresponding objective value F^K comprise the nominal solution set $\{S_{opt,w},F_{opt,w}\}$. Once the response surface has been created, the optimum $S_{opt,w+1}$ having corresponding value $F_{opt,w+1}$ is determined using gradient methods. Sampling is performed at the model optimum vector in order to confirm objective value improvement. If the difference between the current and previous optimum $|F_{opt,w+1} - F_{opt,w}|$ falls below a prespecified criterion Tol_{RSM}, the algorithm terminates with $\{S_{opt,w+1,},F_{opt,w+1}\}$ established as the RSM solution. Otherwise, the iteration index is advanced by unity and another response surface having a new bound radius b_w is constructed at the new $S_{opt,w}$. At any iteration w, the value of b_{w+1} is different from b_w only if the Euclidean distance between the current and previous solution vectors is lower than the current radius b_w. During the later stages of the algorithm, $S_{opt,w+1}$ will be near $S_{opt,w}$, signifying that the basin of the RSM optimum has been found. At this point, a more accurate description of the system behavior near the optimum can be attained using more localized response surfaces. Whenever iterates are close to the boundaries, lower-dimensional response surfaces are created by projecting the model onto constraints so as to prevent model generation based on an asymmetrical arrangement of the feasible sampling data[12]. The RSM-optimal solution is denoted as $F(S_m^R)$. Once its value has been attained, the value of m is increased by unity and the next kriging solution S_m^K is locally optimized. A flowchart of the complete kriging-RSM algorithm is presented in Figure 3.4.

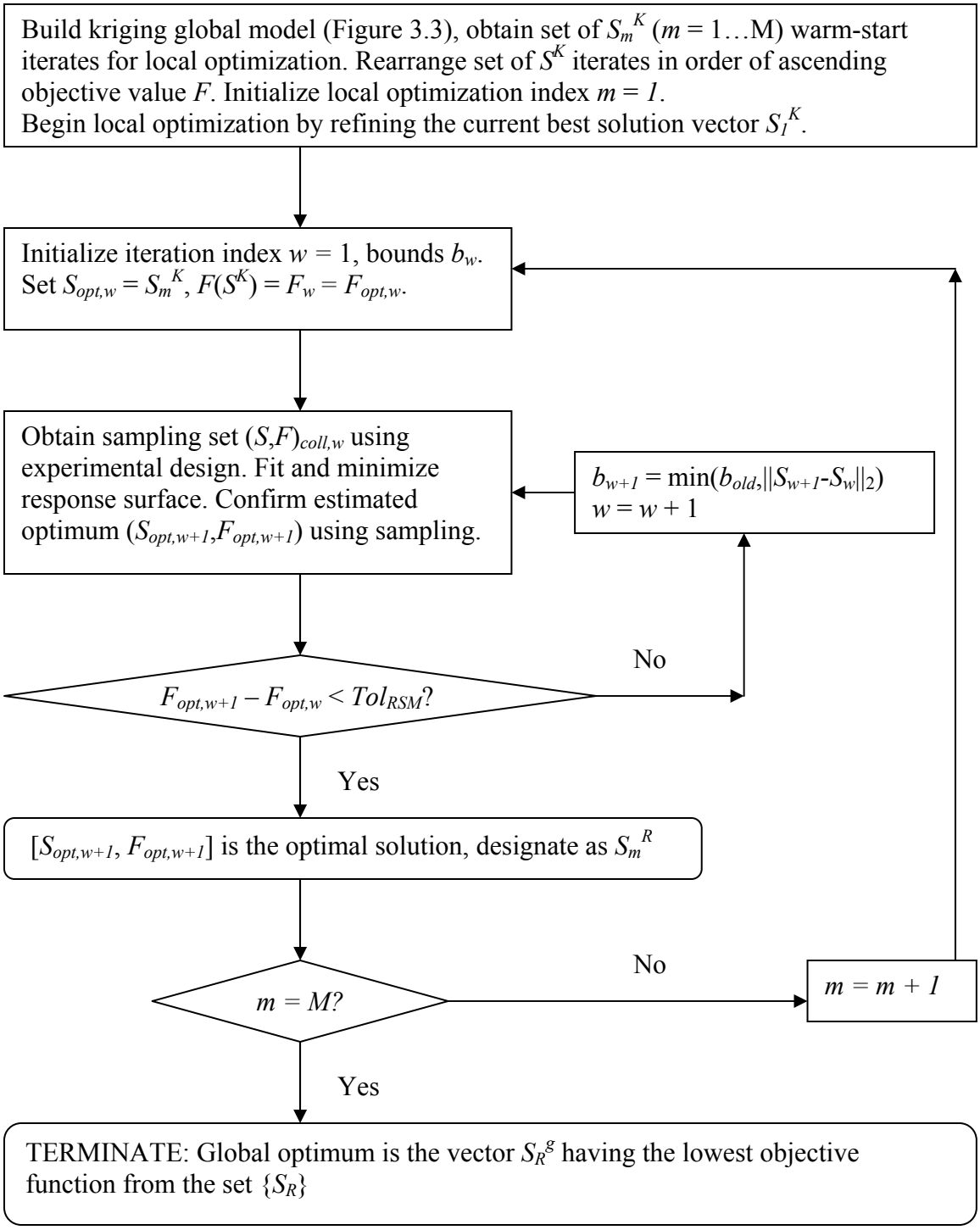


Figure 3.4. Flowchart of the Kriging-RSM algorithm.

For the presented examples, whenever an iterate approaches the feasible region boundary, the computational burden associated with identifying the corresponding constraints is low since the problem sizes are small. However, modifications to the proposed methodology may be necessary for high-dimensional problems containing many constraints and will be addressed in a future work. The kriging-RSM algorithm terminates after all or a subset of the kriging solutions have been refined using RSM, and the global optimum is identified as the vector having the lowest corresponding objective value F as given in Equation (3.1), relative to the set of refined local optima obtained.

3.3 Examples

In this section, the proposed kriging-RSM algorithm is applied to five numerical examples[2]. The first four examples are presented in order of increasing complexity in terms of problem dimensionality and an increasing number of linear/nonlinear constraints. The last example is the kinetics case study introduced in Chapter 2. For each example, a table of computational results is provided that illustrates the performance of four optimization algorithms. The kriging-RSM algorithm refers to the new methodology of applying steepest descent to response surfaces after building a global model as presented in Figure 3.4. The remaining algorithms are stand-alone response surface methods presented in Chapter 2. The second algorithm, DS-RSM, applies direct search in the early stages followed by application of steepest descent to response surfaces once the neighborhood of a local optimum has been found. The third algorithm, RSM-S, applies steepest descent to response surfaces at every iteration. The fourth algorithm, RSM-G, refers to the technique of building a response surface and minimizing with respect to the

entire feasible region in order to obtain global steps to the optimum. For each algorithm, one hundred trials are performed from a set of randomly selected feasible starting iterates. The percentage of iterates converging to the global optimum is presented in the second column of the respective table. Using information taken from the subset of starting iterates successfully finding the global optimum, the average number of iterations required, function calls needed, and CPU time required are also reported. All computational results are obtained using an HP dv8000 CTO Notebook PC with a 1.8 GHz AMD Turion 64 processor.

3.3.1 Six-Hump Camel Back Function

The six-hump camel back function is a well-known global optimization test function that is box-constrained. Introducing both noise and black-box complications into this example, the output z_2 is simulated according to a normally distributed perturbation of the deterministic function. The problem is formulated as shown in Problem (3.11) and the deterministic problem is presented in Figure 3.5:

$$\begin{aligned} &min \;\; z_2 \\ &s.t. \;\; z_2 = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 - \left(-4 + 4x_2^2\right)x_2^2 + N(0, 0.05) \\ &\quad -2 \le x_1 \le 2 \\ &\quad -1 \le x_2 \le 1 \end{aligned} \tag{3.11}$$

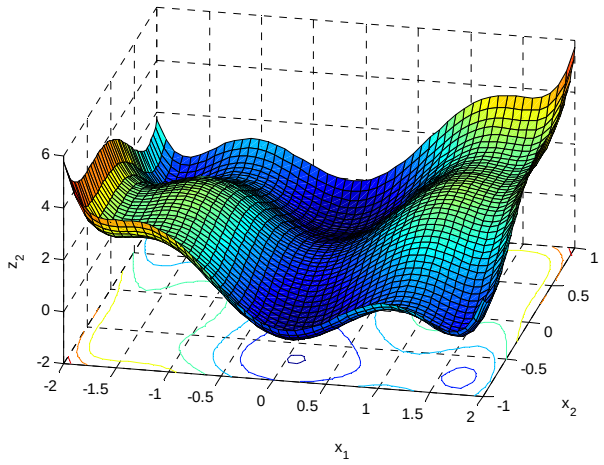


Figure 3.5. Plot of the objective function given in Problem (3.11).

This problem contains four local optima in addition to two global optima and is solved by applying the kriging-RSM algorithm in addition to all three response surface algorithms, DS-RSM, RSM-S, and RSM-G. The objective of applying each optimization algorithm is to identify the global optimum. The results obtained for this problem are presented in Table 3.1.

Table 3.1. Comparison of the performance of the Kriging-RSM algorithm against stand-alone RSM algorithms in finding the global optimum for Problem (3.11).

Opt. Algorithm	% Starting Iterates Finding Global Optimum	# Iterations		# Function Calls			CPU Time (s)		
		K	R	K	R	Total	K	R	Total
K-RSM	81	9	3	47	17	64	5.98	0.09	6.07
DS-RSM	35	N/A	10	N/A	34	34	N/A	0.23	0.23
RSM-S	40	N/A	9	N/A	47	47	N/A	0.28	0.28
RSM-G	30	N/A	7	N/A	41	41	N/A	0.18	0.18

The global minimum is obtained in the fewest number of function evaluations when using the DS-RSM algorithm, but this is to be expected because the set of all optima are fairly evenly spread and have wide basins, meaning that the quadratic curvature is not apparent until the iterates are very close to the optimum. The RSM-S algorithm performs slightly better than the RSM-G algorithm because the two global optima (-0.0898,0.7126) and (0.0898,-0.7126) are located near the center of the feasible region (0,0). When applying the RSM-G method, a quadratic approximation of the function across the entire feasible region places many of the starting iterates in the neighborhoods of locally optimal solutions found at the corners of the feasible region as can be seen in Figure 3.5.

The kriging-RSM algorithm requires approximately 50% additional function evaluations compared to the RSM-S and RSM-G methods, and almost twice as many as the DS-RSM method. However, the additional cost is balanced by the fact that this algorithm leads to global convergence in 81% of the cases. The remaining 19% of the starting iterates successfully found the basin containing either global optimum but terminated at values outside a 2% radius of the optimal solution. The average CPU time required by the Kriging-RSM algorithm is an order of magnitude higher than that of the stand-alone RSM solvers due to the computational expense associated with generating model predictions throughout the feasible region.

3.3.2 Schwefel Function

In this example the four optimization algorithms are applied to the 2-dimensional Schwefel test function, a problem also taken from the global optimization literature. This box-constrained problem is modified to include a linear and a nonlinear constraint. The

black-box function z_2 depends on both x_1 and x_2 and is noisy, modeled by perturbing its deterministic value by a normally distributed error whose standard deviation is 1% of the deterministic function. The problem is formulated as shown below:

$$\begin{aligned} &min \;\; z_2 \\ &s.t. \;\; z_2 = \sum_{i=1}^{2} -x_i sin\sqrt{|x_i|} + N(0,15) \\ &\quad -2x_1 - x_2 \le 800 \\ &\quad 0.004x_1^2 - x_1 + x_2 \le 500 \\ &\quad -500 \le x_1, x_2 \le 500 \end{aligned} \tag{3.12}$$

The deterministic equivalent of this problem is shown in Figure 3.6. This function contains a number of local optima and one global optimum at (420.97, -302.525) with an objective value of -719.53. This problem is selected in order to examine the performance of the stand-alone RSM algorithms when it is the underlying geometry instead of the noise that poses the main complication affecting successful convergence to the global optimum. In Table 3.2, results are presented for this example.

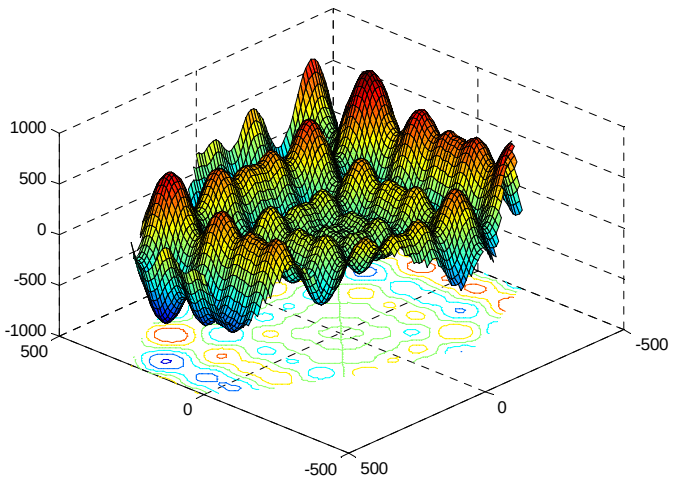


Figure 3.6. Plot of the deterministic objective function given in Problem (3.12).

Table 3.2. Comparison of the performance of the Kriging-RSM algorithm against stand-alone RSM algorithms in finding the global optimum for Problem (3.12).

Opt. Algorithm	% Starting Iterates Finding Global Optimum	# Iterations		# Function Calls			CPU Time (s)		
		K	R	K	R	Total	K	R	Total
K-RSM	100	14	4	74	35	109	6.54	0.11	6.65
DS-RSM	14	N/A	7	N/A	64	64	N/A	0.3	0.3
RSM-S	11	N/A	5	N/A	53	53	N/A	0.23	0.23
RSM-G	11	N/A	5	N/A	49	49	N/A	0.24	0.24

If the starting iterate is located within the neighborhood of the global optimum, it is successfully found using any of the stand-alone RSM methods. However, the global optimum is located in a corner of the feasible region and is surrounded by a set of local optima which other nominal iterates can become trapped in on their path to the global optimum. The average number of function calls required for the DS-RSM method is higher than that observed for the RSM-S and RSM-G methods because the response surface is not created until the end stages of the algorithm.

Because the global optimum lies at the corner of the feasible region, the sequence of iterates follows a longer path towards the basin containing the global optimum without terminating at suboptimal solutions that would be found if response surfaces were created at intermediate stages of the algorithm. The average number of function calls required by the RSM-S and RSM-G methods is approximately the same because the interior candidate values are generally superior to the objective values obtained by sampling at the extremes of the feasible region. Due to the number of local solutions found within the interior of the feasible region, there is a low probability of convergence to the global optimum using the local algorithms.

In contrast, an average of almost twice as many function calls as the local methods are required in order to find a solution when the Kriging-RSM algorithm is applied, but global convergence is observed in all cases. The strategy of building a global model before conducting local search is particularly successful for this problem since convergence to a suboptimal solution using the local methods is avoided. The low CPU time required for solution of this problem using the Kriging-RSM algorithm increases the attractiveness of using this method as an alternative to the stand-alone response surface methods.

3.3.3 Numerical Example 3

This example involves 5 variables with 4 linear constraints and is an example modified from Floudas[29]. The problem is originally presented as an MINLP, and for this example a relaxed NLP is solved by relaxing the integrality constraints. The black-box variable z_2 is a function of two continuous variables and is noisy according to a normally distributed error with standard deviation 0.01. The NLP is formulated as shown in problem (3.13) and a plot of the objective as a function of the two continuous variables is presented in Figure 3.7:

$$
\begin{aligned}
&\min\ F = y_1 + y_2 + y_3 + 5z_2 \\
&s.t.\ \ z_2 = (8x_1^4 - 8x_1^2 + 1)(2x_2^2 - 1) + N(0,0.01) \\
&\quad 3x_1 - y_1 - y_2 \le 0 \\
&\quad 5x_2 - 2y_1 - y_2 \le 0 \\
&\quad -x_1 + 0.1y_2 + 0.25y_3 \le 0 \\
&\quad x_1 - y_1 - y_2 - y_3 \le 0 \\
&\quad 0.2 \le x_1, x_2 \le 1 \\
&\quad 0 \le y_i \le 1 \qquad i = 1...3
\end{aligned}
\tag{3.13}
$$

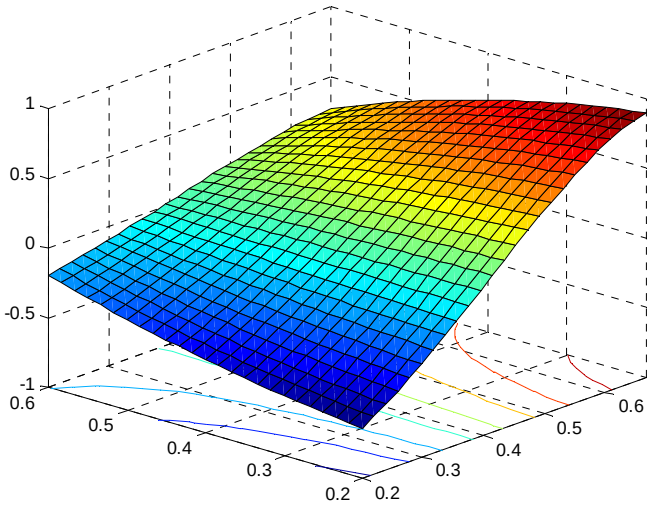


Figure 3.7. Plot of the objective as a function of the continuous variables for Problem (3.13).

A family of global optimal solutions exists for this problem in terms of the binary variables; however, the global optimum of -0.98688 is not achieved unless the optimal vector for the continuous variables is achieved at (0.2,0.2). The results for this example are presented in Table 3.3. The increased CPU time reported for this example using the Kriging-RSM algorithm is higher than that observed for the earlier presented 2-dimensional examples because the kriging predictor is generated over a 5-dimensional grid. Even though it appears from the plot that the nominal iterates in the continuous space should converge to (0.2,0.2) easily, movement is constricted by the feasible region defined by the relaxed binary variables, thereby causing approximately 20 – 30% of the nominal iterates to converge to a suboptimal solution when applying the stand-alone response surface methods. Even though the number of function evaluations required is

higher when using the Kriging-RSM algorithm, global convergence is observed in nearly all cases with only a modest increase in the overall model building costs.

Table 3.3. Comparison of the performance of the Kriging-RSM algorithm against stand-alone RSM algorithms in finding the global optimum for Problem (3.13).

Opt. Algorithm	% Starting Iterates Finding Global Optimum	# Iterations		# Function Calls			CPU Time (s)		
		K	R	K	R	Total	K	R	Total
K-RSM	97	7	7	40	47	87	10.9	0.5	11.4
DS-RSM	70	N/A	13	N/A	40	40	N/A	0.24	0.24
RSM-S	80	N/A	10	N/A	74	74	N/A	0.48	0.48
RSM-G	82	N/A	10	N/A	72	72	N/A	0.75	0.75

3.3.4 Numerical Example 4

This example is also taken from Floudas[29] and involves 11 variables with 11 linear constraints and 3 nonlinear constraints. Although the problem is also formulated as an MINLP, for this example it is solved as a relaxed NLP. The black-box variable z_2 is a function of six continuous variables and is noisy according to a normally distributed error with standard deviation 0.01. The NLP is formulated as shown in problem (3.14):

$$
\begin{aligned}
\textit{min}\quad & z_2 + 5y_1 + 8y_2 + 6y_3 + 10y_4 + 6y_5 + 140 \\
\textit{s.t.}\quad & z_2 = -10x_3 - 15x_5 - 15x_9 + 15x_{11} + 5x_{13} - 20x_{16} + \exp(x_3) \\
& \qquad + \exp(x_5/1.2) - 60\ln(x_{11} + x_{13} + 1) + N(0, 0.01) \\
& -\ln(x_{11} + x_{13} + 1) \le 0 \\
& \exp(x_3) - 10y_1 \le 0 \\
& \exp(x_5/1.2) - 10y_2 \le 0 \\
& -x_3 - x_5 - 2x_9 + x_{11} + 2x_{16} \le 0 \\
& -x_3 - x_5 - 0.75x_9 + x_{11} + 2x_{16} \le 0 \\
& x_9 - x_{16} \le 0 \\
& 2x_9 - x_{11} - 2x_{16} \le 0 \\
& -0.5x_{11} + x_{13} \le 0 \\
& 0.2x_{11} - x_{13} \le 0 \\
& 1.25x_9 - 10y_3 \le 0 \\
& x_{11} + x_{13} - 10y_4 \le 0 \\
& -2x_9 + 2x_{16} - 10y_5 \le 0 \\
& y_1 + y_2 = 1 \\
& y_4 + y_5 \le 1 \\
& a \le x_i \le b \qquad i = 3,5,9,11,13,16 \\
& 0 \le y_i \le 1 \qquad i = 1...5 \\
& a^T = (0,0,0,0,0,0) \qquad b^T = (2,2,2,2,2,3)
\end{aligned}
\tag{3.14}
$$

The solution vector of the deterministic problem is (1.903,2,2,1.403,0.701,2, 0.571,0.429,0.25,0.21,0) and has a corresponding objective value of -0.554. Due to the higher problem dimensionality, the kriging predictor is generated from a set of 1000 feasible points unevenly dispersed throughout the feasible region rather than from an 11-dimensional grid. The results for this example are presented in Table 3.4.

Table 3.4. Comparison of the performance of the Kriging-RSM algorithm against stand-alone RSM algorithms in finding the global optimum for Problem (3.14).

Opt. Algorithm	% Starting Iterates Finding Global Optimum	# Iterations		# Function Calls			CPU Time (s)		
		K	R	K	R	Total	K	R	Total
K-RSM	76	7	17	62	381	443	11.6	9.6	21.2
DS-RSM	55	N/A	24	N/A	855	855	N/A	18.8	18.8
RSM-S	54	N/A	24	N/A	900	900	N/A	20.1	20.1
RSM-G	53	N/A	24	N/A	850	850	N/A	19.2	19.2

For this problem, the number of function calls required to obtain the global optimum is higher by two orders of magnitude compared to the corresponding values obtained from the earlier examples. The significantly higher number of function calls needed when using the response surface methods is due to the increased number of collocation points required in order to build response surfaces in higher dimensions even though the CCD is employed. The average number of function calls required for convergence in the kriging predictor is approximately 14% of the number needed during the refinement stage of the optimization. This result suggests that the kriging predictor sufficiently captures only the coarse geometry. In contrast to the previous examples, when using the local RSM algorithms, a higher number of response surfaces must be obtained before the optimum is found because the problem dimensionality is higher. This leads to increased local model building costs which are on the same order of magnitude as the model building costs for the kriging predictor. However the required number of function evaluations required using the kriging-RSM algorithm is approximately half of the number required using the stand-alone response surface methods, which again emphasizes the success observed

when using information obtained from building the kriging global model to guide the local optimization using RSM.

3.3.5 Kinetics Case Study

The kriging-RSM algorithm is applied to the kinetics case study presented in Chapter 2.3. For this example, a value of N = 100,000 is used to refer to the total number of species particles A and C in the microscale model. Based on this value of N, the amount of noise in the output variable z_2 is described by an additive error term that is normally distributed and has a standard deviation value of σ = 0.011. The optimization results obtained from application of the Kriging-RSM, DS-RSM, RSM-S, and RSM-G algorithms are presented in Table 3.5. The CPU time excludes the time required for each function call in the form of a microscopic model simulation.

Table 3.5. Comparison of the performance of the Kriging-RSM algorithm against stand-alone RSM algorithms in finding the global optimum for Problem (2.7).

Opt. Algorithm	% Starting Iterates Finding Global Optimum	# Iterations		# Function Calls			CPU Time (s)		
		K	RSM	K	RSM	Total	K	RSM	Total
Kriging-RSM	100	9	4	46	25	71	5.27	0.09	5.36
DS-RSM	55	N/A	6	N/A	30	30	N/A	0.09	0.09
RSM-S	54	N/A	5	N/A	35	35	N/A	0.1	0.1
RSM-G	53	N/A	5	N/A	32	32	N/A	0.1	0.1

Even though approximately double the number of function calls are required for convergence to an optimum using the kriging-RSM algorithm, the global minimum is achieved in all cases. Since both the local and global optimum are located in wide

shallow basins near the center of the feasible region, starting iterates are found to converge to either optimum quickly when using the stand-alone response surface methods as seen by the low model building costs.

For all the presented examples, it is seen that the modeling and optimization costs are relatively low when applying the Kriging-RSM algorithm due to the solution of $It_{Krig}N_{Test} + It_{RSM}$ systems of linear equations, where IT_{Krig} and IT_{RSM} represent the number of iterations required for the respective stages of the methodology. In contrast, only It_{RSM} systems of linear equations must be solved when applying any the DS-RSM, RSM-S, or RSM-G optimization algorithms. The value of $IT_{Krig}N_{Test}$ is higher than IT_{RSM}, so the major source of the reported CPU time is attributed to kriging modeling costs. However, for problems of significantly higher dimensions, such as atomistic modeling applications, the overall computational costs will instead be dominated by the time required for either real-time lab experiments or molecular simulation, rather than those associated with model-building.

3.3.6 Loss-In-Weight Feeder Modeling

Loss-in-weight feeders are frequently employed in the pharmaceutical industry for controlling the amount of reactant fed to a process. A common equipment configuration is shown in Figure 3.8[30].

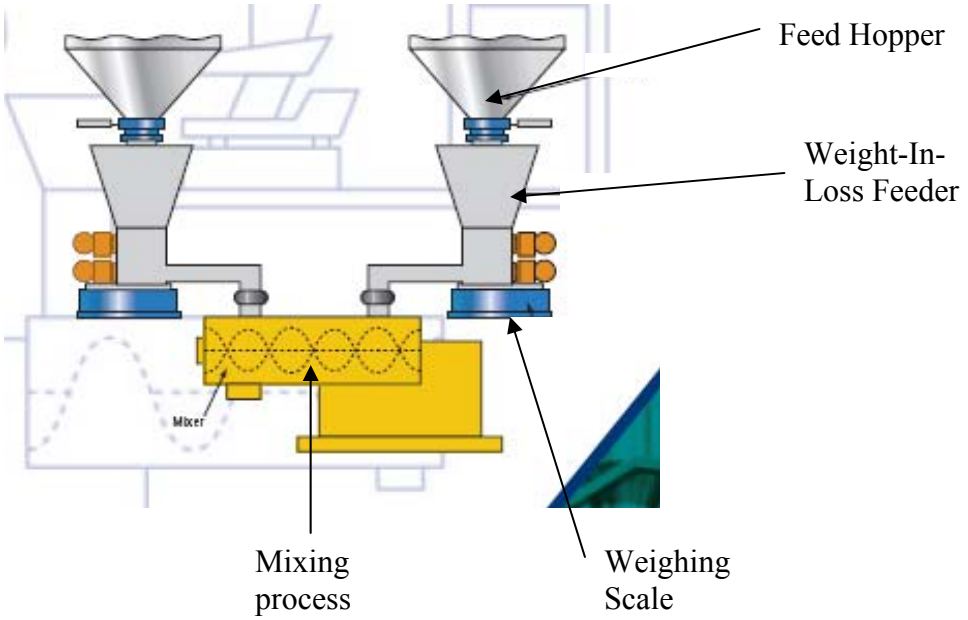


Figure 3.8. Schematic of a Weight-In-Loss Feeder process.

In the configuration shown in Figure 3.14, reactants are loaded into the feed hopper to its fill point. A discharge valve at the bottom of the feeder is opened and a set amount of reactants are fed to a mixing process. The amount of actual reactant discharged over time can be measured using the weighing scale. Based on the information obtained from weighing reactant discharge over short time intervals, the operator can adjust a process controller set point to either slow down or speed up the discharge rate. The feeder process can be considered as a black-box system in which the objective is to generate a model of an output variable, such as the flow variability in the discharge rate, in terms of feeder design variables. This model can be used to identify the best feeder configuration to use for a given application. For this study, both kriging and RSM are applied as surrogate modeling techniques.

For this process, field sampling data have been generated using loss-in-weight feeders provided by K-Tron. All experiments are performed using citric acid, granulated sugar and tea. Samples are taken at every five-second, fifteen-second, and thirty-second time intervals, for a duration of thirty minutes. The mass of samples are measured for each test run and the standard deviation from the average mass is considered to be a measure of the flow variability. The set of input variables are the reactant feed rate, feeder unit screw speed, feeder unit motor speed, and material density. The output variable is the standard deviation of the sample mass. From a randomly chosen set of thirty-four sampling data points, the input-output sampling data of eleven of these points are employed to build the initial kriging model.

Table 3.6. Loss-in-weight Feeders sampling data employed in kriging modeling.

Point	Set Feed Rate (g)	Motor Speed (RPM)	Screw Speed (RPM)	Density (lb/ft^3)	Standard Deviation (g)
1	566.99	380	106	56	
2	715.04	440	122	57	
3	834.74	880	105	19.5	9.3976
4	1574.97	1620	193	19.5	
5	346.49	960	291	22	3.7149
6	1133.98	380	106	56	
7	3149.95	880	105	19.5	
8	692.99	460	128	22	
9	2929.45	440	78	57	
10	4290.23	440	122	57	
11	2078.97	460	128	22	15.0274
12	1700.97	380	106	56	
13	2504.21	880	105	19.5	25.2309
14	4724.92	1620	193	19.5	
15	2145.11	440	122	57	6.6212
16	488.24	440	78	57	2.856
17	1039.48	460	128	22	
18	976.48	440	78	57	4.5968
19	904.03	620	172	56	
20	1464.73	440	78	57	
21	1430.08	440	122	57	
22	692.99	960	291	22	6.2754
23	2712.1	620	172	56	10.3394
24	1808.07	620	172	56	
25	346.49	460	128	22	6.2511
26	1039.48	960	291	22	
27	2078.97	960	291	22	11.4664
28	1574.97	880	105	19.5	
29	3401.94	380	106	56	
30	4724.92	880	105	19.5	
31	3149.95	1620	193	19.5	
32	5424.21	620	172	56	
33	9449.84	880	105	19.5	
34	9449.84	1620	193	19.5	

The objective of the kriging model is to accurately predict the flow variability of the remaining twenty-three sampling points. The predictor is considered accurate once the average predictor value, or average predicted flow variability, of the current kriging model, is within 5% of the corresponding value of its predecessor. The average predictor value corresponds to the variable μ given by Equation (3.9).

For the first iteration, the average value of the kriging model, 9.97 g, is compared against the average value of the eleven sampled points, which is 9.25 g. Since the 7.766% difference between these values exceeds the 5% tolerance, the experimental output data values of two additional sampling vectors are added to the initial subset, and a new kriging model is built. The two additional vectors are selected as sampling points #7 and #28, since the difference between the corresponding predicted and experimental flow variability values is highest among the remaining data points not included in the initial sampling subset. The subsequent kriging predictor, now built from thirteen sampling points, has an average prediction value of 7.79 g.

There is a 21.8% difference between the average prediction value of the new predictor and the 9.97 g mean value corresponding to the first model, and therefore additional refinement is necessary. Two more data points are selected according to the same criterion employed in selecting the first two additional points, and once the updated model is built, the convergence test is again applied. At the third and fourth iterations, the average predicted flow variability values are 7.18 g and 7.40 g, respectively. The 7.40 g value differs from the 7.18 g value by three percent, and since the 5% tolerance criterion is satisfied, the kriging procedure is now terminated. At this point, seventeen sampling points have been used to build the final kriging predictor.

The kriging predictions obtained from models built at each one of the four iterations are presented in Figures 3.9 – 3.12, respectively, in which the magenta squares represent the experimental flow variability values and the blue diamonds refer to kriging predictions. The modeled uncertainty in the flow variability point estimate is expressed in terms of a normal distribution whose mean value is the flow variability point estimate and whose variance is determined according to Equation (3.8). The estimated variance is given by the error bars shown in each one of the four figures. Kriging predictor accuracy is confirmed by observing that most of the experimental standard deviation values given in Table 3.6 fall in the interval defined by the predicted value and the variance. Model prediction can be improved if a stricter convergence criterion is used, such as requiring that the average model value over consecutive iterations differs by no more than one or two percent, for example. However, increased accuracy is obtained at the expense of using additional sampling information for model construction.

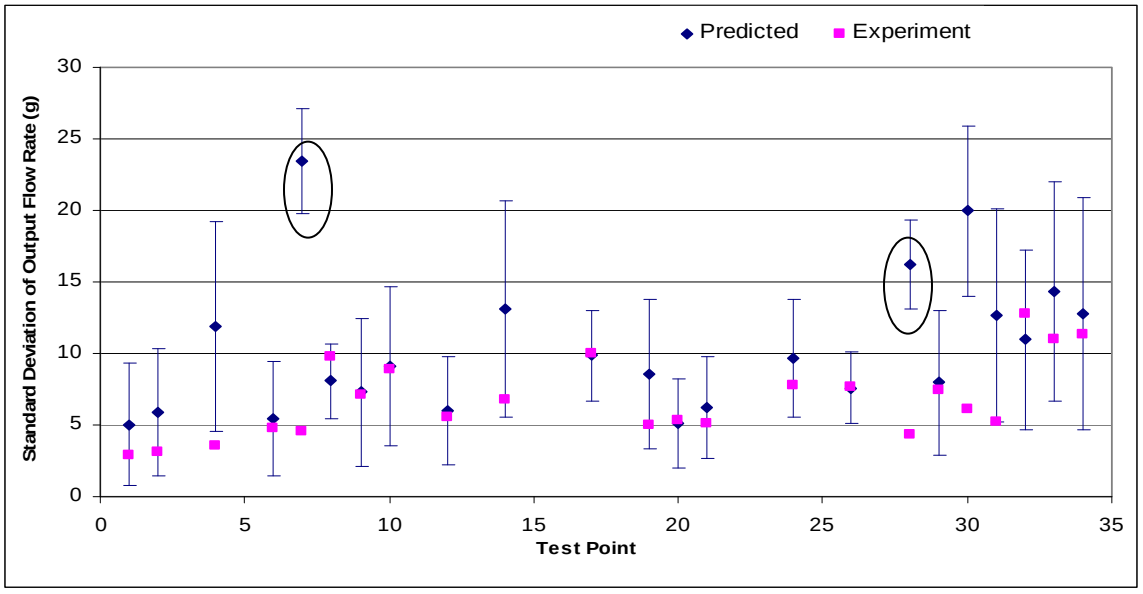


Figure 3.9. Comparison of the first-iteration kriging model predictions against experimental flow variability measurements for the Loss-In-Weight Feeder case study.

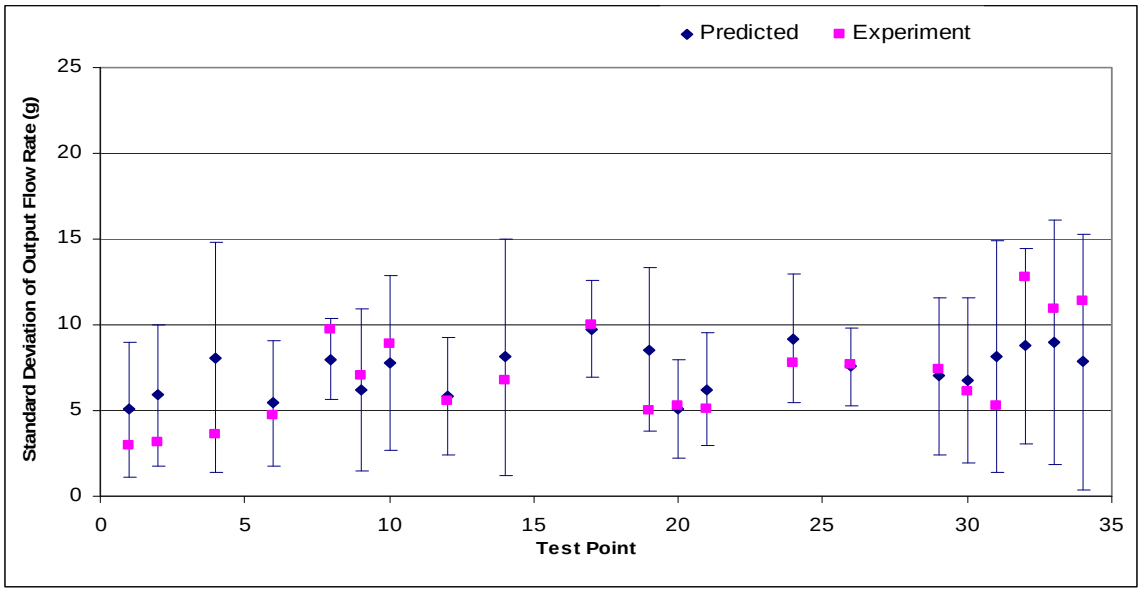


Figure 3.10. Comparison of the second-iteration kriging model predictions against experimental flow variability measurements for the Loss-In-Weight Feeder case study.

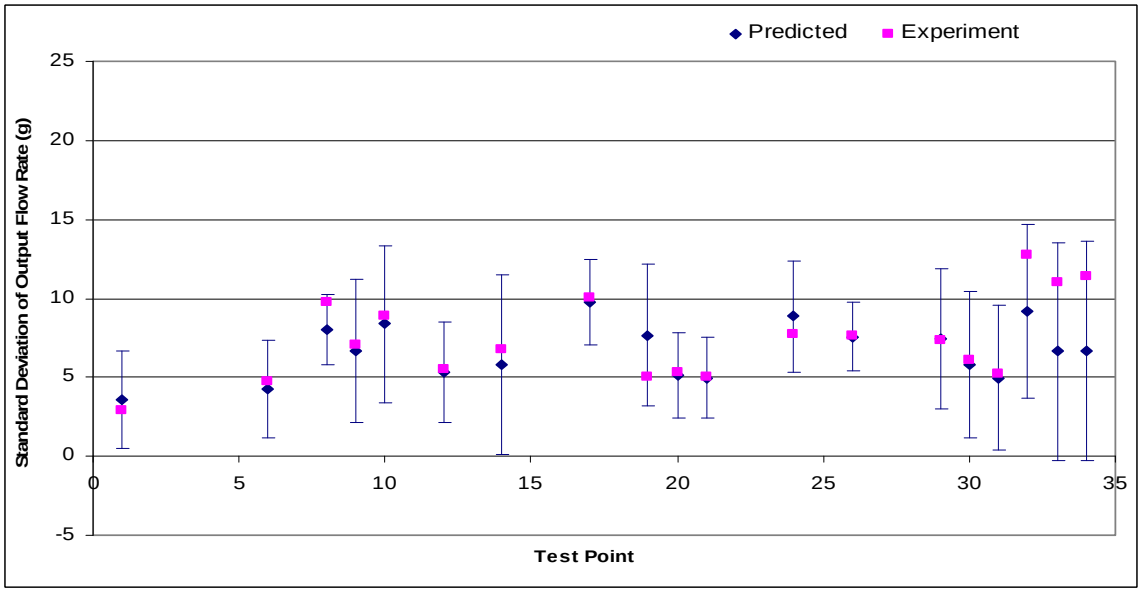


Figure 3.11. Comparison of the third-iteration kriging model predictions against experimental flow variability measurements for the Loss-In-Weight Feeder case study.

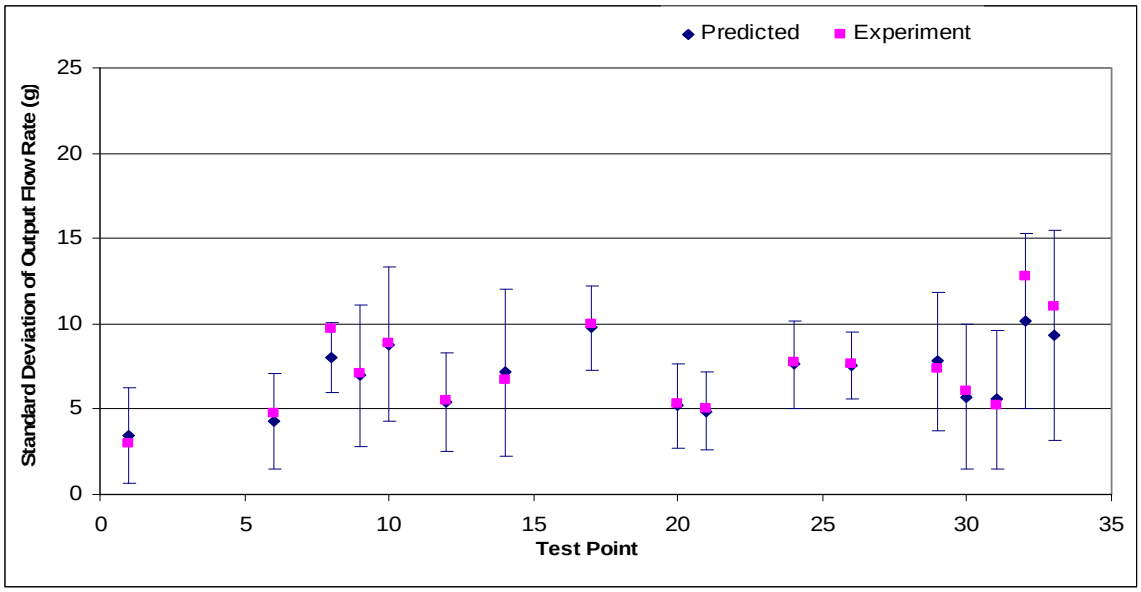


Figure 3.12. Comparison of the fourth-iteration kriging model predictions against the experimental flow variability measurements for the Loss-In-Weight Feeder case study.

Since an accurate kriging predictor is generated in just four iterations, the rule used for the generation of new sampling points for model refinement – that is, selection based on the highest difference between experimental and predicted output values – can be considered a reasonable heuristic to apply for other systems. Since an accurate model is generated at low computational cost, kriging is a computationally cheaper technique compared to more CPU-intensive first-principles modeling techniques such as Discrete Element Method for modeling flow variability in terms of feeder input conditions.

A response surface model is also built using the data presented in Table 3.6. Since seventeen points are used to build an accurate kriging predictor, seventeen points are also used for response surface construction. The sampling data used are shown as follows in Table 3.7, and are chosen according to the rule that the average distance between the testing points and sampling points is minimized.

Table 3.7. Loss-in-weight feeder sampling-point data employed in response surface construction.

Point	Set Feed Rate (g)	Motor Speed (RPM)	Screw Speed (RPM)	Density (lb/ft^3)	Standard Deviation (g)
2	715.04	440	122	57	3.159
3	834.74	880	105	19.5	9.3976
7	3149.95	880	105	19.5	4.5929
9	2929.45	440	78	57	7.0584
11	2078.97	460	128	22	15.0274
13	2504.21	880	105	19.5	25.2309
14	4724.92	1620	193	19.5	6.7285
15	2145.11	440	122	57	6.6212
16	488.24	440	78	57	2.856
23	2712.1	620	172	56	10.3394
24	1808.07	620	172	56	7.7571
25	346.49	460	128	22	6.2511
28	1574.97	880	105	19.5	4.317
30	4724.92	880	105	19.5	6.0776
32	5424.21	620	172	56	12.7948
33	9449.84	880	105	19.5	10.9696
34	9449.84	1620	193	19.5	11.3757

The data are fitted using least squares to a quadratic response surface containing bilinear interaction terms. The resulting closed-form equation is given as follows by Equation (3.15):

$$\begin{aligned} z = {} & 10.09 + 11.94x_1 - 0.76x_2 - 5.22x_3 - 16.46x_1x_2 + 43.38x_1x_3 \\ & - 3.74x_1x_4 + 11.75x_2x_4 - 8.63x_1^2 - 6.16x_2^2 - 6.42x_4^2 \end{aligned} \tag{3.15}$$

in which x_1, x_2, x_3, and x_4 represent set feed rate, motor speed, screw speed and density, respectively, and z is the standard deviation of the output flow variability. The experimental and predicted test point values are shown in Figure 3.13.

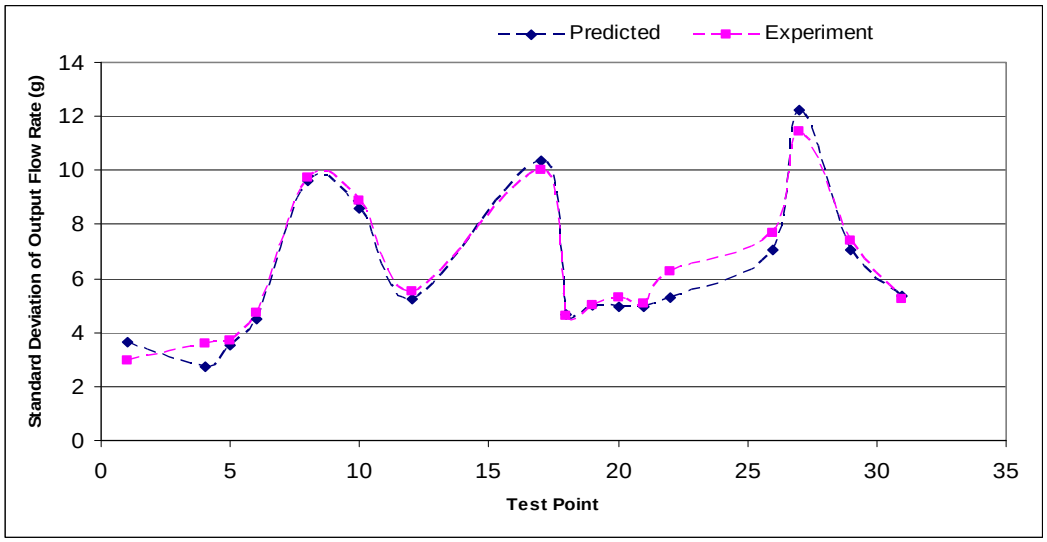


Figure 3.13. Comparison of the response surface predictions against experimental flow variability measurements for the Loss-In-Weight Feeder case study.

The average error between the modeled value and experimental data for the testing points is also 7.02%. This error value is identical to the average error between the predictions and experimental values of the test point set based on the kriging predictor obtained after four iterations. However, different average error values are obtained when the quadratic response surface model is generated from a different set of seventeen sampling points, a significant limitation when compared to the kriging method in that the 7.02% average error is obtained for a randomly selected initial sampling set of eleven points for nominal model building.

The kriging algorithm is self-correcting in that an accurate model can be obtained when a poor model is generated from a naively chosen nominal sampling set, due to the heuristics employed for the selection of additional sampling data to use for model updating. The experimentally obtained output flow variability for half of the sampling

data falls within the predicted confidence intervals generated from a kriging model constructed from the remaining half of the sampling data. As a result, kriging can be employed as an computationally inexpensive method for building accurate global models via iterative refinement.

3.4 Summary

In this chapter, a new kriging-RSM algorithm has been presented for the solution of convex constrained NLPs containing black-box functions and noisy variables. A global model is built using the kriging methodology whereby predictions and variances at discretized test points are obtained using a fitted covariance function built from scattered sampling data. After additional sampling has been performed, the covariance function is updated, leading to better predictions and an improved global model. Once convergence in the average value is observed, regions containing possible local optima are identified and RSM is applied in order to refine the current set of candidate vectors. The global optimum is selected as the best point of the set of local optima. The likelihood of finding the global optimum increases when applying the kriging-RSM algorithm because the global model obtained using kriging allows the set of regions containing potential local optima to be identified. The application of the proposed approach can lead to a substantial increase in the probability of finding the global optimum compared to stand-alone RSM at minimal increased sampling cost due to model construction of the kriging predictor, even though no theoretical guarantees of global optimality are made.

Chapter 4

Mixed-Integer Optimization Considering Continuously-Valued Black-Box Models

The contribution of the work in this chapter is the development of a methodology for the solution of mixed-integer nonlinear programs under uncertainty whose problem formulation is complicated by both noisy variables and black-box functions representing a lack of model equations[31]. The Branch-and-Bound framework is employed to handle the integer complexity whereby the solution to the relaxed LP/NLP subproblem at each node is obtained using both global and local information. Global information is obtained using kriging models used to identify promising neighborhoods for local search. RSM is then employed whereby local models are sequentially optimized to refine the LP/NLP problem optimum. The proposed algorithm is applied to several small process synthesis examples and its effectiveness is evaluated in terms of the number of function calls required, number of times the global optimum is attained, and computational time.

4.1 Introduction

Many process synthesis, design, and operations problems can be modeled as integer programming problems due to choices of process units, operating conditions, or task assignments. However, seldom is all the information available that is required to build a

deterministic analytical model. Since conventional mixed-integer nonlinear program (MINLP) solvers such as Discrete and Continuous Optimizer (DICOPT)[32] require the existence of explicit deterministic equations, they are unable to address systems containing noise and unknown model equations. As a result, process synthesis problems are difficult to solve when the problem formulation contains black-box models for which noisy input-output sampling data are the only information available.

In this chapter, a new algorithm based on a Branch-and-Bound (B&B) main structure is presented as a technique for solving MINLP problems involving noise and black-box models. This work is an extension of the kriging-RSM methodology presented in Chapter 3 that was developed as a technique for solving NLP involving black-box models and noisy variables. The new method addresses the integer complexity using a B&B framework, enabling the previously developed techniques to handle a larger class of problems. In the new algorithm, kriging is used to construct global models of all black-box units. At each node, a kriging predictor describing the behavior of a relaxed NLP objective is employed to identify regions of potential optima. Response surface techniques are then applied to local models in order to refine the set of candidate solutions. The global model building expense is offset by the identification of more reliable lower and upper bounds (LB/UB) at each node, improving the speed at which the global optimum to the MINLP is obtained.

4.1.1 Literature Review

In general, for problems containing explicit analytical models, MINLP solution approaches such as Branch & Bound (B&B)[33,34], Outer Approximation (OA)[35], and

Generalized Benders Decomposition (GBD)[36] split the overall problem into easier NLP and MILP subproblems whose solutions provide a converging sequence of upper and lower bounds. Floudas et. al[37] present a review of the state-of-the-art techniques. In B&B, an NLP is formulated at the first node from the MINLP by relaxing the integrality constraint for all binary variables. New subproblems are created by sequentially branching on binary variables. Values of prior binary variable assignments yielding the best objective solution are retained until the optimum is found. Multivariable branching [38] and parallel branch-and-bound methods[39] accelerate convergence, but the problem may still be computationally very expensive. The OA algorithm relies upon linearization of the objective function and constraints to reduce problem complexity, but relies on differentiability and certain convexity assumptions. Furthermore, the computational cost associated with solving the master MILP increases at each iteration since the problem size increases due to the presence of an accumulating number of linearizations in the form of additional feasible region constraints.

In GBD, the master MILP is formulated using the dual information corresponding to the relaxed NLP problem. Compared to OA, the time required to obtain the solution of the master MILP problem formulated using GBD is less computationally expensive, but more iterations may be required before algorithmic termination. Recently, an alternative framework was proposed based on the ideas of simplicial approximation of the feasible region, which guarantees convergence under specific convexity conditions[40].

The Extended Cutting Plane method[41] successively linearizes the most violated constraint at the predicted minimizer, generating a sequence of nondecreasing lower bounds. While this algorithm does not require solving an NLP, convergence can be slow.

Cutting plane methods have been combined with B&B to produce hybrid methods known as branch-and-cut. In Generalized Disjunctive Programming[42], the constraints are written in terms of logical operators to reduce the computational complexity. The family of αBB algorithms (αBB, SMIN-αBB, GMIN-αBB) targets the solution of twice-differentiable nonconvex NLP and MINLP having restricted participation in the binary variables[43-46]. These methods rely on the generation of valid convex underestimators for the lower bounding problems in order to overcome the algorithmic difficulties presented by the nonconvex functions. There are also many variants of the above methods that exploit special problem structure.

A variety of techniques have also been applied in the field of stochastic programming in order to address the solution of MINLP containing variables with uncertainty. One class of techniques determines the solution based on information obtained from complementary deterministic problems created after obtaining sampling realizations in the uncertain space. Specifically, OA is used to solve MILP and NLP subproblems formulated using the sample average approximation method[47]. Confidence intervals on LB/UB are refined by solving a higher number of replicated subproblems created from an additional number of realizations in the uncertain space. This methodology has been extended by using the simplicial approximation approach to describe the feasible region using a convex hull approximation[40]. However, model equation availability is required in order to obtain linearization information, so these methods cannot be directly applied towards the solution of the problem involving black-box models.

When black-box models are present, a second class of techniques can be used which rely on zero-order techniques to find the integer global solution. Derivative-free methods

can be coupled with process simulators such as PRO-II and ASPEN, thereby enabling the uncertainty complications to be addressed outside of the simulation environment without losing the synthesis capabilities built around deterministic models. This approach has been recently employed whereby a stochastic annealing algorithm has been wrapped around ASPEN in order to obtain the solution of a hydrodealkylation synthesis under uncertainty[48].

With the exception of B&B, a limitation of most of the MINLP algorithms previously described is that differentiability of the objective and constraints is assumed, a condition that is not satisfied if the problem is noisy, involves black-box models and/or uncertainty. Since the B&B-Kriging-RSM algorithm presented in this chapter can overcome these complexities, and its application does not require *a priori* satisfaction of any differentiability conditions, it can therefore serve as a complementary solver for this particular MINLP problem class.

4.1.2 Problem Definition

The problem addressed in this chapter can be expressed in the following form as given in Equation (4.1):

$$
\begin{aligned}
&\min \quad F(x, y, z_1, z_2) \\
&s.t. \quad g(x, y, z_1, z_2) \le 0 \\
&\qquad h(x, y, z_1) = 0 \\
&\qquad z_2(x) = \Gamma(x) + \varepsilon(x) \\
&\qquad \varepsilon(x) \in N(x \mid \mu, \sigma^2) \\
&\qquad N\left(x \middle| \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \\
&\qquad x \in \Re^n, y \in \{0,1\}^q
\end{aligned}
\tag{4.1}
$$

In this formulation, x and y represent continuous and binary variables, respectively. The deterministic variables z_1 describe outputs whose modeling equations $h(x,y,z_1)$ are known. Stochastic output variables z_2 exist when the input-output functionality $\Gamma(x)$ is black-box simulated by a deterministic output perturbed by additive noise $\varepsilon(x)$. The model for $\varepsilon(x)$ is a normally distributed function having mean zero and variance σ^2. Synthesis equations are given by $g(x,y,z_1,z_2)$ which include design constraints, operating specifications, and logical relations. The noise is described by a normally distributed error whose mean μ and variance σ^2 can change depending upon the spatial location of x.

4.2 Solution Approach

The central idea of the proposed algorithm for the solution of problem (4.1) is to use a B&B framework whereby at each node, a kriging predictor of a relaxed NLP objective function is built which serves as a global model. Using the global model of the objective,

promising regions for local search are identified that serve as starting neighborhoods for refinement of the candidate solution set using sequential response surfaces. In order to reduce sampling and model building costs, RSM is applied to coarse kriging predictors for local optimization. Kriging models built at later nodes incorporate both the sampling data used in previous kriging model construction in addition to the sampling data obtained from using RSM.

During the early stages of the MINLP optimization, the computational cost is reduced by (a) use of coarse global models at early nodes and (b) use of weaker stopping tolerances for the kriging and RSM stages. More specifically, at the first node of the B&B tree, the global model is built using k_{Test} points. For each subsequent level of the B&B tree, global model accuracy is improved by using 10 – 25% additional test points relative to the number employed at the previous level. For the examples presented in Section 3, the value of k_{Test} employed at the root node is set at 1,000. A second method of reducing early computation expense relies upon a weak initial stopping criterion for global model improvement which is successively increased. This criterion is based on whether convergence in the sequence of average kriging model values is observed. At the first node, the initial tolerance Tol_{Krig} might be satisfied if the average prediction value falls within 90% of the value at the previous iteration. The value of Tol_{Krig} could be increased to 95% for the second level, and to 99% for all subsequent levels.

Kriging models are built for both the black-box units and the relaxed objective at each node. Since the relaxed NLP objective may differ from node to node depending upon binary variable assignments, a new kriging predictor may need to be constructed for each NLP subproblem. The kriging model of each black-box process describes unit-specific

system behavior, whereas the kriging model of the relaxed NLP objective is created for optimization purposes in order to identify the best regions for local search. Once the global models for all black-box units have been created, this information can be incorporated into the construction of any arbitrary objective while simultaneously avoiding sampling duplication. Once the kriging solution has been refined using RSM, the optimum is classified as a lower or upper bound based on integer feasibility in y. Based on the application of fathoming criteria as described within the B&B algorithm, the details of which are presented in the next subsection, new subproblems are then formulated if a stopping criterion based on the difference in the LB/UB is not met.

4.2.1 B&B Algorithm

The B&B algorithm is used to bracket the integer optimal y_2 solution objective between a converging sequence of lower and upper bounds (LB/UB). Each LB and UB corresponds to the solution attained for a partially relaxed NLP subproblem. At the start of the procedure, the initial LB and UB are set at $-\infty$ and $+\infty$ and the first partially relaxed NLP subproblem is formulated by relaxing all y_2-variables. The optimal solution to the NLP is classified as a LB if it is integer infeasible in the y_2-variables and an UB otherwise. If integer feasibility is not met, two new NLP subproblems can be formulated which require integer feasibility for any or all of the fractional y_2-variables. Based on the application of the floor and ceiling functions to an integer infeasible vector, two disjoint subregions are generated which define the feasible space for each new NLP.

If the global solution has been attained for a partially relaxed NLP subproblem, a solution that is obtained over a reduced feasible region cannot be better than the solution

attained for the parent NLP. Therefore, when the NLP solution has been designated as an UB, no additional subproblems are formulated. Conversely, when the solution is a LB, additional subproblems are created only if the LB is lower than the best UB. As the optimization progresses, a sequence of monotonically increasing LB and monotonically decreasing UB are generated which bracket the objective corresponding to the y_2 integer optimal solution. The procedure is terminated once the list of candidate NLP subproblems is empty, or the LB/UB integrality gap has fallen below a stopping tolerance Tol_{BB}. The integer optimal solution corresponds to the best UB and is designated as the solution to the original MINLP. By combining the kriging-RSM algorithm used for obtaining NLP solutions with B&B, the integer global solution of MINLP can be efficiently found since the B&B fathoming criteria limits the number of NLP subproblems that have to be solved. The source of the computational expense for the optimization lies in generating reliable kriging models for both the black-box models and node-specific relaxed NLP objective functions.

Due to the noise and the presence of black-box units, global optimality cannot be guaranteed. The identification of suboptimal solutions at each node can delay search and even cause integer feasible solutions to be missed. As a result it is unlikely that local optimization using response surfaces can lead to the discovery of the global optimal solution if the correct neighborhood has not first been identified using the kriging predictor. One way of addressing this problem is to apply ideas similar to the ones previously employed[40,47]. Let the number of black-box units be R and let S replicate sets of the R kriging predictors be created based on different nominal sampling sets. The converged kriging mappings are not necessarily the same since they are built using

different initial collocation points. For each s^{th} set of R global models, $s = 1 \ldots S$, the corresponding mapping of the node-specific objective function can be built and the optimal objective function $F_{pred,krig,s}$ obtained. Both the mean and variance of the objective function $\{F_{pred,krig,s} | s=1 \ldots S\}$ are then determined and used as a point estimate and confidence interval for the global solution. If integrality in the binary variables is satisfied, the point estimate is an upper estimate of the upper bound, otherwise, it is an upper estimate of the lower bound. Next, optimization is performed at the RSM level. The best kriging solution of the S replicates is then refined using RSM T times where $T > S$. Let the set of refined RSM solutions be represented by $F_{RSM,t} | t=1 \ldots T$. Both the mean and variance of $[F_{RSM,t} | t=1..T]$ can then be obtained and used as a lower estimate of the solution regardless of whether it is a lower or an upper bound. The tradeoff for the benefit obtained by applying this technique – increased confidence in classifying a solution as the global optimum – is that the sampling costs may become prohibitive. In the next subsection, the details of the comprehensive B&B-Kriging-RSM algorithm are presented.

4.2.2 B&B-Kriging-RSM Algorithm

The B&B-Kriging-RSM algorithm proceeds as follows. First, stopping tolerances are established. The Tol_{Krig} and $Tol_{Krig,Obj}$ parameters are used to terminate kriging predictor improvement for a process unit output and the relaxed NLP subproblem objective at an arbitrary node, respectively. The Tol_{BB} parameter is used to terminate further search at the Branch and Bound stage based on the difference between the LB/UB. The use of the Tol_{BB} parameter is motivated by the need to avoid the additional sampling and

computational expense associated with solving additional NLP subproblems whenever the improvement in the objective is expected to be low.

Sampling data are then obtained in order to create kriging predictors for the black-box units. The relaxed NLP is formulated at the first node and the kriging model of the corresponding objective is generated. The best kriging solutions are identified and sampling is performed at locations of predicted optima to confirm global model reliability for the black-box units. If the difference between the sampled and predicted objective values exceeds $Tol_{Krig,Obj}$, global improvement is considered necessary. It should be noted that a subset of the inputs x in each sampling vector may actually be noisy outputs z_2 from upstream black box units. For each black box unit, the set of sampled data $z_{2,act}$ is compared against the set of corresponding kriging predictions $\tilde{z}_{2,Pred}$. Further sampling is then conducted for the subset of global models in which $|z_{2,act} - \tilde{z}_{2,Pred}| > Tol_{Krig}$ in order to improve the kriging predictors. Once the difference $|f_{act}-f_{pred}|$ falls below $Tol_{Krig,Obj}$, the best kriging solutions f_{pred} are refined based on sequential optimization of response surfaces. An alternative stopping criterion Tol_{Krig} that can be applied requires the determination of the average value of the set of kriging predictions for each iteration as given by Equation (3.9). Once convergence is attained in this average value for all black box units, the optimal kriging solutions for the objective would then be identified for refinement using RSM. Due to feasible region partitioning, the number of sampling points required for local optimization using RSM may decrease as 0-1 assignments are made in the binary variables due to the fact that search for a refined optima now occurs over a reduced feasible region.

For a given NLP subproblem, if the best kriging solution lies within the basin of an optimum, it will be identified using RSM. In order to determine whether additional optima exist, RSM is applied to the next best kriging solution. The corresponding RSM optimal solution is compared to the one already found. If the solution is found to be inferior, further application of RSM towards additional kriging solutions terminates. Otherwise, RSM is then applied to the subsequent optimal kriging candidate in order to determine whether another minimum exists. It should be noted that if the set of points x_k over which the kriging model is built does not contain vectors located near optima, it is possible for some or all of the minima to be missed. In order to overcome this problem, the test set should be comprised of points approximating a uniform coverage of the feasible region. The optimum is then determined to be a LB/UB depending upon 0-1 feasibility in the binary variables.

If the difference between the best LB/UB falls below Tol_{BB}, the overall B&B-Kriging-RSM algorithm terminates; otherwise, additional subproblems are determined according to the Branch and Bound fathoming criteria. A new subproblem is then selected from the candidate set and the kriging model of the relaxed NLP objective for the new problem is constructed. If the set of new candidate subproblems is empty, the algorithm terminates with the best UB established as the solution to Equation (4.1). A flowchart of the proposed algorithm is shown in Figure 4.1 and the effectiveness of the method is demonstrated based on the computational results obtained for the solution of two process synthesis examples presented in the next section.

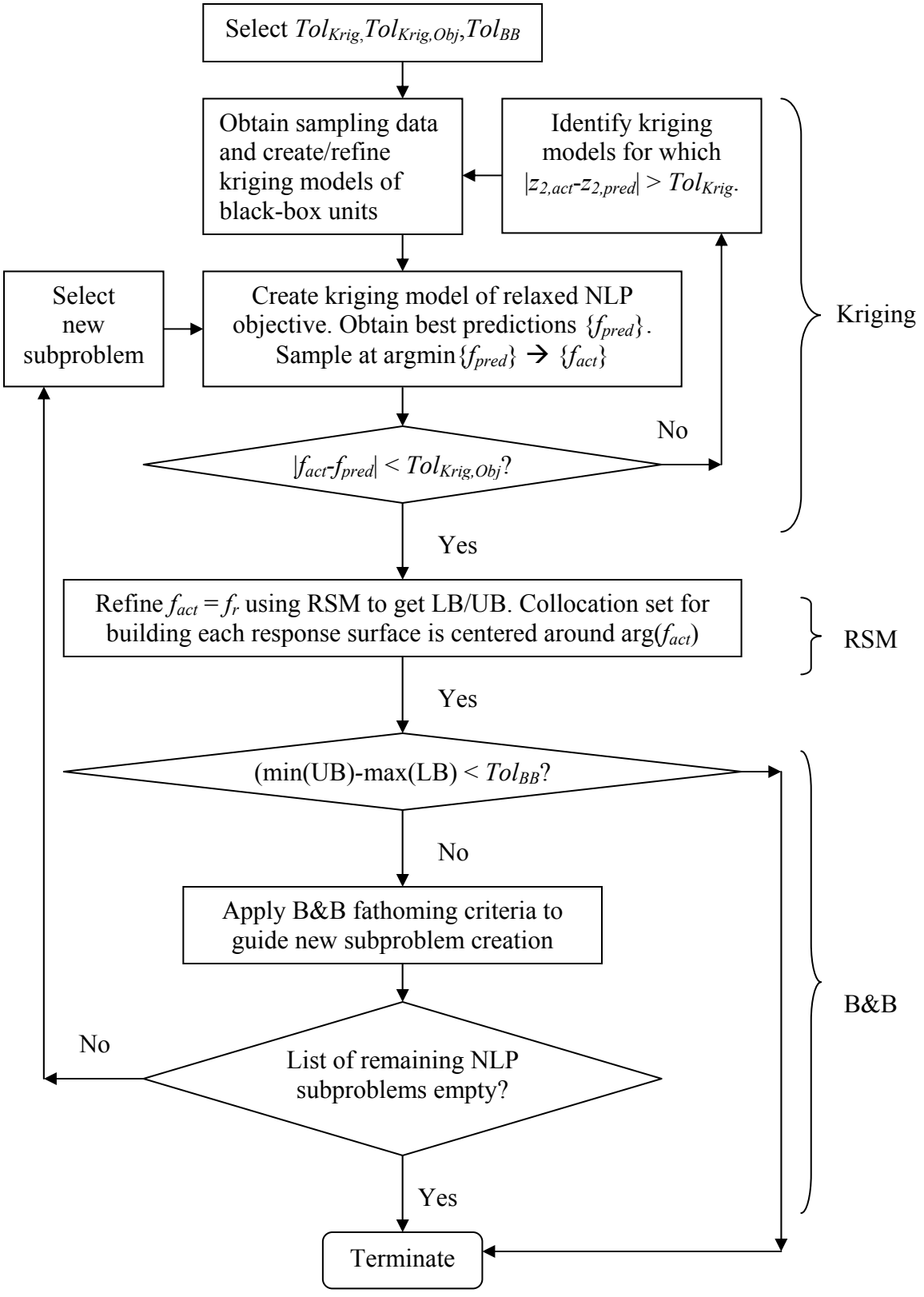


Figure 4.1. Flowchart of the B&B Kriging-RSM algorithm.

4.3 Examples

In this section, the proposed Branch and Bound Kriging-RSM algorithm is applied to two numerical examples[29] and a modified propylene/propane separation synthesis study[49]. For each example, a table of optimization data is provided based on the performance of the proposed Branch and Bound Kriging-RSM algorithm. For each algorithm, 100 trials are performed from a set of randomly selected feasible starting iterates. Based on information taken from the subset of starting iterates successfully finding the global optimum, the average number of nodes visited, iterations required, function calls needed, and CPU time required are also reported. All computational results are obtained using an HP dv8000 CTO Notebook PC containing a 1.8 GHz AMD Turion 64 processor.

4.3.1 Numerical Example 1

This example involves six variables, four linear constraints, and two nonlinear constraints. The black-box variable z_2 is a function of three continuous variables and is made noisy from applying an additive random error to the deterministic output. The random error term is normally distributed and has a standard deviation of 0.01. The problem is formulated as shown in Equation (4.2):

$$
\begin{aligned}
&\min\ F = z_2 + 5y_1 + 6y_2 + 8y_3 + 10 \\
&s.t.\quad z_2 = -7x_6 - 18\ln(x_2+1) - 19.2\ln(x_1 - x_2 + 1) + N(0, 0.01) \\
&\qquad 0.8\ln(x_2+1) + 0.96\ln(x_1 - x_2 + 1) - 0.8x_6 \le 0 \\
&\qquad x_2 - x_1 \le 0 \\
&\qquad x_2 - Uy_1 \le 0 \\
&\qquad x_1 - x_2 - Uy_2 \le 0 \\
&\qquad \ln(x_2+1) + 1.2\ln(x_1 - x_2 + 1) - x_6 - Uy_3 \le 0 \\
&\qquad y_1 + y_2 \le 0 \\
&\qquad y \in \{0,1\}^3\ \ a \le x \le b, x = (x_j : j = 1,2,6) \in \Re^3 \\
&\qquad a^T = (0,0,0), b^T = (2,2,1), U = 2
\end{aligned}
\tag{4.2}
$$

The problem is solved by treating z_2 as a black-box variable as given by Equation (4.3) and optimizing the kriging and response surface models built from x-z_2 sampling data.

$$z_2 = \Gamma\left(x_1, x_2, x_6\right) \tag{4.3}$$

The solution of the deterministic problem is $(x_1,x_2,x_6,y_1,y_2,y_3)$ = (1.301,0,1,0,1,0). The optimization results obtained for this example are presented in Table 4.1 based on one hundred applications of the B&B-Kriging-RSM. For each application of the algorithm, a different nominal sampling set comprised of ten randomly dispersed iterates is used to build the initial kriging model at the root node of the B&B tree. For one such test run, the solution information obtained at each node of a B&B tree is presented in Table 4.2.

Table 4.1. Optimization results obtained for Problem (4.3), based on application of the B&B Kriging-RSM algorithm.

% Starting Iterates Finding Global Optimum	# Nodes	# Function Calls			CPU Time (s)		
		Kriging	RSM	Total	Kriging	RSM	Total
90	6	113	161	274	36.55	0.59	37.14

Table 4.2. Optimization information corresponding to each node of the B&B tree for one trial solution of Problem (4.3).

Node	Fixed 0-1 Vbls.	Optima Information		Kriging			RSM		
	(y_1,y_2,y_3)	$(x_1,x_2,x_6, y_1,y_2,y_3)$	F	It.	FC	CPU Time (s)	It.	FC	CPU Time (s)
1	(-,-,-)	(1.35,0.84,1, 0.45,0.26,0)	1.427	5	43	44.3	5	75	1.02
2	(-,0,0)	(1.05,1.05,0.71, 0.53,0,0)	5.205	4	34	3.06	6	47	0.19
3	(-,1,0)	(1.301,0,1, 0,1,0)	6	3	25	1.58	2	13	0.14
4	(0,0,0)	(0,0,0, 0,0,0)	10.02	1	1	0.03	0	0	0
5	(1,0,0)	(1.7,1.7,0.98, 1,0,0)	7.258	4	34	2.14	3	21	0.08

4.3.2 Numerical Example 2

This example involves 11 variables with 11 linear constraints and 3 nonlinear constraints. The black-box variable z_2 is a function of six continuous variables and is made noisy by adding a random error term that is normally distributed according to a zero mean and a standard deviation value of 0.01. The problem is formulated as shown in Equation (4.4):

$$
\begin{aligned}
\textit{min}\quad & F = z_2 + 5y_1 + 8y_2 + 6y_3 + 10y_4 + 6y_5 + 140 \\
\textit{s.t.}\quad & z_2 = -10x_3 - 15x_5 - 15x_9 + 15x_{11} + 5x_{13} - 20x_{16} + \exp(x_3) \\
& \qquad + \exp(x_5/1.2) - 60\ln(x_{11} + x_{13} + 1) + N(0, 0.01) \\
& -\ln(x_{11} + x_{13} + 1) \le 0 \\
& \exp(x_3) - 10y_1 \le 0 \\
& \exp(x_5/1.2) - 10y_2 \le 0 \\
& -x_3 - x_5 - 2x_9 + x_{11} + 2x_{16} \le 0 \\
& -x_3 - x_5 - 0.75x_9 + x_{11} + 2x_{16} \le 0 \\
& x_9 - x_{16} \le 0 \\
& 2x_9 - x_{11} - 2x_{16} \le 0 \\
& -0.5x_{11} + x_{13} \le 0 \\
& 0.2x_{11} - x_{13} \le 0 \\
& 1.25x_9 - 10y_3 \le 0 \\
& x_{11} + x_{13} - 10y_4 \le 0 \\
& -2x_9 + 2x_{16} - 10y_5 \le 0 \\
& y_1 + y_2 = 1 \\
& y_4 + y_5 \le 1 \\
& a \le x_i \le b \qquad i = 3,5,9,11,13,16 \\
& y_i \in \{0,1\} \qquad i = 1...5 \\
& a^T = (0,0,0,0,0,0) \qquad b^T = (2,2,2,2,2,3)
\end{aligned}
\tag{4.4}
$$

As was done for the numerical example presented in Section 4.3.1, this problem is solved by treating z_2 as a black-box variable as given by Equation (4.5) and optimizing the kriging and response surface models built from x-z_2 sampling data.

$$z_2 = \Gamma\left(x_3, x_5, x_9, x_{11}, x_{13}, x_{16}\right) \tag{4.5}$$

The NLP solution of the deterministic problem is (x_3 , x_5 , x_9 , x_{11} , x_{13} , x_{16}) = (1.903 , 2 , 2 , 1.403 , 0.701 , 2) and (y_1 , y_2 , y_3 , y_4 , y_5) = (0.571 , 0.429 , 0.25 , 0.21 , 0) and has a corresponding objective value of -0.554. The corresponding MINLP (integer) solution is (0 , 2 , 1.078 , 0.652 , 0.326 , 1.078) and (0 , 1 , 1 , 1 , 0) in the continuous and binary variables, respectively. At the first node, the kriging predictor of the objective is

built from ten thousand feasible points unevenly dispersed throughout the feasible region rather than from an 11-D grid, in order to avoid the modeling expense associated with generating over 280 billion kriging predictions. The performance of the B&B-Kriging-RSM method is evaluated by a procedure similar to the one employed for the example given in Section 4.3.1. For each one of one hundred applications of the algorithm, a different nominal sampling set comprised of fifteen randomly dispersed iterates is used to build the initial kriging model at the root node of the B&B tree. The optimization results are presented in Table 4.3. The corresponding solution information obtained at each node of the B&B tree for one test run is presented in Table 4.4.

Table 4.3. Optimization results obtained for Problem (4.4), based on the application of the B&B Kriging-RSM algorithm.

% Starting Iterates Finding Global Optimum	# Nodes	# Function calls			CPU Time		
		Kriging	RSM	Total	Kriging	RSM	Total
97	9	212	877	1089	29.83	11.64	41.47

Table 4.4. Optimization information corresponding to each node of the B&B tree for one trial solution of Problem (4.4).

Node	Fixed Binary Variables					Kriging				RSM		
	y_1	y_2	y_3	y_4	y_5	F	Its	FC	CPU Time (s)	Its	FC	CPU Time (s)
1	----------					0.179	6	43	44.27	19	75	1.02
2	1	0	----			68.536	5	34	3.06	15	47	0.19
3	0	1	----			64.723	4	25	1.58	9	13	0.14
4	0	1	1	0	0	74.542	6	50	0.59	5	35	0.17
5	0	1	1	1	0	73.084	12	34	2.14	2	21	0.08
6	1	0	0	----					Inf			
7	1	0	1	----		83.396	5	34	2.14	5	21	0.08

The integer global optimum is found 90% of the time when the B&B-Kriging-RSM algorithm is applied; for the remainder of the trials, the algorithm terminates at near-optimal solutions. At the first node, a higher number of function calls are required for local optimization relative to the amount required for global modeling. In order to avoid incurring the increased computational expense associated from the generation of a high number of kriging predictions at locations corresponding to a discretized grid, the test point set is generated from one thousand randomly selected feasible points. However, the location of the kriging optimum using the former method is expected to be inferior to the kriging solution that would have been obtained if discretization had been applied. The reason for this is that the average sampling-pair distance for one thousand sampling points will be higher than the corresponding distance based on sampling pairs uniformly distributed over a grid.. As a result, multiple response surfaces are required in order to refine the coarse estimate of the relaxed NLP solution.

The total number of function calls required for the RSM phase at the root node is seventy-five since fifteen function calls are required to build a response surface at each iteration. Each response surface is constructed according to a central composite design (CCD) template since the number of required sampling experiments is lower than the $3^3 = 27$ required based on a three-level factorial design. At subsequent nodes, the values of a subset of the continuous elements of the best kriging solution match the respective variable upper bounds. These solutions are quickly identified as optimal, and therefore their values are held fixed while lower-D response surfaces are then used to optimize the remaining subset of un-optimized continuous variables. Since the number of function calls required to build lower-D response surfaces is lower relative to the number required for an *n*-D response surface, the sampling expense associated with local optimization at the later nodes of the B&B tree is lower.

4.3.3 Case Study: Propylene/Propane Synthesis

For this third example, the B&B Kriging-RSM algorithm is applied to a modified case study taken from Seader[49]. In the original problem, the task is to separate a 60/40 mol% propylene/propane feed into a 99 mol% propylene distillate/95 mol% propane bottoms product from a using consecutive 100-trayed columns. The problem is modified whereby a 50-tray and 75-tray column are used to achieve a 94 mol% propylene distillate, and is formulated as a process synthesis problem in which a decision must be made as to which column sequence is best for attaining the 94% propylene distillate. The feed can enter either one of the columns and the distillate product from the first column

acts as feed to the second. A representation of the column superstructure is presented in Figure 4.2.

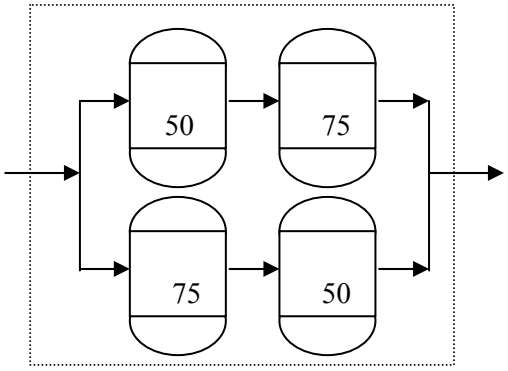


Figure 4.2. Propylene/Propane sequencing problem consisting of two possible sequencing configurations for two distillation towers.

The propylene-rich distillate product exiting the second column is to be sold while the bottoms product from both columns is combined for use elsewhere in the plant. The separation is very difficult due to the close volatility between propylene and propane. A high reflux may be required since 37.5% fewer trays are used than as in the original study, which will increase energy costs. The problem is to determine the column sequence, operating reflux ratios, and reboiler steam inputs maximizing profit and minimizing cost. The problem is formulated in Equation (4.6) where the product recoveries are simulated at 95% of their deterministic value perturbed by a normally distributed error:

$$
\begin{aligned}
max \quad & P = 0.177\left(D_2 x_{Propylene,2}\right) + 0.132\left(D_2 x_{Propane,2}\right) \\
& \quad - 4.41\left(Q_{R1} y_1 + Q_{R2} y_2\right) - 0.2\left(Q_{C1} y_1 + Q_{C2} y_2\right) \\
s.t. \quad & D_2 = 0.95\Gamma_1\left(RR_1, RR_2, Q_{R1}, Q_{R2}, Q_{C1}, Q_{C2}, F\right) + N(0,5) \\
& x_{Propylene,2} = 0.95\Gamma_2\left(RR_1, RR_2, Q_{R1}, Q_{R2}, Q_{C1}, Q_{C2}, F\right) + N(0,0.01) \\
& x_{Propane,2} = 0.95\Gamma_3\left(RR_1, RR_2, Q_{R1}, Q_{R2}, Q_{C1}, Q_{C2}, F\right) + N(0,0.01) \\
& y_1 + y_2 = 1 \\
& 5 \le RR_1, RR_2 \le 20 \\
& 10 \le Q_{R1}, Q_{R2} \le 50 \quad [10^6 \text{ kJ/h}] \\
& F = 11{,}646 \text{ kg/h}, \; x_{Propylene,F} = 0.6, \; x_{Propane,F} = 0.4
\end{aligned}
\tag{4.6}
$$

where the indices $i = 1, 2$ refer to columns 1 and 2, respectively; y_i is the binary variable expressing the existence of column i; D_i, Q_{Ri}, Q_{Ci}, R_{Ri} are the distillate flowrates, reboiler duties, condenser duties, and reflux ratios of column i; $x_{Propylene,2}$, $x_{Propane,2}$, are the propylene and propane mole fractions of the column 2 distillate; F, $x_{Propylene,F}$, and $x_{Propane,F}$ are the column 1 feed rate, propylene and propane feed mole fractions; and Γ_1, Γ_2, Γ_3 represent the black-box models whose respective closed-form equations are inaccessible within the ChemCad process simulator.

The objective represents a profit function where the first and second terms describe the profit obtained by selling the distillate at a price of \$0.177/kg propylene and \$0.132/kg of propane, respectively. The third and fourth terms represent heating and cooling costs, respectively. The ChemCad process simulator is used to simulate both distillation columns represented as black-box units, and is called upon as a slave program from the master driver in Matlab where the modeling and optimization tasks are carried out. Results for this example are presented in Tables 4.5 and 4.6. The process flowsheet represented in Figure 4.3 contains the optimal column sequencing.

Table 4.5. Optimization results obtained for Problem (4.6), based on application of the B&B-Kriging-RSM algorithm.

% Starting Iterates Finding Global Optimum	# Nodes	# Function Calls			CPU Time (s)		
		Kriging	RSM	Total	Kriging	RSM	Total
86	3	74	19	93	382	94	476

Table 4.6. Optimization information corresponding to each node of the B&B tree for one trial solution of Problem (4.6), based on application of the B&B Kriging-RSM algorithm.

Node	Fixed 0-1 Vbls.	Optima Information		Kriging			RSM		
	(y_1,y_2)	(RR_1,RR_2,QR_1,QR_2)	P ($/h)	Its	FC	CPU Time (s)	Its	FC	CPU Time (s)
1	(-,-)	(9.82,10.59,13.2,17.9)	1021	5	52	176	2	12	41
2	(0,1)	(9.49,15.55,22.1,31.4)	875	4	21	89	2	12	37
3	(1,0)	(11.94,9.73, 28.72, 20.85)	931	5	26	113	3	17	43

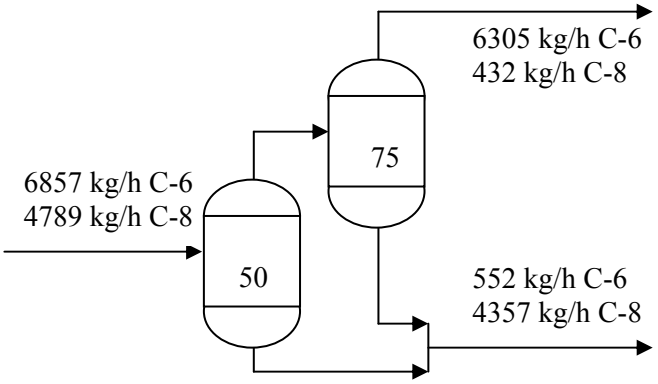


Figure 4.3. Optimal column sequencing for obtaining a 94 mol% propylene distillate.

The computational overhead required for this example, which is given in CPU time (s), is higher than the previous one even though the problem dimensionality is lower due to the computational expense incurred from ChemCad simulation, in which rigorous TOWR distillation models are employed to represent both the 50-tray and 75-tray columns. The B&B Kriging-RSM is successful at finding the global optimum as evidenced by 86% convergence. The optimal plant configuration consists of the 50-tray column preceding the 75-tray column as shown in Figure 4.3. Based on this configuration, the optimal design variables are as follows. The reflux ratios for the first and second columns are 11.94 and 9.73, respectively, while the corresponding steam inputs to each reboiler are 28.72 10^6 kJ/h and 20.85 10^6 kJ/h. An 85% propylene distillate purity is achieved in the first column which is improved to 94% in the second, leading to a $931/h profit. As expected, the tower that has more trays is the one that is assigned to perform the more rigorous task of achieving propylene purity.

4.4 Summary

In this chapter, a new B&B kriging-RSM algorithm has been presented for the solution of constrained MINLPs containing black-box functions and noisy variables. The B&B framework is used to efficiently search in the 0-1 space for the integer global optimum, thereby extending the capabilities of existing work to handle process synthesis problems. Kriging is used to build global models of the black-box functions and the NLP subproblem objective at each node of the binary tree. The surrogate models are used to identify subregions where the relaxed NLP global optimum potentially resides. The best kriging solutions serve as starting iterates for further refinement via optimization of sequential response surfaces. The additional costs resulting from global model creation

are offset by highly successful convergence to the integer global solution as shown by the represented examples.

Chapter 5

Optimization Considering Mixed-Integer Black-Box Models

When black-box systems are functions of both continuous and input variables, RSM and B&B are ineffective for finding the integer optimal solution since the methodology of both techniques can result in sampling being required at fractional integer variable values, which are infeasible. The application of direct search can overcome this problem since optimization occurs based on selection of the integer variable vector having the lowest objective function value relative to a set of candidate solutions. At each iteration, the set of candidate solutions is generated from midpoint-endpoint sampling. Convergence to a global optimum is attained more rapidly if the initial endpoint sampling vectors are defined at the lower/upper bound for each integer variable instead of a more localized interval around the starting iterate. The contribution of the work in this chapter is the presentation of a new MINLP algorithm for problems containing black-box models and noisy variables[50]. The black-box variables can be functions of continuous and/or integer variables. To address the lack of explicit equations, kriging is used to build surrogate data-driven global models used to identify promising solutions for local refinement. The continuous variables are then optimized using a response surface method. The integer variables are optimized using Branch-and-Bound if a continuous

relaxation exists, and direct search otherwise. The four algorithms are unified into a comprehensive approach that can be used to obtain optimal process synthesis and design solutions when noise and black-box models are present. The performance of the proposed algorithm is evaluated based on its application to two industrial case studies.

5.1 Introduction

When black-box models are functions of strictly integer variables, local gradient-based methods such as the response surface methodology (RSM) cannot be applied since fractional values are infeasible. Direct search is thereby proposed for the optimization of this class of variables. In addition, for process synthesis problems, another class of integer variables exists outside the black-box functions for which a continuous relaxation is permitted. In order to obtain an integer optimal solution, Branch-and-Bound (B&B) is proposed for the optimization of this variable class. In the previous chapter, an algorithm incorporating B&B, kriging, and RSM was presented to deal with the class of problems in which the black-box functions depended on continuous variables alone[1]. The methodology is now extended to address the case when the black-box functions depend on both continuous and strictly integer variables. The contribution of the work in this chapter is the development of a comprehensive modeling and optimization algorithm that employs each one of the kriging, RSM, direct search, and B&B methods in order to solve a more general class of MINLP containing black-box models and noise. A review of the MINLP literature for deterministic and stochastic optimization is presented in Section 4.1.1.

5.2 Problem Definition

The problem addressed in this chapter can be expressed in the following form as given in Equation (5.1):

$$
\begin{aligned}
&\textit{min} \;\; F(x, y_1, y_2, z_1, z_2) \\
&\textit{s.t.} \;\; g(x, y_1, y_2, z_1, z_2) \le 0 \\
&\qquad h(x, y_1, y_2, z_1) = 0 \\
&\qquad z_2(x, y_1) = \Gamma(x, y_1) + \varepsilon(x, y_1) \\
&\qquad \varepsilon(x, y_1) \in N(x, y_1 \mid \mu, \sigma^2) \\
&\qquad N\left(\left[x, y_1\right] \middle| \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-\left(\left[x, y_1\right] - \mu\right)^2}{2\sigma^2}\right) \\
&\qquad x \in \Re^n, y_1 \in Z^{q_1}, y_2 \in \{0,1\}^{q_2}
\end{aligned}
\tag{5.1}
$$

Based on this formulation, the vectors of continuous and strict integer variables are given by x and y_1, respectively. The variable Z^{q_1} denotes the q_1-dimensional variable subspace of the strict integer variables, in which each y_1-variable varies over a range of possible integer values. The vector of integer variables having a continuous relaxation is given by y_2. The objective function is represented by F, and the deterministic variables z_1 describe outputs whose modeling equations $h(x,y_1,y_2,z_1)$ are known. The vector of stochastic output variables z_2 exists when the input-output functionality $\Gamma(x,y_1)$ is black-box. The stochastic value of each z_2 variable is modeled as the sum of its corresponding deterministic output and an additive noise component ε that is a normally distributed error term whose mean μ and variance σ^2 may be a function of the input specification (x,y_2). For field experiments, an estimate of the parameters μ and σ^2 can be obtained by conducting replicate experiments for a given sampling vector. It should be noted that the modeled values of μ and σ^2 may need to assume a range of values if it is known from

prior field data that the noise is spatially variant with respect to the input specification (x,y_1), Synthesis equations are given by $g(x,y_1,y_2,z_1,z_2)$ which include design constraints, operating specifications, and logical relations. In the problem formulation, it is assumed that the synthesis variables appear only as part of the feasible region constraints and/or objective function.

5.3 Solution Approach

In the algorithm developed to solve the problem given by Equation (5.1), y_2 is optimized using B&B. At each node of the B&B tree, partially relaxed NLP subproblems are formulated whose solutions correspond to optimal x and y_1. The designation of "partially relaxed NLP" is termed as such because while a continuous relaxation is permitted for y_2, strict integrality must always be enforced for y_1. At the root node, the first partially relaxed NLP is generated by relaxing all y_2. An initial kriging model of the corresponding objective is constructed from a nominal sampling set S^0 in which each sampling vector $\{x,y_1,y_2\}$ is chosen at random from the set of feasible points covering the feasible region. The kriging model is subsequently updated using additional sampling information located at regions of high variance, minimal prediction, and where model behavior significantly changes over consecutive iterations. For each model, the average predictor value is compared to the corresponding one obtained at the previous iteration, and once convergence has been attained, further model refinement is terminated. The local kriging solutions are then identified as warm-start iterates for further local optimization using RSM and direct search. Since no convexity assumptions are required

when applying kriging, the set of potential local optima can be identified once an accurate global model has been obtained.

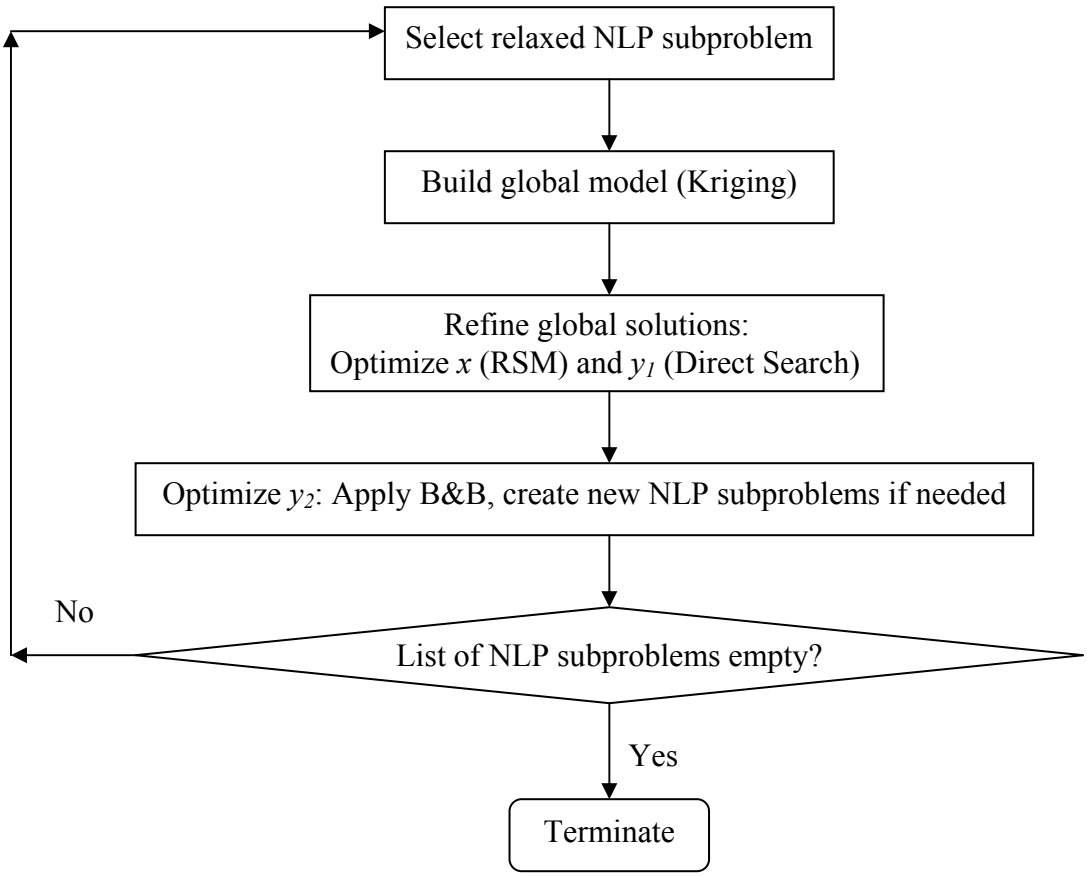


Figure 5.1. Summarized flowchart of the B&B-Kriging-RSM-DS algorithm.

For each kriging solution, the set of $\{x,y_2\}$ variables are locally optimized using RSM for fixed values of y_1. Given a vector S^K representing one of the best kriging solutions, additional sampling data are obtained from a collocation set localized around S^K. A quadratic model is fitted using least squares and then locally optimized. A new model is constructed at the new optimum, and the sequential optimization of response surfaces continues until convergence in the objective has been achieved.

At the start of the response surface algorithm, the iteration index w is initialized at a value of unity. A response surface is built around a starting iterate S^K by fitting sampling data obtained from a collocation set $S_{coll,w}$ determined by the experimental design template and local model radius b_w. The vector S^K and its corresponding objective value F^K comprise the nominal solution set $\{S_{opt,w},F_{opt,w}\}$. Once the response surface has been created, the optimum $S_{opt,w+1}$ having corresponding value $F_{opt,w+1}$ is determined using gradient methods. If the difference between the current and previous optimum $|F_{opt,w+1} - F_{opt,w}|$ falls below a prespecified criterion Tol_{RSM}, the algorithm terminates with $\{S_{opt,w+1,},F_{opt,w+1}\}$ established as the RSM solution. Otherwise, the iteration index is advanced by unity and another response surface having a new design radius b_w is constructed at the new vector $S_{opt,w}$. At any iteration w, the value of b_{w+1} is different from b_w only if the Euclidean distance between the current and previous solution vectors is lower than the current radius b_w. During the later stages of the algorithm, $S_{opt,w+1}$ will be near $S_{opt,w}$, signifying that the basin of the RSM optimum has been found. At this point, a more accurate description of the system behavior near the optimum can be attained using more localized response surfaces defined by smaller design radii b_w. Whenever iterates are close to the boundaries, lower-dimensional response surfaces are created by projecting the model onto constraints so as to prevent model generation based on an asymmetrical arrangement of the feasible sampling data[17] as described in Chapter 2. A flowchart of the algorithm is presented in Figure 5.2.

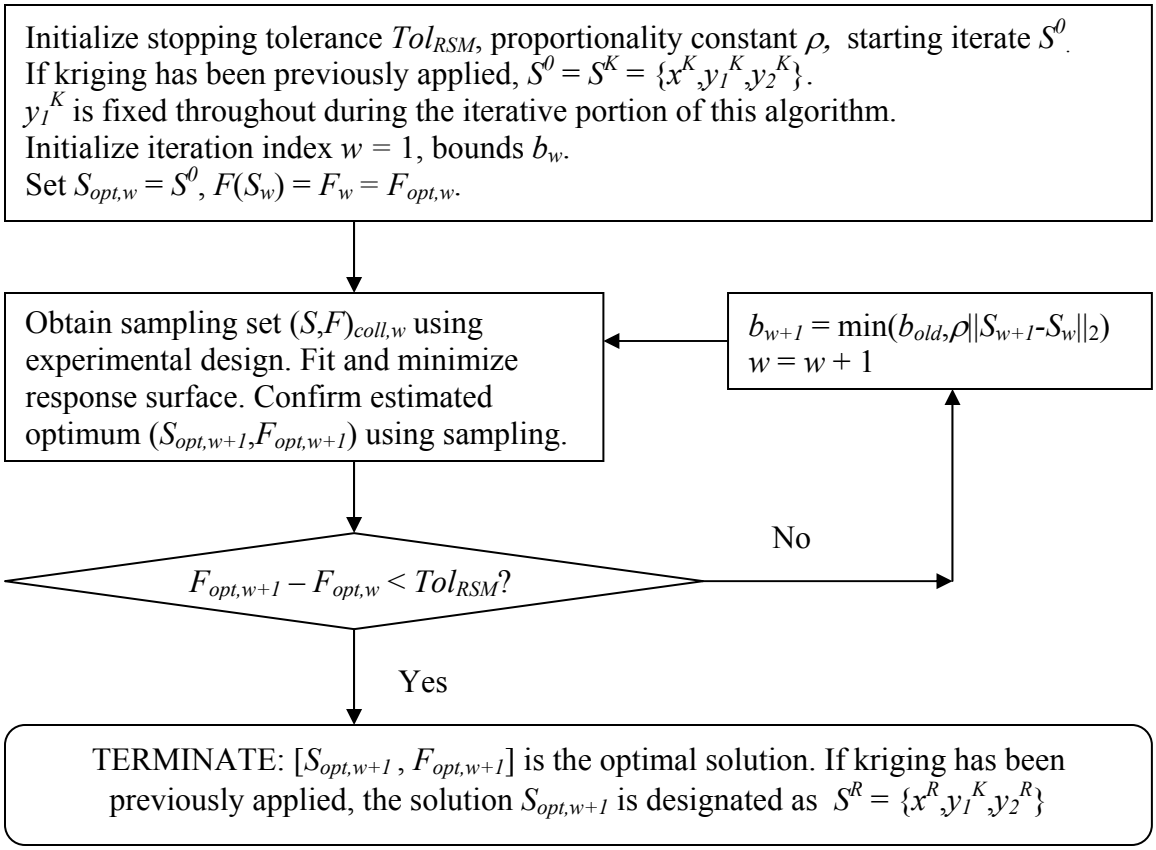


Figure 5.2. Flowchart of the RSM algorithm.

The vector $S^R = \{x^R, y_1^K, y_2^R\}$ is defined as the RSM optimal solution once the difference between the previous and current objective values, has fallen below the user-specified tolerance Tol_{RSM}. Once this occurs, the search for optimal y_1 is now initiated using one of two direct search methods. In the first method, sampling is performed at interval endpoints centered around S^R with respect to the range of each one of the y_1 - variables. The interval is defined according to a stencil whose bracket length is adaptively decreased once the neighborhood containing optimum y_1 has been found. If the initial interval length is greater than unity, smaller intervals are constructed around the current best solution. The procedure terminates when convergence in the objective is achieved or

when the newest interval length requires y_1 to assume fractional values. The vector $S^D = \{x^R, y_1^D, y_2^R\}$ is then established as the solution of the current NLP subproblem. In the second Direct Search method, a global search strategy is employed whereby the objective attained for S^R is compared to the corresponding objectives obtained from sampling at the lowest and highest feasible values for y_1. Smaller search intervals centered around the new y_1-optimal solution are generated, and sampling is conducted at the new endpoints. The best solution is again found, and the procedure is repeated until the stopping criterion has been satisfied.

The solution S^D, while optimal in x, y_1, and relaxed y_2, may not be integer feasible for y_2. Additional subproblems are therefore created when 0-1 integrality in y_2 is not satisfied. Each new subproblem is now restricted to be y_2-feasible for one of the currently noninteger y_2-variables by substituting in values of zero and unity where the given y_2-variable appears in the overall problem. For each one of the newly defined subproblems, a new feasible region, which is a subset of the original problem's feasible region, is defined. One of the new relaxed NLP subproblem is chosen for solution, with the first step requiring that the previously built kriging predictor be refined over the corresponding feasible subregion in order to determine best "warm-start" locations for further local optimization. The B&B Kriging-RSM-Direct Search algorithm terminates when the list of candidate NLP subproblems is exhausted, and the MINLP solution is established as the integer optimal solution in terms of both y_1 and y_2.

The proposed algorithm relies on the sequential optimization of the continuous and integer variables x and y_1, respectively. Because of this, the discovery of a local optimum, if not the global optimum, may prove to be more difficult if x-y_1 separability does not

exist. This problem can be addressed by 1) locally optimizing multiple kriging solutions, or 2) creating multiple kriging models using different nominal sampling sets. The second strategy alleviates the attainment of suboptimal local solutions obtained from a kriging model whose initial sampling set has been poorly selected. However, the recourse costs associated with building replicate global models or conducting extensive local sampling can become prohibitively high, and so an area of ongoing research is focused at determining an optimal sampling arrangement. The sampling-based algorithm presented in Chapter 6 is an example of one possible strategy for addressing this problem.

In the following subsections, the methodology of each one of the direct search algorithms will be presented. The reader is referred to Chapters 2, 3, and 4 for details concerning the application of the other three algorithms – RSM, kriging, and Branch-and-Bound – also employed in the comprehensive procedure used to solve the problem given by Equation (5.1).

5.3.1 Local Optimization Using Direct Search

Since direct search methods do not rely on derivative information in the search for an optimum, convergence can be slow. In order to address this, a global search algorithm is presented in addition to a local method. The local and global techniques are denoted as DS-L and DS-G, respectively. Some common features of both methods are that: 1) search is performed based on simple heuristics, and 2) each y_I variable is sequentially optimized one at a time. These methods are intended as starting points from which more sophisticated variations can be developed as a future work.

The local direct search method, DS-L, is presented first. One of the y_1-variables is selected for optimization, which has lower and upper limits given by Y^{LL} and Y^{UL}. The optimization iteration index m is initialized at unity. The current best solution Y_m is initialized at a nominal sampling vector having corresponding objective $F(Y_m)$. If a warm-start iterate has been attained using kriging and RSM, $F(Y_m)$ is set as the RSM optimum $F(S^R)$. When y_1 is multidimensional, $F(Y_m)$ is instead set as the objective value corresponding to the sampling vector containing the subset of previously optimized y_1-variables. For ease of presentation, F is shown as being a function of only one y_1-variable, since all other components of a multidimensional sampling vector are held constant.

The DS-L algorithm consists of two steps: 1) sampling at points Y_m^L and Y_m^U whose values are lower and higher than Y_m, and 2) finding the best solution of these three values as given by the following:

$$F(Y_{m+1}) = \min \{F(Y_m^L), F(Y_m), F(Y_m^U)\} \tag{5.2}$$

where Y_{m+1} is the optimal sampling point based on the set of objective function values attained at the bracket midpoint and endpoints. The iteration index m is advanced by unity and the procedure is repeated until, 1) Y_m^L and Y_m^U are found to be inferior solutions to Y_m, and 2) no unsampled points lie between Y_m^L, Y_m, and Y_m^U. At this point, the value of Y_m is fixed at its optimal solution Y^D. The DS-L method is then applied to the next y_1-variable. After the set of y_1-variables have been optimized, the procedure is terminated. The method by which Y_m^L and Y_m^U are generated for each iteration m is based on the application of a simple heuristic targeted at accelerating convergence. There are three cases to be considered, according to whether Y_m is: 1) equal to Y^{LL}, 2) equal to Y^{UL}, or

3) between Y^{LL} and Y^{UL}. The equations used to generate the triplet $\{Y_m^L, Y_m, Y_M^U\}$ are presented for each case as follows:

Case L1: Y_m is equal to Y^{LL}.

$$Y_m^L = Y_m \tag{5.3a}$$

$$Y_m = Y_m + N_m \tag{5.3b}$$

$$Y_m^U = Y_m + 2N_m \tag{5.3c}$$

Case L2: Y_m is between Y^{LL} and Y^{UL}.

$$Y_m^L = Y_m - N_m \tag{5.4a}$$

$$Y_m = Y_m \tag{5.4b}$$

$$Y_m^U = Y_m + N_m \tag{5.4c}$$

Case L3: Y_m is equal to Y^{UL}.

$$Y_m^L = Y_m - 2N_m \tag{5.5a}$$

$$Y_m = Y_m - N_m \tag{5.5b}$$

$$Y_m^U = Y_m \tag{5.5c}$$

where N_m is an integer-valued step length parameter that must be initialized. When the range of permissible values for a given y_1-variable is high, initializing N_m at a value greater than unity enables larger steps to be taken towards the optimum. In the DS-L algorithm, N_m is set to be approximately 1/10 of the feasible region range for the respective y_1-variable. As an example, in the second case study, presented in Section 5.4.2, the range of permissible values for the first y_1-variable is given by [1,9], and so N_m is initialized at unity. In contrast, the range of permissible values for the second, third, and fourth y_1 are given by [1,72], so N_m is initialized at ten. As the optimization progresses, the m^{th} iterate Y_m may be located near a boundary. Based on the current value

of N_m, it is possible for the corresponding bracket endpoints Y_m^L and Y_m^U to be defined at infeasible sampling points. To address this, a new value of N_{m+1} is prescribed at each iteration m in order to ensure a symmetrical sampling arrangement as given by Equation (5.6):

$$N_{m+1} = min\{N_m, Y_m - Y_m^L, Y_m^U - Y_m\} \tag{5.6}$$

When the objective value corresponding to the bracket midpoint is found to be lower than the endpoint solutions, the vicinity of the optimum has been located. The value of N_{m+1} is then halved as given by (5.7), in order to refine the optimum using a sequence of smaller brackets.

$$N_{m+1} = \lceil 0.5 N_m \rceil \tag{5.7}$$

Once the midpoint-endpoint bracket length has fallen to unity and no unsampled points remain between Y_m^L, Y_m, and Y_m^U, the *DS-L* algorithm is terminated at the optimal solution. The next y_1-variable is selected for optimization and the *DS-L* procedure is repeated until optimal values have been found for all y_1-variables. The complete y_1-optimal solution set is referred to as y_1^D. A flowchart of the methodology is presented in Figure 5.3.

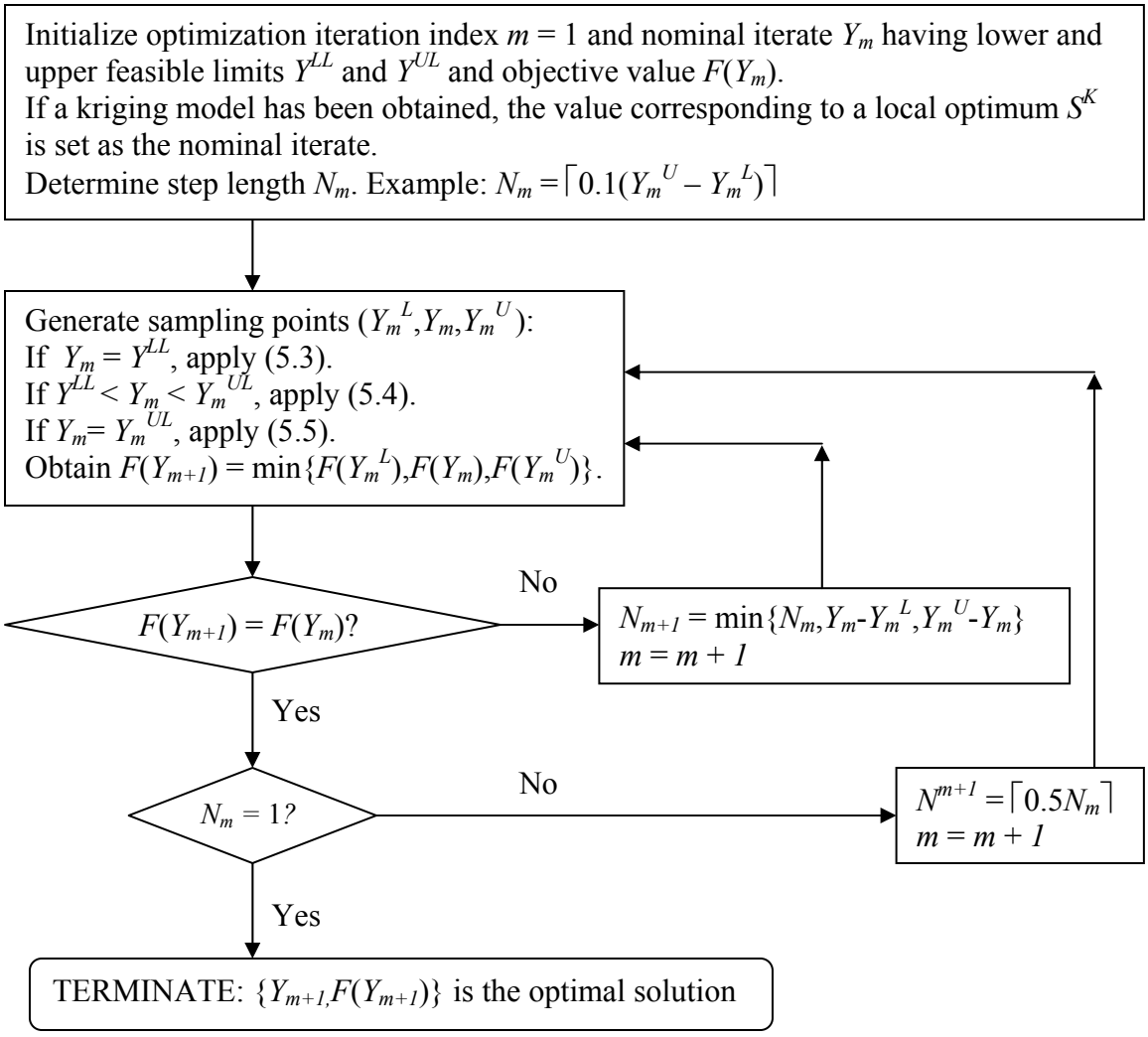


Figure 5.3. Flowchart of the local direct search (DS-L) algorithm employed for the optimization of a single y_1 variable.

5.3.2 Global Optimization Using Direct Search

The main limitation of the DS-L algorithm is that the final solution may not be globally optimal. In order to avoid terminating at a local solution, the DS-G algorithm can instead be applied. The DS-G method relies on global search and differs from its DS-L counterpart in the mechanism by which Y_m^L, Y_m, and Y_m^U are defined. First, the

iteration index m and the current solution Y_m are initialized in the same way as presented for the DS-L method. Nominal bracket endpoints Y_m^L and Y_m^U are initialized at the feasible region boundaries Y^{LL} and Y^{UL}. The bracket midpoint Y_m^C is defined between Y^{LL} and Y^{UL}, with the ceiling function being applied to ensure integrality satisfaction. After sampling is performed at the bracket endpoints and midpoint, the current solution $F(Y_m)$ is updated as given by Equation (5.8):

$$F(Y_{m+1}) = \min\{F(Y_m^L), F(Y_m^C), F(Y_m^U), F(Y_m)\} \tag{5.8}$$

The next step is to define a new bracket over which the global optimum is expected to be found. If the nominal solution at Y_m continues to be the best solution, the new bracket is defined by applying Cases G1 or G2, depending upon whether the nominal value is less than or greater than the initial bracket midpoint $F(Y_m^C)$. Conversely, if the best solution is attained at the bracket midpoint or endpoints, the new bracket is defined by applying Cases G3, G4, or G5, respectively.

Case G1: $F(Y_{m+1}) = F(Y_m),\ Y_m^L < Y_m < Y_m^C$

$$Y_{m+1}^L = Y_m - \min([Y_m^C - Y_m],[Y_m - Y_m^L]) \tag{5.9a}$$

$$Y_{m+1}^C = Y_m \tag{5.9b}$$

$$Y_{m+1}^U = Y_m + \min([Y_m^C - Y_m],[Y_m - Y_m^L]) \tag{5.9c}$$

Case G2: $F(Y_{m+1}) = F(Y_m),\ Y_m^C < Y_m < Y_m^U$

$$Y_{m+1}^L = Y_m - \min([Y_m^U - Y_m],[Y_m - Y_m^C]) \tag{5.10a}$$

$$Y_{m+1}^C = Y_m \tag{5.10b}$$

$$Y_{m+1}^U = Y_m + \min([Y_m^U - Y_m],[Y_m - Y_m^C]) \tag{5.10c}$$

Case G3: $F(Y_{m+1}) = F(Y_m{}^L)$

$$Y_{m+1}{}^L = Y_m{}^L \tag{5.11a}$$

$$Y_{m+1}{}^C = \lceil 0.5(Y_m{}^L + Y_m{}^C) \rceil \tag{5.11b}$$

$$Y_{m+1}{}^U = Y_m{}^C \tag{5.11c}$$

Case G4: $F(Y_{m+1}) = F(Y_m{}^C)$

$$Y_{m+1}{}^L = \lceil 0.5(Y_m{}^L + Y_m) \rceil \tag{5.12a}$$

$$Y_{m+1}{}^C = Y_m{}^C \tag{5.12b}$$

$$Y_{m+1}{}^U = \lceil 0.5(Y_m + Y_m{}^U) \rceil \tag{5.12c}$$

Case G5: $F(Y_{m+1}) = F(Y_m{}^U)$

$$Y_{m+1}{}^L = Y_m \tag{5.13a}$$

$$Y_{m+1}{}^C = \lceil 0.5(Y_m{}^L + Y_m{}^U) \rceil \tag{5.13b}$$

$$Y_{m+1}{}^U = Y_m{}^U \tag{5.13c}$$

Once the new bracket has been generated, the iteration index m is advanced by unity and additional sampling is performed as necessary. The best solution $F(Y_{m+1})$ is again determined based on the bracket endpoint and midpoint solutions, and the procedure is repeated until convergence in the objective has been achieved. In order to increase the probability of finding the global optimum Y^D, a set of brackets can be defined for each new iteration which enclose both the optimal and near-optimal solutions. The additional sampling expense is offset by the increased chances of finding a solution that, at worst, is equivalent to the one obtained using the local method, and at best, is the global optimum. The procedure is then repeated for the remaining y_1-variables and the complete y_1-optimal solution set is again given as $y_1{}^D$. A flowchart of the DS-G method is presented in Figure 5.4.

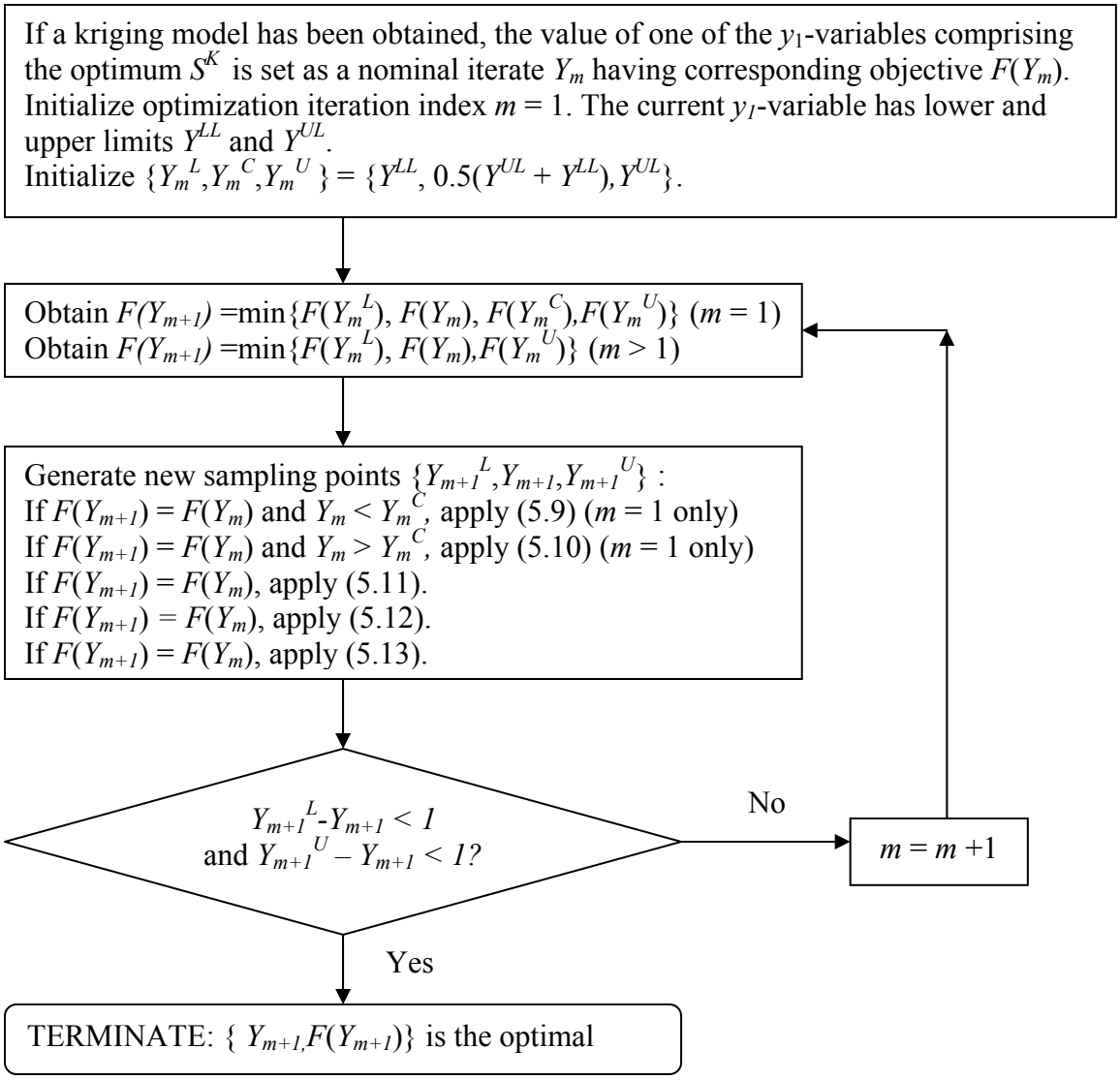


Figure 5.4. Flowchart of the global direct search (DS-G) algorithm employed for the optimization of a single y_1 variable.

The sequential application of kriging, RSM, and Direct Search enables optimal process design and operations policies to be determined for problems containing variable classes x and y_1. When y_2-variables are present, the application of B&B, the methodology

of which is presented in Chapter 4, enables integer optimal y_2 to be obtained since a continuous relaxation exists for this class of variables.

Each one of the B&B, Kriging, RSM, and Direct Search algorithms target the optimization of a different variable class – y_2, x-y_1-y_2, x, and y_1, respectively. The four algorithms are now combined into a comprehensive B&B Kriging-RSM-Direct Search algorithm addressing the MINLP formulated in Equation (5.1). The steps of this algorithm are presented in the next section.

5.3.3 B&B Kriging-RSM-Direct Search Algorithm

The first step of the B&B Kriging-RSM-Direct Search algorithm is to formulate a partially relaxed NLP subproblem that has been relaxed in the y_2-variables. A nominal sampling set is specified and the initial kriging model is built. The model is then iteratively refined based on sampling at: 1) regions of high uncertainty; 2) regions where the minimum value of the objective lies; and 3) regions where significant model change is observed over consecutive iterations, until a stopping criterion has been satisfied. The stopping criterion can be resource-based such as an upper limit on the number of computer or field experiments performed, or model-based such as when convergence has been achieved in the average prediction value. After the stopping criterion has been satisfied, the model predictor values are ranked, and the locally optimal kriging solutions are identified. A kriging solution $S^K = (x,y_1,y_2)^K$ is then selected to be a "warm-start" iterate for local optimization using RSM, the methodology of which is given by the flowchart given in Figure 5.2. The x-y_2 components of the RSM-optimal solution $S^R = (x^R,y_1^K,y_2^R)$ are then fixed, and the y_1-variables are then optimized using the local or

global direct search methods presented in Figures 5.3 or 5.4, respectively. Once convergence in the objective has been attained, the now y_1-optimized solution is given as $S^D = (x^R, y_1^D, y_2^R)$ and the next kriging solution S^K is then locally optimized. Once all kriging solutions have been refined, the partially relaxed NLP solution S_{NLP}^{opt} is designated as the best S^D solution attained, which has corresponding objective value F_{NLP}^{opt}. Due to resource constraints, either in the form of field experiment costs or long simulation times for computer experiments, refinement of multiple local kriging solutions may need to be restricted to a single or subset of the warm-start iterates. For the two examples presented in Section 3, the best kriging solution is the only warm-start iterate that is further refined using RSM/Direct Search.

The final step is to optimize the y_2-variables using B&B in order to obtain the integer optimal solution in y_2. If integer feasibility is not satisfied for the y_2-variables, F_{NLP}^{opt} is classified as a LB and new NLP subproblems are formulated by enforcing integrality on a subset of the y_2-variables. Kriging, RSM, and Direct Search are applied to each new NLP subproblem, and additional LB/UB are established depending on whether the corresponding solution is integer feasible in y_2. New partially relaxed NLP subproblems are created when a solution designated as a LB is superior to the current best UB. Once the list of candidate NLP subproblems is exhausted, or another stopping criterion has been achieved, such as when the LB/UB integrality gap has fallen below the stopping tolerance Tol_{BB}, the procedure is terminated. The y_1-y_2 integer optimal solution vector S_{MINLP}^{opt}, which has corresponding objective F_{MINLP}^{opt}, is then defined as the one for which the lowest UB solution has been attained. The proposed methodology of the unified

algorithm is shown in a flowchart as given by Figure 5.5. In the next section, this algorithm is applied to two industrial case studies to demonstrate proof of concept.

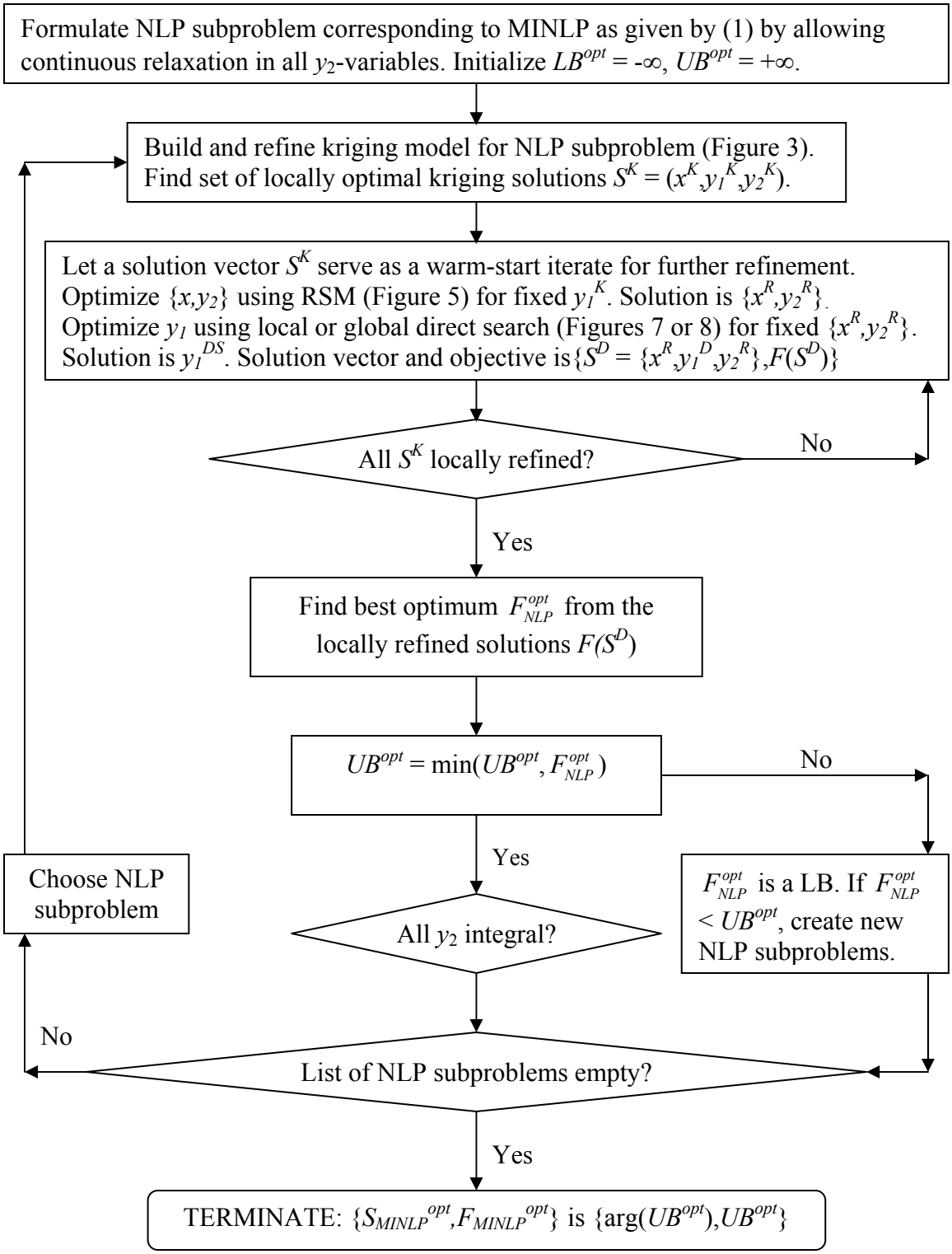


Figure 5.5. Detailed flowchart of the B&B Kriging-RSM-Direct Search algorithm.

5.4 Examples

In this section, the local and global direct search methods are applied within the framework of the Kriging-RSM optimization algorithm in order to identify system optima for two industrial case studies. In the first example, presented in subsection 5.4.1, the objective is to determine a set of process synthesis and design specifications minimizing the monthly operating costs for *t*-butyl methacrylate (*t*-BMA) production[51]. In the second example, presented in subsection 5.4.2, the objective is to determine optimal design specifications for alcohol dehydrogenase manufacture[52]. If kriging, RSM and local direct search are employed to determine the optimal solution, an algorithmic designation of K-R-L is used. Similarly, if kriging, RSM, and global direct search are used, the corresponding designation of K-R-G is employed. For each example, a table of solution information is provided based on application of both the K-R-L and K-R-G algorithms. Since the first problem includes synthesis variables, B&B is also used to obtain the integer optimal y_2-variables. All results are obtained using an HP dv8000 CTO Notebook PC containing a 1.8 GHz AMD Turion 64 processor.

5.4.1. *t*-Butyl Methacrylate Synthesis

t-butyl methacrylate (*t*-BMA) is a monomer used in the production of both industrial and household coatings. For this example, the objective is to minimize the total cost required to satisfy a monthly demand of 457,874 kg 97.5% *t*-BMA. The total cost is the sum of in-house production costs and third-party purchase at $1.10/kg *t*-BMA. In-house production costs are defined as the sum of raw material (RM) and utilities costs. The RM employed in *t*-BMA production are isobutylene (IB) and methacrylic acid (MA) and the

costs for each are \$0.88/kg and \$0.17/kg, respectively. The post-reaction product is a four-component mixture that enters a three-cut separation train. The utilities cost component of the production costs is defined as the sum of the cooling water, chilled water, refrigerant, and steam costs required for the separation train. The problem is formulated as an MINLP that contains synthesis decisions describing the distillation tower sequencing for *t*-BMA purification, and design decisions representing the amount of fresh RM feed, column reflux, reboiler temperature, and column feed tray location. The continuous process is initiated by feeding methacrylic acid (MA) and isobutylene (IB) to a 6.7 m^3 reactor pressurized at 3.426 bar in which the following reactions occur:

$$IB + MA \leftrightarrow t\text{-BMA}$$

$$IB \rightarrow DIB$$

The first reaction describes the reversible acid-catalyzed production of *t*-BMA and the second reaction represents IB dimerization resulting in di-isobutylene (DIB) formation, an unwanted side product. The conversion achieved is approximately 45%, after which caustic 50% NaOH solution is then fed to the reactor to neutralize the H_2SO_4 catalyst. The principal reaction product, comprised of IB, DIB, *t*-BMA, and MA, is flashed at 1.01 bar to remove the majority of IB. The remaining process operations involve further *t*-BMA purification from IB, DIB, and MA using distillation or flash operations. At the end of the purification train, DIB exits as a waste product, *t*-BMA is either sold off or used elsewhere in the plant, and both IB and MA are recycled back to the reactor. The problem is solved using ChemCAD simulation-based optimization subject to the following conditions that are relaxed in the original presentation: 1) application of nonideal thermodynamic models, 2) application of rigorous distillation tower calculations

leading to nonsharp separations, 3) perturbed column distillate and bottoms flowrates that are noisy in order to simulate nonideal process operation, 4) reduction in the amount of IB/MA available for raw materials purchase, thereby limiting maximum production, 5) application of two towers and a flash drum for the separation cascade instead of three towers, and 6) optimization of column feed tray location. The process behavior of the reactor, flash drum, and rigorous distillation column simulation units are considered to be black-box legacy codes since the design equation models are not directly accessible using the ChemCAD simulator. A superstructure of the problem is presented in Figure 5.6 where there are six possible sequences. The notation α/β is used to denote the task of generating a distillate α using a tower consisting of β trays.

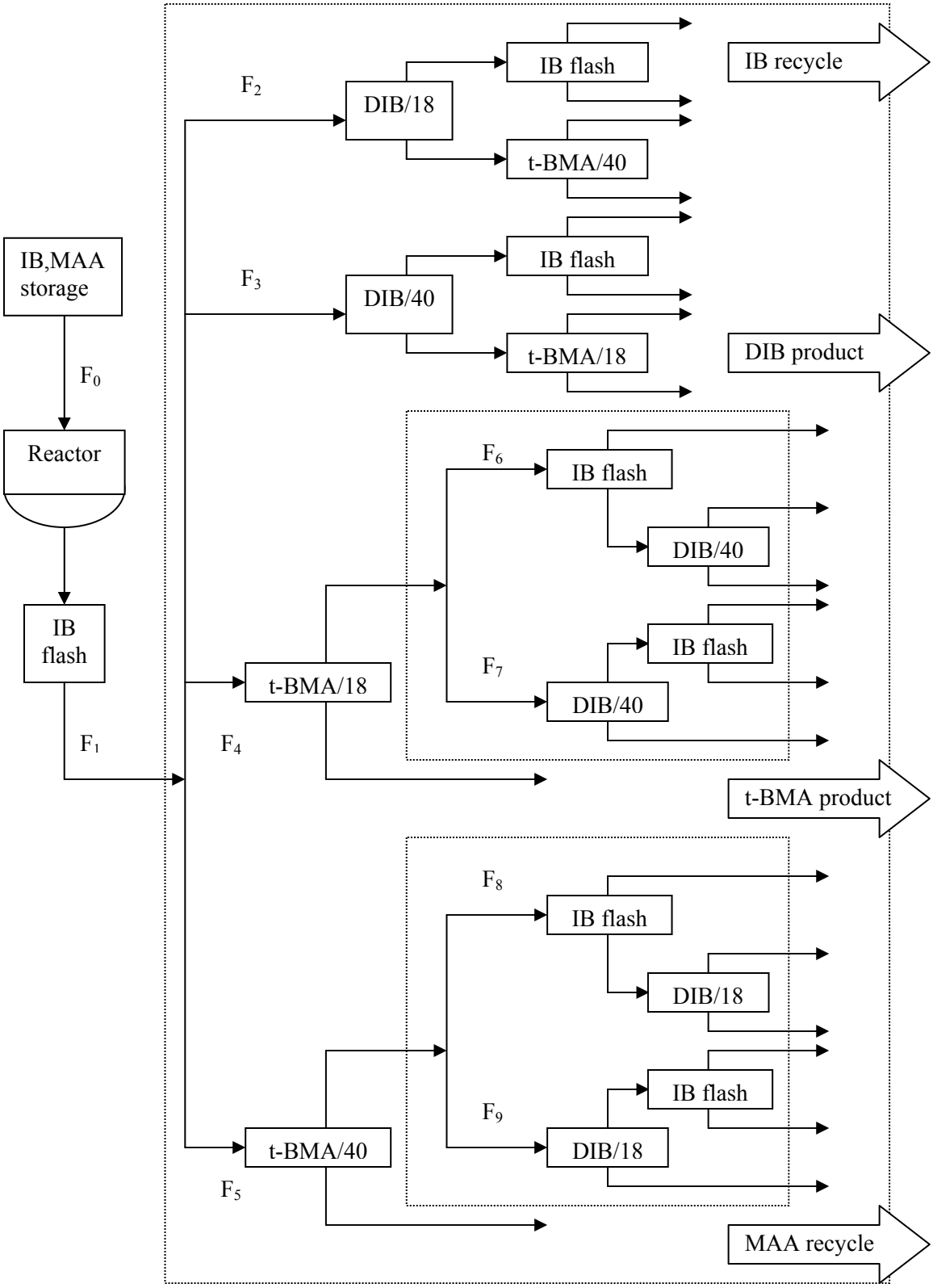


Figure 5.6. Superstructure of t-BMA separation train.

Synthesis and design inputs specifications are provided to the simulator. The synthesis decisions are given in terms of a vector of 0-1 variables specifying the sequencing of the IB/DIB, DIB/*t*-BMA, and *t*-BMA/MA separation tasks. Design decisions are given in terms of continuous variables describing the amount of IB/MA fed to the reactor, column reflux, and bottoms product temperatures, and integer variables denoting column feed tray location. Once the process utilities and product flowrates have been obtained, the operating cost is then determined. There is a tradeoff between *t*-BMA production and the cost of MA/IB purchase. Since the raw materials are expensive, a reduced production of *t*-BMA may be more economical. However, the manufacturing deficiency must then be supplemented by the purchase of additional *t*-BMA in order to meet the required monthly demand. The objective function is therefore given as the sum of raw material (including *t*-BMA) and utility costs. The MINLP is formulated as shown in (5.14) and accompanying cost and thermophysical data are presented in Table 5.1:

$$\min \quad F = 0.17F_{IB} + 0.88F_{MA} + 0.0265F_{H2SO4} + 0.033F_{NaOH} + 1.10F_{t\text{-}BMA} + 0.26V_{CW} + 0.52V_{ChW} + 5.28V_{Ref} + 2.2M_{Stm} \quad (5.14a)$$

$$s.t. \quad F_{IB} + F_{MA} + F_{H2SO4} + F_{NaOH} = F_0 \quad (5.14b)$$

$$F_1 - F_2 - F_3 - F_4 - F_5 = 0 \quad (5.14c)$$

$$F_4 - F_6 - F_7 = 0 \quad (5.14d)$$

$$F_5 - F_8 - F_9 = 0 \quad (5.14e)$$

$$F_i - F_0 y_i \leq 0 \qquad i = 2 \ldots 9 \quad (5.14f)$$

$$F_i \geq 0 \qquad i = 2 \ldots 9 \quad (5.14g)$$

$$y_2 + y_3 + y_4 + y_5 = 1 \quad (5.14h)$$

$$y_4 - y_6 = 0 \quad (5.14i)$$

$$y_4 - y_7 = 0 \tag{5.14j}$$

$$y_5 - y_8 = 0 \tag{5.14k}$$

$$y_5 - y_9 = 0 \tag{5.14l}$$

$$V_{CW} = \frac{\sum Q_{CW}}{\rho_{CW} C_{p,CW} \Delta T_{CW}} \tag{5.14m}$$

$$V_{ChW} = \frac{\sum Q_{ChW}}{\rho_{ChW} C_{p,ChW} \Delta T_{ChW}} \tag{5.14n}$$

$$V_{Ref} = \frac{\sum Q_{Ref}}{\rho_{Ref} C_{p,Ref} \Delta T_{Ref}} \tag{5.14o}$$

$$M_{Stm} = \frac{\sum Q_{Stm}}{\Delta H_{Stm}} \tag{5.14p}$$

$$D_{jk} = \Gamma(RR_k\ ,\ T_{Bot,k},\ Q_{CW,k},\ Q_{ChW,k},\ Q_{Ref,K},\ Q_{Stm,k},\ F_{jk}) \tag{5.14q}$$

$$B_{jk} = \Gamma(RR_k\ ,\ T_{Bot,k},\ Q_{CW,k},\ Q_{ChW,k},\ Q_{Ref,K},\ Q_{Stm,k},\ F_{jk}) \tag{5.14r}$$

$$D_{jk}^{noisy} = (1\text{-}\sigma U(0,1))D_{jk} + \sigma U(0,1)B_{jk} \tag{5.14s}$$

$$B_{jk}^{noisy} = (1\text{-}\sigma U(0,1))D_{jk} + \sigma U(0,1)D_{jk} \tag{5.14t}$$

$$\begin{aligned} &j = \{IB, DIB, t-BMA, MA\} \\ &k = \{DIB/18, DIB/40, t-BMA/18, t-BMA/40\} \\ &y_i = \{0,1\} \qquad i = 2...9 \\ &\sigma = 0 \\ &0.3 \le \omega \le 0.7 \\ &0.2 \le RR_k \le 30 \\ &290 \le T_{Bot,k} \le 351 \end{aligned} \tag{5.14u}$$

Table 5.1. Cost and thermophysical data for the t-BMA methacrylate synthesis case study.

i	Cost of RM i, \$/Unit	Unit	ΔH_i, [kJ/kg]	T_b [K]	ρ_i, [kg/m^3]	$C_{p,i}$ [kJ/(kg K)]	ΔT_i [K]
IB	88.19	100 kg					
MA	16.98	100 kg	----	----	----	----	----
H_2SO_4	2.65	100 kg	----	----	----	----	----
NaOH	3.3	100 kg	----	----	----	----	----
t-BMA	110	100 kg	----	----	----	----	----
Saturated Steam	2.2	1000 kg	2000	457.16	----	----	----
Cooling Water	0.26	10,000 L	----	293.15	1000	4.84	20
Chilled Water	0.52	10,000 L	----	277.6	1000	4.184	20
Refrigerant	5.28	10,000 L	----	177.6	683	1.74	20

The first four terms in the objective function represent raw materials costs related to production. The fifth term represents the cost required for additional *t*-BMA purchase whenever the manufacturing scheme fails to satisfy the monthly demand over the 720-hr operating period. The last four terms of the objective describe utility expenses and are comprised of cooling water, chilled water, refrigerant, and steam costs, respectively. The binary variables y_i describe column or flash unit existence as shown in Figure 5.6 and the variables F_i describe corresponding feed flowrates. The first equation is a RM mass balance, the next ten equations describe the feasible *t*-BMA separation syntheses and the subsequent four equations describe the required utilities consumption. The next two equations represent distillate and bottoms flowrates from separation towers whose model equations are treated as black-box. The final two equations model the way in which noisy distillate and bottoms flowrates D_{jk}^{noisy} and B_{jk}^{noisy} are obtained. The operating ranges for

column reflux and bottoms product temperatures are given with respect to the range of values allowed for all possible separations. The parameter ω is a scaling factor used to restrict the amount of MA/IB fed to the reactor with respect to the specified values in the original study. This parameter is used to simulate the scenario in which raw material availability is limited and complicates the problem by introducing the possibility that additional *t*-BMA may need to be expensively purchased if the reaction product throughput falls short of the demand. The kriging model is built to describe the objective function behavior in terms of the synthesis and design inputs. This global model is then used to guide search towards candidate vectors whose manufacturing schemes require lower utility requirements. B&B is used to determine the optimal *t*-BMA separation sequence from IB/DIB/MA, RSM is used to determine the optimal amount of fresh IB/MA reactor feed, column reflux and reboiler temperatures, and Direct Search (DS) is used to obtain the optimal column feed tray locations.

In Table 5.2, the optima information is provided based on a computational sampling limit of three hundred CHEMCAD simulations. The sampling limit is imposed due to the long computational time associated with attaining convergence in the process superstructure containing ten rigorous TOWR distillation columns. Figure 5.7 presents the Branch-and-Bound tree for one of the optimization trials, which illustrates the mathematical mechanism by which the optimal set of synthesis variables $y_2 \ldots y_9$ are obtained. The optima information for the continuous and black-box variables which correspond to the integer solution obtained at Node 3 of the B&B tree are shown in Table 5.2. The optimal process synthesis consists of, 1) feeding the reaction product to the 18-tray column at the 5^{th} tray to remove IB/DIB, 2) flashing the IB/DIB to remove IB, and 3)

feeding the 18-tray column bottoms to the 40-tray column at the 11th tray location, resulting in the recovery of the t-BMA as a distillate and MA as the bottoms. Both the DIB and MA are recycled to the reactor.

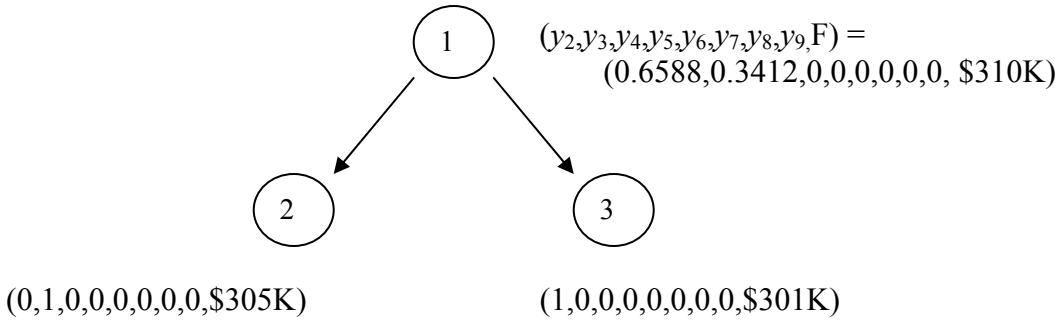


Figure 5.7. Branch-and-Bound tree information showing how the integer optimal process synthesis variables y_2 are determined for the t-BMA separation case study.

Table 5.2. Optima information for the continuous variables at Node 3 of the B&B tree shown in Figure 5.7.

Variable Type	Variable Description	Optimal Values
x_1	Ratio describing the amount of fresh IB/MA fed to reactor relative to the maximum amount that could be fed to the reactor assuming unlimited RM availability	0.5
x_2	DIB/18 column reflux ratio	0.2718
x_3	DIB/18 column reboiler temperature, K	51.44
x_4	t-BMA/40 column reflux ratio	1.513
x_5	t-BMA/40 column reboiler temperature, K	60.57
$y_{1,1}$	Feed Tray (DIB/18 column)	5
$y_{1,2}$	Feed Tray (t-BMA/40 column)	11
$z_{2,1}$	Steam Duty (kg/h)	480.66
$z_{2,2}$	Volume CW (m^3/h)	15.25
$z_{2,3}$	Volume ChW (m^3/h)	10.35
$z_{2,4}$	Volume Refrigerant (m^3/h)	9.525
$z_{2,5}$	IB/MA/H_2SO_4/NaOH RM Costs ($/hr)	464
$z_{2,5}$	IB/MA/H_2SO_4/NaOH Utilities Costs ($/hr)	7.02
$z_{2,5}$	Third-party t-BMA Costs ($/hr)	$0

In Table 5.3, the performance of the K-R-L and K-R-G methods as applied in obtaining the optimal conditions for *t*-BMA production is presented for a set of fifty trials.

Table 5.3. Optimization results obtained for the *t*-BMA methacrylate case study (No Noise), based on application of the B&B Kriging-RSM-DS algorithm.

Method	# sampling points used in building kriging model (Initial,Final)	Optimum monthly cost F_{MINLP}^{opt}	% trials for which F_{MINLP}^{opt} is attained	Sim. Req.	Sim. CPU Time (s)	Modeling and optimization CPU time (s)		
						K	R	L/G
K-R-L	(10,18)	$299,264	94%	150	3658	35	12	3
K-R-L	(15,23)	$299,530	84%	154	3224	38	14	4
K-R-L	(20,28)	$299,017	96%	197	4580	42	13	3
K-R-L	(30,38)	$299,428	81%	224	5508	46	13	3
K-R-G	(10,18)	$299,568	94%	142	3503	33	11	3
K-R-G	(15,23)	$299,314	88%	171	3182	40	12	3
K-R-G	(20,28)	$299,314	84%	159	4081	41	13	4
K-R-G	(30,38)	$299,455	100%	188	5514	48	13	3

The first column contains the sequence of global and local algorithms used in combination with B&B to find the optimal monthly cost F_{MINLP}^{opt}.The second column contains the initial and total number of sampling points used to build the kriging predictor, respectively. The third and fourth columns contain the expected monthly cost and the rate of successful convergence to the optimum, respectively. The success rate is based on the application of the K-R-L or K-R-G algorithms, in combination with B&B, for fifty different trials. For each trial, a unique sampling set, obtained from randomly choosing the initial number of feasible vectors, is employed to build a nominal kriging predictor of the relaxed NLP objective at the root node of the B&B tree. The fifth and sixth columns contain the average number of simulations and corresponding simulation

CPU time. The seventh, eighth, and ninth columns contain the CPU cost associated with the application of the kriging, RSM, and local or global direct search algorithms.

It is observed that an increase in the number of sampling points used to build the nominal kriging predictor does not always increase the chances of finding the optimum. When the K-R-L algorithm is applied, the optimum is found in 84% of the cases when fifteen points are used to build the initial global model, compared with 96% of the cases when only ten points are used. Similar results are observed for the performance of the K-R-G algorithm; the optimum is found in 94% and 88% of the cases when ten and fifteen sampling points are used, respectively.

When the Direct Search algorithms are compared in terms of the number of sampling points used to build the initial global model, the global search algorithm, DS-G, slightly outperforms its local search counterpart, DS-L, in terms of the number of simulations required. This behavior is observed regardless of whether ten, fifteen, twenty, or thirty sampling points are used to build the nominal kriging predictor, suggesting that the global search feature of the DS-G algorithm results in faster discovery of the optimum. Consider a global model initially built using ten sampling points. The optimum is found after an average of 142 simulations when global search is applied, in contrast to 150 simulations required by local search. The monthly cost reported in Table 5.3 represents the total cost required to satisfy the monthly demand of 457,874 kg *t*-BMA. Based on the corresponding optimal synthesis and operating conditions, third-party purchase of *t*-BMA is not required. The in-house production cost of *t*-BMA is \$0.65/kg, a 41% savings achieved when compared to the third-party cost of \$1.10/kg.

The sequential application of B&B, kriging, RSM, and direct search is again applied to the problem for varying nominal sampling set sizes, with the only change now being that process noise has been introduced into the reactor, flash unit, and distillation columns. The value of σ is now set at 0.05. The unified B&B Kriging-RSM-Direct Search algorithms are again applied based on the same eight conditions given in Table 5.3, in which optimal y_1 is determined by local or global search, and in which the number of sampling points used in building the nominal kriging predictor varies between ten and thirty. The algorithm is applied to twenty-five different sampling sets for each of the eight conditions.

Table 5.4. Optimization results obtained for the t-BMA methacrylate case study (With Noise), based on application of the B&B Kriging-RSM-DS algorithm.

Algorithm	# sampling points used in building kriging model (Initial,Final)	Optimum monthly cost F_{MINLP}^{opt}	% Trials for which F_{MINLP}^{opt} is attained	Sim. Req.	Sim. CPU Time (s)	Modeling and optimization CPU time (s)		
						K	R	L/G
K-R-L	(10,18)	$304,318	80%	188	4737	34	2	8
K-R-L	(15,23)	$304,026	84%	161	3016	39	2	9
K-R-L	(20,28)	$304,479	84%	184	3373	33	1	7
K-R-L	(30,38)	$303,968	84%	235	3884	50	2	9
K-R-G	(10,18)	$304,453	72%	157	2732	33	2	9
K-R-G	(15,23)	$304,030	88%	213	3799	36	2	8
K-R-G	(20,28)	$303,883	88%	202	3593	37	2	8
K-R-G	(30,38)	$303,334	72%	278	4943	44	2	9

The value of the optimal objective F_{MINLP}^{opt} is slightly inferior to the value obtained when no noise is present, although the relative production cost of *t*-BMA is $0.664/kg still results in a 39.7% savings in comparison to the third-party price. Successful

convergence to the optimum is generally observed in fewer cases when noise is present regardless of whether local or global direct search is used to determine optimal y_1. However, the comparison is not quite fair since 1) results for the noisy conditions are based on twenty-five trials instead of fifty, and 2) for any given sampling size, a nominal sampling set employed under noisy conditions might be completely different from the sampling sets used when noise was absent. It is observed that the addition of noise for a given sampling set can lead to convergence problems being encountered for downstream units in the simulator, so new nominal sampling sets need to be employed when this problem occurs. The number of simulations required to achieve convergence to F_{MINLP}^{opt} is also generally higher under noisy conditions. Since the number of sampling points employed for global modeling is the same in each of the eight conditions, the additional sampling costs are attributed to local optimization. This suggests, in turn, that the objective values corresponding to the deterministic kriging solutions S^K were lower than those found for the stochastic conditions, indicating that the global geometry was modeled less accurately when noise was present.

For the K-R-L algorithms, as the nominal sampling size increases, a modest 4% improvement in the attainment of F_{MINLP}^{opt} is observed. For the corresponding K-R-G algorithm, a 16% improvement is obtained for sampling sizes of fifteen and twenty points, although the success rate corresponding to ten points for initial global modeling is only 72%. Surprisingly, when the K-R-G method is applied to thirty points, the success rate also decreases to 72%. Since the nominal sampling sets are randomly generated for all cases, this suggests the presence of sampling data which contributes redundant information to the kriging model, and that the success rate could be improved by focusing

on an algorithm specifically focused on the identification of sampling point vectors. In Chapter 6, a centroid-based sampling strategy is proposed to address this problem.

5.4.2. Alcohol Dehydrogenase Purification

The second example is taken from the biochemical engineering literature[52]. Alcohol dehydrogenase (ADH) is an enzyme that is used to convert alcohols to aldehydes and ketones. Since ADH is a naturally occurring chemical found within *S. cerevisiae* cells, it can be produced on a commercial scale via industrial continuous or fed-batch fermentation. However, the primary difficulty is ADH acquisition and purification, since ADH must first be released from the cells, and then separated from the cellular debris. ADH is temperature sensitive and degrades above temperatures of 40^{o}F. There are four main process operations to be considered in ADH production, which are also typical steps for a standard enzyme process: fermentation, homogenization, centrifugation, and precipitation. In this case study, *S. cerevisiae* cells are grown in a fermenter using fed-batch operations. The cells are then disrupted using homogenization, an operation that consists of passing the cells through a blender in order to disrupt the cell walls, enabling the release of intracellular components. In the centrifugation step, the ADH is separated from an intracellular protein contaminant. The ADH leaves in the centrifuge supernatant and residual protein is then precipitated using ammonium sulfate. In order to improve ADH purity and yield, the centrifugation and precipitation operations are repeated. A schematic of the homogenizer and centrifuge are presented in Figures 5.8 and 5.9, followed by a complete process flowsheet shown in Figure 5.10.

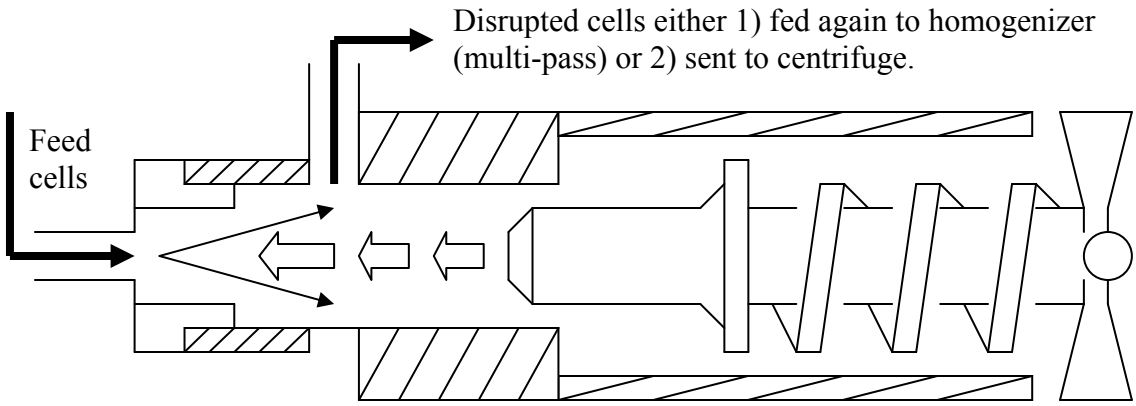


Figure 5.8. Detailed homogenizer schematic.

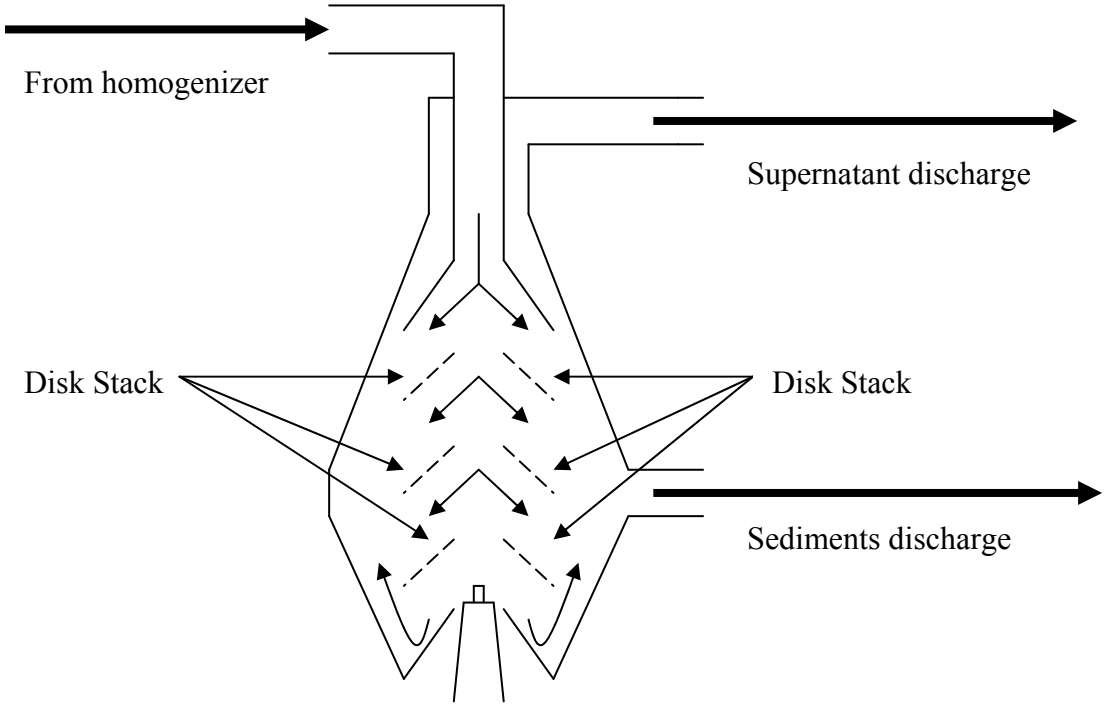


Figure 5.9. Detailed centrifuge schematic.

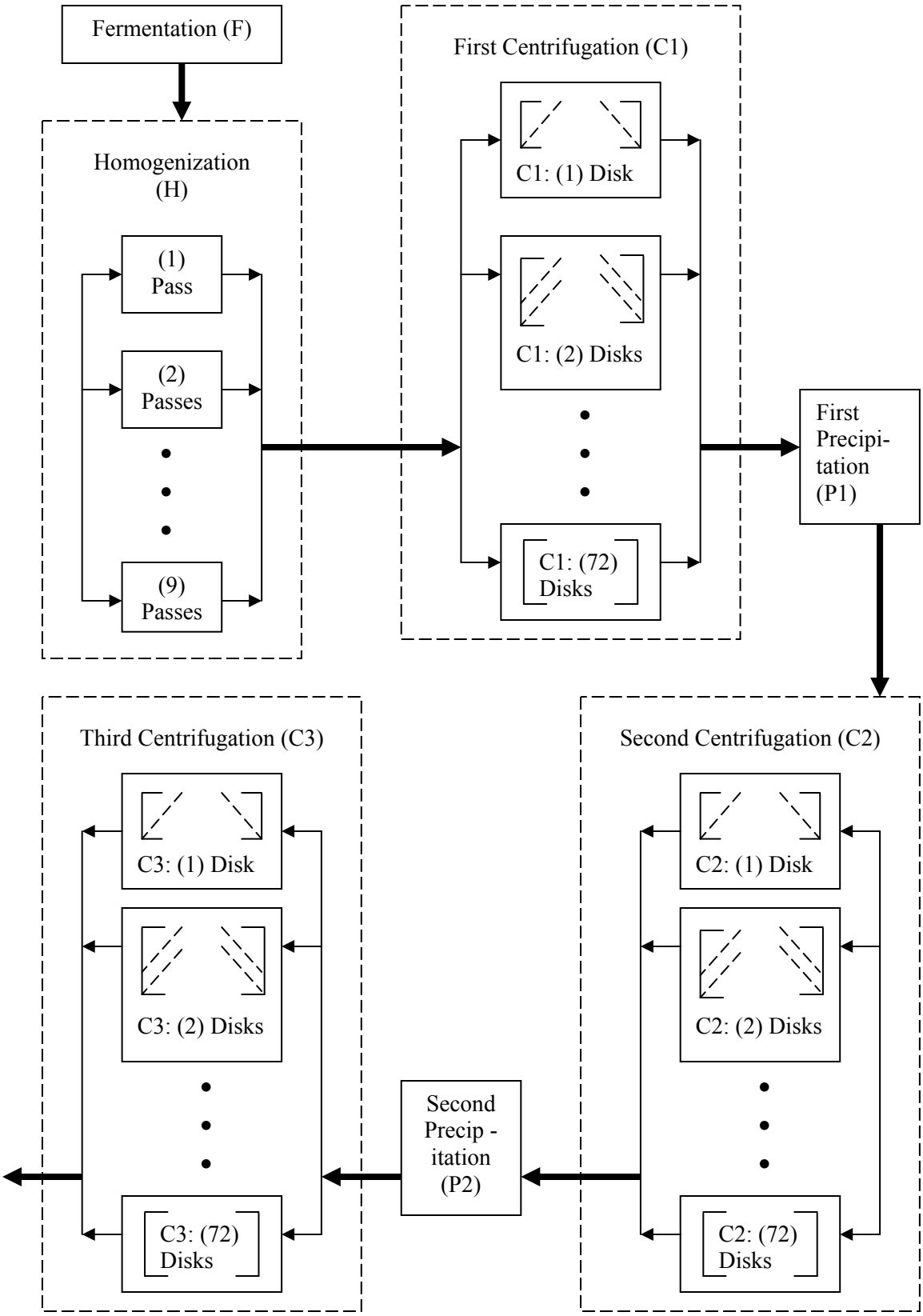


Figure 5.10. Schematic of the process flow diagram for ADH production.

In the schematic, seven unit operations are employed for the process. The variables *F*, *H, Ci,* and *Pi* designate the fermentation, homogenization, i^{th} centrifugation, and i^{th} precipitation process, respectively. The process behavior for each unit operation is considered to be noisy and the equation models are assumed to be inaccessible. A test problem is formulated whereby the objective is to maximize ADH purity and yield while minimizing process operation costs. There are seven continuous variables and four integer variables. The continuous variables are given as follows, where the respective process operation is denoted in parentheses: 1) glucose concentration (F), 2) pressure (H), 3) disk angle (C1, C2, C3), and 4) precipitant concentration (P1, P2). The integer design variables are 1) number of passes (H), and 2) number of disks (C1, C2, C3). The problem formulation is given in (5.15):

$$max \quad F = 2z_1 + 2z_2 - y_1 - y_2 - y_3 - y_4 \tag{5.15a}$$

$$s.t. \quad z_F = \Gamma_1(x_1, N(0,\sigma^2)) \tag{5.15b}$$

$$z_H = \Gamma_2(x_2, y_1, N(0,\sigma^2)) \tag{5.15c}$$

$$z_{C1} = \Gamma_3(x_3, y_2, N(0,\sigma^2)) \tag{5.15d}$$

$$z_{P1} = \Gamma_4(x_4, N(0,\sigma^2)) \tag{5.15e}$$

$$z_{C2} = \Gamma_5(x_5, y_3, N(0,\sigma^2)) \tag{5.15f}$$

$$z_{P2} = \Gamma_6(x_6, N(0,\sigma^2)) \tag{5.15g}$$

$$z_{C3} = \Gamma_7(x_7, y_4, N(0,\sigma^2)) \tag{5.15h}$$

$$1 \le y_1 \le 9 \tag{5.15i}$$

$$1 \le y_2, y_3, y_4 \le 9 \tag{5.15j}$$

$$\sigma = 0.03 \tag{5.15k}$$

The set of z_F, z_{C1}, z_{P1}, z_{C2}, z_{P2}, and z_{C3} variables belong to the class of z_2 variables described in the problem formulation as given by Equation (5.1). Similarly, the y_1, y_2, y_3, and y_4 variables correspond to the "y_2" variable class described in the same problem formulation given by Equation (5.1), even though the notation $y_{2,i}$, $i = 1 \ldots 4$ is not used in the problem given by (5.15), simply for ease of presentation when providing the equations models described below. Since each unit operation is considered to be black-box, this problem was solved using simulation data since field experimental data were unavailable. When possible, the same models used in the original case study have been employed[52]. In the cases where model equations require experimental parameter data, such as determining the amount of ADH present in the cells after fermentation, fitted equations have been employed which match the literature data. The models used to simulate each stage of the ADH process will now be presented in more detail. Since each process operation is treated as black-box and noisy, optimization is performed using surrogate kriging models constructed from sampling data z_F, z_H, z_{Ci}, or z_{Pi} obtained from continuous and integer inputs. The closed-form equations are presented simply to describe process operation and are not directly optimized.

The main goal of fermentation is to grow cells to a high density in order to increase the ADH yield in recognition of the fact that a significant amount may be lost during downstream processing. At the end of the fermentation time period t, cellular growth is arrested and the corresponding diameter indicates whether the cell was in the lag, exponential, or stationary phase. The cell diameter ranges from 2 – 12 micrometers and a simulated size distribution model can be generated using linear combinations of Weibull probability density functions. The total cell number is determined by summing the

number of cells $R(d_i)$ having given diameter d_i, $i = 1 \ldots N_d$, for N_d equispaced intervals. The median cell diameter $d_{c,50}$ and cumulative distribution function scatterpoints $c(d_i)$ are obtained once the total cell number has been obtained. The corresponding mass and mass fraction of cells having diameter d_i entering the homogenizer are given as $m_{c,frac,i}$ and $m_{c,i}$, respectively, as shown in Equation (5.16).

$$m_{c,frac,i} = \frac{R(d_i)}{\sum_{i=1}^{N_d} R(d_i)} \quad , \quad i = 1 \ldots N_d \tag{5.16a}$$

$$m_{c,i} = X \frac{m_{c,frac,i}}{\sum_{j=1}^{N_d} m_{c,frac,j}} \quad , i = 1 \ldots N_d \tag{5.16b}$$

The cumulative size distribution of the cell diameter can be modeled using a Boltzmann equation as given in (5.17a) by fitting a standard deviation parameter w_c to experimental or simulated mass fraction data[53]. It should be noted that $m_{c,frac,i}$ can alternatively be obtained if a value for w_c is prespecified.

$$c(d_i) = \left\{ 1 - \frac{1}{1 + exp\left(\frac{d_i - d_{c,50}}{w_c} \right)} \right\} = \frac{exp\left(\frac{d_i - d_{c,50}}{w_c} \right)}{1 + exp\left(\frac{d_i - d_{c,50}}{w_c} \right)} \quad , \quad i = 1 \ldots N_d \tag{5.17a}$$

$$m_{c,frac,i} = \frac{exp\left(\frac{d_i - d_{c,50}}{w_c} \right)}{(w_c)\left(1 + exp\left(\frac{d_i - d_{c,50}}{w_c} \right) \right)^2} \quad , \quad i = 1 \ldots N_d \tag{5.17b}$$

$$w_c = argmin \sum_{i=1}^{N_d} \left(c(d_i) - m_{c,frac,i} \right) \tag{5.17c}$$

The remaining fermentation equations are presented as follows in Equation (5.18):

$$[X_0, \mu, t, Y_{xs}] = [10, 0.21, 3600, 0.5] \tag{5.18a}$$

$$100 \le s_{in} \le 500 \tag{5.18a}$$

$$X = X_0 \exp(\mu t) \tag{5.18c}$$

$$S_c = \frac{X - X_0}{Y_{xs}} \tag{5.18d}$$

$$V_{out} = V_0 + \frac{S_c}{s_{in}} \tag{5.18e}$$

$$m_{e,ins} = [0.95 + N(0,0.1)][(-695.05e7)\mu^2 + (131.97e7)\mu + 7.9234e7] \tag{5.18f}$$

$$m_{p,ins} = 1.125 m_{e,unr} \tag{5.18g}$$

$$e_{ins,out} = \frac{m_{p,ins}}{V_{out}} \tag{5.18h}$$

$$p_{ins,out} = \frac{m_{e,ins}}{V_{out}} \tag{5.18i}$$

$$\left[V_{out,f}, e_{ins,out,f}, p_{ins,out,f}\right] = \left[V_{out,f}, e_{ins,out,f}, p_{ins,out,f}\right] \tag{5.18j}$$

$$x_1 = s_{in} \tag{5.18k}$$

$$z_F = \left[X, S_c, V_{out,f}, m_{p,unr}, m_{e,unr}, e_{ins,out,f}, p_{ins,out,f}, m_{c,frac}, m_c, c, w_c\right] \tag{5.18l}$$

where the vector of fermenter output variables z_F consists of the amount of biomass X created, amount of glucose substrate consumed S_c, final broth volume $V_{out,f}$, mass of unreleased, and currently insoluble, protein $m_{p,unr}$, and corresponding mass of intracellular enzyme $m_{e,unr}$, respectively. The size-distributed cellular mass, mass fractions, and Boltzmann parameters are included for completion. The glucose substrate concentration S_0 is also referred to as x_1 in Equation (5.15b) in keeping with the

designation of continuous variables as x to be consistent with model (1). Equations (5.24) – (5.26) comprise the fermenter model equations and are symbolized by $\Gamma_1(x, N(0,\sigma^2))$.

At the end of the fermentation, the biomass broth is fed to the homogenizer where ADH release is achieved via pressurized cellular disruption. The fermentation broth can be passed through the homogenizer up to nine times, with each additional pass resulting in a higher number of cell walls bursting apart. If too few passes are used, a high number of cell walls may remain intact, causing the ADH to remain inaccessible resulting in a lowered process yield. However, employing too many passes may lead to micronization of the cellular wall debris, causing separation of ADH from the debris and the intracellular protein contaminant to become more difficult. A simplified process flow schematic is shown in Figure 5.11 which corresponds to the detailed equipment schematic presented in Figure 5.9.

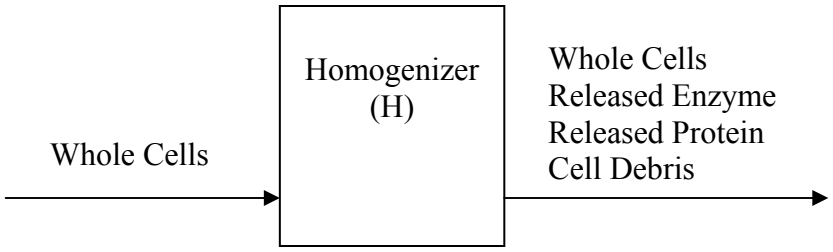


Figure 5.11. Schematic of the homogenizer process.

Since the biomass is comprised of cells having different diameters, a model is proposed by Groep[52] describing the pressure required for cell breakage, since cells having a larger diameter require a higher disruption pressure. The model describing the threshold breakage pressure $P_{c,i}$ is given by Equation (5.19e), where χ is defined as a cell strength parameter. The constants P_{c0} and P_{cN} denote the threshold pressures required to induce

cell breakage for cells having the lowest and highest diameters d_1 and d_{ND}, respectively. The pressure P is a manipulated parameter given as x_2 whose range varies between 689,476 Pa and 3.447e6 Pa. The homogenizer effluent consists of undisrupted cells, released protein and ADH, and cell wall debris. The equations representing the mass of undisrupted cells $m_{c,out}$, released enzyme concentration $e_{sol,out}$, and protein concentration $p_{sol,out}$ released, are also included in the other equations given by (5.19):

$$[\chi, P_{c0}, P_{cN}, k_c] = [1, (275,790), (620,528), 4.619e6] \tag{5.19a}$$

$$[m_{c,in}, V_{in}, e_{ins,in}, p_{ins,in}] = [m_c, V_{out,f}, e_{ins,out,f}, p_{ins,out,f}] \tag{5.19b}$$

$$1 \le N \le 9 \tag{5.19c}$$

$$689,476 \le P \le 3.447e6 \tag{5.19d}$$

$$P_{c,i} = \chi\left(P_{c0} + \frac{(d_i - d_1)(P_{cN} - P_{c0})}{(d_{ND} - d_i)} \right) \quad , \quad i = 1 \ldots N_d \tag{5.19e}$$

$$m_{c,out,i} = m_{c,in,i} exp\left(-k_c N^{\beta_c} (P - P_{c,i})^{\alpha_c} \right) \quad , \quad i = 1 \ldots N_d \tag{5.19f}$$

$$e_{sol,out} = e_{ins,in} \left(\frac{\sum_{i=1}^{N_d} m_{c,in,i} \left(1 - exp(-k_c N^{\beta_c} (P - P_{c,i})^{\alpha_c} \right)}{\sum_{i=1}^{N_d} m_{c,in,i}} \right) \tag{5.19g}$$

$$p_{sol,out} = p_{ins,in} \left(\frac{\sum_{i=1}^{N_d} m_{c,in,i} \left(1 - exp(-k_c N^{\beta_c} (P - P_{c,i})^{\alpha_c} \right)}{\sum_{i=1}^{N_d} m_{c,in,i}} \right) \tag{5.19h}$$

$$e_{ins,out} = \left(\frac{m_{e,unr}}{V_{in}} \right) \tag{5.19i}$$

$$p_{ins,out} = \left(\frac{m_{p,unr}}{V_{in}} \right) \tag{5.19j}$$

The cell size distribution for disrupted cells can also be fitted to a Boltzmann equation in order to model the cumulative size distribution as given by Equation (5.20a). The median cell diameter $d_{q,50}$ is obtained in a similar manner to that of how $d_{c,50}$ was obtained. For the disrupted cells, the size-distributed mass $m_{q,out,i}$ and mass fraction $m_{q,frac,i}$ models are given as shown in Equations (5.20b) and (5.20a), respectively. It is assumed that any undisrupted cells exiting the homogenizer will remain intact after further processing. Based on this assumption, the undisrupted cells are then considered, in addition to the cell debris, to be waste particles. The corresponding size-distributed waste particle mass fraction is obtained by summing the respective distributions of undisrupted cells and cellular debris. The equations given by (5.19) and (5.20) comprise the homogenizer model equations and are symbolized by $\Gamma_2(x_2, y_1, N(0,\sigma^2))$.

$$m_{q,frac,i} = \frac{exp(d_i - d_{q,50})}{\left(w_q\right)\left(1+exp(d_i - d_{q,50})\right)^2} \quad , \quad i = 1...N_d \tag{5.20a}$$

$$m_{q,out,i} = m_{q,frac,i}\left(\frac{X - Vp_{sol,out} - Ve_{sol,out} - \sum_{j=1}^{N_d} m_{c,out,j}}{\sum_{j=1}^{N_d} m_{q,frac,j}}\right) \ , \ i = 1...N_d \tag{5.20b}$$

$$m_{w,out,i} = m_{c,out,i} + m_{q,out,i} \quad , \quad i = 1...N_d \tag{5.20c}$$

$$\left[m_{c,out,h}, e_{sol,out,h}, p_{sol,out,h}, e_{ins,out,h}\right] = \left[m_{c,out}, e_{sol,out}, p_{sol,out}, e_{ins,out}\right] \tag{5.20d}$$

$$\left[p_{ins,out,h}, m_{q,frac,h}, m_{q,out,h}, m_{w,out,h}\right] = \left[p_{ins,out}, m_{q,frac}, m_{q,out}, m_{w,out}\right] \tag{5.20e}$$

$$[x_2, y_1] = [P, N] \tag{5.20f}$$

$$z_H = \left[m_{c,out,h}, e_{sol,out,h}, p_{sol,out,h}, m_{q,frac,h}, m_{q,out,h}, m_{w,out,h}\right] \tag{5.20g}$$

Once cell disruption has occurred, the next task is ADH purification from the cell debris, remaining undisrupted biomass, and protein contaminant. In the first step, the heavier cell debris and remaining whole cells are separated from the ADH and protein using centrifugation[54]. The ADH and protein leave in a supernatant stream while the cell debris and whole cells exit in a sediment stream. A process schematic is shown in Figure 5.12.

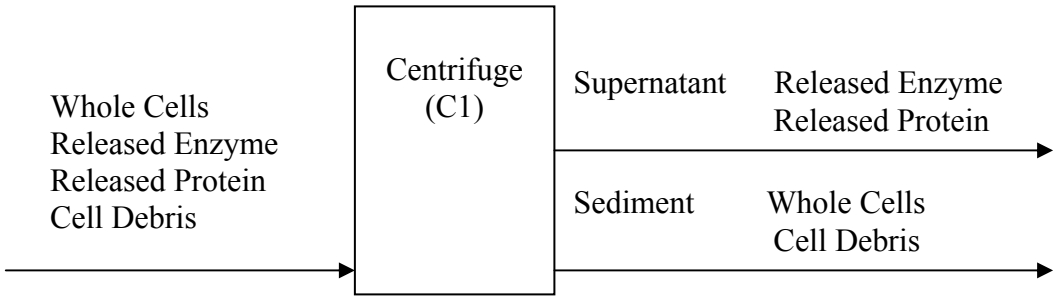


Figure 5.12. Schematic of the first centrifuge process (C1).

The centrifuge model used is a disk-type centrifuge, shown in Figure 5.9. The incoming feed enters the top of the centrifuge and flows downward through a conical disk stack having narrow flow channels between each disk. The centrifuge is horizontally agitated such that the feed stream splits off into smaller streams entering each of the flow channels. The heavier particles settle to the bottom of each channel, slide down to the bottom of the disk stack via mechanical agitation and gravity, and then exit the centrifuge at the base of the body. The lighter particles remaining in suspension now comprise a clarified liquid phase and are pumped out of the centrifuge at the top of the unit. Due to the nonuniform cell size distribution, not all waste particles exit in the sediment phase. The grade efficiency $T(d_i)$, given by (5.21c), is a function fitted to experimental data

describing the percentage of solids recovered in the sediment phase stream, based on particles having diameter d_i. If a cell diameter d_i falls below a critical diameter d_c, the cell is assumed to be light enough such that it exits with the supernatant. The parameters k and n are regression constants having randomly initialized values for simulation purposes. The remaining parameters consist of a hindered settling factor f_s, gravitational constant g, volumetric throughput Q, carrier fluid viscosity η, density difference between liquid and solid phases $\Delta\rho$, outer and inner disk diameters R_o and R_i, and angular bowl velocity ω.

$$\left[k, n, f_S, \eta\right] = \left[0.9501, 0.4621, 1.6, 1.005e^{-3}\right] \tag{5.21a}$$

$$\left[\Delta\rho, R_o, R_i, \omega\right] = \left[191.8, 0.076, 0.036, 960\right] \tag{5.21b}$$

$$T(d_i) = 1 - exp\left(-\left(k\frac{d_i}{d_c}\right)^n\right) \quad , \quad i = 1 \ldots N_d \tag{5.21c}$$

$$d_c = f_s\sqrt{\frac{18Q\eta}{\Delta\rho\Sigma g}} \tag{5.21d}$$

$$\Sigma = \frac{2\pi Z\omega^2\left(R_0^3 - R_i^3\right)}{3g\tan\theta} \tag{5.21e}$$

$$30^\circ \le \theta \le 75^\circ \tag{5.21f}$$

$$1 \le Z \le 72 \tag{5.21g}$$

The variables represented in the supernatant equations are given as follows: 1) size-distributed mass of waste solids $m_{w,out,i}$, 2) waste solids concentration w_{out}, 3) dissolved enzyme concentration $e_{sol,out}$, 4) dissolved protein concentration $p_{sol,out}$, 5) intracellular enzyme concentration $e_{ins,out}$, and 6) intracellular protein concentration $p_{ins,out}$. The constants λ_1 and λ_2, respectively, designate the fraction of dissolved enzyme or protein

that becomes denatured or exits in the sediment phase stream. The equations given by (5.20) and (5.21) comprise the centrifuge model equations and are symbolized by $\Gamma_3(x_3, y_2, N(0, \sigma^2))$.

$$\left[V_{in}, e_{sol,in}, p_{sol,in}, m_{w,in}\right] = \left[V, e_{sol,out,h}, p_{sol,out,h}, m_{w,out,h}\right] \tag{5.22a}$$

$$\left[m_{c,in}, e_{ins,in}, p_{ins,in}\right] = \left[m_{c,out,h}, e_{ins,in,h}, p_{ins,in,h}\right] \tag{5.22b}$$

$$\lambda_1 = 0.05(1 + N(0, \sigma^2)) \tag{5.22c}$$

$$\lambda_2 = 0.05(1 + N(0, \sigma^2)) \tag{5.22d}$$

$$f_{sup} = 0.95(1 + N(0, \sigma^2)) \tag{5.22e}$$

$$V_{sup} = f_{sup} V_{in} \tag{5.22f}$$

$$e_{sol,out} = \frac{e_{sol,in} V_{in}}{V_{sup}} (1 - \lambda_1)(1 - \lambda_2) \tag{5.22g}$$

$$p_{sol,out} = \frac{p_{sol,in} V_{in}}{V_{sup}} (1 - \lambda_1)(1 - \lambda_2) \tag{5.22h}$$

$$m_{w,out,i} = (1 - T(d_i)) m_{w,in,i} \quad , \quad i = 1 \ldots N_d \tag{5.22i}$$

$$m_{c,out,i} = \left(\frac{m_{w,out,i}}{m_{w,in,i}} \right) m_{c,in,i} \quad , \quad i = 1 \ldots N_d \tag{5.22j}$$

$$e_{ins,out} = e_{ins,in} \sum_{i=1}^{N_d} m_{c,out,i} \tag{5.22k}$$

$$p_{ins,out} = p_{ins,in} \sum_{i=1}^{N_d} m_{c,out,i} \tag{5.22l}$$

$$w_{out} = \frac{\sum_{i=1}^{N_d} m_{w,out,i}}{V_{sup}} \tag{5.22m}$$

$$\left[m_{w,out,C1}, m_{c,out,C1}, \lambda_{1,C1}, \lambda_{2,C1}\right] = \left[m_{w,out}, m_{c,out}, \lambda_1, \lambda_2\right] \tag{5.22n}$$

$$\left[f_{sup,C1}, V_{sup,C1}, e_{sol,out,C1}, p_{sol,out,C1}\right] = \left[f_{sup}, V_{sup}, e_{sol,out}, p_{sol,out}\right] \quad (5.22o)$$

$$\left[e_{ins,out,C1}, p_{ins,out,C1}, w_{out,C1}\right] = \left[e_{ins,out}, p_{ins,out}, w_{out}\right] \quad (5.22p)$$

$$\left[x_3, y_2\right] = \left[\theta, Z\right] \quad (5.22q)$$

$$z_{C1} = \left[m_{w,out,C1}, m_{c,out,C1}, \lambda_{1,C1}, \lambda_{2,C1}, f_{sup,C1}, V_{sup,C1}, e_{sol,out,C1}, p_{sol,out,C1}, e_{ins,out,C1}, p_{ins,out,C1}, w_{out,C1}\right] \quad (5.22r)$$

The centrifuge supernatant is fed to a tank whereby an ammonium sulfate precipitant is then added in order to precipitate out the protein contaminant. A process schematic is presented in Figure 5.13.

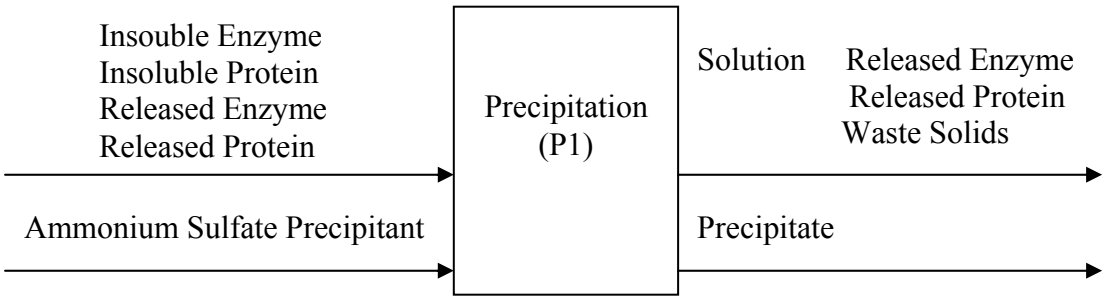


Figure 5.13. Schematic of the first precipitation process (P1).

The amount of precipitant V_z needed is determined by the desired output precipitant concentration z_{out} as given by Equation (5.23c). The models for V_z and V_{out} are not treated as black-box since these output variables become explicitly known once z_{out} has been specified. These models would be represented in the formulation given by Equation (5.1) by the equality constraints $h(x,z_1)$, where x and z_1 correspond to the vectors $[V_{in}, z_{in}]$ and $[V_z, V_{out}]$, respectively:

$$\left[V_{in}, z_{in}\right] = \left[V_{sup,C1}, 0\right] \quad (5.23a)$$

$$10 \le z_{out} \le 40 \tag{5.23b}$$

$$V_z = \frac{V_{in}(z_{out} - z_{in})}{100 - z_{out}} \tag{5.23c}$$

$$V_{out} = V_{in} + V_z \tag{5.23d}$$

The parameters ϕ_e and ϕ_p describe the soluble enzyme and protein fractions after precipitation. The equations shown in (5.24) comprise the remaining precipitation model equations and are symbolized by $\Gamma_4(x_4, N(0,\sigma^2))$.

$$\left[e_{sol,in}, e_{ins,in}, p_{sol,in}\right] = \left[e_{sol,out,C1}, e_{ins,out,C1}, p_{sol,out,C1}\right] \tag{5.24a}$$

$$\left[p_{ins,in}, w_{in}\right] = \left[p_{ins,out,C1}, w_{out,C1}\right] \tag{5.24b}$$

$$\phi_e = 0.9(1 + N(0,\sigma^2)) \tag{5.24c}$$

$$\phi_p = 0.9(1 + N(0,\sigma^2)) \tag{5.24d}$$

$$e_{sol,out} = \frac{\phi_e e_{sol,in} V_{in}}{V_{out}} \tag{5.24e}$$

$$e_{ins,out} = \frac{(e_{ins,in} + (1-\phi_e)e_{sol,in})V_{in}}{V_{out}} \tag{5.24f}$$

$$p_{sol,out} = \frac{\phi_p p_{sol,in} V_{in}}{V_{out}} \tag{5.24g}$$

$$p_{ins,out} = \frac{(p_{ins,in} + (1-\phi_p)p_{in})V_{in}}{V_{out}} \tag{5.24h}$$

$$w_{out} = \frac{w_{in} V_{in}}{V_{out}} \tag{5.24i}$$

$$\left[z_{out,P1}, V_{z,P1}, V_{out,P1}, \phi_{e,P1}, \phi_{p,P1}, e_{sol,out,P1}\right] = \left[z_{out}, V_z, V_{out}, \phi_e, \phi_p, e_{sol,out}\right] \tag{5.24j}$$

$$\left[e_{ins,out,P1}, p_{sol,out,P1}, p_{ins,out,P1}, w_{out,P1}\right] = \left[e_{ins,out}, p_{sol,out}, p_{ins,out}, w_{out}\right] \tag{5.24k}$$

$$x_4 = z_{out} \tag{5.24l}$$

$$z_{P1} = \left[\phi_{e,P1}, \phi_{p,P1}, e_{sol,out,P1}, e_{ins,out,P1}, p_{sol,out,P1}, p_{ins,out,P1}, w_{out,P1} \right] \tag{5.24m}$$

The remaining solution from the precipitation is passed through another centrifuge (C2) in order to separate the enzyme from the remaining waste solids in the form of cell debris and undisrupted cells. A process schematic is shown in Figure 5.14.

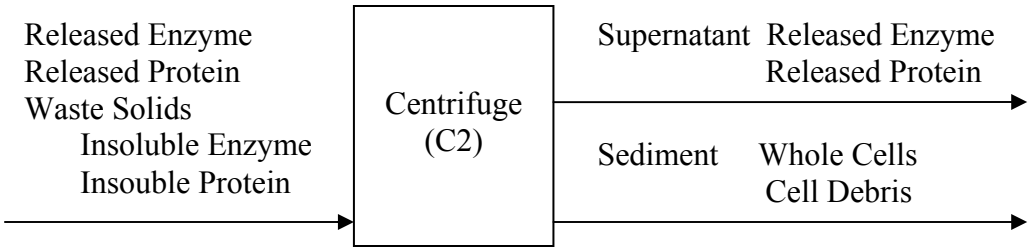


Figure 5.14. Schematic of the second centrifuge process (C2).

The process models used for C2 are the same as those employed for C1 as given by Equations (5.21) and (5.22), where θ and *Z* are now given as x_5 and y_3, respectively. The corresponding parameter values for the input variables in Equation (5.22a) are provided on the RHS of Equation (5.25):

$$\left[V_{in}, e_{sol,in}, p_{sol,in}, m_{w,in} \right] = \left[V_{out,P1}, e_{sol,out,P1}, p_{sol,out,P1}, m_{w,out,C1} \right] \tag{5.25a}$$

$$\left[m_{c,in}, e_{ins,in}, p_{ins,in} \right] = \left[m_{c,out,C1}, e_{ins,out,P1}, p_{ins,out,P1} \right] \tag{5.25b}$$

The corresponding output variables are designated as follows in Equation (5.26). The equations given by Equations (5.21), (5.22), (5.25), and (5.26) comprise the model equations for the second centrifugation and are symbolized by $\Gamma_5(x_5, y_3, N(0, \sigma^2))$.

$$\left[m_{w,out,C2}, m_{c,out,C2}, \lambda_{1,C2}, \lambda_{2,C2} \right] = \left[m_{w,out}, m_{c,out}, \lambda_1, \lambda_2 \right] \tag{5.26a}$$

$$\left[f_{sup,C2}, V_{sup,C2}, e_{sol,out,C2}, p_{sol,out,C2} \right] = \left[f_{sup}, V_{sup}, e_{sol,out}, p_{sol,out} \right] \tag{5.26b}$$

$$\left[e_{ins,out,C2}, p_{ins,out,C2}, w_{out,C2}\right] = \left[e_{ins,out}, p_{ins,out}, w_{out}\right] \tag{5.26c}$$

$$\left[x_5, y_3\right] = \left[\theta, Z\right] \tag{5.26d}$$

$$z_{C2} = \left[m_{w,out,C2}, m_{c,out,C2}, \lambda_{1,C2}, \lambda_{2,C2}, f_{sup,C2}, V_{sup,C2}, e_{sol,out,C2}, p_{sol,out,C2}, e_{ins,out,C2}, p_{ins,out,C2}, w_{out,C2}\right] \tag{5.26e}$$

The residual protein in the supernatant is now precipitated a second time according to the same process schematic as given in Figure 5.13. The amount of precipitant needed for the second precipitation is given by Equation (5.27c) and the remaining precipitant equations given in (5.27) comprise a second set of explicitly known equality constraints *h*.

$$\left[V_{in}, z_{in}\right] = \left[V_{sup,C2}, z_{out,P1}\right] \tag{5.27a}$$

$$max(z_{in} + 10, 40) \le z_{out} \le 70 \tag{5.27b}$$

$$V_z = \frac{V_{in}(z_{out} - z_{in})}{100 - z_{out}} \tag{5.27c}$$

$$V_{out} = V_{in} + V_z \tag{5.27d}$$

The input data required for Equations (5.24a) and (5.24b) are now given by the RHS of the first equation in (5.28a) and (5.28b). The corresponding output variables given in Equations (5.25i) and (5.25j), in turn, are redefined by the LHS of the variable designations given in Equations (5.28c) and (5.28d). Together with Equation (5.24), the equations given by (5.27) and (5.28) comprise the model equations for the second precipitation and are symbolized by $\Gamma_6(x_6, N(0,\sigma^2))$.

$$\left[e_{sol,in}, e_{ins,in}, p_{sol,in}\right] = \left[e_{sol,out,C2}, e_{ins,out,C2}, p_{sol,out,C2}\right] \tag{5.28a}$$

$$\left[p_{ins,in}, w_{in}\right] = \left[p_{ins,out,C2}, w_{out,C2}\right] \tag{5.28b}$$

$$\left[z_{out,P2}, V_{z,P2}, V_{out,P2}, \phi_{e,P2}\right] = \left[z_{out}, V_z, V_{out}, \phi_e\right] \tag{5.28c}$$

$$\left[\phi_{p,P2}, e_{sol,out,P2}\right] = \left[\phi_e, \phi_p, e_{sol,out}\right] \tag{5.28d}$$

$$\left[e_{ins,out,P2}, p_{sol,out,P2}, p_{ins,out,P2}, w_{out,P2}\right] = \left[e_{ins,out}, p_{sol,out}, p_{ins,out}, w_{out}\right] \tag{5.28e}$$

$$x_6 = z_{out} \tag{5.28f}$$

$$z_{P2} = \left[\phi_{e,P2}, \phi_{p,P2}, e_{sol,out,P2}, e_{ins,out,P2}, p_{sol,out,P2}, p_{ins,out,P2}, w_{out,P2}\right] \tag{5.28g}$$

The solution from the second precipitation is fed to a centrifuge one last time in order to improve ADH purity. The process schematic differs from the process schematic given in Figure 5.14 in that this time, the purified enzyme is component-rich in the sediment phase stream. The final ADH purity is given by z_1 and the corresponding recovery percentage, compared to the total amount available from fermentation, is denoted by z_2. Together with Equation (5.21), the equations given in (5.29) comprise the third centrifuge model equations, symbolized by $\Gamma_7(x_7, y_4, N(0, \sigma^2))$.

$$\left[V_{in}, e_{sol,in}, p_{sol,in}, m_{w,in}\right] = \left[V_{out,P2}, e_{sol,out,P2}, p_{sol,out,P2}, m_{w,out,C2}\right] \tag{5.29a}$$

$$\left[m_{c,in}, e_{ins,in}, p_{ins,in}\right] = \left[m_{c,out,C2}, e_{ins,out,P2}, p_{ins,out,P2}\right] \tag{5.29b}$$

$$\lambda_1 = 0.05(1 + N(0, \sigma^2)) \tag{5.29c}$$

$$f_{sed} = 0.8(1 + N(0, \sigma^2)) \tag{5.29d}$$

$$V_{sed} = f_{sed} V_{out} \tag{5.29e}$$

$$e_{ins,out} = \frac{e_{ins,in} V_{in}}{V_{out}} (r_e)(1 - \lambda_2) \tag{5.29f}$$

$$m_{w,out,i} = T(d_i)) m_{w,in,i} \quad , \quad i = 1 \ldots N_d \tag{5.29g}$$

$$p_{ins,out} = \frac{p_{ins,in} V_{in}}{V_{out}} (r_p)(1-\lambda_2) \tag{5.29h}$$

$$e_{sol,out} = \frac{e_{sol,in} V_{in}}{V_{out}} \lambda_1 \tag{5.29i}$$

$$p_{sol,out} = \frac{p_{sol,in} V_{in}}{V_{out}} \lambda_1 \tag{5.29j}$$

$$z_2 = \left(\frac{e_{sol,out} V_{sed}}{\sum_{i=1}^{N_d} m_{w,out,i} + p_{sol,out} V_{sed} + e_{sol,out} V_{sed}} \right) \times 100 \tag{5.29k}$$

$$\left[\lambda_{1,C3}, \lambda_{2,C3}, f_{sed,C3}, V_{sed,C3}, m_{w,out,C3} \right] = \left[\lambda_1, \lambda_2, f_{sed}, V_{out}, m_{w,out} \right] \tag{5.29l}$$

$$\left[e_{sol,out,C3}, p_{sol,out,C3} \right] = \left[e_{sol,out}, p_{sol,out} \right] \tag{5.29m}$$

$$\left[e_{ins,out,C3}, p_{ins,out,C3} \right] = \left[e_{ins,out}, p_{ins,out} \right] \tag{5.29n}$$

$$[x_7, y_4] = [\theta, Z] \tag{5.29o}$$

$$z_{C3} = \left[\lambda_{1,C3}, \lambda_{2,C3}, f_{sed,C3}, V_{sed,C3}, e_{sol,out,C3}, p_{sol,out,C3}, e_{ins,out,C3}, p_{ins,out,C3} \right] \tag{5.29p}$$

In Table 5.5, the optimal value of the objective function F, as given by Equation (5.15a), is reported based on the application of RSM and local direct search to the best kriging solution S^K obtained from a partially refined model. A partially refined model is one for which further updating is terminated after an arbitrarily chosen limit of thirty objective function evaluations.

Table 5.5. Performance of the K-R-L algorithm for Problem (5.15), based on local optimization of a partially refined kriging global model.

Algorithm	F_{MINLP}^{opt}	% improvement relative to nominal F_{MINLP}^{opt}	Simulations Required		CPU Time (s)
			Algorithm	Total	
None (nominal sampling set only)	198	---------	---------	15	0.83
Kriging	217	11.9	10	25	3.90
RSM	230	18.7	44	69	5.84
Local Direct Search	244	26.1	27	98	4.03

The nominal kriging predictor is constructed from fifteen sampling vectors, each consisting of input points $\{x_1 \ldots x_7, y_1 \ldots y_4\}$. The value of the nominal best solution F_{MINLP}^{opt} is 198 and is the optimal objective obtained based on the sampled data. After model refinement has occurred, an 11.9% improvement is observed in the objective. Once both RSM and the DS-L algorithms have also been applied, the total improvement rises to 26.1%. The corresponding objective has a value of 244, which is attained after eighty-three additional function evaluations. A modest 7% improvement is observed after RSM optimization at a cost of forty-four function evaluations. This suggests that the "warm-start" iterate attained from the unrefined global model is still relatively far away from the refined local solution, and that further global refinement could result in a lower resource cost during the subsequent local optimization. A set of complementary results is presented in Table 5.6, in which global direct search is applied instead of its local counterpart.

Table 5.6. Performance of the K-R-G algorithm for Problem (5.15), based on global optimization of a partially refined kriging global model.

Algorithm	F_{MINLP}^{opt}	% improvement relative to nominal F_{MINLP}^{opt}	Simulations Required		CPU Time (s)
			Algorithm	Total	
None (nominal sampling set only)	198	---------	---------	15	1.04
Kriging	215	9.4	9	24	3.61
RSM	227	15.9	44	68	6.12
Global Direct Search	286	48.8	41	109	6.33

The optimal solution F_{MINLP}^{opt} has an objective value of 286 and an additional 22.7% improvement is attained relative to the best solution using the K-R-L method, at the cost of an additional eleven function calls. Although this solution is superior to the corresponding objective found using local search, it cannot be confirmed as a global solution due to problem nonconvexity based on the presence of the nonlinear and bilinear x-y_1 variable terms. In Tables 5.6 and 5.7, a complementary set of results are presented in which local optimization is performed on the best kriging solution S^K obtained from a fully refined kriging model.

Table 5.7. Performance of the K-R-L algorithm for Problem (5.15), based on local optimization of a completely refined kriging global model.

Algorithm	F_{MINLP}^{opt}	% improvement relative to nominal F_{MINLP}^{opt}	Simulations Required		CPU Time (s)
			Algorithm	Total	
None (nominal sampling set only)	198	---------	---------	15	0.98
Kriging	240	23.1	49	64	12.62
RSM	252	28.9	42	106	5.62
Local Direct Search	269	37.5	27	133	3.96

Table 5.8. Performance of the K-R-G algorithm for Problem (5.15), based on global optimization of a completely refined kriging global model.

Algorithm	F_{MINLP}^{opt}	% improvement relative to nominal F_{MINLP}^{opt}	Simulations Required		CPU Time (s)
			Algorithm	Total	
None (nominal sampling set only)	198	---------	---------	15	1.07
Kriging	241	28.7	51	66	13.23
RSM	253	34.8	43	109	5.76
Global Direct Search	286	54.1	38	147	5.72

These results are obtained without applying any *a priori* resource restrictions on the amount of sampling directed at global model refinement. Instead, the stopping criterion applied is based on convergence in the average value of the kriging predictor as described in Figure 4. Although the value of F_{MINLP}^{opt} rises to 269 when local search is employed, no increase is observed when global search has been applied. These findings suggest that a better objective can be attained using local search based on a fully developed kriging model. However, a better solution can be discovered using global search, however, even without a completely refined global mapping being generated, suggesting that the problem nonconvexity involving the y_1-variable terms can cause the local direct search algorithm to become trapped in a suboptimal solution. The discovery of a superior objective using the global direct search method is an example of how global direct search is effective, at least for this example, in overcoming problem nonconvexity.

5.5 Summary

In this chapter, a unified algorithm has been presented which integrates Direct Search with B&B, Kriging, and RSM in order to address process synthesis and design problems containing black-box functions which can depend on both continuous and integer variables. B&B is used to optimize integer variables having a continuous relaxation while RSM is used to optimize the continuous variables. The integer variables appearing in the black-box functions are optimized using either local or global direct search, and it is found that global search leads to better solutions being obtained based on algorithm performance for two presented case studies. The unified B&B Kriging-RSM-Direct Search algorithm proceeds as follows whereby at each node of a B&B tree, kriging is used to build global models of partially relaxed NLP subproblem objectives. The surrogate models are used to identify subregions containing potential local optima and the best kriging solutions serve as starting iterates for further optimization using RSM and direct search. The additional costs resulting from global model creation are offset by successful convergence to improved local solutions.

Notation

General

x = vector of continuous variables

y = vector of integer-valued variables

y_1 = vector of integer design variables

y_2 = vector of synthesis variables

z_1 = vector of output variables whose input-output models are known

z_2 = vector of output variables whose input-output models are black-box

$\Re^n$ = subspace of continuous variables

q_1 = subspace of integer design variables

q_2 = subspace of synthesis variables

g = feasibility constraint

h = feasibility constraint or explicitly known closed-form process models

Γ = input-output models lacking closed-form equations

F = objective function

μ = mean value for a noisy, black-box model output variable

σ^2 = estimated or simulated standard deviation when process noise is modeled as a normally distributed random function

x^0 = set of continuous variables in any nominal sampling vector

y_1^0 = set of integer design variables in any nominal sampling vector

y_2^0 = set of synthesis variables in any nominal sampling vector

Ω = nominal sampling set

S^K = kriging-optimal solution

x^K = set of kriging-optimal continuous variables

y_1^K = set of kriging-optimal integer design variables

y_2^K = set of kriging-optimal synthesis variables

x^R = set of RSM-optimal continuous variables

y_2^R = set of RSM-optimal synthesis variables

y_1^D = optimal sampling vector with respect to y_1

S^R = optimal sampling vector with respect to x and y_2

S^D = optimal sampling vector with respect to x, y_1, and y_2

RSM algorithm

RSM = response surface methodology

n = RSM problem dimension

w = RSM iteration index

$S_{coll,w}$ = set of sampling points used to build the w^{th} response surface

S^0 = nominal locally optimal solution vector to be locally optimized with respect to x and relaxed y_2

F^0 = objective value corresponding to S^0

F^K = kriging-optimal objective function value

$S_{opt,w}$ = current best solution vector

$F_{opt,w}$ = current best objective value

$S_{opt,w+1}$ = sampling vector obtained from optimizing the w^{th} response surface

$F_{opt,w+1}$ = objective function value corresponding to w^{th} RSM-optimal sampling vector

b_w = w^{th} response surface radius

Tol_{RSM} = stopping tolerance for RSM algorithm

Direct Search algorithm

DS = direct search

DS-L = local direct search algorithm

DS-G = global direct search algorithm

m = iteration index

Y^{LL} = lowest feasible value for a strict integer variable

Y^{UL} = highest feasible value for a strict integer variable

Y_m = m^{th} best solution for a strict integer variable

Y_m^L = sampling point corresponding to the m^{th} bracket low endpoint

Y_m^C = sampling point corresponding to the m^{th} bracket midpoint

Y_m^U = sampling point corresponding to the m^{th} bracket high endpoint

$F(\cdot)$ = objective function value corresponding to sampling point $(\cdot)$

N_m = m^{th} bracket midpoint-endpoint interval length

Y^D = integer-optimal solution for a strict integer variable

Branch-and-Bound algorithm/Unified algorithm

B&B = Branch-and-Bound algorithm

LB = lower bound

UB = upper bound

S_{NLP}^{opt} = best locally optimized solution vector corresponding to a partially relaxed NLP

F_{NLP}^{opt} = objective value corresponding to best local solution found for a partially relaxed NLP

S_{MINLP}^{opt} = MINLP solution vector that is integer optimal in y_1 and y_2

F_{MINLP}^{opt} = MINLP objective function value corresponding to integer-optimal y_1 and y_2

Tol_{BB} = stopping tolerance for B&B algorithm

Example 1

t-BMA = tert-butyl methacrylate

MA = methacrylic acid

IB = isobutylene

DIB = di-isobutylene

H_2SO_4 = sulfuric acid catalyst

NaOH = catalyst neutralizer

RM = raw material

F_0 = reactor fresh feed stream [kg/h]

F_{IB} = fresh isobutylene feed stream [kg/h]

F_{MA} = fresh methacrylic acid feed stream [kg/h]

F_{H2SO4} = fresh sulfuric acid catalyst feed stream [kg/h]

F_{NaOH} = fresh catalyst neutralizer feed stream [kg/h]

$F_{t\text{-}BMA}$ = *t*-BMA feed stream containing purchased *t*-BMA[kg/h]

V_{CW} = volume of cooling water needed for *t*-BMA process [m^3/h]

V_{ChW} = volume of chilled water needed for *t*-BMA process [m^3/h]

V_{Ref} = volume of refrigerant needed for *t*-BMA process [m^3/h]

M_{Stm} = mass of steam needed for *t*-BMA process [kg/h]

F_1 = reactor exit stream [kg/h]

F_2 = feed stream for separation train 1 [kg/h]

F_3 = feed stream for separation train 2 [kg/h]

F_4 = feed stream for separation trains 3 and 4 [kg/h]

F_5 = feed stream for separation trains 5 and 6 [kg/h]

F_6 = feed stream for separation train 3 [kg/h]

F_7 = feed stream for separation train 4 [kg/h]

F_8 = feed stream for separation train 5 [kg/h]

F_9 = feed stream for separation train 6 [kg/h]

y_i = synthesis variable indicating existence of feed stream F_i [-]

Q_{CW} = cooling water duty [J/h]

ρ_{CW} = cooling water density [kg/m^3]

$C_{p,CW}$ = cooling water heat capacity [kJ/kg K]

ΔT_{CW} = minimum approach temperature for cooling water [K]

Q_{ChW} = chilled water duty [J/h]

ρ_{ChW} = chilled water density [kg/m^3]

$C_{p,ChW}$ = chilled water heat capacity [kJ/kg K]

ΔT_{CW} = minimum approach temperature for chilled water [K]

Q_{Ref} = refrigerant duty [J/h]

ρ_{Ref} = chilled water density [kg/m^3]

C_{Ref} = chilled water heat capacity [kJ/kg K]

ΔT_{Ref} = minimum approach temperature for refrigerant [K]

D_{jk} = j^{th} component-rich distillate obtained for the k^{th} separation [kg/h]

B_{jk} = j^{th} component-rich bottoms obtained for the k^{th} separation [kg/h]

D_{jk} = j^{th} component-rich distillate obtained for the k^{th} separation [kg/h]

B_{jk} = j^{th} component-rich bottoms obtained for the k^{th} separation [kg/h]

D_{jk}^{noisy} = j^{th} component-rich distillate obtained for the k^{th} separation when noise exists [kg/h]

B_{jk}^{noisy} = j^{th} component-rich bottoms obtained for the k^{th} separation when noise exists [kg/h]

ω = scaling parameter limiting the amount of RM fed to reactor [-]

RR_k = reflux ratio for the k^{th} separation [-]

$T_{reb,k}$ = reboiler temperature for k^{th} separation [K]

Example 2

General

i = dummy index for a repeated process

ADH = alcohol dehydrogenase

F = fermentation process

H = homogenization process

$Ci = i^{th}$ centrifugation process

$Pi = i^{th}$ precipitation process

z_1 = ADH yield [kg/h]

z_2 = ADH purity [-]

y_1 = number of homogenizer passes [-]

y_2 = number of centrifuge disks for first centrifugation [-]

y_3 = number of centrifuge disks for second centrifugation [-]

y_4 = number of centrifuge disks for third centrifugation [-]

x_1 = glucose concentration [kg/m^3]

x_2 = homogenizer pressure [N/m^2]

x_3 = centrifuge disk stack angle for first centrifugation [rad]

x_4 = precipitant concentration for first precipitation [kg/m^3]

x_5 = centrifuge disk stack angle for second centrifugation [rad]

x_6 = precipitant concentration for second precipitation [kg/m^3]

x_7 = centrifuge disk stack angle for third centrifugation [rad]

z_F = any fermentation output variable for the fermentation process [-]

z_H = any homogenization output variable for the homogenization [-]

z_{C1} = any output variable for the first centrifugation process [-]

z_{P1} = any output variable for the first precipitation process [-]

z_{C2} = any output variable for the second centrifugation process [-]

z_{P2} = any output variable for the second precipitation process [-]

z_{C3} = any output variable for the third centrifugation process [-]

α = standard deviation parameter designating process noise intensity [-]

Fermentation

i = dummy index designating a given cell diameter

$d_i = i^{th}$ cellular diameter [m]

$R(d_i)$ = final number of cells having diameter d_i [-]

$d_{c,50}$ = median cell diameter [m]

N_d = number of size-differentiated cell diameters [-]

$m_{c,frac,i}$ = mass fraction of cells having diameter equal to d_i after fermentation [-]

$m_{c,in,i}$ = final mass of cells having diameter d_i [kg]

$m_{c,frac}$ = size-distributed cell mass fraction after fermentation [-]

m_c = size-distributed cell mass after fermentation [kg]

$c(d_i)$ = fraction of cells having diameter equal to or less than d_i [-]

w_c = Boltzmann parameter for modeling cdf of undisrupted cells [-]

X_0 = initial biomass concentration [kg/m^3]

μ = cellular growth rate [s^{-1}]

t = fermentation time period [s]

X = final biomass concentration [kg/m^3]

S_c = amount of substrate consumed [kg]

Y_{xs} = yield coefficient of enzyme on substrate [U/kg]

s_{in} = initial glucose concentration [kg/m^3]

V = final broth volume [m^3]

V_0 = initial broth volume [m^3]

$m_{e,unr}$ = mass of ADH inside undisrupted cells [kg]

$m_{p,unr}$ = mass of protein inside undisrupted cells [kg]

Homogenization

χ = cell strength parameter [-]

P_{c0} = threshold breakage pressure for strongest cells having lowest diameter [Pa]

P_{cN} = threshold breakage pressure in strongest cells having highest diameter [Pa]

N = number of homogenizer passes [-]

k_c = cell disruption rate constant, [$N^{0.4}$ Pa]

$m_{c,in}$ = size-distributed cell mass in feed broth [kg]

V_{in} = feed broth volume [m^3]

P = homogenizer pressure [Pa]

$P_{c,i}$ = threshold pressure required for disruption of cells having diameter d_i [Pa]

d_1 = lowest cell diameter [m]

d_{ND} = highest cell diameter [m]

$m_{c,out,i}$ = mass of undisrupted cells having diameter d_i leaving homogenizer [kg]

α_c = cell disruption constant [-]

β_c = cell disruption constant [-]

$e_{sol,out}$ = concentration of dissolved ADH in homogenized broth [U/m^3]

$p_{sol,out}$ = concentration of dissolved protein in homogenized broth [kg/m^3]

$e_{ins,out}$ = concentration of intracellular ADH in homogenized broth [U/m^3]

$p_{ins,out}$ = concentration of intracellular protein in homogenized broth [kg/m^3]

$m_{q,frac,i}$ = mass fraction of disrupted cells having diameter d_i [kg]

$d_{q,50}$ = median diameter of disrupted cells [m]

w_q = Boltzmann parameter for modeling cdf of disrupted cells [-]

$m_{q,out,i}$ = mass of cellular debris having diameter d_i [kg]

$m_{w,out,i}$ = mass of waste solids having diameter d_i [kg]

Centrifugation

$T(d_i)$ = grade efficiency measuring recovery of cells having diameter d_i [-]

d_c = critical diameter below which any smaller cells leave in the supernatant [m]

k = grade efficiency parameter [-]

n = grade efficiency parameter [-]

f_s = settling parameter [-]

g = gravitational constant [m/s^2]

Q = volumetric centrifuge throughput [m^3/s]

η = carrier viscosity [N s/m^2]

$\Delta\rho$ = density difference between liquid and solid phases [kg/m^3]

R_o = outer disk radius [m]

R_i = inner disk radius [m]

ω = angular bowl velocity [rad/s]

Σ = equivalent centrifuge settling area [m^2]

Z = number of disks in centrifuge disk stack [-]

θ = disk angle [rad]

V_{in} = feed broth volume [m^3]

$e_{sol,in}$ = concentration of dissolved ADH in feed broth [U/m^3]

$p_{sol,in}$ = concentration of dissolved protein in feed broth [kg/m^3]

$e_{ins,in}$ = concentration of intracellular ADH in feed broth [U/m^3]

$p_{ins,in}$ = concentration of intracellular protein in feed broth [kg/m^3]

$e_{sol,out}$ = concentration of dissolved ADH in supernatant [U/m^3]

$p_{sol,out}$ = concentration of dissolved protein in supernatant [kg/m^3]

$e_{ins,out}$ = concentration of intracellular ADH in supernatant [U/m^3]

$p_{ins,out}$ = concentration of intracellular protein in supernatant [kg/m^3]

$m_{c,in}$ = size-distributed mass of undisrupted cells in feed [kg]

$m_{c,in,i}$ = mass of undisrupted cells in feed having diameter d_i [kg]

$m_{w,in}$ = size-distributed mass of feed waste solids [kg]

$m_{w,in,i}$ = mass of feed waste solids having diameter d_i [kg]

λ_1 = fraction of dissolved enzyme or protein denatured from process operation [-]

λ_2 = fraction of dissolved enzyme or protein exiting in sediment stream [-]

f_{sup} = fraction of feed stream exiting as the supernatant [-]

V_{sup} = supernatant volume [m^3]

V_{sed} = sediment broth volume [m^3]

$m_{w,out,\,i}$ = mass of supernatant waste solids having diameter d_i [kg]

$m_{w,out}$ = size-distributed mass of supernatant waste solids [kg]

$m_{w,out,\,i}$ = mass of undisrupted cells having diameter d_i in supernatant [kg]

$m_{w,out}$ = size-distributed mass of undisrupted cells in supernatant [kg]

w_{out} = concentration of supernatant waste solids [kg/m^3]

Precipitation

V_z = precipitant volume needed [m^3]

z_{in} = precipitant input concentration [kg/m^3]

z_{out} = precipitant output concentration [kg/m^3]

V_{out} = broth volume after precipitant addition [m^3]

ϕ_e = fraction of soluble enzyme precipitated [-]

ϕ_p = fraction of soluble protein precipitated [-]

$e_{sol,in}$ = concentration of dissolved ADH in feed broth [U/m^3]

$p_{sol,in}$ = concentration of dissolved protein in feed broth [kg/m^3]

$e_{ins,in}$ = concentration of intracellular ADH in feed broth [U/m^3]

$p_{ins,in}$ = concentration of intracellular protein in feed broth [kg/m^3]

$e_{sol,out}$ = concentration of dissolved ADH remaining in precipitated solution [U/m^3]

$p_{sol,out}$ = concentration of dissolved protein remaining in precipitated solution [kg/m^3]

$e_{ins,out}$ = concentration of intracellular ADH remaining in precipitated solution [U/m^3]

$p_{ins,out}$ = concentration of intracellular protein remaining in precipitated solution [kg/m^3]

w_{out} = waste solids concentration in unprecipitated solution [kg/m^3]

w_{out} = waste solids concentration in precipitated solution [kg/m^3]

Chapter 6

Centroid-Based Sampling Strategy for Kriging-Based Global Modeling

The kriging algorithm presented in Chapter 3 is an iterative global modeling technique in which initial models built from randomly chosen sampling sets are refined based on the incorporation of additional sampling information. The generation of a sampling set via randomization creates the possibility that a poor initial model will be constructed. Model accuracy can be improved based on sampling at locations where the predicted uncertainty is highest, where model discrepancy in the predicted objective is highest between consecutive iterations, and where the predicted objective is lowest relative to the corresponding values at nearby test points. However, the overall sampling expense required to obtain an accurate model can be higher if extensive model refinement is required. The contribution of the work in this chapter is the presentation of a new sampling strategy is presented for kriging-based global modeling. The strategy is employed within a kriging/response surface (RSM) algorithm for solving NLP containing black-box models[55]. As mentioned in previous chapters, black-box models describe systems lacking the closed-form equations necessary for conventional gradient-based optimization. System optima can be alternatively found by building iteratively updated kriging models, and then refining local solutions using RSM. The application of the new

sampling strategy enables accurate global models to be obtained at lower sampling expense relative to a strategy employing randomized and heuristic-based sampling for initial and subsequent model construction, respectively. Based on the new strategy, the initial kriging model is built using sampling information obtained at the feasible region's convex polytope and centroid. Updated models are obtained by incorporating additional sampling information obtained at Delaunay triangulation centroids. The new sampling algorithm is applied within the kriging-RSM framework to several numerical and industrial examples to demonstrate proof of concept.

6.1 Introduction

In Chapter 3, the solution of process design problems whose problem formulation contains black-box models and noisy variables has been addressed using a kriging-RSM strategy that targets the attainment of globally optimal operating conditions. The kriging-RSM algorithm relies on the generation of an accurate surrogate model using the kriging methodology as presented in Chapter 3, in order to identify warm-start iterates for further local optimization using RSM, the methodology of which is presented in Chapter 2. At each iteration of the kriging method, the previous global model is updated using additional sampling information collected at points of interest identified from the earlier predictor, such as where the model predictions are lowest, or alternatively where the model uncertainty is highest. The initial model is built from a set of randomly selected feasible points dispersed throughout the feasible region. Different nominal models are generated from different initial sampling sets. However the overall sampling expense is higher when poor initial models are built. The generation of inaccurate nominal models

suggests that the initial sampling information fails to contain the information needed in order for the kriging-based global modeling algorithm to accurately identify important system behavior in the form of ridges, valleys, optima, or stationary points. Therefore, based on the current sampling strategy of performing 1) random sampling for initial modeling, and 2) heuristic-based sampling at model-based points of interest, there exists an opportunity for reducing the global modeling sampling expense. The goal of reducing sampling expense is motivated further by the need to eliminate random sampling as a method for initial modeling. The attractiveness of a surrogate model-building technique is increased if it can be known *a priori* how many sampling experiments are needed before an accurate model can be reasonably expected. Because random sampling is inherently uncertain, the number of sampling experiments required before an accurate estimator is built can vary over a wide range. A technique which does not rely on random sampling removes the question of how much additional sampling expense is required due to the initial set having been chosen with some measure of uncertainty.

The contribution of the work presented in this chapter is the presentation of a sampling technique that relies on sampling at Delaunay triangle centroids. The new technique has been successfully employed in attaining accurate global kriging models and the main advantages of the method are that, 1) a complete set of local and global optima can be found for NLP containing black-box models and/or noisy variables, and 2) fewer sampling experiments are required to find the complete set of optima relative to a sampling strategy employing randomized/heuristic sampling. Therefore, the centroid-based sampling technique is proposed as a novel sampling approach for other sampling-based modeling methods. The main features of the new sampling technique involve:

1) sampling at the vertices and centroid of the feasible region convex polytope for initial modeling, and 2) sampling at the Delaunay triangle centroids for model refinement.

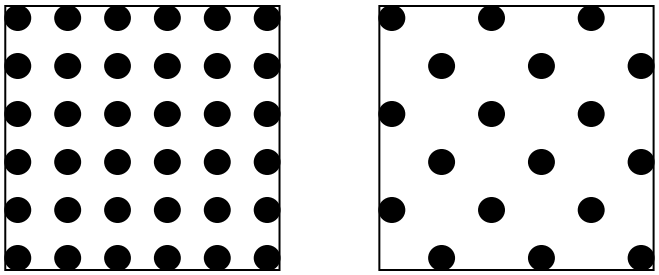
6.1.1 Literature Review

The field of sampling-based techniques focuses on not only optimizing the number of and spatial arrangement of sampling points, but also the test set at which predictions are to be obtained. The designation "sampling techniques" can refer to either, 1) an algorithm specifically targeted at identifying where field/computational experiments should be performed, as is the goal of the proposed method in this paper, or alternatively, 2) a method for identifying the locations where model predictions should be generated. Considering the first class of sampling techniques, four main algorithmic subclasses exist for choosing any n_p samples from a set of k_{test} feasible vectors for modeling: 1) random, 2) systematic, 3) stratified, and 4) cluster. Random sampling, however computationally convenient to implement, has the limitation that the information obtained from a sampling set poorly representing the feasible region may fail to result in accurate model development. Systematic sampling is another frequently employed method which relies on a set of heuristics for identifying sampling vectors, such as every seventh feasible point from the set of ordered k_{Test} points. A key limitation of this technique is that a poorly chosen value for n_p can result in too few sampling experiments being performed, resulting in the generation of an inaccurate model, or conversely, too many sampling experiments being conducted, in the sense that redundant system information is identified from sampling points located close to one another. The centroid-based sampling technique used in kriging modeling, the algorithmic details of which are presented in

more detail in Section 6.2.1, is an example of a systematic sampling procedure. For this method, the sampling expense is controlled by sampling at a single point within any Delaunay triangle, and terminating the algorithm after the fourth iteratively improved global model has been built. Stratified sampling such as Latin Hypercube Sampling[56] can be considered as the application of any other sampling algorithm to each one of a subset of the k_{Test} points separated by some stratification rule. The advantage of this technique is that a more uniform sampling arrangement is generated in contrast to random sampling. Cluster sampling requires sampling to be performed for all sampling vectors in a given strata of k_{Test} points, while neglecting the performance of any sampling in other strata. The advantage of cluster sampling is that an extensive amount of system information can be obtained over a localized subregion of interest, such when neighborhoods containing potential optima are identified and additional sampling information is needed to refine the current solution. The sampling templates used in RSM, given in Figure 6.9, can be considered as cluster sampling algorithms since no sampling is performed outside the subregions of interest.

For the class of methods directed at the identification of the locations of all k_{Test} points requiring model generation, the generation of model estimates at discretized feasible region locations as shown in Figure 6.1(a) is the most intuitive and easily implementable algorithm. However, this method becomes impractical as the problem dimensionality increases, since the number of test points is a multiplicative function of the number of grid points generated for each dimension. Since one goal of modeling techniques is that the modeling algorithm itself be as computationally inexpensive as possible, alternative pseudo-uniform sampling schemes such as Latinized, Hammersley,

and Halton sampling as shown in Figure 6.1(b) have been employed to alleviate this issue for problems whose feasible regions have initially been predominantly hypercuboidal but have since been extended to other symmetrical regions.

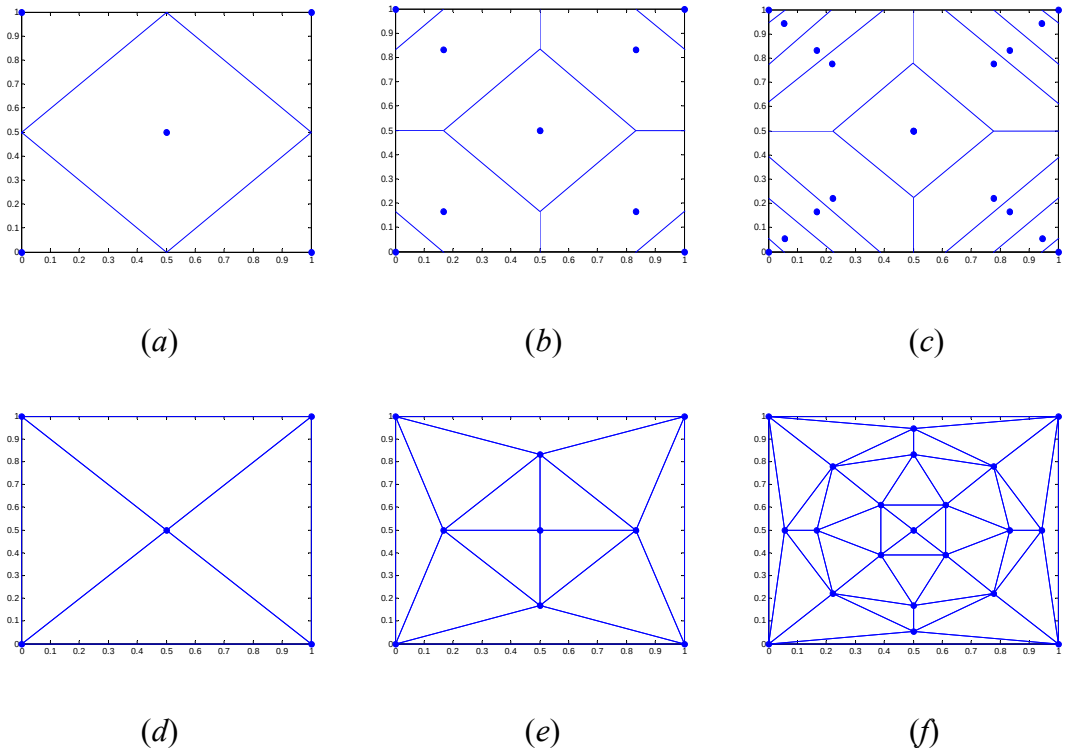


(a) (b)

Figure 6.1. Generation of a test point set based on discretization (a) or the application of a Latinized/Hammersley/ Halton algorithm (b).

These sampling methods generate sampling points in each one of a set of smaller nonoverlapping hypercubes. The Centroidal Voronoi tessellation techniques applied by Romero et. al[57] are effective for generating pseudo-uniform sets using modified Voronoi diagrams when the problem dimensionality increases, a limitation of the aforementioned sampling techniques. The Voronoi diagram is considered to be the dual of a Delaunay triangulation; however the proposed sampling method in this paper relies on sampling at Delaunay simplex centroids instead of the corresponding Voronoi simplex centroids. The number of simplex vertices in a Delaunay triangle is fixed at (n+1), where n is the problem dimensionality, whereas for a Voronoi simplex it is often greater than (n+1). Some of the Voronoi simplices will have a small hypervolume, while others will have a much larger one as shown in Figures 6.2(a) – 6.2(c). Conversely, the volume of the

simplices formed by applying Delaunay triangulation, on the other hand, are generated so as to have approximately equal volume, as shown in Figures 6.2(d) – 6.2(f). Therefore, the application of a Voronoi-based sampling technique in which samples are obtained at the centroids of iteratively updated Voronoi diagrams, can result in redundant sampling information being obtained as a result of sampling points being close to one another between neighboring Voronoi simplices as shown in Figure 6.2(c). In the proposed sampling method, this problem is addressed by terminating the kriging-RSM algorithm after the initial model has been refined only two or three times.



(*a*) (*b*) (*c*)

(*d*) (*e*) (*f*)

Figure 6.2. Generation of sampling points concentrated along feasible region diagonals when sampling is performed at simplex centroids for a set of iteratively updated Voronoi diagrams (a) – (c), a nonuniform sampling arrangement, in contrast to the more uniform sampling arrangement obtained from sampling at simplex centroids of iteratively updated Delaunay triangulations (d) – (f).

Other sampling designs optimize some other estimator property, such as minimization of estimator variance at a given subset of points of interest, maximization of the estimator confidence region, or a user-designed merit function created for comparison of multiple design alternatives. Modified central composite designs, squared-error designs, orthogonal arrays, and minimax designs are a few of the designs reviewed in more detail in Simpson et. al[58]. In the next section, the targeted NLP problem is first mathematically formulated. Following this, the algorithmic details of the new centroid-based sampling method, implemented within a kriging-RSM framework for solving this NLP, are then presented.

6.1.2 Problem Definition

The problem addressed in this work can be expressed in the following form:

$$
\begin{aligned}
&\textit{min} \quad F(x, z_1, z_2) \\
&\textit{s.t.} \quad g(x, z_1, z_2) \le 0 \\
&\qquad h(x, z_1) = 0 \\
&\qquad z_2(x) = \Gamma(\mathrm{x}) + \varepsilon(x) \\
&\qquad \varepsilon(x) \in N\left(x \mid \mu, \sigma^2\right) \\
&\qquad N\left(x \middle| \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \\
&\qquad x \in \Re^n
\end{aligned}
\tag{6.1}
$$

Based on this formulation, the vector of continuous variables is given by x. The objective function is represented by F, and the deterministic variables z_1 describe outputs whose modeling equations $h(x,z_1)$ are known. The vector of stochastic output variables z_2 exists when the input-output functionality $\Gamma(x)$ is black-box. The stochastic value of each z_2 variable is modeled as the sum of its corresponding deterministic output and an

additive noise component ε that is a normally distributed error term whose mean μ and variance σ^2 may be a function of the input specification (x,y_2). For field experiments, an estimate of the parameters μ and σ^2 can be obtained by conducting replicate experiments for a given sampling vector. It should be noted that the modeled values of μ and σ^2 may need to assume a range of values if it is known from prior field data that the noise is spatially variant with respect to x. Design and operating equations are given by $g(x,z_1,z_2)$, where the feasible space for x may be further constrained based on the feasible space defined by $z_2(x)$.

In our previous work[21,22], a Branch-and-Bound Kriging-RSM-Direct Search algorithm has been successfully applied to process synthesis problems in which integer variables appear both inside and outside the black-box function as presented more fully in Chapter 5. The formulation of this problem class is the analog of (6.1) which would now contain two integer variable sets (y_1,y_2) as additional dependent variables for F, g, h, and z_2 as given by Equation (5.1). Both integer variable sets are required in order to separately designate the integer variables residing inside or outside of Γ. Even though the sampling strategy presented in this paper is applied to problems containing strictly continuous variables, a natural future work would be the extension of this technique to this more general problem class.

6.2 Solution Approach

The optimization strategy used to obtain the solution of the problem described by Equation (6.1) relies on generating an iterative sequence of kriging global models in order to identify promising warm-start iterates for local optimization. At each stage of the

kriging algorithm, new sampling data are used to update the previously built global model. Over the course of applying kriging, a sequence of global models is generated, with the late models closely resembling one another. Once the system predictor has converged, the global model is considered accurate and the best local solutions are then identified for subsequent local optimization via response surface methodology (RSM). The application of RSM to at least a subset, if not all of, the kriging model optima improves the chances of not only finding a system global optimum, but also establishing a confidence in its classification as a global optimum due to the successful identification of additional local optima having inferior objective values *F*. A flowchart summarizing the kriging-RSM algorithm, which relies on centroid-based sampling for global model construction and refinement, is shown in Figure 6.3 and is referred to as the KC-R algorithm for the remainder of this paper.

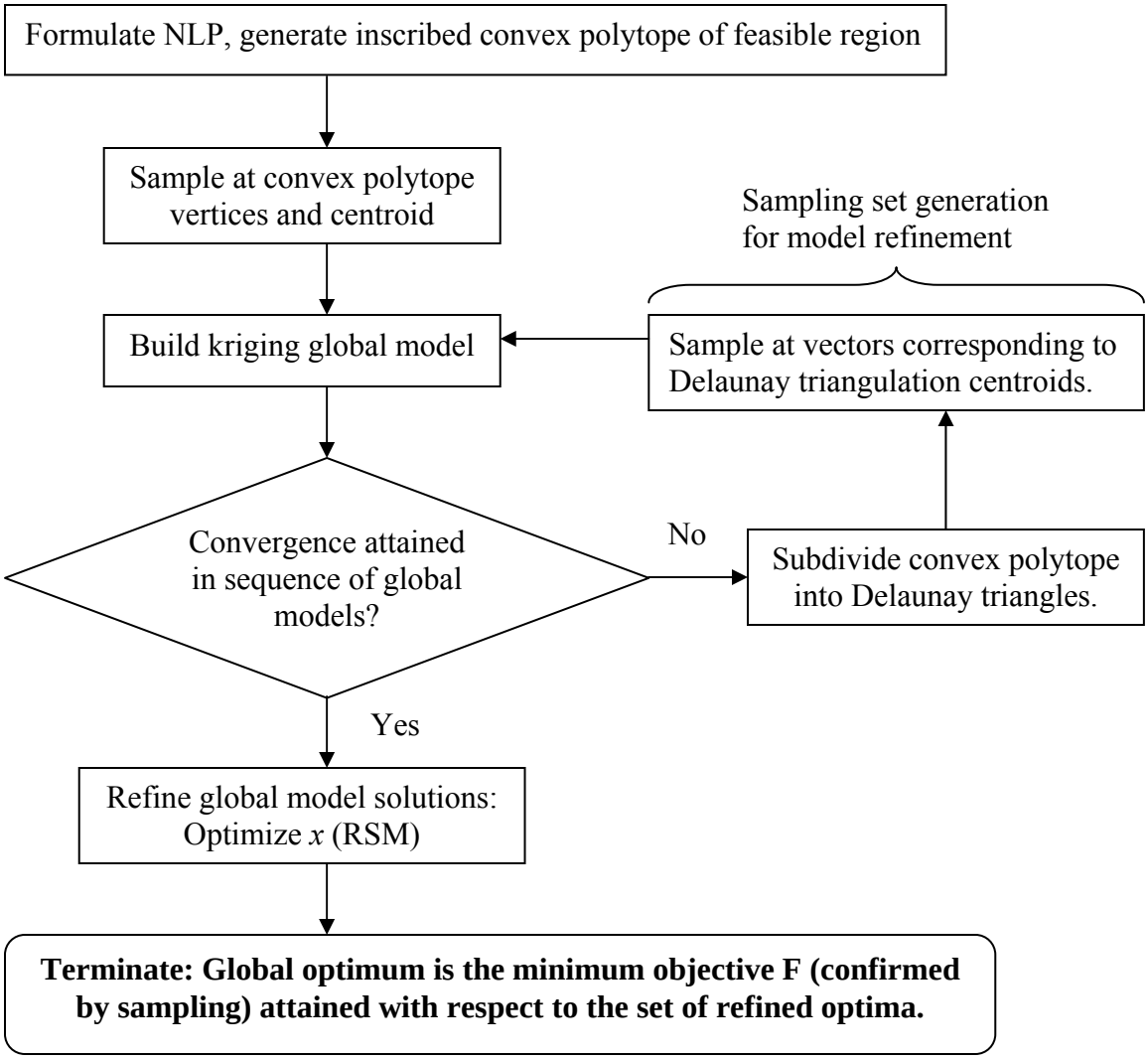


Figure 6.3. Flowchart of centroid-based sampling strategy embedded within a kriging-RSM modeling/optimization algorithm; this technique is designated as the KC-R method.

The basic steps for building and refining global models based on the incorporation of sampling information obtained at Delaunay triangle centroids are as follows: 1) sample at the feasible region convex polytope and its centroid, 2) build the global model, 3) sample

at Delaunay triangle centroids, and 4) repeat steps 2) and 3) until the sequence of global models has converged. In Figure 6.4, sampling templates are presented which correspond to the construction and refinement of a 2-D global model whose output variable z_2 is a function of two variables x_1 and x_2, and whose feasible region is box-constrained.

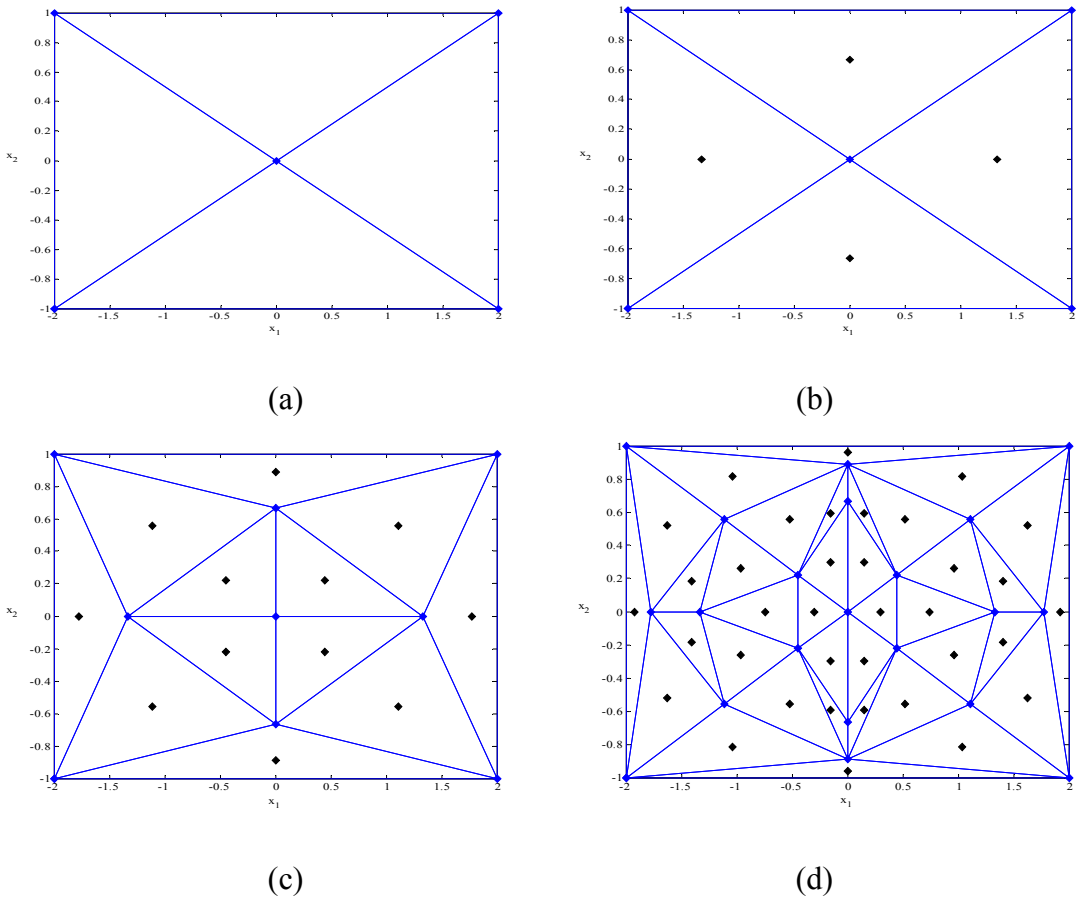


Figure 6.4. Generation of sampling templates used to build kriging models for a 2-D box-constrained problem, in which five sampling points are used for initial model construction (a), and four (b), twelve (c), and thirty-six (d) additional sampling points are used in subsequent model building.

In Figure 6.4(a), the Delaunay triangulation of the initial sampling set is shown. Once the initial model has been built, it is not yet considered to be an accurate estimator, and therefore requires refinement using new sampling information. The new sampling set is comprised of vectors corresponding to the Delaunay triangle centroid locations, designated by the black dots, as shown in Figure 6.4(b). If the second model, built using the set of nine sampling points, is still considered inaccurate, another Delaunay triangulation is performed. The centroids of the Delaunay simplices are shown in Figure 6.4(c), and the number of sampling points in the new sampling set is twelve. If the third model, constructed from twenty-one sampling points, still requires refinement, Delaunay triangulation is performed yet again. The thirty-six Delaunay triangle centroids are shown in Figure 6.4(d). After the fourth iteration, it is supposed that for this hypothetical system the sequence of global models has converged. At this point, the set of local kriging solutions are identified, and further local optimization using RSM is initiated.

The kriging solutions are defined as the vectors for which the objective function value F is lower than the corresponding objectives for the set of nearest neighbors. For all the presented examples in Section 6.3, the set of test points is generated using discretization. The generation of a gridded test point set can result in a very high number of test points for dimension $n \geq 10$, increasing the modeling/optimization computational expense.

In addition, at each iteration of the centroid-based kriging algorithm, a call is made to the Matlab 2008b implementation of the Delaunay triangulation algorithm. This particular version of the Delaunay triangulation algorithm relies on convex hull construction using the qHull method[58], whose computational complexity increases rapidly as a function of problem dimension and therefore is nontrivial when the problem

dimensionality is greater than ten. If n_p is defined as the total number of sampling points, Barber et al.[59] conjecture that the computational time required to obtain the convex hull of r points having a maximum number of facets f_r is O(n_p log r) for problems of dimensionality equal to or less than three, and O($n_p f_r/r$) when the problem dimensionality is greater than four. The computational complexity required to determine the number of facets based on r points for a problem of dimension n determined according to the following equation:

$$f_r = O\left(\frac{r^{\lfloor n/2 \rfloor}}{\lfloor n/2 \rfloor !}\right) \tag{6.2}$$

A more efficient Delaunay triangulation algorithm is required to alleviate this problem; therefore the kriging-RSM modeling/optimization algorithm employing centroid-based sampling for kriging model construction is efficient only for problems with less than ten input variables. In subsections 6.2.1, 6.2.2, and 6.2.3, respectively, the algorithmic details of the centroid-based sampling strategy, the kriging technique, and response surface methodology are presented.

6.2.1 Centroid-Based Sampling Strategy

For kriging-based global modeling, sampling is initially performed at the vertices of the region's convex polytope and its centroid. One condition required for application of the KC-R version of the kriging-RSM algorithm is that the problem's feasible region be defined by a convex polytope. The RSM phase of the optimization algorithm consists of the sequential optimization of local models; however early termination of the method can occur once iterates have reached a boundary and the optimal step direction requires

movement outside the feasible region. For an n-dimensional problem, the set of M convex polytope boundaries can be given in terms of a matrix formulation:

$$Ax \le b \tag{6.3a}$$

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \qquad m = 1 \ldots M \tag{6.3b}$$

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \qquad m = 1 \ldots M \tag{6.3c}$$

A vector x is a member of the convex polytope if it satisfies Equation (6.4) for any n constraints and is also feasible with respect to the others.

$$\tilde{A}x = \tilde{b} \tag{6.4}$$

where $\tilde{A}$ is an arbitrary (n x n) submatrix of A and $\tilde{b}$ is the corresponding (n x 1) submatrix of b. In Figure 6.5, the convex polytope is shown by the black dots that serve as the vertices of a diamond-shaped feasible region.

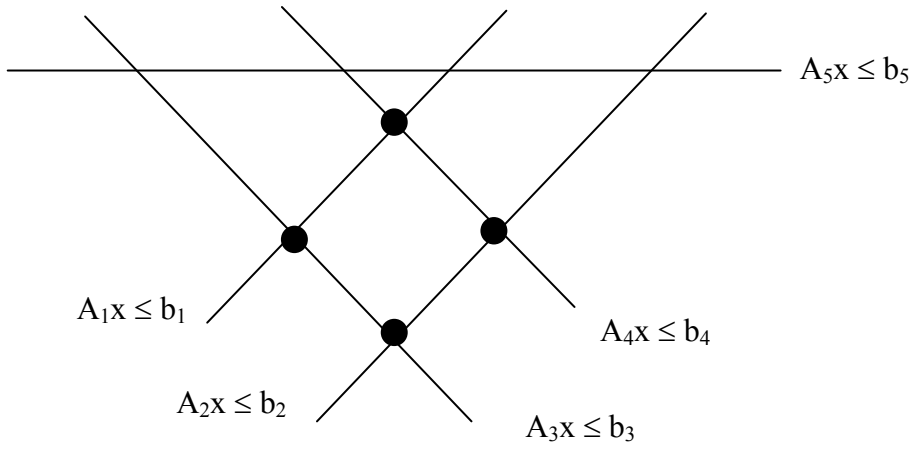


Figure 6.5. Application of a feasibility check to identify the set of sampling points located at the convex polytope vertices.

Once the set of vertices comprising the convex polytope has been obtained, the next step is to generate the centroid of these points. Let the set of N_V convex polytope vertices be denoted as V, and let the i^{th} vertex be denoted as $x_{V,i}$. The corresponding convex polytope centroid can then be obtained as the arithmetic mean of the vertex points as given in the following equation:

$$x_C = \frac{\sum_{i=1}^{N_V} x_{V,i}}{N_V} \tag{6.5}$$

For a convex polytope, its centroid will always reside within the feasible region. The strategy of including the convex polytope centroid as a sampling point for initial modeling is motivated by the idea that the system information obtained from sampling at this point is equivalent to the information obtained from sampling at a sequence of points converging to the convex polytope centroid. Once the vector x_c has been determined, the sampling set S^1 used to build the initial kriging-based global model is defined as follows:

$$S^1 = \{V, x_c\} = \left\{ x_{V,i}\big|_{i=1\ldots n}, x_c \right\} \tag{6.6}$$

Once the initial kriging model is generated, it may not accurately describe the system behavior at locations inside the feasible region. Model inaccuracy occurs because the kriging-based global modeling is an interpolatory method whereby predictions are generated as a weighted sum of nearby sampled function values. Kriging model behavior resembles the behavior exhibited by models constructed using inverse distance weighting methods, in that model accuracy generally decreases as a function of increasing distance between a test point and the nearby sampling points. The centroid location is likely to be far away from the convex polytope vertices, and model inaccuracy will be highest at test points approximately midway between the centroid and the vertices. In order to attain significant model improvement, additional sampling is therefore performed at locations where the uncertainty is highest.

To address this problem, a Delaunay triangulation is applied to the current sampling set S^m, where S^m refers to the additional sampling data used together with the S^{m-1} dataset to build the m^{th} kriging model as presented in the kriging algorithm flowchart shown in Figure 6.8. The Delaunay triangulation algorithm attempts to subdivide the feasible region into a set of adjacent, nonoverlapping simplicial subregions having uniform volume. The Delaunay triangulation of a convex polytope having k possible subdivisions will be the one that contains the highest minimum simplicial angle. Each simplicial subregion contains $(n+1)$ vertices which serve as the vectors used in the corresponding centroid calculation given by Equation (6.5). The new sampling set S^{m+1} is then generated as the set of all Delaunay triangulation centroids.

In Table 6.1, the differences between the current and new sampling algorithms are presented. Either one of the sampling algorithms can be applied within the framework of the kriging method in order to iteratively generate an accurate global model. The details of the kriging technique are presented in the next subsection.

Table 6.1. Differences between the two sampling strategies for building iteratively updated global kriging models.

	Old Sampling Strategy	New Sampling Strategy
Model Creation	Sample randomly	Sample at convex polytope and centroid
Model Refinement	Sample at locations where there is: 1) minimum model prediction 2) maximum model uncertainty 3) highest current/previous model discrepancy	Sample at each one of the N_D Delaunay triangulation centroids

6.2.2 Kriging Methodology

The kriging model consists of predictions obtained at all test points S_k, in which each kriging estimate is determined as a weighted sum of nearby sampled function values. The kriging model is built with respect to the NLP objective since it is the objective function that is being optimized instead of the process outputs. The steps for obtaining a prediction at S_k are as follows: 1) determination of covariance function coefficients based on sampling data; 2) calculation of the covariance $Cov(d_{i,k})$ between the test point and each nearby sampling point; 3) generation of weighting values λ for each sampling point S_i close to S_k after solving the linear system $C\lambda = D$, where the elements of C and D are

$\{S_i,S_j\}$ and $\{S_i,S_k\}$ covariance values, respectively; and 4) estimation of the kriging predictor. The details of the methodology will now be presented.

From the set of sampling data contained in the current sampling set S^m, where m denotes the iteration index, squared output value differences $F_{i,j}$ are calculated as given by Equation (6.7) and plotted relative to sampling-pair distances as given by Equation (6.8):

$$F_{i,j} = \left[F(S_i) - F(S_j) \right]^2 \qquad i, j = 1 \ldots S^m, i \neq j \qquad (6.7)$$

$$d_{i,j} = \| S_i(x) - S_j(x) \|_2 \qquad (6.8)$$

From a scatter plot of $F_{i,j}$ as a function of $d_{i,j}$, a semivariance function is then fitted. Due to the plot complexity as shown in Figure 6.6(a), the best fit to one of the established semivariance models in the literature[17] is not usually immediately apparent.

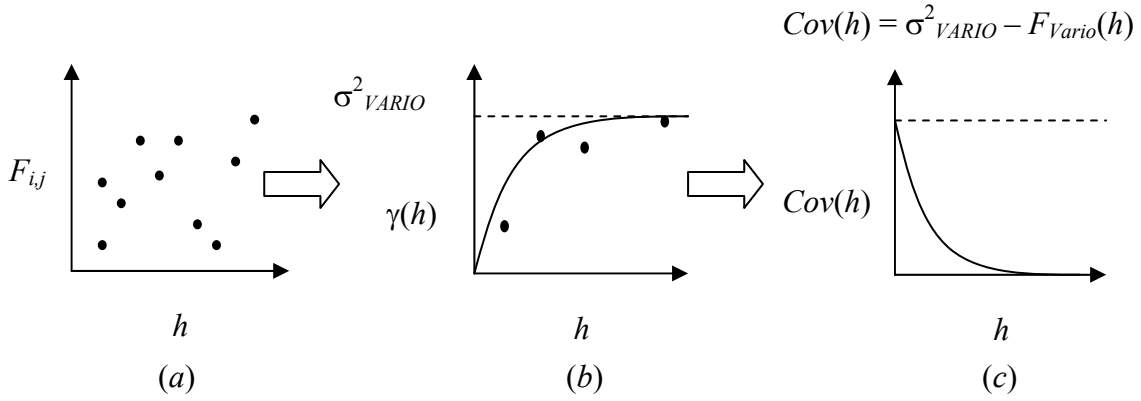


Figure 6.6. Data smoothing applied to squared function differences $F_{i,j}$ (a) in order to obtain a semivariogram fit (b) and covariance function fit (c).

To alleviate this problem, data smoothing is applied, and the semivariance function is then fitted to the reduced set of scatterpoints known as semivariances γ as shown in Figure 6.6(b). A set of P equidistant intervals are defined between zero and the

maximum $d_{i,j}$ distance. The p^{th} interval midpoint is denoted by h_p, and the semivariance corresponding to the p^{th} interval, $\gamma(h_p)$, is obtained by averaging the set of squared function differences falling inside this interval as given by Equation (6.9):

$$\gamma(h_p) = \frac{1}{2N(h_p)} \sum_{r=1}^{N(h_p)} F_{i,j} \qquad p = 1...P, \quad i \neq j \qquad (6.9)$$

where $N(h_p)$ is the number of sampling pairs $\{S_i, S_j\}$ whose separation distance $d_{i,j}$ lies inside the p^{th} interval. The semivariance function behavior typically rises from zero to an asymptotic maximum known as the sill σ^2_{VARIO}. In order to generate the corresponding covariance function as displayed in Figure 6.6(c), the semivariance function is then reflected between the *x*-axis and the sill. Once the covariance function has been obtained, the covariance between any two S_i-S_j or S_i-S_k vectors is determined by substituting $d_{i,j}$ or $d_{i,k}$ into the model equation. The kriging weights are then obtained as the solution of a linear system of equations in which the LHS consists of a matrix of $\{S_i,S_j\}$ covariances, and the RHS consists of a vector of $\{S_i,S_k\}$ covariances between $k_{Cluster}$ nearest-neighbor sampling points S_i and S_k. If the weights are forced to sum to unity, the linear system can be recast in a Lagrangian formulation as given by Equation (6.10):

$$\begin{bmatrix} \lambda(S_k) \\ \lambda'(S_k) \end{bmatrix} = \begin{bmatrix} Cov(d_{i,j}) & 1 \\ 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} Cov(d_{i,k}) \\ 1 \end{bmatrix} \quad i, j = 1...k_{Cluster} \;,\; i \neq j \qquad (6.10)$$

where $\lambda(S_k)$ and $\lambda'(S_k)$ represent the weight vector and the Lagrange multiplier, respectively. Once the weights are obtained, the kriging prediction $F(S_k)$ and its expected variance $\sigma^2_k(S_k)$ are obtained according to Equations (6.11) and (6.12), respectively:

$$F(S_k) = \sum_{i=1}^{k_{Cluster}} F(S_i)\lambda(S_i) \qquad (6.11)$$

$$\sigma_k^2(S_k) = \sigma_{VARIO}^2 - \sum_{i=1}^{k_{Cluster}} \lambda(S_i)\, Cov(d_{i,k}) - \lambda'(S_k) \qquad (6.12)$$

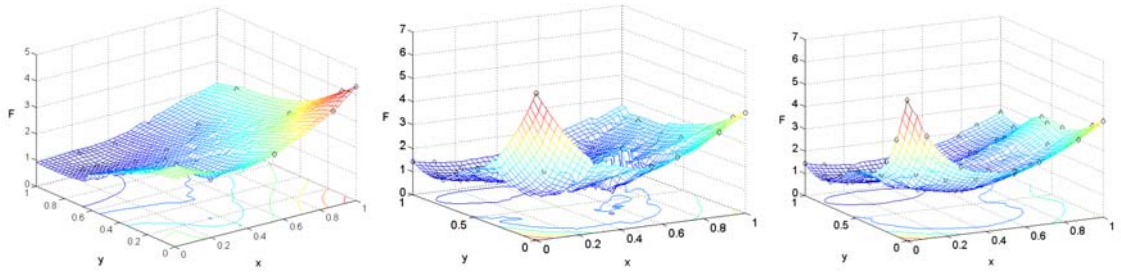
The methodology is then applied to another test point until objective function estimates at all test points have been generated. If additional sampling data are obtained, a new covariance function can be generated. Based on the updated covariance function, new kriging estimates can be obtained for all k_{Test} sampling points and a refined global model can be created. For each global model, its corresponding average predictor value μ is compared against its counterpart based on the previous model. Once convergence has been achieved in μ, further refinement is terminated.

Let the iteration index m refer to any property based on the m^{th} kriging model, and let Tol_{Krig} be a percentage stopping tolerance. A sample range for Tol_{Krig} would be any value between one and ten percent. The m^{th} average prediction value μ_m is defined as the average of the set of kriging predictions and sampled function values as given by Equation (6.13):

$$\mu_m = \frac{1}{\left(card(S^m) + k_{Test,m}\right)} \left(\sum_{i=1}^{card(S^m)} F_i(S_i) + \sum_{r=1}^{k_{Test,m}} F_r(S_k) \right) \qquad (6.13)$$

where card(S^m) refers to the number of sampled function values in the current sampling set S^m and $k_{Test,m}$ refers to the number of test points employed in constructing the m^{th} global model. A nominal value of μ_0 is obtained by averaging the sampled function values from S^1. Once the m^{th} global model has been constructed, μ_m/μ_{m-1} is evaluated. If this ratio falls inside the interval ($1 \pm Tol_{Krig}$), the m^{th} global model is considered accurate, and no additional updating occurs. Otherwise, another model is built using additional sampling data. The locations of new candidate sampling points are obtained using either

one of the two methods described in Table 6.1. When the non-centroid-based sampling method is employed, global model improvement is enforced by requiring that all new sampling points reside some minimum distance apart from one another. This practice ensures that the new sampling set will not consist of clustered points located at a single high-variance region, thereby de-emphasizing local model refinement. A set of global models is presented in Figure 6.7 which shows iterative stabilization of the kriging predictor after it has been updated using the randomized/heuristic-based sampling strategy described in Table 6.1.



(*a*) (*b*) (*c*)

Figure 6.7. Kriging model generated at initial (a), intermediate (b), and final (c) stages.

The procedure for obtaining a prediction at S_k, referred to as the kriging algorithm, can be summarized as follows. First, the feasible region is characterized and iteration index m is initialized at unity. A nominal sampling set S^1 is specified using one of the two techniques presented in Table 6.1. The location S_k is specified and $k_{Cluster}$ nearest-neighbor sampling points are chosen from S^1 that are nearest to S_k as given by Equation (6.8). If the centroid-based sampling technique is employed, the value of $k_{Cluster}$ is set at $(n+1)$, otherwise for the non-centroid-based method, referred to as the K_{NC} algorithm, it is a user-determined parameter that is set at seven for the presented examples in Section 3.

Semivariances are then generated using all sampling data from S^m. The best semivariance model is fitted using least squares, and the complementary covariance function is then obtained. The matrices on the RHS of Equation (6.10) are then constructed from submatrices $h_{Cluster}$, h_0, C, and D, as given by Equations (6.14) – (6.17). The matrices C and D are augmented in order to remain consistent with the Lagrangian formulation given in (6.10).

$$h_{Cluster} = \begin{bmatrix} 0 & d_{1,2} & \cdots & d_{1,k_{Cluster}} \\ d_{2,1} & 0 & \cdots & d_{2,k_{Cluster}} \\ \vdots & \vdots & \ddots & \vdots \\ d_{k_{Cluster},1} & d_{k_{Cluster},2} & \cdots & 0 \end{bmatrix} = \left[d_{i,j} \right] \qquad i, j = 1 \ldots k_{Cluster} \tag{6.14}$$

$$h_0 = \begin{bmatrix} d_{1,k} \\ d_{2,k} \\ \vdots \\ d_{k_{Cluster},k} \end{bmatrix} = \left[d_{i,k} \right] \qquad i = 1 \ldots k_{Cluster} \tag{6.15}$$

$$C = \begin{bmatrix} Cov(h_{Cluster}) & 1 \\ 1 & 0 \end{bmatrix} \tag{6.16}$$

$$D = \begin{bmatrix} Cov(h_0) \\ 1 \end{bmatrix} \tag{6.17}$$

The kriging weights λ are then obtained from solving the linear system of equations as presented in (6.10) and the prediction $F(S_k)$ and its variance $\sigma_k^2(S_k)$ are determined. The weights are then recalculated for each one of the remaining k_{Test} sampling vectors in order to generate corresponding estimates for $F(S_k)$. Once the global mapping has been constructed, the value of μ_m is determined from Equation (6.13) and compared against μ_{m-1}. If convergence is not achieved, the iteration index is advanced by unity and

additional sampling is performed based on application of the sampling rules as given in Table 6.1. A new covariance function is built, new kriging estimates $F(S_k)$ are generated, and an updated mapping is built. The procedure is terminated once convergence has been achieved in μ_m and the best local solutions are then identified for local optimization using RSM, whose algorithmic details are presented in the next subsection. A flowchart of the kriging algorithm is presented in Figure 6.8.

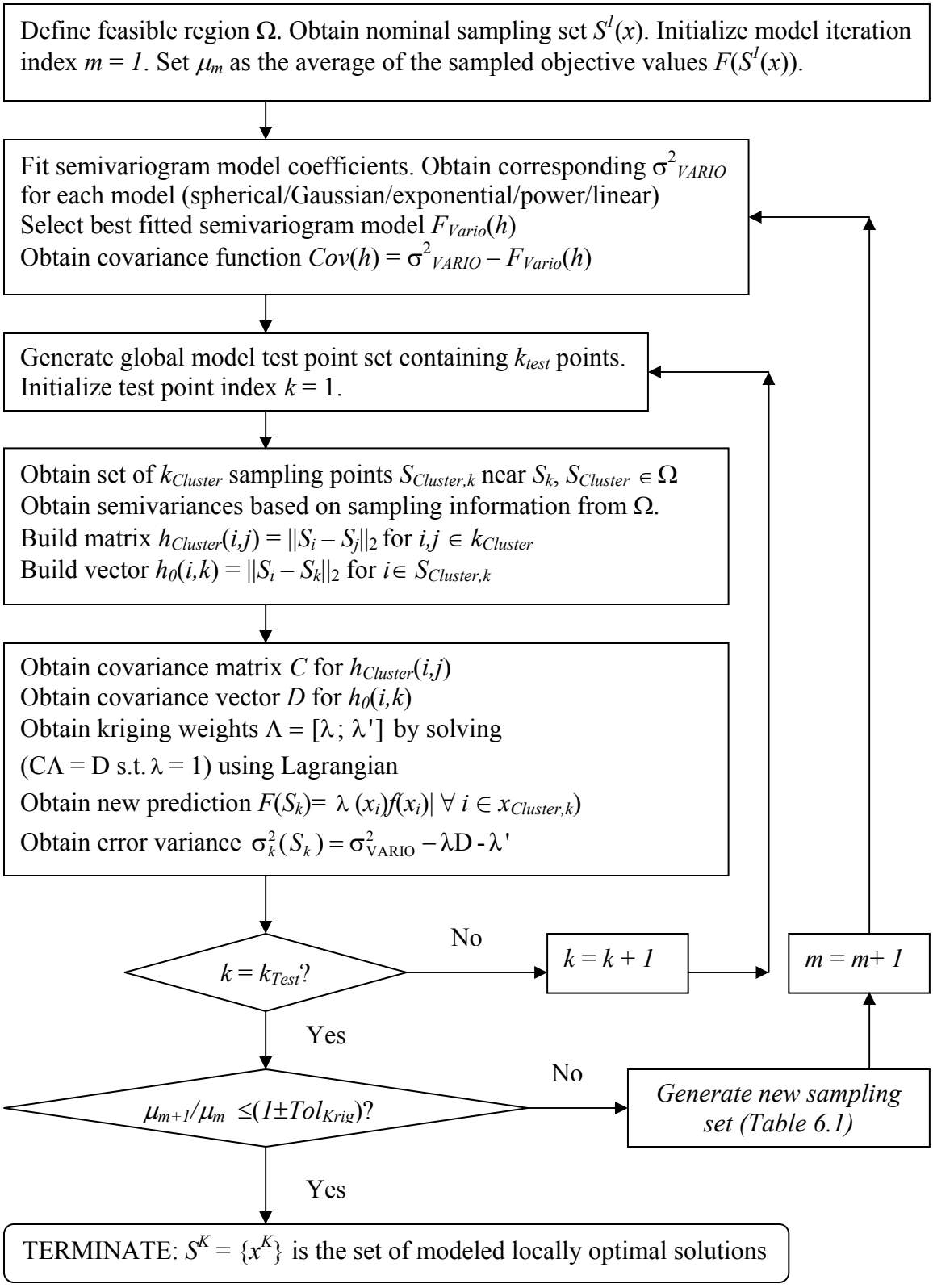
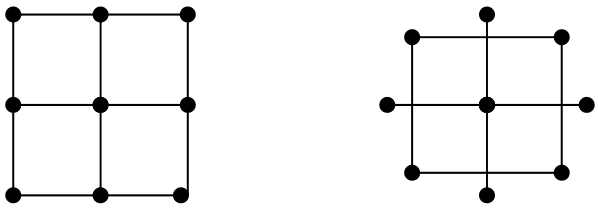


Figure 6.8. Kriging algorithm flowchart for building/refining a data-driven global model.

6.2.3 Local optimization using RSM

The response surfaces used in this chapter are quadratic polynomials containing bilinear terms which can be easily optimized using standard gradient techniques. If additional sampling at the response surface optimum yields a better solution, a new model is then built which is centered around this iterate, and the process continues until the objective has converged. The set of input points for building the response surface conforms to a stencil arrangement known as an experimental design that is centered about an iterate, which can be any of the best kriging solutions[14,24]. The factorial design for a problem in two dimensions requires a^b points, where a is the number of factors and b is the number of continuous input variables. For problems whose feasible regions are described by linear constraints, a factorial design, shown in Figure 6.9(a) for a 2-D problem, enables system behavior along a linear boundary to be effectively described.



(a) (b)

Figure 6.9. Factorial (a) and Central Composite Design (b) for 2-D response surface generation.

An alternative to the factorial design is the central composite design (CCD) shown in Figure 6.9(b) for a problem in two dimensions, which requires $(1 + 2n + 2^n)$ sampling points for a problem of dimension n. Response surfaces built according to the factorial

design are defined over an *n*-D hyper-rectangular region; however the same models built according to the CCD are instead defined over an *n*-D spherical region. When iterates are located along a linear boundary, the system behavior is captured more effectively if the response surface is built according to the factorial design. However, the CCD design is associated with a lower sampling expense relative to a factorial design since data are not obtained at every factor-level combination.

At the start of the algorithm, the iteration index w is initialized at a value of unity. A response surface is built around a kriging-optimal solution S^K by fitting sampling data obtained from a collocation set $S_{coll,w}$. The vectors which comprise $S_{coll,w}$ are determined by applying either one of the factorial or CC design templates for a predetermined initial model radius b_w. For the examples presented in Section 6.3, the nominal value of b_w is set at ten percent of each variable's operability range as defined by the difference in corresponding lower and upper bounds. The vector S^K and its corresponding objective value F^K comprise the nominal solution set $\{S_{opt,w},F_{opt,w}\}$. Once the response surface has been created, the optimum $S_{opt,w+1}$ having corresponding value $F_{opt,w+1}$ is determined using gradient methods. Sampling is performed at the model optimum vector in order to confirm objective value improvement. If the difference between the current and previous optimum $|F_{opt,w+1} - F_{opt,w}|$ falls below a prespecified criterion Tol_{RSM}, the algorithm terminates with $\{S_{opt,w+1,},F_{opt,w+1}\}$ established as the RSM solution. Otherwise, the iteration index is advanced by unity and another response surface having a new bound radius b_w is constructed at the new $S_{opt,w}$. At any iteration w, the value of b_{w+1} is different from b_w only if the Euclidean distance between the current and previous solution vectors is lower than the current radius b_w. During the later stages of the algorithm, $S_{opt,w+1}$ will

be near $S_{opt,w}$, signifying that the basin of the RSM optimum has been found. At this point, a more accurate description of the system behavior near the optimum can be attained using more localized response surfaces. Whenever iterates are close to the boundaries, lower-dimensional response surfaces are created by projecting the model onto constraints so as to prevent model generation based on an asymmetrical arrangement of the feasible sampling data[12]. A flowchart of the algorithm is presented in Figure 6.10.

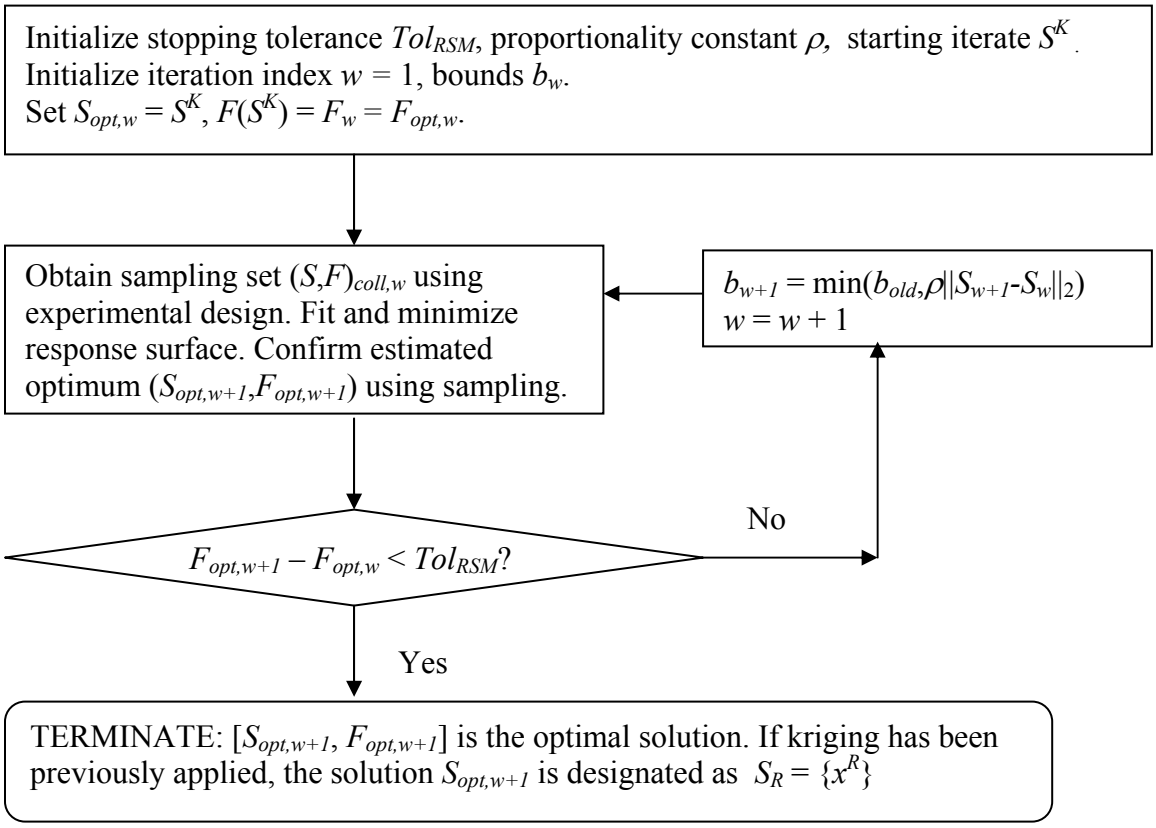


Figure 6.10. Flowchart of the RSM algorithm.

The RSM-optimal solution is denoted as $F(S^R) = F(x^R)$. Once its value has been attained, the remaining locally optimal kriging solutions are also optimized using RSM.

6.3 Examples

In this section, the performance of the centroid-based sampling algorithm, as implemented within the kriging-RSM optimization method, is evaluated based on its application to three global optimization test functions and two industrial case studies[54]. In the first case study, presented in subsection 6.3.4, the objective is to determine an optimal set of reaction conditions, given in terms of two species concentrations. This problem is solved initially with the only constraints being simple lower and upper bounds for each of the two reaction species. The problem is then solved when additional constraints are present, in order to demonstrate the applicability of the centroid-based sampling strategy to problems defined by non box-constrained feasible regions. In the second case study, presented in subsection 6.3.5, the objective is to identify the maximum reaction rate possible for simultaneous diffusion-reaction occurring in a catalyst pellet. For coupled heat and mass transfer, multiple possible reaction rates exist, and therefore it is important that application of the kriging-RSM algorithm leads to identification of all reaction rates. The physical first-principles equations are recast in a discretized form using orthogonal collocation, and the optimization problem involves a least-squares objective. For this case study, the kriging-RSM method is applied as an equation-solving algorithm.

The notations KC-R and KNC-R are used to refer to application of the kriging-RSM algorithm which does, and does not, employ the centroid-based sampling strategy for kriging model updating, respectively. More specifically, the notation KNC-R refers to the kriging algorithm in which the initial global model is built from ten dispersed sampling points, and in which subsequent models are updated based on data collected at regions of highest model variance, lowest prediction, and where the current and previous model

predictions differ the most. Three sampling points are chosen for each of the three heuristic criteria, for a total of nine sampling points to use in model updating. For each example, a table of solution information is provided based on application of each one of the KC-R and KNC-R methods. All results are obtained using an HP dv8000 CTO Notebook PC with a 1.8 GHz AMD Turion 64 processor.

6.3.1 Six-Hump Camel Back Function

The six-hump camel back function is a well-known 2-D global optimization test function that is box-constrained. Introducing black-box complications into this example, the output z_2 is a function of two continuous variables x_1 and x_2. The NLP is formulated as shown below in problem (6.18) and a deterministic plot of $z(x_1,x_2)$ is presented in Figure 6.11:

$$\begin{aligned} &min \;\; z_2 \\ &s.t. \;\; z_2 = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 - \left(-4 + 4x_2^2\right)x_2^2 \\ &\quad -3 \le x_1 \le 3 \\ &\quad -2 \le x_2 \le 2 \end{aligned} \tag{6.18}$$

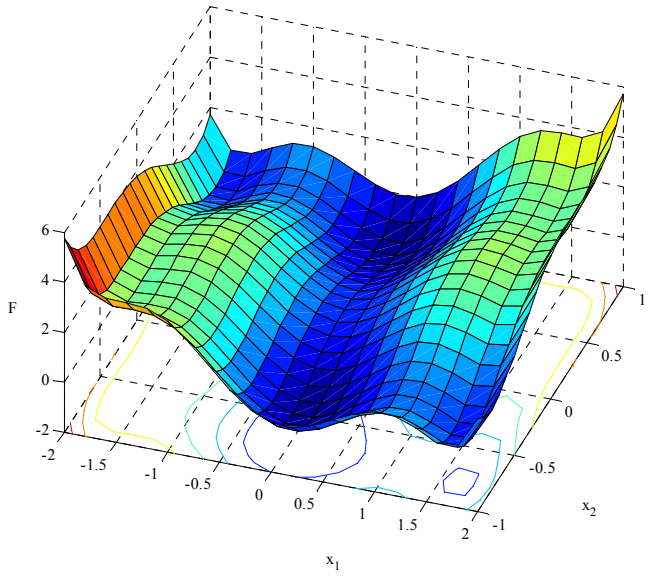


Figure 6.11. Plot of the objective function given in Problem (6.18).

The NLP given by problem (6.18) implies that only the single global optimum is to be found. However, this problem contains four local optima in addition to two global optima, and the goal of applying the centroid-based (KC-R) and non-centroid-based (KNC-R) sampling methods within the kriging-RSM optimization algorithm is not to identify simply only one global optimum, but the complete set of optima. Solution information obtained from applying the KC-R and KNC-R techniques are presented in Tables 6.2 and 6.3, respectively.

Table 6.2. Optimization results obtained for Problem (6.18) based on application of the KC-R algorithm.

# Iterations		# Function Calls			Optima Found			CPU Time		
KC	R	KC	R	Total = KC + Σ(R)	x_1	x_2	F	KC	R	Total
4	3	57	27	223	-0.0899	0.71098	-1.03161	1.609	1.047	2.656
N/A	3	0	27		0.0899	-0.71098	-1.03161	N/A		
	5	0	26		-1.7011	0.79460	-0.21538			
	5	0	26		1.7011	-0.79460	-0.21538			
	2	0	18		-1.6060	-0.56716	2.10427			
	2	0	18		1.6060	0.56716	2.10427			

Table 6.3. Optimization results obtained for Problem (6.18) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
4	13	30	59	89	2	2/100	1.211	0.352	1.563
5	15	43	89	132	3	11/100	2.003	0.911	2.913
6	20	48	121	168	4	38/100	2.090	1.097	3.187
9	21	71	135	206	5	33/100	3.158	1.317	4.474
12	24	103	153	256	6	16/100	4.885	1.450	6.335

Based on the optimization information presented in Table 6.2, all six optima are successfully identified when the centroid-based sampling strategy is used to build the global models. In contrast, when randomized/heuristic sampling was employed for global model building/refinement, only sixteen of 100 trials resulted in all optima being found as reported in Table 6.3. Moreover, the number of required function calls increased by 15% relative to the 223 required by the KC-R method. For the KNC-R method, in which a randomized sampling set was generated for initial model building in each of 100 trials,

the two global optima were successfully discovered for all cases. In addition, for 87% of the cases, at least two additional local optima were found. However, since it is not known *a priori* how many optima exist for a black-box function, the identification of the remaining local optima by the KC-R method motivates its use as a competitive modeling/optimization algorithm relative to the KNC-R technique. Figure 6.12 shows the sequence of iteratively updated global models using the KC-R algorithm; while the initial models are poor, the fourth model as presented in Figure 6.12(d) appears to be a good approximation of the deterministic complement shown in Figure 6.11.

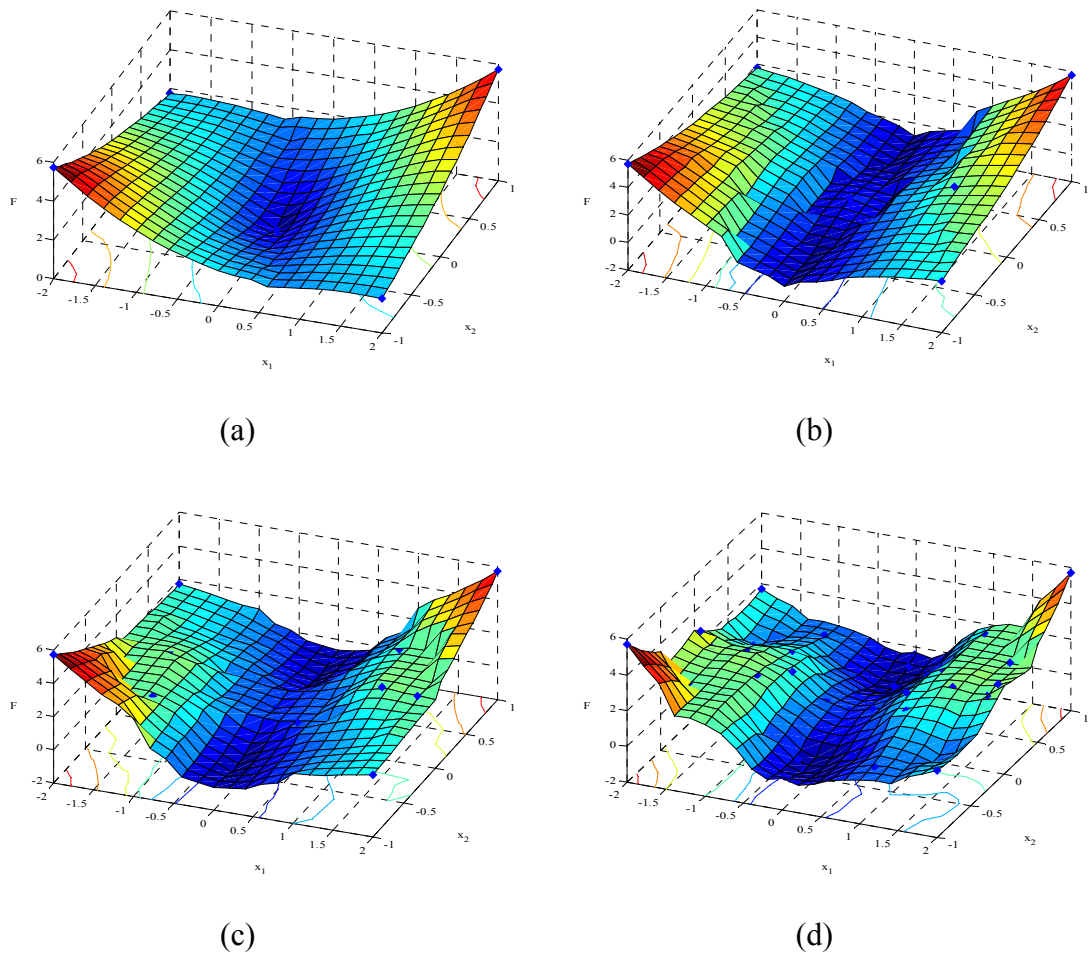


Figure 6.12. (a) – (d) Iteratively refined kriging models of the objective function given in Problem (6.18), based on application of the KC-R algorithm.

6.3.2 Branin Function

In this example the KC-R and KNC-R optimization algorithms are applied to the 2-dimensional Branin global optimization test function. The black-box function z_2 depends on both x_1 and x_2, the box-constrained problem is formulated as shown in problem (6.19):

$$\begin{aligned} &min \;\; z_2 \\ &s.t. \;\; z_2 = \left(x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6 \right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \\ &\quad -5 \le x_1 \le 10 \\ &\quad 0 \le x_2 \le 15 \end{aligned} \tag{6.19}$$

The deterministic function, shown in Figure 6.13, contains three global optima located at (-π,12.275), (π,2.275), (3π,2.475) having objective value $z_2 = 0.97887$. This problem was selected in order to confirm that all three optima would be successfully identified using the KC-R method. In Tables 6.4 and 6.5, the optimization results obtained for this example using both the KC-R and KNC-R methods are presented.

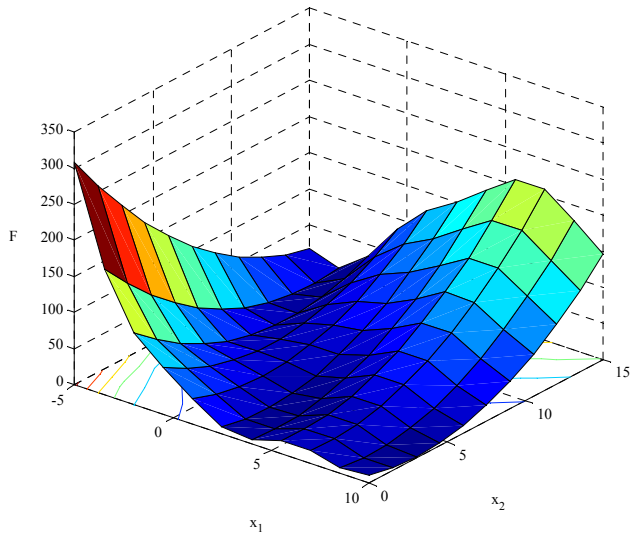


Figure 6.13. Plot of the objective function given in Problem (6.19).

Table 6.4. Optimization results obtained for Problem (6.19) based on application of the KC-R algorithm.

# Iterations		# Function Calls			Optima Found			CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)	x_1	x_2	F	KC	R	Total
4	4	57	36	203	9.4245	2.4774	0.39789	0.9688	0.7188	1.6875
N/A	5	0	44		3.1417	2.2759	0.39789	N/A		
	7	0	62		-3.1414	12.2753	0.39789			

Table 6.5. Optimization results obtained for Problem (6.19) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# global optima found (N_{Opt})	# cases in which only N_{Opt} global optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
5	24	36	174	210	2	16/100	1.081	1.749	2.830
6	24	49	187	236	3	84/100	1.179	1.552	2.731

The number of function calls required to attain all three global optima increases by 16% when the KNC-R algorithm is used, relative to the KC-R method. In comparing the performance of the KC-R and KNC-R methods when all optima were found, 57 and 49 sampling experiments were performed during the global modeling phase; the information obtained from eight additional sampling points, in addition to the sampling locations of all previous vectors, motivates the emphasis on sampling location for global modeling. The amount of sampling required for local optimization, in terms of the number of function calls, is (36+44+62) = 142 for the KC-R algorithm, a 32% reduction relative to the 187 sampling experiments required for local optimization based on global models built using the KNC-R method. The set of iteratively refined global models built using the KC-R method are shown in Figure 6.14, in which again the poor early models rapidly

improve based on the additional sampling data obtained from sampling at the third and fourth set of Delaunay triangulation centroids.

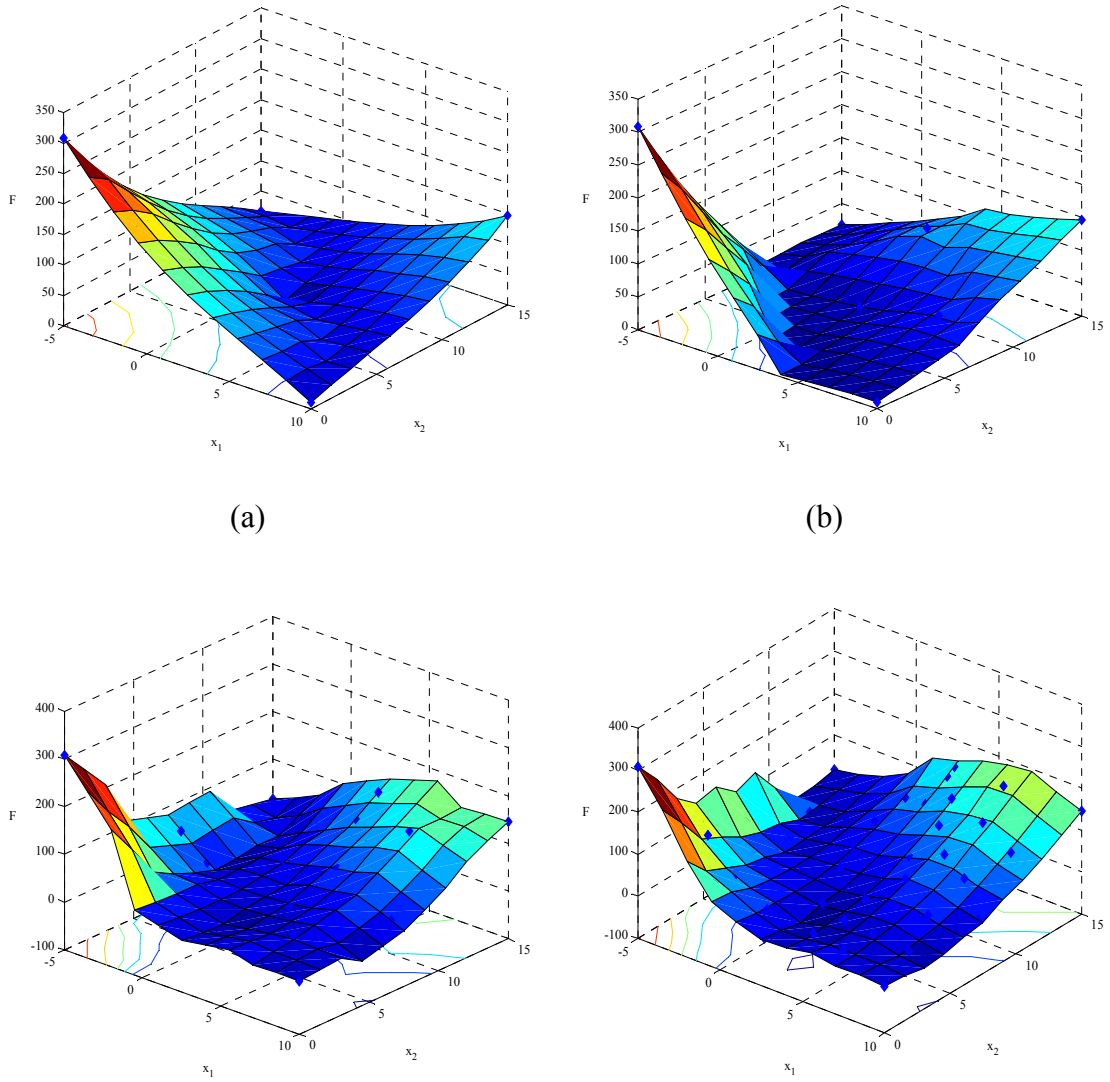


Figure 6.14. Iteratively refined kriging models of the objective function given in Problem (6.19), based on application of the KC-R algorithm.

6.3.3 Schwefel Function

The 2-dimensional Schwefel test function is a problem also taken from the global optimization literature. This box-constrained problem is modified to include both a linear and a nonlinear constraint as shown in problem (6.20). The black-box variable z_2 is a sinusoidal function of x_1 and x_2 and therefore its nonconvex deterministic behavior resembles that of a noisy function. The problem is formulated as shown below and a 3-D plot is shown in Figure 6.15. Due to the function nonconvexity induced by the additive sinusoidal terms, a contour plot is also provided in Figure 6.16.

$$
\begin{aligned}
&min \;\; z_2 \\
&s.t. \;\; z_2 = \sum_{i=1}^{2} -x_i sin\sqrt{|x_i|} \\
&\quad -2x_1 - x_2 \le 800 \\
&\quad 0.004x_1^2 - x_1 + x_2 \le 500 \\
&\quad -500 \le x_1, x_2 \le 500
\end{aligned}
\tag{6.20}
$$

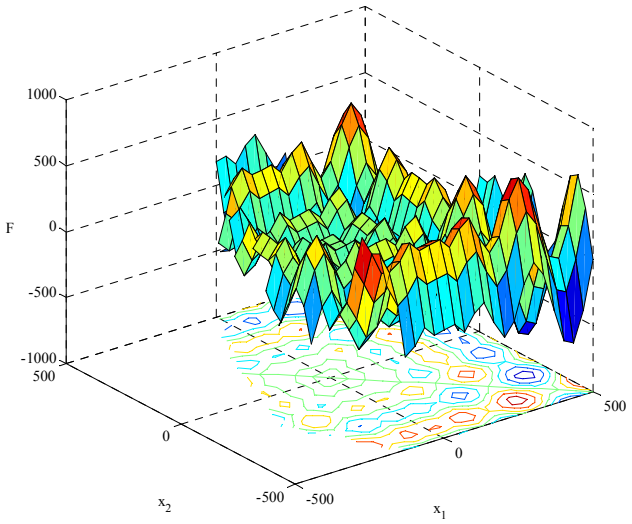


Figure 6.15. 3-D plot of the deterministic objective function given in Problem (6.20).

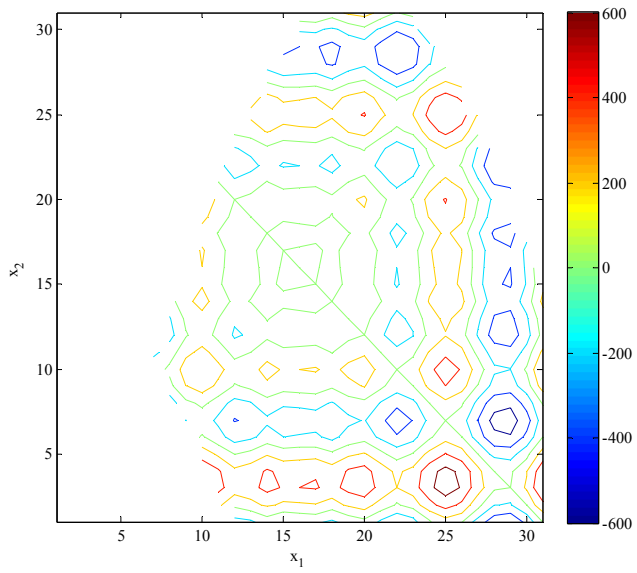


Figure 6.16. Contour plot of the deterministic objective function given in Problem (6.20).

This function contains a number of local optima and one global optimum at (420.97,-302.525) having an objective value of -719.53. The global optimum is located in a corner of the feasible region and is surrounded by a set of local optima. Due to the number of inferior local optima, the KC-R and KNC-R algorithms will be applied in order to find the set of five dispersed optima whose objective values are lower than -500. This problem is selected in order to demonstrate the application of the centroid-based sampling technique when a nonlinear constraint exists. For the initial modeling, the sampling set is obtained at locations which correspond to the convex polytope inscribed within the nonlinear feasible region. For this problem, even if a nominal noise term were additively applied to z_2, the underlying geometry would still pose the main complication affecting successful convergence to the global optimum. The optima information are presented in Table 6.6 along with additional computational results based on application

of the KC-R algorithm. Corresponding computational results based on application of the KNC-R method are presented in Table 6.7.

Table 6.6. Optimization results obtained for Problem (6.20) based on application of the KC-R algorithm.

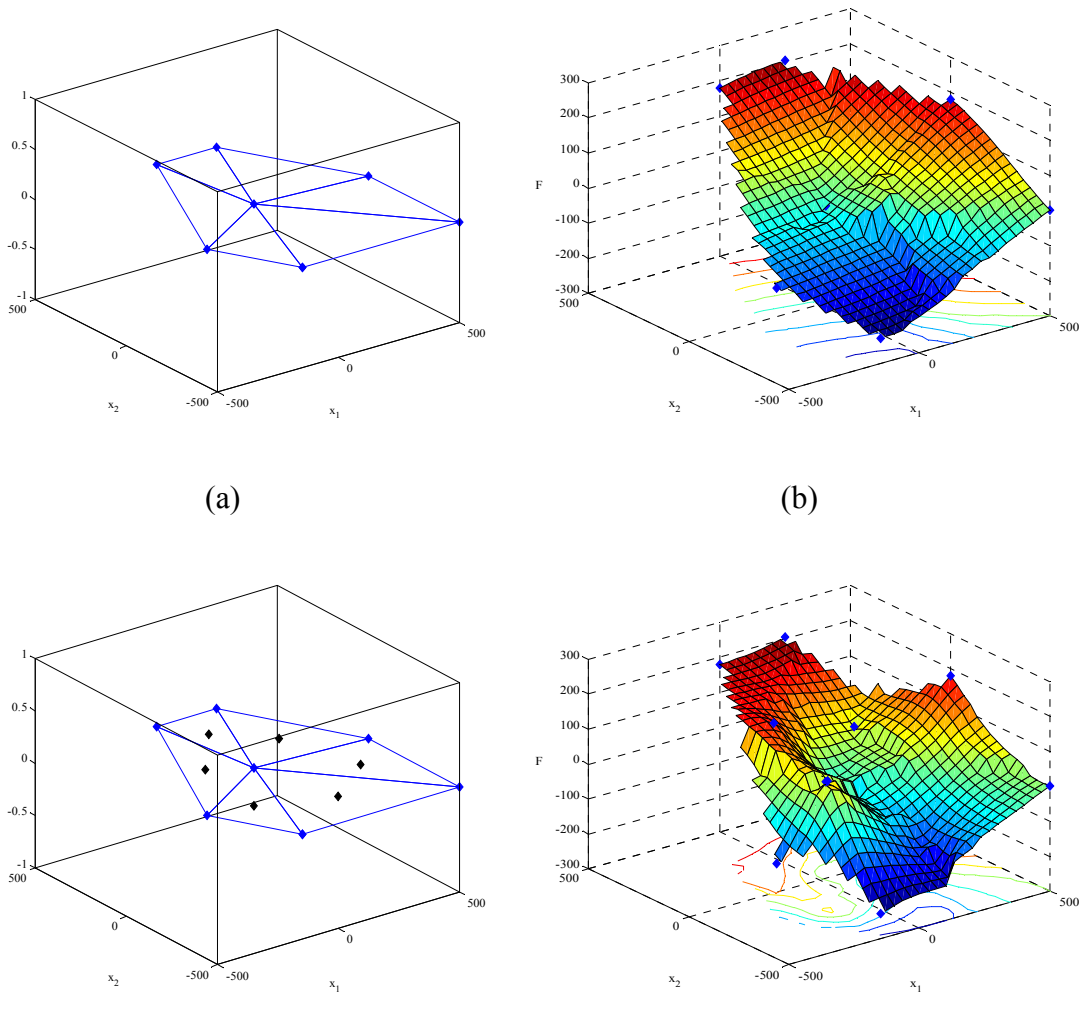
# Iterations		# Function Calls			Optima Found			CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)	x_1	x_2	F	KC	R	Total
4	2	73	31	230	420.775	-302.028	-719.491	3.391	3.766	7.156
N/A	1	0	30		203.783	420.958	-620.826	N/A		
	11	0	20		416.871	202.563	-618.513			
	3	0	40		420.734	-123.123	-541.478			
	7	0	36		202.105	-301.280	-501.820			

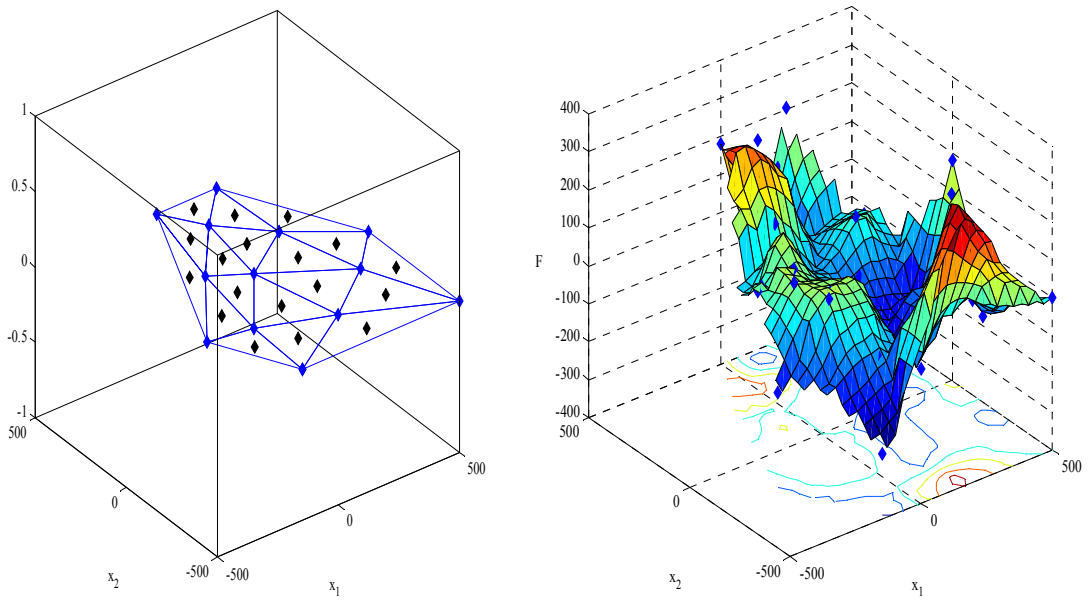
Table 6.7. Optimization results obtained for Problem (6.20) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
32	29	283	186	469	5	86/100	26.337	4.638	30.974

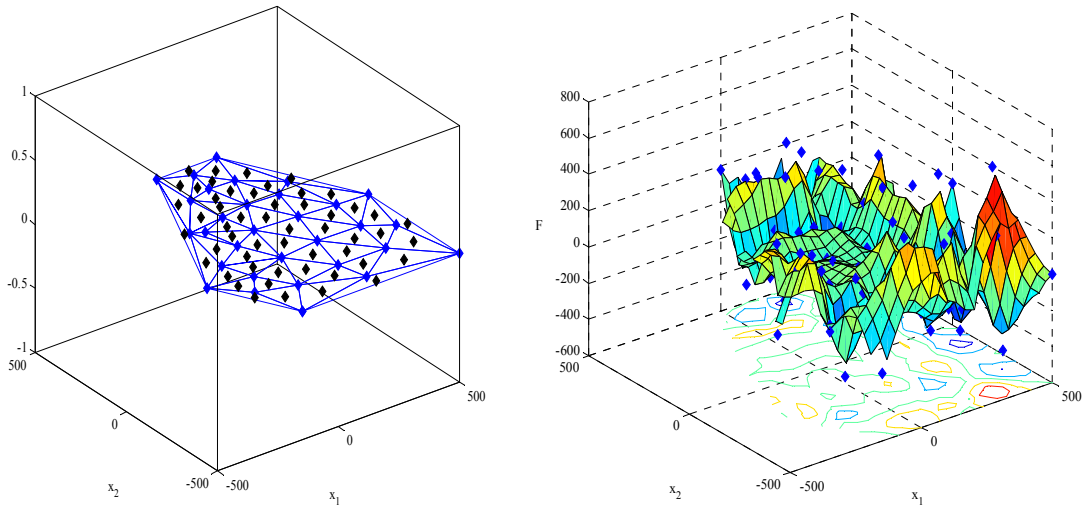
When the KNC-R method is applied, the set of five optima are found at an 86% success rate and require just over twice the 230 required by the KC-R procedure. Due to the sinusoidal behavior of the function, no function "settling" is observed as the model is refined. The objective value varies over a high range, between -/+ 800, and there exist many shallow/deep local optima distributed over both small and large subregions. Since the objective function value can vary over a wide range, between -/+800, when additional sampling is performed at locations where the model predictions are smallest, the model

variance is highest, or where the model has changed most between iterations, the average model value for the kriging predictor can change significantly between iterations. For the KNC-R algorithm, thirty-two iteratively refined global models are required before local optimization is initiated, indicating that the termination criterion of requiring model convergence may be effective for successful identification of warm-start iterates for RSM refinement, but ineffective with respect to minimizing the amount of sampling needed for accurate global modeling. A set of plots showing the sampling locations and iteratively improved global models built using the KC-R method is presented in Figure 6.17.





(e) (f)



(g) (h)

Figure 6.17. Iteratively refined kriging models (b), (d), (f), and (h), of the objective function given in Problem (6.20), based on application of the KC-R algorithm, with accompanying sampling plots containing previously sampled points (vertices) and the new sampling set (interior points) (a), (c), (e), (g).

6.3.4 Example 4: Kinetics Case Study

For this example, the KC-R and KNC-R methods are applied to a realistic kinetics case study taken from Bindal et al.[18]. There are five reaction species, two of which are system inputs. The reactions are assumed to occur in an ideal CSTR and the reaction network considered is a modification of the Fuguitt and Hawkins mechanism[17]given by: $A \rightarrow E$, $A \rightarrow D$, $B \leftrightarrow D$, $C \leftrightarrow 2D$, and $2A \rightarrow C$. Only A and C enter the reactor having concentrations C_A^0, C_C^0 given in units of mol/m^3. The dynamic behavior of the system is described by problem (6.21).

$$
\begin{aligned}
min \quad & F = 4\left(X-0.6\right)^2 + 4\left(Y-0.4\right)^2 + sin^3\left(\pi X\right) + 0.4 \\
& X = 0.1428 C_C^{SS} - 0.357 C_D^{SS} \\
& Y = -0.1428 C_C^{SS} + 2.857 C_D^{SS} + 1.0 \\
& \frac{dC_A}{dt} = \frac{F_R}{V}\left(C_A^0 - C_A\right) - k_1^f C_A - k_2^f C_A - k_5^f C_A^2 \\
& \frac{dC_B}{dt} = \frac{F_R}{V}\left(-C_B\right) - k_3^f C_B + k_3^r C_D \\
& \frac{dC_C}{dt} = \frac{F_R}{V}\left(C_C^0 - C_C\right) - k_4^f C_C + 0.5 k_4^r C_D^2 + 0.5 k_5^f C_A^2 \\
& \frac{dC_D}{dt} = \frac{F_R}{V}\left(-C_D\right) + k_2^f C_A + k_3^f C_B - k_3^r C_D + 2 k_4^r C_C - k_4^r C_D^2 \\
& \frac{dC_E}{dt} = \frac{F_R}{V}\left(-C_E\right) + k_1^f C_A \\
& \left[k_1^f, k_2^f, k_3^f, k_3^r\right] = \left[3.33384, 0.26687, 0.29425, 0.14940\right] \\
& \left[k_4^f, k_4^r, k_5^f, F_R, V\right] = \left[0.011932, 1.8957\mathrm{e}^{-3}, 9.9598\mathrm{e}^{-6}, 0.008, 0.1\right] \\
& 3\mathrm{e}^4 \le C_A^0 \le 3\mathrm{e}^4 \\
& 0 \le C_C^0 \le 10\,\mathrm{e}^4
\end{aligned}
\tag{6.21}
$$

The variables C_C^{SS} and C_D^{SS} represent the macroscale steady-state values of C_C and C_D, respectively. The rate constants, input flow rate, and reactor volume are similarly given by k, F_R, and V, respectively. A plot of the objective F as a function of the input variables

C_A^0 and C_C^0 is shown in Figure 6.18. In the following two subsections, this problem is solved as a box-constrained and then non-box constrained NLP.

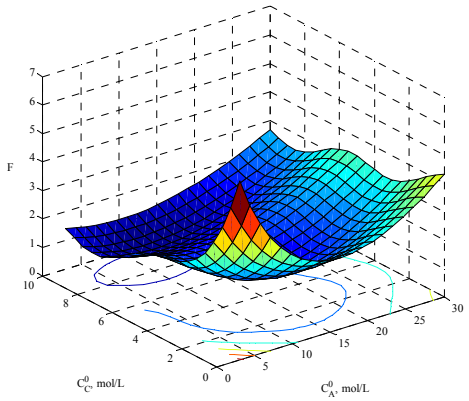


Figure 6.18. Plot of the objective function given in Problem (6.21).

6.3.4.1 Box-Constrained Problem With Noise

The deterministic solution of problem (6.21) yields a global optimum of F = 0.7422 at $[C_A^0, C_C^0]$ = [10.117, 8.378] and a local optimum of F = 1.2364 at [13.202,3.163]. In order to introduce black-box complications, the rate equations are assumed to be unknown, so a microscopic model is used instead, represented using a lattice containing N molecules. The microscopic model is generated by first translating concentrations to molecular variables, evolving the microscopic system using the Gillespie algorithm[25], and then mapping back the final measured variables to concentrations. The noise in the output concentrations thereby arises as a function of how coarse the microscopic model is. Steady-state solution vectors are obtained from an initial point by running the microscale simulations for a long time horizon, after which the objective function can be evaluated. The optimization problem is formulated as shown in problem (6.22), where the steady-state concentrations C_C^{SS} and C_D^{SS} are treated as the black-box system output variables:

$$
\begin{aligned}
min \quad & F = 4\left(X-0.6\right)^2 + 4\left(Y-0.4\right)^2 + sin^3\left(\pi X\right) + 0.4 \\
& X = 0.1428C_C^{SS} - 0.357C_D^{SS} \\
& Y = -0.1428C_C^{SS} + 2.857C_D^{SS} + 1.0 \\
& \left[z_{2,1}, z_{2,2}\right] = \left[C_C^{SS}, C_D^{SS},\right] = \Gamma\begin{pmatrix} C_A^0, C_C^0, k_1^f, k_2^f, k_3^f, k_3^r, \\ k_4^f, k_4^r, k_5^f, F_R, V, N \end{pmatrix} \\
& 3\mathrm{e}^4 \le C_A^0 \le 3\mathrm{e}^4 \\
& 0 \le C_C^0 \le 10\,\mathrm{e}^4
\end{aligned}
\tag{6.22}
$$

The variance of the objective can be evaluated over *k* replicate simulations at a nominal species concentration vector $x = [C_A{}^0, C_C{}^0]$ as:

$$
\sigma^2 = Var(\varepsilon) = Var\left\{F_i(x, N) \mid i = 1...j\right\} \tag{6.23}
$$

For this example, the noise applied to the objective is in the form of a normally distributed error having standard deviations of $\sigma = 0$ and $\sigma = 0.011$, which corresponds to microscale system sizes of N = 1e6 and N = 100,000, respectively. Therefore, the problem given by (6.21) can be restated as an optimization problem which contains a noisy objective function as given by problem (6.24):

$$
\begin{aligned}
min \quad & F = 4\left(X-0.6\right)^2 + 4\left(Y-0.4\right)^2 + sin^3\left(\pi X\right) + 0.4 + N\left(0, \sigma^2\right) \\
& X = 0.1428C_C^{SS} - 0.357C_D^{SS} \\
& Y = -0.1428C_C^{SS} + 2.857C_D^{SS} + 1.0 \\
& \left[z_{2,1}, z_{2,2}\right] = \left[C_C^{SS}, C_D^{SS},\right] = \Gamma\begin{pmatrix} C_A^0, C_C^0, k_1^f, k_2^f, k_3^f, k_3^r, \\ k_4^f, k_4^r, k_5^f, F_R, V \end{pmatrix} \\
& 3\mathrm{e}^4 \le C_A^0 \le 3\mathrm{e}^4 \\
& 0 \le C_C^0 \le 10\,\mathrm{e}^4
\end{aligned}
\tag{6.24}
$$

For the local optimization using RSM, the value of Tol_{RSM} is 0.01, meaning that the difference between the current and previous optimum objective values, as given by $|F_{opt,w+1} - F_{opt,w}|$, must fall below 0.01 before the RSM algorithm is terminated. In order to improve global optimum accuracy, once this termination criteria is met, another response

surface is built and optimized using a design radius b_w half that of the previous one, in an effort to further zoom in on the best solution. In Tables 6.8 – 6.11, computational results are presented based on the application of the KC-R and KNC-R kriging algorithms. Tables 6.8 and 6.9 present results when the problem given by (6.24) is solved deterministically; based on this condition, value of σ is zero. Tables 6.10 and 6.11 present complementary results when problem (6.24) is solved under the condition that σ is 0.011. For all tables, it should be noted that the reported CPU times refer to the computational expense required for strictly modeling and optimization, and excludes the time required for each function call in the form of a microscopic model simulation.

Table 6.8. Optimization results obtained for Problem (6.24) under the condition that σ = 0, based on application of the KC-R algorithm.

# Iterations		# Function Calls			Optima Found			CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)	x_1	x_2	F	KC	R	Total
4	3	57	26	101	10.1178	8.3787	0.74221	1.5313	0.2188	1.75
N/A	2	0	18		13.2025	3.1756	1.23644	N/A		

Table 6.9. Optimization results obtained for Problem (6.24) under the condition that σ = 0, based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
5	3	42	21	63	1 (global only)	47/100	1.899	0.168	2.067
6	7	44	57	101	2 (global + local)	53/100	1.921	0.495	2.416

When problem (6.24) is solved deterministically, the number of function calls required for attainment of both optima is identical when either one of the KC-R or KNC-R methods is used. However, even though the global optimum is found in all cases when the KNC-R method is used, the chances that both optima are identified when the KNC-R method is employed is only 53%.

Table 6.10. Optimization results obtained for Problem (6.24) under the condition that $\sigma = 0.011$, based on application of the KC-R algorithm.

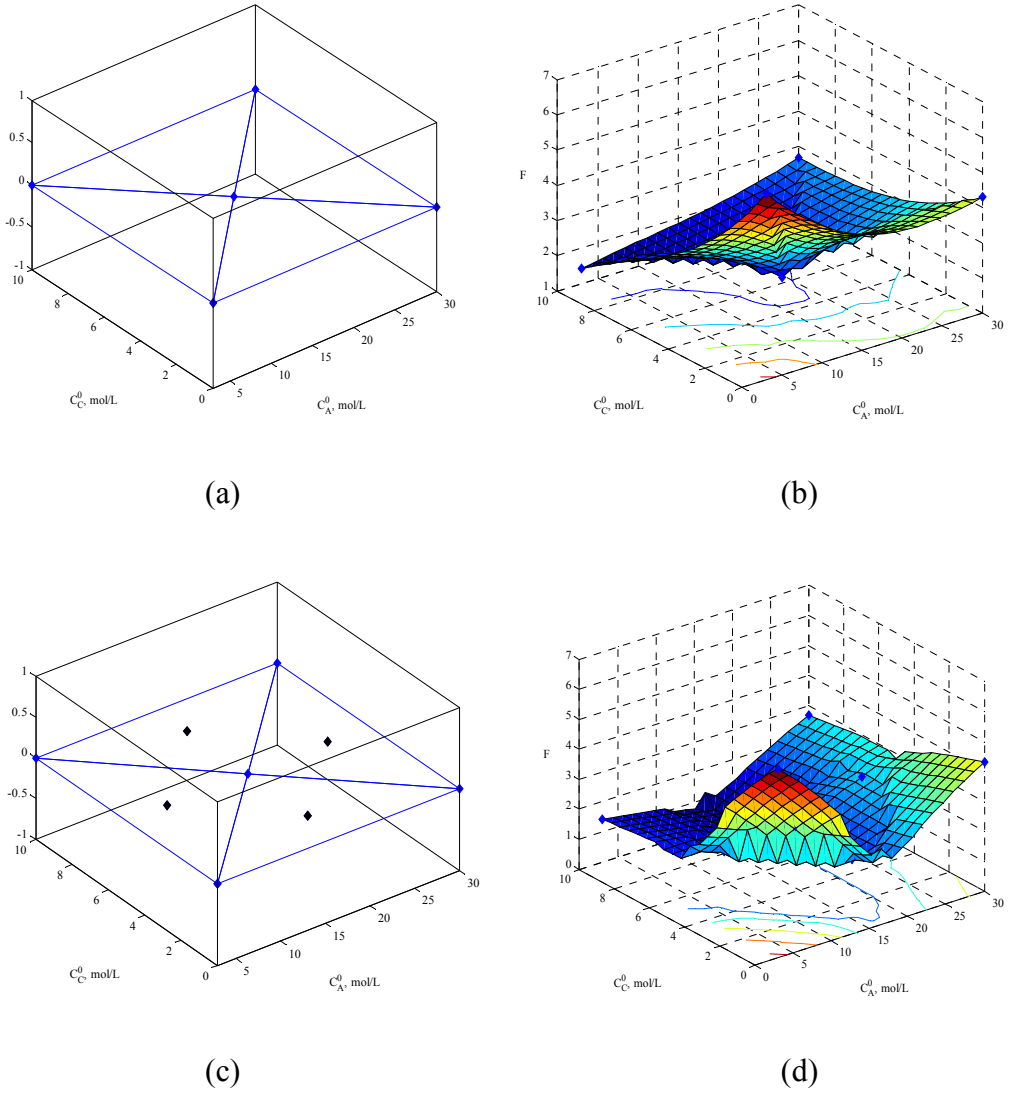
# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)			KC	R	Total
4	7	57	59	116	1 (local only)	17/100	1.783	0.563	2.347
4	9	57	75	132	1 (global only)	25/100	1.901	0.858	2.758
4	10	57	79	136	2 (global + local)	39/100	1.869	0.765	2.634

Table 6.11. Optimization results obtained for Problem (6.24) under the condition that $\sigma = 0.011$, based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
6	9	48	65	113	1 (local only)	9/100	2.450	0.721	3.170
6	5	45	39	84	1 (global only)	48/100	2.169	0.394	2.563
5	11	41	78	119	2 (global + local)	22/100	1.874	0.818	2.691

When noise is applied to the problem given by Equation (6.24), the chances of finding at least one of the optima are (17 + 25 + 39) = 81% or (9 + 48 + 22) = 79%, for the

KC-R and KNC-R methods, respectively. The global optimum is found in (25+39) = 64 cases when the KC-R algorithm is applied, and in (48+22) = 70 cases when the KNC-R method is used instead. However, both optima are identified in 39 cases when the KC-R method is used, nearly double the number of cases (22) when the KNC-R method is employed. In Figure 6.19, the sequence of iteratively refined global models built using the centroid-based sampling strategy is shown, along with the corresponding sampling templates.



(a) (b)

(c) (d)

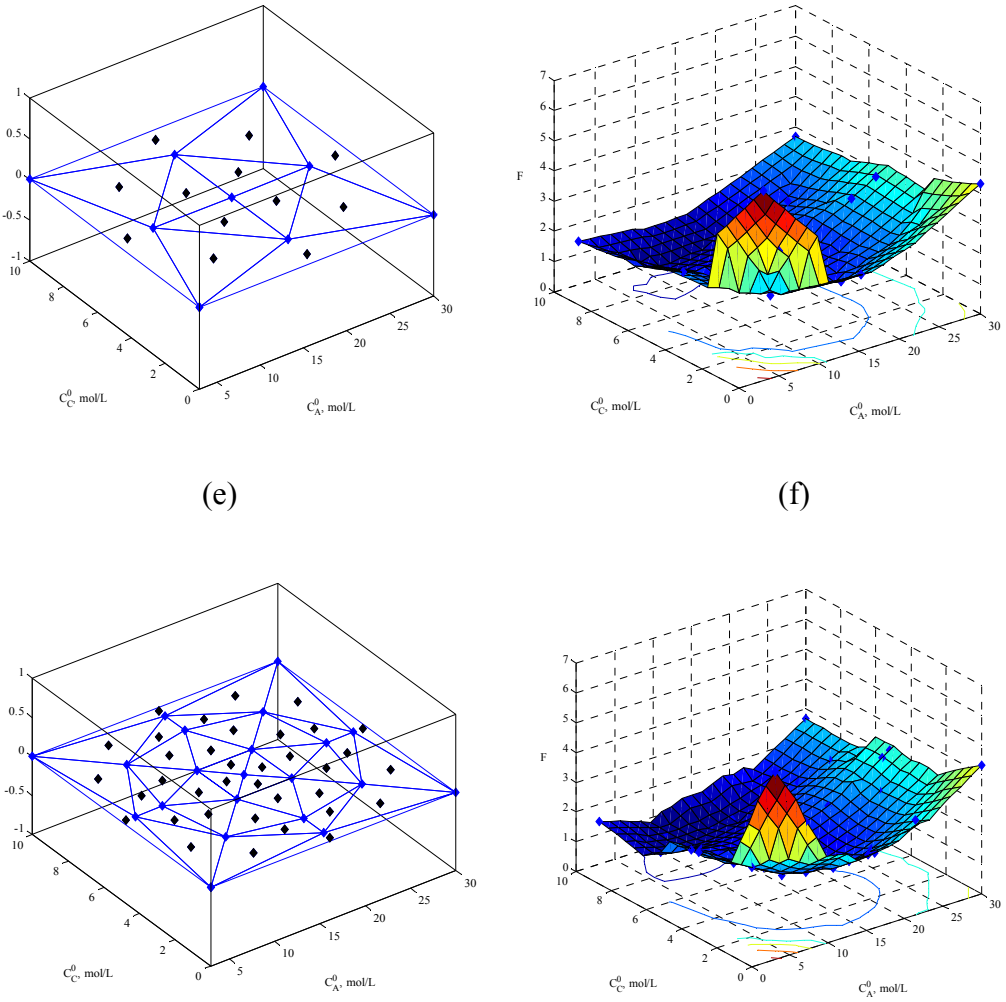


Figure 6.19. Iteratively refined kriging models (b), (d), (f), (h) of the objective function given in Problem (6.21), based on application of the KC-R algorithm, with accompanying sampling set plots (a), (c), (e), (g).

6.3.4.2 Non-Box-Constrained Problem

In order to demonstrate the applicability of centroid-based sampling to a problem whose feasible region is non-box constrained, problem (6.22) is modified to include three

additional linear constraints. The new problem is given by problem (6.25) and is solved under deterministic conditions. A plot of the function is shown in Figure 6.20.

$$
\begin{aligned}
min \quad & F = 4\left(X-0.6\right)^2 + 4\left(Y-0.4\right)^2 + sin^3\left(\pi X\right) + 0.4 \\
& X = 0.1428C_C^{SS} - 0.357C_D^{SS} \\
& Y = -0.1428C_C^{SS} + 2.857C_D^{SS} + 1.0 \\
& z_2 = \left[C_C^{SS}, C_D^{SS}\right] = \Gamma\begin{pmatrix} C_A^0, C_C^0, k_1^f, k_2^f, k_3^f, k_3^r, \\ k_4^f, k_4^r, k_5^f, F_R, V, N \end{pmatrix} \\
& C_A^0 - 11.5681C_C^0 \le -69.4928 \\
& C_A^0 + 1.6615C_C^0 \le 13.8 \\
& C_A^0 + 1.8C_C^0 \le 33.6 \\
& 3\text{e}^4 \le C_A^0 \le 3\text{e}^4 \\
& 0 \le C_C^0 \le 10\,\text{e}^4
\end{aligned}
\qquad (6.25)
$$

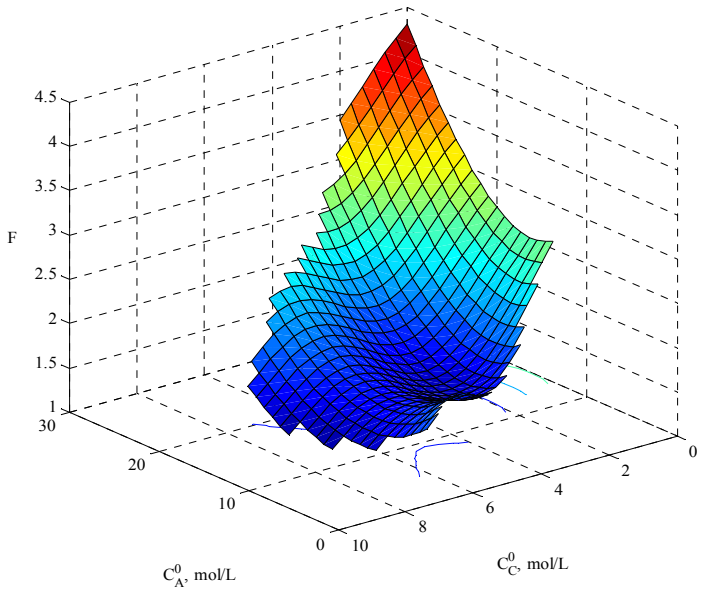
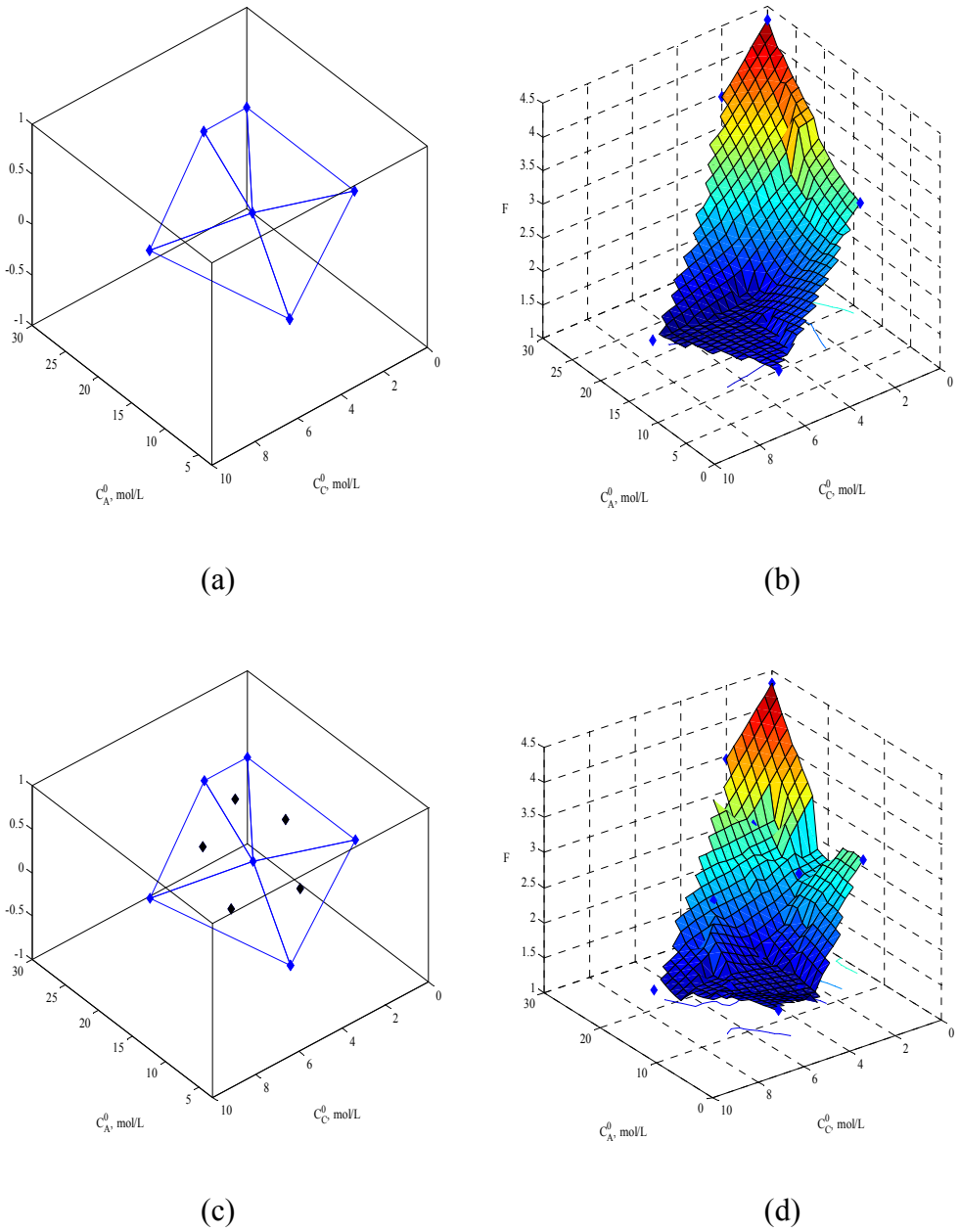
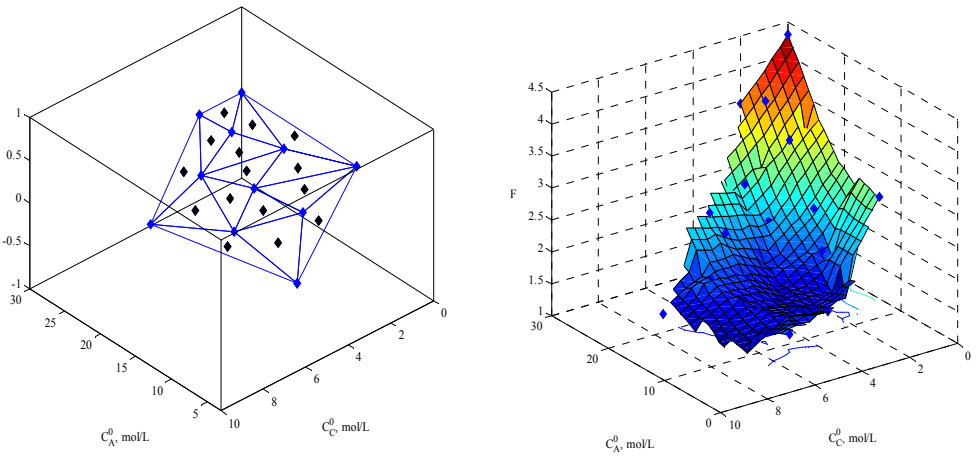


Figure 6.20. Plot of the objective function given in Problem (6.25).

The presence of the additional constraints alters the shape of the original box-shaped region to that of a pentagon. For this problem, the nominal sampling set is now

comprised of, 1) the five vertices of the reduced feasible region, rather than the four used for the box-constrained region, and 2), the centroid location given with respect to these five vertices. The set of sampling templates and global models corresponding to the application of centroid-based sampling for global modeling within the KC-R algorithm are shown in Figure 6.21.





(e) (f)

Figure 6.21. Iteratively refined kriging models (b), (d), (f) of the objective function given in Problem (6.25), based on application of the KC-R algorithm, with accompanying sampling set plots (a), (c), (e).

The solution set of the optimization problem given in Equation (6.25) consists of, 1) a constrained global optimum at $[C_A^0, C_C^0]$ = [12.6140, 7.0977] whose corresponding objective value is 1.0424, and 2) a local optimum at $[C_A^0, C_C^0]$ = [13.20247,3.1756], whose corresponding objective value is 1.23644. Tables 6.12 and 6.13 present computational results based on application of the KC-R and KNC-R algorithms to solve this problem. For this problem, the KC-R algorithm is terminated after two iterations, or after the global model has been updated only once. Due to the smaller feasible region size, early termination is applied to avoid obtaining sampling information at smaller Delaunay triangle centroids that might fail to provide any new function behavior not already captured by the previous sampling set.

Table 6.12. Optimization results obtained for Problem (6.25) based on application of the KC-R algorithm.

# Iterations		# Function Calls			Optima Found			CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)	C_A^0	C_C^0	F	KC	R	Total
2	3	11	24	76	12.7946	7.1133	1.0424	0.7500	0.3438	1.0938
N/A	3	0	38		13.203	3.1944	1.2366	N/A		

Table 6.13. Optimization results obtained for Problem (6.25) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
4	8	31	46	77	1 (global only)	36/100	1.210	0.369	1.579
4	11	32	65	97	2 (global + local)	64/100	1.148	0.533	1.681

Both the local and global optima are found using the KC-R method in approximately the same number of average function calls required for only the global optimum to be identified by the KNC-R algorithm. For the KNC-R method, the number of sampling experiments required to obtain the local optimum increases by 27% relative to the 76 required by the KC-R algorithm. Furthermore, both the local and global optima are identified with only a 64% success rate when the KNC-R technique is employed.

6.3.5 Simultaneous Diffusion and Reaction in a Catalyst Pellet

In this second industrial case study, taken from Lucia et. al[60], the kriging-RSM algorithms are applied as equation-solving methods in order to identify the best modeled

reaction rate of a chemical species undergoing simultaneous diffusion and reaction through a spherical catalyst pellet as shown in Figure 6.22. The physical behavior of this coupled heat-mass transfer problem can be described by a PDE having specified boundary conditions. NLP solvers may fail to yield closed-form solutions of this differential equation system, but may succeed at alternatively identifying the solutions to a discretized complement of the original problem. The physical system models and discretization methods described below are the same as given by Lucia et. al[58], and the K_C and K_{NC} algorithms are applied as equation solvers once the optimization problem has been reformulated in terms of a minimum set of nonlinear equations.

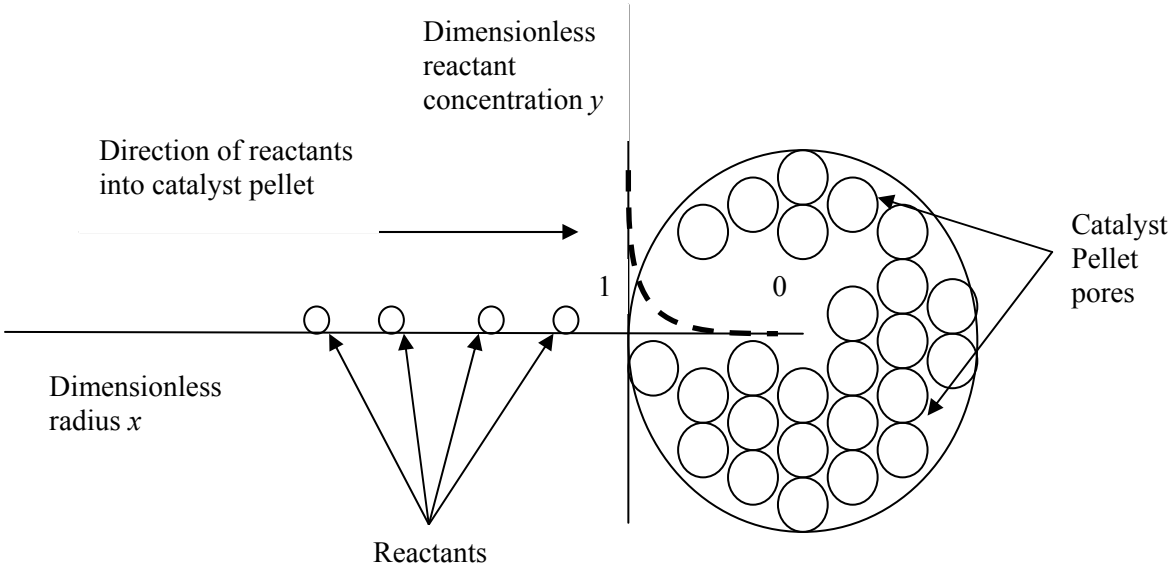


Figure 6.22. Dimensionless concentration profile of reactants undergoing simultaneous diffusion and reaction as they move through the pores of a spherical catalyst pellet.

The dimensionless non-isothermal species and heat transport model is given by Equation (6.26).

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} - \phi^2 y \exp\left[\frac{\gamma\beta(1-y)}{1+\beta(1-y)}\right] = 0 \qquad (6.26)$$

The boundary conditions specify the following behavior: 1) at the pellet surface, the dimensionless species concentration is unity as shown in Equation (6.27), and 2) at the pellet center, the species concentration change is zero, as given in Equation (6.28). The reaction rate η is given by Equation (6.29).

$$y\big|_{x=1} = 1 \tag{6.27}$$

$$\left.\frac{dy}{dx}\right|_{x=0} = 0 \tag{6.28}$$

$$\eta = \frac{3}{\phi^2}\left.\frac{dy}{dx}\right|_{x=1} \tag{6.29}$$

The complete optimization problem is formulated in problem (6.30):

$$\begin{aligned} &\min \ F \\ &s.t. \quad F = \frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} - \phi^2 y \exp\left[\frac{\gamma\beta(1-y)}{1+\beta(1-y)}\right] \\ &\qquad y\big|_{x=1} = 1 \\ &\qquad \left.\frac{dy}{dx}\right|_{x=0} = 0 \\ &\qquad 0 \le y \le 1 \\ &\qquad [\beta, \phi, \gamma] = [0.6, 0.2, 30] \end{aligned} \tag{6.30}$$

In order to identify the set of possible reaction rates η, orthogonal collocation over finite elements (OCFE) is used to discretize the problem given by Equations (6.26) – (6.28). From the set of discretized equations, variable elimination can be performed, reducing the problem to a set of nonlinear equations given in terms of dimensionless concentrations $y(x_i)$ for i equispaced intervals between [0,1]. Once a vector $y(x_i)$ has been obtained which satisfies the nonlinear equations, the appropriate elements of this vector are substituted into the OCFE derivative expression at $x = 1$, thereby yielding a value for η. This process can be repeated for all solution vectors y, until all values of η have been determined.

The dimensionless radius x, which varies between zero at the pellet center and unity at the surface, is subdivided into N_E equidistant elements, each one containing N_N nodes. For each element, a unique approximation is generated for the dimensionless species concentration y as a function of radial distance x from the pellet center. C^0 continuity is enforced at adjacent finite element junctions whereby the piecewise approximations corresponding to two adjacent finite elements must agree at the common node location x. In addition, C^1 continuity is similarly enforced by requiring that the piecewise approximation gradients also agree. The orthogonal collocation equations are given as follows:

$$y^{[i]} = \sum_{j=1}^{N_N} L_{(i-1)(N_E-1)+j} y_{(i-1)(N_E-1)+j} \qquad i = 1...N_E \qquad (6.31)$$

$$L_{(i-1)(N_E-1)+j} = \prod_{\substack{\{j,k\}=N_N(i-1)+1 \\ j \neq k}}^{N_N i} \frac{x - x_k}{x_j - x_k} \qquad i = 1...N_E \qquad (6.32)$$

$$\frac{d^2 y^{[i]}}{dx^2} + \frac{2}{x}\frac{dy^{[i]}}{dx} - \phi^2 y \exp\left(\left. \frac{\gamma\beta\left(1 - y^{[i]}\right)}{1 + \beta\left(1 - y^{[i]}\right)} \right|_{x=(i-1)/N_E + j/\left[(N_N-1)N_E\right]} \right) = 0 \qquad (6.33)$$

$$i = 1...N_E - 1, \; j = 1...N_N - 2$$

$$\left. y^{[N_E]} \right|_{x=1} = 1 \qquad (6.34)$$

$$\left. \frac{dy^{[1]}}{dx} \right|_{x=0} = 0 \qquad (6.35)$$

$$\left. y^{[i]} \right|_{x=i/N_E} = \left. y^{[i+1]} \right|_{x=i/N_E} \qquad i = 1...N_E - 1 \qquad (6.36)$$

$$\left. \frac{dy^{[i]}}{dx} \right|_{x=i/N_E} = \left. \frac{dy^{[i+1]}}{dx} \right|_{x=i/N_E} \qquad i = 1...N_E - 1 \qquad (6.37)$$

Equation (6.31) provides the piecewise approximation for y over a given element i, defined as $y^{[i]}$ and valid over the interval $[(i\text{-}1)/N_E, i/N_E]$, $i = 1 \ldots N_E$. For each approximation, $y^{[i]}$ is expressed as the sum of N_N nodal approximations $y_{(i-1)(NE-1)+j}$, each weighted by a corresponding Lagrange polynomial $L_{(i-1)(NE-1)+j}$, for i=1…N_E and $j = 1 \ldots N_N$, as given in Equation (6.32). Equation (6.33) is the diffusion-reaction transport equation that must be satisfied at all interior nodes for each of the i finite element approximations $y^{[i]}$. Equations (6.34) and (6.35) correspond to the boundary conditions specified by (6.27) and (6.28). Equations (6.36) and (6.37) are the C^0 and C^1 continuity equations denoting the conditions that the i^{th} piecewise approximation $y^{[i]}$ and its gradient $dy^{[i]}/dx$ must match the corresponding $i^{th}+1$ approximations $y^{[i+1]}$ and $dy^{[i+1]}/dx$, at locations x which correspond to interior nodes, respectively. In the following subsections, two example discretization cases will be considered: 1) two elements and three nodes, and 2) five elements and three nodes.

6.3.5.1 Discretization based on two elements and three nodes

For a discretization containing two elements and three nodes, the equations are provided as follows for $N_E = 2$ and $N_N = 3$:

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]_{0 \le x \le 0.5} = \left[y^{[1]}, \frac{dy^{[1]}}{dx}, \frac{d^2y^{[1]}}{dx^2} \right]_{0 \le x \le 0.5} \tag{6.38}$$

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]_{0.5 \le x \le 1} = \left[y^{[2]}, \frac{dy^{[2]}}{dx}, \frac{d^2y^{[2]}}{dx^2} \right]_{0.5 \le x \le 1} \tag{6.39}$$

$$\left[\frac{d^2y^{[1]}}{dx^2} + \frac{2}{x}\frac{dy^{[1]}}{dx} - \phi^2 y^{[1]} \exp\left(\frac{\gamma\beta\left(1 - y^{[1]}\right)}{1 + \beta\left(1 - y^{[1]}\right)} \right) \right]_{x=0.25} = 0 \tag{6.40}$$

$$\left[\frac{d^2y^{[2]}}{dx^2}+\frac{2}{x}\frac{dy^{[2]}}{dx}-\phi^2 y^{[2]}\exp\left(\frac{\gamma\beta\left(1-y^{[2]}\right)}{1+\beta\left(1-y^{[2]}\right)}\right)\right]\Bigg|_{x=0.75}=0 \tag{6.41}$$

$$y^{[2]}\Big|_{x=1}=1 \tag{6.42}$$

$$\left.\frac{dy^{[1]}}{dx}\right|_{x=0}=0 \tag{6.43}$$

$$y^{[1]}\Big|_{x=0.5}=y^{[2]}\Big|_{x=0.5} \tag{6.44}$$

$$\left.\frac{dy^{[1]}}{dx}\right|_{x=0.5}=\left.\frac{dy^{[2]}}{dx}\right|_{x=0.5} \tag{6.45}$$

where Equations (6.38) and (6.39) designate the OCFE polynomial expressions valid for the general variables y, dy/dx, and d^2y/dx^2 given in (6.26) – (6.29). For example, over the interval $0 \le x \le 0.5$, the polynomial given by $y^{[1]}$, shown below in Equation (6.53), is a model describing the decrease in dimensionless species concentration y; the corresponding expression given by $y^{[2]}$ models the behavior of y over the corresponding interval $0.5 \le x \le 1$. Equations (6.40) and (6.41) describe the coupled heat and mass transport described by Equation (6.33). The OCFE expressions corresponding to the boundary, C^0 continuity, and C^1 continuity conditions given by (6.34) – (6.37) are represented by Equations (6.42) – (6.45). The Lagrange polynomials are obtained using Equation (6.32) and are shown below in Equations (6.46) – (6.52):

$$[x_1, x_2, x_3] = [0, 0.25, 0.5] \tag{6.46}$$

$$[x_4, x_5, x_6] = [0.5, 0.75, 1] \tag{6.47}$$

$$L_1 = \frac{(x-x_2)}{(x_1-x_2)}\frac{(x-x_3)}{(x_1-x_3)} = 8x^2-6x+1 \tag{6.48}$$

$$L_2 = \frac{(x-x_1)}{(x_2-x_1)}\frac{(x-x_3)}{(x_2-x_3)} = -16x^2 + 8x \tag{6.49}$$

$$L_3 = \frac{(x-x_1)}{(x_3-x_1)}\frac{(x-x_2)}{(x_3-x_2)} = 8x^2 - 2x \tag{6.50}$$

$$L_4 = \frac{(x-x_5)}{(x_4-x_5)}\frac{(x-x_6)}{(x_4-x_6)} = 8x^2 - 14x + 6 \tag{6.51}$$

$$L_5 = \frac{(x-x_4)}{(x_5-x_4)}\frac{(x-x_6)}{(x_5-x_6)} = -16x^2 + 24x - 8 \tag{6.52}$$

$$L_6 = \frac{(x-x_4)}{(x_6-x_4)}\frac{(x-x_5)}{(x_6-x_5)} = 8x^2 - 10x + 3 \tag{6.53}$$

The discretized concentration equations for y are obtained from substitution of the equations provided in (6.48) – (6.53) into Equation (6.31), and are shown in Equations (6.54) and (6.55). The corresponding gradient expressions dy/dx and d^2y/dx^2 are then easily generated from differentiation of Equations (6.54) and (6.55).

$$y^{[1]} = L_1 y_1 + L_2 y_2 + L_3 y_3 \tag{6.54}$$

$$y^{[2]} = L_4 y_4 + L_5 y_5 + L_6 y_6 \tag{6.55}$$

$$\frac{dy^{[1]}}{dx} = (16x-6)y_1 + (-32x+8)y_2 + (16x-2)y_3 \tag{6.56}$$

$$\frac{dy^{[2]}}{dx} = (16x-14)y_4 + (-32x+24)y_5 + (16x-10)y_6 \tag{6.57}$$

$$\frac{d^2y^{[1]}}{dx^2} = 16y_1 - 32y_2 + 16y_3 \tag{6.58}$$

$$\frac{d^2y^{[2]}}{dx^2} = 16y_4 - 32y_5 + 16y_6 \tag{6.59}$$

The equations given by (6.54) - (6.59) are then substituted into Equations (6.33) – (6.37) in order to obtain the set of OCFE equations corresponding to a two-element, three-node discretization of the problem given by Equations (6.26) – (6.28):

$$-32y_2 + 32y_3 - \phi^2 y_2 \exp\left(\frac{\gamma\beta(1-y_2)}{1+\beta(1-y_2)}\right) = 0 \tag{6.60}$$

$$10.\bar{6}y_4 - 32y_5 + 21.\bar{3}y_6 - \phi^2 y_5 \exp\left(\frac{\gamma\beta(1-y_5)}{1+\beta(1-y_5)}\right) = 0 \tag{6.61}$$

$$y_6 = 1 \tag{6.62}$$

$$-6y_1 + 8y_2 - 2y_3 = 0 \tag{6.63}$$

$$y_3 = y_4 \tag{6.64}$$

$$2y_1 - 8y_2 + 6y_3 = -6y_4 + 8y_5 - 2y_6 \tag{6.65}$$

Due to the presence of the linear equations given in (6.62) –(6.65), variable elimination can be performed and the problem can be reduced to one that is described in terms of the two variables y_2 and y_5 which appear in the nonlinear terms of Equations (6.60) and (6.61). The reformulated equations for y_1 and y_3, given in terms of y_2 and y_5 are shown below in Equations (6.66) and (6.67):

$$y_1 = \frac{20}{17}y_2 - \frac{4}{17}y_5 - \frac{1}{17} \tag{6.66}$$

$$y_3 = \frac{8}{17}y_2 + \frac{12}{17}y_5 - \frac{3}{17} \tag{6.67}$$

Sample values for β, γ, and ϕ in the reduced problem are arbitrarily given values of 0.6, 30, and 0.2, respectively. The variables y_2 and y_5 represent the dimensionless concentrations at corresponding dimensionless locations x_2 = 0.25 and x_5 = 0.75, respectively and therefore vary between zero at the pellet center and unity at the pellet

surface. Problem (6.30) is recast as the NLP given in problem (6.68), where a solution is identified as one in which the sum of the squared constraint violations $z_{1,1}$ and $z_{1,2}$, uniquely specified for a vector $\{y_2,y_5\} \in [0\ 1]^2$, falls below a given tolerance. A plot of the objective function in terms of the dimensionless concentrations y_2 and y_5, corresponding to physical locations at x = 0.25 and 0.75 with respect to the radial axis of the catalyst pellet, is shown in Figure 6.23.

$$
\begin{aligned}
&\min F = \sum_{n=1}^{2} z_{1,n}^2 \\
&s.t.\ \ z_{1,1} = -32y_2 + 32y_3 - \phi^2 y_2 \exp\left(\frac{\gamma\beta(1-y_2)}{1+\beta(1-y_2)}\right) \\
&\qquad z_{1,2} = 10.\bar{6}y_3 - 32y_5 + 21.\bar{3} - \phi^2 y_5 \exp\left(\frac{\gamma\beta(1-y_5)}{1+\beta(1-y_5)}\right) \\
&\qquad 0 \le [y_2, y_5] \le 1 \\
&\qquad [\beta, \phi, \gamma] = [0.6, 0.2, 30]
\end{aligned}
\tag{6.68}
$$

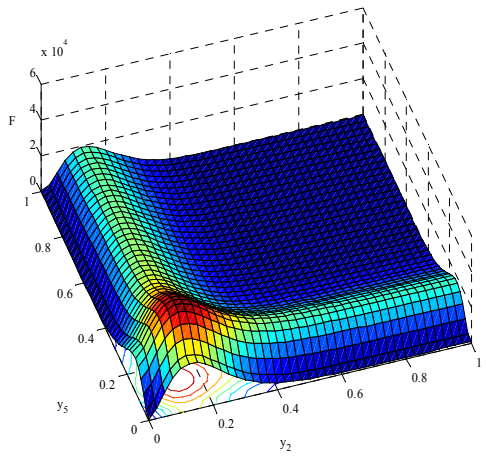


Figure 6.23. Plot of the dimensionless concentration y for a 2-element, 3-node OCFE approximation of the spherical catalyst pellet model given by Equations (6.26) – (6.28).

The solution of this problem consists of two global minima, located at $[y_2,y_5]$ = [0.66594,0.92606] and [0.99313,0.99685]. For both global minima, the objective F

assumes a value falling below a tolerance of 1e-7. There are two difficulties presented when this problem is solved using GAMS and the barrier-terrain method of Lucia et. al. In the former case, the GAMS solver CONOPT fails to identify the two global minima unless the initial box-constrained region is reduced to a fine subregion enclosing each of the global optima. In the latter case, the foundation of the terrain method relies on solutions being connected by smooth valleys such that a set of solutions can be easily identified once one solution has been found, by moving along eigenvector paths while remaining inside the feasible region. However, if a problem contains two optima that are not connected to each other by a smooth valley, movement along eigenvector paths from the first located optimum may not result in discovery of the second optimum, a characteristic noted as a fundamental problem of the terrain method. The kriging-RSM method is therefore applied as an alternative technique for identifying the global optima. Because it does not rely on movement along valleys, it avoids this main difficulty of the terrain method. Tables 6.14 and 6.15 present computational results for the application of both the KC-R and KNC-R algorithms to solve the 2-element, 3-node problem

Table 6.14. Optimization results obtained for Problem (6.68)

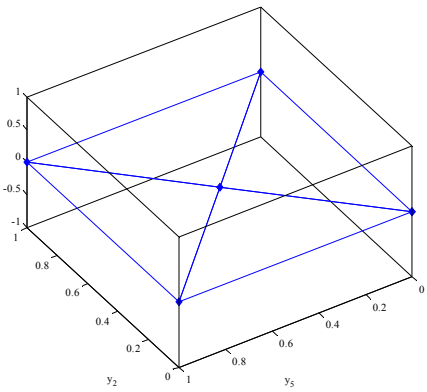
based on application of the KC-R algorithm.

# Iterations		# Function Calls			Optima Found			CPU Time		
KC	R	KC	R	Total = KC + Σ(R)	y_2	y_5	F	KC	R	Total
4	5	57	34	237	0.99310	0.99684	5.65e-7	3.7188	1.9063	5.625
N/A	10	0	72		0.66600	0.92606	1.91e-6	N/A		

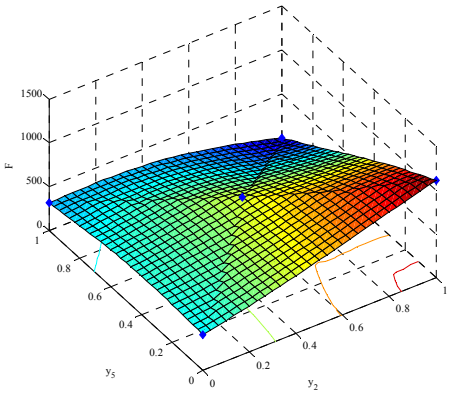
Table 6.15. Optimization results obtained for Problem (6.68) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
9	25	73	143	215	1 (local only)	21/100	8.461	1.653	10.114
7	25	56	155	212	1 (global only)	8/100	6.676	1.318	7.994
10	33	80	216	296	2 (global + local)	69/100	9.583	2.179	11.762

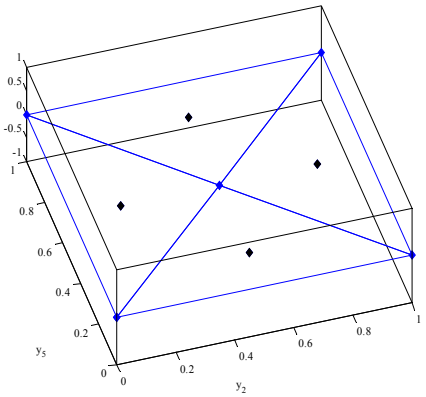
The number of sampling experiments required in order for both the local and global optimum to be identified using the KNC-R method increases by 24.9% relative to the 237 required by the KC-R algorithm; furthermore both optima are identified only at a 69% success rate. The discovery of at least the local optimum is identified in 90% of the cases and is easy to find since it is located close to a feasible point vertex at [1,1]. This sampling point is included in the nominal sampling set for the centroid-based method, and is a likely sampling point during the early stages of the KNC-R method since feasible vectors having high model uncertainty are often identified at points along the feasible region boundaries. The set of iteratively improved global models using the KC-R method are shown in Figure 6.24.


1
0.5
0
-0.5
-1
1
0.8
0.6
0.4
0.2
0
y2
0
0.2
0.4
0.6
0.8
1
y5

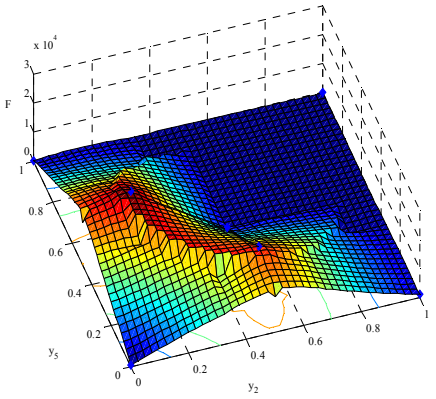
(a)


1500
1000
500
F
0
1
0.8
0.6
0.4
0.2
0
y5
0
0.2
0.4
0.6
0.8
1
y2

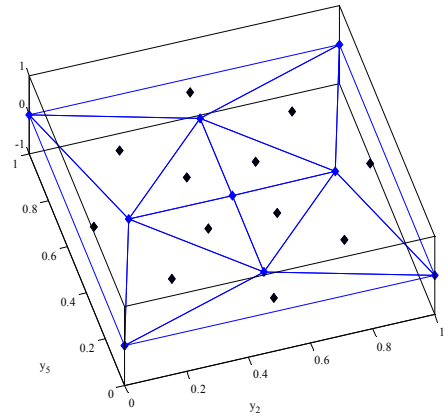
(b)


1
0.5
0
-0.5
-1
1
0.8
0.6
0.4
0.2
0
y5
0
0.2
0.4
0.6
0.8
1
y2

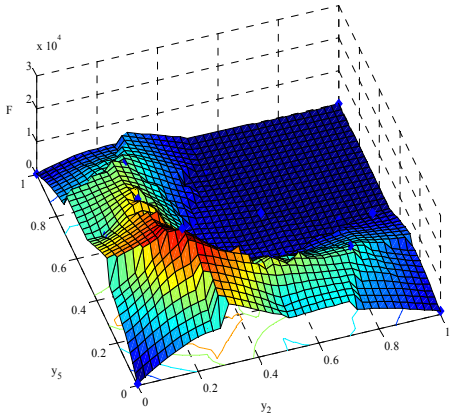
(c)


x 10^4
3
2
1
0
F
1
0.8
0.6
0.4
0.2
0
y5
0
0.2
0.4
0.6
0.8
1
y2

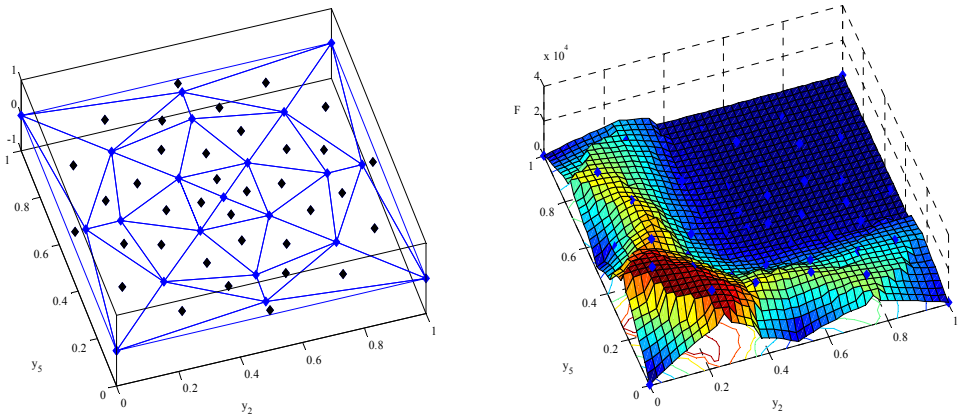
(d)


1
0
-1
1
0.8
0.6
0.4
0.2
0
y5
0
0.2
0.4
0.6
0.8
1
y2

(e)


x 10^4
3
2
1
0
F
1
0.8
0.6
0.4
0.2
0
y5
0
0.2
0.4
0.6
0.8
1
y2

(f)



(g) (h)

Figure 6.24. Iteratively refined kriging models (b), (d), (f), (h) of the objective function given in Problem (6.68), based on application of the KC-R algorithm, with accompanying sampling set plots (a), (c), (e), (g).

6.3.5.2 Discretization based on five elements and three nodes

For a discretization containing five elements and three nodes, the equations are as follows, where $N_E = 5$ and $N_N = 3$:

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]\Big|_{0 \le x \le 0.2} = \left[y^{[1]}, \frac{dy^{[1]}}{dx}, \frac{d^2y^{[1]}}{dx^2} \right]\Big|_{0 \le x \le 0.2} \tag{6.69}$$

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]\Big|_{0.2 \le x \le 0.4} = \left[y^{[2]}, \frac{dy^{[2]}}{dx}, \frac{d^2y^{[2]}}{dx^2} \right]\Big|_{0.2 \le x \le 0.4} \tag{6.70}$$

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]\Big|_{0.4 \le x \le 0.6} = \left[y^{[3]}, \frac{dy^{[3]}}{dx}, \frac{d^2y^{[3]}}{dx^2} \right]\Big|_{0.4 \le x \le 0.6} \tag{6.71}$$

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]\Big|_{0.6 \le x \le 0.8} = \left[y^{[4]}, \frac{dy^{[4]}}{dx}, \frac{d^2y^{[4]}}{dx^2} \right]\Big|_{0.6 \le x \le 0.8} \tag{6.72}$$

$$\left[y, \frac{dy}{dx}, \frac{d^2y}{dx^2} \right]\Big|_{0.8\le x\le 1} = \left[y^{[5]}, \frac{dy^{[5]}}{dx}, \frac{d^2y^{[5]}}{dx^2} \right]\Big|_{0.8\le x\le 1} \tag{6.73}$$

$$\left[\frac{d^2y^{[1]}}{dx^2} + \frac{2}{x}\frac{dy^{[1]}}{dx} - \phi^2 \exp\left(\frac{\gamma\beta\left(1-y^{[1]}\right)}{1+\beta\left(1-y^{[1]}\right)} \right) \right]\Big|_{x=0.1} = 0 \tag{6.74}$$

$$\left[\frac{d^2y^{[2]}}{dx^2} + \frac{2}{x}\frac{dy^{[2]}}{dx} - \phi^2 \exp\left(\frac{\gamma\beta\left(1-y^{[2]}\right)}{1+\beta\left(1-y^{[2]}\right)} \right) \right]\Big|_{x=0.3} = 0 \tag{6.75}$$

$$\left[\frac{d^2y^{[3]}}{dx^2} + \frac{2}{x}\frac{dy^{[3]}}{dx} - \phi^2 \exp\left(\frac{\gamma\beta\left(1-y^{[3]}\right)}{1+\beta\left(1-y^{[3]}\right)} \right) \right]\Big|_{x=0.5} = 0 \tag{6.76}$$

$$\left[\frac{d^2y^{[4]}}{dx^2} + \frac{2}{x}\frac{dy^{[4]}}{dx} - \phi^2 \exp\left(\frac{\gamma\beta\left(1-y^{[4]}\right)}{1+\beta\left(1-y^{[4]}\right)} \right) \right]\Big|_{x=0.7} = 0 \tag{6.77}$$

$$\left[\frac{d^2y^{[5]}}{dx^2} + \frac{2}{x}\frac{dy^{[5]}}{dx} - \phi^2 \exp\left(\frac{\gamma\beta\left(1-y^{[5]}\right)}{1+\beta\left(1-y^{[5]}\right)} \right) \right]\Big|_{x=0.9} = 0 \tag{6.78}$$

$$y^{[1]}\Big|_{x=0.2} = y^{[2]}\Big|_{x=0.2} \tag{6.79}$$

$$y^{[2]}\Big|_{x=0.4} = y^{[3]}\Big|_{x=0.4} \tag{6.80}$$

$$y^{[3]}\Big|_{x=0.6} = y^{[4]}\Big|_{x=0.6} \tag{6.81}$$

$$y^{[4]}\Big|_{x=0.8} = y^{[5]}\Big|_{x=0.8} \tag{6.82}$$

$$\frac{dy^{[1]}}{dx}\Big|_{x=0.2} = \frac{dy^{[2]}}{dx}\Big|_{x=0.2} \tag{6.83}$$

$$\left.\frac{dy^{[2]}}{dx}\right|_{x=0.4} = \left.\frac{dy^{[3]}}{dx}\right|_{x=0.4} \tag{6.84}$$

$$\left.\frac{dy^{[3]}}{dx}\right|_{x=0.6} = \left.\frac{dy^{[4]}}{dx}\right|_{x=0.6} \tag{6.85}$$

$$\left.\frac{dy^{[4]}}{dx}\right|_{x=0.8} = \left.\frac{dy^{[5]}}{dx}\right|_{x=0.8} \tag{6.86}$$

$$[x_1, x_2, x_3] = [0, 0.1, 0.2] \tag{6.87}$$

$$[x_4, x_5, x_6] = [0.2, 0.3, 0.4] \tag{6.88}$$

$$[x_7, x_8, x_9] = [0.4, 0.5, 0.6] \tag{6.89}$$

$$[x_{10}, x_{11}, x_{12}] = [0.6, 0.7, 0.8] \tag{6.90}$$

$$[x_{13}, x_{14}, x_{15}] = [0.8, 0.9, 1] \tag{6.91}$$

$$L_1 = \frac{(x-x_2)}{(x_1-x_2)}\frac{(x-x_3)}{(x_1-x_3)} = 50x^2 - 15x + 1 \tag{6.92}$$

$$L_2 = \frac{(x-x_1)}{(x_2-x_1)}\frac{(x-x_3)}{(x_2-x_3)} = -100x^2 + 20x \tag{6.93}$$

$$L_3 = \frac{(x-x_1)}{(x_3-x_1)}\frac{(x-x_2)}{(x_3-x_2)} = 50x^2 - 5x \tag{6.94}$$

$$L_4 = \frac{(x-x_5)}{(x_4-x_5)}\frac{(x-x_6)}{(x_4-x_6)} = 50x^2 - 35x + 6 \tag{6.95}$$

$$L_5 = \frac{(x-x_4)}{(x_5-x_4)}\frac{(x-x_6)}{(x_5-x_6)} = -100x^2 + 60x - 8 \tag{6.96}$$

$$L_6 = \frac{(x-x_4)}{(x_6-x_4)}\frac{(x-x_5)}{(x_6-x_5)} = 50x^2 - 25x + 3 \tag{6.97}$$

$$L_7 = \frac{(x-x_8)}{(x_7-x_8)}\frac{(x-x_9)}{(x_7-x_9)} = 50x^2 - 55x + 15 \tag{6.98}$$

$$L_8 = \frac{(x-x_7)}{(x_8-x_7)}\frac{(x-x_9)}{(x_8-x_9)} = -100x^2 + 100x - 24 \tag{6.99}$$

$$L_9 = \frac{(x-x_7)}{(x_9-x_7)}\frac{(x-x_8)}{(x_9-x_8)} = 50x^2 - 45x + 10 \tag{6.100}$$

$$L_{10} = \frac{(x-x_{11})}{(x_{10}-x_{11})}\frac{(x-x_{12})}{(x_{10}-x_{12})} = 50x^2 - 75x + 28 \tag{6.101}$$

$$L_{11} = \frac{(x-x_9)}{(x_{11}-x_9)}\frac{(x-x_{10})}{(x_{11}-x_{10})} = -100x^2 + 140x - 48 \tag{6.102}$$

$$L_{12} = \frac{(x-x_{10})}{(x_{12}-x_{11})}\frac{(x-x_{11})}{(x_{12}-x_{11})} = 50x^2 - 65x + 21 \tag{6.103}$$

$$L_{13} = \frac{(x-x_{14})}{(x_{13}-x_{14})}\frac{(x-x_{15})}{(x_{13}-x_{15})} = 50x^2 - 95x + 45 \tag{6.104}$$

$$L_{14} = \frac{(x-x_{13})}{(x_{14}-x_{13})}\frac{(x-x_{15})}{(x_{14}-x_{15})} = -100x^2 + 180x - 80 \tag{6.105}$$

$$L_{15} = \frac{(x-x_{13})}{(x_{15}-x_{13})}\frac{(x-x_{14})}{(x_{15}-x_{14})} = 50x^2 - 85x + 36 \tag{6.106}$$

$$y^{[1]} = L_1 y_1 + L_2 y_2 + L_3 y_3 \tag{6.107}$$

$$y^{[2]} = L_4 y_4 + L_5 y_5 + L_6 y_6 \tag{6.108}$$

$$y^{[3]} = L_7 y_7 + L_8 y_8 + L_9 y_9 \tag{6.109}$$

$$y^{[4]} = L_{10} y_{10} + L_{11} y_{11} + L_{12} y_{12} \tag{6.110}$$

$$y^{[5]} = L_{13} y_{13} + L_{14} y_{14} + L_{15} y_{15} \tag{6.111}$$

$$\frac{dy^{[1]}}{dx} = (100x - 15)y_1 + (-200x + 20)y_2 + (100x - 5)y_3 \tag{6.112}$$

$$\frac{dy^{[2]}}{dx} = (100x - 35)y_4 + (-200x + 60)y_5 + (100x - 25)y_6 \tag{6.113}$$

$$\frac{dy^{[3]}}{dx} = (100x - 55)y_7 + (-200x + 100)y_8 + (100x - 45)y_9 \tag{6.114}$$

$$\frac{dy^{[4]}}{dx} = (100x - 75)y_{10} + (-200x + 140)y_{11} + (100x - 65)y_{12} \tag{6.115}$$

$$\frac{dy^{[5]}}{dx} = (100x - 95)y_{13} + (-200x + 180)y_{14} + (100x - 85)y_{15} \tag{6.116}$$

$$\frac{d^2y^{[1]}}{dx^2} = 100y_1 - 200y_2 + 100y_3 \tag{6.117}$$

$$\frac{d^2y^{[2]}}{dx^2} = 100y_4 - 200y_5 + 100y_6 \tag{6.118}$$

$$\frac{d^2y^{[3]}}{dx^2} = 100y_7 - 200y_8 + 100y_9 \tag{6.119}$$

$$\frac{d^2y^{[4]}}{dx^2} = 100y_{10} - 200y_{11} + 100y_{12} \tag{6.120}$$

$$\frac{d^2y^{[5]}}{dx^2} = 100y_{13} - 200y_{14} + 100y_{15} \tag{6.121}$$

$$\frac{d^2y^{[5]}}{dx^2} = 100y_{13} - 200y_{14} + 100y_{15} \tag{6.122}$$

$$-200y_2 + 200y_3 - \phi^2 y_2 \exp\left(\frac{\gamma\beta(1-y_2)}{1+\beta(1-y_2)}\right) = 0 \tag{6.123}$$

$$66.\bar{6}y_4 - 200y_5 + 133.\bar{3}y_6 - \phi^2 y_5 \exp\left(\frac{\gamma\beta(1-y_5)}{1+\beta(1-y_5)}\right) = 0 \tag{6.124}$$

$$80y_7 - 200y_8 + 120y_9 - \phi^2 y_8 \exp\left(\frac{\gamma\beta(1-y_8)}{1+\beta(1-y_8)}\right) = 0 \tag{6.125}$$

$$85.\overline{714285}y_{10} - 200y_{11} + 114.\overline{285714}y_{12} - \phi^2 y_{11} \exp\left(\frac{\gamma\beta(1-y_{11})}{1+\beta(1-y_{11})}\right) = 0 \tag{6.126}$$

$$88.\bar{8}y_{13} - 200y_{14} + 111.\bar{1}y_{15} - \phi^2 y_{14} \exp\left(\frac{\gamma\beta(1-y_{14})}{1+\beta(1-y_{14})}\right) = 0 \tag{6.127}$$

$$y_{15} = 1 \tag{6.128}$$

$$-15y_1 + 20y_2 - 5y_3 = 0 \tag{6.129}$$

$$y_3 = y_4 \tag{6.130}$$

$$y_6 = y_7 \tag{6.131}$$

$$y_9 = y_{10} \tag{6.132}$$

$$y_{12} = y_{13} \tag{6.133}$$

$$5y_1 - 20y_2 + 15y_3 = -15y_4 + 20y_5 - 5y_6 \tag{6.134}$$

$$5y_4 - 20y_5 + 15y_6 = -15y_7 + 20y_8 - 5y_9 \tag{6.135}$$

$$5y_7 - 20y_8 + 15y_9 = -15y_{10} + 20y_{11} - 5y_{12} \tag{6.136}$$

$$5y_{10} - 20y_{11} + 15y_{12} = -15y_{13} + 20y_{14} - 5y_{15} \tag{6.137}$$

From the set of equations given by (6.128) – (6.137), y_1, y_3, y_6, y_9, and y_{12} can be reformulated in terms of the variables appearing in the nonlinear terms of Equations (6.123) – (6.127), namely y_2, y_5, y_8, y_{11}, and y_{14}, as given below in Equations (6.138) – (6.142):

$$y_1 = \frac{1}{3363}\left(3940y_2 - 676y_5 + 116y_8 - 20y_{11} + 4y_{14} - 1\right) \tag{6.138}$$

$$y_3 = \frac{1}{1121}\left(544y_2 + 676y_5 - 116y_8 + 20y_{11} - 4y_{14} + 1\right) \tag{6.139}$$

$$y_6 = \frac{1}{3363}\left(-280y_2 + 1960y_5 + 1972y_8 - 340y_{11} + 68y_{14} - 17\right) \tag{6.140}$$

$$y_9 = \frac{1}{1121}\left(16y_2 - 112y_5 + 656y_8 + 660y_{11} - 132y_{14} + 33\right) \tag{6.141}$$

$$y_{12} = \frac{1}{3363}\left(-8y_2 + 56y_5 - 328y_8 + 1912y_{11} + 2308y_{14} - 577\right) \tag{6.142}$$

The NLP corresponding to problem (6.30), for a discretization consisting of five elements and three nodes by the OCFE method, is given as follows:

$$\begin{aligned}
&\min F = \sum_{n=1}^{5} z_{1,n}^2 \\
&s.t.\ \ z_{1,1} = -200y_2 + 200y_3 - \phi^2 y_2 \exp\left(\frac{\gamma\beta(1-y_2)}{1+\beta(1-y_2)}\right) \\
&\qquad z_{1,2} = 66.\bar{6}y_4 - 200y_5 + 133.\bar{3}y_6 - \phi^2 y_5 \exp\left(\frac{\gamma\beta(1-y_5)}{1+\beta(1-y_5)}\right) \\
&\qquad z_{1,3} = 80y_7 - 200y_8 + 120y_9 - \phi^2 y_8 \exp\left(\frac{\gamma\beta(1-y_8)}{1+\beta(1-y_8)}\right) \\
&\qquad z_{1,4} = 85.\overline{714285}y_{10} - 200y_{11} + 114.\overline{285714}y_{12} - \phi^2 y_{11} \exp\left(\frac{\gamma\beta(1-y_{11})}{1+\beta(1-y_{11})}\right) \\
&\qquad z_{1,5} = 88.\bar{8}y_{13} - 200y_{14} + 111.\bar{1} - \phi^2 y_{14} \exp\left(\frac{\gamma\beta(1-y_{14})}{1+\beta(1-y_{14})}\right) \\
&\qquad 0 \le [y_2, y_5, y_8, y_{11}, y_{14}] \le 1 \\
&\qquad [\beta, \phi, \gamma] = [0.6, 0.2, 30]
\end{aligned} \tag{6.143}$$

In Tables 6.16 – 6.18, computational results are presented based on the application of the KC-R and KNC-R algorithms to solve the 5-element, 3-node problem given by problem (6.143). Table 6.17 contains the vector information for the two optima.

Table 6.16. Optimization results obtained for problem (6.143) based on the application of the KC-R algorithm.

# Iterations		# Function Calls			CPU Time (s)		
KC	R	KC	R	Total = KC + Σ(R)	KC	R	Total
4	5	280	876	9,686	52.3	36.5	88.8
N/A	36	0	8,530		N/A		

Table 6.17. Optima vectors obtained for problem (6.143) based on application of the KC-R algorithm.

Optima Found					
y_2	y_5	y_8	y_{11}	y_{14}	F
0.99199	0.99263	0.99394	0.99594	0.99853	8.80e-5
0.45717	0.76123	0.89781	0.95473	0.98784	1.81e-7

Table 6.18. Optimization results for problem (6.143) based on application of the KNC-R algorithm.

# Iterations		# Function Calls			# optima found (N_{Opt})	# cases in which only N_{Opt} optima are found/ total # cases	CPU Time (s)		
KNC	R	KNC	R	Total = KNC + Σ(R)			KNC	R	Total
6	12	52	1,918	1,970	1 (local only)	14/100	11.84	2.72	14.57
7	55	53	12,588	12,641	1 (global only)	35/100	12.41	88	100
11	67	96	15,029	15,124	2 (global + local)	40/100	22.13	144	166

The optima vectors differ from those found using a two-element, three-node OCFE discretization because the equations for y, dy/dx, and d^2y/dx^2 are different. The number of sampling experiments required in order for both the local and global optimum to be identified using the KNC-R method increases by 56.1% relative to the 9,686

required by the KC-R algorithm; furthermore both optima are identified only at a 40% success rate. The discovery of at least the global optimum is identified in 75% of the cases. The computational time required for global modeling using the KC-R algorithm is now higher than that required for the KC-R method because the computational time required for Delaunay triangulation increases rapidly when the number of simplex points, and, to a lesser extent, the number of sampling points used in the triangulation, both increase. The solution of this problem was also attempted using OCFE approximations for a five-element, four-node problem, whose reduced problem dimensionality was ten. However, the computational time required for performing the third-iteration Delaunay triangulation became prohibitive, as a new sampling set comprised of 11-point Delaunay triangle centroids could not be obtained even after eleven hours of real clock time had elapsed. Since the limiting factor in the application of the KC-R algorithm to a 10-D problem is the computational complexity of qHull, one natural extension of this work would therefore be the application of a Delaunay triangulation method employing an alternative convex hull algorithm in place of qhull whose computational complexity is lower.

6.4 Summary

In this chapter, a centroid-based sampling algorithm has been presented for iterative global modeling using kriging. The new sampling algorithm is motivated by the opportunity to reduce the sampling expense associated with modeling, which can have practical implications in terms of the resource costs associated with performing field experiments or conducting computationally expensive simulations. The new sampling technique is applied within a kriging-RSM algorithm in order to obtain the complete

solution set of three global optimization test problems and two case studies, for problems containing up to five variables. The promise of the centroid-based sampling algorithm is measured in terms of the sampling expense required to identify all optima for each of the problems, and is compared against the corresponding sampling expense associated with the application of a sampling method employing 1) random sampling for initial modeling, and 2) sampling at locations of minimum prediction, maximum uncertainty, and where there is high model change between the current and previous model.

It is found that application of the new sampling technique results in all optima being found at lower or equivalent sampling expense compared to the sampling algorithm which relies on randomized/heuristic sampling, when the problem dimensionality is low. When the problem dimensionality is higher than five, the Matlab 2008b Delaunay triangulation code, which employs the qHull algorithm, becomes computationally expensive to run, resulting in a significant increase in the global modeling time. Therefore, the centroid-based sampling algorithm has currently been successfully applied for problems containing up to five variables. Each kriging model is refined using additional sampling information, and once the global model is accurate, the best solutions serve as starting iterates for further optimization using RSM. The additional costs resulting from global model creation are offset by successful convergence to improved local solutions.

Chapter 7

Summary and Future Work

The contribution of the work in this dissertation has targeted the optimization of problems containing black-box models and noisy data. The black-box model description is applied when the equations employed by conventional gradient-based algorithm are either missing, as in the case of new research technologies, or, when they do exist, are either inaccurate, inaccessible, or mathematically intractable. In order to address the lack of directly accessible model equations, kriging and response surface methodologies are employed to create global and local surrogate models.

Accurate models are generated using both kriging and RSM based on iterative refinement, in which additional sampling information is used to update earlier models. At the global level, a centroid-based sampling technique has been successfully employed to reduce the overall sampling expense associated with building accurate kriging models. At the local level, adaptive sampling templates have been developed to accelerate movement towards an optimum whenever iterate are near constraints. The generation of lower-D response surfaces projected onto constraints can significantly reduce sampling expense when the problem dimensionality is higher than five. The integration of multiple optimization techniques is a central feature of this work in which the sequential application of Branch-and-Bound, kriging, RSM, and direct search has provided a method for obtaining the solution of problems in which integer variables may also be present both inside and outside the black-box models. Each one of the developed

algorithms - kriging-RSM, B&B Kriging-RSM, B&B-Kriging-RSM-DS, and centroid-based Kriging-RSM – can be employed as either a competitive alternative or a cross-validation technique relative to other algorithms for finding a problem's global optimum.

There remain some areas where additional work can confirm and/or improve the promise of the developed algorithms.

Firstly, in regards to the adaptive experimental designs presented in Chapter 2, the factorial and central composite designs were the main templates employed. Since sampling expense reduction is the key metric used for evaluating algorithmic effectiveness and efficiency, the construction of response surfaces using adaptive templates based on other template such as fractional factorial designs could result in lower sampling being required in the search for an optimum.

Secondly, in regards to the fact that it is not always known *a priori* how much sampling is required in order to obtain an accurate kriging model, a theoretical study of the heuristics employed for generating the sampling set used in kriging model refinement could provide information as to how much additional sampling is required before convergence is attained in the average predictor value.

Thirdly, the generation of a mathematically tractable analytical kriging predictor could be used to identify optima based on gradient calculations rather than by point estimation on a coarse sampling grid. As described in Chapter 3, a kriging prediction is the weighted sum of nearby sampled objective function values, and the weights are obtained by solution of a linear system of covariance information. Based on variable elimination in the linear system, each one of the weights can be analytically given as a nonlinear function of both sampling-pair and sampling-point/test-point covariances.

However, the closed-form expressions become mathematically intractable when the problem dimensionality increases above five. At the same time, the modeling expense multiplicatively increases as the problem dimension increases. The motivation for generating a closed-form predictor would be that model optima, and other locations of interest, such as vectors where there is high model uncertainty, could be determined strictly from the sampling information without requiring the high computational expense associated with generating a prohibitive number of kriging estimates.

Fourthly, the promise of the centroid-based kriging-RSM algorithm needs to be confirmed for problems whose dimensionality is greater than five and for problems in which integer variables are present both inside and outside the black-box models as given by Problems (4.1) and (5.1). The version of qHull employed for generating Delaunay triangulations is based on the seminal work performed by Barber[57] in 1996, and more efficient Matlab-compatible algorithms are expected to have been developed over the past twelve years. In addition, the performance of the centroid-based sampling technique needs to be validated against other sampling templates in terms of the sampling expense associated with the discovery of problem optima. In this work, it has proven superior relative to a methodology that relies on a combination of randomized sampling for initial modeling and heuristic-based sampling for subsequent modeling. However, its performance has not yet been compared against the Centroidal Voronoi tessellation techniques employed by Romero[55] or other iterative sampling methods.

Fifthly, the employment of dynamic kriging could result in significant computational savings for time-dependent CPU-intensive simulation problems. For all the presented problems, the solution vectors are comprised of optimal values for design variables that

are not time-dependent. As an example, consider the kinetics case study presented in Section 2.3 and re-examined in Sections 3.3.5 and 6.3.4. The objective was to obtain optimal reactant concentrations for two species leading to the optimization of a objective function that was a function of steady-state species concentrations. For this problem, only the initial and steady-state species concentrations were of interest. If kriging were successfully applied as a dynamic predictor, substantial computational savings could be obtained since a complete evolution of the microscale system to steady-state each time a function call is required for modeling and optimization would no longer be necessary.

Bibliography

1. Davis, E. and M. Ierapetritou, Adaptive Optimisation of noisy black-box functions inherent in microscopic models, Comp. Chem. Eng. 31, 466, 2007.
2. Davis, E., Ierapetritou, M. A kriging method for the solution of nonlinear programs with black-box functions. AIChE J. **2007**, 53, 2001.
3. Jones, D. A Taxonomy of Global Optimization Methods Based on Response Surfaces. J. Glob. Opt. **2001**, 21, 345.
4. Jones, D., Schonlau, M., Welch, W. Efficient Global Optimization of Expensive Black-Box Functions. J. Global Opt. **1998**, 13, 455.
5. Regis, R., Shoemaker, C. Improved strategies for radial basis function methods for global optimization. J. Global Opt. **2007**, 37, 113.
6. Gutmann, H-M. A Radial Basis Function Method for Global Optimization. J. Global Opt. **2001**, 19, 201.
7. Box, G., Hunter, S., Hunter, W. G. Statistics for Experimenters: Design, Innovation, and Discovery (2nd ed.); Wiley-Interscience: New York, 2005.
8. Myers, R., Montgomery, D. Response Surface Methodology. New York, NY: John Wiley & Sons, Inc., 2002.
9. Barton, R. R., Ivey, J. S. Nelder-Mead Simplex Modifications for Simulation Optimization. Manag. Sci. **1996**, 42, 954.
10. Jones, D. Perttunen, C. Stuckman, B. Lipschitzian optimization without the Lipschitz constant. J. Opt. Theory App. **1993**, 79, 157.
11. Huyer, W., Neumaier, A. Global Optimization by Multilevel Coordinate Search. J. Global Opt. **1999**, 14, 331.
12. Storn, R., Price, K. Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. J. Glob. Opt. **1997**, 11, 341.
13. Brekelmans, R., et. al. Gradient Estimation Schemes for Noisy Functions. J. Opt. Theory Appl. **2005**, 126, 529.
14. Choi, T.D., Kelley, C.T. Superlinear Convergence and Implicit Filtering. SIAM J. Opt. **2000**, 10, 1149.
15. Gilmore, P., Kelley, C.T. An Implicit Filtering Algorithm for Optimization of Functions with Many Local Minima. SIAM J. Opt. **1995**, 5, 269.
16. Isukapalli, S. Uncertainty Analysis of Transport-Transformation Models. Ph.D. thesis, Rutgers University, 1999.
17. Floudas, C.A., Pardalos, P.M. A Collection of Test Problems for Constrained Global Optimization Algorithms. New York, NY: Springer, 1991.
18. Bindal, A., Ierapetritou, M.G., Balakrishnan, S., Makeev, A., Kevrekidis, I., and Armaou, A. Equation-free, coarse-grained computational optimization using timesteppers. Chem. Eng. Sci. **2006**, 61, 779.
19. Gillespie, D. T. A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Reactions. J. Comp. Phys. **1976**, 22, 403.
20. Krige, D. A statistical approach to some mine valuations and allied problems at the Witwatersrand., Master's thesis at the Univ. Witwatersrand, 1951.

21. Goovaaerts, P. Geostatistics for Natural Resources Evolution; Oxford University Press: New York, 1997.
22. Cressie, N. Statistics for spatial data. New York, NY: Wiley, 1993.
23. Isaaks, E. Srivistava, R. Applied geostatistics. New York, NY: Oxford Univ. Press, 1989.
24. Matheron, G. Principles of geostatistics. Econ. Geo. 1963; 58:1246-1266.
25. Sacks, J., Welch, W. J., Mitchell, T.J., Wynn, H.P. Design and Analysis of Computer Experiments. Stat. Sci. **1989**, 4, 409.
26. Santner T. Williams B. Notz. W. The Design and Analysis of Computer Experiments. New York, Springer, 2003.
27. Sacks, J., Schiller, S.B., Welch, W.J. Designs for computer experiments. Technometrics. **1989a**, 31, 41.
28. Maranas, C., Floudas, C. Finding All Solutions of Nonlinearly Constrained Systems of Equations. J. Glob. Opt. **1995**, 7,143.
29. Floudas, C. Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications. Oxford University Press, New York (1995)
30. Weight-In-Feeder Product Illustration. Retrieved 13 August 2008, from the World Wide Web. http://www.rospen.com/downloads/Loss_In_Weight_Feeding.pdf
31. Davis, E., Ierapetritou, M. A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. J. Glob. Opt. **2007**, DOI 10.1007/s10898-007-9217-2.
32. Kocis, G., Grossmann, I. Computational Experience with DICOPT solving MINLP Problems in Process Systems Engineering. Comp. Chem. Eng. **1989**, 13, 307.
33. Gupta, O., Ravindran, V. Branch and Bound Experiments in Convex Nonlinear Integer Programming. Manag. Sci. **1985**, 31, 1533.
34. Fletcher, R., Leyffer, S. Solving Mixed Integer Nonlinear Programs by Outer Approximation. Math. Prog. **1996**, 66, 327.
35. Duran, M., Grossmann, I. An Outer Approximation Algorithm For A Class of Mixed-Integer Nonlinear Programming. Math. Prog. **1986**, 36, 307.
36. Geoffrion, A. Generalized Benders Decomposition. J. Opt. Th. & App. **1972**, 10, 237.
37. Floudas, C. et. al. Global optimization in the 21st century: Advances and Challenges. **2005**, 29, 1185.
38. Achterberg, T., Koch, T., Martin, A. Branching rules revisited. Op. Res. Let. **2005**, 33, 42.
39. Gendron, B., Crainic, T. Parallel Branch-and-Bound Algorithms: Survey and Synthesis. Op. Res. Let. **1994**, 42, 1042.
40. Goyal, V., Ierapetritou, M. Stochastic MINLP optimization using simplicial approximation. Comp. Chem. Eng. **2007**, 31, 1081.
41. Westerlund, T., Pettersson, F. A cutting plane method for solving convex MINLP problems. Comp. Chem. Eng. **1995**, 19, S131.
42. Raman, R., Grossmann, I. Modeling and computational techniques for logic based integer programming. Comp. Chem. Eng. **1994**, 18, 563.
43. Adjiman, C.S., Androulakis, I.P.., Floudas, C.A. Global Optimization of MINLP Problems in Process Synthesis and Design, Comp. Chem. Eng. **1997**, 21, S445.

44. Adjiman, C.S., Androulakis, I.P., Floudas, C.A. A global optimization method, αBB, for general twice-differentiable constrained NLPs – II. Implementation and computational results. Comp. Chem. Eng. **1998**, 22, 1159.
45. Adjiman, C.S., Androulakis, I.P., Floudas, C.A. Global Optimization of Mixed-integer Nonlinear Problems. AIChE J. **2000**, 46, 1769.
46. Adjiman, C.S., Androulakis, I.P., Maranas, C.D., Floudas, C.A. A global optimization method, alpha BB, for process design. Comp. Chem. Eng. **1996**, 20, S419.
47. Wei, J. Realff, J. Sample average approximation methods for stochastic MINLPs. Comp. Chem. Eng. **2004**, 28, 333.
48. Chaudhuri, P., Diwekar, U. Synthesis approach to the determination of optimal waste blends under uncertainty. **1999**, 45, 1671.
49. Seader, J., Henley, E. Separation Process Principles. John Wiley and Sons, New York (2005).
50. Davis, E., Ierapetritou, M. A Kriging-Based Approach to MINLP containing Black-Box Models and Noise. Ind. Eng. Chem. Res. **2008**, DOI 10.1021/ie800028a.
51. McKetta, J. Encyclopedia of Chemical Processing and Design Volume 3; Marcel Dekker: New York/Basel, 1976.
52. Groep, M.E., et. al. Performance modeling and simulation of biochemical process sequences with interacting unit operations. Biotech. Bioeng. **2000**, 67, 300.
53. Siddiqi, S.F., et. al. The effects of fermentation conditions on yeast cell debris particle size distribution during high pressure homogenisation. Bioproc. Eng. **1995**, 14, 1.
54. Mannweiler, K., Hoare, M. The scale-down of an industrial disc stack centrifuge. Bioprocess Eng. **1992**, 8, 19.
55. Davis, E., Ierapetritou, M. A Centroid-Based Sampling Strategy for Kriging Global Modeling and Optimization. Submitted to AIChE J., **2008**.
56. Kleijnen, J. An overview of the design and analysis of simulation experiments for sensitivity analysis. Europ. J. Op. Res. **2005**, 164, 287.
57. Romero, V., Burkardt, J., Gunzburger, M., Petersen, J. Comparison of pure and "Latinized" centroidal Voronoi tessellation against various other statistical sampling methods. Rel. Eng. Sys. Safety. **2006**, 91, 1266.
58. Simpson, T., Lin, D., Chen, W. Sampling Strategies for Computer Experiments: Design and Analysis. Intl. J. Rel. App. **2001**, 2, 209.
59. Barber, C. Dobkin, D., Huhdanpa,, H. The Quickhull Algorithm for Convex Hulls. ACM Trans. Math. Soft. **1996**, 22, 469.
60. Lucia, A., Gattupalli, R., Kulkarni, K., Linninger, A. A Barrier-Terrain Methodology for Global Optimization. IECR, **2008**, 47, 2666.

Edgar F. Davis

Curriculum Vitae

Education

09/03 – 08/08	Ph.D. Chemical and Biochemical Engineering Rutgers, The State University of New Jersey
09/03 – 05/06	M.S., Chemical and Biochemical Engineering Rutgers, The State University of New Jersey
09/99 – 12/01	B.S., Chemical Engineering Texas A&M University
01/02 – 05/03	Designs Engineer, ChevronTexaco

Publications

- Jia, Z., Davis, E., and M. G. Ierapetritou. Predictive Modeling for Mixing and Feeding Processes using Kriging and Response Surfaces. *Manuscript in preparation, 2008*.

- Davis, E. and Ierapetritou, M. A Centroid-Based Sampling Strategy for Kriging Global Modeling and Optimization. Submitted for publication, AIChE J., 2008.

- Davis, E., Ierapetritou, M. A Kriging-Based Approach to MINLP containing Black-Box Models and Noise. Ind. Eng. Chem. Res. **2008**, DOI 10.1021/ie800028a

- Davis, E. and Ierapetritou, M. A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. J. Glob. Opt. DOI 10.1007/s10898-007-9217-2.

- Davis, E. and Ierapetritou, M. A kriging method for the solution of nonlinear programs with black-box functions. AIChE J. **2007**, 53, 2001.

- Davis, E. and M. Ierapetritou, Adaptive Optimisation of noisy black-box functions inherent in microscopic models, Comp. Chem. Eng., **2007**, 31, 466.