Sampling High-Dimensional Design Spaces for Analysis and Optimization

Bemonsteren van hoogdimensionale ontwerpruimtes voor analyse en optimalisatie

Keiichi Ito

Promotoren: prof. dr. ir. T. Dhaene, dr. ir. I. Couckuyt
Proefschrift ingediend tot het behalen van de graad van
Doctor in de ingenieurswetenschappen: computerwetenschappen

UNIVERSITEIT
GENT

Vakgroep Informatietechnologie
Voorzitter: prof. dr. ir. D. De Zutter
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2016 - 2017

ISBN 978-90-8578-952-9
NUR 950, 980
Wettelijk depot: D/2016/10.500/84

UNIVERSITEIT
GENT

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur
Vakgroep Informatietechnologie

Promotoren:	prof. dr. ir. T. Dhaene - Vakgroep Informatietechnologie
	dr. I. Couckuyt - Vakgroep Informatietechnologie
Leden van de examencommissie:	Prof. dr. ir. Gert De Cooman (voorzitter) -Vakgroep Elektronica en Informatiesystemen
	Dr. Yann Caniou - Noesis Solutions
	Prof. dr. Dirk Deschrijver - Vakgroep Informatietechnologie
	Prof. dr. ir. Luc Dupré - Vakgroep Elektrische energie, Systemen en Automatisering
	Prof. dr. Bo Liu - Glyndwr University
	Dr. Selvakumar Ulaganathan - Noesis Solutions

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Informatietechnologie
Technologiepark Zwijnaarde 15, iGent, B-9052 Gent, België

Tel.: +32 9 331 49 00
Fax.: +32 9 331 48 99

Dankwoord

I would like to sincerely thank my promotor, Prof. Tom Dhaene of Ghent University and Roberto d'Ippolito of Noesis Solutions for patiently supervising this research since October 2011. In particular, the IWT Baekeland Mandate program, which Prof. Dhaene proposed me to apply, has enabled me to get out of unemployment and do research. I have been very grateful for the arrangement. Ivo Couckuyt and Silvia Poles, my coauthors of journal papers, have provided much support in the writing and exposition of ideas, which I have appreciated immensely.

Joris Degroot and Selvakumar Ulaganathan have provided important support for using the artery simulation code. Prof. Tsugukiyo Hirayama, Prof. Yoshiaki Hirakawa, and Tatsumi Sakurai have provided crucial support in the seaplane research. I would like to express many thanks for all of the collaborations and help. I have also been supported by the administrative staffs of Ghent University and Noesis Solutions. The points of contact have been with Martine Buysse, Muriel Vervaeke, Davinia Stevens, and Inge Bergers. Thanks a lot for all the helpful support. When I was in IBCN, I had the pleasure of informal discussions with and lunch-time companies of Krishnan Chenmmangat, Domenico Spina, Elizabeth Rita Samuel, Prashant Singh, and Joachim van der Herten. Thank you all for the good times.

Finally, I would like to also express a big gratitude to my partner Megumi Yamamoto who has supported me throughout.

Ghent, November 2016
Keiichi Ito

Table of Contents

List of Figures

List of Tables

List of Acronyms

B

BESTCOM	Belgian Network on Stochastic Modelling, Analysis, Design and Optimization of Communication Systems
BFGS	Broyden Fletcher Goldfarb Shanno
B-WISE	Bayesian Regression Modeling With Interactions and Smooth Effec

D

DOE	Design of Experiments

E

EGO	Efficient Global Optimization
EWIS	Electrical Wiring Interconnection System

F

FDSA	Finite Difference Stochastic Approximation
FR	Fletcher-Reeves

G

GA	Genetic Algorithm
GP	Gaussian Process

H

HAROS-HD	Hybrid Adaptive Robust Optimization Strategy for EWIS High-Dimensional Systems
HDMR	High Dimensional Model Representation

I

IBBT	Interdisciplinary Institute for Broadband Technology
IBCN	Internet Based Communication Networks and Services research group
IWT	Institute for the Promotion of Innovation Through Science and Technology

L

LLE	Local Linear Embedding
L-BFGS	Limited-Memory BFGS

P

PCA	Principal Component Analysis
PR	Polak-Riviere
PSO	Particle Swarm Optimization

S

SOM	Self-Organizing Map
SOMBAS	Self-Organizing Map Based Adaptive Sampling
SPSA	Simultaneous Perturbation Stochastic
SQP	Sequential Quadratic Programming
SVM	Support Vector Machines

Samenvatting
– Summary in Dutch –

De basisveronderstelling van deze thesis is dat numerieke simulaties in de meeste ingenieursproblemen onzekerheden met zich meebrengen omwille van de onderliggende fysiche modellen, ontbrekende variabelen of numerieke fouten. Meer nog, de rekenkost van de objectieven kan duur uitvallen en het aantal variabelen in deze functies kan groot zijn (hoog-dimensioneel). Zeer vaak zijn de simulaties niet compleet of hebben ze vanaf het begin niet voldoende betrouwbaarheid. Anderzijds streven de meeste onderzoeken in optimalisatie, meta-heuristieken en surrogaatmodellen ernaar om de nauwkeurigheid van hun oplossing te verbeteren door het minimum van een functie te vinden. Dit kan in een zeker opzicht een overbodige of buitensporige zoektocht zijn, gezien vanuit het standpunt van de ontwerper.

Het idee is dus om de nauwkeurigheid te vervangen door iets waardevols voor ontwerpingenieurs, die al dan niet de hoogwaardige simulaties van het probleem tot hun beschikking hebben. In eerste instantie stel ik een methode voor die zoekt naar diverse oplossingen die voldoen aan zekere criteria met betrekking tot de objectieven en die goed schaalbaar is naar hoog-dimensionele problemen: SOMBAS (Self-Organizing Maps Based Adaptive Sampling). Vervolgens stel ik een methode voor die interacties kan ontdekken tussen ontwerpvariabelen, hierbij gebruikmakend van minder functie-evaluaties (vergeleken met de methode die gebruikt maakt van de totale gevoeligheid en Sobol index), Interaction Indices (interactie-indices). Tenslotte stel ik een aanpassing van een bestaande stochastische optimalisatiemethode voor, die het gemakkelijker maakt om de methode op te zetten en het hiermee gepaarde vallen en opstaan vermindert of zelfs totaal uitsluit, A_SPSA (Adaptive Initial Step Simultaneous Perturbation Stochastic Approximation).

SOMBAS kan aanzien worden als een steekproefmethode (Design of Experiments) die rekening houdt met de waarde van de resultaten, of als een optimalisatie algoritme dat zoekt naar diversiteit als de waarde van het objectief onder een bepaalde drempelwaarde ligt. Bij ingenieursontwerpproblemen worden vaak in een vroeg stadium reeds een aantal mogelijks concurrentiele ontwerpen voorgesteld. Het doel van deze methode is om te helpen bij zo'n proces door bij benadering een set van reele ontwerpvariabelen te identificeren die leiden tot de gewenste resultaten van een functie of van een computersimulatie. De voorgestelde methode steunt zich niet op geparametriseerde statistische verdelingen, en kan steekproeven ne-

men van multi-modale en niet-convexe verdelingen. Voorts levert de voorgestelde verdienste-functie (merit function) ruimtevullende eigenschappen door de voorkeur te geven aan nieuwe punten die verder weg liggen van de reeds bestaande punten. De resultaten tonen aan dat met dit nieuwe adaptieve steekproefalgoritme op een efficiente manier meerdere haalbare oplossingen kunnen bekomen worden. Onze vernieuwende bijdrage is het herhaaldelijk gebruik van de SOM (zelforganiserende kaarten) bij het aanleren van de dichtheid om haalbare of goede ontwerpen te identificeren, en het toont een zeer snelle toename in de verhouding tussen het aantal haalbare oplossingen en het totale aantal functie-evaluaties. Een toepasssingsvoorbeeld op het ontwerp van een planerende romp (zoals bijvoorbeeld gebruikt in motorboten en watervliegtuigen) toont de verdiensten aan van de aanpak met regio's van haalbare oplossingen vergeleken met huidige trends en ontwerpregels. Bovendien speelden de goed verdeelde punten van de voorgestelde methode een belangrijke rol in de verbetering van de voorspellingsperformantie van een classificatieprobleem aangepakt met SVM (Support Vector Machines - ondersteuningsvector machines).

SOMBAS leert en voegt nieuwe steekproefelementen toe in gebieden waar de resultaten gunstig zijn, en laat hierbij de dichtheid van deze punten progressief toenemen in deze regio's. Voor de geteste functies heeft de voorgestelde methode zijn concurrentiele voordelen getoond ten opzichte van twee evolutionaire algoritmes en het nieuwste van het nieuwste evolutionair algoritme bijgestaan door een surrogaatmodel, waarbij het aantal ontwerpdimensies aangroeide van 20 tot 100. Resultaten tonen aan dat aanpak waarbij dichtheden geleerd worden, een efficient alternatief kan zijn voor de gebruikelijke aanpak met surrogaatmodellen.

Interaction Index (interactie index) is een middel om een gevoeligheidsanalyse te doen, dat nuttig kan zijn bij het opdelen van de originele hoog-dimensionele ontwerpruimte van een objectief functie in een aantal functies met laag-dimensionele ontwerpruimtes. De bijdrage is in het gebruik van heteroscedasticiteit van marginale verdelingen in het opmenten van interacties. De berekening verloopt zeer gelijkaardig aan die van de eerste-orde gevoeligheidsindices in de brute Sobol aanpak. De voorgestelde interactie index kan het relatieve belang kwantificeren van de interagerende ontwerpvariabelen. Bovendien kan de detectie van (niet-)interactie voor doorlichting gedaan worden met slechts $4n + 2$ functie evaluaties, waarbij n het aantal ontwerpvariabelen voorstelt.

A_SPSA lost de moeilijkheid op om de initiele stapgrootte te bepalen voor SPSA (Simultaneous Perturbation Stochastic Approximation). Als de stapgrootte te groot is, is het mogelijk dat de schatting van de oplossing niet convergeert. De voorgestelde methode met adaptieve stapgrootte verkleint automatisch de initiele stapgrootte van de SPSA zodat de vermindering van de functiewaarde van het objectief op een betrouwbaardere manier gebeurt. Tien wiskundige functies met elk 3 verschillende verstoringsniveaus zijn gebruikt om de doeltreffendheid van het voorgestelde idee empirisch aan te tonen. Een voorbeeld over het schatten van de ontwerpparameters van een niet-lineair dynamisch systeem is ook bijgevoegd.

In bijlage worden twee toepassingen van SOMBAS beschreven. Een daarvan gaat over het gebruik van SOMBAS in een ensemble van surrogaatmodellen (en-

semble modeling). We tonen een manier aan om niet-lineaire regressie te doen zonder hierbij gebruik te maken van de kleinste-kwadraten fout. A priori wordt een zeker storingsniveau verondersteld op de bemonsterde data, en een set van regressiemodellen wordt berekend om de “gemiddelde”representatie af te leiden en de daarbij horende variantie. Initiele resultaten tonen aan dat de aanpak om diversiteit op te zoeken verschillende surrogaatmodellen kan laten passen op data met storing, in plaats van plaats van gebruik te maken van de gebruikelijke kleinstekwadraten methode.

Het andere toepassingsvoorbeeld gaat over het gebruik van SOMBAS in een hybride optimalisatie algoritme voor het massale verminderen van gebruikte elektrische kabels in een vliegtuig. Een hoog-dimensionele optimalisatie van discrete ontwerpvariabelen is aangepakt gebruikt makend van een hybride oplossing van verschillende methodes, waaronder een opsplitsing van het probleem gebaseerd op grafen. Het doel was om een optimalisatiemethode te ontwikkelen die snel, nauwkeurig (in het identificeren van de beste oplossing) en schaalbaar (tot een hoog-dimensioneel probleem) is. De gelijktijdige verbetering van de nauwkeurigheid (om een lager gewicht te vinden), de snelheid (een kleiner aantal functieevaluaties) en de schaalbaarheid naar hoog-dimensionele problemen moet nog worden aangetoond.

Tenslotte wordt een algemene heuristiek gesuggereerd voor het verfijnen van de performantie van de methode: er lijkt een afweging te bestaan tussen nauwkeurigheid, snelheid en schaalbaarheid voor het oplossen van een gegeven set van problemen met een gegeven rekenbudget. Twee van deze drie kunnen gelijktijdig verbeterd worden ten koste van het verslechteren van de derde. Een bijkomende performantie dimensie “algemeenheid”zou kunnen toegevoegd worden aan deze afweging, die de toepasbaarheid van een methode opmeet voor verschillende soorten problemen. In dit geval zouden alle drie performantie maatstaven, zijnde nauwkeurigheid, snelheid en schaalbaarheid, gelijktijdig kunnen verbeterd worden tegen de kost van verminderde algemeenheid (i.e. een kleinere set van toepasbare problemen). Dit wil zeggen dat men een meer gespecialiseerde methode gaat ontwikkelen die de gemeenschappelijkheid van een meer specifieke set van problemen gaat uitbuiten waarop de methode toepasbaar is.

Summary

The basic assumption of this thesis is that, in most engineering design problems, numerical simulations entail uncertainties because of the physics, missing variables, or numerical errors. Furthermore, the computational cost of objective functions can be expensive and the number of input variables of these functions may be large (high-dimensional). Very often, the simulations are not complete or of high-fidelity from the beginning. On the other hand, most research efforts in optimization, metaheuristics, and surrogate modeling methods strive to enhance the accuracy of their solution through finding the minimum of a function. This can be, in a sense, an unnecessary or an excessive pursuit from the standpoint of a design practitioner.

Thus, the idea is to exchange the accuracy with something valuable for design engineers who may or may not have the high-fidelity simulation of the problem. Firstly, I propose a method to search for diverse solution satisfying certain objective criteria and that scales well to high-dimensional problems (SOMBAS: Self-Organizing Map Based Adaptive Sampling). Then, I propose a method to detect interactions between design variables at a fewer number of function evaluations than the method using total sensitivity and Sobol Index (Interaction Indices). Finally, I propose a modification of an existing stochastic approximation optimization method that makes it easier to set up, reducing or even eliminating the trial-and-error runs (A_SPSA: Adaptive Initial Step Simultaneous Perturbation Stochastic Approximation).

SOMBAS can be thought of as a Design of Experiments method that takes output values into account or an optimization algorithm that seeks diversity if the objective value is under a given threshold. In engineering design, a set of potentially competitive designs is conceived in the early part of the design process. The purpose of this method is to help such a process by approximate identification of a set of inputs of real variables that return desired responses from a function or a computer simulation. The proposed method does not rely on parameterized distributions, and can sample from multi-modal and non-convex distributions. Furthermore, the proposed merit function provides infill characteristics by favoring sampling points that lay farther from existing points. The results indicate that multiple feasible solutions can be efficiently obtained by the new adaptive sampling algorithm. The iterative use of the SOM in density learning to identify feasible or good designs is our new contribution and it shows a very rapid increase in the number of feasible solutions to the total number of function evaluation ratio. Application examples to planing hull designs (such as in powerboats and seaplanes)

indicate the merits of the feasible region approach to observing trends and design rules. Additionally, the well-distributed sampling points of the proposed method played favorable effect in improving the prediction performance of a classification problem learned by a Support Vector Machine.

SOMBAS learns and adds new samples in the domains where output values are favorable, progressively increasing the density of sample points in these regions. For the functions tested, the proposed method has shown competitive advantages over two evolutionary algorithms and one state-of-the-art surrogate model assisted evolutionary algorithm as the input variable dimensionality grew from 20 to 100. Results show that the density learning approach can be an effective alternative to the conventional surrogate model learning approach.

Interaction Index is a sensitivity analysis tool that can be useful in decomposing the original high-dimensional input space of an objective function into a set of functions with low-dimensional input spaces. The contribution is in the use of heteroscedasticity of marginal distributions in the measurement of interactions. The computation is very similar to first-order sensitivity indices by Sobol' in brute-force approach that computes averages of output values from Monte Carlo samples at every value (level) of input variables. The proposed interaction index can quantify the relative importance of interacting input variables. Furthermore, detection of (non-)interaction for screening can be done with as few as $4n + 2$ function evaluations, where n is the number of input variables.

A_SPSA solves the difficulty of setting up the initial step sizes for the Simultaneous Perturbation Stochastic Approximation (SPSA). If the step size is too large, the solution estimate may fail to converge. The proposed adaptive stepping method automatically reduces the initial step size of the SPSA so that reduction of the objective function value occurs more reliably. Ten mathematical functions each with three different noise levels were used to empirically show the effectiveness of the proposed idea. A parameter estimation example of a nonlinear dynamic system is also included.

In appendices, two applications of SOMBAS is described. One is SOMBAS in ensemble modeling. We show a way of non-linear regression without resorting to the least-square-error. A certain noise level is a priori assumed on the sampled data output, and a set of regression models are drawn to infer an *average* representation and accompanying variance. It shows proof-of-concept results on the diversity-seeking approach to fit multiple surrogates to noisy data instead of the usual least-square-error fit.

The other is the use of SOMBAS in a hybrid optimization algorithm for mass reduction of an electrical wire system in aircraft. A high-dimensional optimization of discrete input variables is tackled using hybrids of methods including a graph-based problem decomposition. The objective was to create an optimization method that was fast, accurate (in identifying the best solution), and scalable (to a high-dimensional problem). Negative results were obtained at the time of reporting.

Finally, an overall heuristic for refining methodology performances is suggested: there seems to be a trade-off between accuracy, speed, and scalability for solving a given set of problems with a given computational budget. Two out of

these three can be improved at the expense of the remaining one. An additional performance dimension *generality* could be added to this trade-off, measuring a method's applicability to different kinds of problems. In this case, all three performance measures namely accuracy, speed, and scalability could be simultaneously improved at the expense of reduced generality (i.e. a smaller set of applicable problems). That is, one designs a more specialized method exploiting the commonality of a more specific set of problems that the method applies to.

1 Introduction

"Festina lente. (Make haste slowly.)"

–Author unknown

1.1 Background

The motivation behind the research presented in this thesis is that in engineering design environments, optimization often fails to give satisfactory results or even unusable results. The reasons are the following. The simulation workflow that defines the input design variables and returns the objective value is not perfect. There are numerical and modeling errors [1]. On top of that, optimization is seldom holistic, i.e., there are factors that are not considered in generating the objective value [2, 3].

Complex engineering problems use high fidelity simulation models and expensive experiments. However, ever increasing competition imposes time and budget constraints on the development of new products. Therefore, we want to infer the maximum of information from a limited number of simulation runs and still obtain competitive solutions and products.

We should remember that we never achieve 100% accurate simulation or experiments [4] and if they are high fidelity, they are expensive. Furthermore, as the computational power increases, problems keep scaling towards a higher number of design parameters and data size. Such high-dimensional problems abound in designing of complex systems such as aircraft, car, and network systems such

as smart grids for electrical power generation and distribution to name a few. In multidisciplinary settings, high-dimensional and time-consuming simulations are very likely. However, to the author's experience most design optimization and surrogate modeling literature shows cases of less than 20 design parameters or high-dimensional problems optimized or modeled with a large number of function evaluations. In practice, optimization is mostly applied at a component level or in a simplified system. There seems to be an unfilled gap for practical optimization and analysis methods that deal with high-dimensional and expensive functions.

The practical approach of engineering design is that quantitative accuracy of predicted output is not always of primary importance but relative merits between different designs are. Very often a "competitive" solution is enough instead of the best possible solution. This is especially true if the problem is multiobjective or highly constrained. This is also a practical way to mitigate the risk of the unknown or the known-but-not-considered. The power of engineering is in approximation. The objective is to devise practical sampling (or optimization) and analysis methods in the face of uncertainty and inaccuracy.

1.2 Problem Domain

We deal with the problem of analyzing the real variable input and output relationships of simulation codes. We consider a vector of input variables and a scalar output in order to look for inputs that satisfy certain conditions in the output (i.e. optimization, surrogate modeling, and design of experiments) or quantify the effect of input variables (interaction) on the output (i.e. sensitivity analysis). We assume a situation in which the evaluation of such functions (e.g. simulation codes) are expensive and only a limited number trials can be made. The number of input variables can be relatively large (up to 100).

1.3 Central Theme

The central theme of this thesis is about "relaxing". Instead of looking for a single instance, relaxed methods typically look for a set, an interval, or a density distribution or tolerates such variance for approximations. Another form of relaxation is the adaptive nature of a method handling different situations. In short, we trade preciseness or accuracy with some kind of efficiency or reduction of cost. The inspiration came from the probabilistic argument of Ordinal Optimization [5].

1.3.1 Ordinal Optimization

Ordinal Optimization proposed by Ho et al. [6] employs a different paradigm optimization where one seeks to identify a subset of good designs based on the order-

ing of alternative designs which can be determined based on a very rough or cheap simulation models. Lau and Ho [7] describe the high probability that the selected subset actually contains good designs.

Define a good enough set $\mathbf{G}$ as a subset consisting of the top n % in the design space $\mathbf{\Theta}$. This subset is the truly good ones that we seek to identify. On the other hand, define the selected subset $\mathbf{S}$ as a subset of $\mathbf{\Theta}$ in which the members are picked out by the designer using certain evaluation scheme (be it by some simulation, past experience, or fortune telling) but without the knowledge of their true performances. The objective of Ordinal Optimization is to match $\mathbf{G}$ and $\mathbf{S}$ up to at least a certain level. They refer to this degree matching as "alignment level". Ordinal Optimization is based on two principles. (i) The order is much easier to determine than the actual value – It is easier to determine if $A > B$ than to determine $A - B =?$, (ii) Softening the goal makes a hard problem easier. Instead of asking for the best, let us settle for the good enough with high probability. To make this point clearer, Ho et al. [5] show the following simple calculations. If we can sample uniformly in $\mathbf{\Theta}$, the probability that one of the N samples falls in the top n-percent of $\mathbf{\Theta}$ is

$$\begin{aligned} P &= 1 - \text{Prob}\{\text{all } N \text{ samples not in the top } n\text{-percent of } \mathbf{\Theta}\} && (1.1)\\ &= 1 - \left(1 - \frac{n}{100}\right)^N. && (1.2) \end{aligned}$$

For $N = 1000$ and $n = 5$, we have $(1-5/100)^{1000} \simeq 5.29 \times 10^{-23}$, which makes $P \simeq 1$. Now, let us define

$$\begin{aligned} g &= |\mathbf{G}| && (1.3)\\ s &= |\mathbf{S}|, && (1.4) \end{aligned}$$

where $|\cdot|$ gives the number of elements of the set. Suppose we blindly pick the elements of $\mathbf{\Theta}$ to form $\mathbf{S}$, then the probability that $\mathbf{S}$ and $\mathbf{G}$ shares no common element is given by

$$\text{Prob}\left[|\mathbf{S} \cap \mathbf{G}| = 0\right] = \frac{\binom{N-g}{s}}{\binom{N}{s}}. \quad (1.5)$$

Thus, the probability that at least one of the selected samples $\mathbf{S}$ is indeed a member of the good enough samples $\mathbf{G}$ is

$$\begin{aligned} \text{Prob}\left[|\mathbf{S} \cap \mathbf{G}| \geq 1\right] &= 1 - \text{Prob}\left[|\mathbf{S} \cap \mathbf{G}| = 0\right] && (1.6)\\ &= 1 - \frac{\binom{N-g}{s}}{\binom{N}{s}} && (1.7)\\ &= 1 - \frac{\frac{(N-g)!}{s!(N-g-s)!}}{\frac{N!}{s!(N-s)!}} && (1.8)\\ &= 1 - \frac{(N-g)(N-g-1)\cdots(N-g-s+1)}{(N)(N-1)\cdots(N-s+1)} && (1.9) \end{aligned}$$

Since

$$\frac{N-g-i}{N-i} \leq \frac{N-g}{N} = 1 - \frac{g}{N}$$

for all $i = 0, 1, \ldots, s-1$, we have

$$\text{Prob}\left[|\mathbf{S} \cap \mathbf{G}| \geq 1\right] \geq 1 - \left(1 - \frac{g}{N}\right)^s. \tag{1.10}$$

Furthermore, $1 - x \leq e^{-x}$ holds for all x, so we have

$$\text{Prob}\left[|\mathbf{S} \cap \mathbf{G}| \geq 1\right] \geq 1 - e^{-\frac{gs}{N}}. \tag{1.11}$$

Thus, the probability of correctly identifying at least one good enough candidate approaches exponentially to one with increasing size of $\mathbf{S}$ and $\mathbf{G}$. This is the result for the blind picking case. Yet, for $s = g = 50$ and $N = 1000$, the probability of identifying at least one good enough (top 5 %) solution in the randomly picked set $\mathbf{S}$ is at least 91.8 %. If some inference can be made to the expected performances to the N samples, no matter how crude, the chances could be even better than what we just obtained.

1.3.2 The Lessons

The above claim contains some remarkable insight. Although the argument was based on sets with a finite number of elements, relaxing the objective from finding the "best" to a few of top few % has enabled an exponential convergence rate with a random selection process. Furthermore, the above argument did not mention the design variables, whose number could be high, but only a finite set of alternative designs sampled uniformly from the design space. Unlike an early stopping of optimization which normally does not have control over the level of the relaxation, ordinal optimization can control it by specifying the probability of finding the desired number of elements of the desired level of performance. This has a few additional lessons to take away beside the obvious benefit of relaxing the target requirement. At a more abstract level, we have the following: 1) Changing objectives. There could be other measures that can be exploited easily that in turn helps in improving a more difficult measure that we were after. 2) Eliminating needs. We may be seeking something unnecessarily. The lessons are reflected in the following chapters.

1.4 Thesis Organization

In Chapter 2, we will relax the target from finding the best to finding a diverse solution set that satisfies certain objective criteria. It uses Self-Organizing Maps to create a discrete set of sampling points which roughly corresponds to the samples

from the density (or probability distribution) of good points in the design space. The proposed algorithm Self-Organizing Maps Based Adaptive Sampling (SOMBAS) was tested on three mathematical functions and an engineering problem to see its space-filling characteristics and scalability.

In Chapter 3, we take a look at the optimization characteristics of SOMBAS. Seven test functions were solved at three different number of dimensions and three different function evaluation budgets. The results of minimization were compared with three different algorithms which share similar design aspects as those of SOMBAS.

In Chapter 4, we propose a new index which tells the degree of interaction among input variables in generating the variance in the output. Traditionally, global sensitivity analysis requires a lot of function evaluations to obtain accurate values, e.g., using Monte Carlo Integrations. The proposed method relaxes the need for accurate integration by measuring heteroscedasticity instead of the difference between total sensitivity and first order sensitivity. Four mathematical functions and one engineering example are analyzed and compared with the classical method of evaluating interactions.

In Chapter 5, we relax the need to find a good initial setting for an optimization algorithm called Simultaneous Perturbation Stochastic Approximation (SPSA). Ten mathematical functions and a parameter estimation problem are used to compare the performance of the classical SPSA and the proposed adaptive initial step SPSA.

Lastly, appendix chapters describe supplementary materials. In Appendix A, we describe briefly a case of relaxing the need to find the best-fit surrogate model using SOMBAS to fit a set of surrogate models. Then show an example of looking for a minimum response solution from a set of noisy measurements. In Appendix B, an industrial application case is summarized in which hybrid methods attempts to achieve an uncompromising performance improvement over the existing optimization method in terms of minimum objective value found and a number of function evaluations.

1.5 Relevant Literature

This section is a survey of relevant research and school of thoughts that formed the foundation of this thesis.

1.5.1 High-Dimensional Optimization

This subsection describes research specifically targeting optimization of high-dimensional problems.

Fletcher-Reeves (FR) [8] and Polak-Riviere (PR) [9] are two non-linear conjugate gradient methods developed in the sixties. They require no matrix inversion, only three vector information besides the input vector (i.e. current gradient, updated gradient, and updated search direction) are stored in each iteration, and are linearly convergent. FR has a weakness such as once deterioration in search direction happens the subsequent iterations become very slow. PR, on the other hand, may not be monotonically decreasing. In [10], some variants of FR and PR are described that mitigate the shortcomings.

Limited-Memory Quasi-Newton Method (L-BFGS) maintains compact approximations of Hessian matrices [10]. Instead of storing full dense Hessian matrices, they save only a few vectors of length equal to the number of input parameters. The number of vectors stored can be specified by the user. For the constrained problems, Sequential Quadratic Programming (SQP) using L-BFGS has also been investigated by employing the same limited-memory formulation to the Hessian of the Lagrangian function [11].

The non-linear conjugate gradient and Limited-Memory Quasi-Newton require information of the gradient of objective functions. If they are not obtainable cheaply or the objective functions are noisy, the application of these gradient based methods may not be feasible. The following methods are designed to avoid this problem.

Pattern-Search Methods choose a finite set of search directions at each iteration and evaluate objective function at a given step length along each of these directions. This is a search over a grid which can be modified at each iteration. If a point with significant improvement in objective value is found, it is adopted as a new center point from which a new set of search directions is determined. The number of function evaluations can be saved, at given iteration, by not computing all the directions once a point at which substantial improvement occurs is found. Torczon and Trosset [12] do not recommend using Pattern-Search to high dimensional problems. However, the method is robust against noisy and non-smooth functions [13] and could potentially be useful if combined with surrogate models. Local convergence of Pattern-Search Methods is slower compared to gradient based methods. On the other hand, Pattern-Search Methods are good at locating a general region of a stationary point from any given starting point [12].

Cooperative Co-evolution decomposes the original problem into sub-components of lower orders, and solves these sub-components separately. Then a process called co-adaptation (co-evolution) is performed so as to take into account the interaction between the input parameters. In [14], the parameters are grouped randomly to sub-components in each iteration.

In Simultaneous Perturbation Stochastic Approximation (SPSA), the initial design parameter vector θ of p-dimensions is perturbed simultaneously in every dimension i.e., by adding and subtracting. a perturbation vector Δ of p-dimensions,

thus obtaining an estimate of the gradient. Unlike the traditional finite differencing approach, it only takes two function evaluation to obtain the estimate of the gradient. Yet, the number of iteration needed for convergence to the optimum is said to be more or less the same with Finite-Difference Stochastic Approximation (FDSA) [15], which in essence is an approximate Steepest-Descent Method that uses finite-differencing to approximate the partial derivatives along each of the p parameters. Thus, the number of function evaluation of SPSA is p-fold smaller compared to FDSA [16]. An extension to this method exists to include second-order (Hessian) effects to accelerate convergence [17, 18].

1.5.2 Surrogate Modeling

This subsection describes surrogate modeling addressing the challenges of constructing a representation of the computationally expensive model from sparse samples i.e. fighting the "curse of dimensionality".

1.5.2.1 Model Order Reduction

The most straightforward way of fighting the curse is to reduce the model to a simpler model requiring fewer state variables or parameters.

Screening is a process of identifying influential parameters to the objectives among all the parameters in consideration. Under the mild interaction assumption one can employ Fractional Factorial design [19] for this purpose. Morris [20] proposed a one-at-a-time method which provides qualitative sensitivity analysis at a cost proportional to the number of random replicates times the number of dimensions, where number of random replicates tend to be modest (say 4 to 10).

Kernel PCA is a non-linear version of Principal Component Analysis (PCA). It transforms the input parameter space into a nonlinear feature space (kernel space) and performs PCA in this space. PCA is a coordinate transformation method that aligns its axes to the direction of largest variance [21].

High-Dimensional Model Representation (HDMR) is a method to represent a function with a sum of a set of functions of increasing dimensions [22, 23]. This lets us decompose a function into elementary effects (i.e. sum of functions depending on one input parameter) and interaction effects (sum of function depending on 2 or greater number of parameters). Practical engineering experience tells us that elementary effects and low-order interaction are usually enough to capture the characteristics of the input - output relations engineers want to model. This idea of representing the original function with a set of lower dimension functions is also used in B-WISE model in [24].

In a very high dimensional space, any two random vectors tend to become close to orthogonal (if you take the inner product of two large random vectors with the mean value at zero, it tends to zero). Fast Jonson-Lindenstrauss Transform

[25] takes this property to project the original high dimensional vectors to a small number of random vectors.

1.5.2.2 Ensemble Modeling

Ensemble Modeling fits multiple surrogate models to a given set of input-output data. Committee Machines [26] show some approaches regarding this task: Averaging, Bagging, Boosting, and Mixture of Experts. The benefit of ensemble modeling is that by using multiple models one can cancel out errors of the individual models. Also, each individual may specialize in different sub-domains of the input space so that local accuracy is improved. Ensemble modeling has been used in evolving population of surrogate models to find an "optimal" surrogate model [27].

Model Averaging is a process of taking output from multiple models (one can fit different surrogate models to the same set of data) and taking a (weighted) average of them.

In Bagging, K data points are sampled from the original data set of size K with replacement. It is repeated M times and obtain M different data sets. These data sets are used to train M different models, the output of which can be averaged.

In Boosting, committee members are trained sequentially, and the training of particular committee member is dependent on the training and performance of previously trained members. This is essentially fitting surrogates on the error of previously fit models.

Mixture of Experts is a system in which each surrogate models (neural networks) are responsible for modeling different regions in input space.

Multifidelity Modeling in which expensive simulation model is combined with cheaper (often semi-empirical) model to alleviate the computational burden is another form of Ensemble Modeling. A successful implementation of this type is Space Mapping [28]. It performs a mathematical mapping between the spaces of parameters of two different models, which maps the fine model parameter space to the coarse model parameter space such that the responses of the coarse model adjust for the responses of the fine model within some local modeling region around the optimal coarse model solution.

1.5.2.3 Distance Metrics

The basic building component of most surrogate models is in the determination of distance between two samples. According to Denison [29], the Euclidean distance typically employed in low dimensional space becomes ineffective in a high-dimensional space.

Fractional Order Distance Metrics is a way to mitigate this curse of dimensionality. While the Euclidian distance is computed as "square-root of sum-of-

squares", often denoted as l^2-norm. One can replace the 2 with a fraction or a real variable p such as l^p with $p < 1$. Although $p < 1$ violates the mathematical definition of metrics (does not satisfy triangular inequality), Aggarwal et al. [30] reported that, in high-dimensional space, this improved the performance of nearest-neighbor classification algorithm compared to that using the Euclidean distance. Flexer and Schnitzer [31] proposed a method to determine an appropriate l^p-norm from a given dataset for nearest-neighbor classification.

1.5.3 Adaptive Sampling

Adaptive sampling techniques combine the idea of the design of experiment and optimization. It sequentially determines the next sampling point in the input space based on some performance information based on the output from the original function, the output from the surrogate model, or the combination of both.

Efficient Global Optimization (EGO) [32] uses Kriging as the surrogate model. Since Kriging provides a measure of uncertainty in its predicted output, it is possible to calculate, given input parameters, how much probability exists in finding a better point than what has been found so far. EGO searches the input space that would give the maximum expected improvement (first moment of probability of improvement) on the Kriging model.

Torczon's Merit Function [33] subtracts a distance measure from the surrogate model's output. The distance measure is the distance between the current point and the nearest sample in the database defining the surrogate model. Thus, optimizing on merit function avoids sampling at points too close to previously sampled points. This should provide a balance between exploration and exploitation similar to EGO.

Local Linear Approximation - Voronoi (LOLA-Voronoi) [34] takes half of its sampling uniformly in the input space and the remaining half from where the gradient of the response is steep. The gradient is estimated using existing samples in the database of the surrogate model. Since it performs denser sampling where the objective function exhibit high non-linearity, the accuracy of the surrogate model is improved compared to a surrogate model fit on uniform sampling with the same number of samples.

1.6 Research contributions

1. A new merit function that performs both optimization and space-filling task (Ch. 2).

2. A new algorithm that adaptively samples from a population density (Ch. 2).

3. A new index that shows whether an input variable change has an additive effect in the output variance or if it has an interaction with other input variables (Ch. 4).

4. A new adaptive algorithm that facilitates the setup of Simultaneous Perturbation Stochastic Approximation (Ch. 5).

1.7 Publications

The results obtained during this Ph.D. research are published in scientific journals and presented at a series of international conferences. The following list provides an overview of the publications during my Ph.D. research.

1.7.1 Publications in international journals (listed in the Science Citation Index [1])

1. **Keiichi Ito**, Ivo Couckuyt, Roberto d'Ippolito, Tom Dhaene. *Design Space Exploration using Self-Organizing Map Based Adaptive Sampling*. Published in Journal of Applied Soft Computing, DOI: 10.1016/j.asoc.2016.02.036, Vol. 43, pp. 337 - 346, June 2016.

2. **Keiichi Ito**, Ivo Couckuyt, Silvia Poles, Tom Dhaene. *Variance-based interaction index measuring heteroscedasticity* . Published in Computer Physics Communications, DOI: 10.1016/j.cpc.2016.02.032, Vol. 203, pp. 152 - 161, June 2016.

3. **Keiichi Ito**, Tom Dhaene. *Adaptive initial step size selection for Simultaneous Perturbation Stochastic Approximation* . Published in SpringerPlus, DOI: 10.1186/s40064-016-1823-3, Vol. 5, No. 1, pp.1 - 18, February 2016.

4. **Keiichi Ito**, Yoshiaki Hirakawa, Tsugukiyo Hirayama, Tatsumi Sakurai, Tom Dhaene. *Longitudinal Stability Augmentation of Seaplanes in Planing* . Published in AIAA Journal of Aircraft, DOI: 10.2514/1.C033588, Vol. 53, No. 5, pp. 1332 - 1342, September 2016.

1.7.2 Publications in international conferences (listed in the Science Citation Index [2])

None.

[1] The publications listed are recognized as 'A1 publications', according to the following definition used by Ghent University: A1 publications are articles listed in the Science Citation Index, the Social Science Citation Index or the Arts and Humanities Citation Index of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper.

[2] The publications listed are recognized as 'P1 publications', according to the following definition used by Ghent University: P1 publications are proceedings listed in the Conference Proceedings Ci-

1.7.3 Publications in other international conferences

1. **Keiichi Ito**, Tom Dhaene, Naji El Masri, Roberto d'Ippolito, Joost Van de Peer. *Self-Organizing Map Based Adaptive Sampling.* Published in proceedings of 5th International Conference on Experiments/Process/System Modeling/Simulation/Optimization (5th IC-EpsMsO, Athens, Greece, July 3 - 6, 2013), Vol II, pp. 504 - 513, 2013, ISBN:978-618-80527-2-7 or 978-618-80527-0-3.

2. **Keiichi Ito**, Yoshiaki Hirakawa, Tsugukiyo Hirayama, Tatsumi Sakurai, Tom Dhaene. *Longitudinal Stability Augmentation of Seaplanes in Planing.* Published in proceedings of AIAA Modeling and Simulation Technologies Conference (Aviation 2015, Dallas, Texas, June 22 - 26, 2015), DOI: 10.2514/6.2015-2498.

1.7.4 Publications in national conferences

None.

tation Index - Science or Conference Proceedings Citation Index - Social Science and Humanities of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper, except for publications that are classified as A1.

References

[1] Y. Tenne. *A computational intelligence algorithm for simulation-driven optimization problems*. Advances in Engineering Software, 47:62 – 71, 2012.

[2] T. Ulrich and L. Thiele. *Maximizing Population Diversity in Single-objective Optimization*. In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, pages 641–648, New York, NY, USA, 2011. ACM. Available from: http://doi.acm.org/10.1145/2001576.2001665, doi:10.1145/2001576.2001665.

[3] A. A. Taflanidis and J. L. Beck. *An efficient framework for optimal robust stochastic system design using stochastic simulation*. Computer Methods in Applied Mechanics and Engineering, 198(1):88 – 101, 2008. Computational Methods in Optimization Considering Uncertainties. Available from: http://www.sciencedirect.com/science/article/pii/S0045782508001461, doi:http://dx.doi.org/10.1016/j.cma.2008.03.029.

[4] T. Oden, R. Moser, and O. Ghattas. *Computer Predictions with Quantified Uncertainty, Part I*. SIAM News, 43(9):1, November 2010.

[5] Y.-C. Ho, Q.-C. Zhao, and Q.-S. Jia. *Ordinal Optimization: Soft Optimization for Hard Problems*. Springer Science+Bussiness Media, 2007.

[6] Y. C. Ho, R. S. Sreenivas, and P. Vakili. *Ordinal Optimization of DEDS*, 1996.

[7] E. Lau and Y. Ho. *Universal Alignment Probabilities and Subset Selection for Ordinal Optimization*. J. Optimization Theory and Applications, 93:455–489, 1997.

[8] R. Fletcher and C. M. Reeves. *Function minimization by conjugate gradients*. The Computer Journal, 7(2):149–154, February 1964. Available from: http://dx.doi.org/10.1093/comjnl/7.2.149, doi:10.1093/comjnl/7.2.149.

[9] E. Polak and G. Ribière. *Note sur la convergence de méthodes de directions conjugées*. Revue Française d'Informatique et de Recherche Opérationnelle, 16:35 – 43, 1969.

[10] J. Nocedal and S. J. Wright. *Numerical Optimization, 2nd. Ed*. Springer Science + Business Media, 2006.

[11] P. E. Gill, W. Murray, and M. A. Saunders. *SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization*. SIAM Review, 47(1):99 – 131, 2005.

[12] V. Torczon and M. W. Trosset. *From Evolutionary Operation to Parallel Direct Search: Pattern Search Algorithms for Numerical Optimization*. Computing Science and Statistics, 29:396–401, 1998.

[13] T. G. Kolda, R. M. Lewis, and V. Torczon. *Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods*. SIAM Review, 45(3):385 – 482, 2003.

[14] Z. Yang, K. Tang, and X. Yao. *Large scale Evolutionary Optimization Using Cooperative Coevolution*. Information Sciences, 178:2985 – 2999, 2008.

[15] J. Kiefer and J. Wolfowitz. *Stochastic Estimation of the Maximum of a Regression Function*. Annals of Mathematical Statistics, 23:452–466, September 1952.

[16] J. C. Spall. *An Overview of the Simultaneous Perturbation Method for Efficient Optimization*. Johns Hopkins APL Technical Digest, 19(4):482–492, 1998.

[17] J. C. Spall. *Adaptive Stochastic Approximation by the Simultaneous Perturbation Method*. Transactions on Automatic Control, 45(10):1839–1853, October 2000.

[18] X. Zhu and J. C. Spall. *A modified second-order SPSA optimization algorithm for finite samples*. International Journal of Adaptive Control and Signal Processing, 16:397–409, 2002.

[19] R. H. Myers and D. C. Montgomery. *Response Surface Methodology: Process and Product in Optimization Using Designed Experiments*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1995.

[20] M. D. Morris. *Factorial sampling plans for preliminary computational experiments*. Technometrics, 33:161 – 174, 1991.

[21] S. Marsland. *Machine Learning: An Algorithmic Introduction*. CRC Press, New Jersey, USA, 2009.

[22] I. M. Sobol. *Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates*. Mathematics and Computers in Simulation, 55:271–280, 2001.

[23] H. Rabitz and O. Aliş. *General foundations of high-dimensional model representations*. Journal of Mathematical Chemistry, 25:197–233, 1999. 10.1023/A:1019188517934. Available from: http://dx.doi.org/10.1023/A:1019188517934.

[24] P. Gustafson. *Bayesian Regression Modeling with Interaction and Smooth Effects*. Journal of the American Statistical Association, 95(451):795 – 806, September 2000.

[25] N. Ailon and B. Chazelle. *Faster Dimension Reduction*. Communications of the ACM, 53(2):97 – 104, February 2010.

[26] V. Tresp. *Committee Machines*. In Y. H. Hu and J.-N. Hwang, editors, Handbook for Neural Network Signal Processing. CRC Press, 2001.

[27] I. Couckuyt, F. D. Turck, T. Dhaene, and D. Gorissen. *Automatic Surrogate Model Type Selection During the Optimization of Expensive Black-Box Problems*. In Proceedings of the 2011 Winter Simulation Conference, pages 4274 – 4284, 2011.

[28] J. W. Bandler, M. Biernacki, and S. H. Chen. *Space Mapping Technique for Electromagnetic Optimization*. IEEE Transaction on Microwave Theory and Techniques, 42(12):2536 – 2544, December 1994.

[29] D. Denison, C. Holmes, B. Mallick, and A. Smith. *Bayesian Methods for Nonlinear Classification and Regression*. Wiley Series in Probability and Statistics. John Wiley & Sons, 2002. Available from: http://books.google.be/books?id=SIlDWySNuXgC.

[30] C. Aggarwal, A. Hinneburg, and D. Keim. *On the surprising behavior of distance metrics in high dimensional space*. Database Theory - ICDT 2001, Lecture Note in Computer Science, 1973:420–434, 2001.

[31] A. Flexer and D. Schnitzer. *Choosing l^p norms in high-dimensional spaces based on hub analysis*. Neurocomputing, 169:281 – 287, 2015. Learning for Visual Semantic Understanding in Big DataESANN 2014Industrial Data Processing and AnalysisSelected papers from the 22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2014)Selected papers from the 11th World Congress on Intelligent Control and Automation (WCICA2014). Available from: http://www.sciencedirect.com/science/article/pii/S0925231215004336, doi:http://dx.doi.org/10.1016/j.neucom.2014.11.084.

[32] D. R. Jones, M. Schonlau, and W. J. Welch. *Efficient Global Optimization of Expensive Black-Box Functions*. Journal of Global Optimization, 13(4):455–492, December 1998. Available from: http://dx.doi.org/10.1023/A:1008306431147, doi:10.1023/A:1008306431147.

[33] V. Torczon and M. W. Trosset. *Using approximations to accelerate engineering design optimization*. Technical report, Institute for Computer Applications in Science and Engineering (ICASE), 1998.

[34] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene. *A novel hybrid sequential design strategy for global surrogate modeling of computer experiments*. SIAM Journal on Scientific Computing, 33(4):1948–1974, 2011. Available from: http://dx.doi.org/10.1137/090761811.

2

Self-Organizing Map Based Adaptive Sampling

⋆⋆⋆

K. Ito, I. Couckuyt, R. d'Ippolito, T. Dhaene.

" Design Space Exploration using Self-Organizing Map Based Adaptive Sampling ".

Published in Applied Soft Computing, DOI 10.1016/j.asoc.2016.02.036, vol. 43, pp. 337 - 346, June 2016.

⋆⋆⋆

In engineering design, a set of potentially competitive designs is conceived in the early part of the design process. The purpose of this research is to help such a process by investigating algorithm that enables approximate identification of a set of inputs of real variables that return desired responses from a function or a computer simulation. We explore sequential or adaptive sampling methods based on Self-Organizing Maps (SOM). The proposed method does not rely on parametrized distributions, and can sample from multi-modal and non-convex distributions. Furthermore, the proposed merit function provides infill characteristics by favoring sampling points that lay farther from existing points. The results indicate that multiple feasible solutions can be efficiently obtained by the new adaptive

sampling algorithm. The iterative use of the SOM in density learning to identify feasible or good designs is our new contribution and it shows a very rapid increase in number of feasible solutions to total number of function evaluation ratio. Application examples to planing hull designs (such as in powerboats and seaplanes) indicate the merits of the feasible region approach to observe trends and design rules. Additionally, the well distributed sampling points of the proposed method played favorable effect in improving the prediction performance of a classification problem learned by Support Vector Machine.

2.1 Introduction

In today's engineering, it is common practice to use computer simulations to understand the behavior of complex systems and optimize their parameters to obtain satisfactory designs before building actual physical prototypes. The goal of an engineer is not only to optimize the systems, but also to understand what makes a design good. Engineers may also need to deal with simulation models that may not represent all the effects necessary to make a right decision. The question - particularly in the early stage of the design process - is often not about finding the best parameter values, but establishing what parameter values would generate a satisfactory design. At a more pragmatic simulation level, engineers often confine the simulation runs to parameter settings for which results are trustworthy [1]. For example, a simulation may crash or its solution may not converge. Such information may not be available until the simulation is running. Furthermore, there are widespread incentives to reduce the number of simulation runs since high fidelity simulations often require a lot of time and computational resources as evidenced in the research of surrogate model based optimization [2–4] and multifidelity optimization [5, 6] methods. Our research is motivated by situations in which efficient identification of diverse designs satisfying minimum specifications and understanding about the problem are more important than finding an accurate single optimal solution.

We propose a new adaptive sampling algorithm that uses a Self-Organizing Map (SOM) [7, 8] to perform importance sampling: Self-Organizing Map Based Adaptive Sampling (SOMBAS). The fundamental idea is an algorithm that learns to select new samples in the region of interest, using the density learning mechanism in SOM. It is similar to Monte Carlo Optimization methods such as Probability Collectives [9] and Cross-Entropy Methods [10] or Subset Optimization methods [11, 12]. However, we do not use parameterized probability density functions to represent solutions. Instead, SOM is used to obtain a set of solutions as represented by the weight vectors in each cell of the map. SOM represents a densely sampled region as a larger area on the map than a sparsely sampled region. Therefore, a weight vector can be considered as being analogous to an instance

of a random vector drawn from a probability density distribution. Furthermore, these vectors are mutated to improve diversity. The training set can be iteratively enriched with, or replaced by, new samples that exhibit desirable responses (or objective values). SOM is retrained in each iteration with the updated training set. To enhance uniform sampling in the region of interest, a new merit function is also proposed. The flowchart of SOMBAS is shown in Fig. 2.1.

The idea of using the weight vectors as candidate solutions can also be seen in [13]. They train SOM from a set of experiments and look for a best candidate solution from the SOM weight vector. The chosen weight vectors are taken as representing an average solution in their respective neighborhoods of good solutions. However, their method does not entail further sampling, and does not have the density learning notion. Our new method substantially extends the idea by making the search process adaptive (i.e. iterates to further explore good solutions).

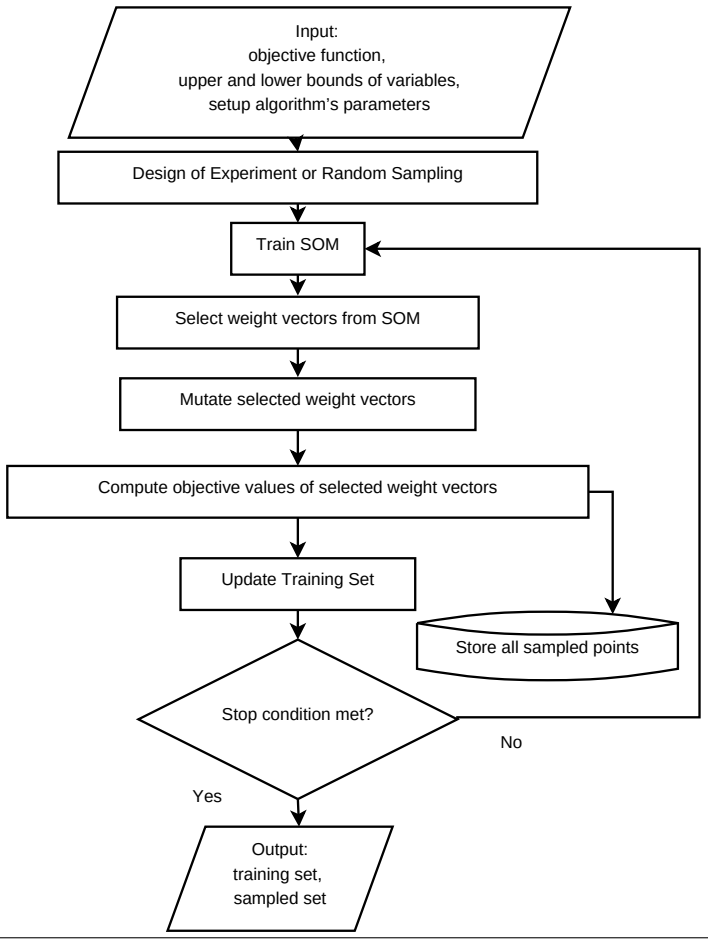
A preliminary version of SOMBAS was presented in [14]. Current work investigates the scalability of SOMBAS to high-dimensional problems and application to a conceptual design example giving extra insights of the parameters on the results.

SOMBAS does not entail any modification of SOM just as in [15]. Therefore, different implementations of SOM or other density learning algorithms can be used instead. It focuses on interesting regions of the input space by modifying the sample densities. While Kita et al. use SOM to do clustering (niching), we use SOM to generate new samples according to the density of its training samples. Their paper shows its advantage in multimodal functions and relative weakness in non-separable unimodal functions. Our algorithm, on the other hand, shows no such tendency.

[16] and [17] have proposed a sequential sampling approach that samples uniformly from the region of interest specified by upper and lower bounds on the output. They extended the Efficient Global Optimization (EGO) [18] and used prediction variances to determine new sampling points that would likely produce an output in the desired range and away from existing samples. The fundamental difference between their work and this paper is that we do not fit interpolating surrogate models that require optimization of the surrogate model hyperparameters. In SOMBAS, no optimality on SOM training is imposed and a user can specify the number of training iterations. Our novelty is in the application of SOM in adaptive sampling scheme. This enables us to sample from distributions without the need to parametrize them. Furthermore, SOM is scalable to high-dimensional input space.

[19] and [20] have proposed algorithms with identical objectives as SOMBAS. They propose diversity measures in their evolutionary algorithms and explicitly optimize for this measures. However, their methods involve multidimensional integrations or matrix inversions that would make the algorithms difficult to apply in high-dimensional problems. In SOMBAS, diversity is kept by a simple merit

Figure 2.1 High level flowchart of SOMBAS.



function that takes the distance to the nearest-neighbor into account and a mutation algorithm.

2.2 Self-Organizing Map Based Adaptive Sampling (SOMBAS)

We use Self-Organizing Maps'(SOM) weight vectors as a representation of a sample distribution. Typically, SOM is represented as a two-dimensional array of cells (be it hexagonal or square shaped). Each of these cells has a weight vector associated with it. In this work, the weight vector is a set of continuous design variables that represents an instance of a possible new solution. We initially assign random

numbers to the vector elements. Then, the weight vectors are learned from a given set of training samples. The trained weight vectors can be considered to be a finite sample representation of the training sample distribution. The weight vectors $\mathbf{w}_j$ are updated using the following equation for a given training sample $\mathbf{t}$.

$$\mathbf{w}_j(k+1) = \mathbf{w}_j(k) + h_{cj}(k)\left[\mathbf{t}(k) - \mathbf{w}_j(k)\right], \tag{2.1}$$

where j is a spatial index that identifies the cells in SOM, k is the training iteration index, h_{cj} is a neighborhood function that depends on the distance between $\mathbf{w}_c$ and $\mathbf{w}_j$ on the map where $\mathbf{w}_c$ is the closest weight vector to the training sample $\mathbf{t}(k)$ in the Euclidean sense. The neighborhood function decreases as the distance between the cells becomes far apart on the map. Thus, given a training sample and the closest matching weight vector, the farther cells on the map receive less influence of the weight update. The shape and magnitude of $h_{cj}(k)$ are changed as k increases in such a way that the second term (the weight update term) on the right-hand side of equation (2.1) reduces the radius and magnitude of influence.

Algorithm 1 shows a high-level description of the Self-Organizing Map Based Adaptive Sampling. In each iteration, the trained Self-Organizing Map (SOM) produces new solution candidates and their corresponding objective values. Weight vector selection is based on these estimated values. Perturbations are applied to these selected vectors, and their objective values are computed, replacing the estimated values. Updating of the training set is performed and a subset of these selected samples are included in the training set to train the SOM of next iteration.

Algorithm 1 SOM BASED ADAPTIVE SAMPLING

```
Generate N samples to create initial training set
while Termination condition not met do
    Train SOM using the normalized training set
    for all cells satisfying SELECTION CONDITION do
        Perturb the weight vectors of the selected cells according to MUTATION
    end for
    Un-normalize the perturbed samples
    Evaluate true output of the perturbed and unperturbed samples
    UPDATE TRAINING SET
end while
```

The probability of a weight vector being selected depends on how close its objective value estimate is to the known smallest value. Note that the objective values in the weight vectors of SOM are estimates. The selection condition is

$$r < \exp\left(\frac{y_{\min} - \hat{y}}{T}\right), \tag{2.2}$$

where $0 \leq r < 1$ is a random number drawn from a uniform distribution, $y_{\min}$ is the smallest output in the training sample, and $\hat{y}$ is the estimated objective value from the weight vector. The temperature $0.01 \leq T \leq 10$ defines how selective the condition is and a smaller value of T results in fewer new samples added to the training data set. The pseudocode of this selection condition is given in Algorithm 2.

Algorithm 2 SELECTION CONDITION

1: Let $y_{\min}$ be the smallest output in the training set X, $\hat{y}$ be output from a cell weight vector, and T be the selectivity parameter (or Temperature)
2: Generate a uniform random number $0 \leq r < 1$ and check the following:
3: **if** $r < \exp\left(\frac{y_{\min}-\hat{y}}{T}\right)$ **then**
4: Corresponding weight vector is selected
5: **end if**

We consider a case in which we seek to minimize an objective value y below certain threshold L. Below this threshold, diversity of solutions is sought. In this paper, we will call such a search as *feasible region identification* or *feasible region search*. One idea is to use a merit function similar to those described in [21]. One could give a better chance of being selected to points (i.e. cell weight vector) that are distant from existing training samples regardless of y value. To achieve this, we propose the following formula for the merit function.

$$F = \max(L, y) - \rho \min(\|\mathbf{s} - \mathbf{t}_i\|_2),\ i = 1, 2, \cdots, N \tag{2.3}$$

where $\mathbf{s}$ is the input vector for which F needs to be minimized, $\mathbf{t}_i$ is a set of target samples from which minimum distance to the input vector $\mathbf{s}$ is calculated, N is the number of such target vectors, and ρ is a weight constant. Unlike Torczon's merit function, our merit function incorporates a "truncation" value L below which only the separation from other target vectors $\mathbf{t}_i$ matters. To minimize this merit function, one needs $y < L$ and maximize the distance to the nearest target vector $\min(\|\mathbf{s} - \mathbf{t}_i\|_2)$. In our case, target vectors are the training set and the input vector $\mathbf{s}$ is the selected weight vector from SOM. The algorithm to replace the output with this merit function is described in Algorithm 3. If $\hat{y}$ is greater than the threshold L, both $\hat{y}$ and the new weight vector's distance from the training set are taken into account. If $\hat{y}$ is less than L, then the distance to the nearest training vector is the only term affecting the objective value and smaller F is obtained when the weight vector's distance to the nearest neighbor is larger. The ρ in equation 2.3 is a positive weighing constant

Mutation, as described in Algorithm 4, is applied to the selected weight vectors. We use the weight vectors as the centers of multivariate Gaussian distributions. The covariance matrix is obtained from the selected weight vectors. We use

Algorithm 3 MERIT FUNCTION

1: Let L denote the value below which objective or output y is considered to be "good enough", $\mathbf{s}$ denote a weight vector $(\mathbf{x}^T, \hat{y})$ from SOM, and $\mathbf{t}_{i=1,2,\ldots,N}$ denote the training samples
2: **if** Trunc is specified **then**
3: Normalize L ($\mathbf{s}$, and $\mathbf{t}_i$ are already normalized)
4: $\hat{y} \leftarrow \max(L, \hat{y}) - \rho \min(\|\mathbf{s} - \mathbf{t}_i\|_2)$
5: Normalize $\hat{y}$
6: **end if**
7: Use this $\hat{y}$ in SELECTION CONDITION

an idea similar to CMA-ES [22] to update the covariance matrices. The covariance matrix in the current iteration is combined with the covariance matrix computed in the previous iteration: $0.2C + 0.8C_{old}$. This is to avoid adapting too quickly to a local minimum. On top of that, we multiply a factor which is different whether the previous iteration produced a new minimum or not. If the previous iteration achieved a new minimum, we apply an expansion factor F_e, to which we assign a real value larger than 1. On the other hand, if the previous iteration did not produce a new minimum, we multiply a contraction factor F_c, to which we assign a real value between 0 and 1. The covariance matrix is the same for all the selected weight vectors. Each weight vector is perturbed by sampling from the multivariate Gaussian distribution. Mutation is very important to avoid premature convergence in SOMBAS.

Algorithm 4 MUTATION

1: Let F_c be contraction factor and F_e be expansion factor
2: Let P_m be mutation probability
3: Given training samples, compute covariance matrix C, and let the covariance matrix from previous iteration be C_{old}
4: **if** current $y_{\min} <$ previous $y_{\min}$ **then**
5: $C = F_e\,(0.2C + 0.8C_{\text{old}})$
6: **else**
7: $C = F_c\,(0.2C + 0.8C_{\text{old}})$
8: **end if**
9: For each selected sample, perturb it by sampling from multivariate normal distribution with center at the selected sample with covariance C.
10: Replace a parameter in the selected vectors with the mutated one at probability of P_m

After the perturbation of new samples, the training set is updated. Algorithm 5 and Algorithm 6 are two such methods. Algorithm 5 has a faster convergence but

Algorithm 5 UPDATE TRAINING SET 1

Add the perturbed samples to the training set
if Training set sample size larger than maximum sample size **then**
    Sort the training set with respect to output value
    Remove the worst samples to make the training sample size equal to maximum sample size
end if

Algorithm 6 UPDATE TRAINING SET 2

for all perturbed weight vectors' response y_p **do**
    Randomly pick one of the training sample, and obtain its response y_t
    Obtain d_p, the nearest neighbor distance of perturbed sample to sampled points thus far and d_t the nearest neighbor distance of the training sample to sampled points thus far
    if $\max(L, y_p) = \max(L, y_t)$ and $d_p > d_t$ **then**
        Replace the training sample with the perturbed weight vector
    else if $y_p < y_t$ **then**
        Replace the training sample with the perturbed weight vector
    end if
end for

is more prone to lose diversity in the training set prematurely compared to Algorithm 6. In the latter method, if $\max(L, y)$ of the new perturbed sample and that of the randomly selected training sample are the same, the replacement of the selected training sample takes place only if the new perturbed sample has a larger distance to its nearest neighbor than the distance of the training sample to its nearest neighbor. Otherwise, the new perturbed sample replaces the training sample when the new sample has a smaller objective value. The nearest neighbors are searched among all the sampled points. In the next section, we will use Algorithm 6.

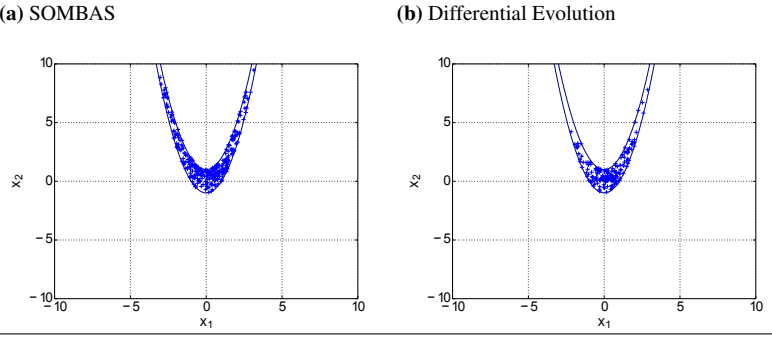
2.3 Experiments

2.3.1 Feasible Region Identification

In this subsection, we consider a constraint satisfaction problem in which there is a constraint on the objective (to be below a certain threshold value) but one would like to know what inputs would satisfy this condition. Ideally, one would like to have as much variety as possible in the collection of inputs that we obtain as solutions.

The analysis and evaluation of the results are not straightforward because we

Figure 2.2 Rosenbrock function: sample distribution satisfying objective condition $f \leq 100$.



do not have established performance measures. [23] and [19] show some possibilities, but they do not scale well to high-dimensional problems or can cause numerical problems in matrix inversion. We first resort to visual cues, and then propose some performance measures.

We use the two dimensional Rosenbrock function for non-convex region identification, the two dimensional Rastrigin function for discrete region identification, and the Hollow Beam [24, p. 35] problem with two design variables for feasible region with sharp corners. Then we look into 30 and 100 dimensional Rosenbrock and Rastrigin functions to see the scalability of SOMBAS to high-dimensional problems.

SOMBAS is compared with Differential Evolution (DE). DE learns from the distribution of its population to choose the next sampling points. It does not rely on any parameterized model of the distribution, but the difference vectors adapt to the objective function landscape. [25, pp. 44 - 47] called this property *contour matching* and described it as one of the advantages of DE. Furthermore, it is not restricted to low dimensional problems as in [19]. Therefore, DE is suited for the identification of the input regions of the functions described above. The purpose of the comparison is to elucidate differences between the two algorithms that have similar characteristics, but serve different purposes, namely optimization and feasible region identification.

In Fig. 2.2 through Fig. 2.4, we visualized different types of feasible domains and the sampling (shown in cross) inside them. The contour plots show the boundaries of the domains.

SOMBAS was able to produce a fairly uniform infill of samples in the 2D input domain for the functions tested. Since DE also has distribution learning characteristics, it did very well in the feasible domain infill task.

To base feasible region identification on a more statistical ground, we repeated

Figure 2.3 Rastrigin function: sample distribution satisfying objective condition $f \leq 10$.

(a) SOMBAS **(b)** Differential Evolution

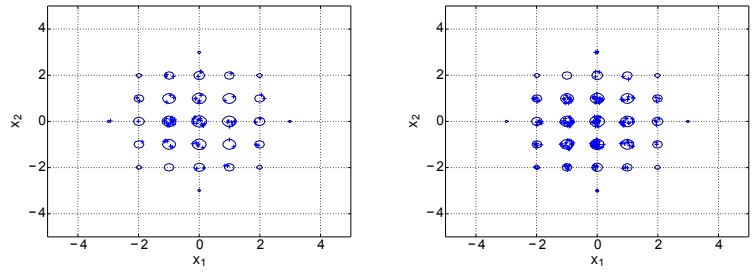


Figure 2.4 Hollow Beam function: sample distribution satisfying constraints.

(a) SOMBAS **(b)** Differential Evolution

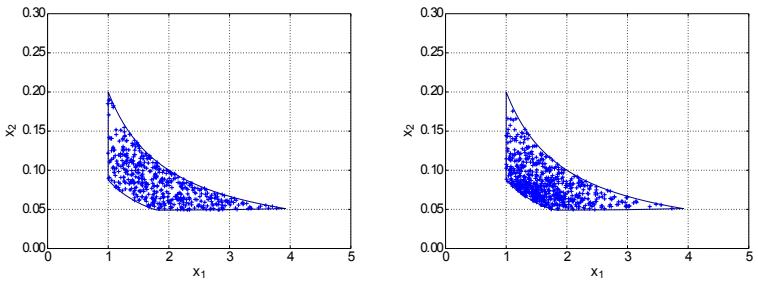


Table 2.1 Average Nearest Neighbor Distances and Space Filling Measures of Sampled Points by SOMBAS and DE.

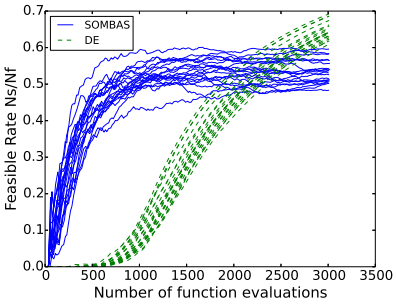
Function	Algorithm	$\tilde{\tilde{d}}$	$\tilde{\sigma}_d$	$\tilde{N}_s$	$\tilde{N}_f$	$\frac{\tilde{N}_s}{\tilde{N}_f}$	$\tilde{\tilde{d}} \times \tilde{N}_s$	$\frac{\tilde{l}}{\tilde{N}_f}$
Rosenbrock	SOMBAS	1.14	0.84	207	786	**0.27**	**227**	0.30
	DE	1.37	1.11	158	689	0.23	213	**0.31**
Rastrigin	SOMBAS	0.38	0.28	146	1136	0.13	53.3	0.05
	DE	0.41	0.31	180	989	**0.18**	**70.4**	**0.07**
Hollow Beam	SOMBAS	0.17	0.11	138	342	**0.40**	**23.1**	**0.07**
	DE	0.20	0.13	112	395	0.28	21.8	0.06

the sampling task for each of the three equations 20 times. We terminated the sampling when all the training sample achieved the objective value $f \leq L$, where $L = 100$ for Rosenbrock, $L = 10$ for Rastrigin, and $L = 0$ for Hollow Beam. Table 2.1 shows the results. Here, $\tilde{\tilde{d}}$ is the average distance of nearest neighbors. The tilde on top of the symbol signifies averaging and the nearest neighbor has two tildes corresponding to the average in the feasible domain and an average of 20 runs. The $\tilde{N}_f$ is the average number of function evaluations, $\tilde{N}_s$ is the average number of samples in the feasible domain, $\tilde{\sigma}_d$ is the average standard deviation of distances to the nearest neighbors. We also define the feasible rate $\tilde{N}_s/\tilde{N}_f$, coverage length $\tilde{l} = \tilde{\tilde{d}} \times \tilde{N}_s$ and coverage rate $\tilde{l}/\tilde{N}_f$. The feasible rate gives the ratio of the number of feasible samples meeting the truncation value L to the total number of function evaluations. Higher the better. The coverage length gives the efficiency of infill. Higher the better. However, since $\tilde{\tilde{d}}$ is scale dependent, the relative importance of having small N_s but large $\tilde{\tilde{d}}$ or large N_s but small $\tilde{\tilde{d}}$ will be different from problem to problem. The coverage length is meaningful only when we compare different methods on the same feasible domain identification problem. The coverage rate is defined as the coverage length per function evaluation. The larger the value the better, and it is also scale-dependent. For these two dimensional examples, DE and SOMBAS did not show marked difference in performance. SOMBAS showed some advantage in feasible rate for Hollow Beam and DE showed some advantage in Rastrigin in terms of coverage length.

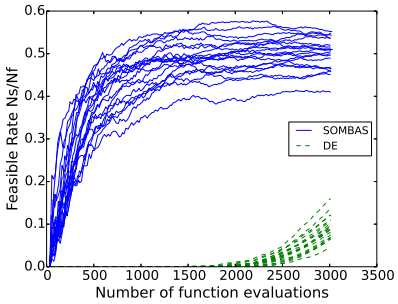
In higher dimensional problems, it is often the case that there is no feasible solution in the beginning. The algorithm has to search and fill the feasible region. Fig. 2.5 shows the histories of feasible rates with respect to number of function evaluations N_f. The criteria of feasibility were set to $f \leq 20 \times D$ for Rastrigin function and $f \leq 5000 \times D$ for Rosenbrock function. Both DE and SOMBAS were run 20 times. SOMBAS shows very rapid gains in the feasible rate N_s/N_f

Figure 2.5 Evolution of Feasible Rate N_s/N_f of SOMBAS and DE on test functions.

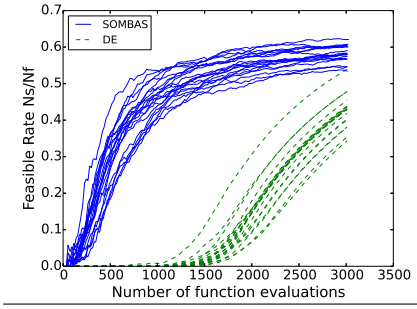
(a) Rastrigin 30 dimensions, feasible solution as $f \leq 600$



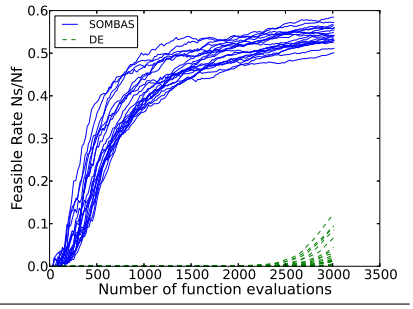
(b) Rastrigin 100 dimensions, feasible solution as $f \leq 2000$



(c) Rosenbrock 30 dimensions, feasible solution as $f \leq 150000$



(d) Rosenbrock 100 dimensions, feasible solution as $f \leq 500000$



compared to DE. Moreover, with the feasibility condition set above, the number of function evaluations to reach a given feasible rate (below 0.6) is about the same for both 30-dimensional functions and 100-dimensional functions. However, this is not the case with DE. For DE, the number of function evaluations necessary to reach a given feasible rate seems to be proportional to the number of dimensions. The wiggles on the evolution curves of SOMBAS show that some portion of the sampled points is infeasible and its proportion varies from iteration to iteration and eventually stagnates around certain values of feasible rate N_s/N_f. On the other hand, DE is an optimization algorithm and it keeps searching for lower objective values. Thus, the wiggles (not visible in the plot) disappear once the populations are well within the feasible domain. This, in turn, indicates that for DE the feasible rate N_s/N_f can eventually reach higher rate than SOMBAS as the number of function evaluations is increased.

2.3.2 Engineering Application

In this subsection, we present an application example of SOMBAS using a simulation code for stability analysis of planing crafts [26–28]. This example uses SOMBAS in a very simplified simulation based design task. A boat or a seaplane is in a planing condition when it is traveling on water and most of the lifting force of the water comes from hydrodynamic pressure exerted on its hull, buoyancy playing minor to negligible role in its steady state equilibrium. In such condition, the craft may be subject to a dangerous longitudinal oscillation mode called porpoising. Porpoising is a coupled oscillatory motion between heaving and pitching that can manifest when seaplanes and power boats are traveling on water at planing speed. This motion, once manifested, is often unstable and can pose a significant risk to the safe operation of these vehicles.

We numerically simulated a 1/3 scale towed tank model as reported in [26]. The model was a catamaran, so we employed the half body representation for simplicity. The model was parameterized as a two or seven variable design problem. The dynamic stability was computed using small perturbation analysis as presented in Faltinsen's book [27, chap. 9]. The two degrees of freedom dynamics - pitching and heaving - were thus represented as a couple of second order differential equations with respect to time. Then, we use a state-space formulation to represent this dynamical system as a system of first order differential equations,

$$\dot{x} = Kx, \tag{2.4}$$

where $x = [\dot{\eta}_3, \dot{\eta}_5, \eta_3, \eta_5]^T$ and η_3 is a displacement upward and η_5 is a pitch-up rotation. The dot above denotes time derivative. The K is a 4×4 matrix and contains information about the inertia, damping, and force reactions. By computing the eigenvalues of this matrix we obtain the stability of this dynamical system. If one of the eigenvalues has positive real part, it means the system has an unstable oscillation mode.

In this example, we try to seek a stable design at a given planing speed by varying the design variables. For the two-design-variable case, we use the longitudinal distance of CG along the keel line l_{cg} measured from the step or transom, and vertical distance of CG from the keel line vcg. For the seven-design-variable case we use beam length B, deadrise angle β (in degrees), pitching moment of inertia I_{55},thrust line distance f from CG (positive when pitch-up moment results) and thrust line angle with respect to keel line (positive upwards) ϵ. Fig. 2.6 shows a diagram describing the design variables except the inertial variable I_{55}. For each design, a trim position needs to be calculated by an iterative root finding process and then Ks are determined via semi-empirical equations based on the trim position. Thus, the mapping from design variables to maximum eigenvalues is non-linear and non-analytical involving loops and branching in the algorithm of the function. This is a typical situation in engineering applications.

Figure 2.6 Diagram of planing hull cut-out.

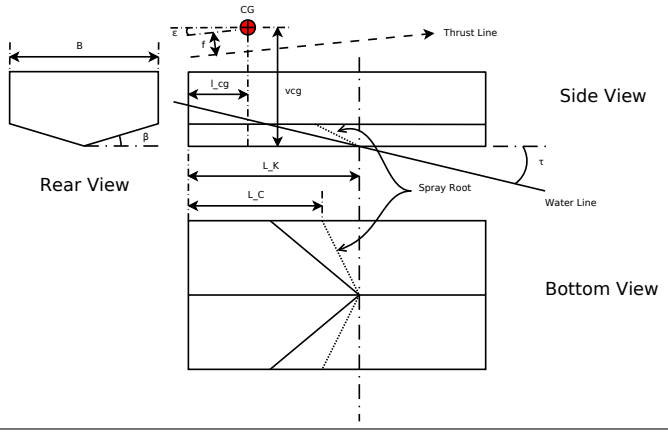


Figure 2.7 Contour and scatter plot of the real part of the largest eigenvalues of oscillation modes (negative values indicate stable modes) with respect to l_{cg} and vcg, both non-dimensionalized with respect to B.

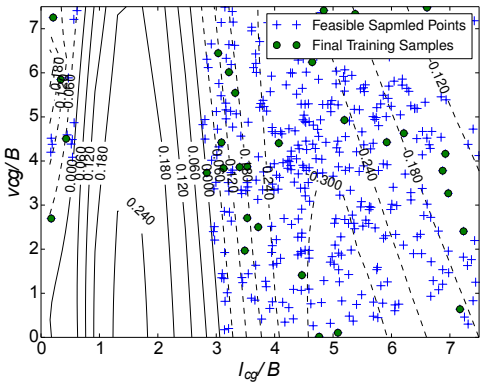


Fig. 2.7 shows a contour plot of the largest real part of non-dimensionalized eigenvalues with respect to l_{cg} and vcg along with sampled points by SOMBAS in two-design-variable case. SOMBAS was set to search feasible designs by setting $L = 0$ in our merit function. That is, if the largest eigenvalue was less than 0, the design was considered stable and thus satisfactory. The sampled points that satisfy $L < 0$ are shown along with the final location of the training samples for SOM. One can see that there is an interval of l_{cg} that produces unstable designs. There is also a large domain that is stable. A small portion of the design space near the transom or very small value of l_{cg} generates stable designs, and most seaplanes have this configuration to facilitate the pitch up at the moment of take off.

Fig. 2.8a shows a scatter plot matrix of the seven design variable case. The matrix shows a series of two-dimensional projection plots of the seven design variables. The dots indicate that sampled designs satisfy the stability condition set by the value given in L. In other words, any blank region on the map suggests that no design satisfying the stability criteria has been sampled, which implies that such region is an unfeasible domain. The lower triangular cells show the absolute values of correlation coefficients calculated from the sample points satisfying the stability criteria $L = 0$. Again, it clearly shows the unstable "band" for l_{cg} at the top row of the scatter plot matrix. Other parameters do not show clear unfeasible regions. Further restriction was applied by setting $L = -0.3$ and the results are shown in Fig. 2.8b. It shows some interesting trend. For example, vcg tends to lower value as the eigenvalue becomes more negative. On the other hand, the beam length B tends to larger values as the eigenvalue becomes more negative. In the current setup, the deadrise angle β shows a positive correlation with f. Likewise, l_{cg} and B show a positive correlation.

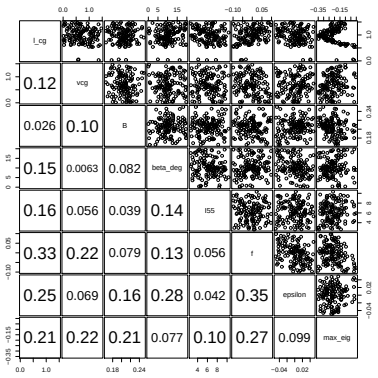
2.3.3 Machine Learning Application

We continue with the above example, but this time, would like to fit a classifier on top of the sampled points. If a classifier is constructed, a new point's feasibility can be predicted without evaluating the original (possibly expensive) function. The planing stability example in the previous subsection can be considered as a binary classification problem once enough number of design points are sampled. We employed Support Vector Machine (SVM) [29, 30] to learn the classification problem from the points sampled by SOMBAS, DE, and random sampling. The SVM can be used to learn the boundary separating the stable and unstable design from a given set of sampled points and can give a prediction whether a new instance of design is stable or not.

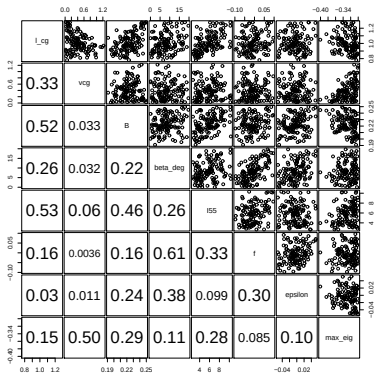
We run the seven-design-variable case searching for solutions with maximum eigenvalues of oscillation modes less than -0.3 for a given number of simulation budget, namely 1000, 2000, and 4000. Let us call the designs satisfying this

Figure 2.8 Scatter Matrix showing distribution of feasible designs.

(a) Maximum eigenvalues of oscillation modes less than 0, ($N_f = 170, N_s = 132$)



(b) Maximum eigenvalues of oscillation modes less than -0.3, ($N_f = 442, N_s = 123$)



condition as stable designs. A test set with 2000 designs randomly sampled from the domain is prepared to evaluate the performance of the SVM classifiers. For performance measure, we use the F_1 score, which is defined as follows.

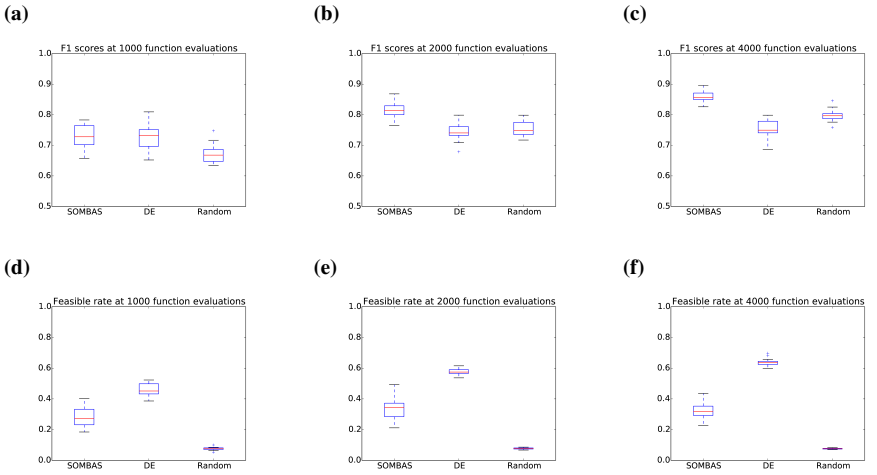
$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}, \tag{2.5}$$

where, in our case, the precision P is the proportion of stable designs (according to the simulation) among all designs predicted as stable (by a classifier) in the test set, and the recall R is the proportion of designs correctly predicted as stable (by a classifier) among all the stable designs in the test set (according to the simulation). In this measure, too liberal (e.g., $P \simeq 0$, $R \simeq 1$) or too conservative (e.g., $P \simeq 1$, $R \simeq 0$) classifiers get low score values. $F_1 = 1$ means perfect prediction.

In the top row of Fig. 2.9, we have shown the distribution of F_1 obtained by fitting Support Vector Machines (SVM) to the sampled points from SOMBAS, DE, and random sampling. The box plots of F_1 was computed from 20 independent runs of each of the three sampling methods. The figures 2.9a, 2.9b, and 2.9c show the results of different function evaluation budgets of the planing craft simulation, 1000, 2000, and 4000 respectively. In the bottom row of Fig. 2.9, we have the box plots of feasibility rates N_s/N_f of the three sampling methods at corresponding sampling budgets.

In Fig. 2.9, we observe two trends. The first trend is that the F_1 scores for SVM on SOMBAS and random samples increases as sampling budget increases from 1000 to 4000 while the improvement of SVM on DE is rather small if any and the first and the third quartile of the F_1 scores remain between 0.7 and 0.8. The second trend is that, contrary to the F_1 scores, the feasible rates N_s/N_f for DE

Figure 2.9 Planing stability prediction performance of Support Vector Machine using samples from SOMBAS and DE. Box plots show the distributions of 20 independent runs at budgets of 1000, 2000, and 4000 function evaluations.



increases from around 0.5 to around 0.7 while those for SOMBAS increases very little staying around 0.3 and those for random sampling stays practically constant at 0.07.

These two trends are due to the fact that DE is minimum seeking and SOMBAS and random sampling are space filling. Since SOMBAS searches and fills out the feasible region, the feasible rates reached higher values than those of random sampling. On the other hand, being space filling in the feasible domain, SOMBAS inevitably keeps sampling also from the infeasible domain, because when the mutation happens one does not know if the perturbed points will lie inside the feasible domain. This causes the stagnation of the feasible rate N_s/N_f, but it is beneficial for a classification model as evidenced by superior F_1 scores in Fig. 2.9b and in Fig. 2.9c. In principle, the F_1 score of SVM using random sampling should eventually catch up with that of SOMBAS as the number of sampled points is increased. On the other hand, DE (or any optimization method) keeps searching for the smaller response and the sampling concentrates around the points with minimal responses. Since this trend does not help in defining the boundary between feasible and infeasible, the F_1 score stagnates after a certain number of function evaluations, but the feasible rate N_s/N_f will keep increasing if the minimum is in the interior of the feasible domain.

Table 2.2 to Table 2.4 summarize the results of the significance of differences in F_1 score and feasible rate N_s/N_f distributions among different sampling methods and sampling budgets in Fig. 2.9. Wilcoxon Rank Sum Test was used. In

Table 2.2 Hypothesis test of shift in F_1 score distributions in Fig. 2.9 among different sampling methods (p-values shown in the bracket)

Sampling Methods	1000 eval.	2000 eval.	4000 eval.
SOMBAS vs. DE	Null(0.841)	Alt.$(3.407e-07)$	Alt.$(1.451e-11)$
SOMBAS vs. Random	Alt.$(3.926e-05)$	Alt.$(1.281e-06)$	Alt.$(1.233e-07)$
DE vs. Random	Alt.$(1.831e-05)$	Null(0.2852)	Alt.$(1.917e-05)$

Table 2.3 Hypothesis test of shift in F_1 score distributions in Fig. 2.9 among different sampling budgets (p-values shown in the bracket)

No. Function Evaluations	SOMBAS	DE	Random
1000 vs. 2000	Alt.$(1.407e-09)$	Null(0.1555)	Alt.$(2.952e-07)$
2000 vs. 4000	Alt.$(1.061e-07)$	Null(0.3104)	Alt.$(1.407e-05)$
1000 vs. 4000	Alt.$(1.451e-11)$	Alt.(0.03264)	Alt.$(6.786e-08)$

this test, the parametric distributions of two random variables are not assumed and sample size of the two variables can be different. It tests whether the distribution of the two random variables, say X and Y, are the same after a translation of size k. That is,

$$P(X < x) = P(Y < x + k) \tag{2.6}$$

for all x. The null hypothesis is $k = 0$ and the alternative hypothesis is $k \neq 0$.

The hypothesis tests in Table 2.2 support (at 0.05 significance level) that the shifts in distributions of F_1 scores exist between SVM obtained from SOMBAS and DE in Fig. 2.9b and in Fig. 2.9c. The improvement of F_1 scores of random sampling based SVM relative to F_1 scores of DE based SVM (from Fig. 2.9a to Fig. 2.9c) is also supported by the significance test.

Table 2.3 summarizes whether F_1 score distributions in Fig. 2.9 differ between sampling budgets 1000, 2000, and 4000 function evaluations. For SOMBAS and random sampling, the shifts in the distributions were detected. On the other hand, the null hypothesis was not rejected for DE when the budget was increased from 1000 to 2000 and from 2000 to 4000, although between 1000 and 4000 the alternative hypothesis of $k \neq 0$ was supported. This supports the observation that the increase in F_1 score of SVM using samples from DE is not as rapid as those of the remaining two sampling methods.

Table 2.4 shows whether the shift in the distribution of the feasible rate N_s/N_f exists between different sampling budgets. For SOMBAS null hypothesis was kept between 2000 and 4000 function evaluations. For random sampling, no shifts in the distributions were detected among the three sampling budgets. On the other hand, shifts were supported for all the three comparisons for DE. This supports

Table 2.4 Hypothesis test of shift in the feasible rate N_s/N_f distributions in Fig. 2.9 among different sampling budgets (p-values shown in the bracket)

No. Function Evaluations	SOMBAS	DE	Random
1000 vs. 2000	Alt.(0.01674)	Alt.$(6.767e-08)$	Null(0.6259)
2000 vs. 4000	Null(0.6017)	Alt.$(1.427e-07)$	Null(0.1675)
1000 vs. 4000	Alt.(0.04018)	Alt.$(6.767e-08)$	Null(0.8497)

along with Fig. 2.9 that DE's feasible rate N_s/N_f kept increasing when the sampling budget increased. On the other hand, the feasible rate for SOMBAS stagnated and that of the random sample showed no shift in distribution (which was expected from the law of large numbers).

2.4 Conclusions

SOMBAS is able to select samples in the design space below a given threshold value, and in addition, it is able to do so in a space-filling way. Our approach to feasible region identification is different from binary classification methods in Machine Learning. Classification methods require both positive and negative samples from the outset of the learning iteration. SOMBAS, on the other hand, will search for feasible regions, even if all of the initial training samples were unfeasible.

SOMBAS' efficient acquisition of feasible solutions in higher dimensions, namely for the 30 and 100-dimensional Rastrigin function and Rosenbrock function, was shown to be superior to DE. It can be argued that feasible region identification becomes identical to optimization when the feasible region becomes infinitesimally small. For example, we could set $f \leq 10^{-6}$ as the feasible region in the 30-dimensional Rastrigin function. In this case, DE would be a better choice. Further research is needed to understand the relationship between accurately finding an optimum point and efficiently identifying a feasible region.

In the engineering example, we identified input values that generate satisfactory designs. By looking at multiple solutions, it enabled the extraction of design knowledge regarding how the design parameters interact under certain stability criteria. This is a significant advantage with respect to standard optimization techniques.

In the application SOMBAS in Machine Learning example, in which Support Vector Machine was used to learn a binary classification model from the sampled data, the accuracy of prediction improved as the number of samples increased and the number of feasible samples for a given function evaluation budget was substantially higher than the random sampling. On the other hand, DE achieved a steady

increase in the proportion of number feasible samples (feasible rate N_s/N_f) while the accuracy of prediction (F_1 score) stagnated as the number of data increased. It would be beneficial to investigate the merit of applying SOMBAS to different Machine Learning tasks and methods.

2.5 Acknowledgments

Keiichi Ito has been funded by the Institute for the Promotion of Innovation through Science and Technology (IWT) through the Baekeland Mandate program. Ivo Couckuyt is a post-doctoral research fellow of the Research Foundation Flanders (FWO). This research has also been funded by the Interuniversity Attraction Poles Programme BESTCOM initiated by the Belgian Science Policy Office.

Appendix

2.A Test Functions

In the following, we describe the test functions used in this paper. The θ^* denotes the globally optimum solution vector, and $f(\theta^*)$ its response value. The D denotes the number of dimensions. The upper and lower bounds of θ s are by default $-10 \leq \theta_j \leq 10$ where $\theta = [\theta_0, \theta_1, \ldots, \theta_{D-1}]^T$.

Rosenbrock

$$f(\theta) = \sum_{j=0}^{D-2} \big(100(\theta_{j+1} - \theta_j^2)^2 + (\theta_j - 1)^2\big), \tag{2.7}$$
$$j = 0, 1, \ldots, D-1, \quad D > 1,$$
$$f(\theta^*) = 0, \quad \theta_j^* = 1.$$

Rastrigin

$$f(\theta) = \sum_{j=0}^{D-1} \big(\theta_j^2 - 10\cos(2\pi\theta_j) + 10\big), \tag{2.8}$$
$$j = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_j^* = 0.$$

Hollow Beam

Let

$$\begin{aligned} w &= 88.9\theta_0\theta_1 - 17.8, \\ g1 &= 0.0885 - \theta_0\theta_1, \\ g2 &= 0.994 - \theta_0, \\ g3 &= 0.05 - \theta_1. \end{aligned}$$

If $g1 > 0$ or $g2 > 0$ or $g3 > 0$,

$$f(\theta) = \max(0, g1) + \max(0, g2) + \max(0, g3), \tag{2.9}$$

else,

$$f(\theta) = w. \tag{2.10}$$

Table 2.B.1 Parameters setups for DE for the three test functions in Table 2.1

Function	NP	CR	F
Rosenbrock	45	0.9	0.5
Rastrigin	35	0.2	0.8
Hollow Beam	30	0.65	0.75

Table 2.B.2 Parameters setups for SOMBAS for the three test functions in Table 2.1

Function	L	ρ	NT	SOM size	T	$P_{mutation}$	F_e	F_c
Rosenbrock	100	2.0	45	6×6	0.5	1.0	2.0	0.5
Rastrigin	10	2.0	35	6×6	0.5	1.0	2.0	0.5
Hollow Beam	0	2.0	30	6×6	0.5	1.0	2.0	0.5

For this problem, the objective is to find $\theta = [\theta_1, \theta_2]^T$ such that $f(\theta) < 0$. The lower and upper bounds of θs for this problem are $0 < \theta_0 \leq 5$, $0 < \theta_1 \leq 0.3$.

2.B Parameter Setups

In the following tables, column name NP signifies number of population in DE and NT signifies number of training samples for Self-Organizing Maps in SOMBAS. F and CR are scale factor and cross-over probability as typically defined for the classical DE [25, pp. 38,39]. The number of iteration for the Self-Organizing Map was set between 10 and 40 with no appreciable effect on the results whether one set the number to 10 or 40. The parameter setups of DE and SOMBAS for the feasible region identification in Table 2.1 are given in Table 2.B.1 and Table 2.B.2 respectively.

For SOMBAS, the parameters were set up such that the number of feasible solutions N_f would be more or less the same as those of DE.

Fig. 2.5 was created with with the setups described in Table 2.B.3 for DE and Table 2.B.4 for SOMBAS. In the engineering example of planing craft stability, we setup SOMBAS as $L = 0$, or $-0.3, \rho = 0.1, NT = 36$, SOM size $= 6 \times 6, T = 1, P_{mutation} = 1, F_e = 2.0, F_c = 0.75$.

The subsequent results of SVM classification problem were obtained using SVC function in "scikit-learn" module [29] with Radial Basis Function (RBF) kernel, penalty parameter $C = 10000$ and kernel coefficient $\gamma = 0.5$. The DE and SOMBAS parameters for the Machine Learning problem shown in Fig. 2.9 are given in Table 2.B.5 and Table 2.B.6 respectively. The random sampling was done using a uniform random number generator in Python.

Table 2.B.3 Parameters setups for DE for Fig. 2.5

Function	NP	CR	F
Rosenbrock	35	0.9	0.5
Rastrigin	35	0.2	0.8

Table 2.B.4 Parameters setups for SOMBAS for Fig. 2.5

Function	L	ρ	NT	SOM size	T	$P_{mutation}$	F_e	F_c
Rosenbrock	$5000 \times D$	1.0	35	6×6	1	1.0	1.2	0.15
Rastrigin	$20 \times D$	1.0	35	6×6	1	1.0	2.0	0.15

Table 2.B.5 Parameters setups for DE for Fig. 2.9

Function	NP	CR	F
Planing Craft	40	1.0	0.75

Table 2.B.6 Parameters setups for SOMBAS for Fig. 2.9

Function	L	ρ	NT	SOM size	T	$P_{mutation}$	F_e	F_c
Planing Craft	-0.3	0.2	40	5×5	4	1.0	1.2	1.0

References

[1] Y. Tenne. *A computational intelligence algorithm for simulation-driven optimization problems*. Advances in Engineering Software, 47:62 – 71, 2012.

[2] A. J. Keane and P. B. Nair. *Computational Approach for Aerospace Design*. John Wiley & Sons, 2005.

[3] I. Couckuyt, F. D. Turck, T. Dhaene, and D. Gorissen. *Automatic Surrogate Model Type Selection During the Optimization of Expensive Black-Box Problems*. In Proceedings of the 2011 Winter Simulation Conference, pages 4274 – 4284, 2011.

[4] B. Liu, Q. Zhang, and G. G. E. Gielen. *A Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Medium Scale Expensive Optimization Problems*. IEEE Transaction on Evolutionary Computation, 18(2):180 – 192, April 2014. doi:10.1109/TEVC.2013.2248012.

[5] S. Koziel and L. Leifsson. *Surrogate-Based Aerodynamic Shape Optimization by Variable-Resolution Models*. AIAA Journal, 51(1):94 – 106, 2013. doi:10.2514/1.J051583.

[6] N. Courrier, P.-A. Boucard, and B. Soulier. *The use of partially converged simulations in building surrogate models*. Advances in Engineering Software, 67:186–197, 2014.

[7] J. Kangas and T. Kohonen. *Developments and applications of the self-organizing map and related algorithms*. Mathematics and Computers in Simulation, 41:3 – 12, 1996.

[8] T. Kohonen. *Essentials of the Self-organizing Map*. Neural Networks, 37:52–65, January 2013. Available from: http://dx.doi.org/10.1016/j.neunet.2012.09.018, doi:10.1016/j.neunet.2012.09.018.

[9] D. Rajnarayan, D. Wolpert, and I. Kroo. *Optimization Under Uncertainty Using Probability Collectives*. In 11th AIAA/ISSMO Multidisciplinary Analysis and Optimisation Conference, Portsmouth, Virginia, 6 - 8 September 2006.

[10] D. P. Kroese, S. Porotsky, and R. Y. Rubinstein. *The Cross-Entropy Method for Continuous Multi-Extremal Optimization*. Methodology and Computing in Applied Probability, 8:383 – 407, 2006.

[11] A. A. Taflanidis and J. L. Beck. *An efficient framework for optimal robust stochastic system design using stochastic simulation*. Computer Methods in Applied Mechanics and Engineering, 198(1):88 – 101,

2008. Computational Methods in Optimization Considering Uncertainties. Available from: http://www.sciencedirect.com/science/article/pii/S0045782508001461, doi:http://dx.doi.org/10.1016/j.cma.2008.03.029.

[12] A. Taflanidis and J. Beck. *Stochastic Subset Optimization for optimal reliability problems*. Probabilistic Engineering Mechanics, 23(2 - 3):324 – 338, 2008. 5th International Conference on Computational Stochastic Mechanics. Available from: http://www.sciencedirect.com/science/article/pii/S0266892007000501, doi:http://dx.doi.org/10.1016/j.probengmech.2007.12.011.

[13] M. Liukkonen, E. Havia, H. Leinonen, and Y. Hiltunen. *Quality-oriented optimization of wave soldering process by using self-organizing maps*. Applied Soft Computing, 11(1):214 – 220, 2011. Available from: http://www.sciencedirect.com/science/article/pii/S1568494609002245, doi:http://dx.doi.org/10.1016/j.asoc.2009.11.011.

[14] K. Ito, T. Dhaene, N. E. Masri, R. d'Ippolito, and J. V. de Peer. *Self-Organizing Map Based Adaptive Sampling*. In Proceedings of 5th International Conference on Experiments/Process/System Modeling/Simulation/Optimization (5th IC-EpsMsO), volume II, pages 504 – 513, Athens, Greece, July 3 - 6 2013. ISBN:978-618-80527-2-7 or 978-618-80527-0-3.

[15] E. Kita, S. Kan, and Z. Fei. *Investigation of self-organizing map for genetic algorithm*. Advances in Engineering Software, 41:148 – 153, 2010.

[16] I. Couckuyt, J. Aernouts, D. Deschrijver, F. Turck, and T. Dhaene. *Identification of quasi-optimal regions in the design space using surrogate modeling*. Engineering with Computers, 29(2):127–138, 2013. Available from: http://dx.doi.org/10.1007/s00366-011-0249-3, doi:10.1007/s00366-011-0249-3.

[17] D. Gorissen, K. Crombecq, I. Couckuyt, T. Dhaene, and P. Demeester. *A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design*. Journal of Machine Learning Research, 11:2051 – 2055, July 2010.

[18] D. R. Jones, M. Schonlau, and W. J. Welch. *Efficient Global Optimization of Expensive Black-Box Functions*. Journal of Global Optimization, 13(4):455–492, December 1998. Available from: http://dx.doi.org/10.1023/A:1008306431147, doi:10.1023/A:1008306431147.

[19] M. Emmerich, A. H. Deutz, and J. Kruisselbrink. *On Quality Indicators for Black-Box Level Set Approximation*. In EVOLVE- A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation, volume 447 of *Studies in Computational Intelligence*, pages 157–185. Springer Berlin Heidelberg, 2013.

[20] T. Ulrich and L. Thiele. *Maximizing Population Diversity in Single-objective Optimization*. In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, pages 641–648, New York, NY, USA, 2011. ACM. Available from: http://doi.acm.org/10.1145/2001576.2001665, doi:10.1145/2001576.2001665.

[21] V. Torczon and M. W. Trosset. *Using approximations to accelerate engineering design optimization*. Technical report, Institute for Computer Applications in Science and Engineering (ICASE), 1998.

[22] N. Hansen and A. Ostemeier. *Adapting Arbitrary Normal Mutation Distributions in Evolution Strategies: The Covariance Matrix Adaptation*. In Proceedings of the 1996 IEEE Conference on Evolutionary Computation, pages 312 – 317, 1996.

[23] A. R. Solow and S. Polasky. *Measuring biological diversity*. Environmental and Ecological Statistics, 1(2):95–103, 1994. Available from: http://dx.doi.org/10.1007/BF02426650, doi:10.1007/BF02426650.

[24] P. Y. Papalambros and D. J. Wilde. *Principle of Optimal Design*. Cambridge University Press, 2000.

[25] K. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution: APractical Approach to Global Optimization*. Springer, 2005.

[26] Y. Hirakawa, T. Takayama, A. Kosaki, H. Kikuchi, T. Hirayama, and T. Sakurai. *Model Experiment of a Suppression-System for Wave Impact and Porpoising Phenomena*. Conference Proceedings of The Japan Society of Naval Architects and Ocean Engineers (in Japanese), 3:239–242, 2006.

[27] O. M. Faltinsen. *Hydrodynamics of High-Speed Marine Vehicles*. Cambridge University Press, 2005.

[28] K. Ito, Y. Hirakawa, T. Hirayama, T. Sakurai, and T. Dhaene. *Longitudinal Stability Augmentation of Seaplanes in Planing*. In Proceedings of AIAA Modeling and Simulation Technologies Conference (Aviation 2015), Dallas, Texas, June 22 - 26 2015. AIAA.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12:2825–2830, 2011.

[30] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 2nd edition, 1995.

3

SOMBAS in Optimization

⋆ ⋆ ⋆

K. Ito, I. Couckuyt, and T. Dhaene.

"Optimization of High-Dimensional Expensive Problems Using Self-Organizing Map Based Adaptive Sampling".

To be submitted

⋆ ⋆ ⋆

Surrogate model based optimization is an effective way of minimizing an objective of expensive functions. The drawback of this approach is that it is not scalable to large input dimensions. We approach this problem using the density learning characteristics of Self-Organizing Maps. The proposed algorithm learns and adds new samples in the domains where outputs are favorable, progressively increasing the density of sample points in these regions. For the functions tested, the proposed method has shown competitive advantages over two evolutionary algorithms and one state-of-the-art surrogate model assisted evolutionary algorithm as the input variable dimensionality grew from 20 to 100. Our results show that our density learning approach can be an effective alternative to the conventional surrogate model learning approach.

3.1 Introduction

Increasingly complex computational models are being used in engineering. These models often require a large number of variables to be tuned in design optimization. Moreover, high-fidelity simulation models take longer to compute which limits the number of runs that can be performed in a reasonable amount of time. These situations pose a challenging situation for the optimization tasks. Surrogate model based optimization methods [1–4] has been a major way to tackle these expensive objective functions but the management of surrogate models remained a challenge when the input variables were large. Liu et al. [4] tackled this challenge by incorporating a dimension reduction (Sammon Mapping) before the Gaussian Process surrogate modeling to mitigate the so-called "curse of dimensionality". Differential Evolution was used for optimization. A similar approach but without using surrogate models was taken by Boschetti [5]. He reduced the input space dimension using a Local Linear Embedding (LLE) and employed Genetic Algorithm (GA) or Particle Swarm Optimization (PSO) for optimization. Surrogate models and dimension reductions are two ways of optimizing either expensive (i.e. a small number of function evaluations) or high-dimensional problems. However, optimization methods for expensive and high-dimensional functions are rare to the authors' knowledge.

3.2 Method

We tackle the problem of optimizing expensive functions with high-dimensional input space. We consider a situation in which it is desired to reduce the objective value as much as possible in a limited number of function evaluation. We will approach the problem not by using surrogate models that interpolate the data points but by learning the region inside the input space where a small output is likely to result, that is, by density learning. We use the Self-Organizing Map Based Adaptive Sampling (SOMBAS) [6, 7] to do this. We set the "truncation" value L in SOMBAS to the known optimal value of well-known test function and investigate how much it reduces the objective value in a given number of function evaluation budget. The truncation value L is a parameter that you can set up for SOMBAS to let the algorithm search for diversity (i.e. maximize distance to nearest sampled points instead of minimizing the objective value) when the objective values are below the value given in L.

Note that in SOMBAS, the learning of interesting input region is done using Self-Organizing Map (SOM). The learning cost of SOM increases linearly with respect to the number of dimensions and $O(NT \cdot NC)$ with respect to the number of sample points, where NT is the number of training samples and NC is the number of cells (or weight vectors) in SOM. Both NT and NC can be set by the user at

a convenient size (typically in the order of 10 or 100) and training SOM does not entail any inversion of matrices. Furthermore, the density learning with SOM does not have to be accurate and a small number of iterations (typically in the order of 10) of batch learning suffices. The cost for the training sample updating in a feasible region search is at most $O(NT \cdot N_f)$, where N_f is the number of function evaluation performed up to the iteration in question. However, in optimization, the cost of the training sample updating is $O(NT \cdot NC)$ because the nearest neighbor distances d are not calculated (i.e., objective values are not below the truncation value L). The inversion of the covariance matrix in multivariate Gaussian perturbations performed on the candidate samples scale as $O(D^3)$, where D is the number of dimensions. However, the inversion is performed only once per iteration and SOMBAS has been tested to work with D of up to 1000. The inversion cost of the covariance matrix is negligible up to this number of dimensions. In this work, we deal with D of up to 100 to see how the optimization performance scales with respect to D under a limited number of function evaluation budget.

3.3 Experiments

We take two steps. In the first step, we look at the convergence characteristics of SOMBAS. In the second step, we focus on optimization under very limited function evaluation budgets and at different numbers of input dimensions. To investigate the optimization capability of SOMBAS, We used the following functions: Rosenbrock, Rasgtrigin, Rotated Ellipsoid, Ackley, Manevich, Griewank, and Ellipsoid. These functions are described in 3.A.

The result is compared with those of the popular Differential Evolution (DE) in the first step, and with DE, Evolution Strategy (ES), and Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Medium-Scale Computationally Expensive Optimization Problems (GPEME) [4] in the second step. For DE and ES, we employed the classical DE1 (or DE/rand/1/bin) as described in [8, 9] and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [10]. These two algorithms are well documented and readily available [11, 12]. The stopping criteria for the benchmark functions were the best solution reaching the objective value below 1.0×10^{-6}, or a maximum number of function evaluations reached, or the difference between the worst sample and the best sample in the solution set becomes less than 1.0×10^{-6}. We will call this the "gap tolerance". The solution set refers to the training set in SOMBAS and the population in DE.

Optimization was run multiple times and average value of the number of function evaluations $\tilde{N}_f$ and average minimum objective value reached $\tilde{f}_{min}$ were computed. The global minima of the seven functions are 0 and the corresponding input values are $x_i = 1,\ i = 1, \cdots, D$ for Rosenbrock and Manevich and $x_i = 0$ for the remaining three functions. The D denotes the number of dimensions.

For the 30 dimensional problems ($D = 30$) shown in Table 3.1, the number of training samples in SOMBAS and the population sizes of DE were set to be equal, between 20 and 45, depending on the function to be minimized. We review the performance of optimization methods at three different numbers of function evaluations: 2,000, 20,000 and 200,000. Since both SOMBAS and DE have multiple solutions computed at each iteration, the actual numbers of function evaluations are always larger than the stopping conditions. Table 3.1 summarizes the mean of minimum function values reached and mean number of function evaluations performed. For 2,000 and 20,000 function evaluation runs, the numbers are average of 20 runs and for the 200,000 they are the average of 5 runs. Upper and lower bounds of inputs to the test functions were set to -10 and 10 respectively.

Table 3.1 Optimization results of 30 dimensional functions after 2,000, 20,000, and 200,000 function evaluations (average of 20 runs). The mean of minimum objectives obtained in 20 runs is shown under $\tilde{f}_{min}$ and the standard deviation of the minimum objectives is shown in the brackets. Entries with "n.a." indicate that optimizations have already converged.

	$SOMBAS$		DE	
Function	$\tilde{N}_f$	$\tilde{f}_{min}(Std.)$	$\tilde{N}_f$	$\tilde{f}_{min}(Std.)$
Rosenbrock	2019	$\mathbf{193}(15.2)$	2025	$4.25e+05(1.81e+05)$
Rastrigin	2013	$\mathbf{189}(18.9)$	2030	$293(23.1)$
Rotated Ellipsoid	2003	$\mathbf{63.5}(19.4)$	2010	$418(62.3)$
Ackley	2005	$\mathbf{4.08}(0.373)$	2020	$6.12(0.371)$
Manevich	2014	$\mathbf{0.0177}(0.0161)$	2010	$0.101(0.0319)$
Rosenbrock	20018	$\mathbf{59.4}(18.80)$	20025	$683(351)$
Rastrigin	20014	$\mathbf{48.1}(50.9)$	20020	$76.8(9.44)$
Rotated Ellipsoid	20012	$\mathbf{5.55}(2.50)$	20010	$18.3(7.43)$
Ackley	17691	$1.48(1.05)$	20020	$\mathbf{0.000373}(6.25e-05)$
Manevich	14310	$1.45e-06(1.86e-06)$	13163	$\mathbf{9.41e-07}(5.28e-08)$
Rosenbrock	190155	$\mathbf{3.12}(2.58)$	200025	$7.21(1.26)$
Rastrigin	63723	$18.7(4.38)$	115031	$\mathbf{8.97e-07}(9.52e-08)$
Rotated Ellipsoid	200011	$0.000399(0.000561)$	152358	$\mathbf{9.65e-07}(3.77e-08)$
Ackley	$n.a.$	$n.a.$	31104	$\mathbf{9.60e-07}(1.54e-08)$
Manevich	$n.a.$	$n.a.$	$n.a.$	$n.a.$

In Table 3.1, we observe that the Ackley and the Manevich functions converge prematurely for SOMBAS while the Manevich function converges successfully in about 13,000 function evaluations for DE. It also shows that by 200,000 function evaluations DE has minimized the function values below the 1.0×10^{-6} threshold except for the Rosenbrock function. For SOMBAS, Rastrigin converged to local optima and the Rotated Ellipsoid function has not converged at 200,011 evaluations. Roughly speaking, DE seems to be more accurate when we have function evaluation counts of over 200,000. On the other hand, the minimum objective values achieved by SOMBAS seems to be smaller compared to those of DE when the numbers of function evaluations are less than 20,000.

Since the above results were obtained with the number of training samples of SOMBAS equal or similar to the population sizes of DE that were found in [8, 9], we performed optimization runs with a larger number of training samples and population size. In particular, we modified the number of function evaluations to 2000 and the number of training samples of SOMBAS and the population size of DE to 900 while all the remaining parameters were kept the same. Results are shown in Table 3.2. It clearly shows that the minimum function values found by SOMBAS are substantially better than those found by DE. Also, compared to the function values attained in Table 3.1, DE has shown greater increases in function values compared to the increases for SOMBAS.

Table 3.2 Effect of a large number of training samples for SOMBAS and population size for DE (900) in optimization of 30-dimensional functions for relatively small number of function evaluations (2000) (average of 20 runs). The mean of minimum objectives obtained in 20 runs is shown under $\tilde{f}_{min}$ and the standard deviation of the minimum objectives is shown in the brackets.

	$SOMBAS$		DE	
Function	$\tilde{N}_f$	$\tilde{f}_{min}(Std.)$	$\tilde{N}_f$	$\tilde{f}_{min}(Std.)$
Rosenbrock	2283	$\mathbf{201}(4.08)$	2700	$1.33e+06(2.36e+05)$
Rastrigin	2083	$\mathbf{219}(15.4)$	2700	$732(37.8)$
Rotated Ellipsoid	2379	$\mathbf{11.9}(16.1)$	2700	$533(73.2)$
Ackley	2353	$\mathbf{2.38}(0.270)$	2700	$12.5(0.284)$
Manevich	2196	$\mathbf{0.0982}(0.0282)$	2700	$3.41(0.979)$

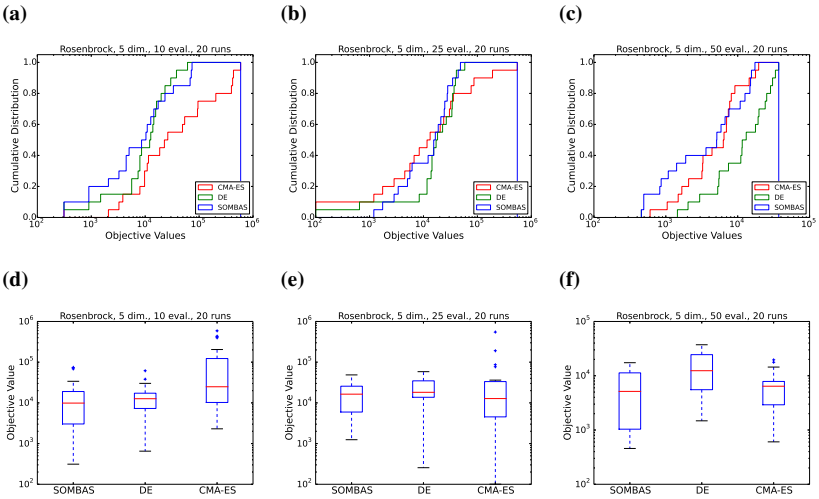
In the second step of this subsection, we focused on the optimization scenario with a very limited function evaluation budget. Here, different numbers of dimensions as well as numbers function evaluations were investigated. On the same five functions investigated so far, optimizations were performed in 5, 50, and 100 dimensions with function evaluation budgets of 2, 5, and 10 times the number of dimensions. At each combination of dimension and function evaluation budget, 20 optimization runs were performed for statistical robustness. We also applied Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [10] besides DE for comparison. Again, upper and lower bounds of inputs to the test functions were set to -10 and 10 respectively.

Table 3.3 summarizes statistics organized with respect to the five functions and the three optimization methods. It shows the combined statistics of minimum function responses attained in the given number of dimensions and function evaluations. The rows for function "All" indicate the statistics of all five function responses combined. We see that for the five functions, each with nine different combinations of dimensions and number of function evaluation budget, SOMBAS

attained on average the minimum response among the three optimization methods. With Rosenbrock, an order of magnitude smaller mean-minimum response was achieved compared to the other two. Similarly, roughly two times smaller mean minimum response was achieved with Rastrigin. In Table 3.3, the number of training samples of SOMBAS was fixed to 10 and the Self-Organizing Map size was also fixed to 5×5 cells in the rectangular cell arrangement. Further details on the setup of algorithm parameters of both DE and SOMBAS can be found in 3.C.

Fig. 3.1 to Fig. 3.3 show the empirical cumulative distributions and box plots of minimum response achieved by the three methods for the Rosenbrock function. SOMBAS reached about an order of magnitude smaller values compared to DE and CMA-ES on 50 and 100 dimensional Rosenbrock functions when the function evaluation budget was $2 \times D$. At a larger number of function evaluations ($> 10 \times D$), the advantage starts to fade away. In five dimensions, the difference between the three methods is small regardless of the function evaluation budgets. The other four benchmark functions also show similar trends. The average of minimum function value found $\tilde{f}_{min}$ in each of nine different combination of dimensions D and numbers of function evaluations $m \times D$ is listed in Table 3.B.1. The m is a multiplication factor to D to obtain the number of function evaluations at which iteration of the optimization is stopped.

Figure 3.1 Distribution of f_{min} on 5 dimensional Rosenbrock Function after 10 (left column), 25 (middle column), and 50 (right column) function evaluations.



In Table 3.4, summary statistics of the same experiments as in Table 3.3 (and Table 3.B.1 in Appendix) are shown with respect to different dimensionality of the problems. In this table, the results of the five test functions at three differ-

Figure 3.2 Distribution of f_{min} on 50 dimensional Rosenbrock Function after 100 (left column), 250 (middle column), and 500 (right column) function evaluations

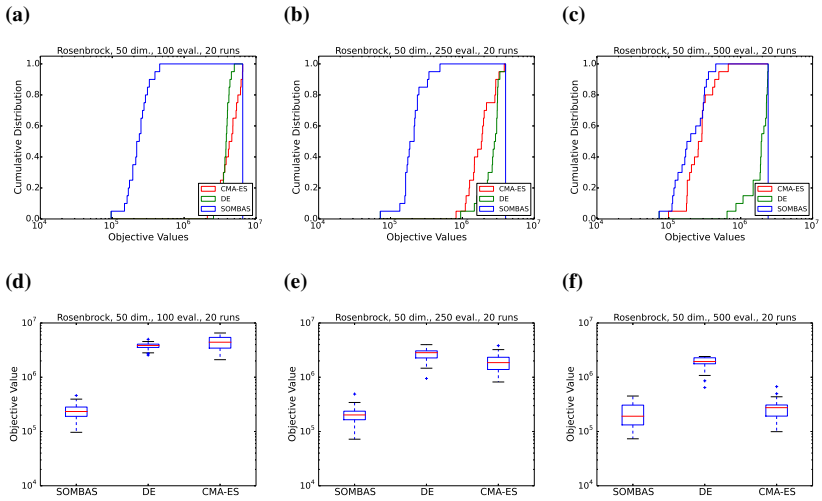
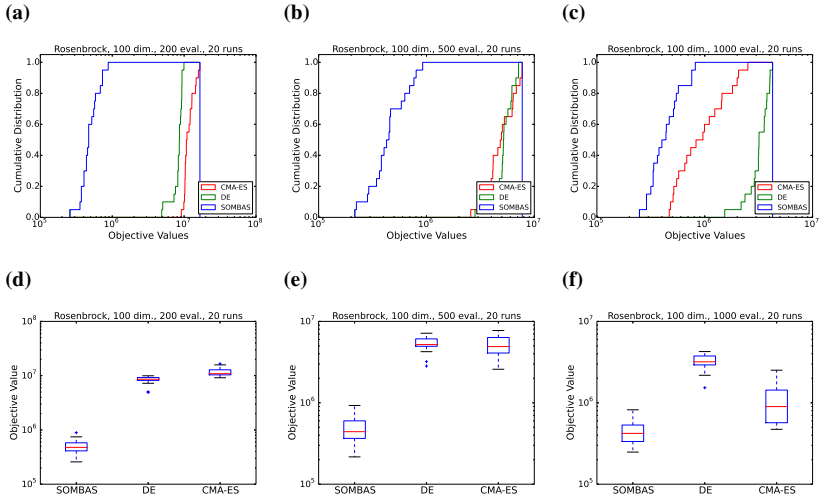


Figure 3.3 Distribution of f_{min} on 100 dimensional Rosenbrock Function after 200 (left column), 500 (middle column), and 1000 (right column) function evaluations.



ent m (numbers of function evaluations) are combined. The average minimum reached by SOMBAS at 50-dimensional problems and 100-dimensional problems

are 10 times smaller than those reached by DE or CMA-ES. For five-dimensional problems, the average minimum of SOMBAS is about 3/4 of DE and about 1/4 of CMA-ES.

The differences in time costs of the optimizations of the five functions by SOMBAS, DE, and CMA-ES can be seen in Table 3.5. The time required to complete the optimization of each of the 100-dimensional problems with 1000 function evaluations was measured. Table 3.5 shows the statistics of 20 runs of optimization per function. The algorithm parameters are the same as the Table 3.3 and Table 3.4. We observe that SOMBAS takes the longest among the three methods. However, for most engineering optimization applications in which a function evaluation can take hours, the time costs shown in the table are insignificant and the number of function evaluations would be a more significant performance measure.

In Table 3.3 and Table 3.4, the number of training samples in SOMBAS has been kept constant. In Table 3.6 we compare two kinds of setup for SOMBAS. The first one is the setup used in the previous two tables (3.3 and 3.4) with fixed training size and SOM size. The second set up is with varying training size and SOM size according to the dimensionality of the problem. In this setup, we let the number of training samples be equal to the dimensionality D of the problem. The SOM size is also adapted accordingly: $\lfloor\sqrt{D}\rfloor \times \lfloor\sqrt{D}\rfloor$, where $\lfloor\ \rfloor$ is the floor function. The functions, their dimensionality and function evaluation budgets are the same as Table 3.3 and Table 3.4. The dimensionality adapted setup gave another 10 fold decrease in the average minimum function value reached compared to what we had in Table 3.3 and Table 3.4. We have tried only one variant here, but further investigation on how to choose the right number of training samples and SOM dimensions could be interesting.

We now look into some more algorithm parameter sensitivities of SOMBAS but the functions and other setup remains the same as Table 3.3. Selectivity parameter T (Table 3.7), mutation probability P_m (Table 3.8), and weight constant ρ in the merit function (Table 3.9) were considered.

Larger values of T produce smaller minimum function values on average as observed in Table 3.7. If $T \gg 4$ there is no selection and essentially all weight vectors from SOM are evaluated to get the responses of the objective function and become candidates for new training samples. Therefore, all the "selection" takes place at training sample update. If $T \ll 1$, on the other hand, most of the weight vectors from SOM do not survive the selection process and only a few are actually evaluated with the objective function. This, in principle, sounds more economical than the former case, but it relies heavily on the response estimate of the SOM, and it also risks losing diversity very quickly. This table shows that, for optimization, it is better to set T sufficiently large, e.g. $T \approx 4$.

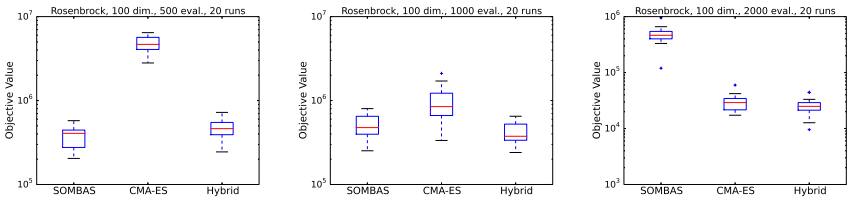
In Table 3.8 we see that mutation does not alter the optimization performance in a significant way. To have $P_m < 1.0$ means that some of the elements of the

weight vector may remain unperturbed. If this probability is set to be very low, it resembles a coordinate search around each of the training samples. If it is set to 1, it is similar to sampling from kernel density functions in which training sample represents the centers of multi-variate Gaussian distribution. This table suggests that P_m is not a critical parameter in optimization search when the number of function evaluations is small compared to what is needed for convergence, but further investigation may be necessary by taking statistics with respect to different kinds of objective functions, such as separability and multi-modality and different function evaluation budgets.

The ρ does not show significant effect on optimization either. As observed in Table 3.9, the average minimum achieved for both $\rho = 2$ and $\rho = 0.01$ are about the same. If $\rho \gg 1$, it becomes similar to Maximin Sampling in which the objective is to maximize the minimum distance to the training samples [13]. Probably the concentration of training samples around small responses that happens as a result of the training sample updating is far more significant than the diversification pressure provided by the ρ. There may be some refinement opportunity in the implementation of the merit function.

Figure 3.4 Box plot showing effect of hybrid algorithm: the first $2 \times D$ function evaluations are performed with SOMBAS and then the best sample is provided as the starting point of the CMA-ES that runs up to the allowed maximum number of function evaluations.

(a) 100 dim. 500 eval. **(b)** 100 dim. 1000 eval. **(c)** 100 dim. 2000 eval.



In optimization tasks with input dimensions $D > 20$, it is probably a good idea to use SOMBAS at an early stage, particularly if the problems' input parameters are of higher dimensions. In our experiments, this means the first $m \times D$ samples where m is an integer between 1 and 10 and D is the number of dimensions and taking values such as $D = 30, 50, \text{or} 100$. This enables the identification of good starting points for a more accurate optimization algorithms such as DE and CMA-ES for further refinement of the solutions. Table 3.10 shows an example of such idea. Here we implemented a simple hybrid algorithm "Hybrid". This algorithm uses SOMBAS for the first $2 \times D$ function evaluations. Then, the best solution found is used as a starting point for CMA-ES. In the table, summary statistics are shown for the same functions at the same dimensions as before. The function

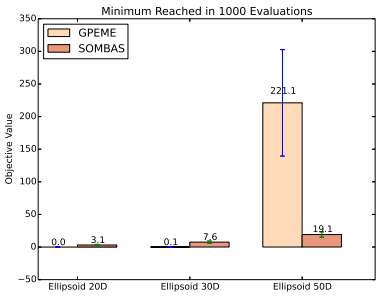
evaluation budget was set to 5, 10, and 20 times the number of dimensions D instead of 2, 5, and 10 in the previous tables. The summary statistics suggest that "Hybrid" shows superior performance compared to SOMBAS or CMA-ES alone. Figure 3.4 shows the box plots indicating the spread of f_{min} achieved for Rosenbrock at 100 dimensions at the three evaluation budgets. We observe that Hybrid attains fast reduction of responses at 500 function evaluations in par with SOMBAS. At 2000 function evaluations, CMA-ES and Hybrid are almost even. Thus, for this example, if the number of function evaluations is larger than 2000, there would be no reason to use Hybrid. However, if the cost of a single function evaluation is large, there is value in obtaining good solutions from the early stage of the optimization. Figure 3.4 also explains why Hybrid shows the best performance in Table 3.10. It is because the mean is the result from three budgets $5 \times D$, $10 \times D$, and $20 \times D$ function evaluations. SOMBAS lags behind considerably in $20 \times D$. CMA-ES is slow in $5 \times D$. Hybrid, although may or may not be the best in any of these budgets, is always close to the objective values obtained by the better performing one of the remaining two algorithms. Thus on average, is the best by not being the worst performing algorithm. Thus, Hybrid gave us the "minimum regret" option of the three.

In Table 3.11and in its bar chart representation, Figure 3.5, we compare SOMBAS and Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Medium-Scale Computationally Expensive Optimization Problems (GPEME) [4]. GPEME is a surrogate model assisted optimization method that employs Sammon mapping to map the original design space to a lower dimensional space. The Gaussian Process modeling [14] of the objective function is done on the lower-dimensional input space.

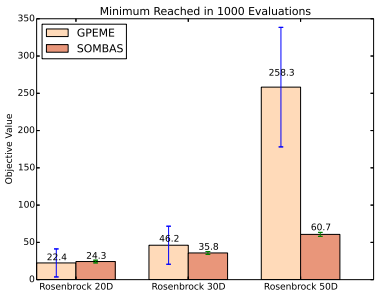
The test functions employed are different from the previous experiments in order to compare with the results in the publication. The upper and lower bounds of the input variables are, thus, matched to those described in the paper [4]. The number function evaluations were also set to 1000 and 20 independent runs were performed for each function optimizations as described in the paper [4]. Figure 3.5 shows, in bar chart, the results of optimizing the four functions in three different dimensions namely, 20, 30, and 50 dimensions summarized in Table 3.11. The height of the bar indicates the minimum value achieved at the end of 1000 function evaluations, and the whiskers indicate its standard deviation of the 20 runs. We can see that up to 30 dimensions, GPEME can be more accurate than SOMBAS whereas in 50 dimensions SOMBAS reached far lower values in all four objective functions. DE results are also listed in Table 3.11. The values of DE are different from what is given in the paper [4] because we have used different setups for DE parameters.

Figure 3.5 Bar charts showing minimum objective values obtained by GPEME and SOMBAS: 20 runs of 1000 function evaluations each were performed.

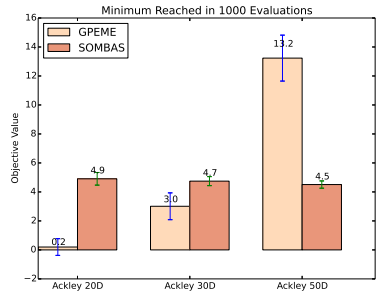
(a) Ellipsoid



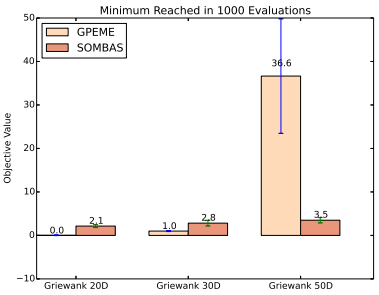
(b) Rosenbrock



(c) Ackley



(d) Griewank



3.4 Conclusion

In optimization tasks, SOMBAS has shown a rapid reduction of objective function values at a relatively small number of function evaluations and high numbers of input dimensions, say in the number of function evaluations less than ten times the number of input dimensions and the number of dimensions being between 30 and 100. In particular, SOMBAS has shown fast reduction of objective values (without resorting to surrogate modeling of the objective function) when compared with DE, CMA-ES, and GPEME in limited function evaluation budgets. The computational experiments have shown that this was most prominent when the number of function value evaluations were limited, such as $2 \times D$, with $D = 50$. As the number function evaluations increased or the number of dimensions of the inputs gets smaller, the relative advantage fades away as more accurate methods such as CMA-ES, GPEME becomes more efficient. Density learning and adaptive sampling can be an efficient method to deal with high-dimensional and expensive

objective functions.

In future work, it would be beneficial to test on even higher dimensional problems with inputs in the order of 1000 dimensions and higher.

Table 3.3 Summary of 20 minimization runs for 5 test functions at 3 different dimensions and 3 different settings of maximum number of function evaluations. "All" is the combined statistics of the five benchmark functions.

Function	Methods	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
All	SOMBAS	900	**49,031.200**	**138,932.200**	0.311	**926,876.000**
	DE	900	561,146.400	1,653,569.000	**0.239**	9,943,861.000
	CMA-ES	900	554,937.200	2,038,124.000	0.351	16,586,288.000
Rosenbrock	SOMBAS	180	**242,465.200**	**223,384.400**	311.851	**926,876.000**
	DE	180	2,801,761.000	2,724,343.000	255.317	9,943,861.000
	CMA-ES	180	2,770,141.000	3,833,323.000	**100.565**	16,586,288.000
Rastrigin	SOMBAS	180	**618.578**	**476.474**	24.862	**1,477.860**
	DE	180	1,412.729	1,138.204	**23.125**	3,446.527
	CMA-ES	180	1,203.552	1,020.696	31.790	3,469.831
Rot. Ellip.	SOMBAS	180	**2,060.542**	**2,503.437**	**1.524**	10,500.050
	DE	180	2,540.826	2,587.632	7.910	**9,833.558**
	CMA-ES	180	3,323.752	3,504.440	1.608	16,548.950
Ackley	SOMBAS	180	**7.779**	**1.119**	**3.626**	**11.553**
	DE	180	11.617	1.677	6.984	14.069
	CMA-ES	180	10.849	2.700	4.930	15.773
Manevich	SOMBAS	180	**3.972**	**3.545**	0.311	20.053
	DE	180	5.408	3.792	**0.239**	**16.856**
	CMA-ES	180	6.263	9.317	0.351	91.240

Table 3.4 Summary of 20 runs of minimizing 5 functions at 3 different dimensions and 3 different settings of maximum number of function evaluations.

Dim.	Methods	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
5	SOMBAS	300	**2,893.125**	9,579.070	0.605	74,336.480
	DE	300	3,630.300	**9,474.233**	2.323	**61,622.490**
	CMA-ES	300	12,396.820	64,659.340	**0.537**	589,876.300
50	SOMBAS	300	**45,950.500**	**99,895.630**	**0.311**	**490,143.000**
	DE	300	552,548.200	1,189,206.000	0.996	4,989,037.000
	CMA-ES	300	443,619.500	1,235,422.000	0.414	6,532,953.000
100	SOMBAS	300	**98,249.990**	**208,299.500**	0.395	**926,876.000**
	DE	300	1,127,261.000	2,484,626.000	**0.239**	9,943,861.000
	CMA-ES	300	1,208,795.000	3,197,257.000	0.351	16,586,288.000

Table 3.5 Time costs of optimization of the five functions at 100 dimensions and 1000 function evaluations. Statistics of 20 runs. CPU: Intel Core 2 Duo 3.16 GHz.

Method	Function	Mean (sec.)	St. Dev. (sec.)	Median (sec.)
SOMBAS	Rosenbrock	8.467	0.126	8.469
DE	Rosenbrock	**0.302**	**0.007**	**0.297**
CMA-ES	Rosenbrock	0.918	0.098	0.891
SOMBAS	Rastrigin	9.378	0.079	9.375
DE	Rastrigin	**0.958**	**0.007**	**0.953**
CMA-ES	Rastrigin	1.801	0.086	1.781
SOMBAS	Rot. Ellip.	11.080	0.139	11.031
DE	Rot. Ellip.	**3.072**	**0.008**	**3.078**
CMA-ES	Rot. Ellip.	3.648	0.070	3.625
SOMBAS	Ackley	9.836	0.480	9.812
DE	Ackley	**0.878**	**0.034**	**0.890**
CMA-ES	Ackley	1.399	0.335	1.422
SOMBAS	Manevich	8.994	0.351	8.860
DE	Manevich	**0.432**	**0.008**	**0.437**
CMA-ES	Manevich	1.366	0.121	1.304

Table 3.6 Summary of SOMBAS with different number of training samples minimizing 5 test functions at 3 different dimensions and 3 different settings for maximum number of function evaluations.

Train. Samp.	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
Fixed to 10	900	45,777.970	129,870.800	0.285	963,875.900
Equal to D	900	4,792.621	19,410.150	0.093	357,146.100

Table 3.7 Summary of SOMBAS with different selectivity T minimizing 5 test functions at 3 different dimensions and 3 different settings for maximum number of function evaluations.

Selectivity	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
Low, $T = 4.0$	900	4,591.983	16,695.700	0.105	208,590.500
High, $T = 0.5$	900	7,178.841	30,777.100	0.060	489,114.600

Table 3.8 Summary of SOMBAS with different Mutation Probability minimizing 5 test functions at 3 different dimensions and 3 different settings for maximum number of function evaluations

Mutation Prob.	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
1.0	900	4,874.758	18,699.480	0.109	306,843.900
0.25	900	5,094.691	23,469.330	0.164	387,932.800

Table 3.9 Summary of SOMBAS with different Merit Weight ρ performing 5 test function at 3 different dimensions and 3 different settings for maximum number of function evaluations.

ρ	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
2	900	4,023.054	13,481.460	0.114	208,199.800
0.01	900	4,384.766	19,332.080	0.097	470,756.700

Table 3.10 Summary of an effect of hybrid algorithm "Hybrid": the first $2 \times D$ function evaluations are performed with SOMBAS and then the best sample is provided as the starting point of the CMA-ES that runs up to the allowed maximum number of function evaluations (5,10, or 20)$\times D$.

Methods	N	$\tilde{f}_{min}$	St. Dev.	Min	Max
SOMBAS	900	45,092.970	128,860.000	0.311	952,357.800
CMA-ES	900	180,432.600	793,864.900	0.297	6,440,175.000
Hybrid	900	**30,461.020**	**103,880.300**	**0.227**	**722,556.700**

Table 3.11 Comparing SOMBAS to a state-of-the-art optimization method for expensive objective function optimization. Statistics of 20 runs.

	DE		GPEME		SOMBAS	
Function	$\tilde{f}_{min}$	St. Dev.	$\tilde{f}_{min}$	St. Dev.	$\tilde{f}_{min}$	St. Dev.
Ellipsoid 20D	80.471201	20.8733580	**1.30E-05**	**2.18E-05**	3.118508	0.9714572
Ellipsoid 30D	377.501744	69.1929920	**0.0762**	**0.0401**	7.55524	1.6964039
Ellipsoid 50D	1635.988263	243.5844059	221.0774	81.6123	**19.131114**	**4.1030721**
Rosenbrock 20D	1269.905151	206.4475974	**22.4287**	18.7946	24.298632	**1.8222784**
Rosenbrock 30D	2925.170609	510.1286707	46.1773	25.5199	**35.836207**	**1.9172995**
Rosenbrock 50D	7587.404909	806.0995888	258.2787	80.1877	**60.722905**	**2.6847508**
Ackley 20D	12.885580	1.0100859	**0.199**	**0.5771**	4.905016	0.4359553
Ackley 30D	16.180565	0.6248096	**3.0105**	0.925	4.745343	**0.3114214**
Ackley 50D	18.095235	0.3721917	13.2327	1.5846	**4.50843**	**0.2508913**
Griewank 20D	83.901426	13.8013613	**0.0307**	**0.0682**	2.149197	0.3456406
Griewank 30D	193.039613	34.8963740	**0.9969**	**0.108**	2.824373	0.6443779
Griewank 50D	488.303479	44.7642325	36.6459	13.1755	**3.497098**	**0.609253**

Appendix

3.A Test Functions

In the following, we describe the test functions used in this paper. The θ^* denotes the globally optimum solution vector, and $f(\theta^*)$ its response value. The D denotes the number of dimensions. The upper and lower bounds of θ s are by default $-10 \leq \theta_j \leq 10$ where $\theta = [\theta_0, \theta_1, \ldots, \theta_{D-1}]^T$.

Rosenbrock

$$f(\theta) = \sum_{j=0}^{D-2} \big(100(\theta_{j+1} - \theta_j^2)^2 + (\theta_j - 1)^2\big), \tag{3.1}$$
$$j = 0, 1, \ldots, D-1, \quad D > 1,$$
$$f(\theta^*) = 0, \quad \theta_j^* = 1.$$

Rastrigin

$$f(\theta) = \sum_{j=0}^{D-1} \big(\theta_j^2 - 10\cos(2\pi\theta_j) + 10\big), \tag{3.2}$$
$$j = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_j^* = 0.$$

Rotated Ellipsoid

$$f(\theta) = \sum_{i=0}^{D-1} \left(\sum_{j=0}^{i} \theta_j^2 \right)^2, \tag{3.3}$$
$$j = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_j^* = 0.$$

Ackley

$$
\begin{aligned}
f(\theta) \quad = \quad & -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=0}^{D-1}\theta_j^2}\right) \\
& -\exp\left(\frac{1}{D}\sum_{j=0}^{D-1}\cos(2\pi\theta_j)\right) \\
& +20-\exp(1), \\
& j=0,1,\ldots,D-1, \\
& f(\theta^*)=0, \quad \theta_j^*=0.
\end{aligned}
\tag{3.4}
$$

Manevich

$$
\begin{aligned}
f(\theta) = \sum_{j=0}^{D-1}\left[(1-\theta_j)^2/2^j\right], \\
j=0,1,\ldots,D-1, \\
f(\theta^*)=0, \quad \theta_j^*=1.
\end{aligned}
\tag{3.5}
$$

Griewank

$$
\begin{aligned}
f(\theta) = 1+\sum_{i=0}^{D-1}\frac{\theta_i^2}{4000}-\prod_{i=0}^{D-1}\cos(\frac{\theta_i}{\sqrt{i}}), \\
i=0,1,\ldots,D-1, \\
f(\theta^*)=0, \quad \theta_i^*=0.
\end{aligned}
\tag{3.6}
$$

Ellipsoid

$$
\begin{aligned}
f(\theta) = \sum_{i=0}^{D-1} i\theta_i^2, \\
i=0,1,\ldots,D-1, \\
f(\theta^*)=0, \quad \theta_i^*=0.
\end{aligned}
\tag{3.7}
$$

3.B Detailed Statistics of Optimization

Table 3.B.1 shows the statistics of optimizing the five function N (=20) times at three different number of function evaluation setting and 3 different dimensions D. Table 3.B.1 is the detailed version of Table 3.3 and Table 3.4. Each row corresponds to a particular combination of algorithm, function, function dimension and number of function evaluation setting. The settings for the number of function evaluations can be obtained by computing $m \times D$. When the total number of function evaluation exceeds $m \times D$ the iteration is stopped. So the actual total

number of function evaluation will be slightly larger than $m \times D$ but on average smaller than $m \times D + NP$ or $m \times D + NT$ where NT is number of training samples in SOMBAS and NP is number of population in DE and CMA-ES. The numbers after the algorithm names are simply for indexing purposes. The $\tilde{f}_{min}$ denotes the average of minimum objective value found.

Table 3.B.1: Statistic of the optimization result of the five functions

	Function	D	m	N	f_{min}	sd	median	min	max
SOMBAS1	Ackley	5	2	20	7.65	1.72	7.84	4.71	10.84
SOMBAS2	Manevich	5	2	20	8.96	5.67	7.27	1.26	20.05
SOMBAS3	Rastrigin	5	2	20	64.74	13.35	62.91	41.01	93.03
SOMBAS4	Rosenbrock	5	2	20	18569.84	24378.93	9879.17	311.85	74336.48
SOMBAS5	Rotated_Ellipsoid	5	2	20	23.69	15.72	18.10	7.58	62.01
SOMBAS6	Ackley	50	2	20	8.17	0.54	8.27	7.08	9.20
SOMBAS7	Manevich	50	2	20	2.50	1.27	2.37	0.55	4.89
SOMBAS8	Rastrigin	50	2	20	629.14	69.02	635.36	481.50	744.80
SOMBAS9	Rosenbrock	50	2	20	245174.52	85930.43	233640.35	96604.97	460465.71
SOMBAS10	Rotated_Ellipsoid	50	2	20	1404.76	788.39	1236.69	420.87	2883.94
SOMBAS11	Ackley	100	2	20	8.06	0.52	8.04	7.09	9.05
SOMBAS12	Manevich	100	2	20	2.75	2.01	1.92	0.90	9.44
SOMBAS13	Rastrigin	100	2	20	1262.76	83.83	1252.91	1145.86	1477.86
SOMBAS14	Rosenbrock	100	2	20	516668.16	153260.73	476799.87	257858.17	891665.11
SOMBAS15	Rotated_Ellipsoid	100	2	20	5146.12	2053.19	4863.28	2137.07	8979.88
SOMBAS16	Ackley	5	5	20	8.12	1.90	8.08	5.33	11.55
SOMBAS17	Manevich	5	5	20	5.86	3.63	5.86	0.82	15.86
SOMBAS18	Rastrigin	5	5	20	58.34	14.65	55.36	30.86	90.00
SOMBAS19	Rosenbrock	5	5	20	18142.54	13937.96	16373.71	1244.82	48682.00
SOMBAS20	Rotated_Ellipsoid	5	5	20	26.96	13.48	27.06	5.13	56.19
SOMBAS21	Ackley	50	5	20	7.82	0.39	7.81	7.19	8.52
SOMBAS22	Manevich	50	5	20	3.33	2.29	2.91	0.31	7.96
SOMBAS23	Rastrigin	50	5	20	592.01	41.84	590.35	511.41	665.38
SOMBAS24	Rosenbrock	50	5	20	216663.34	88879.12	202004.43	72135.10	490143.00
SOMBAS25	Rotated_Ellipsoid	50	5	20	1184.16	792.58	1022.01	400.63	3642.19
SOMBAS26	Ackley	100	5	20	7.94	0.47	8.01	7.20	8.70
SOMBAS27	Manevich	100	5	20	2.45	1.14	2.43	0.77	5.00
SOMBAS28	Rastrigin	100	5	20	1203.89	70.94	1205.79	1095.80	1350.49
SOMBAS29	Rosenbrock	100	5	20	481002.08	197394.08	441589.43	216456.31	926876.04
SOMBAS30	Rotated_Ellipsoid	100	5	20	5813.78	2617.17	5630.58	1364.07	10500.05
SOMBAS31	Ackley	5	10	20	6.41	1.24	6.38	3.63	8.45
SOMBAS32	Manevich	5	10	20	4.96	3.28	3.61	0.60	13.97
SOMBAS33	Rastrigin	5	10	20	46.37	8.45	47.17	24.86	62.84
SOMBAS34	Rosenbrock	5	10	20	6406.99	5904.64	5146.50	456.96	17388.16
SOMBAS35	Rotated_Ellipsoid	5	10	20	15.44	11.46	11.94	1.52	45.86
SOMBAS36	Ackley	50	10	20	7.87	0.40	7.88	7.23	8.75
SOMBAS37	Manevich	50	10	20	2.27	1.58	1.63	0.55	6.36
SOMBAS38	Rastrigin	50	10	20	553.07	53.34	551.38	478.11	680.64
SOMBAS39	Rosenbrock	50	10	20	221899.16	102804.65	191273.56	73392.04	450433.35
SOMBAS40	Rotated_Ellipsoid	50	10	20	1125.41	668.67	1119.62	322.03	3287.19
SOMBAS41	Ackley	100	10	20	7.97	0.41	7.95	7.30	8.67
SOMBAS42	Manevich	100	10	20	2.67	2.27	1.84	0.40	10.61
SOMBAS43	Rastrigin	100	10	20	1156.89	105.86	1150.45	937.48	1343.55
SOMBAS44	Rosenbrock	100	10	20	457659.74	166298.66	421045.39	248017.41	820097.62
SOMBAS45	Rotated_Ellipsoid	100	10	20	3804.56	1710.20	3348.31	1642.06	7382.68
DE1	Ackley	5	2	20	9.83	1.64	10.00	6.98	13.00
DE2	Manevich	5	2	20	8.11	4.45	6.93	2.32	16.86
DE3	Rastrigin	5	2	20	75.01	26.30	80.93	23.12	104.34
DE4	Rosenbrock	5	2	20	15431.40	14411.08	12656.79	650.90	61622.49
DE5	Rotated_Ellipsoid	5	2	20	29.67	13.60	26.82	7.91	54.04
DE6	Ackley	50	2	20	13.63	0.26	13.69	13.21	14.07
DE7	Manevich	50	2	20	8.88	3.24	8.60	2.38	14.91
DE8	Rastrigin	50	2	20	1597.66	90.08	1624.37	1435.55	1731.01
DE9	Rosenbrock	50	2	20	3749176.09	636617.95	3847948.36	2561727.10	4989037.00
DE10	Rotated_Ellipsoid	50	2	20	2433.63	658.71	2178.34	1614.18	4125.06
DE11	Ackley	100	2	20	13.63	0.18	13.65	13.14	13.88
DE12	Manevich	100	2	20	6.45	2.96	6.36	1.51	13.98
DE13	Rastrigin	100	2	20	3226.87	119.53	3242.54	2844.17	3446.53
DE14	Rosenbrock	100	2	20	8326549.17	1313243.16	8594077.60	4892880.62	9943861.32
DE15	Rotated_Ellipsoid	100	2	20	7388.87	1308.02	7155.51	4964.50	9833.56
DE16	Ackley	5	5	20	10.37	1.52	10.29	7.64	12.90
DE17	Manevich	5	5	20	7.19	3.31	6.55	2.48	14.31
DE18	Rastrigin	5	5	20	72.22	16.21	75.29	38.13	102.62
DE19	Rosenbrock	5	5	20	22962.32	14958.30	18179.80	255.32	58442.46
DE20	Rotated_Ellipsoid	5	5	20	30.09	14.72	32.20	10.25	59.46
DE21	Ackley	50	5	20	12.63	0.30	12.68	12.07	13.31
DE22	Manevich	50	5	20	4.45	1.77	4.25	1.90	8.20
DE23	Rastrigin	50	5	20	1364.31	60.01	1368.45	1256.04	1467.90
DE24	Rosenbrock	50	5	20	2644198.37	709142.42	2835509.13	948411.36	3987364.62
DE25	Rotated_Ellipsoid	50	5	20	1752.92	306.97	1716.48	1290.13	2378.15
DE26	Ackley	100	5	20	12.46	0.21	12.48	11.91	12.88
DE27	Manevich	100	5	20	2.37	1.00	2.13	0.97	5.51
DE28	Rastrigin	100	5	20	2794.39	78.34	2805.26	2599.25	2896.06
DE29	Rosenbrock	100	5	20	5304925.42	1128370.47	5190736.04	2826552.39	7151082.44
DE30	Rotated_Ellipsoid	100	5	20	5484.56	976.99	5382.92	3665.70	7443.04

Table 3.B.1: (continued)

	Function	D	m	N	f_{min}	sd	median	min	max
DE31	Ackley	5	10	20	9.63	1.12	9.50	7.92	11.94
DE32	Manevich	5	10	20	7.64	3.12	6.02	4.43	15.04
DE33	Rastrigin	5	10	20	77.10	12.97	75.01	49.06	100.93
DE34	Rosenbrock	5	10	20	15689.69	11153.08	12379.21	1476.81	37246.29
DE35	Rotated_Ellipsoid	5	10	20	34.22	12.83	34.53	10.97	54.99
DE36	Ackley	50	10	20	11.25	0.29	11.30	10.33	11.62
DE37	Manevich	50	10	20	2.78	0.99	2.73	1.00	5.77
DE38	Rastrigin	50	10	20	1157.32	67.95	1141.15	1041.36	1301.12
DE39	Rosenbrock	50	10	20	1885117.10	520682.45	1946726.37	646065.77	2410599.40
DE40	Rotated_Ellipsoid	50	10	20	1371.31	257.24	1351.32	877.29	1920.03
DE41	Ackley	100	10	20	11.12	0.24	11.17	10.74	11.55
DE42	Manevich	100	10	20	0.80	0.26	0.75	0.24	1.29
DE43	Rastrigin	100	10	20	2349.69	95.45	2366.29	2193.52	2494.73
DE44	Rosenbrock	100	10	20	3251801.22	693314.95	3189301.44	1532889.32	4259556.24
DE45	Rotated_Ellipsoid	100	10	20	4342.15	742.59	4247.66	2695.27	5845.17
CMA.ES1	Ackley	5	2	20	10.39	2.58	10.76	5.67	14.20
CMA.ES2	Manevich	5	2	20	21.52	19.94	15.36	6.19	91.24
CMA.ES3	Rastrigin	5	2	20	132.06	47.43	136.15	44.86	246.73
CMA.ES4	Rosenbrock	5	2	20	123442.65	182490.42	24886.76	2303.90	589876.31
CMA.ES5	Rotated_Ellipsoid	5	2	20	84.88	95.18	63.38	7.97	435.41
CMA.ES6	Ackley	50	2	20	13.02	0.77	13.09	11.75	14.69
CMA.ES7	Manevich	50	2	20	5.91	2.91	5.60	1.63	13.35
CMA.ES8	Rastrigin	50	2	20	1456.48	197.41	1485.03	993.13	1769.37
CMA.ES9	Rosenbrock	50	2	20	4403283.64	1397692.69	4421958.02	2106134.26	6532952.66
CMA.ES10	Rotated_Ellipsoid	50	2	20	3058.67	957.80	2887.86	1306.61	5718.68
CMA.ES11	Ackley	100	2	20	13.84	0.51	13.95	12.44	14.62
CMA.ES12	Manevich	100	2	20	4.44	1.71	4.43	1.71	7.62
CMA.ES13	Rastrigin	100	2	20	3072.23	218.97	3083.54	2601.79	3469.83
CMA.ES14	Rosenbrock	100	2	20	11835024.56	2090781.70	10875301.53	9149150.08	16586288.40
CMA.ES15	Rotated_Ellipsoid	100	2	20	9476.02	2495.04	9599.46	6035.41	16548.95
CMA.ES16	Ackley	5	5	20	8.76	2.29	8.19	4.93	12.89
CMA.ES17	Manevich	5	5	20	9.20	7.54	7.46	1.10	31.68
CMA.ES18	Rastrigin	5	5	20	68.52	26.47	64.92	33.96	134.47
CMA.ES19	Rosenbrock	5	5	20	55409.95	124781.09	12762.14	100.57	549938.57
CMA.ES20	Rotated_Ellipsoid	5	5	20	52.94	27.24	57.10	7.16	101.39
CMA.ES21	Ackley	50	5	20	12.01	1.42	11.80	10.15	15.64
CMA.ES22	Manevich	50	5	20	3.83	1.77	3.44	1.34	8.73
CMA.ES23	Rastrigin	50	5	20	1149.72	124.46	1142.14	903.57	1452.32
CMA.ES24	Rosenbrock	50	5	20	1951639.70	805122.15	1859109.56	817644.54	3824926.85
CMA.ES25	Rotated_Ellipsoid	50	5	20	2217.37	626.52	2110.28	1398.10	3766.36
CMA.ES26	Ackley	100	5	20	12.58	0.97	12.70	11.17	15.54
CMA.ES27	Manevich	100	5	20	2.81	1.20	2.80	0.35	6.05
CMA.ES28	Rastrigin	100	5	20	2363.10	275.19	2299.74	1807.79	2836.49
CMA.ES29	Rosenbrock	100	5	20	5187494.11	1483994.33	4906044.22	2581357.39	7727621.83
CMA.ES30	Rotated_Ellipsoid	100	5	20	7722.19	1723.09	7508.26	5226.76	11372.76
CMA.ES31	Ackley	5	10	20	6.95	1.39	6.66	5.05	9.94
CMA.ES32	Manevich	5	10	20	4.99	5.19	4.27	0.54	25.26
CMA.ES33	Rastrigin	5	10	20	54.91	18.19	52.50	31.79	102.71
CMA.ES34	Rosenbrock	5	10	20	6615.80	5334.96	6411.34	604.94	19590.35
CMA.ES35	Rotated_Ellipsoid	5	10	20	28.79	21.82	23.48	1.61	76.78
CMA.ES36	Ackley	50	10	20	9.89	2.46	9.22	7.72	15.77
CMA.ES37	Manevich	50	10	20	2.04	0.71	1.98	0.41	3.14
CMA.ES38	Rastrigin	50	10	20	796.13	92.50	793.57	627.70	987.57
CMA.ES39	Rosenbrock	50	10	20	289019.59	132309.11	274963.66	99319.73	672945.96
CMA.ES40	Rotated_Ellipsoid	50	10	20	1624.61	238.10	1627.01	1054.15	2013.40
CMA.ES41	Ackley	100	10	20	10.21	1.96	9.67	7.78	15.53
CMA.ES42	Manevich	100	10	20	1.63	0.68	1.52	0.59	3.36
CMA.ES43	Rastrigin	100	10	20	1738.82	183.92	1741.80	1384.45	2256.79
CMA.ES44	Rosenbrock	100	10	20	1079342.62	611708.11	897874.92	472041.35	2514209.69
CMA.ES45	Rotated_Ellipsoid	100	10	20	5648.31	1049.50	5593.88	3606.10	7472.06

3.C Parameter Setups

This section describes the parameter settings of Differential Evolution (DE) and Self-Organizing Map Based Adaptive Sampling (SOMBAS) used in generating the results in this paper. The parameter settings are by no means optimal, but it is given here for reproducibility and describing some reasoning that went behind it. In the following tables, column name NP signifies a number of population in DE and NT signifies a number of training samples for Self-Organizing Maps in SOMBAS. F and CR are scale factor and cross-over probability as typically defined for the classical DE [9, pp. 38,39]. The number of iteration for the Self-Organizing Map

was set between 10 and 40 with no appreciable effect on the results whether one set the number to 10 or 40. It is found during the trial runs of SOMBAS of the five functions in 30 dimensions that $1 < F_e \leq 10$ and $0.01 < F_c \leq 1$ have a minor impact on the performance of the optimization. In the feasible region identification, small $F_c \simeq 0.1$ improves feasible rate N_s/N_f. In this paper, we did not investigate the effect of F_e and F_c closely, but they merit further investigation in the future.

Table 3.C.1 summarizes the settings for DE in the Table 3.1. For the set up of Table 3.C.1,we have consulted section 3.4.1 of [9]. The NP for Rosenbrock, Rotated Ellipsoid and Manevich was set manually by several trial runs. The CRs are set high at 0.9 if a test function is non-separable and 0.2 if it is separable. The Fs were set to 0.5 for all the functions for fast reduction of objective values. The same setting was kept for DE in Table 3.3 and on.

Table 3.C.1 Parameters setups for DE for the five test functions in Table 3.1

Function	NP	CR	F
Rosenbrock	45	0.9	0.5
Rastrigin	35	0.2	0.5
Rotated Ellipsoid	30	0.9	0.5
Ackley	20	0.2	0.5
Manevich	30	0.2	0.5

In the optimization of 30 dimensional functions, NT in SOMBAS was set equal to NP in DE. This is partly because we didn't know the best algorithm parameter setting for SOMBAS and good setting was known for Ackley, Rastrigin and Rosenbrock from [9]. However, we modified the DE setting for Rosenbrock slightly from what is given in the book, $CR = 0.9, F = 0.8, NP = 60$ to the values seen in Table 3.C.1. From the trial runs we found that our setting gave smaller objective values. Table 3.C.2shows the complete setup.

Table 3.C.2 Parameters setups for SOMBAS for the five test functions in Table 3.1

Function	L	ρ	NT	SOM size	T	$P_{mutation}$	F_e	F_c
Rosenbrock	None	2.0	45	7×7	1.0	1.0	2.0	1.0
Rastrigin	None	1.0	35	6×6	1.0	1.0	1.5	1.0
Rot. Ellip.	None	2.0	30	5×5	1.0	1.0	1.1	1.0
Ackley	None	1.0	20	4×4	1.0	1.0	1.5	1.0
Manevich	None	2.0	30	6×6	1.0	1.0	2.0	1.0

Table 3.C.3 shows the parameter set up for SOMBAS in Table 3.3. Given our empirical knowledge from the two 30 dimensional tests in Table 3.1 and 3.2, NT

was set to a small number, 10 for faster calculation and T to a generous value 4.0 to compensate for the small NT. The setting was the same for all five functions. The CMA-ES parameter that we touched to produce the results in Table 3.3 was the

Table 3.C.3 Parameters setups for SOMBAS for the five test functions in Table 3.3 and 3.4

L	ρ	NT	SOM size	T	$P_{mutation}$	F_e	F_c
None	0.01	10	5×5	4.0	0.25	1.5	1.0

initial sigma of the diagonal covariance matrix. We used $\sigma_0 = 5.0$. The population was automatically determined by the algorithm as $4 + \lfloor 3 \log(D) \rfloor$.

References

[1] A. J. Keane and P. B. Nair. *Computational Approach for Aerospace Design*. John Wiley & Sons, 2005.

[2] I. Couckuyt, F. D. Turck, T. Dhaene, and D. Gorissen. *Automatic Surrogate Model Type Selection During the Optimization of Expensive Black-Box Problems*. In Proceedings of the 2011 Winter Simulation Conference, pages 4274 – 4284, 2011.

[3] D. R. Jones, M. Schonlau, and W. J. Welch. *Efficient Global Optimization of Expensive Black-Box Functions*. Journal of Global Optimization, 13(4):455–492, December 1998. Available from: http://dx.doi.org/10.1023/A:1008306431147, doi:10.1023/A:1008306431147.

[4] B. Liu, Q. Zhang, and G. G. E. Gielen. *A Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Medium Scale Expensive Optimization Problems*. IEEE Transaction on Evolutionary Computation, 18(2):180 – 192, April 2014. doi:10.1109/TEVC.2013.2248012.

[5] F. Boschetti. *A local linear embedding module for evolutionary computation optimization*. Journal of Heuristics, 14(1):95–116, 2008. Available from: http://dx.doi.org/10.1007/s10732-007-9030-6, doi:10.1007/s10732-007-9030-6.

[6] K. Ito, T. Dhaene, N. E. Masri, R. d'Ippolito, and J. V. de Peer. *Self-Organizing Map Based Adaptive Sampling*. In Proceedings of 5th International Conference on Experiments/Process/System Modeling/Simulation/Optimization (5th IC-EpsMsO), volume II, pages 504 – 513, Athens, Greece, July 3 - 6 2013. ISBN:978-618-80527-2-7 or 978-618-80527-0-3.

[7] K. Ito, I. Couckuyt, R. d'Ippolito, and T. Dhaene. *Design Space Exploration using Self-Organizing Map Based Adaptive Sampling*. Applied Soft Computing, 43:337 – 346, 2016. Available from: http://www.sciencedirect.com/science/article/pii/S1568494616300874, doi:http://dx.doi.org/10.1016/j.asoc.2016.02.036.

[8] R. Storn and K. Price. *Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces*. Technical Report TR-95-012, International Computer Science Institute, 1947 Center Street, Berkeley, CA 94704, USA, 1995.

[9] K. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution: APractical Approach to Global Optimization*. Springer, 2005.

[10] N. Hansen and A. Ostemeier. *Adapting Arbitrary Normal Mutation Distributions in Evolution Strategies: The Covariance Matrix Adaptation*. In Proceedings of the 1996 IEEE Conference on Evolutionary Computation, pages 312 – 317, 1996.

[11] R. M. Storn. *Differential Evolution Homepage*, 2014. Available from: http://www1.icsi.berkeley.edu/~storn/code.html.

[12] N. Hansen. *The CMA Evolution Strategy*, January 2014. Available from: https://www.lri.fr/~hansen/cmaesintro.html.

[13] M. Johnson, L. Moore, and D. Ylvisaker. *Minimax and maximin distance designs*. Journal of Statistical Planning and Inference, 26:131 – 148, 1990.

[14] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. *Design and Analysis of Computer Experiments*. Statistical Science, 4(4):409 – 435, 1989.

4
Interaction Index

⋆ ⋆ ⋆

K. Ito, I. Couckuyt, S. Poles, T. Dhaene.

“ Variance-Based Interaction Index Measuring Heteroscedasticity ”.

Published in Computer Physics Communications, DOI 10.1016/j.cpc.2016.02.032, vol. 203, pp. 152 - 161, June 2016.

⋆ ⋆ ⋆

This work is motivated by the need to deal with models with high-dimensional input spaces of real variables. One way to tackle high-dimensional problems is to identify interaction or non-interaction among input parameters. We propose a new variance-based sensitivity interaction index that can detect and quantify interactions among the input variables of mathematical functions and computer simulations. The computation is very similar to first-order sensitivity indices by Sobol'. The proposed interaction index can quantify the relative importance of input variables in interaction. Furthermore, detection of non-interaction for screening can be done with as low as $4n+2$ function evaluations, where n is the number of input variables. Using the interaction indices based on heteroscedasticity, the original function may be decomposed into a set of lower dimensional functions which may then be analyzed separately.

4.1 Introduction

In today's engineering, computer simulations are widely used to understand the behavior of complex systems and to optimize their input variables to obtain satisfactory designs before actual physical prototypes are built. The simulators are usually black-box or too complex to render a mathematical approach feasible. Sensitivity analysis enables us to understand how the changes in input variables affect the variance of the output.

As a part of the sensitivity analysis, identifying interacting and additive-effect variables is important in design optimization and engineering analysis of black box models. Two input variables are said to interact if their effect on the output cannot be expressed as a sum of their single effects. If the variable is additive (non-interacting), that variable can be treated independently from other variables. Then, we can separate our effort between the analysis of the interacting part which is often the more subtle and difficult part, and the analysis of the additive effect part. In this study, we will treat a methodology to detect and quantify this interaction of input variables to deterministic black-box models.

A widely recognized way of quantifying interaction is by calculating the difference between total effect indices and first order sensitivity indices in variance-based global sensitivity analysis [1–3]. In practice, the effectiveness of this method hinges on the accuracy of the sensitivity indices, which may demand a very high number of Monte Carlo sampling.

On the other hand, there are one-at-a-time methods often used for screening important variables by estimating average partial derivative magnitudes of the outputs obtained from factor levels (i.e., sampling on grid points) or as perturbations of Monte Carlo samples [4–7]. In these methods, interaction effects are not distinguished from non-linear effects of a particular input variable [4, 5, 7] or it is computed in a factorial design manner [6] with a preferred number of factors, for example 3,6,10,15,...

We propose an approach to decompose a high-dimensional problem into a set of lower dimensional problems via novel interaction indices which use the heteroscedasticity of marginal distributions. Heteroscedasticity refers to the circumstance in which the variability of a variable is unequal across the range of values of a second variable (a factor) that predicts it. Calculation of these interaction indices is a simple extension to Sobol indices [8] and gives information about particular variable(s) being non-interacting or interacting with other variables. The method uses Monte Carlo integration, but is very robust against loss of accuracy even when the number of random samples is modest. Due to this property, the proposed method can be used for both quantification and screening of interaction among input variables depending on the computational budget.

In the following discussions, $E(\ )$ denotes the expectation or the average value

of the variable inside the bracket. Likewise, $V(\,)$ denotes the variance. Sometimes, we put a subscript below V to clarify the source of the variance. We also employ an indexing convention $-i$ to denote "all other indices except i". For example,

$$\underset{x_{-i}}{V}\left(y|x_i\right)$$

means variance of y given x_i (so the variance of y comes from the variance of sources other than x_i, thus $\underset{x_{-i}}{V}$).

4.2 Sobol' Indices and High-Dimensional Model Representation (HDMR)

Consider a deterministic model

$$y = f\left(\mathbf{x}\right)$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a vector of n input variables and y is the model output. $f(\mathbf{x})$ can be decomposed into a form referred to as the high-dimensional model representation.

$$\begin{aligned} f\left(\mathbf{x}\right) &= f_0 + \sum_i f_i\left(x_i\right) + \sum_{i<j} f_{ij}\left(x_i, x_j\right) \\ &\quad + \sum_{i<j<k} f_{ijk}\left(x_i, x_j, x_k\right) + \ldots \end{aligned} \tag{4.1}$$

This decomposition of the function is not unique as the lower order can be selected arbitrarily and the highest order term can be written as the difference between $f(\mathbf{x})$ and the lower order terms. However, if the average of each of the term in the summands of the right hand side of equation (4.1) is set to zero (e.g., $\int f_i(x_i)dx_i = 0$) and f_0 is set to be a constant, the expression is proven to be unique [8]. The terms are given as follows:

$$f_0 = E\left(y\right) \tag{4.2}$$

$$f_i\left(x_i\right) = E\left(y|x_i\right) - f_0 \tag{4.3}$$

$$\begin{aligned} f_{ij}\left(x_i, x_j\right) &= E\left(y|x_i, x_j\right) - f_i\left(x_i\right) \\ &\quad - f_j\left(x_j\right) - f_0 \end{aligned} \tag{4.4}$$

$$\begin{aligned} f_{ijk}\left(x_i, x_j, x_k\right) &= E\left(y|x_i, x_j, x_k\right) - f_{ij}\left(x_i, x_j\right) \\ &\quad - f_{ik}\left(x_i, x_k\right) - f_{jk}\left(x_j, x_k\right) \\ &\quad - f_i(x_i) - f_j(x_j) - f_k(x_k) \\ &\quad - f_0. \end{aligned} \tag{4.5}$$

The first order sensitivity index for variable x_i is given by

$$S_i = \frac{\underset{x_i}{V}\left[E\left(y|x_i\right)\right]}{V\left(y\right)}, \tag{4.6}$$

and if we calculated the indices to the highest order, we have

$$\sum_{i=1}^{n} S_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} S_{ij} + \ldots + S_{1\ldots n} = 1. \tag{4.7}$$

The $S_{i\in\{1,2,\ldots,n\}}$ are called first-order Sobol' indices or sensitivity indices [2, 9].

The total effect index [1] includes interaction effects in addition to the first-order sensitivity indices, and can be defined as

$$S_{Ti} = 1 - S_{-i}, \tag{4.8}$$

where S_{-i} signifies the sum of all the sensitivity indices except those that include variances due to x_i. For example, if $i \in \{1, 2, 3\}$, the total effect index of x_1 is

$$\begin{aligned} S_{T1} &= S_1 + S_{12} + S_{13} + S_{123} \\ &= 1 - S_2 - S_3 - S_{23}. \end{aligned} \tag{4.9}$$

The total effect index defined in equation (4.8) is useful in variable screening. Variables with $S_{Ti} \simeq 0$ can be held constant at an arbitrary value within its lower and upper bounds since it means that the variable's value does not contribute to the variance in the output[1]. The first order sensitivity indices in equation (4.6) alone cannot be used for this purpose if there is a significant amount of interactions among the variables.

4.3 Computation

We now formulate a way to compute the first order sensitivity indices. It is also assumed that the function $f(\mathbf{x})$ is square integrable in $\mathbf{x} \in \Omega$ where Ω is a n-dimensional domain of integration of real variables. Uniform distributions are assumed on the inputs, and inputs are uncorrelated with each other. The total variance is therefore

$$\begin{aligned} D &= \underset{\mathbf{x}}{V}\left(f\left(\mathbf{x}\right)\right) \\ &= \frac{1}{\mathcal{V}} \int\limits_{\mathbf{x}\in\Omega} f^2\left(\mathbf{x}\right) d\mathbf{x} - f_0^2, \end{aligned} \tag{4.10}$$

[1] Strictly speaking, this holds only to a probability [8, Theorem 2].

where $\mathcal{V} = \int_{\mathbf{x}\in\Omega} d\mathbf{x}$ and $d\mathbf{x} = dx_1 dx_2 \ldots dx_n$. The average of $f(\mathbf{x})$ is given by

$$f_0 = \frac{1}{\mathcal{V}} \int\limits_{\mathbf{x}\in\Omega} f(\mathbf{x})\, d\mathbf{x}.$$

The multidimensional integral of equation (4.10) can be computed using Monte Carlo integration. Similarly,

$$f_i(x_i) = \frac{1}{\mathcal{V}_{-i}} \int\limits_{\mathbf{x}\in\Omega_{-i}} f(x_1, x_2, \ldots, x_n)\, dx_{-i} - f_0, \tag{4.11}$$

where $dx_{-i} = dx_1 dx_2 \ldots dx_{i-1} dx_{i+1} \ldots dx_n$, Ω_{-i} is the domain of integration with x_i fixed, and

$$\mathcal{V}_{-i} = \int_{\mathbf{x}\in\Omega_{-i}} dx_{-i}.$$

We also define $V(y|x_i)$ for later use in our discussion,

$$\underset{x_{-i}}{V}(y|x_i) = \frac{1}{\mathcal{V}_{-i}} \int\limits_{\mathbf{x}\in\Omega_{-i}} f^2(x_1, x_2, \ldots, x_n)\, dx_{-i} - f_0^2. \tag{4.12}$$

This is the variance of y when x_i is fixed at a certain value. Again, in equations (4.11) and (4.12), the integrations are performed using the Monte Carlo method, but this time x_i is held constant. By fixing x_i at various values, we can conduct the next integration to obtain $V[E(y|x_i)]$.

$$D_i = \underset{x_i}{V}[E(y|x_i)] = \underset{x_i}{V}(f_i(x_i)) = \frac{1}{\mathcal{V}_i} \int\limits_{x_i\in\Omega_i} f_i^2(x_i)\, dx_i \tag{4.13}$$

where Ω_i is the domain of integration for x_i, and $\mathcal{V}_i$ is the domain interval length of x_i. Then,

$$S_i = \frac{D_i}{D}. \tag{4.14}$$

The computation of $f_i(x_i)$ at different values of x_i to calculate D_i in equation (4.13) is a brute-force approach. It requires $m \times (n \times l + 1)$ function evaluations, where m is the number of Monte Carlo samples, n is the number of input variables, and l is the number of different x_i values that are used to compute equation (4.13). There is a more efficient method in which all S_i and S_{Ti} are calculated in $m \times (n + 2)$ function evaluations [10] provided that all input variables' distributions are independent.

Note that estimators have been recently developed to extend [10] to the case of correlated and dependent inputs [11–15].

4.4 Interactions in Reliability and Optimization

In the process of optimization, for example, minimizing y by judicious choice of x_i, one would also be interested in the variance of y given x_i, $V(y|x_i)$ or more generally, the distribution of y given x_i. Let us denote such distribution (or probability density function) as $p(y|x_i)$. This information can easily be obtained during the calculation of the first order Sobol' Indices. This information can be used in three ways. First, it tells you for what value of x_i one could possibly have the smallest y. Second, it tells you if x_i has any interaction with other variables. Finally, it tells you what value of x_i would satisfy certain reliability criteria. That is, one could draw a threshold value for y beyond which these variances should not exceed.

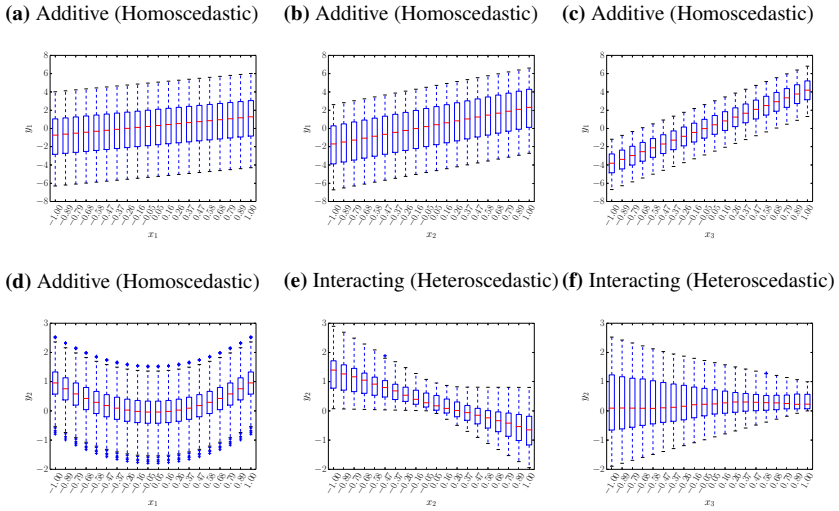
If some or all of the x_i contain uncertainties such that their intervals cannot be reduced beyond a certain level, the resulting $p(y|x_i)$s will represent the uncertainties in the output due to the uncertainties in these x_i. For reliability purposes, one may also be interested in $\max(y|x_i)$ which is the maximum y (that occurred in Monte Carlo simulations) given x_i.

Figure 4.1 shows an example of representing $p(y|x_i)$ as box plots. The example shows the spreads of two outputs y_o, $o \in \{1, 2\}$ i.e. $p(y_o|x_i)$ in vertical axes with respect to three input variables x_1, x_2, and x_3. We see by visual inspection that y_1 is composed of purely additive effects from x_1, x_2 and x_3 because all the spreads of $p(y_1|x_1)$, $p(y|x_2)$ and $p(y|x_3)$ as shown by the box sizes are constant across different values of x_1, x_2 and x_3, respectively (i.e. Homoscedastic behavior). If x_i produces an additive effect in the output, it should only cause a shift in the mean of $p(y|x_i)$ according to equation (4.1).

On the other hand, x_2 and x_3 have interactions in y_2 because the spreads of $p(y_2|x_2)$ and $p(y_2|x_3)$ are not constant. If the output is determined only by the three inputs, we can conclude that x_2 and x_3 interact with each other in y_2. The quadratic effect x_1 to y_2 is additive since the $p(y_2|x_1)$ stays constant. If a variable does not interact, it can be treated independently, with other variables fixed or vice versa, without any loss of information.

In the above example, the marginal spread of an output $p(y|x_i)$ was expressed as box plots as one would get from the brute-force approach, but the marginal scatter plot of y vs. x_i as one would obtain from the efficient computations [10] can also be informative for the three purposes above.

Figure 4.1 Illustrative Functions: the distributions $p(y|x_i)$ of equations 4.23 and 4.24.



4.5 Interaction Indices

In order to quantify the interaction of input variables, we propose the following interaction index,

$$I_i^2 = \frac{\underset{x_i}{V}\left[\underset{x_{-i}}{V}(y|x_i)\right]}{V^2(y)}, \tag{4.15}$$

or its square root form,

$$I_i = \frac{\sqrt{\underset{x_i}{V}\left[\underset{x_{-i}}{V}(y|x_i)\right]}}{V(y)}, \tag{4.16}$$

where we can compute $\underset{x_{-i}}{V}(y|x_i)$ from equation (4.12). We can then set a threshold ϵ below which we say that the input x_i does not have significant interaction with other input variables and thus can be treated independently. Note that the interaction index I_i is domain dependent. Even if the underlying function is the same, different Ω produce different values of I_i in general. For example, two input variables x_i and x_j, with $i \neq j$, may be interacting if varied substantially but may be non-interacting if varied by a small amount around certain points. The ϵ is typically very small near the arithmetic precision. Mathematically speaking, $I_i = 0$ for non-interacting input x_i and $I_i > 0$ for interacting x_i.

We can extend this concept to detect two and higher dimensional subproblems.

$$I_{ij}^2 = \frac{V\left[V(y|x_i,x_j)\right]}{V^2(y)}, \tag{4.17}$$

$$I_{ijk}^2 = \frac{V\left[V(y|x_i,x_j,x_k)\right]}{V^2(y)}, \tag{4.18}$$

$$\cdots$$

The indices I_{ij} can be interpreted as follows. Let i and j be the indices whose input variable has shown to have interaction with other input variables: $I_i > \epsilon,\ I_j > \epsilon$, and $i < j$. Then, $0 \leq I_{ij} \leq \epsilon$ means that input combinations specified by x_i and x_j produce an additive effect to the output y. This means that there is no higher order interaction for this particular pair of input variables x_i and x_j. In the HDMR expression in equation (4.1), it means that a term $f_{ij}(x_i, x_j)$ is not zero. On the other hand, $I_{ij} > \epsilon$ implies second or higher order interactions exist with some other input variables. For I_{ijk} and higher follows the same argument.

4.6 The Basic Idea Step by Step

To clarify the idea of using heteroscedasticity in detecting (non-)interactions, let us consider the following two equations

$$y_1 = x_1 + x_2, \tag{4.19}$$

$$y_2 = x_1 \cdot x_2. \tag{4.20}$$

We will carry out the brute-force calculation of

$$\underset{x_i}{V}\left[\underset{x_{-i}}{V}\left(y|x_i\right)\right]$$

step by step. The calculation will be done with $m = 2$ and $l = 2$.

Let us start with the (contrived) two sample points given in Table 4.1. In Table 4.2, the sample points were replaced with $x_1 = 1$ and in Table 4.3 with $x_1 = 3$. From Table 4.2 and Table 4.3 we can calculate $\underset{x_{-1}}{V}\left(y_1|x_1\right)$ and $\underset{x_{-1}}{V}\left(y_2|x_1\right)$ at the two x_1 locations, namely 1 and 3. These are tabulated in Table 4.4. Note that for y_1, its values were simply shifted by 2 if you compare Table 4.2 and Table 4.3. Thus, in Table 4.4, $\underset{x_{-1}}{V}\left(y_1|x_1\right)$ are identical at both $x_1 = 1$ and at $x_1 = 3$. This is

Table 4.1 Initial two samples

x_1	x_2	y_1	y_2
1	2	3	2
3	4	7	12

Table 4.2 x_1 fixed at 1

x_1	x_2	y_1	y_2
1	2	3	2
1	4	5	4

Table 4.3 x_1 fixed at 3

x_1	x_2	y_1	y_2
3	2	5	6
3	4	7	12

because the x_{-1} (x_2 in this case) values were identical in both tables and x_1 is an additive contribution for y_1. For y_2, the multiplicative contribution of x_1 renders different $\underset{x_{-1}}{V}(y_2|x_1)$ between $x_1 = 1$ and $x_1 = 3$ as observed in Table 4.4. With this heteroscedasticity, we say that x_1 and x_2 are interacting.

Thus, from Table 4.4 we compute

$$\underset{x_1}{V}\left[\underset{x_{-1}}{V}(y_1|x_1)\right] = 0, \tag{4.21}$$

$$\underset{x_1}{V}\left[\underset{x_{-1}}{V}(y_2|x_1)\right] = 16. \tag{4.22}$$

The same procedure can be repeated for x_2.

The column change at x_i leaves other columns x_{-i} unchanged (as observed in Table 4.2 and Table 4.3), thus if x_i contribution to an output y is additive, $V(y|x_i)$ remains unchanged throughout the different values in x_i. This suggests that for screening purposes, we can let the Monte Carlo samples very low, and in the example above we had $m = 2$, the minimum to compute a variance. Of course, at such low number, we cannot hope to have an accurate $V(y|x_i)$ because the distribution $p(y|x_i)$ will not be represented adequately. However, if x_i is non-interacting,

$$\underset{x_i}{V}\left[\underset{x_{-i}}{V}(y|x_i)\right]$$

should give zero to an arithmetic precision. As m is increased, $V(y|x_i)$ becomes accurate and a quantitative ordering of interaction among different input variables becomes possible. Furthermore, this "variance of variance" is never negative due to its sum-of-squares computations.

Table 4.4 Variances of y_1 and y_2 at $x_1 = 1, 3$

x_1	$\underset{x_{-1}}{V}(y_1|x_1)$	$\underset{x_{-1}}{V}(y_2|x_1)$
1	1	1
3	1	9

The normalizing factor $1/V(y)$ in equation (4.16) is rather arbitrary, one could have equally applied, for example,

$$\frac{1}{\sum_{i=1}^{n} \underset{x_i}{V}\left[\underset{x_{-i}}{V}(y|x_i)\right]}$$

to mimic probability measures. However, in our opinion, this would not add much to the intuitive appeal and we have opted for the simpler expression.

A physical interpretation of I_i is as follows. Consider a Dirac delta function $\delta(x_i - a)$, which is a distribution of x_i and has a probability mass of 1 at $x_i = a$ and zero anywhere else. The interaction index I_i shows the sensitivity (variance) of $\underset{x_{-i}}{V}(y|x_i)$ with respect to a when the original uniform distribution of x_i is replaced by $\delta(x_i - a),\ a \in \Omega_i$. Here, Ω_i is simply a real closed interval between upper and lower bounds of x_i. If you need to know which input variable x_i, if made deterministic, would make the uncertainty in the output y most different depending on its input value a, the indices can be useful.

One may wonder, given a dataset of unknown input distributions, if $I_i = 0$ would imply that the covariance between x_i and another x_j with $j \neq i$ would also be zero. However, this is not necessarily the case. An easy counter example is letting $x_2 \sim N(x_1, 1)$ in equation (4.19). That is, x_2 are drawn from a normal distribution with mean x_1 with a constant standard deviation $\sigma = 1$. In this case, $Cov(x_1, x_2) > 0$ but $I_i = 0$.

4.7 Comparison

It is also possible to evaluate interaction via the total effect indices and first order Sobol indices, $S_{Ti} - S_i$. However, there are some important differences between I_i and $S_{Ti} - S_i$.

First, $S_{Ti} - S_i$ gives the variance in expected values of output y due to x_i that are not due to the first-order terms of equation (4.3) but by the second-order terms of equation (4.4) or higher. So it is a combined effect of more than one input variables to obtain the average output, for example $E(y|x_i, x_j)$. Fixing x_i and x_j with different combinations of values generates $V[E(y|x_i, x_j)]$ to obtain S_{ij}. In contrast, I_i is a "first-order" index. Fixing x_i at various values generates various $V(y|x_i)$ to obtain $V[V(y|x_i)]$. For example, consider again Figure 4.1. From Figure 4.1e and Figure 4.1f one would guess $I_2 < I_3$ because by visual inspection, the difference in variance given a specific value in x_i is greater for $V(y_2|x_3)$ than $V(y_2|x_2)$. $S_{Ti} - S_i$ does not give information about the relative importance between x_i and x_j in driving the variance of y_2. On the other hand, I_i does not distinguish the additive effect and the interaction effect of a single input

variable. If x_i interacts, it does not by itself give any indication of the elementary effect that it may have as in S_i.

Second, the detection of non-interaction $I_i = 0$ is not sensitive to the accuracy of $V(y|x_i)$. As long as $V(y|x_i)$ is computed with the same samples in x_{-i}, $V(y|x_i)$ remains constant throughout different values of x_i if x_i gives only an additive effect to the output. In other words, if we have a matrix with m rows of Monte Carlo samples with n columns corresponding to the number of input variables and replace column i with a value for x_i, and compute the corresponding outputs to obtain $V(y|x_i)$, this variance is identical regardless of the value of x_i when variable x_i is not interacting with other input variables. Thus, I_i should show zero to arithmetic precision if x_i does not interact with other variables. If the typical output variance $V(y|x_i)$ is in the order of 10^0, non-interaction would typically produce $V[V(y|x_i)] \simeq 10^{-16}$ when computations are done in double precision. On the other hand, $S_{Ti} - S_i$ is subject to the Monte Carlo integration inaccuracy.

For first order sensitivity calculations, m is typically in the order of 1000 or above and l is typically 50 or above in our experience. However, for screening purposes I_i can be computed with m and l as low as 2 giving $4n + 2$ function evaluations. We need two samples to compute the output variances at two different values of an input variable and check that the variances of the two output values do not change with respect to the values of the input variable.

Lastly, for quantitative uses, the computation of I_i does not require any further function evaluation (i.e., computation of response y) beyond what is required for the computation of first order Sobol indices S_i in brute-force approach. Computing S_{Ti} in brute-force approach is often infeasible (requiring computation of up to $n - 1$ order Sobol indices), but efficient ways exist [10]. Furthermore, surrogate modeling techniques that facilitate the acquisition of S_i and S_{Ti} exist such as using Polynomial Chaos Expansions [16–18]. We expect that there are shortcuts to economize the computation of I_i as well. This is an open research topic and future work.

4.8 Examples

In this section, five functions will be analyzed using the proposed interaction indices and the conventional method of using the difference between total effects and first order sensitivity indices of Sobol'. The inputs will be assumed to be random variables with uniform distributions between upper and lower bounds. The numerical results and plots were obtained using a 32-bit version of Python 2.7.5, Numpy 1.8.0, Scipy 0.13.2, and Matplotlib 1.3.1.

The $S_{Ti} - S_i$ is calculated using the methods described in 4.A. The I_i is calculated using the "brute-force" approach. In both $S_{Ti} - S_i$ and I_i, uniform random sampling is used for Monte Carlo integrations.

4.8.1 Illustrative Functions

Consider the following simple example.

$$y_1 = x_1 + 2x_2 + 4x_3 \tag{4.23}$$
$$y_2 = x_1^2 - x_2 + x_2x_3 \tag{4.24}$$

where $-1 < x_1, x_2, x_3 \leq 1$. Figure 4.1 shows marginal distributions as box plots. For these plots, Monte Carlo sampling was performed using the brute-force approach with $m = 200$ and $l = 20$.

The interaction indices are shown in Table 4.1. The zero entries in Table 4.1

Table 4.1 First order interaction indices for the Illustrative Functions

	y_1	y_2
I_1^2	0.000	0.000
I_2^2	0.000	0.036
I_3^2	0.000	0.573

indicate that corresponding variables do not interact with other variables. For y_2, x_1 is non-interacting, but x_2 and x_3 are interacting.

Table 4.2 shows the result of calculating $S_{Ti} - S_i$ with $m = 30200$. The column for y_1 and the entry for x_1 under the column for y_2 show zeros if we round to the second decimal place. For the y_2 column, the entry for x_2 and x_3 show the

Table 4.2 $S_{Ti} - S_i$ for the Illustrative Functions

	y_1	y_2
$S_{T1} - S_1$	0.00	0.00
$S_{T2} - S_2$	0.00	0.21
$S_{T3} - S_3$	0.00	0.21

interaction. Equations (4.25) to (4.27) show the expressions of $S_{Ti} - S_i$ for y_2. The reason that

$$S_{T2} - S_2 = S_{T3} - S_3$$

in Table 4.2 can be understood from equations (4.26) and (4.27).

$$S_{T1} - S_1 = S_{12} + S_{13} + S_{123} = 0, \tag{4.25}$$
$$S_{T2} - S_2 = S_{12} + S_{23} + S_{123} = S_{23}, \tag{4.26}$$
$$S_{T3} - S_3 = S_{13} + S_{23} + S_{123} = S_{23}, \tag{4.27}$$

because $S_{12} = S_{13} = S_{123} = 0$. The difference between Table 4.1 and Table 4.2 illustrates the difference between the two methods of detecting interactions and

non-interactions. The reason for $I_2 < I_3$ in Table 4.1 can be understood by factoring equation (4.24) as in equation (4.28),

$$y_2 = x_1^2 + x_2 \cdot (-1 + x_3). \tag{4.28}$$

For the given upper and lower bounds of x_2 and x_3, we have $-2 < -1 + x_3 \leq 0$ and $-1 < x_2 \leq 1$. Thus, if we sample x_2 and x_3 uniformly between -1 and 1, we have the following. If we let $x_2 = 1$ or -1, then we get the largest $V(y_2|x_2)$ with

$$V[x_2 \cdot (-1 + x_3)|x_2 = \pm 1] = \frac{1}{3}. \tag{4.29}$$

On the other hand, if we let $x_3 = -1$, then

$$V[x_2 \cdot (-1 + x_3)|x_3 = -1] = \frac{4}{3}, \tag{4.30}$$

and $V(y_2|x_3)$ is largest. Furthermore,

$$V[x_2 \cdot (-1 + x_3)|x_2 = 0] = 0, \tag{4.31}$$

$$V[x_2 \cdot (-1 + x_3)|x_3 = 1] = 0. \tag{4.32}$$

Thus,

$$\frac{I_3^2}{I_2^2} = \frac{V[V(y_2|x_3)]}{V[V(y_2|x_2)]} = \frac{4^2}{1^2} = 16, \tag{4.33}$$

which confirms Table 4.1.

4.8.2 Ishigami Function

Ishigami function [19, 20] is a three-variable function with an interaction between two of its input variables.

$$y_1 = \sin x_1 + a \sin^2 x_2 + b x_3^4 \sin x_1 \tag{4.34}$$

where $-\pi < x_1, x_2, x_3 < \pi$. In this paper, we set $a = 7$ and $b = 0.1$.

Figure 4.1 confirms visually that x_1 and x_3 are the interacting variables with their heteroscedastic behaviors. Figure 4.2 shows the distribution of values of $S_{Ti} - S_i$ and I_i of 20 independent runs for the function (4.B). As stated before, $S_{Ti} - S_i$ is calculated with $m \times (n + 2)$, and I_i with $m \times (n \times l + 1)$ function evaluations. Three different settings are tried out with different values for m and n. The difference in the values of m between the two methods is to make two methods perform about the same number of function evaluations.

As can be observed in Figure 4.2a to Figure 4.2c, $S_{Ti} - S_i$ loses accuracy as m becomes smaller. At a low number of m such as in Figure 4.2c, it would be impossible to detect interactions happening between x_1 and x_3 or the additive

Figure 4.1 Ishigami Function: distributions of $p(y|x_i)$ or the marginal views

(a) Interacting (Heteroscedastic) **(b)** Additive (Homoscedastic) **(c)** Interacting (Heteroscedastic)

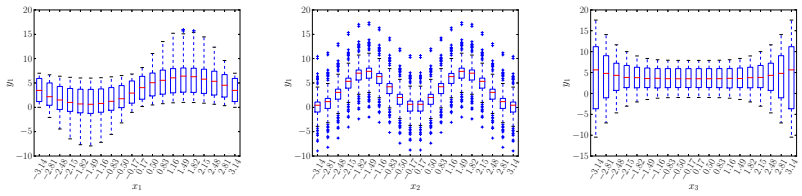
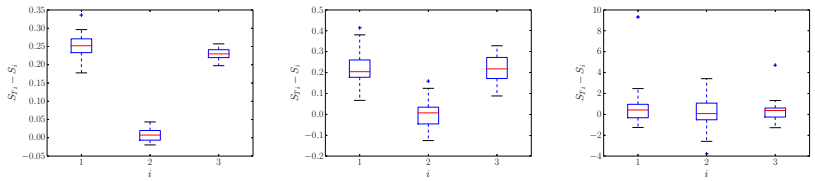


Figure 4.2 Ishigami Function: box plots show the distribution of indices values of 20 runs.

(a) $m = 2440$ **(b)** $m = 244$ **(c)** $m = 3$



(d) $m = 200, l = 20$ **(e)** $m = 20, l = 20$ **(f)** $m = 2, l = 2$

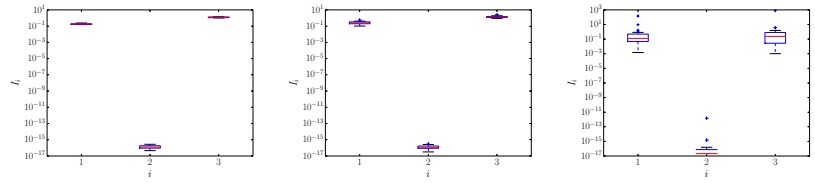
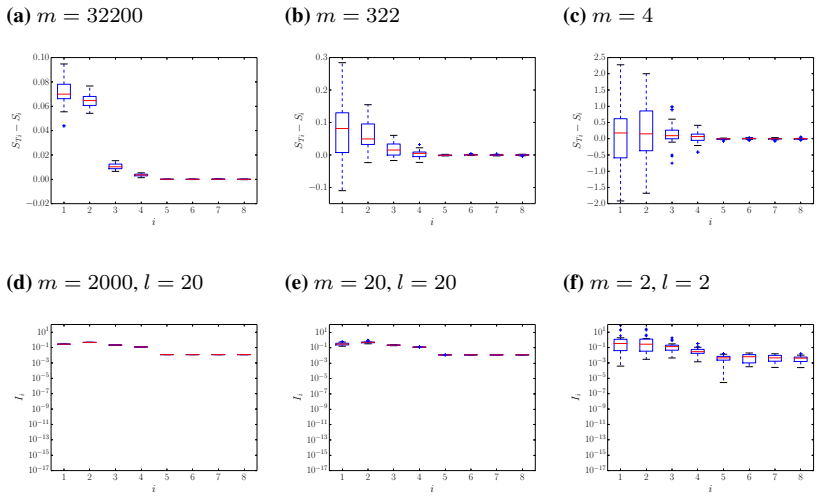


Figure 4.3 G Function: box plots show the distribution of indices values of 20 runs.



effect of x_2. On the other hand, for I_i as in Figure 4.2d to Figure 4.2f, even though their values become less accurate as m is decreased, the non-interaction of x_2 can clearly be detected by setting a threshold ϵ, for example $\epsilon = 10^{-9}$. We also see the relative importance of x_3 compared to x_2 in Figure 4.2d and Figure 4.2e in terms of variance of distribution $p(y|x_i)$ or $\underset{x_i}{V}[\underset{x_{-i}}{V}(y|x_i)]$.

4.8.3 G-Function

Sobol's G-function [10, 21, 22] is a test function for which global sensitivity can be controlled via its parameters. We use an eight-dimensional setting described in [22].

$$y_1 = \prod_{i=1}^{n} g_i(x_i) \tag{4.35}$$

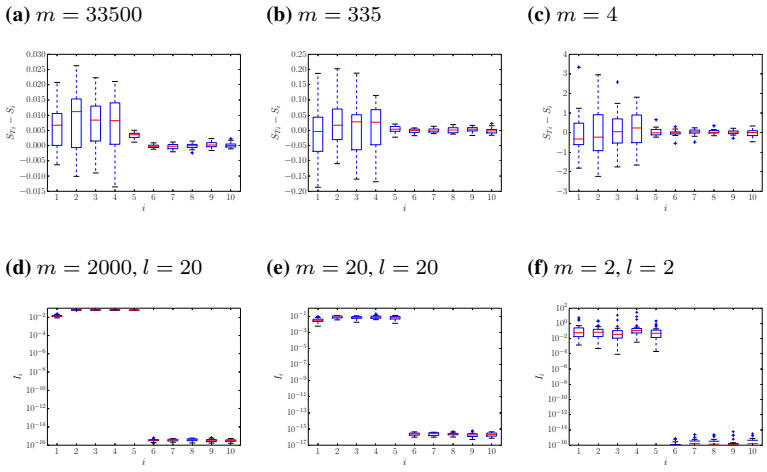
where

$$g_i(x_i) = \frac{|4x_i - 2| + a_i}{1 + a_i},\ 0 \leq x_i < 1, \tag{4.36}$$

with $n = 8$, and $\{a_i\} = \{0, 1, 4.5, 9, 99, 99, 99, 99\}$. For x_i with $a_i = 0$, the variable is very important. On the other hand, if $a_i = 99$, x_i's effect is negligible, but still interacting with other variables.

In Figure 4.3, we see that I_i cannot reliably quantify relative importance of each variable when m is very low as observed in Figure 4.3f. However, we can still see that all the variables from x_1 to x_8 that they are all interacting since

Figure 4.4 Rosenbrock - Sphere Function: box plots show the distribution of indices values of 20 runs.



$I_i > 10^{-6} \gg 10^{-16}$. On the other hand, Figure 4.3c shows that $S_{Ti} - S_i$ is too unreliable at this number of samples.

4.8.4 Rosenbrock - Sphere Function

This function is simply a combination of two famous functions. We set the first five dimensions to be the inputs to the Rosenbrock function and the remaining five to be the inputs to the sphere function.

$$y_1 = \sum_{i=1}^{\lfloor n/2 \rfloor - 1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] + 1000 \sum_{i=\lfloor n/2 \rfloor + 1}^{n} x_i^2 \tag{4.37}$$

where $-10 \leq x_i \leq 10$, and $n = 10$.

For this function, x_6 to x_{10} have no interactions while the first five variables have interactions. We see in Figure 4.4a that $S_{Ti} - S_i$ cannot provide a reliable quantitative information about interaction, even at fairly high number of function evaluations: $33500 \times (10 + 2) = 402000$. The small interaction values make it difficult to be detected under Monte Carlo integration accuracy. On the other hand, Figure 4.4d to Figure 4.4f show that, for I_i, non-interacting variables remained discernible, even if the accuracy of indices deteriorated (as evidenced by the increase in the spread of box plots).

4.8.5 Artery Simulation

In this example, we investigate an application of our method to a physics-based problem of parameter identification. The simulation code we use was developed by [23]. This code has recently been used as an example problem for Gradient Enhanced Kriging [24] since the function exposes the gradient as well as the objective value. For our purpose, we will ignore the gradients and treat it as a scalar function with vector input consisting of the parameters we want to identify.

The code simulates the hemodynamics of the arterial system as one-dimensional fluid-structure interaction problem. Figure 4.5 shows a schematic of an axisymmetric model of the artery system along with its boundary conditions. The modeled blood flow in an artery is the unsteady flow of an incompressible, inviscid fluid, in a straight, flexible tube. The flow rate at the inlet is prescribed as a function of time. The outlet has velocity extrapolated using the velocities of the last two segments and relates to the output pressure using the Windkessel model [25, 26]. The Windkessel model represents the hemodynamics of the circulation downstream. Its dynamics is expressed using an electrical circuit analogy. A so-called generalized string model is applied to the structure. This is a linear elasticity theory for a thin cylindrical tube with membrane deformations [27, 28].

In this exercise, the inputs x_i are the modulus of elasticities of the artery at $n-1$ segments and the value of capacitance of the downstream boundary condition, totaling n input variables. We let $n = 20$. The output y is the sum of squared error between the simulated values and reference values (a priori obtained by the same simulation code in this example) of the radii of the artery at these segments. The sum y is over all time steps and all artery segments. This sum y is normalized so that it will not exceed 1. Exact match in time histories of radii between the given reference values and the simulation would give zero in the output. In a real situation, the reference values of radii would come from non-invasive measurements such as from ultrasound imagery.

Thus, the function we are analyzing can be expressed as

$$y = f(\mathbf{T}, \mathbf{x}),$$

where $\mathbf{T}$ is the matrix containing reference values of the radii of all $n-1$ segments for all timesteps, and $\mathbf{x}$ is the vector containing input variables x_i. The $\mathbf{T}$ is given, and we sample $\mathbf{x}$ to see whether its elements interact to obtain the output y. We pretend that we do not know the input $\mathbf{x}$ that generated the reference time histories of the radii $\mathbf{T}$, but have a rough idea to form the domain of the function. Specifically we create a $\pm$ 50% interval around nominal values E_0 and C_0 (Table 4.3). We investigate how the input variables interact to produce (the sum of errors in) the output. The inputs x_i are scaled to take values between -1 and 1 such as the

following.

$$E_i = E_0\left(1+\frac{1}{2}x_i\right),\ i\in\{1,\cdots,n-1\}, \tag{4.38}$$

$$C = C_0\left(1+\frac{1}{2}x_n\right), \tag{4.39}$$

where E_i is the modulus of elasticity of $n-1$ artery segments and C is the capacitance in the Windkessel model representing the compliance of the arterial system. Table 4.3 show the parameter values used in the artery model.

Figure 4.5 The diagram of an artery model with blood flowing in from left with prescribed time-dependent velocities and flowing out at the right with the Windkessel model pressure. The segments (eight in the figure), the radius r, the wall thickness h and the length l are shown. The prescribed inlet flow rate is given by $u_0(t) = 0.23 + 0.21\sin\left(2\pi\frac{t}{t_b}\right) + 0.11\cos\left(4\pi\left(\frac{t}{t_b}-0.2\right)\right) + 0.07\cos\left(6\pi\left(\frac{t}{t_b}-0.2\right)\right)$, where t_b is the pulse period.

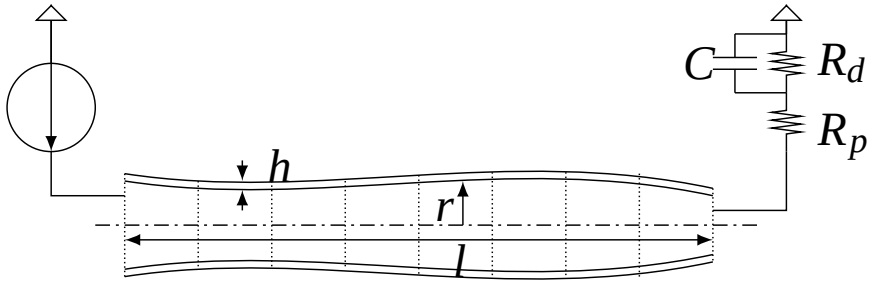


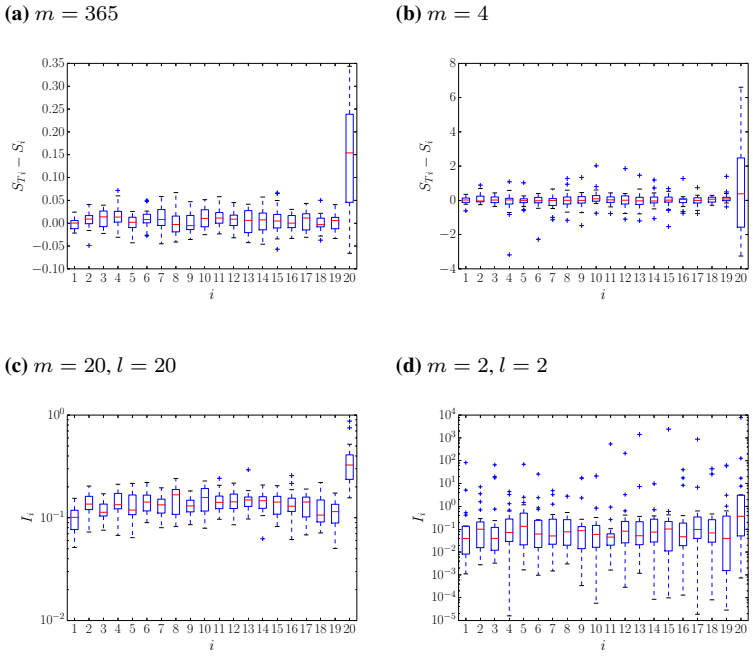
Table 4.3 Parameter values used in the artery model

r_0	3×10^{-3} m	E_0	4×10^{5} Pa
h	3×10^{-4} m	C_0	6.35×10^{-10} m^3/Pa
l	0.126 m	R_d	1.768×10^{9} Pa s/m^3
t_b	1 s	R_p	2.834×10^{8} Pa s/m^3

The results of computations of $S_{Ti}-S_i$ and I_i are shown in Figure 4.6. The computation time to obtain Figure 4.6a and Figure 4.6c combined was 233412 seconds or approximately 65 hours on a laptop computer with Intel Core2 Duo 2.8 GHz CPU and 4.0 GB RAM. For the computation of Figure 4.6b and Figure 4.6d combined, the elapsed time was 2552 seconds or about 43 minutes on the same computer.

For this problem, one would expect that all parameters have interactions. However, $S_{Ti}-S_i$ values in Figure 4.6a and Figure 4.6b were not consistent enough

Figure 4.6 Artery fluid-structure simulation for model calibration of 19 elasticity parameters ($i \in \{1, \cdots, 19\}$) and a downstream compliance parameter (the capacitance, $i = 20$): box plots show the distribution of index values of 20 runs.



throughout the 20 runs to show the interactions of elasticities of the arteries, with many of the indices showing below zero values. On the other hand, we obtained $I_i > 10^{-6} \gg 10^{-16}$ in Figure 4.6c and Figure 4.6d and one would be able to confirm the interactions.

The capacitance or the compliance parameter C (at $i = 20$) gave markedly higher values for both $S_{Ti} - S_i$ and I_i in Figure 4.6a and Figure 4.6c. This can be understood from the fact that the parameter is part of the downstream boundary condition affecting the time histories of radii of all the 19 upstream segments. In either case, the spread of the boxes indicates that the numbers of m in Figure 4.6a and Figure 4.6c were not large enough to show the relative importance of interactions among the elasticities of the artery segments. The same holds for smaller m. With Figure 4.6b and Figure 4.6d, neither $S_{Ti} - S_i$ nor I_i were able to capture reliably the salient importance of the capacitance parameter ($i = 20$).

4.9 Discussion and Outlook

As can be seen in Figure 4.2 and Figure 4.3, I_i and $S_{Ti} - S_i$ do not necessarily give a consistent ranking of importance (i.e., importance ordering of interacting input variables differ between the two methods). This is due to the fact that the indices evaluate the interaction in different ways as explained in Section 4.6 and 4.7. In I_i, it quantifies how sensitive the output variance is if we fix x_i to a value "a" rather than another value "b", for example. On the other hand, $S_{Ti} - S_i$ quantifies the uncertainty in output remaining after subtracting the main effect uncertainty. Therefore, if the parameter x_i is uncertain by nature the more relevant interaction measure would be $S_{Ti} - S_i$. However, if we can turn x_i into a deterministic variable and we can choose its value, I_i can give an appropriate measure. The implication of this difference merits further studying.

In practical situations in which the calculation of y given an input vector $\mathbf{x}$ is expensive, the computation of variance based sensitivity indices and quantitative interaction analysis of input variables may be prohibitive due to the number of model evaluations needed to do the Monte Carlo integration. In such cases, fitting surrogate models to the dataset computed by the original model may be useful. Surrogate models [29, 30] are approximations to the original function and are much cheaper to compute than the original model. It is usually fitted to a finite number of input-output data obtained from the original model (usually a complex simulation model). Kriging and Radial Basis Functions are some of the popular surrogate models performing interpolations.

There are also regression methods based on HDMR [31–33]. The basis functions in these are polynomials. The representations are usually truncated at second order or so, thus ignoring higher order terms and interactions. Let us denote the output produced by the surrogate model as $\hat{y}$. We can compute the indices based on $\hat{y}$'s. However, information about interactions may be inaccurate due to the approximate nature of the surrogate model. Furthermore, interpolating surrogate models are usually not very scalable to high-dimensional problems. Our proposed method could be applied to the high-fidelity model for screening purposes, and potentially for determining what interaction terms to include in HDMR based regressions. The same method could then be applied for quantitative purposes in the reduced problem (possibly on a surrogate model). Further research would be beneficial to see the actual merit of this approach.

4.10 Conclusion

The interaction index exposes each variable's importance in influencing the variance in the output through interaction. Its accuracy does not directly depend on the accuracy of the Monte Carlo integration, but on the change in the sample marginal

distribution or heteroscedasticity. The examples showed its robustness in detecting and quantifying interactions among input variables. This is expected to be useful in (robust) optimization and surrogate modeling typical in engineering analysis and design. Further application to industrial problems is needed to understand the effectiveness of the proposed index. Also, further research would be useful to exploit the concept described in this paper to develop a surrogate model assisted optimization algorithm that is scalable to high-dimensional problems.

Acknowledgments

Keiichi Ito has been funded by the Institute for the Promotion of Innovation through Science and Technology (IWT) through the Baekeland Mandate program. Ivo Couckuyt is a post-doctoral research fellow of FWO-Vlaanderen. This research has also been funded by the Interuniversity Attraction Poles Programme BESTCOM initiated by the Belgian Science Policy Office.

Appendix

4.A Monte Carlo Estimation of Indices

In the following, we give the Monte Carlo estimation of the indices. The method for I_i follows the so called "brute-force" method that would entail a double loop in a computer code. We consider an n-dimensional unit hypercube domain for notational brevity. Let $\mathbf{A}$ and $\mathbf{B}$ be two matrices with uniform random value elements between 0 and 1. The two matrices have the size of m rows and n columns. Let j and i be row and column indices, respectively. The notation ${\mathbf{A}_\mathbf{B}}^{(i)}$ means that all columns are from $\mathbf{A}$ except column i which is from $\mathbf{B}$. For total variance of output y, we can use

$$D \simeq \frac{1}{2m-1} \sum_{j=1}^{2m} f(\mathbf{C})_j^2 - f_{0_C}^2 \tag{4.40}$$

where $\mathbf{C}$ is the concatenated matrix of $\mathbf{A}$ and $\mathbf{B}$ with $2m$ rows and n columns and f_{0_C} is the mean of $f(\mathbf{C})_j$, or alternatively:

$$D_A \simeq \frac{1}{m-1} \sum_{j=1}^{m} f(\mathbf{A})_j^2 - f_{0_A}^2, \tag{4.41}$$

$$D_B \simeq \frac{1}{m-1} \sum_{j=1}^{m} f(\mathbf{B})_j^2 - f_{0_B}^2, \tag{4.42}$$

$$D_{AB} \simeq \frac{1}{2m-1} \sum_{j=1}^{2m} f(\mathbf{C})_j^2 - \frac{1}{m-1} \sum_{j=1}^{m} f(\mathbf{A})_j f(\mathbf{B})_j, \tag{4.43}$$

where f_{0_A} and f_{0_B} are the mean of $f(\mathbf{A})_j$ and $f(\mathbf{B})_j$, respectively. In our calculation of S_i and S_{Ti}, we used equation (4.40). The best-practice [10] recommends to compute the D_i in equation (4.13) as in the following.

$$D_i \simeq \frac{1}{m-1} \sum_{j=1}^{m} f(\mathbf{B})_j \left(f({\mathbf{A}_\mathbf{B}}^{(i)})_j - f(\mathbf{A})_j \right) \tag{4.44}$$

Thus, first-order sensitivity index is

$$S_i = \frac{D_i}{D}.$$

For total effects,

$$S_{Ti} \simeq \frac{1}{2D(m-1)} \sum_{j=1}^{m} \Big(f(\mathbf{A})_j - f(\mathbf{A_B}^{(i)})_j \Big)^2. \tag{4.45}$$

Let k be the index of l levels of x_i. We designate kth level of x_i as x_{ik} and matrix $\mathbf{A}$ with ith column replaced by element x_{ik} as $\mathbf{A}^{(i)}_{x_{ik}}$. Then, interaction indices can be computed from

$$\begin{aligned} \underset{x_i}{V} \left[\underset{x_{-i}}{V} (y|x_i) \right] \simeq & \frac{1}{l-1} \sum_{k=1}^{l} \left(\frac{1}{m-1} \sum_{j=1}^{m} f(\mathbf{A}^{(i)}_{x_{ik}})^2_j - f^2_{o_A} \right)^2 \\ & - \mu^2_{V(f(A^{(i)}_{x_i}))} \end{aligned} \tag{4.46}$$

where $\mu_{V(f(A^{(i)}_{x_i}))}$ is the average variance of $f(A^{(i)}_{x_i})$ when x_i is varied through l levels. Then, I^2_i can be obtained by dividing the result from equation (4.46) by D^2_A.

4.B Sample Size for Box Plots

Our objective in the box plots was not to support any significance tests, but to show qualitatively the problems that may arise. The number of independent runs was not determined on statistically rigorous grounds, but by the desire to keep the computational costs to an easily manageable level. There seems to be no theoretical foundation of how large the sample size for box plots should be, except that it should be at least 5 [34]. There is no universally agreed method of computing the box boundaries. We employ the Tukey-Style box plots as implemented in the Python module Matplotlib, in which the whiskers extend up to 1.5 times the Inter Quartile Range. The choice of 20 independent runs of $S_{Ti} - S_i$ and I_i estimations to generate the box plots in Figure 4.2, 4.3, 4.4, and 4.6 was determined taking into account the guidelines given by [34] and [35].

References

[1] T. Homma and A. Saltelli. *Importance Measures in Global Sensitivity Analysis of Nonlinear Models*. Reliability Engineering and System Safety, 52:1 – 17, 1996.

[2] I. M. Sobol'. *Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates*. Mathematics and Computers in Simulation, 55:271–280, 2001.

[3] A. Saltelli. *Making Best Use of Model Evaluations to Compute Sensitivity Indices*. Computer Physics Communications, 145:280 – 297, 2002.

[4] M. D. Morris. *Factorial Sampling Plans for Preliminary Computational Experiments*. Technometrics, 33(2):pp. 161–174, 1991.

[5] F. Campolongo, A. Saltelli, and J. Cariboni. *From screening to quantitative sensitivity analysis. A unified approach*. Computer Physics Communications, 182(4):978 – 988, 2011. doi:http://dx.doi.org/10.1016/j.cpc.2010.12.039.

[6] A. Saltelli, F. Campolongo, and J. Cariboni. *Screening important inputs in models with strong interaction properties*. Reliability Engineering & System Safety, 94(7):1149 – 1155, 2009. Special Issue on Sensitivity Analysis. doi:http://dx.doi.org/10.1016/j.ress.2008.10.007.

[7] S. Kucherenko, M. Rodriguez-Fernandez, C. Pantelides, and N. Shah. *Monte Carlo evaluation of derivative-based global sensitivity measures*. Reliability Engineering & System Safety, 94(7):1135 – 1148, 2009. Special Issue on Sensitivity Analysis. doi:http://dx.doi.org/10.1016/j.ress.2008.05.006.

[8] I. M. Sobol'. *Sensitivity Estimates for Nonlinear Mathematical Models*. Mathematical Modeling and Computational Experiment, 1(4):407 – 414, 1993.

[9] K. Chan, A. Saltelli, and S. Tarantola. *Sensitivity Analysis of Model Output: Variance-Based Methods Make the Difference*. In Proceedings of the 1997 Winter Simulation Conference, pages 261–268, 1997.

[10] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola. *Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index*. Computer Physics Communications, 181(2):259 – 270, 2010. doi:http://dx.doi.org/10.1016/j.cpc.2009.09.018.

[11] T. A. Mara, S. Tarantola, and P. Annoni. *Non-parametric methods for global sensitivity analysis of model output with dependent inputs*. Environmental Modelling & Software, 72:173 – 183,

2015. Available from: http://www.sciencedirect.com/science/article/pii/S1364815215300153, doi:http://dx.doi.org/10.1016/j.envsoft.2015.07.010.

[12] T. A. Mara and S. Tarantola. *Variance-based sensitivity indices for models with dependent inputs*. Reliability Engineering & System Safety, 107:115 – 121, 2012. {SAMO} 2010. Available from: http://www.sciencedirect.com/science/article/pii/S0951832011001724, doi:http://dx.doi.org/10.1016/j.ress.2011.08.008.

[13] S. Kucherenko, S. Tarantola, and P. Annoni. *Estimation of global sensitivity indices for models with dependent variables*. Computer Physics Communications, 183(4):937 – 946, 2012. Available from: http://www.sciencedirect.com/science/article/pii/S0010465511004085, doi:http://dx.doi.org/10.1016/j.cpc.2011.12.020.

[14] E. Borgonovo. *A new uncertainty importance measure*. Reliability Engineering & System Safety, 92(6):771–784, 2007.

[15] E. Borgonovo, W. Castaings, and S. Tarantola. *Moment independent importance measures: new results and analytical test cases*. Risk Analysis, 31(3):404–428, 2011.

[16] B. Sudret. *Global sensitivity analysis using polynomial chaos expansions*. Reliability Engineering & System Safety, 93(7):964 – 979, 2008. Bayesian Networks in Dependability. Available from: http://www.sciencedirect.com/science/article/pii/S0951832007001329, doi:http://dx.doi.org/10.1016/j.ress.2007.04.002.

[17] R. Pulch, E. J. W. ter Maten, and F. Augustin. *Sensitivity analysis and model order reduction for random linear dynamical systems*. Mathematics and Computers in Simulation, 111:80 – 95, 2015. Available from: http://www.sciencedirect.com/science/article/pii/S037847541500004X, doi:http://dx.doi.org/10.1016/j.matcom.2015.01.003.

[18] G. T. Buzzard and D. Xiu. *Variance-Based Global Sensitivity Analysis via Sparse-Grid Interpolation and Cubature*. Communications in Computational Physics, 9(3):542 – 567, March 2011.

[19] T. Ishigami and T. Homma. *An importance quantification technique in uncertainty analysis for computer models*. In Uncertainty Modeling and Analysis, 1990. Proceedings., First International Symposium on, pages 398–403. IEEE, Dec 3 - 5 1990. doi:10.1109/ISUMA.1990.151285.

[20] I. Sobol' and Y. Levitan. *On the use of variance reducing multipliers in Monte Carlo computations of a global sensitivity index*. Computer Physics

Communications, 117:52 – 61, 1999. doi:http://dx.doi.org/10.1016/S0010-4655(98)00156-8.

[21] I. Sobol, S. Tarantola, D. Gatelli, S. Kucherenko, and W. Mauntz. *Estimating the approximation error when fixing unessential factors in global sensitivity analysis*. Reliability Engineering & System Safety, 92(7):957–960, July 2007. doi:http://dx.doi.org/10.1016/j.ress.2006.07.001.

[22] D. Gatelli, S. Kucherenko, M. Ratto, and S. Tarantola. *Calculating first-order sensitivity measures: A benchmark of some recent methodologies*. Reliability Engineering & System Safety, 94(7):1212 – 1219, 2009. Special Issue on Sensitivity Analysis. doi:http://dx.doi.org/10.1016/j.ress.2008.03.028.

[23] J. Degroote, M. Hojjat, E. Stavropoulou, R. W uchner, and K.-U. Bletzinger. *Partitioned solution of an unsteady adjoint for strongly coupled fluid-structure interactions and application to parameter identification of a one-dimensional problem*. Structural and Multidisciplinary Optimization, 47(1):77–94, 2013. Available from: http://dx.doi.org/10.1007/s00158-012-0808-2, doi:10.1007/s00158-012-0808-2.

[24] S. Ulaganathan, I. Couckuyt, T. Dhaene, J. Degroote, and E. Laermans. *Performance study of gradient-enhanced Kriging*. Engineering with Computers, pages 1–20, 2015. Available from: http://dx.doi.org/10.1007/s00366-015-0397-y, doi:10.1007/s00366-015-0397-y.

[25] I. E. Vignon-Clementel, C. Figueroa, K. Jansen, and C. Taylor. *Outflow boundary conditions for 3D simulations of non-periodic blood flow and pressure fields in deformable arteries*. Computer methods in biomechanics and biomedical engineering, 13(5):625–640, 2010.

[26] N. Westerhof, J.-W. Lankhaar, and B. E. Westerhof. *The arterial windkessel*. Medical & biological engineering & computing, 47(2):131–141, 2009.

[27] J.-F. Gerbeau and M. Vidrascu. *A quasi-Newton algorithm based on a reduced model for fluid-structure interaction problems in blood flows*. ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique, 37(4):631–647, 2003.

[28] A. Quarteroni, M. Tuveri, and A. Veneziani. *Computational vascular fluid dynamics: problems, models and methods*. Computing and Visualization in Science, 2(4):163–197, 2000.

[29] A. J. Keane and P. B. Nair. *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. John Wiley & Sons, 2005.

[30] D. Gorissen, K. Crombecq, I. Couckuyt, T. Dhaene, and P. Demeester. *A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design*. Journal of Machine Learning Research, 11:2051 – 2055, July 2010.

[31] H. Rabitz and O. F. Aliş. *General foundations of high-dimensional model representations*. Journal of Mathematical Chemistry, 25(2-3):197–233, 1999. doi:10.1023/A:1019188517934.

[32] G. Li, S. W. Wang, H. Rabitz, S. Wang, and P. Jaffe. *Global uncertainty assessments by high dimensional model representations (HDMR)*. Chemical Engineering Science, 57(21):4445–4460, 2002. Available from: http://dx.doi.org/10.1016/S0009-2509(02)00417-7, doi:10.1016/S0009-2509(02)00417-7.

[33] G. Li, H. Rabitz, J. Hu, Z. Chen, and Y. Ju. *Regularized random-sampling high dimensional model representation (RS-HDMR)*. Journal of Mathematical Chemistry, 43(3):1207–1232, 2008. Available from: http://dx.doi.org/10.1007/s10910-007-9250-x, doi:10.1007/s10910-007-9250-x.

[34] M. Krzywinski and N. Altman. *Points of Significance: Visualizing samples with box plots*. Nature Methods, 11:119 – 120, January 2014. doi:doi:10.1038/nmeth.2813.

[35] Minitab Express Support. *Interpret the key results for boxplot*. Webpage, Oct 2015. Available from: http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/basic-statistics/graphs/boxplot/interpret-the-results/key-results/.

5

Adaptive Initial Step Size Selection for Simultaneous Perturbation Stochastic Approximation

⋆⋆⋆

K. Ito, T. Dhaene.

“ Adaptive Initial Step Size Selection for Simultaneous Perturbation Stochastic Approximation ”.

Published in SpringerPlus, DOI 10.1186/s40064-016-1823-3, vol. 5, no. 1, pp. 1 - 18, 2016.

⋆⋆⋆

A difficulty in using Simultaneous Perturbation Stochastic Approximation (SPSA) is its performance sensitivity to the step sizes chosen at the initial stage of the iteration. If the step size is too large, the solution estimate may fail to converge. The proposed adaptive stepping method automatically reduces the initial step size of the SPSA so that reduction of the objective function value occurs more reliably. Ten mathematical functions each with three different noise levels were used to empirically show the effectiveness of the proposed idea. A parameter estimation example of a nonlinear dynamical system is also included.

5.1 Introduction

Simultaneous Perturbation Stochastic Approximation (SPSA) [1] is an optimization algorithm that uses only objective function measurements in the search of solutions. Applications of SPSA include model-free predictive control [2–4], signal timing for vehicle timing control [5], air traffic network [6], and marine vessel traffic management [7]. More applications are mentioned in the introductory article by Spall [8]. SPSA has been used successfully in many optimization problems that have high-dimensional input parameter space and the objective value is not deterministic [9].

In this optimization method, the initial design parameter vector θ of D-dimensions is perturbed simultaneously in every dimension, i.e. by adding and subtracting a perturbation vector $\mathbf{\Delta}$ of D-dimensions, thus obtaining an estimate of the gradient vector g. Unlike the traditional finite differencing approach, it only takes two function evaluations to obtain the estimate of the gradient. Yet, the number of iteration needed for convergence to the optimum is said to be more or less the same with Finite-Difference Stochastic Approximation (FDSA) [10], which in essence is an approximate steepest-descent method that uses finite-differencing to approximate the partial derivatives along each of the D parameters. Thus, the number of function evaluations of SPSA is D-fold smaller compared to FDSA [8]. An extension to this method exists to include second-order (Hessian) effects to accelerate convergence [11–13]. However, we will not treat this enhancement here.

The problem solved by SPSA in this work can be formulated as following.

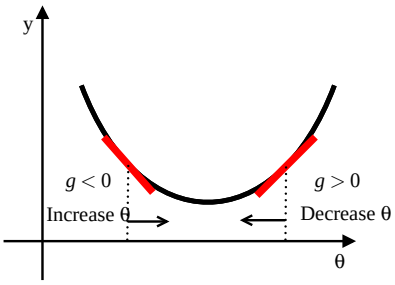
$$\min_{\theta \in \Theta} f(\theta), \tag{5.1}$$

where $f(\theta)$ is the objective function and θ is a D-dimensional vector of parameters. We assume that each element in the vector θ is a real number and has upper and lower bounds that defines the Cartesian product domain Θ. The SPSA and FDSA procedures are in the general recursive form:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \hat{g}_k(\hat{\theta}_k), \tag{5.2}$$

where $\hat{g}_k(\hat{\theta}_k)$ is the estimate of the gradient vector $g(\hat{\theta})$ at iteration k based on the measurements of the objective function. The a_k is the step size at iteration k. Equation (5.2) is analogous to the gradient descent algorithm in nonlinear programming, in which g_k is the gradient of the objective function $\nabla f(\hat{\theta}_k)$. The difference is that in equation (5.2), $\hat{g}_k$ represent gradients stochastically and the effect of the noise or deviation from the true gradient is expected to cancel out as the iteration count k increases. The step sizes a_k are normally prescribed in SPSA and FDSA as a function of k just like the Simulated-Annealing's [14] cooling schedule. This is because these methods do not assume deterministic responses in the measurements of the objective function values. Thus, unlike the nonlinear

Figure 5.1 Objective value minimization using gradient descent (one variable): if gradient g is positive at θ_k then move to $\theta_{k+1} < \theta_k$, if gradient g is negative then move to $\theta_{k+1} > \theta_k$



programming counterparts, adaptation of step sizes based on gradients and amount of descent achieved (such as in the line search) is usually not done in the stochastic approximation optimization methods. The rationale of the equation (5.2) is intuitively depicted in Figure 5.1 for one variable case.

Under appropriate conditions, the iteration in equation (5.2) will converge to the optimum θ^* in some stochastic sense [15, 16, p. 183]. The hat symbol indicates an "estimate". Thus, $\hat{\theta}_k$ denotes the estimate of the optimum θ^* at iteration k. Let $y(\cdot)$ denote a measurement of the objective function $f(\cdot)$ at parameter value denoted by " $\cdot$ " and c_k be some small positive number. The measurements are assumed to contain some noise, i.e. $y(\cdot) = f(\cdot) + \text{noise}$. In SPSA, the ith component $\hat{g}_{ki}(\hat{\theta}_k)$ of the gradient vector $\hat{g}_k(\hat{\theta}_k)$ is formed from a ratio involving the individual components in the perturbation vector and the difference in the two corresponding measurements. For two-sided simultaneous perturbations, we have

$$\hat{g}_{ki}(\hat{\theta}_k) = \frac{y(\hat{\theta}_k + c_k\mathbf{\Delta}_k) - y(\hat{\theta}_k - c_k\mathbf{\Delta}_k)}{2c_k\Delta_{ki}}, \tag{5.3}$$

where the D-dimensional random perturbation vector

$$\mathbf{\Delta}_k = (\Delta_{k0}, \Delta_{k1}, \ldots, \Delta_{k(D-1)})^T, \tag{5.4}$$

follows a specific statistical distribution criterion. Here, i is the parameter index. A simple choice for each component of $\mathbf{\Delta}_k$ is to use Bernoulli ± 1 distribution, which is essentially a random switching between +1 and -1. The Bernoulli distribution is proven to be an optimal distribution for the simultaneous perturbation [17]. Note also that in the equation (5.3), we do not evaluate $y(\hat{\theta}_k)$. The recursive equation (5.2) proceeds with only the responses from the two perturbed inputs $y(\hat{\theta}_k + c_k\mathbf{\Delta}_k)$ and $y(\hat{\theta}_k - c_k\mathbf{\Delta}_k)$.

The choice of a_k and c_k is critical to the performance of SPSA and suggested

values can be found in [18]. At given iteration k:

$$a_k = \frac{a}{(A+k+1)^\alpha}, \tag{5.5}$$

$$c_k = \frac{c}{(k+1)^\gamma}, \tag{5.6}$$

where

$$\begin{aligned}
\alpha &= 0.602 \\
\gamma &= 0.101 \\
c &\simeq \text{standard deviation of measurement noise} \\
A &\leq 10\% \text{ of maximum number of iterations} \\
a &= \delta\hat{\theta}_{0_{\min}} \frac{(A+1)^\alpha}{|\hat{g}_{0i}(\hat{\theta}_0)|} \\
k &= \text{iteration index starting with 0} \\
\delta\hat{\theta}_{0_{\min}} &= \text{smallest initial change desired in a parameter}
\end{aligned}$$

The setting for α and γ above are not optimal in the asymptotic sense, but are adapted to finite iteration settings satisfying convergence conditions [16, p. 162-164]. In practice, one of the drawbacks of SPSA is that one has to find good values for a and c, as both affect the performance of the algorithm [16, pp. 165-166] [19–23]. However, for c, we have a tangible measure, which is the output measurement error [18], to select a proper value up front. If the function response is noiseless, c is usually not a critical parameter. On the other hand, a is more problematic, because no clear measure exists. It is possible to work with $\delta\hat{\theta}_{0_{\min}}$ instead of a, but a priori assignment of its value is still non-trivial if little is known about the function that we are trying to optimize.

A larger value of a generally produces better results compared to a smaller value of a. This is because in finite-sample setting, larger a allows the algorithm to move in bigger steps towards the solution. However, this also increases the chance that the optimization diverges to a worse solution than the starting point. Very often, the user of SPSA has to find as big a as possible that would not cause divergence.

To avoid divergence, an adaptation called "blocking" exists [18, 22] in which the objective values at $\hat{\theta}_k$ is evaluated in addition to the two perturbations. If the new objective function value is "significantly worse" than the current objective function value, the updating of $\hat{\theta}_k$ does not happen. The extra function evaluation at each iteration increases the cost of iteration by 33 %. In addition, a problem dependent threshold parameter to block the $\hat{\theta}_k$ update needs to be set up by the user.

Another way to mitigate divergence is to modify the gradient approximation $\hat{g}_k$ by "scaling" and "averaging" [24, 25]. However, the methods proposed in the

literature require set up of additional threshold parameters critical to their performance. Furthermore, their methods require additional gradient estimations per iteration.

Stochastic Gradient Descent (SGD) methods use noisy information of the gradient of the objective functions. On the other hand, Stochastic Approximation methods such as FDSA and SPSA only uses measurement of noisy objective values. Therefore, adaptive determination of step sizes based on (approximate) gradients and inverse Hessians in SGD literature (such as in [26, 27]) may not be directly applicable to or feasible in SPSA. Convergence conditions also differ between the two. Although this does not exclude the possibility of successful import of ideas from SGD literature, in this work, we will not delve into this direction.

This work provides a solution to determine the appropriate values of a by introducing an adaptive scheme as discussed in section 5.2. It does not require any additional objective function evaluations per iteration nor extra problem dependent parameters to set up.

5.2 Adaptive Initial Step Sizes

To remedy the sensitivity to a, we propose an adaptive stepping algorithm. At the end of each iteration k, we perform the adjustment described in Algorithm 7.

Algorithm 7 Adaptive Initial Step

1: **if** $\min\{y(\hat{\theta}_k + c_k\mathbf{\Delta}_k), y(\hat{\theta}_k - c_k\mathbf{\Delta}_k)\} - y(\hat{\theta}_0) \geq 0$ **then**
2: $\quad \hat{\theta}_{k+1} = \hat{\theta}_b$, where $\hat{\theta}_b$ gives the best y so far
3: $\quad a \leftarrow 0.5a$
4: **end if**

The condition requires that at least one of the two parameter perturbations produce a better (smaller) measurement of the objective function than that of initial guess of parameters $\hat{\theta}_0$ to proceed without modifying a. Therefore, at each iteration k, the smaller of the two measurements of the objective function values of perturbed parameters is compared to that of the initial value at iteration $k = 0$. If the measurements of the objective values of the perturbed parameters are larger, $\hat{\theta}_k$ is reset to θ_b, which is the point that gave the minimum in the history of iteration and a is reduced to half of its previous value. A pseudocode of the proposed SPSA with the adaptive initial step is shown in Algorithm 8. The difference between the standard SPSA and our SPSA is in line 10.

Algorithm 8 Pseudocode of the Proposed Algorithm

1: Initialize a and c (or set $\delta\hat{\theta}_{0_{\min}} \simeq \min(\text{upper bound} - \text{lower bound})$, and $c \simeq$ std of response noise). Set maximum number of iterations maxiter.
2: Obtain initial measurement $y(\hat{\theta}_0)$, and let $\theta_b = \hat{\theta}_0$.
3: **for** k = 0 to maxiter **do**
4: Compute $\mathbf{\Delta}_k$ and c_k.
5: Evaluate $y(\hat{\theta}_k + c_k\mathbf{\Delta}_k)$ and $y(\hat{\theta}_k - c_k\mathbf{\Delta}_k)$.
6: Record the input parameter vector as $\hat{\theta}_b$ if better minimum in y is obtained.
7: Compute $\hat{g}_{ki}(\hat{\theta}_k)$.
8: Compute a_k.
9: $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k\hat{g}_k(\hat{\theta}_k)$.
10: Perform Algorithm 7.
11: **end for**

5.3 Comments on Convergence

Currently available theories of stochastic algorithms are almost all based on asymptotic properties with $k \to \infty$, and SPSA is no exception. For given conditions [16, p. 183], SPSA is proven to converge to a local optima almost surely. However, under limited function evaluation budget, we frequently encounter situations in which SPSA returns worse solution than the initial i.e. divergence. The method we propose is a practical remedy conceived in a finite k setting. We will show, in the next section, its effectiveness empirically via numerical experiments with k in the order of 10^3.

For $\hat{\theta}_k$ to converge to the optimal solution θ^* in *infinite* steps, the following conditions are required for a_k and c_k [1]: $a_k, c_k > 0$ for all k; $a_k, c_k \to 0$ as $k \to \infty$; $\sum_{k=0}^{\infty} a_k = \infty$, and $\sum_{k=0}^{\infty} \left(\frac{a_k}{c_k}\right)^2 < \infty$. With Algorithm 7, $\sum_{k=0}^{\infty} a_k = \infty$ is not guaranteed. For example, if the reduction of a happens in every iteration k, the sum is convergent. In practice, the numbers of function evaluations are finite, and reductions of a are expected to happen only a limited number of times. Therefore, this violation is expected to pose little problem.

The intention of the proposed method is not to modify the asymptotic convergence rate of the original SPSA algorithm [16, p.p. 186 - 188]. The adaptive step takes place only if it is suspected that the objective value has become larger than at the starting point $\hat{\theta}_0$. The probability of Algorithm 7 taking place is expected to go to zero under reasonable signal-to-noise ratio as $f(\hat{\theta}_k)$ decreases. The worst situation that can happen is that the every perturbation $c_k\mathbf{\Delta}_k$ produces worsening moves and no improvement is obtained compared to the starting point θ_0. In section 5.4, we will confirm empirically what we have described about the conver-

gence in finite k settings ($k \sim 10^3$).

Another reason to take the objective value at the starting point as the threshold value to judge divergence is that if we update this value with $y(\hat{\theta_k})$, where $k > 0$, we may risk picking a point that is too low due to the noise incurred in the measurement y. This in turn inhibits further improvement of $\hat{\theta_k}$ for lower objective values.

In the following section, the smallest output of mathematical functions will be sought using the standard SPSA and our adaptive initial stepping SPSA. This will show the sensitivity of the function value in the final iteration to the initial step size $\delta\hat{\theta}_{0_{\min}}$ and so the sensitivity to a, and how the adaptive initial stepping substantially mitigates the difficulty to find the proper initial perturbation magnitude.

5.4 Computational Results

In this section, we will compare the original SPSA and our modified SPSA as described in Algorithm 8 using 10 analytical test functions and a parameter estimation example of a nonlinear dynamic system.

5.4.1 Test Functions

To see the effect of the new adaptive stepping algorithm in SPSA, the minimum points of ten different mathematical test functions were sought. Except for Griewank function, the following conditions were applied. The functions' responses were minimized from arbitrary starting points $\hat{\theta}_0 \in [-2, 2]^D$ (D-dimensional product space with lower bound -2 and upper bound 2). If $\hat{\theta}_k = [\hat{\theta}_{k0}, \hat{\theta}_{k1}, \cdots, \hat{\theta}_{ki}, \cdots, \hat{\theta}_{k(D-1)}]^T$ exceeded $[-10,\ 10]$ in any of its D dimensions, that parameter was replaced by -10 if it was less than -10 or was replaced by 10 if it was larger than 10. For Griewank function, it was randomly started from $\hat{\theta}_0 \in [-120,\ 120]^D$. If $\hat{\theta}_k$ exceeded $[-600,\ 600]$ in any of its D dimensions, that parameter was replaced by -600 if it was less than -600 or was replaced by 600 if it was larger than 600. For all ten functions, the iteration was stopped when 2000 evaluations of the objective function were reached. For convenience, we will label our proposed algorithm as "A_SPSA" and the standard SPSA as "SPSA".

The optimizations for each of the ten objective functions were started from 20 different starting points. After the 2000 iterations, the distributions of objective values were plotted with respect to $\delta\hat{\theta}_{0_{\min}}$. Eleven different values of $\delta\hat{\theta}_{0_{\min}}$ between 1.0×10^{-4} and 1.0×10^{1} (up to 1.0×10^{2} for Griewank) were used to make the plot. The dimensions of the functions were set to be 20, i.e. $D = 20$.

The definitions of the ten functions are given in the following. The Rosenbrock

function is described as

$$f(\theta) = \sum_{i=0}^{D-2} \big(100(\theta_{i+1} - \theta_i^2)^2 + (\theta_i - 1)^2\big), \tag{5.7}$$
$$i = 0, 1, \ldots, D-1, \quad D > 1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 1.$$

The Sphere function is described as

$$f(\theta) = \sum_{i=0}^{D-1} \theta_i^2, \tag{5.8}$$
$$i = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 0.$$

The Schwefel function is described as

$$f(\theta) = \sum_{j=0}^{D-1} \left(\sum_{i=0}^{j} \theta_i \right)^2, \tag{5.9}$$
$$i = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 0.$$

The Rastrigin function is described as

$$f(\theta) = \sum_{i=0}^{D-1} \big(\theta_i^2 - 10\cos(2\pi\theta_i) + 10\big), \tag{5.10}$$
$$i = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 0.$$

The Skewed Quartic function [16, ex. 6.6] is described as

$$f(\theta) = (\mathbf{B}\theta)^T \mathbf{B}\theta + 0.1 \sum_{i=0}^{D-1} (\mathbf{B}\theta)_i^3 + 0.01 \sum_{i=0}^{D-1} (\mathbf{B}\theta)_i^4, \tag{5.11}$$
$$i = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 0.$$

where the matrix $\mathbf{B}$ in the Skewed Quartic function is a square matrix with upper triangular elements set to 1 and the lower triangular elements set to zero. The Griewank function is described as

$$f(\theta) = 1 + \sum_{i=0}^{D-1} \frac{\theta_i^2}{4000} - \prod_{i=0}^{D-1} \cos\left(\frac{\theta_i}{\sqrt{i}}\right), \tag{5.12}$$
$$i = 0, 1, \ldots, D-1,$$
$$f(\theta^*) = 0, \quad \theta_i^* = 0.$$

The Ackley function is described as

$$\begin{aligned} f(\theta) \quad = \quad & -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=0}^{D-1}\theta_i^2}\right) \\ & -\exp\left(\frac{1}{D}\sum_{i=0}^{D-1}\cos(2\pi\theta_i)\right) \\ & +20-\exp(1), \\ & i=0,1,\ldots,D-1, \\ & f(\theta^*)=0, \quad \theta_i^*=0. \end{aligned} \tag{5.13}$$

The Manevich function is described as

$$\begin{aligned} f(\theta) = \sum_{i=0}^{D-1}\left[(1-\theta_i)^2/2^j\right], \\ i=0,1,\ldots,D-1, \\ f(\theta^*)=0, \quad \theta_i^*=1. \end{aligned} \tag{5.14}$$

The Ellipsoid function is described as

$$\begin{aligned} f(\theta) = \sum_{i=0}^{D-1} i\theta_i^2, \\ i=0,1,\ldots,D-1, \\ f(\theta^*)=0, \quad \theta_i^*=0. \end{aligned} \tag{5.15}$$

The Rotated Ellipsoid function is described as

$$\begin{aligned} f(\theta) = \sum_{i=0}^{D-1}\left(\sum_{j=0}^{i}\theta_j^2\right)^2, \\ i=0,1,\ldots,D-1, \\ f(\theta^*)=0, \quad \theta_i^*=0. \end{aligned} \tag{5.16}$$

Each of Figure 5.A.1 to Figure 5.10 show three different cases of noisy measurements of the outputs. The subfigures (a) have no noise added, subfigures (b) and (c) have Gaussian noise added to the true output with standard deviation σ of 0.1 and 1.0 respectively. In all the three noise levels of the ten functions, $c = 0.2$ was used.

A general trend observed from the figures is that when the initial step size is large, the original SPSA tends to diverge to big objective values. The SPSA with the proposed initial step size reduction, on the other hand, effectively mitigates this

divergence problem producing smaller objective values in general as the (a priori) initial step size is increased. This is because if the two function evaluations in the iteration are not smaller than the starting point value $f(\hat{\theta}_0)$, the algorithm will reduce the step size (by halving a) and restart at $\hat{\theta}_b$, which is the point that gave the smallest output in the history of iterations. However, note that the iteration index k in a_k and c_k is not reinitialized. For the ten functions tested, A_SPSA achieved its best performance when $\delta\hat{\theta}_{0_{\min}}$ was close to 10 or 100 for Griewank function. This indicates that one can simply set the minimum perturbation $\delta\hat{\theta}_{0_{\min}}$ close to the magnitude of the difference between upper and lower bound of the parameter in consideration. This may not be a guarantee for the best results but doing so does not cause the optimization to diverge to large responses and the results achieved are not substantially worse than the cases with best settings for a.

As mentioned earlier, the value for c is important when the measurements of y contain noise. Figure 5.11 shows how the choice of c affects the outcome of optimizations. The figure shows the case of the 20 dimensional Sphere Function with Gaussian noise having standard deviation $\sigma = 0.1$. Among the three values of c, namely 0.01, 0.1 and 1.0, $c = \sigma = 0.1$ gave the best results for A_SPSA. At $c = 1.0$, however, A_SPSA showed little improvement in the objective value regardless of $\delta\hat{\theta}_{0_{\min}}$ magnitude. This is caused by a becoming prematurely too small in the divergent early iterations. On the other hand, the standard SPSA showed a good reduction at $\log_{10}(\delta\hat{\theta}_{0_{\min}}) = -2.0$, and -1.5. at both $c = 0.1$ and 1.0. This implies that for A_SPSA, a range of values of good c can be narrower than that of the standard SPSA. On the other hand, the choice of $\delta\hat{\theta}_{0_{\min}}$ (and therefore a) is much easier for A_SPSA. We can, for example, let $\delta\hat{\theta}_{0_{\min}} \simeq \min(U - L)$, where $\min(U - L)$ is the minimum difference between upper and lower bounds of the domain of parameter vector θ. In practice, it is better to scale all the input dimensions to fall in similar or equal intervals.

Figure 5.12 shows the results of optimizing the Rosenbrock and Rastrigin functions using three different values of multiplication factor of a: 0.1, 0.5, and 0.9. The difference in multiplication factor does not change the general trend that larger $\delta\hat{\theta}_{0_{\min}}$ produces better results and that divergence does not occur. One could tune the value of the multiplication factor, but the default value of 0.5 that we showed in the Algorithm 7 generally produces satisfactory results compared to other values of multiplication factors between 0 and 1. The Figure 5.12 (b) also shows that $\delta\hat{\theta}_{0_{\min}} \simeq \min(U - L)$ may not be an optimal setting since smaller value $\delta\hat{\theta}_{0_{\min}} \simeq 10^{-1.5}$ is shown to produce better optimization results when the reduction rate is slow at 0.9. This implies that in a bumpy (highly multimodal) function like Rastrigin, the slow decrease in a can adversely affect the minimization of the objective value by a large number of resets to θ_b. The opposite is true with Rosenbrock function in (a), in which the slow reduction factor 0.9 gave the best result at $\delta\hat{\theta}_{0_{\min}} \simeq 10^{1}$.

For all the mathematical functions tested in this work, optimization using SPSA diverges almost surely if the $\delta\hat{\theta}_{0_{\min}}$ is large. However, A_SPSA and SPSA give closely matching results when the initial step sizes are relatively small (i.e., the left hand side of the plots in Figure 5.A.1 to Figure 5.10). This is because, in cases that divergence does not happen, the adaptation of a does not take place in A_SPSA and therefore SPSA and A_SPSA have identical behavior. This is a confirmation that Algorithm 7 does not alter, in any significant way, the finite sample convergence characteristics of the original SPSA when the divergence does not manifest.

5.4.2 Nonlinear Dynamics Example

We consider a parameter estimation problem with Lorenz attractor. Its nonlinear dynamics is described as

$$\frac{dx_1}{dt} = s(x_2 - x_1), \tag{5.17}$$

$$\frac{dx_2}{dt} = x_1(r - x_3) - x_2, \tag{5.18}$$

$$\frac{dx_3}{dt} = x_1 x_2 - b x_1. \tag{5.19}$$

We seek to identify the system parameters $\theta = [s, r, b]$ by minimizing the one-time-step-ahead prediction error L_k of the state $\mathbf{x}_{k+1}$ given the current state $\mathbf{x}_k = [x_{k1}, x_{k2}, x_{k3}]^T$. We use fourth-order Runge-Kutta method to obtain $\mathbf{x}_{k+1}$.

Let us denote $\hat{\mathbf{x}}_{k+1}$ as one-time-step-ahead prediction given by the estimated system with parameters $\hat{\theta}_k$ but based on x_k which was obtained with the true system parameters. Then, we can define the prediction error as

$$L_k(\mathbf{x}_k, \hat{\theta}_k) = [\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1}]^T \cdot [\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1}]. \tag{5.20}$$

Thus, the optimization to be solved is

$$\min_{\theta \in \Theta} L_k(\mathbf{x}_k, \theta). \tag{5.21}$$

The index k above is the same as the index k in the SPSA algorithms. So the SPSA iteration proceeds along with the time steps of the dynamic system to compute L_k.

We set the true parameters to be $\theta = [10, 28, 8/3]$ and pretend to not to know them. We set the time increment to be $\Delta t = 0.005$ and simulate from $t = 0$ to 20, obtaining target state $\mathbf{x}_k$ with $k = 0, 1, 2, \ldots, 4000$. We let $\delta\hat{\theta}_{0_{\min}} \in \{0.001, 0.01, 1, 10, 100, 1000\}$ and at each value of $\delta\hat{\theta}_{0_{\min}}$ we run both A_SPSA and SPSA 20 times.

For this problem, we set the parameter space as three-dimensional product space $\mathbf{\Theta} = [0, 500]^3$. The initial state is $\mathbf{x}_0 = [2, 3, 4]^T$. The initial guess (starting point) of the parameter set $\hat{\theta}_0$ is a random pick from $\mathbf{\Theta}$.

Figure 5.13 show the box plots of final L_k when started from different values of $\delta\hat{\theta}_{0_{\min}}$. The smallest median of final L_k is obtained at $\delta\hat{\theta}_{0_{\min}} = 10$ for SPSA and $\delta\hat{\theta}_{0_{\min}} = 100$ and 1000 for A_SPSA. The best medians of final L_k obtained for A_SPSA (5.62×10^{-15}) is smaller compared to that of SPSA (3.10×10^{-13}). However, both SPSA and A_SPSA had some runs that did not converge to the above mentioned near-zero L_k values even at these $\delta\hat{\theta}_{0_{\min}}$.

Again, for A_SPSA, the best setting were obtained when $\delta\hat{\theta}_{0_{\min}}$ was set to large values near the order of magnitude of the distance between upper and lower bound of the domain, while for SPSA, the best $\delta\hat{\theta}_{0_{\min}}$ was at an interior value between 10^{-3} and 10^{3}.

Figure 5.14 shows the trajectory of the reference Lorenz attractor and the simulation of the Lorenz attractor whose system parameters s, r, and b were successfully identified by A_SPSA. The time t is run from 0 to 20 starting from the same initial condition used in the identification. The figure shows excellent match.

Figure 5.15 shows the box plots of parameters estimated by A_SPSA and SPSA starting at their best $\delta\hat{\theta}_{0_{\min}}$ settings. The corresponding statistics are shown in Table 5.1 and Table 5.2. The boxes appear collapsed as single horizontal lines at medians since the spaces between first quartiles and third quartiles are very narrow. Some non-converging cases are visible as dots on the figure. The figure and the tables show that the parameter estimates are more consistent from run to run in A_SPSA than that of SPSA as A_SPSA has narrower first and third quartile differences.

Table 5.1 Statistics of identified Lorenz Attractor parameters by 20 SPSA runs at $\delta\hat{\theta}_{0_{\min}} = 10$

	method	s	r	b	Pred. Err. L_{4000}
1	A_SPSA: 0	Min. : 0.00	Min. : 8.017	Min. :0.000	Min. : 0.0000
2	SPSA :20	1st Qu.: 10.00	1st Qu.: 28.000	1st Qu.:2.642	1st Qu.: 0.0000
3		Median : 10.00	Median : 28.000	Median :2.667	Median : 0.0000
4		Mean : 55.94	Mean : 45.534	Mean :2.311	Mean : 1.3645
5		3rd Qu.: 11.11	3rd Qu.: 36.817	3rd Qu.:2.667	3rd Qu.: 0.1017
6		Max. :477.04	Max. :328.504	Max. :3.261	Max. :19.6773

Table 5.2 Statistics of identified Lorenz Attractor parameters by 20 A_SPSA runs at $\delta\hat{\theta}_{0_{\min}} = 100$

	method	s	r	b	Pred. Err. L_{4000}
1	A_SPSA:20	Min. : 0.000	Min. : 0.000	Min. : 0.0000	Min. : 0.0000
2	SPSA : 0	1st Qu.: 10.000	1st Qu.: 28.000	1st Qu.: 2.6667	1st Qu.: 0.0000
3		Median : 10.000	Median : 28.000	Median : 2.6667	Median : 0.0000
4		Mean : 68.069	Mean : 31.816	Mean : 24.8487	Mean : 1.2328
5		3rd Qu.: 10.000	3rd Qu.: 28.000	3rd Qu.: 2.6667	3rd Qu.: 0.0000
6		Max. :500.000	Max. :156.811	Max. :438.8246	Max. :15.6654

5.5 Conclusion

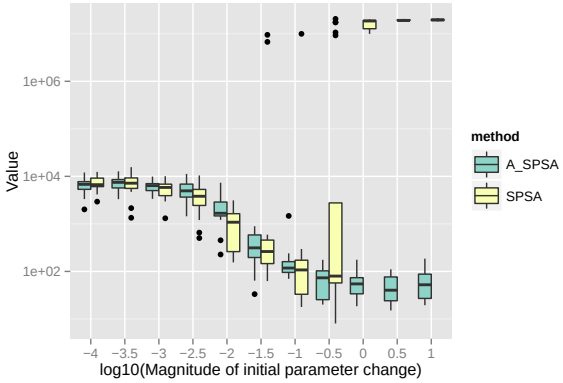
With the adaptive initial step algorithm, one can avoid divergence in SPSA iterations. Moreover, with a large initial step size, the SPSA algorithm with the adaptive initial step algorithm was able to find equal or better solutions compared to the original SPSA for all the ten mathematical function minimization problems that we have tested. In the nonlinear dynamics example, the new algorithm was able to find system parameters more precisely. The proposed method may not eliminate the need of tuning the parameters of SPSA algorithms, but it facilitates the process by eliminating the risk of solution divergence and reducing the trial-and-error effort. Further testing of the algorithm with different test functions, noise distributions, and industrial use-cases would be beneficial. The improvement proposed in this work is expected to be valuable when the objective functions are costly to evaluate or if the algorithm is employed inside another algorithm such as machine learning or target tracking, for manual tuning of the parameters would be cumbersome in such cases. As a future work, it would be beneficial to investigate under what conditions the probability of the proposed adaptation (i.e. going into if-branch in Algorithm 7) happening tends to zero as iteration k tends to infinity.

Acknowledgements

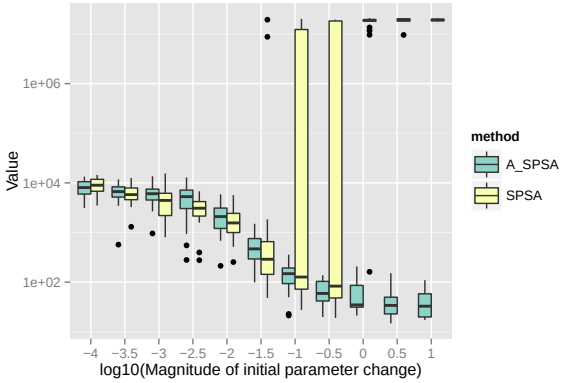
The authors would like to thank James C. Spall for constructive comments on the proposed method. Keiichi Ito has been funded by the Institute for the Promotion of Innovation through Science and Technology (IWT) through the Baekeland Mandate program. This research has also been funded by the Interuniversity Attraction Poles Programme BESTCOM initiated by the Belgian Science Policy Office.

Figure 5.1 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Rosenbrock".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

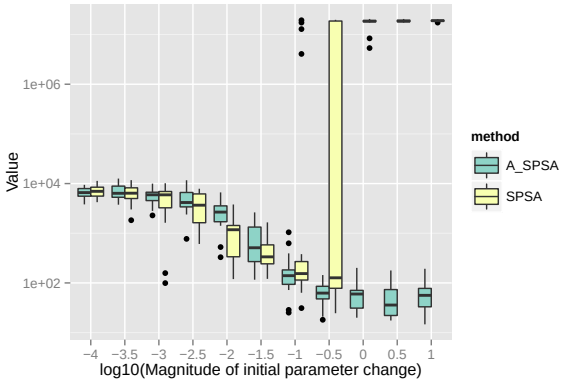
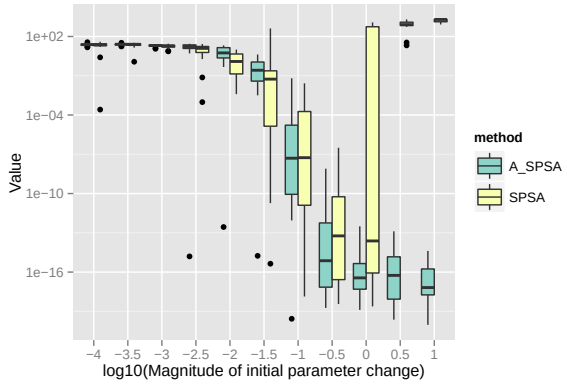
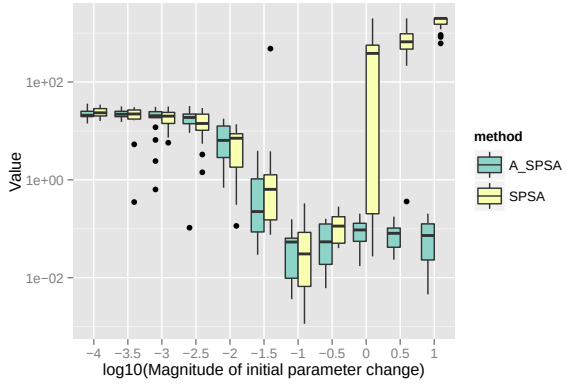


Figure 5.2 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Sphere".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

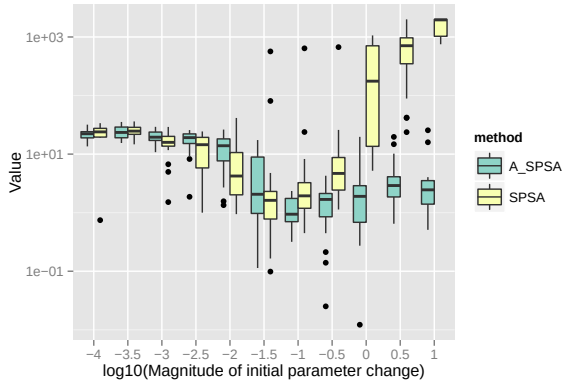
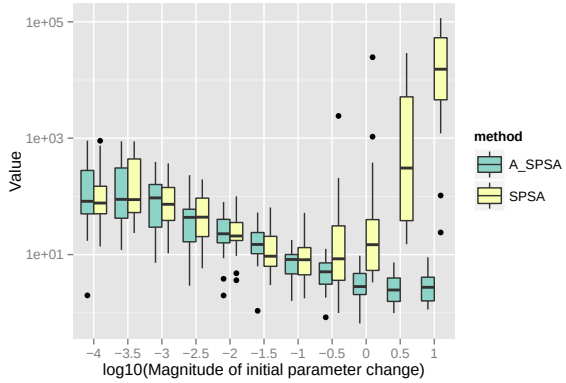
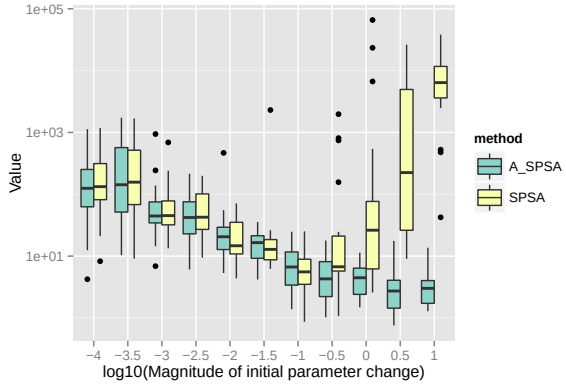


Figure 5.3 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Schwefel".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

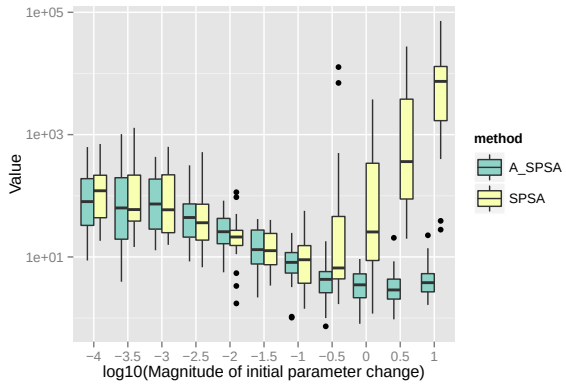
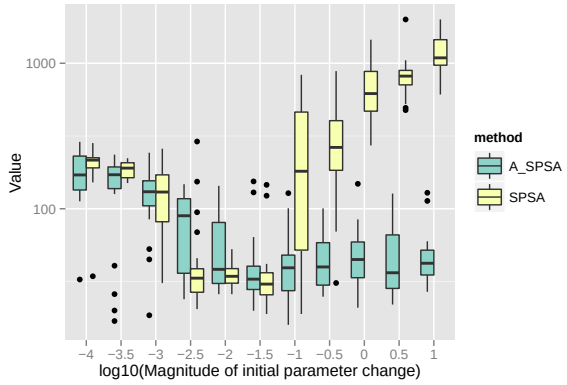
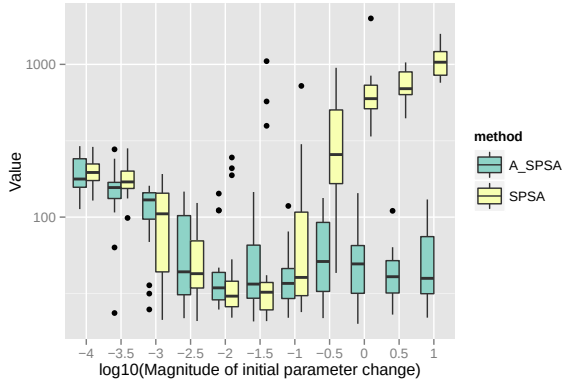


Figure 5.4 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Rastrigin".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

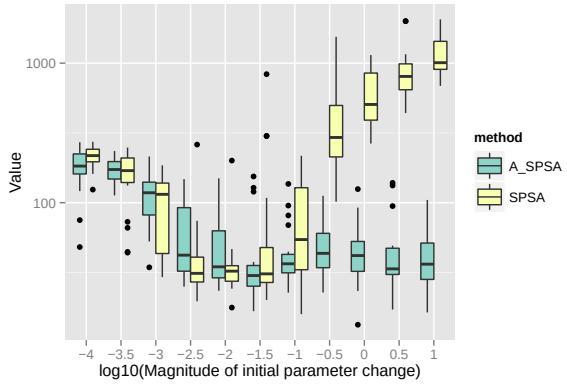
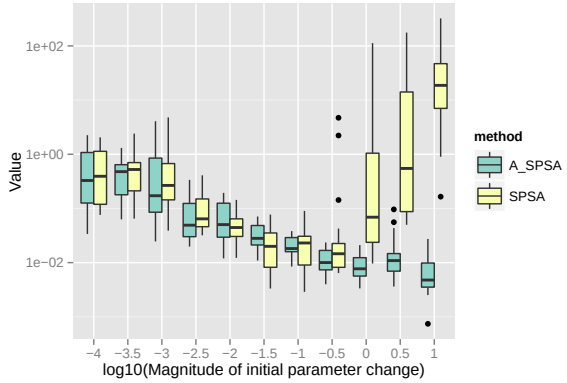
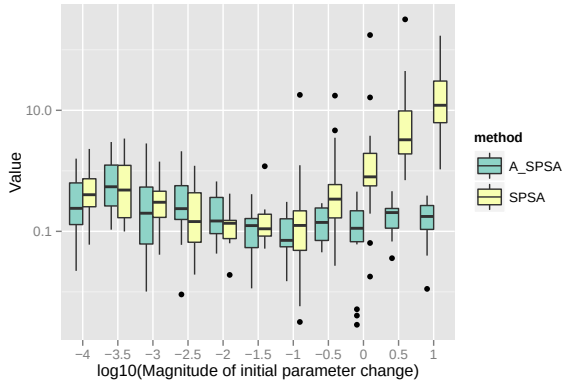


Figure 5.5 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Skewed Quartic".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

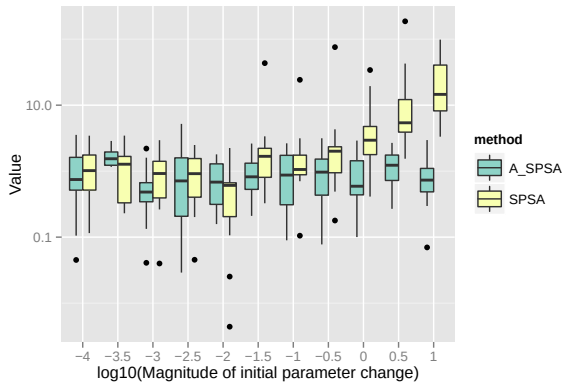
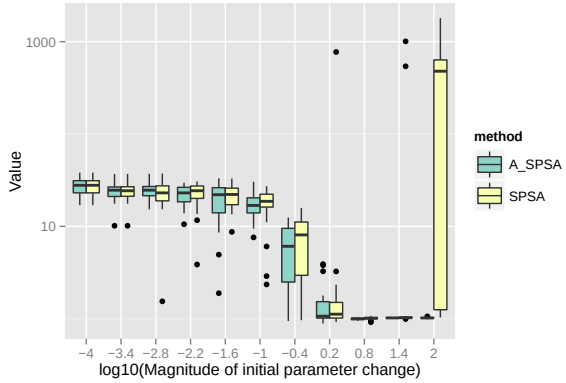
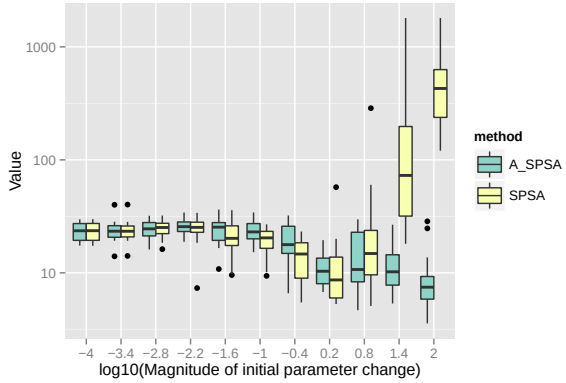


Figure 5.6 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Griewank".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

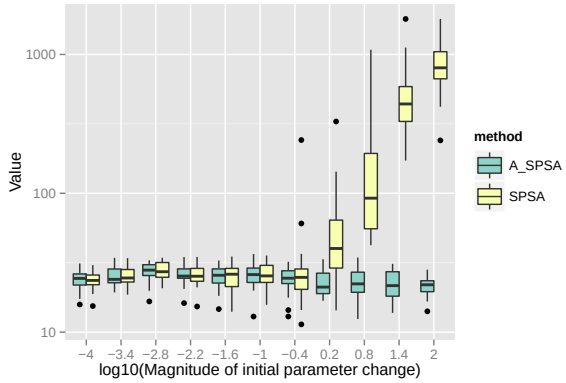
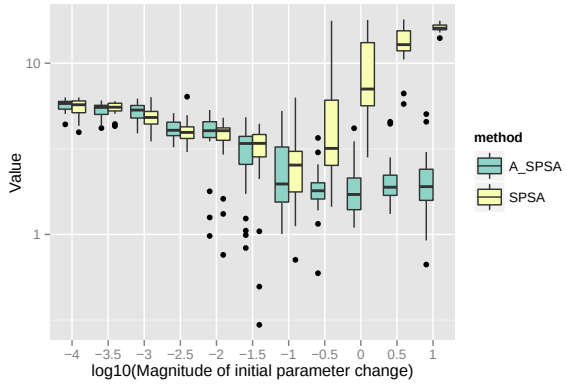
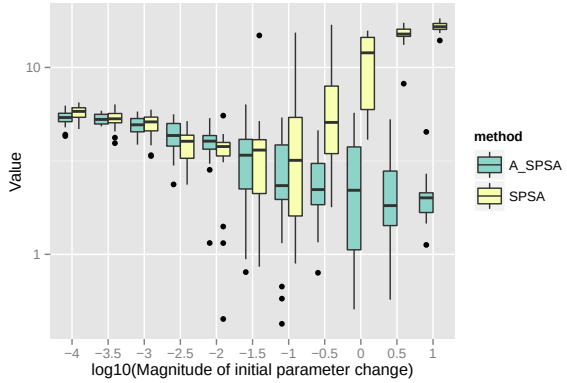


Figure 5.7 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Ackley".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

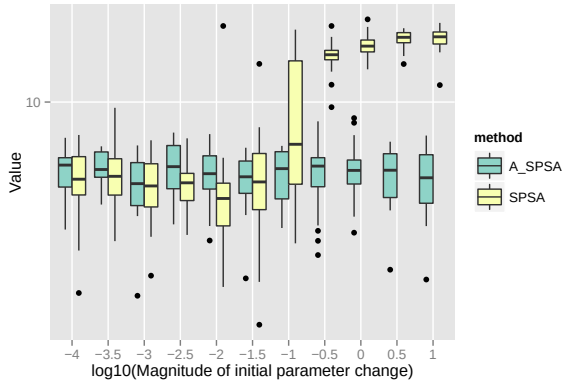
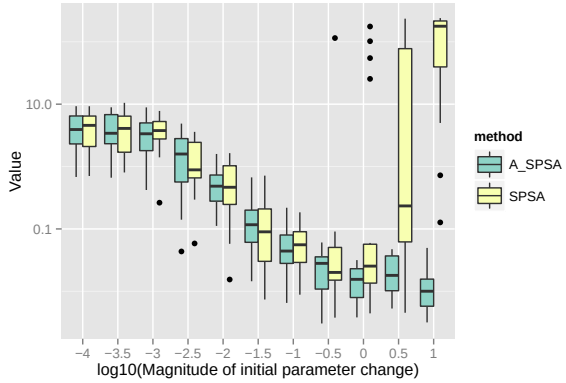
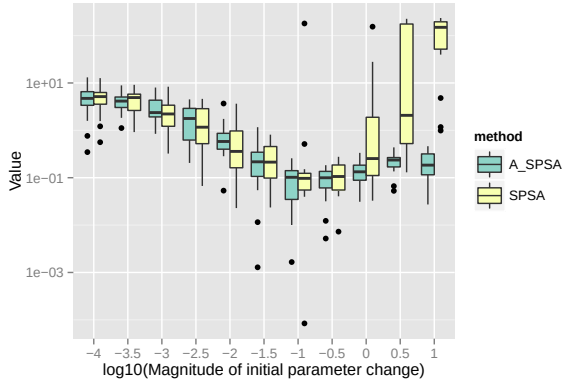


Figure 5.8 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Manevich".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

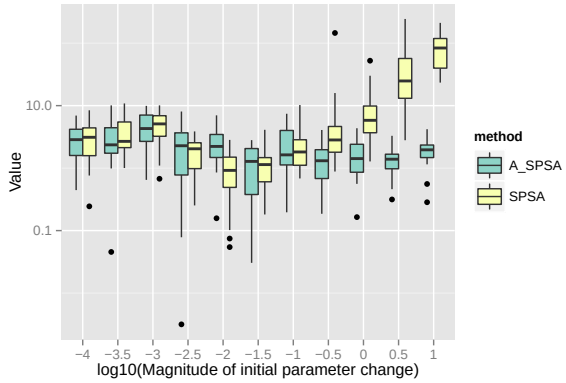
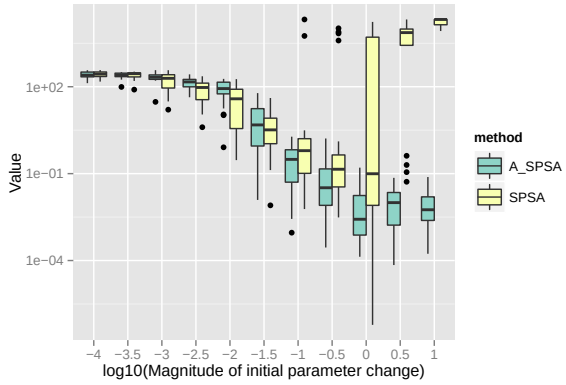
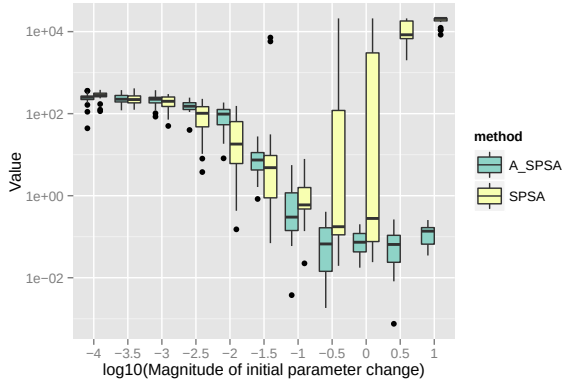


Figure 5.9 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Ellipsoid".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

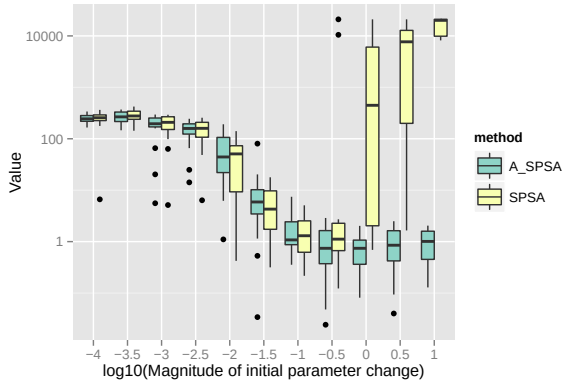
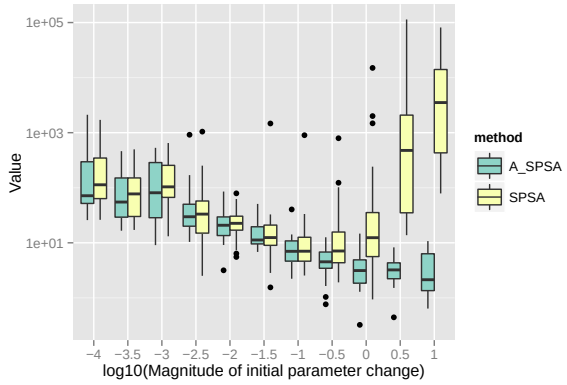
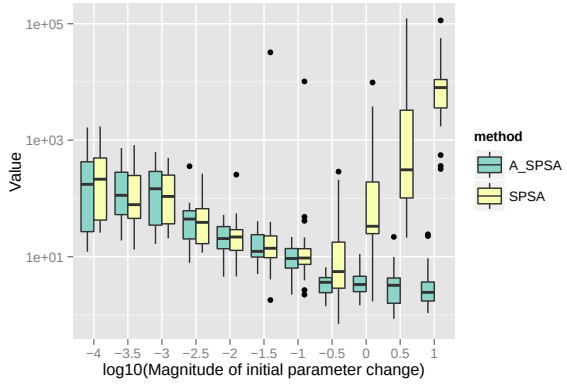


Figure 5.10 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Rotated Ellipsoid".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

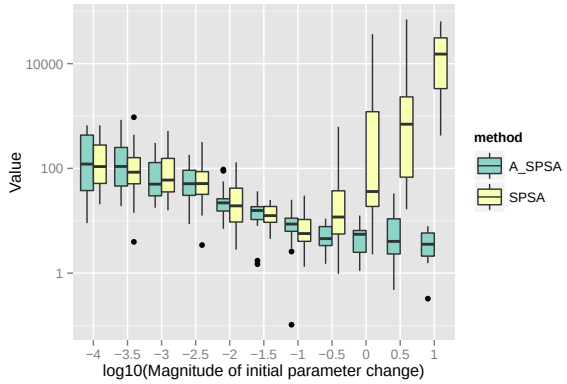
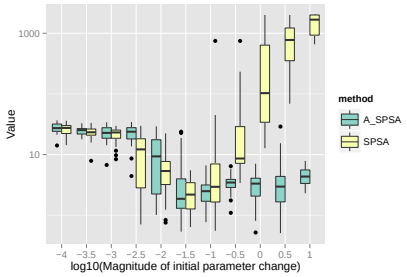
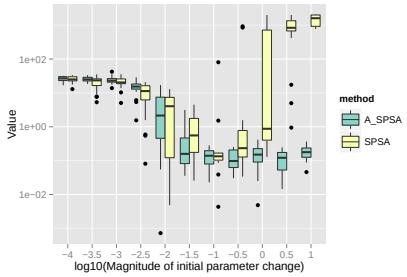


Figure 5.11 Effect of choice of c to the final response of "Sphere" with Gaussian noise of $\sigma = 0.1$ after 2000 function evaluations.

(a) $c = 0.01$



(b) $c = 0.10$



(c) $c = 1.00$

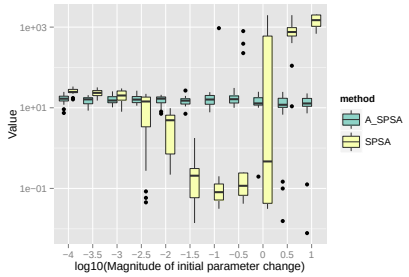
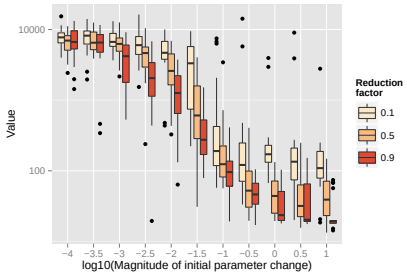


Figure 5.12 Effect of choice of the reduction factor of a to the responses after 2000 function evaluations.

(a) Rosenbrock (no noise)



(b) Rastrigin (no noise)

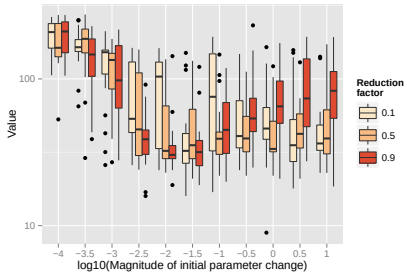


Figure 5.13 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of L_{4000} (after 8000 function evaluations)

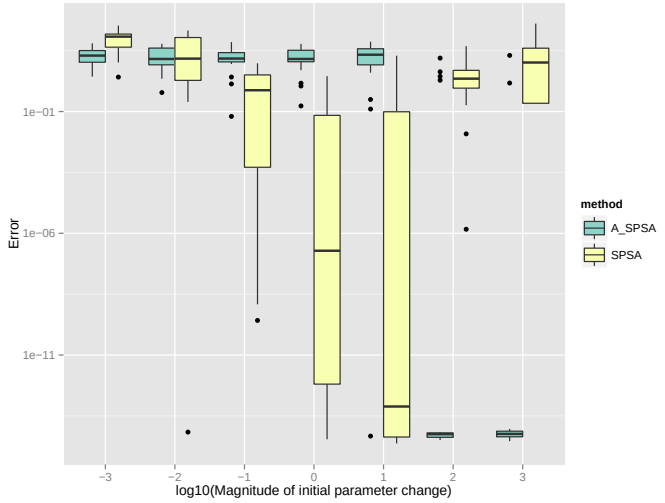


Figure 5.14 State evolution of the target and identified Lorenz attractor, $t = 0$ to 20

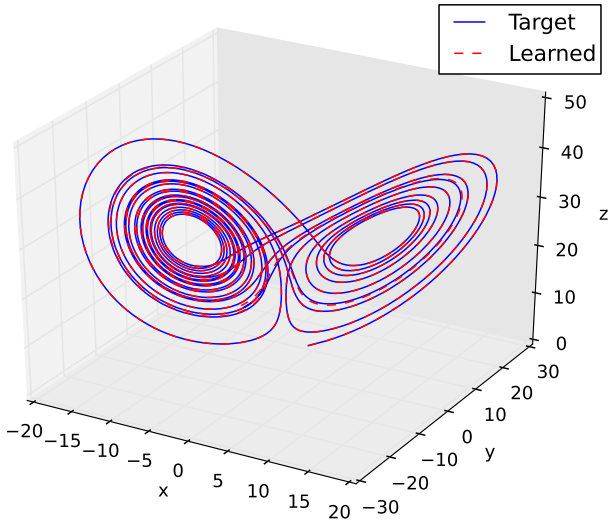
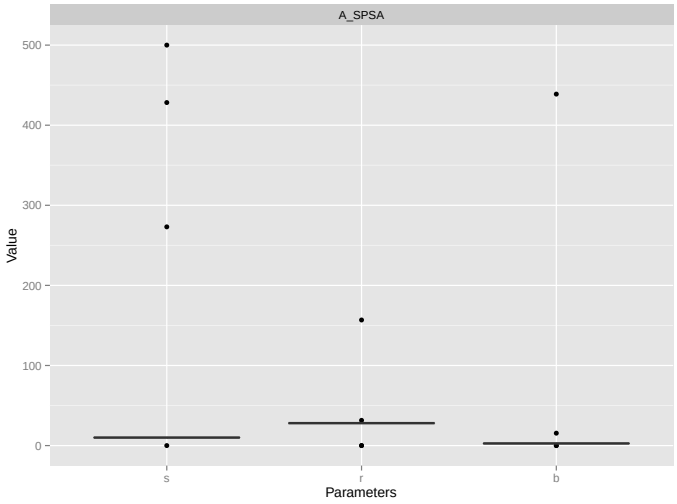
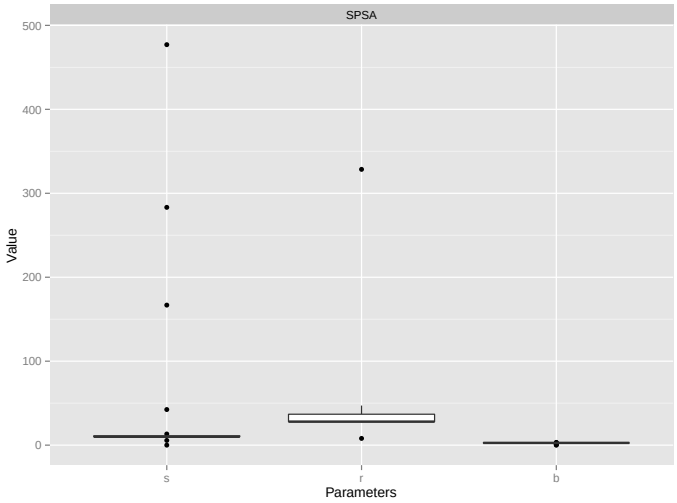


Figure 5.15 Distribution of the parameters identified by A_SPSA and SPSA

(a) A_SPSA with $\delta\hat{\theta}_{0_{\min}} = 100$



(b) SPSA with $\delta\hat{\theta}_{0_{\min}} = 10$



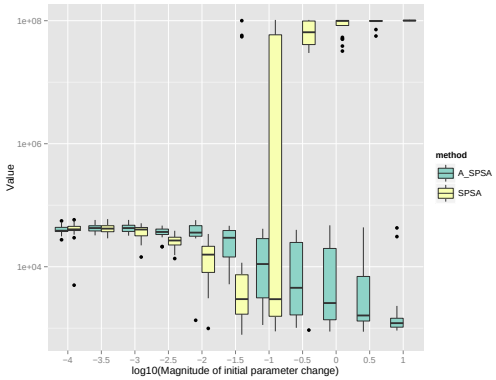
Appendix

5.A Selected Results in 100 Dimensions

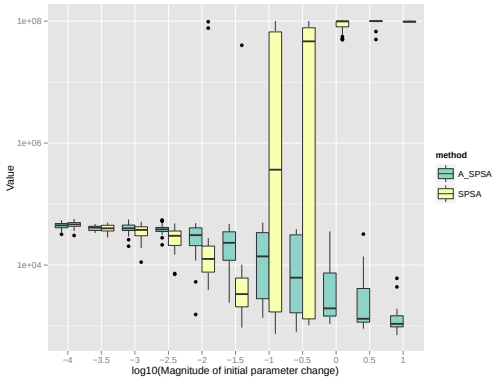
In this section, we show results of optimizing five test functions in 100 dimensions using A_SPSA and SPSA. The general observation remain the same as discussed in section 5.4. A_SPSA effectively eliminates divergence when the initial perturbation is large.

Figure 5.A.1 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Rosenbrock".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

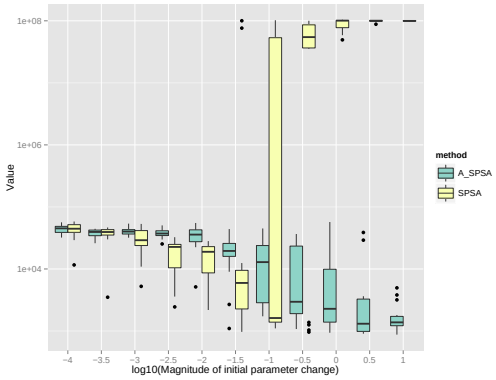
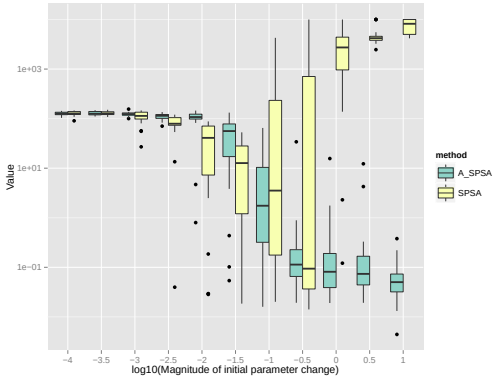
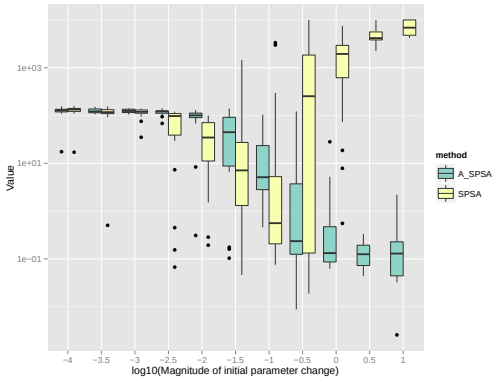


Figure 5.A.2 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Sphere".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

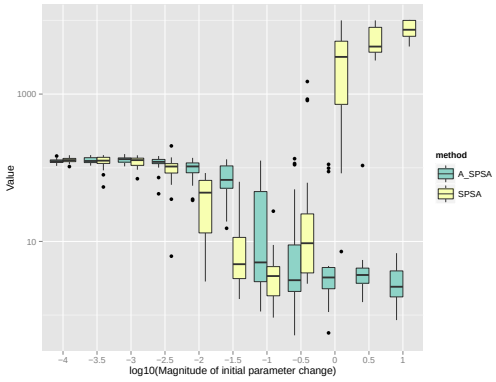
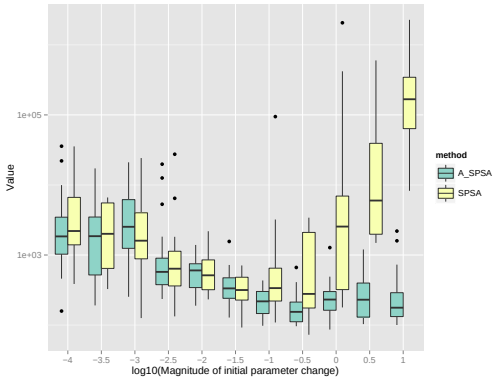
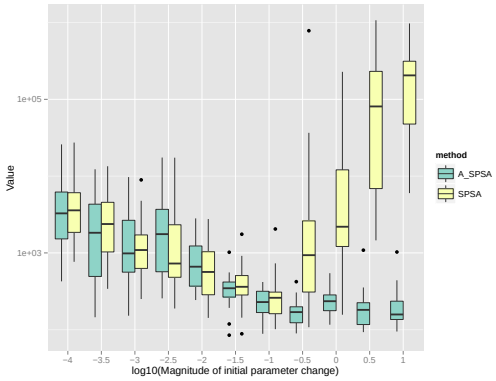


Figure 5.A.3 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Schwefel".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

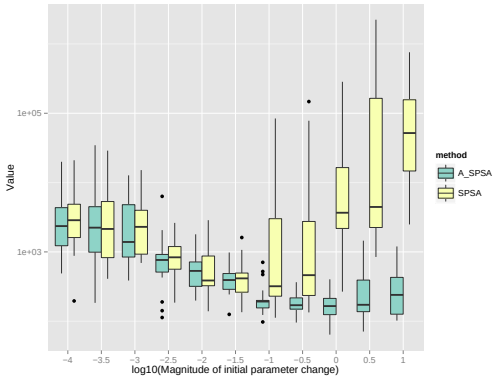
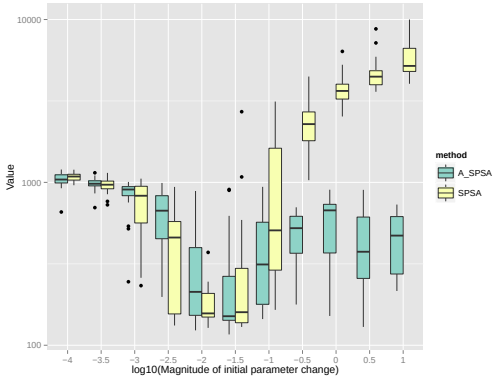
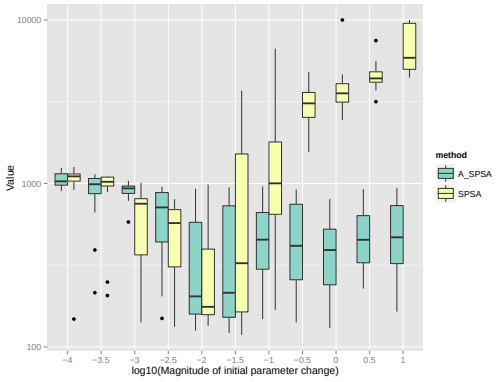


Figure 5.A.4 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Rastrigin".

(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$

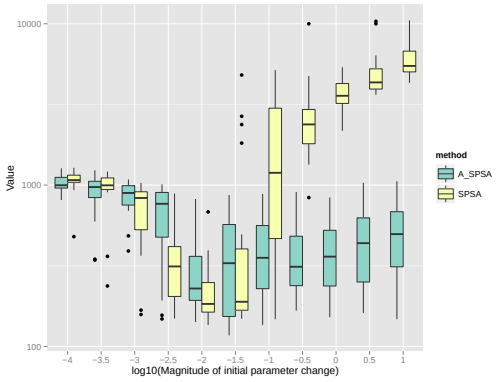
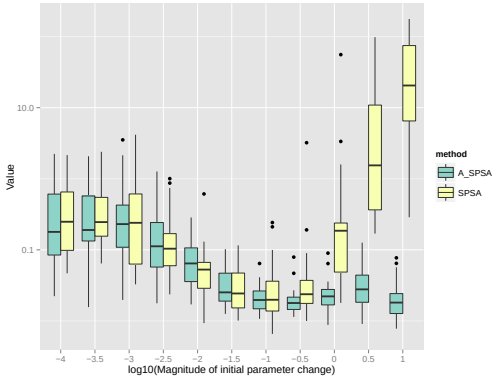
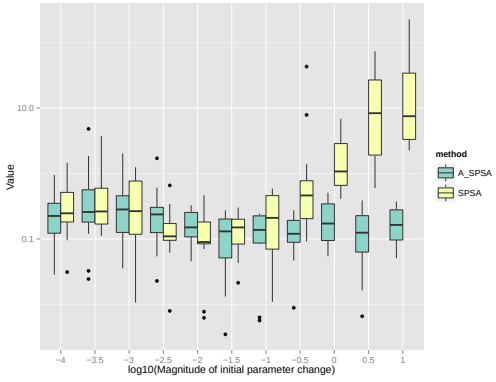


Figure 5.A.5 Initial parameter change $\delta\hat{\theta}_{0_{\min}}$ and distribution of responses after 2000 function evaluations for "Skewed Quartic".

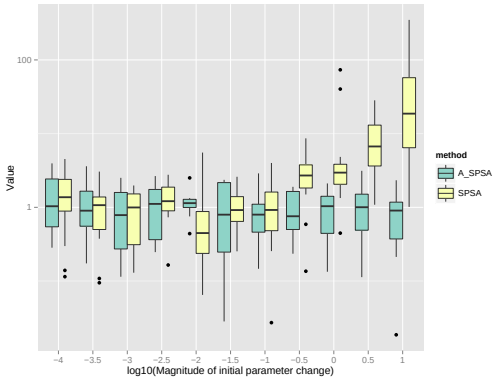
(a) No noise



(b) $\sigma = 0.10$



(c) $\sigma = 1.0$



References

[1] J. C. Spall. *Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation*. IEEE Transactions on Automatic Control, 37(3):332 – 341, March 1992.

[2] N. Dong and Z. Chen. *A novel ADP based model-free predictive control*. Nonlinear Dynamics, 69(1-2):89–97, 2012. Available from: http://dx.doi.org/10.1007/s11071-011-0248-3, doi:10.1007/s11071-011-0248-3.

[3] N. Dong and Z. Chen. *A novel data based control method based upon neural network and simultaneous perturbation stochastic approximation*. Nonlinear Dynamics, 67(2):957–963, 2012. Available from: http://dx.doi.org/10.1007/s11071-011-0039-x, doi:10.1007/s11071-011-0039-x.

[4] H.-S. Ko, K. Y. Lee, and H.-C. Kim. *A Simultaneous Perturbation Stochastic Approximation (SPSA)-Based Model Approximation and its Application for Power System Stabilizers*. International Journal of Control, Automation, and Systems, 6(4):506–514, August 2008.

[5] J. C. Spall and D. C. Chin. *Traffic-Responsive Signal Timing for System-Wide Traffic Control*. Transportation Research, 5(Part C):153–163, 1997.

[6] N. L. Kleinman, S. D. Hill, and V. A. Ilenda. *SPSA/SIMMOD Optimization of Air Traffic Delay Cost*. In Proceedings of the American Control Conference, pages 1121–1125, Albuquerque, New Mexico, USA, 4-6 June 1997.

[7] R. Burnett. *Application of Stochastic Optimization to Collision Avoidance*. In Proceedings of the American Control Conference, pages 2789–2794, Boston, Massachusetts, USA, 29 June-2 July 2004.

[8] J. C. Spall. *An Overview of the Simultaneous Perturbation Method for Efficient Optimization*. Johns Hopkins APL Technical Digest, 19(4):482–492, 1998.

[9] *Simultaneous Perturbation Stochastic Approximation: A method for System Optimization*. http://www.jhuapl.edu/SPSA/index.html.

[10] J. Kiefer and J. Wolfowitz. *Stochastic Estimation of the Maximum of a Regression Function*. Annals of Mathematical Statistics, 23:452–466, September 1952.

[11] J. C. Spall. *Adaptive Stochastic Approximation by the Simultaneous Perturbation Method*. Transactions on Automatic Control, 45(10):1839–1853, October 2000.

[12] J. C. Spall. *Feedback and Weighting Mechanisms for Improving Jacobian Estimates in the Adaptive Simultaneous Perturbation Algorithm*. IEEE Transactions on Automatic Control, 54(6):12161229, 2009.

[13] X. Zhu and J. C. Spall. *A modified second-order SPSA optimization algorithm for finite samples*. International Journal of Adaptive Control and Signal Processing, 16:397–409, 2002.

[14] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. *Optimization by Simulated Annealing*. Science, 220(4598):671 – 680, May 13 1983.

[15] S. Finck and H.-G. Beyer. *Performance analysis of the simultaneous perturbation stochastic approximation algorithm on the noisy sphere model*. Theoretical Computer Science, 419:50 – 72, 2012. Available from: http://www.sciencedirect.com/science/article/pii/S0304397511009340, doi:http://dx.doi.org/10.1016/j.tcs.2011.11.015.

[16] J. C. Spall. *Introduction to Stochastic Search and Optimization, Estimation, Simulation and Control*. Wiley-Interscience, 2003.

[17] P. Sadegh and J. C. Spall. *Optimal Random Perturbations for Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation*. In Proceedings of the American Control Conference, pages 3582–3586, Albuquerque, NM, USA, 4-6 June 1997.

[18] J. C. Spall. *Implementation of the Simultaneous Perturbation Algorithm for Stochastic Optimization*. IEEE Transactions on Aerospace and Electronic Systems, 34(3):817–823, July 1998.

[19] M. U. Altaf, A. W. Heemink, M. Verlaan, and I. Hoteit. *Simultaneous Perturbation Stochastic Approximation for Tidal Models*. Ocean Dynamics, 61:1093 – 1105, 2011.

[20] X. Shen, M. Yao, W. Jia, and D. Yuan. *Adaptive complementary filter using fuzzy logic and simultaneous perturbation stochastic approximation algorithm*. Measurement, 45(5):1257 – 1265, 2012. Available from: http://www.sciencedirect.com/science/article/pii/S0263224112000267, doi:http://dx.doi.org/10.1016/j.measurement.2012.01.011.

[21] M. Radac, R. Precup, E. Petriu, and S. Preitl. *Application of IFT and SPSA to Servo System Control*. IEEE Transactions on Neural Networks, 22(12):2363–2375, Dec 2011. doi:10.1109/TNN.2011.2173804.

[22] D. Easterling, L. Watson, M. Madigan, B. Castle, and M. Trosset. *Parallel deterministic and stochastic global minimization of functions with very*

many minima. Computational Optimization and Applications, 57(2):469 – 492, 2014. Available from: http://dx.doi.org/10.1007/s10589-013-9592-1, doi:10.1007/s10589-013-9592-1.

[23] A. Taflanidis and J. Beck. *Stochastic Subset Optimization for optimal reliability problems*. Probabilistic Engineering Mechanics, 23(2 - 3):324 – 338, 2008. 5th International Conference on Computational Stochastic Mechanics. Available from: http://www.sciencedirect.com/science/article/pii/S0266892007000501, doi:http://dx.doi.org/10.1016/j.probengmech.2007.12.011.

[24] S. Andradóttir. *A Scaled Stochastic Approximation Algorithm*. Management Science, 42(4):475–498, 1996. Available from: http://pubsonline.informs.org/doi/abs/10.1287/mnsc.42.4.475, arXiv:http://pubsonline.informs.org/doi/pdf/10.1287/mnsc.42.4.475, doi:10.1287/mnsc.42.4.475.

[25] Z. Xu and X. Wu. *A new hybrid stochastic approximation algorithm*. Optimization Letters, 7(3):593–606, 2013. Available from: http://dx.doi.org/10.1007/s11590-012-0443-2, doi:10.1007/s11590-012-0443-2.

[26] M. D. Zeiler. *ADADELTA: An Adaptive Learning Rate Method*. arXiv:1212.5701v1, [cs.LG], 2012. Available from: http://arxiv.org/abs/1212.5701.

[27] L. Bottou. *Large-scale machine learning with stochastic gradient descent*. In Proceedings of COMPSTAT'2010, pages 177–186. Springer, 2010.

6 Conclusion

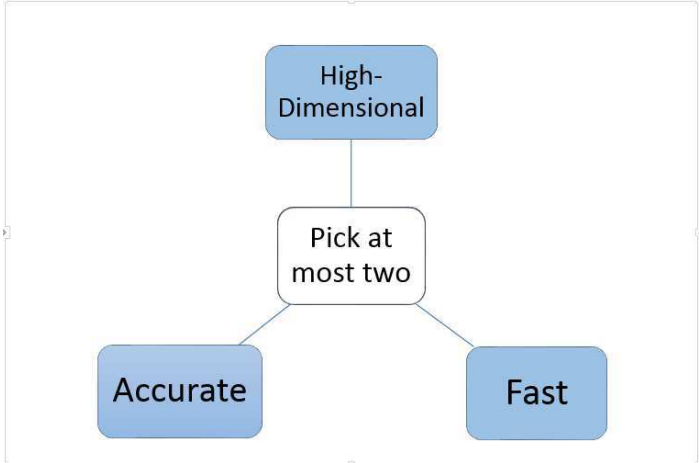
6.1 General Thoughts

Ordinal optimization already trades accuracy in favor of efficiency. I would like to suggest the scalability as a third trade-off. I would like to propose an empirical rule summarized in Figure 6.1. Given a finite computational resource and a specific kind of problem to solve, there seems to be a trade-off of three performances in computational methods, namely 1) accuracy (or precision), 2) efficiency (or convergence rate), and 3) scalability to higher dimensions. In multi-objective optimization language, these three objectives are said to form a Pareto-Front.

For example, in optimization, if an algorithm is very accurate in finding the minimum and does so in a relatively small number of function evaluations, this method is unlikely to be scalable to high-dimensional problems. On the other hand, if a method is able to scale to large dimensions and be efficient in attaining a relaxed objective, accuracy or precision of the objective has been compromised. If one needs to solve a high-dimensional problem with very accurate final results, then one needs to expect a large number of function evaluations. For most problems, the number of dimensions is not a controllable variable. Therefore, the job of an engineer is to employ an *approximation* to get the solution *just right* within a budget.

In our case, SOMBAS traded accuracy of finding the best with the efficiency of finding the feasible or reducing objective in a very limited number of function evaluations in a high-dimensional space (evidence shown up to 100 dimensions).

Figure 6.1 A performance triangle model: given a set of problems to be solved, an algorithm can be considered to possess a combination of three performances (scalability to high-dimensional problems, efficiency in reaching (a target value or a rank of) a solution, and accuracy of the solution) of which two can be improved by sacrificing the remaining performance.



It showed its effectiveness in finding feasible solutions in high-dimensions with a limited number of function evaluations when compared to DE and CMA-ES. It works on the same philosophy as Ordinal Optimization but in a real-number space.

Interaction indices using brute-force calculation showed that by relaxing the need for accurate Monte Carlo integration, one could obtain an index that is efficient in detecting interaction and robust in its quantification, which returns zero to arithmetic precision if the effect of an input variable is additive in the output variance. It is interesting that the time-consuming brute-force approach benefits both the efficiency in screening and the precision in quantitative evaluation of interactions. For screening, it is possible to scale to higher dimensions, whereas for quantitative evaluation, it is yet limiting. This is an example of changing the objective (from Total Sensitivity to Heteroscedasticity) that brings a different level of accuracy and speed for detecting and quantifying interaction, but the trade-off relations of Figure 6.1 still holds.

The adaptive initial step SPSA relaxed or completely eliminated the need for proper tuning of the algorithm to avoid divergence in the limited number of function evaluations. However, this also entails the risk of being stuck at the starting point without any improvement.

In the HAROS-HD report summarized in Appendix B, two hybrid optimization strategies were proposed to improve on all three measures: accuracy, speed, and scalability. At the moment of the reporting, however, it did not show evidence

that it improved on the reference optimization method in terms of minimum objective value found. Since there is still room for improvement by tuning the algorithm parameters, there is a chance that the proposed algorithm improves over the reference methods by reaching a smaller minimum in the same computational budget. If the heuristic is valid, the tuning probably will bring the performance only on par with the reference method at a similar number of function evaluations, but not much further.

However, it is worth investigating if the new hybrid methods have been faster than the reference method in reaching fairly good solutions on par with values in practice. Furthermore, the new method provides a decomposition of the problem to a set of lower dimensional problems as well as identifying *feasible regions* in the form of hypercubes (sets of upper and lower bounds of input parameters). These are added value not obtainable by routine application of traditional optimization algorithm which only finds a single solution. Further investigation and reporting on these aspects need to be done.

Another performance measure *generality* could be added to Figure 6.1. That way, we may be able to improve on the remaining three simultaneously. We can say that most methods pursue any of the other three performances by specializing (compromising generality) to deal with a specific kind of problem or by exploiting specific information pertaining to the problem to be solved. If the generality is to be fixed, the trade-off between accuracy, speed, and scalability seems inevitable.

6.2 Impact

SOMBAS contributes to a new way of performing feasible region search and optimization. The method enables an efficient reduction in objective values in a limited number of function evaluations when the function has a large number of input variables (up to 100 has been investigated). Unlike surrogate model assisted methods, it is not subject to the exponential growth in the number of samples to adequately model the original function. The density learning approach using SOM has lower complexity than surrogate modeling techniques that typically require inverting a matrix to fit an interpolating function. The density representation need not be accurate and learning algorithm of SOM is numerically very robust. The merit function used in SOMBAS enables space filling characteristics in the feasible region and this function could also be used in different optimization algorithms.

The new interaction index enables a robust detection of non-additive (interacting) effects of input variables on the output. The detection is not subject to Monte Carlo integration accuracy enabling the application to weak interaction cases as well as strong ones with as low as $4D + 2$ function evaluations, where D is the number of input variables. If a variable is non-interacting, the proposed computation of its interaction index returns zero to arithmetic precision. The same method

can be used to rank the importance of the interaction of each variable at a larger number of samples, but it is subject to the Monte Carlo integration accuracy.

SPSA is robust against noisy functions and requires only two function evaluations per iteration to estimate gradients irrespective of input dimensions. The proposed adaptive initial step size for SPSA effectively avoids divergence of solutions to larger objective values often encountered when using the method. This reduces the trial error runs to set up an appropriate initial perturbation step size and facilitates its integration with other algorithms.

6.3 Potential Areas of Future Research

SOMBAS and its merit function formulations can be applied inside many other algorithms, and some are described in the appendices. Creating variations of SOMBAS merit investigation. In particular, the fundamental idea of adaptive density learning could be an effective approach to high-dimensional and expensive function optimization. In SOMBAS, vicinity information on the Self-Organizing Map has not been used in its search algorithms. This information could be used to perform a more refined search. The human judgment could also take part in the iterations using the SOM as an interface. Furthermore, the simultaneous improvement of accuracy (finding smaller mass), speed (fewer number of function evaluations), and scalability to high-dimensional problems is yet to be demonstrated in HAROS-HD project.

The interaction index will probably benefit by looking into a more efficient way of computing them (albeit less accurate). Further investigation of its nature and theoretical difference from the other interaction measures are needed. Furthermore, an interaction index that allows correlated inputs would be very useful. The indices could be applied as part of a new kind of optimization method that would learn and exploit interaction and sensitivity information during its iterations. For example, a combination with Monte Carlo Optimization methods such as the Cross-Entropy Method [1] and Probability Collectives [2] would be interesting.

The adaptive initial step reduction approach could be applied to other stochastic approximation methods. There is also potential to further refine the method by also allowing it to adaptively enlarge the step size.

References

[1] D. P. Kroese, S. Porotsky, and R. Y. Rubinstein. *The Cross-Entropy Method for Continuous Multi-Extremal Optimization*. Methodology and Computing in Applied Probability, 8:383 – 407, 2006.

[2] D. Rajnarayan, D. Wolpert, and I. Kroo. *Optimization Under Uncertainty Using Probability Collectives*. In 11th AIAA/ISSMO Multidisciplinary Analysis and Optimisation Conference, Portsmouth, Virginia, 6 - 8 September 2006.

A

SOMBAS in Ensemble Modeling

In this appendix, we show a way of non-linear regression without resorting to the least-square-error. A certain noise level is apriori assumed on the sampled data output, and a set of regression models are drawn to infer an "average" representation and accompanying variance.

⋆ ⋆ ⋆

K. Ito, I. Couckuyt, and T. Dhaene.

"Ensemble Modeling for Minimization of Noisy Expensive Functions: Preliminary Results".

⋆ ⋆ ⋆

This chapter shows preliminary results for fitting multiple surrogates to a noisy function and sampling new points based on the values returned by these surrogates. For simplicity and illustration purposes, we show a case where the function depends on only to one variable but the measurement of the responses are noisy.

A.1 Introduction

There are situations in which one would like to estimate a function underlying a set of noisy observations. There are also times when one needs to find the best input values to a system whose response are not deterministic. Furthermore, the cost of observations/experiment may be expensive and large sample size cannot be obtained. The proposed algorithm addresses these kind of situations. The objective in this situation is not to fit interpolating surrogate models but rather to avoid over-fitting. We assume that we know the responses are noisy (e.g. expected measurement error) but we do not know the underlying function. If we are just trying to find a set of likely parameter values of the underlying function, it is identical to filtering or parameter estimation. If we are trying to minimize the expected value of the responses, we are dealing with stochastic optimization. In both cases, this document addresses them using multiple surrogates and adaptively sampling from them to generate new points. The novelty here is that we do not aim at the least-square error. Instead, we set a threshold error tolerance. Then, an algorithm tries to find surrogate models that achieve prediction errors of the sampled points below this threshold. The diversity of surrogates is expected to automatically cancel out the variance component to avoid overfitting, even though we do not know the complexity (e.g. polynomial order) of the underlying function. We use statistical information of responses to adaptively sample new points for the purpose of narrowing down parameter estimate uncertainties or inferring an optimal point (an input that generates the minimum average of responses).

A.2 Methods

We use Self-Organizing Map Based Adaptive Sampling (SOMBAS) [1] to fit the ensemble of models. Each cell of SOM represents a set of parameters for a surrogate model. Thus, the SOM represents the ensemble of surrogate models. SOMBAS will guide the weight vectors in the SOM to satisfy a certain threshold error level provided by the user.

We conduct two kinds of experiments. In the first one, we fit an ensemble of polynomial functions of order five (six unknown parameters) to model a target function of second order (defined by three parameters). The measurements, however, are contaminated by Gaussian noise of average zero and known standard deviation σ. The polynomial parameters (coefficients) can assume values between -100 and 100. The objective is to estimate the three parameters. The domain of the polynomial is set to $-5 \leq x < 5$.

The second experiment is the estimation of the minimum of the target function. Again the measurements of the responses are contaminated by a Gaussian noise. The new points are sampled one at a time, starting from a random pick of the first

point from the target function domain, $-5 \leq x < 5$, sequentially adding a new point x_k where minimum of the target function is expected, taking into account the uncertainty of the estimate of the minmum through the variance of the ensemble of the surrogate functions (Eq. A.5).

In both of the experiments, the target or the true function is

$$f = 6 - 5x + x^2, \tag{A.1}$$

and the optimal point is $x^* = 2.5$, $f^* = -0.25$. The measurement of the target is given by

$$y = f + s, \tag{A.2}$$

where $s \sim N(0, \sigma)$. The threshold L (i.e. the tolerance of error for the fit surrogates) used in SOMBAS is set as the following.

$$L_{m,\sigma} = m \cdot \sigma^2 \tag{A.3}$$

where m is the number of points to which the surrogate models are fit. The objective to be minimized in the selection phase in SOMBAS is, thus,

$$F_j = \max\left(L_{k,\sigma}, \sum_{i=1}^{k}(y_i - \hat{f}_j)^2\right) - d_k(w_j) \tag{A.4}$$

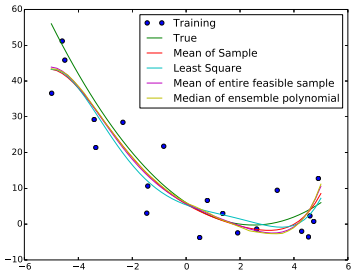
where k is the iteration index, j is the SOM cell index, $\hat{f}_j$ is the response estimated in jth cell in SOM, and $d_k(w_j)$ is the distance to the nearest training weight vector for the jth cell's weight vector. In the first experiment, $k = m$. That is, m training samples (or input-output pairs) are used to calculate the objective values F_j for each cell j. In the second experiment, a new training sample is added sequentially, thus $k = 1, 2, 3, \ldots, m$. Here, m is the total number of iterations one would like to perform (or the number of training samples one would like to add sequentially) to estimate the minimum of the unknown target function f.

A.3 Results

In the first experiment (Figure A.1), twenty points are sampled from the noisy function y and a set of 64 surrogates are fit by running SOMBAS. The noise level is set to $\sigma = 10$. The threshold is set to $L = 20 \times 10^2 = 2000$. Figure A.1a shows the results of the different representations of the ensemble as a single function. The dots are the 20 noisy samples. The True line shows the plot of f. The Mean of Sample line shows the line drawn by mean of final SOM weight vectors from SOMBAS. The Least Square line indicates the conventional least-square error fit. The Mean of entire feasible sample line indicates the line drawn by the mean weight vector computed from all the feasible weight vectors in the history

Figure A.1 Function representations with an ensemble of surrogate models

(a) Polynomials fit to 20 points of data



(b) Ensemble of polynomials fit to the 20 points of data

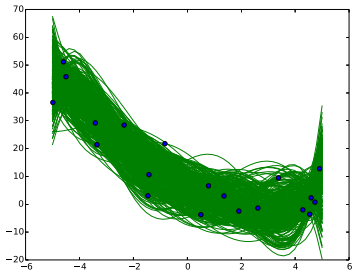


Table A.1 Estimates of coefficients

Coeff. of Term:	constant	1st	2nd	3rd	4th	5th
True	6	-5	1	0	0	0
Best	6.295	-3.744	0.5901	-0.2428	0.01203	0.01047
Mean	5.593	-4.388	0.8827	-0.1294	-0.002774	0.006636
Std.	3.991	2.779	0.8407	0.4024	0.02762	0.01199

of SOMBAS iterations. The Median of ensemble polynomial line shows the curve of the median of the responses from all the feasible weight vectors in the history of SOMBAS. In Figure A.1b, we plot the entire ensemble of curves with feasible errors sampled by SOMBAS.

Table **??** shows the estimates of coefficients of f. These are taken from the final SOM weight vectors. True gives the coefficients of target function f. Best gives the coefficients with the least errors among the SOM weight vectors i.e. the shortest Euclidean distance to the true coefficients. Mean is the average of the SOM weight vectors, and Std. is the standard deviation from the mean of the SOM weight vectors. The true coefficients lie within one standard deviation from the mean.

In the second experiment, we search for the input x giving minimum of f from the noisy measurements y. We do this by sequentially sampling one point at a time. The first point is drawn from a uniform random distribution covering the domain $-5 \leq x < 5$. The subsequent points are determined by searching

$$x = \arg\min \left(E[\hat{f}_j(x)] - \sqrt{V[\hat{f}_j(x)]} \right) \tag{A.5}$$

where $E[\hat{f}_j(x)]$ is the mean of the ensemble of responses at a query point x constructed from the entire history of SOMBAS iterations and $V[\hat{f}_j(x)]$ is the variance

of the ensemble responses to the same query point x. The search of x for equation A.5 is performed by running Differential Evolution (DE). Once the DE finds the solution based on the ensemble model responses, a measurement of the noisy function is made at the point and is used in the training of the ensemble model (using SOMBAS) in the following iteration. This process is repeated for specified number of times. Here we will conduct a sampling of ten points. In Figure A.2, we see how this sequential sampling progressively samples from the domain where the output variance is large and the mean response is low. The true function is shown in red for reference. The algorithm does not have any knowledge of it except that the user has specified a parameter space for a polynomial of order five.

Figure A.3a shows the kernel density distribution of the sampled x. Figure A.3b shows corresponding distribution of x in box plot. Figure A.3c shows the distribution of responses from the ensemble surrogate models at the median of sampled x.

A.4 Conclusion

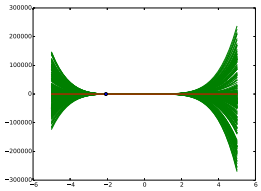
The first experiment of fitting the ensemble to a set of points show that the tolerance satisfaction approach using SOMBAS can get a fairly close regression models as that obtained by the least-square error method. The second experiment took advantage of the fact that there are variations in the ensemble models. The sequential sampling approach seems to both explore the large variance region and exploit the low average response region to successively close-in on the minimum response solution even in the presence of noise in the measurements of outputs.

References

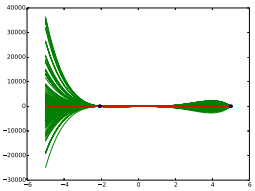
[1] Keiichi Ito, Ivo Couckuyt, Roberto d'Ippolito, and Tom Dhaene. Design space exploration using self-organizing map based adaptive sampling. *Applied Soft Computing*, 43:337 – 346, 2016.

Figure A.2 Sequential search for minimum response sampling from a noisy function. The true function f is indicated in red.

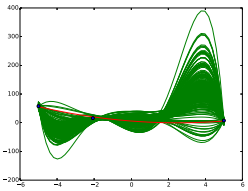
(a) Iteration 1



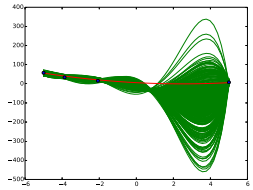
(b) Iteration 2



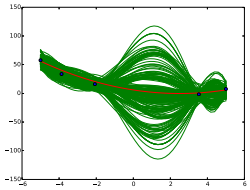
(c) Iteration 3



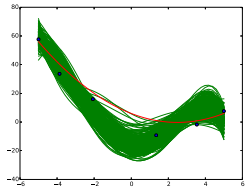
(d) Iteration 4



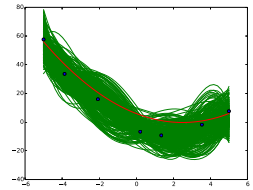
(e) Iteration 5



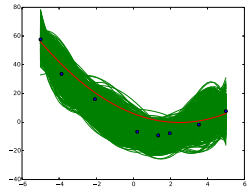
(f) Iteration 6



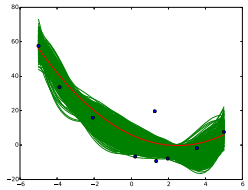
(g) Iteration 7



(h) Iteration 8



(i) Iteration 9



(j) Iteration 10

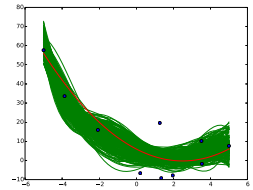
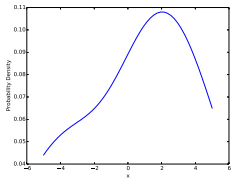
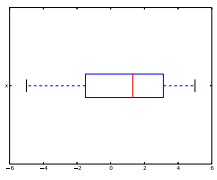


Figure A.3 Estimation of the solution of minimum f after 10 noisy measurements.

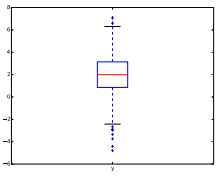
(a) Kernel density estimation of the sampled solution x



(b) Box plot of the sampled solution x



(c) Box plot of ensemble output $\hat{f}_i$ at the median of the sampled solution x.



B

HAROS-HD Project Report Summary

In this appendix, a high-dimensional optimization of discrete input variables is tackled using hybrids of methods. The objective was to create an optimization method that was fast, accurate (in identifying the best solution), and scalable (to a high-dimensional problem). This chapter summarizes the reports submitted to the European Commission.

⋆ ⋆ ⋆

Project management and reporting: Roberto d'Ippolito
Hybrid strategy and workflow: Massimo D'Auria
Graph decomposition: Silvia Poles
SOMBAS and Annealed Hooke and Jeeves: Keiichi Ito

"Hybrid Adaptive Robust Optimization Strategy for EWIS High Dimensional Systems".

European Project, FP7-JTI, project reference: 619198.

⋆ ⋆ ⋆

B.1 Introduction

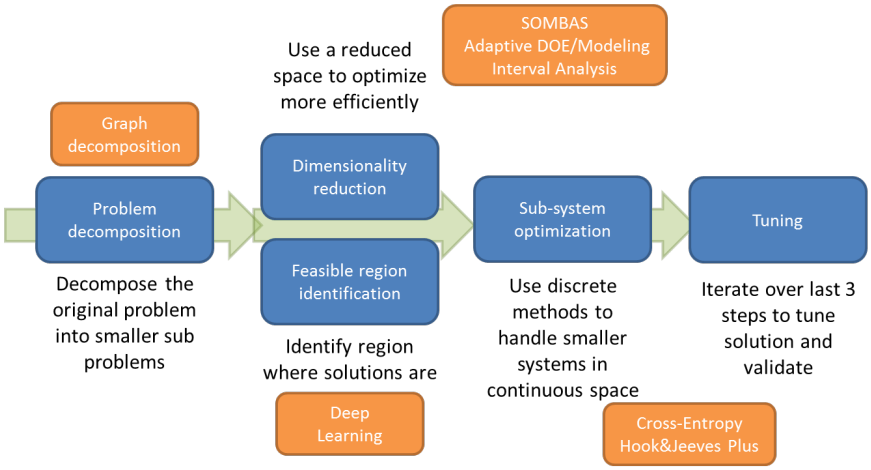
Solving the problem of complexity might be one of the major goals of the 21st century: in the engineering field, this is translated into the high number of parameters to take into consideration, as development projects are getting bigger and often too complex to be grasped entirely by one single person. In particular, complexity in modern aircraft is increasing significantly. They incorporate more electric systems than before since other subsystems that used to be pneumatic or hydraulic are being replaced by electric ones. As a consequence, the wire harnesses that are used to connect those systems to each other must also convey more signals and power.

Electrical systems are used for flight control, sensors, engine control, flight management, communication, in-flight entertainment and many more systems. Connecting the electrical power sources and consumers throughout the aircraft is done by the Electrical Wiring Interconnection System (EWIS). The EWIS is the entire collection of electrical wiring (conductors), connectors, bus bars, shielding, sleeves, pressure seals, brackets, etc. in the aircraft. The entire wiring system is produced in modular components (for manufacturing and production purposes), so-called wire harnesses in which wires are bundled. At the final assembly, those wire harnesses are integrated by attaching all the connectors. For manufacturing and assembly reasons, production breaks are used. The electrical cables linking sources and loads pass through several harnesses. These cables are sized for the current they have to carry, with respect to some thermal and voltage drop constraints. As such, the cable gauge sizing problem is a multi-physics problem. An electrical link between a source and a load can have several cables with different gauges, allowing mass optimization. Thermal and electrical aspects are to be taken into account in the first place, but other types of constraints exist (e.g. connectors' properties, electromagnetic environment created by the current return network, the structure of the aircraft, the fuselage, and the couplings that may occur between all these elements) thus increasing the number of total variables and constraints of the design problem. Given the large amount of design variables and input/output constraints that describe the design space of real gauge sizing problems, the challenge of optimizing such system is considered to be a high-dimensional, non-linear and discrete/continuous problem.

The goal of this project is to link the unique state-of-the-art surrogate modelling technologies available at Noesis to develop new surrogate-based optimization techniques and software solutions suitable to solve large-scale optimization problems. The resulting hybrid, adaptive and robust optimization strategy will allow optimization of high dimensional systems (HAROS-HD, Hybrid Adaptive Robust Optimization Strategy for High Dimensional systems) by means of the smart adoption of model order reduction techniques coupled with surrogate models.

This report summarizes the result of optimizing cable gauges that come in a

Figure B.1 Overview of the general strategy (the orange boxes show candidate algorithms)



discrete set of sizes to minimize the total mass of the cables in EWIS. The discrete variables are ordered. That is, the values that each of the input variables can be set only to predefined values, but they can be sorted from smallest to largest just like standardized thicknesses of metal sheets. Two new hybrid optimization strategies are proposed and their best solutions are compared with reference solutions, which are best solutions found using an in-house code developed by an EWIS company.

B.2 Methods

The HAROS-HD concept originated from engineering design situations in which accuracy of the optimized result is as important as the efficient identification of the "good input space" (also called "feasibility region"). Based on this consideration, the HAROS-HD challenge has been addressed by developing a different approach based on the pre-conditioning of the optimization problem with machine learning algorithms applied to engineering cases. As such, the effort is shifted from the optimization challenge to the "feature discovery" process, where engineering features of the design and solution spaces are "discovered" and exploited to perform a much faster and tailored optimization process. The overall implementation logic is illustrated in Figure B.1. Two prototypes have been implemented out of this general scheme and are described in the next two subsections. Then, the graph decomposition and Annealed Hooke & Jeeves method that are used in the optimization strategy are described.

B.2.1 Optimization Strategy First Prototype

In this scheme (Figure B.2), it is possible to recognize the key steps of the general analysis approach described in Figure B.1. In particular:

1. The problem conversion is the first step being performed. This step converts the current definition of the wire harness problem in the Labinal format into a correctly formulated wire harness optimization problem. This converted problem is fully representative of all the inputs, outputs, objectives, and constraints needed to formulate a complete and consistent optimization problem

2. The second step is performed by using SOMBAS. SOMBAS performs two sub-steps in its formulation:

   (a) A dimensionality reduction: in this sub-step the overall design space is 'projected' into a bidimensional space represented in terms of a self-organizing map (SOM). This SOM is trained to identify specific areas of the SOMBAS objective function.

   (b) A feasible region identification: this sub-step is directly related to the dimensionality reduction. In fact, the identification of the feasible region is performed on the trained SOM and allows the adaptive sampling approach of SOMBAS to focus attention in areas that are more promising in terms of feasibility. This step is repeated a number of times till a certain predefined amount of samples is computed. Normally this amount allows the iteration of the SOMBAS algorithm for about 5-6 times, which has proven to be sufficient for adequate convergence.

3. The third step is performed by using the Annealed Hooke & Jeeves (H&J). H&J is started from the best feasible point that SOMBAS could identify. However, this point may be biased by the vicinity of a local optimal configuration. For this reason, the H&J algorithm has been made robust with respect to local optimal configurations and considers also non-optimal search paths so to identify new possible optimal points.

B.2.2 Optimization Strategy Final Prototype

The final prototype made a significant step further in the complexity of the strategy and has implemented a more sophisticated set of strategies. An overview of the final prototype is here shown in Figure B.3.

In this scheme, it is possible to recognize how the key steps of the first approach described in Figure B.2 have been significantly extended. In particular:

1. The problem conversion is still the first step being performed.

Figure B.2 Overview of the first prototype (the green box indicate the EWIS definition parsing)

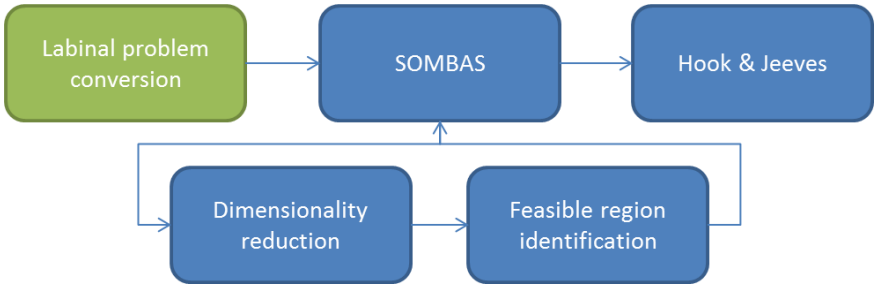
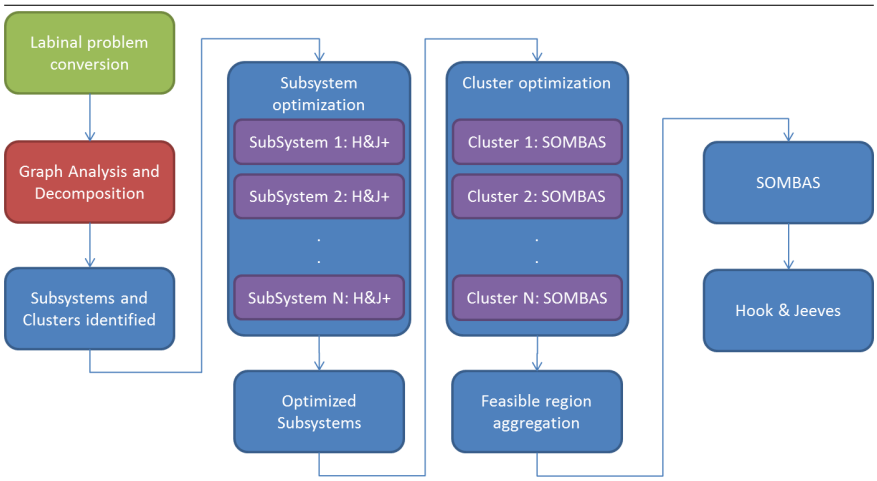


Figure B.3 Overview of the final prototype (H&J+ denotes Annealed Hooke & Jeeves)



2. The second step is the graph analysis and decomposition. In this step, the overall structure of the wire harness problem is analyzed so to identify nodes that have different degrees of importance (or centrality) in the wire harness optimization problem. Details of the model reduction technique based on graph decomposition methods are given in subsection B.2.3. The outcomes of this step are a set of clusters of cables that partition the original problem in sub-problems of different size and importance so that the subsystems and clusters identified can be then optimized and analyzed further. In this case, 2 different sets are identified: subsystems and clusters. Subsystems here are sets of very few cables (max 8 cables) that are not connected to the rest of the wire harnesses. These are essentially cables that can be optimized separately without approximation. Clusters are, on the other hand, sets of many cables that share a similar range of importance in terms of centrality. These are analysed with the feasible region identification approach.

3. The third step optimizes the subsystems. The subsystems are optimized by using short and focused Hook and Jeeves runs because the identification of a feasible region is not a complex problem and can be sorted out directly with H&J.

4. The fourth step is performed by using different SOMBAS processes, one for each cluster. This step is repeated a number of times till a certain predefined amount of samples is computed. Normally this amount allows the iteration of the SOMBAS algorithm for about 5-6 times, which has proven to be sufficient for adequate convergence. Once this phase is completed, there will be a number of feasible regions equal to the number of identified clusters. These feasible regions have been computed on dimensional partitions of the whole system and need to be aggregated to identify the overall feasible region.

5. The fifth step is to aggregate the different feasible regions by running a shorter (in terms of simulations) SOMBAS procedure, but higher in terms of dimensions. Here the overall system is reassembled (except for the subsystems) and the different feasible regions are first intersected and then expanded to match the real global feasible region, taking into account all the interactions between the different clusters that have been neglected in step 4.

6. The sixth step is then to perform the Annealed Hooke & Jeeves (H&J) on the overall feasible region. H&J is started from the best feasible point that SOMBAS could identify on the global feasible region.

The procedure above has been run on 5 test cases. All the use cases have been provided by Labinal with one objective only (Total Mass).

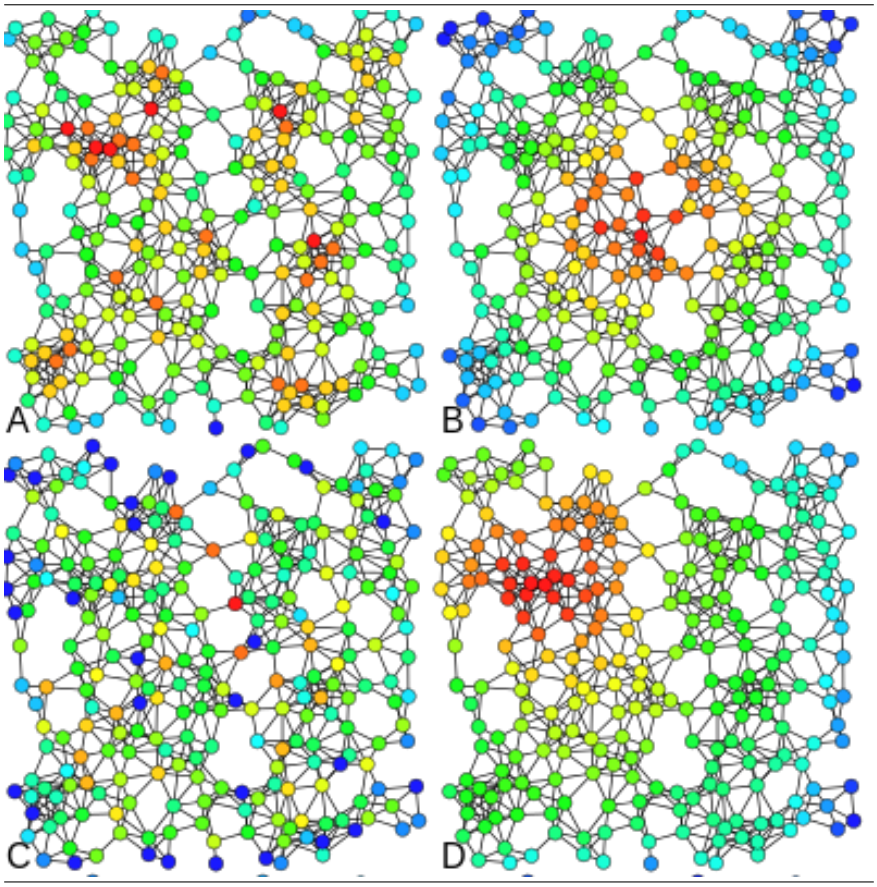
B.2.3 Graph Decomposition

Graph theory is used to model the electrical wire harness networks mapping nodes with cables and edges with thermal and electric constraints. This formulation is used to perform an analysis of the centrality of certain cables with respect to the overall network and to identify the subsets of the overall graph that contribute more to the objectives and the constraints handling. Once these subsets are identified based on their specific centrality index, then a clustering process groups all nodes with the same importance together and identify the subsystems that may be used for graph decomposition in sub graphs loosely coupled together. We have considered 4 centrality indexes. Only 2 of them were eventually used: Degree Centrality and Eigenvector Centrality. Wikipedia gives a brief description of each of the four centrality:

- Degree centrality: degree centrality, which is defined as the number of links incident upon a node (i.e., the number of ties that a node has). The degree can be interpreted in terms of the immediate risk of a node for catching whatever is flowing through the network.
- Closeness centrality: In connected graphs there is a natural distance metric between all pairs of nodes, defined by the length of their shortest paths. The farness of a node x is defined as the sum of its distances from all other nodes, and its closeness was defined by Bavelas as the reciprocal of the farness. Thus, the more central a node is the lower its total distance from all other nodes.
- Betweenness centrality: Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes.
- Eigenvector centrality: Eigenvector centrality is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes. That is, connections to nodes with many edges are more valuable than connections to nodes with few edges.

For the HAROS-HD context, once the nodes' centrality are assessed, a clustering approach is used on the nodes using the centrality measure. The objective is to cluster nodes that are more central and optimize them first. The rationale is that the system can be partitioned in subsystems of decreasing centrality value, thus optimizing first the more central nodes and then the less central ones. For this purpose, the K-Means algorithm is used. The K-Means algorithm is a clustering method that is popular because of its speed and scalability. K-Means is an iterative

Figure B.4 Examples of A) Degree centrality; B) Closeness centrality; C) Betweenness centrality; D) Eigenvector centrality



process of moving the centers of the clusters, or the centroids, to the mean position of their constituent points, and re-assigning instances to their closest clusters. The K is a hyperparameter that specifies the number of clusters that should be created; K-Means automatically assigns observations to clusters but cannot determine the appropriate number of clusters. K must be a positive integer that is less than the number of instances in the training set. Sometimes, the number of clusters is specified by the clustering problem's context. That is, the value of K is derived from the problem's context. In the HAROS-HD case, the number of centroids will be estimated using the elbow method.

The elbow method plots the value of the cost function produced by different values of K. As K increases, the average distortion will decrease; each cluster will have fewer constituent instances, and the instances will be closer to their respective centroids. However, the improvements to the average distortion will decline as K increases. The value of K at which the improvement to the distortion declines the most is called the elbow. The outcome of the graph decomposition procedure will then be a set of K subsystems, each of them sharing similar levels of centrality, that can be optimized separately assuming that the interactions between subsystems are small enough to affect the major part of the optimization problem. This hypothesis will be removed in the tuning phase of the strategy (see Figure B.1) and the system will be re-assembled and a small final optimization loop will be run to correct the neglected interactions between the subsystems.

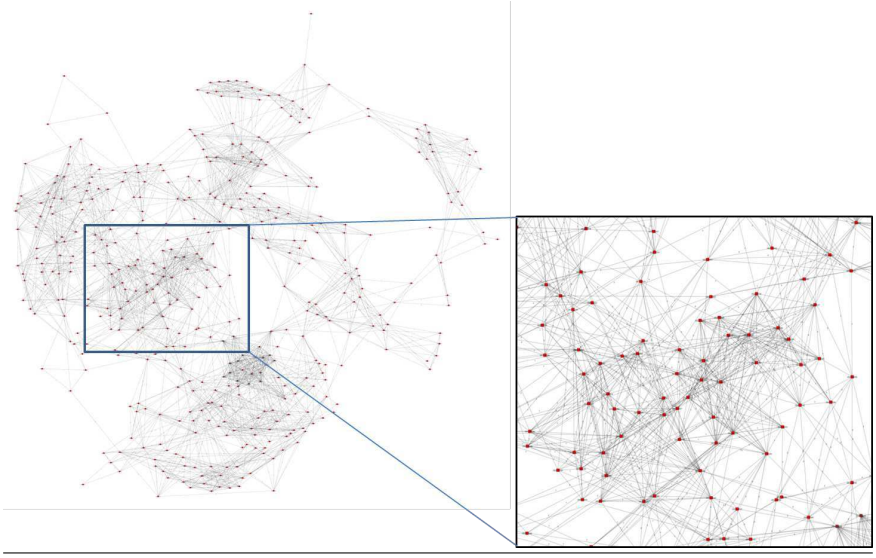
In Figure B.5, the graph generated by the analysis of the 48 harnesses case is rendered, highlighting the different aspects of the graph itself, the connections and the potential clusters based on the different centrality measures.

B.2.4 Annealed Hooke & Jeeves Method for discrete parameters

Hooke & Jeeves method [1] was originally conceived as a direct search method in the real variable input space. Thus, it is a continuous variable optimization method that does not rely on gradient information. In the current implementation, the method is modified to work in a discrete variable input space assuming that the discrete values are ordered such as in integers and ASME standard aluminum plate thicknesses.

Given a starting point, the algorithm will perturb each of the problem variables one at a time (Figure B.6). Since in our case the variables are discrete it simply changes the variable to the neighboring options of the ordered set. The perturbation direction is randomly chosen with a 50% chance to pick a larger value and a 50% chance to pick a smaller value. If the current value of the variable is at the upper or lower bound, it has a 50% chance to move to the adjacent interior value or a 50% chance to move to two steps interior value. In any case, if the perturbation results

Figure B.5 Visual representation of the graph generated from the analysis of the 48 Harness case



in a smaller objective value, that move is accepted and kept at that value and the algorithm moves to the next variable. If the move produces larger objective value, the perturbation is accepted with a certain probability. If rejected, the variable will assume the original value and the algorithm will move to the next variable. Once all the variables are perturbed, a vector move is performed. The vector move is a simultaneous perturbation in the direction accepted in the last perturbation of each variable. Figure B.7 gives the outline of the Annealed Hooke & Jeeves Algorithm.

Unlike the original implementation in which each newly accepted point has to be a strict reduction in the objective value, our implementation allows an increase of the objective value in a Simulated Annealing [2] way. That is, at the beginning of the search such move has a good probability that gets accepted but as the iteration proceeds the probability of accepting worsening move becomes progressively more unlikely.

Figure B.6 One-at-a-Time move of Hooke & Jeeves Method of a three variable problem

Figure B.7 Outline of Annealed Hooke & Jeeves Method

1. Perturb one parameter at a time (pick one of the neighbor options i.e. one size larger or smaller discrete parameter value).
2. Accept new configuration if it is better than the previous one or satisfies Simulated Annealing condition (accept worsening moves under certain probability)
3. Record the accepted perturbation direction for each parameter
4. Once all the parameters are tried out, perturb simultaneously with the recorded perturbation direction vector and accept if better solution is produced
5. Repeat until stopping condition is met

B.3 Results

As it can be noted from Table B.1, the HAROS-HD strategy is performing slightly better than the internal Labinal optimization approach (the reference results) in the 9 and 25 cables test cases (smaller ones) while it is close to the Labinal optimization approach for the other cases, with a slightly increasing difference in total mass (i.e. HAROS-HD is slightly higher in terms of total mass with respect to the internal Labinal methodology). The current results show also that the second version of the prototype is not yet performing better than the first one. This is because the internal tuning parameters of the different algorithms were not yet optimized to the point to exploit completely the synergic effect of the methodologies described in Figure B.3. This set of not yet optimized parameters does not actually depend on the specific problem but rather to the mechanism used by the different algorithms to pass information to each other in a efficient way. In particular, work will need to be done as future plans for:

- Tuning of the SOMBAS parameters for the subsystem feasible region
- Tuning of the SOMBAS parameters for the overall feasible region
- Tuning of the Annealed H&J parameters for the final optimization
- Tuning of the amount of total samples that are assigned to the different phases of the analysis (subsystem feasible region, overall feasible region refinement, final optimization) so that each algorithm can maximize its efficiency The key benefits that the second HAROS prototype has delivered and consolidated can be summarized as:
- The HAROS-HD optimization is fully flexible: the strategy does not change depending on the problem size, but it adapts at runtime.

Table B.1 Results of benchmark (best obtained)

Number of Cables	Number of Constraints	Practice Kg.	Reference Kg.	First Proto. Kg.	Final Proto. Kg.
9	134	4.28	4.01	3.96	3.93
25	924	9.40	7.86	7.78	7.74
80	3725	28.90	23.53	24.47	28.31
153	5871	23.79	18.12	18.90	20.78
441	10084	107.30	80.32	83.76	93.52

- The addition of more constraints is fully supported and no changes in the strategy are needed as long as the conversion properly defines a well-posed optimization problem. As such, the introduction of other kinds of constraints apart from the thermal and voltage drop ones does not require any change in the strategy itself.
- Parallel calculations have been performed, thus speeding up the calculation time with currently 16 parallel processes for the SOMBAS algorithm. The H&J algorithm cannot be parallelized due to its specific formulation.
- Much more information about the system and its subsystems has been gained in the second prototype versus the first. This information can be used for further optimization, constraint, and sensitivity analysis. Moreover, data mining algorithms can also be used as a future extension.

B.4 Conclusion

HAROS-HD optimization strategy has been implemented, deployed and successfully tested, providing promising results in terms of performance with respect to standard optimization algorithms and current design practice. The results so far achieved demonstrate the effectiveness of the methodologies described and their applicability to all the use cases defined for this project. As future work, the margins of improvement that can be achieved relate mainly to:

- Addition of more constraints types (electromagnetic, other)
- Addition of more objectives
  - E.g. cost functions
- Use of adaptive constraints handling to gain more efficiency
- Perform multi-level, multi-disciplinary optimization
  - Within the larger design process

- With sub workflows
- With other disciplines, solvers,...

References

[1] R. Hooke and T. A. Jeeves. *" Direct Search" Solution of Numerical and Statistical Problems*. Journal of the ACM, 8(2):212–229, April 1961. Available from: http://doi.acm.org/10.1145/321062.321069, doi:10.1145/321062.321069.

[2] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. *Optimization by Simulated Annealing*. Science, 220(4598):671 – 680, May 13 1983.

C

User's Guide to SOMBAS

⋆⋆⋆

Keiichi Ito

⋆⋆⋆

This chapter is intended to inform the user of SOMBAS about when to use it as well as an effective initial set up and the meaning of its parameters.

C.1 Objective of SOMBAS

SOMBAS is an acronym of Self-Organizing Map Based Adaptive Sampling method. Given a threshold value in the objective, it tries to find as diverse a set of solution as possible satisfying the threshold as the upper bound of the objective value (*feasible region search*). Otherwise, it is a good method to reduce the objective function value of a high-dimensional input space within a very limited number of function evaluations (*optimization*).

C.2 When to Use

1. Your simulation or objective function is not accurate, thus you want an approximate solution.
2. You are looking for a working set of input parameters for your simulation satisfying some threshold or tolerance.
3. Your budget for the number of function evaluations is tight and the number of input parameters is large.
4. You do not want the best according to the objective function, but you want diverse solutions satisfying the upper bound on the objective value.

C.3 Limitations

SOMBAS was designed to work with real input variables. Ordered dicrete variables (e.g. integer) can be handled by implementing appropriate mappings (e.g. rounding to nearest integers). Objective value must be a scalar (single objective). Constraints other than the truncation value L (described below) can be handled through a penalty function method.

C.4 Performance Envelope

Let D be the number of dimensions in the input space, and $f : R^D \rightarrow R$ (D dimensional real number input, one dimensional real number output) be the objective function. SOMBAS is likely to show advantages compared to other methods in the following situations.

1. $20 \lesssim D \lesssim 100$. It is confirmed to work up to $D \simeq 1000$.
2. In optimization, it is advantageous to use SOMBAS when the maximum number of function evaluations is limited, $M \lesssim 10D$. For feasible region search, larger the maximum number of function evaluations the better.
3. The objective function f does not have to be continuous or smooth. It can contain noise that varies at each function call, i.e. the function does not have to be deterministic.

C.5 Parameter Setup

The following description should not be taken as definitive, but as generally "adequate" setup. The right set up of number of training samples and map size is most

important. Next comes the selectivity temperature and weight constant for diversity in Merit Function. The following four points are problem dependent and need to be set by the user. The other parameters (explained in the next section) can be left at default values for the first run.

- Number of training samples $\lfloor 0.2M \rfloor \lesssim N \lesssim \lfloor 0.7M \rfloor$, where M is maximum number of function evaluations.
- Map size $\lfloor \sqrt{0.1N} \rfloor \lesssim n \lesssim \lfloor \sqrt{2N} \rfloor$, and a good first shot would be $\lfloor \sqrt{N} \rfloor$.
- Maximum number of function evaluations $D \lesssim M \lesssim 10D$ for optimization. If M is larger than this, other methods may be more interesting depending on the problem at hand. If desired, one can perform for example $10D$ evaluations using SOMBAS and after that switch to other optimization methods.
- Set the truncation value L appropriately. It should be the objective function value that the user would be happy to achieve. For example a value considered competitive on the market or a suboptimal but for sure achievable value. If L is not achievable, it is equivalent to performing optimization.

C.6 Description of Parameters

C.6.1 Number of training samples

Self-Organizing Map (SOM) is trained using the training samples. Then SOMBAS decides on the next sampling points based on the trained SOM. The samples are randomly distributed at first, but are replaced by more competitive ones as iteration number increases. Number of training samples is equivalent to population size in evolutionary algorithms.

C.6.2 Truncation Value L

It is the value at which the objective values are truncated. Below this value, no matter how smaller the objective value is, it is always treated equal to the truncation value. Then, SOMBAS tries to find new samples that have larger nearest neighbor distance from sampled points in the history (Summary) than those for the current training samples.

C.6.3 Map size

Size of SOM. SOM in SOMBAS is two dimensional and square shaped. Map size refers to the number of cells in one dimension. The number of cells on the map is therefore square of the map size. In many cases, better performance is

obtained when the number of cells is larger than the number of training samples. For example, if you have a number of training samples as 100, a good map size to try would be anything between 4 and 15 according to the previous section.

C.6.4 Weight constant for diversity in Merit Function ρ

SOMBAS uses a merit function $F = \max(L, \hat{y}) - \rho \min d$, where L is the truncation value, $\hat{y}$ is the output estimate of a query point $\mathbf{w}$, and $\min d$ is the distance of the nearest training sample from the query point. The weight constant for diversityρ controls the importance of the distance to the nearest training sample. Usually, it is not a critical parameter and can be left at the default value of 2.

C.6.5 Selectivity Temperature T

This is the same parameter as the temperature in Simulated Annealing (SA). All the SOM weight vectors are assigned a merit value according to the Merit Function F. Setting the selectivity temperature T low, will only select a small number of weight vectors of SOM as the candidate for the new training samples. In general, $1 \lesssim T \lesssim 4$ for optimization and $0.05 \lesssim T \lesssim 1$ for feasible region search, but more experiments are needed to confirm this.

C.6.6 Probability of mutation

This is the probability that a weight vector selected as a new training sample candidate gets perturbed. So a probability 0.5 means that among all the selected candidates, half of them is perturbed by a random vector. It is also possibble to apply the probability by dimensions. For example a probability of 0.5 in this case means that about 50% of the candidate's input variables gets perturbed. As a default, 1 works fine in most cases.

C.6.7 Expansion Factor F_e and Contraction Factor F_c

These two parameters control the magnitude of mutation. They are factors that are multiplied to the covariance matrix of the mutation vector generated from Gaussian multivariate distribution. Expansion factor is applied when in the previous iteration a new minimum is found. Otherwise the contraction factor is applied. Typical values for F_e are $1.1 \sim 2.0$ and for F_c are $0.5 \sim 1.0$. In many cases, $F_e = 2.0$ and $F_c = 0.5$ work fine.

C.6.8 Number of SOM training iterations

SOM's weight vectors need to be trained before subset of them get selected as candidates for new training samples. This number specifies how many iterations

to perform on the SOM training. The default value 40 is usually a good number.